305-727
Issue 1

AT&T 3B2 Computer
UNIX® System V Release 3
System Administrator's Guide

NOTICE

The information in this document is subject to change without notice. AT&T assumes no responsibility for any errors that may appear in this document.

Copyright© 1989 AT&T
All Rights Reserved
Printed in U.S.A.

TRADEMARKS

The following is a listing of the trademarks that are used in this manual:

- DOCUMENTER'S WORKBENCH — Trademark of AT&T
- FREON — Registered trademark of E. I. DuPont de Nemours & Co., Inc.
- Hayes — Registered trademark of Hayes Microcomputer Products, Inc
- Micom — Registered Trademark of Micom Systems, Inc
- Penril — Registered Trademark of Penril Corporation, Inc
- Rixon — Registered Trademark of Rixon, Inc
- SCOTCH — Registered trademark of 3M
- TELETYPE — Registered trademark of AT&T
- UNIX — Registered trademark of AT&T
- Ventel — Registered trademark of Ven-Tel, Inc
- WE — Registered trademark of AT&T
- WREN — Trademark of Control Data Corporation

ORDERING INFORMATION

Additional copies of this document can be ordered by calling

Toll free:	1-800-432-6600	In the U.S.A.
	1-800-255-1242	In Canada
Toll:	1-317-352-8557	Worldwide

OR by writing to:

AT&T Customer Information Center
Attn: Customer Service Representative
P.O. Box 19901
Indianapolis, IN 46219

TRAINING INFORMATION

The *AT&T Education and Training Catalogue of Courses* and course schedules are free and can be ordered by contacting your account executive or by calling toll free:

1-800-554-6400

Training information can also be accessed electronically through our computerized catalogue:

DIAL: 1-800-662-0662 or 1-614-764-5566
LOGIN: comcats
PASSWORD: 88cat

Table of Contents

Part 2: Support

Appendices, Glossary, Index

List of Figures

Introduction

This guide describes procedures you use in the administration of an AT&T 3B2 computer (Version 3) running the UNIX* System V Release 3 operating system. It is designed to accomplish the following objectives:

- Instruct you how to do the administrative tasks of a UNIX System V operating system
- Give you background information about when and why these tasks are desirable
- Give you a quick reference to administrative procedures.

Who Should Use This Guide?

If you are the operator/administrator of your 3B2 computer, you should use this guide. Be sure that you are using the appropriate guide for the system you have. This manual is for Version 3 computers only. To determine the version of your computer, enter the following command:

```
uname -v
```

If your system is a version other than Version 3, you should use the *User's and System Administrator's Guide* that came with your computer.

What Does a System Administrator Do?

The System Administrator has two main tasks:

- Determining what rules are needed to govern the use of the computer system
- Implementing those rules to give the maximum amount of computing service for the system users, consistent with the physical limitations of the machine.

---

* Registered trademark of AT&T

Obviously, if you are the only user of your 3B2 computer, these tasks consist simply of those things you do to keep the machine running and your programs and data from disappearing permanently. If, on the other hand, your 3B2 computer will be used by several other people, the tasks become more complex, and you are required to be aware of the needs of your whole user community.

Some Assumptions About Your Experience

It is assumed that you know the mechanics of using a computer terminal to enter commands, and that you have an awareness of such UNIX System V fundamentals as the directory structure and the shell. It is also assumed that you feel comfortable using the 3B2 computer; you know how to turn it on and how to use such things as the floppy disk drive. If you feel unsure of your knowledge on these points, you might find it helpful to refer to the *Owner/Operator Manual* that came with your system.

How This Guide Is Organized

There are two main parts to this guide:

Part 1—Procedures

Part 1 is ten sets of step-by-step procedures that tell you how to keep your 3B2 computer in operation. Each set of tasks is related to a general topic, such as User Services or Processor Operations.

Part 2—Support Information

Part 2 is ten chapters of more detailed information about each of the ten sets of procedures. The chapters are numbered to parallel the procedures of Part 1.

In the back of the book there are four appendices, a glossary, and an index.

How to Use This Guide

1. **To begin with, use the procedures in Part 1.**

   These procedures lead you through administrative tasks without requiring preliminary knowledge or experience in that area.

2. **Use the chapters in Part 2 to learn more about what the procedures do.**

   They explain what is going on in the procedures and are background information about the basic elements of the UNIX System V operating system.

3. **Use the Index.**

   The index is a cross-reference so you can uncover all the places in the guide where a topic is discussed.

4. **Use the Glossary to look up definitions of terms that are unfamiliar.**

5. **As you gain experience, use the guide for reference.**

About the Procedures

A table at the beginning of each procedure gives you information in capsule form. Table entries appear only when the information is relevant. The tables follow the following style:

Purpose	Summary for what the procedure is used.
When Performed	When you should schedule the procedure.
Starting Conditions	The state the computer should be in when you begin the procedure. Any special login requirements.
sysadm Menu	The part of the System Administration Menu package that contains the subcommands to do the procedure.
Commands	The commands used to do the procedure.
Firmware Programs	The names of programs to be run in firmware mode.
Bootable Programs	The names of programs used to boot the system.
Media	Floppy disks or tapes used in the procedure.
Time	About how long the procedure takes.
Cautions	Special instructions you must follow before or during the procedure to ensure the integrity of your system software and user files.
References	The chapter and section in Part 2 where the topic is more fully discussed. Other UNIX System V manuals where additional information can be found.

System Administration Commands

Most of the procedures are based on menus of the System Administration Menu package. This package consists of a hierarchical arrangement of interactive screens that lead you through system administration tasks. It is described in the *Owner/Operator Manual* that came with your system.

The procedures shown in this guide bypass the higher level System Administration menus and take you directly to the subcommands. Subcommands are the equivalent of menu selections from lower level menus. If you prefer, you may start at the main menu with the basic command:

```
# sysadm
```

You can get to the submenu level with a command like the following:

```
# sysadm filemgmt
```

System States

In some procedures, we say that a particular system state is required. Usually this means that the system must be in either the single- or multiuser state. The single-user state corresponds to run level 1, while the multiuser state corresponds to run levels 2 or 3. Procedures for bringing the system to different system states are found in "Processor Operations Procedures in Part 1." See the section on "Levels of Operation" in Chapter 3 of Part 2 for more information on system states.

Logins

In some procedures, we state that a particular login is required. This frequently means that you must be logged in as **root** to do the procedure. The phrase "an authorized login" is also used. The authorized login is typically a special administrative or system login name required for a normal user to do the procedure (see Chapter 1 of Part 2 for a list of these logins).

Passwords

It is strongly recommended that you set up and use passwords for administrative and system logins (see Procedure 1.5 in "System Identification and Security Procedures" of Part 1 for information on how to do this). In the procedures, it is assumed that such password protection has been established. When you enter a **sysadm** command as an ordinary user, you are prompted for a password. This entry is shown as follows:

```
$ sysadm backup
Password:
```

At this point, to go ahead with the procedure, you are required to enter a password. As always the case in the UNIX System V, the password is not echoed (displayed) on your screen.

In procedures that require you to be logged in as **root** (that is, the super user), you are not prompted for the **sysadm** password. Also, the pound sign (#) prompt is used for the **root** login. The following is an example:

```
# sysadm backup
```

Notation Conventions

Whenever the text includes examples of output from the computer and/or commands entered by you, the standard notation scheme that is common throughout UNIX System V documentation is as follows:

- Text that you type in from your terminal is shown in **bold** type.
- Text that is printed on your terminal by the computer is shown in `constant width` type.
- Comments and explanations within a display are shown in *italic* type and are indented to separate them from the text that represents computer output or input.

  Italics are also used to show substitutable values, such as *file*, when the format of a command is shown.
- There is an implied **RETURN** at the end of each command and menu response you enter.
- Where you may be expected to enter only a **RETURN** (as in the case where you are accepting a menu default), the symbol <**CR**> is used.
- The dollar sign (**$**) and pound sign (**#**) symbols are the standard default prompt signs for an ordinary user and **root**, respectively.

  $ – means you are logged in as an ordinary user
  # – means you are logged in as **root**.
- When the **#** prompt is used in an example, it means the command illustrated may be used only by **root**.
- When the full path name of a command is shown in an example (like **/etc/fsck**), the command must be entered that way.

Command References

When commands are mentioned in a section of the text for the first time, a reference to the manual section where the command is described is included in parentheses: **command**(section). The numbered sections are located in the following manuals:

Sections (1), (6)	*User's Reference Manual*
Sections (1), (1M), (4), (7), (8)	*User's and System Administrator's Reference Manual*
Sections (1), (2), (3), (4), (5)	*Programmer's Reference Manual*

Information in the Examples

While every effort has been made to present displays of information just as they appear on your terminal, it is possible that your system will produce slightly different output. Some displays depend on a particular machine configuration that may differ from yours. Changes between releases of the UNIX system software may cause small differences in what appears on your terminal. All the displays in this guide were taken from a dual Small Computer System Interface (SCSI) disk system and may differ from the single SCSI disk system.

Part 1: Procedures

System Identification and Security Procedures

System Identification and Security Procedures

The following procedures are covered in this section:

Procedure 1.1	**Check Console Terminal Configuration** To assure that the system console terminal is properly configured.
Procedure 1.2	**Activate/Deactivate Console Logger** To turn the system console logger on or off.
Procedure 1.3	**Set Time and Date** To set the time and date of the internal system clock.
Procedure 1.4	**Establish or Change System and Node Names** To define the formal system name, especially for the computer to be a node in a network.
Procedure 1.5	**Assign Passwords to Administrative and System Logins** To assign passwords to administrative and system logins.
Procedure 1.6	**Forgotten Root Password Recovery** To recover from forgetting or from corrupting the root password.
Procedure 1.7	**Forgotten Firmware Password Recovery** To recover from forgetting or from corrupting the firmware password.
Procedure 1.8	**Enable/Disable Shadow Password** To convert/unconvert to/from shadow password operation.
Procedure 1.9	**Display Password Information** To display password status and aging information for all logins or for a selected login.
Procedure 1.10	**Set Password Aging Information** To set password aging information for a login.
Procedure 1.11	**Lock/Unlock a Login** To lock/unlock a login so it cannot/can be used.
Procedure 1.12	**Enable/Disable Unsuccessful Login Logging** To enable/disable the tracking of repetitive unsuccessful attempts to log in on a port.

Procedure 1.1: Check Console Terminal Configuration

Purpose	To assure that communication with the system is maintained.
References	The *Owner/Operator Manual* that came with your system. The *Operator's Guide* for your console terminal.

All system administration functions are performed at the console terminal, which is initially set at a 9600 baud rate. The console port may operate at another data rate. To change the console baud rate, use the **stty**(1) command. This will write the value for the console terminal in the Nonvolatile Random Access Memory (NVRAM). The **gettydefs**(4) command uses the value stored in NVRAM when the system is booted. Therefore, the console baud rate is the same when operating in the UNIX system and in the firmware mode.

There are many combinations of baud rates for the "CONSOLE" and "CONTTY" ports. Some of the combinations are not supported; therefore, the proven baud rate combinations of 1200, 4800, and 9600 are suggested.

Here are some things you might want to do to make sure your console terminal is configured properly. Use the *Operator's Guide* for your terminal to learn how to make these equipment checks.

- Verify that the Input/Output (I/O) terminal speed option is equal to the console tty setting (console tty is set to 9600 baud at the factory, but may be changed).
- Set the interface to 8-bit American Standard Code for Information Interchange (ASCII), full duplex, with a parity of "none" or "space," depending on the terminal you have.
- If you lose communication with the system, check to see if the terminal is still plugged in.
- You should not lose communication with the system when going to the firmware mode because the input/output terminal speed option is set to the same baud rate as the NVRAMs. Refer to the "Clearing Trouble" section in the *Owner/Operator Manual* that came with your system.

A printer should be part of the console equipment configuration because it gives a record of exactly what was done and how the system responded. It is the system log, and it is especially helpful when you run diagnostics. If your console terminal has this capability, the best method is to hook the printer directly to the system console. (If you have an AT&T 4425 terminal, there is an auxiliary port on the rear of the terminal. Other manufacturers' brands offer the same feature.) The console printer should be independent of the Line Printer Spooling system.

Procedure 1.2: Activate/Deactivate Console Logger

Purpose	To activate/deactivate the console logger.
Starting Conditions	System state—multiuser or single user. Login—**root**.
Commands	**conslog**(1M) **[-a ¦ -d ¦ -r]**
Cautions	Deactivate the console logger before executing commands that put the console in raw mode (for example, **pg, vi**), or commands that are required to be attached to a tty device when the console logger is activated. Do not **cat** the conlog file from the console when the logger is active. Use the **conslog -r** command to read or look at the file when you are at the console terminal.
References	Chapter 1, "System Identification and Security."

The console logger feature will store a "soft copy" of all the console Input/Output (I/O) in a disk resident file. The **conslog -a** command allows the system administrator to activate this feature, **conslog -d** deactivates the logger, and **conslog -r** displays the contents of the conlog file. The **conslog -r** command should only be executed at the console terminal.

When the UNIX operating system is booted, the console logger is not running. If the console terminal is available for any user, it is suggested that the console logger be running. However, your local policies or procedures may dictate the use of this feature.

The conlog file is located in **/usr/adm**; this file will grow until the UNIX system file size limit is reached. At this point an error message will be displayed. The conlog file needs to be moved (cleared out) periodically. The following procedure will move (clear) the conlog file.

Step 1: Deactivate the console logger by using the **conslog -d** command.

Step 2: Reactivate the console logger; the following display shows this command and system response. By reactivating the console logger, it will automatically move the old log into a time-dated file.

```
# conslog -a
                        ****  NOTICE  ****
/usr/adm/conlog is being saved in /usr/adm/conlog12080935.
/usr/adm/conlog12080935 should be removed if no longer needed.
```

Step 3: If you do not need to save the log file, remove it.

Procedure 1.3: Set Time and Date

Purpose	To synchronize system time with clock time or to reset the system time after it has been corrupted.
Starting Conditions	System state —multiuser, for synchronizing system time with current clock time —single user, for setting the date ahead. Login—**root**, to reset clock with **date**(1).
sysadm Menu	SYSTEM SETUP
Commands	**sysadm datetime**(1) **date**(1)—requires logging in as **root**
Caution	Go to single-user mode (Procedure 3.3) if you are setting the date ahead.

Setting the date ahead by one or more days should be done in the single-user mode. Setting the date ahead while in the multiuser mode with **cron** running should be avoided. The **cron** program will try to "catch-up" for the interval involved. All the processes that were scheduled to run in the interval are started by **cron**.

When the UNIX operating system is booted, you may receive a reminder to set the time-of-day clock. This should happen infrequently. Setting the clock, however, is normally required when you do the following:

- First use the system
- Reset NVRAM with the floppy key
- Run the time-of-day clock interactive diagnostic test.

Step 1: To set the time and the date use the System Administration Menu **datetime**. For example:

```
# sysadm datetime

Running Subcommand 'datetime' from menu 'syssetup',
SYSTEM SETUP


Current time and time zone is: 04:59 EDT
Change the time zone? [y, n, q, ?] n
Current date and time:  Tue. 03/29/88 05:00
Change the date and time: [y, n, q, ?] y
Month   default 03      (1-12): <CR>
                (Using <CR> to accept the default)
Day     default 29      (1-31): <CR>
Year    default 88      (70-99): <CR>
Hour    default 05      (0-23): <CR>
Minute  default 00      (0-59): 04
Date and time will be set to:  03/29/88 05:04. OK? [y, n, q] y
The date and time are now changed.
cron aborted: SIGTERM
The cron has been restarted to pick up the new time and/or time zone.
```

Step 2: The clock also can be set using the **date** command, without using the **sysadm** menus. You must be logged in as **root** to use **date** to change the date and time, and it is suggested that the system be in single-user mode. The arguments to the **date** command are in the sequence of month, day, hour, minute, and year.

```
# date 0329050488
Tue Mar 29 05:04:00 EST 1988
```

Procedure 1.4: Establish or Change System and Node Names

Purpose	To define the system and node names: —for a new 3B2 computer —after using the floppy key —when reconfiguring the system to include basic networking.
Starting Conditions	System state—multiuser or single user. To use **sysadm** in single-user mode, you must mount **/usr**. Login—authorized user or **root**.
sysadm Menu	SYSTEM SETUP
Commands	**sysadm nodename**(1) **uname**(1) **-S** —need **root** login
Caution	Do not change system and node names arbitrarily. Changes must be coordinated with your networking connections.
References	Chapter 9, "Basic Networking" of Part 2.

The system and node names of the 3B2 computer can be set by any of the following means.

- Using the **sysadm nodename** command
- Using the **uname** command
- Using the floppy key resets NVRAM, which causes the system and node names to be reset to the default values. See the *Owner/Operator Manual* that came with your system for more details about making the floppy key.
- Reconfiguring the operating system after changing the names of SYS and NODE tunable parameters. See Chapter 6, "Performance Management" in Part 2, for more details.

Choose one of the following methods for establishing or changing the system and node name.

System Administration Menu—nodename

The following command line entries and system responses show the setting of the node name using the **sysadm nodename** command. The node name is then output using the **uname** command. The contents of the **/etc/rc.d/nodename** file is the result of the execution of **sysadm nodename**.

```
# sysadm nodename
Running subcommand 'nodename' from menu 'syssetup',
SYSTEM SETUP

This machine is currently called "unix".
Do you want to change it? [y, n, ?, q] y
What name do you want to give it? [q] ABcd5678
# uname -a
ABcd5678 ABcd5678 3.2.2 3 3B2
#
```

Command—uname

Using **uname** to change the system and node names is not as permanent as using **sysadm nodename**. Whenever the system is rebooted, the system and node names assigned to the system are those last entered through **sysadm nodename** and residing in the file **/etc/rc.d/nodename**. This file is not set up or changed by **uname**. Thus, it is a good practice to use the command **sysadm nodename** to change the system and node names. The following shows how to display and change the system and node names by using **uname**.

```
# uname -a
unix unix 3.2.2 3 3B2
# uname -S abcdefghijk
uname: name must be <= 8 letters
# uname -S ABcd5678
# uname -a
ABcd5678 ABcd5678 3.2.2 3 3B2
#
```

Procedure 1.5: Assign Passwords to Administrative and System Logins

Purpose	To permit controlled access to various administrative and special system functions.
Starting Conditions	Multiuser or single-user state, any login. You must mount **/usr** to run this procedure in single-user mode.
sysadm Menu	SYSTEM SETUP
Commands	**sysadm admpasswd**(1) **sysadm syspasswd**(1) **passwd**(1)
References	Chapter 1, "System Identification and Security."

There are special system logins and administrative commands that can be assigned passwords. The administrative logins are: **setup**, **powerdown**, **sysadm**, **checkfsys**, **makefsys**, **mountfsys**, and **umountfsys**. The system logins are: **root**, **sys**, **adm**, **bin**, **uucp**, **nuucp**, **lp**, **daemon**, and **trouble**. Also, if AT&T Framed Access Command Environment (FACE) Utilities are installed there are two additional logins that require passwords: **vmsys** and **oasys**. See Chapter 1, "System Identification and Security," for more information on these logins.

After loading the operating system, you can only log in as **root** or **setup** because all other administrative and system logins are "locked." After loading the AT&T FACE Utilities, you need to assign passwords to the **vmsys** and **oasys** logins which initially have no passwords. To assign passwords to the administrative logins, log in as **root** and use the **sysadm admpasswd** command. To assign passwords to the system logins and to any other logins, log in as **root** and use the **passwd** command. Note that the **sysadm syspasswd** command can be used to assign passwords to the system logins only if no passwords are assigned, and the logins are unlocked.

Since the system is delivered with only the **root** system login accessible, executing the **sysadm syspasswd** command typically results in the following screen display.

```
#  sysadm syspasswd

Running subcommand 'syspasswd' from menu 'syssetup',
SYSTEM SETUP


        All the system logins on this machine have passwords.
To change the password of one of those logins, you must either login as
that ID, which means you must know the current password, or as "root",
which means you must know the root password.  Use the passwd command to
change the password.

#
```

Step 1: Assign a password to the **setup** login using the **sysadm admpasswd** command. The other administrative logins are "skipped" in this example, which leaves these logins "locked."

Procedure 1.5: Assign Passwords to Administrative and System Logins—

```
# sysadm admpasswd

Running subcommand 'admpasswd' from menu 'syssetup',
SYSTEM SETUP

Do you want to give passwords to administrative logins? [y, n, ?, q] y

The login 'setup' does not have a password.
Do you want to give it one? [y, n, ?, q] y
New password:
Re-enter new password:

The login 'powerdown' already has a password.
Do you want to change the password, delete it, or skip it? [c, d, s, q, ?] s
Password unchanged.

The login 'sysadm' already has a password.
Do you want to change the password, delete it, or skip it? [c, d, s, q, ?] s
Password unchanged.

The login 'checkfsys' already has a password.
Do you want to change the password, delete it, or skip it? [c, d, s, q, ?] s
Password unchanged.

The login 'makefsys' already has a password.
Do you want to change the password, delete it, or skip it? [c, d, s, q, ?] s
Password unchanged.

The login 'mountfsys' already has a password.
Do you want to change the password, delete it, or skip it? [c, d, s, q, ?] s
Password unchanged.

The login 'umountfsys' already has a password.
Do you want to change the password, delete it, or skip it? [c, d, s, q, ?] s
Password unchanged.

For more information about passwords and their use,
read the SECURITY chapter of the Owner/Operator Manual.
For more about assigning passwords,
see the chapter on SIMPLIFIED SYSTEM ADMINISTRATION.
```

Step 2: Assign passwords to the system logins and to any other logins, log in as **root** and use the **passwd** command. The following screen display shows how to assign passwords to **root**, **vmsys** and **oasys** logins.

```
# passwd
passwd:  Changing password for root
New password:
Re-enter new password:
# passwd vmsys
New password:
Re-enter new password:
# passwd oasys
New password:
Re-enter new password:
#
```

Step 3: To check the status of all passwords enter:

```
# passwd -sa
root  PS
daemon  LK
bin  LK
sys  LK
adm  LK
uucp  LK
nuucp  LK
trouble  LK
lp  LK
setup  PS
powerdown  LK
sysadm  LK
checkfsys  LK
makefsys  LK
mountfsys  LK
umountfsys  LK
vmsys  PS  03/28/88  7  63
oasys  PS  03/28/88  7  63
#
```

The password status information shows the all logins either have passwords assigned (PS) or are locked (LK). (Logins with no passwords are identified by an NP.)

Procedure 1.6: Forgotten Root Password Recovery

Caution: Restoring the original operating system is a drastic way of recovering from a forgotten root password. Make every effort to remember or discover the root password before performing this procedure.

Purpose	To recover from a forgotten **root** password by restoring the original operating system (before the password was assigned).
Starting Conditions	Multiuser state, logged in as an authorized user. Power off, not logged in.
sysadm Menu	MACHINE MANAGEMENT FILE MANAGEMENT
Commands	**sysadm backup**(1) **sysadm firmware**(1)
Bootable Program	The **/unix**—boot program [from a Small Computer Systems Interface (SCSI) Cartridge Tape].
Media	The Operating System Utilities cartridge tape.

There are two methods you can use to recover from a forgotten **root** password. Both these methods assume that you cannot restore the **/etc/passwd** or **/etc/shadow** file from another login. The first method, **sysadm** method, requires you to know the password that is assigned to the **sysadm** command. The second method, partial restore method, is much more time consuming and may result in the loss of some data.

sysadm Method of Recovering the Root Password

Step 1: Inform any users that are logged on to log off.

Step 2: Enter the command:

sysadm backup

Step 3: Do a complete backup of the **usr** and any other file systems needed (see Procedure 5.4, "File System Backup and Restore").

Step 4: Go to the firmware mode. Enter:

sysadm firmware

A series of messages are displayed ending with the following:

```
FIRMWARE MODE
```

Step 5: Enter the firmware password (**mcp** is the default password).

Step 6: In response to the prompt:

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

Enter **/unix**, selecting the Load Device Option 1 (SCSI) and selecting the Load Device Option subdevice 1 (tape). Reload the UNIX operating system from the Operating System Utilities cartridge tape, selecting the full restore option (refer to Procedure 3.9, "Reload the Operating System").

Step 7: When the full restore is done, log in as **root** and reload the other utilities that were installed on your system. Refer to the *Owner/Operator Manual* that came with your system.

Step 8: After you have the other utilities reinstalled, you will need to restore the **usr** and other file systems from backups, as required (refer to Procedure 5.4, "File System Backup and Restore").

Step 9: Store the backup and Operating System Utilities cartridge tape in a safe place.

Partial Restore Method of Recovering the Root Password

Step 1: If the system is off, go ahead to Step 2; otherwise, press the power switch to the standby position.

Step 2: After the hard disk has stopped spinning, press the power switch to the ON position. When the word `DIAGNOSTICS` is displayed, press the reset switch (described in the *Read Me First* manual). This will put the system in firmware mode.

Step 3: Enter the firmware password (**mcp** is the default password).

Step 4: In response to the prompt:

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

Enter **/unix**, selecting the Load Device Option 1 (SCSI) and selecting the Load Device Option subdevice 1 (tape). Reload the UNIX operating system from the Operating System Utilities cartridge tapes, selecting the partial restore option (refer to Procedure 3.9, "Reload the Operating System").

Step 5: When you get the `Console Login` prompt, log in as root and enter the command:

sysadm backup

Step 6: Do a complete backup of the **usr** and any other needed file systems (see Procedure 5.4, "File System Backup and Restore").

Step 7: Go to the firmware mode. Enter:

sysadm firmware

Observe a series of messages that end with:

```
FIRMWARE MODE
```

Step 8: Enter the firmware password (**mcp** is the default password).

Step 9: In response to the prompt:

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

Enter **/unix**, selecting the Load Device Option 1 (SCSI) and selecting the Load Device Option subdevice 1 (tape). Reload the UNIX Operating System from the Operating System Utilities cartridge tape, selecting the full restore option (refer to Procedure 3.9, "Reload the Operating System").

Step 10: When the full restore is done, log in as **root** and reload the other utilities that were installed on your system. Refer to the *Owner/Operator Manual* that came with your system.

Step 11: After you have the other utilities reinstalled, you will need to restore the **usr** and other file system from backups, as required (refer to Procedure 5.4, "File System Backup and Restore").

Step 12: Store the backup and Operating System Utilities cartridge tapes in a safe place.

Procedure 1.7: Forgotten Firmware Password Recovery

Purpose	To restore the default firmware password.
Starting Conditions	Any system state (except firmware mode), any login.
Media	The floppy key floppy disk.

If you have forgotten your firmware password, the following procedure can be done by anyone with access to the machine and the floppy key floppy disk. Refer to the *Owner/Operator Manual* that came with your system for more information about the floppy key. Knowledge of any logins or passwords is not required.

Step 1: Turn off the 3B2 computer by using the power switch.

Step 2: After the power has been removed, insert the floppy key into the integral floppy disk drive.

Step 3: Press the power switch to the ON position.

Step 4: The floppy key can be removed from the drive after the `FW WARNING` message is output. Store the floppy key in a safe place.

The following display represents a typical output when the system is powered up using the floppy key.

```
SELF-CHECK

FW WARNING:  NVRAM DEFAULT VALUES ASSUMED

DIAGNOSTICS  PASSED

UNIX(R) System V Release 3.2.2 AT&T 3B2 Version 3
Node unix
Total real memory  = 8388608
Available memory   = 5351424
********************************************************************************

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.

(File systems are checked before mounting.)

mount -f S5 /dev/dsk/c1t1d0s8  /usr2
mount -f S5 /dev/dsk/c1t1d1s2  /usr
Generating a new /unix
AT&T 3B2 SYSTEM CONFIGURATION:

Memory size: 8 Megabytes
System Peripherals:
        Device Name         Subdevices       Extended Subdevices

        SBD
                          Floppy Disk
        SCSI
        (S.E. Bus ID0)
                          SD01 ID1
                                           317 Megabyte Disk ID0
                                           317 Megabyte Disk ID1
                          ST01 ID2
                                           Tape ID0
        EPORTS
        EPORTS
        EPORTS
        MAU
The system is ready.

Console Login:
```

Note: When NVRAM is cleared, the firmware password is reset to its default value (**mcp**), and if the console baud rate was previously changed, it is reset to 9600. The time-of-day clock must be restored (refer to Procedure 1.3, "Set Time and Date"). The system and node name must also be restored (refer to Procedure 1.4, "Establish or Change System and Node Names").

A feature of the 3B2 computer is a battery powered calendar that preserves the correct time and date even when the power is turned off. If anything does happen to the battery, or if NVRAM is cleared, the time and date will be lost and the `Time of Day Clock needs Restoring` message will be displayed during powerup.

Procedure 1.8: Enable/Disable Shadow Password

Purpose	To convert/unconvert to/from shadow password operation.
Starting Conditions	System state—multiuser or single user. The **/usr** file system must be mounted. Login—**root**.
Commands	**/usr/bin/pwconv**(1M) **/usr/bin/pwunconv**(1M)
Caution	Modification of the **/etc/passwd** and **/etc/shadow** password files using an editor such as **ed** or **vi** is strongly discouraged. All modifications to the password files should be done using **sysadm**(1), **passwd**(1), or **passmgmt**(1M) commands.
References	Chapter 1, "System Identification and Security."

Command—pwconv

The following screen display shows how to convert to (enable) shadow password operation using the **pwconv** command. A password consistency check is done on the initial conversion from a one password environment to a two password file environment. The consistency check reports logins not having passwords. The screen display shows that executing the **pwconv** command a second time returns only the root prompt.

```
# pwconv
pwconv: WARNING user rar has no password
# pwconv
#
```

Command—pwunconv

The **pwunconv** command disables the shadow password feature. The following screen display shows how to unconvert (disable) shadow password operation using the **pwunconv** command.

```
# pwunconv
#
```

Procedure 1.9: Display Password Information

Purpose	To display password status and aging information for all logins or for a selected login.
Starting Conditions	System state—multiuser or single user. Login—**root**.
Commands	**passwd**(1M) **-s [-a]** **passwd**(1M) **-s [** *name* **]**
References	Chapter 1, "System Identification and Security."

Display all Password Status and Aging Information

All password status and aging information is displayed by the **passwd -sa** command. Password status information is coded as follows:

LK Locked password

NP No password is assigned

PS Password is assigned.

The password aging information is displayed after the password status in the following format:

mm/dd/yy minimum maximum

mm/dd/yy	The date the password was last changed. All zeros (00/00/00) mean that the password must be changed at the next log in.
minimum	The minimum number of days required between password changes.
maximum	The maximum number of days the password is valid.

In the following screen display, only the **vmsys**, **oasys**, **rar**, and **cms** logins have password aging.

```
# passwd -sa
root  PS
daemon  LK
bin  LK
sys  LK
adm  LK
uucp  LK
nuucp  PS
trouble  LK
lp  LK
setup  LK
powerdown  LK
sysadm  LK
checkfsys  LK
makefsys  LK
mountfsys  LK
umountfsys  LK
vmsys  NP  00/00/00 7 63
oasys  NP  00/00/00 7 63
rar  NP  00/00/00  7  30
cms  PS  03/26/88  7  30
#
```

Display Password Status and Aging Information for a Login

The following screen display shows how to use the **passwd -s** command to display the password status and aging information for the **rar** and **cms** logins.

```
# passwd -s rar
rar  NP  00/00/00  7  30
# passwd -s cms
cms  PS  03/26/88  7  30
#
```

Procedure 1.10: Set Password Aging Information

Purpose	To set password aging information for a login.
Starting Conditions	System state—multiuser or single user. Login—**root**.
Command	**/bin/passwd**(1M) **[-n** *min* **] [-f] [-x** *max* **]** *name*
References	Chapter 1, "System Identification and Security."

Add or Change Password Aging Information for a Login

The following screen display shows how to set the password aging information for a login. In this example, the *minimum* number of days permitted between changes is set to 7 and the *maximum* number of days for which the password is valid is set to 30. The attributes of the password for the **rar** login are displayed using the **passwd -s** command before and after changing the password aging information.

```
# passwd  -s rar
rar  PS
# passwd  -n 7 -x 30 rar
# passwd  -s rar
rar  PS  00/00/00  7  30
#
```

Expire a Password for a Login

Expiring a password forces the user to change the password at the next login. There are two ways to expire a password such that existing password aging information can be retained or removed after the user changes the password.

Retain Existing Password Aging Information

The **passwd -f** *name* command is used to expire the password and retain the existing password aging information, if present, for the *name* login. If the new password is to have new password aging, the **-n** and **-x** options must also be used, as applicable. The following screen display shows how to force the **rar** login to change password at the next login and retain the existing password aging information. The attributes of the password for the **rar** login are displayed using the **passwd -s** command before and after expiring the password.

```
# passwd -s rar
rar  PS  03/27/88  7  30
# passwd -f rar
# passwd -s rar
rar  PS  00/00/00  7  30
```

Remove Existing Password Aging Information

The **passwd -n** *min* **-x** *max name* command is used to expire a password and remove the existing password aging information if present for the *name* login. This is a special case of password aging where setting the *minimum* and *maximum* values to the same value requires the user to change the password at the next log in. No further password aging is applied to the login. The following screen display shows how to force the **rar** login to change password at the next log in. The attributes of the password for the **rar** login are displayed using the **passwd -s** command before and after expiring the password.

```
# passwd -s rar
rar  PS  03/27/88  7  30
# passwd -nl -xl rar
# passwd -s rar
rar  PS  03/27/88  1  1
#
```

After the user logs in and changes the password, the password attributes are as follows.

```
# passwd -s rar
rar  PS
#
```

Turn Off Password Aging for a Login

The **passwd -x -1** *name* command is used to turn off password aging for the *name* login. Password aging for a login is turned off by setting the *maximum* field to a **-1**. In the following screen display, the **passwd -s** is used to show the password attributes before and after the turning off password aging for the **rar** login.

```
# passwd -s rar
rar  PS  03/27/88  7  30
# passwd -x -1 rar
# passwd -s rar
rar  PS
#
```

Prevent the User From Changing the Password

The **passwd -n** *min -x max name* command is used to prevent the user from changing the password for the *name* login. This is a special case of password aging where setting the *minimum* to a value greater than the *maximum* requires **root** to change the password. The following screen display shows how to set the password aging information for the **rar** login so that only **root** can change the password. The attributes of the password for the **rar** login are displayed using the **passwd -s** command before and after setting the password aging values.

```
# passwd -s rar
rar  PS  03/27/88  7  30
# passwd -nl -x0 rar
# passwd -s rar
rar  PS  03/27/88  1  0
#
```

Procedure 1.11: Lock/Unlock a Login

Purpose	To lock/unlock a login so it cannot/can be used.
Starting Conditions	System state—multiuser or single user. Login—**root**.
Command	**/bin/passwd**(1M) **[-l | -d]** *name*
References	Chapter 1, "System Identification and Security."

Locking a Login

The **passwd -l** *name* command is used to lock the *name* login. The following screen display shows how to lock the **rar** login so it cannot be used. The attributes of the password for the **rar** login are displayed using the **passwd -s** command before and after locking the login.

```
# passwd -s rar
rar  PS  03/27/88  7  30
# passwd -l rar
# passwd -s rar
rar  LK  03/27/88  7  30
#
```

Unlocking a Login

A login is unlocked by **root** using the **passwd** command to either assign a password to the login or to delete the password.

The password is deleted for a login by using the **passwd -d** command. The login is not prompted for a password when the password is deleted for the login. In the following screen display, the attributes of the password for the **rar** login are displayed using the **passwd -s** command before and after the password is deleted.

```
# passwd -s rar
rar   PS   03/27/88   7   30
# passwd -d rar
# passwd -s rar
rar   NP   03/27/88   7   30
#
```

The **root** login can use the **passwd** command to assign a password to any login. The following screen shows how to assign a password to the **rar** login. Remember that passwords are not echoed. The attributes of the password for the **rar** login are displayed using the **passwd -s** command before and after the password is assign.

```
# passwd -s rar
rar   NP   03/27/88   7   30
# passwd rar
New password:
Re-enter new password:
# passwd -s rar
rar   PS   03/27/88   7   30
#
```

Procedure 1.12: Enable/Disable Unsuccessful Login Logging

Purpose	To enable/disable the tracking of repetitive unsuccessful attempts to log in on a port.
Starting Conditions	System state—multiuser or single user. The **/usr** file system must be mounted. Login—**root**.
Commands	**/bin/chgrp**(1) **/bin/chmod**(1) **/bin/rm**(1)
Caution	The **/usr/adm/loginlog** file can grow in size very rapidly. To prevent the file from getting too large, it is important to occasionally check and clear the contents of the /usr/adm/loginlog file.
References	Chapter 1, "System Identification and Security."

Enable Unsuccessful Login Attempt Logging

As the system is delivered, unsuccessful login attempt logging is turned off (disabled). To enable logging of unsuccessful login attempts, create the **/usr/adm/loginlog** file. The following screen display shows how to create the **/usr/adm/loginlog** file and establish the correct permissions.

```
# > /usr/adm/loginlog
ls -l /usr/adm/loginlog
-rw-rw-r--   /root      xxx            0 Mar 31 02:29 /usr/adm/loginlog
# chmod 600 /usr/adm/loginlog
# chgrp sys /usr/adm/loginlog
# ls -l /usr/adm/loginlog
-rw-------   1 root     sys            0 Mar 31 02:30 /usr/adm/loginlog
#
```

Disable Unsuccessful Login Attempt Logging

To disable logging of unsuccessful login attempts, remove the **/usr/adm/loginlog** file using the **rm**(1) command. The following screen display shows how to remove the **/usr/adm/loginlog** file.

```
#  rm /usr/adm/loginlog
#
```

User Services Procedures

User Services Procedures

The following procedures are covered in this section:

Procedure 2.1 **Add Users or Groups**
To add information about new users of the system or to name groups.

Procedure 2.2 **Modify User or Group Information**
To change information about users or groups.

Procedure 2.3 **Delete Users or Groups**
To remove users or groups from the system.

Procedure 2.4 **List Users or Groups**
To display information about users or groups.

Procedure 2.5 **Write to All Users**
To send a message to all logged-in users.

Procedure 2.1: Add Users or Groups

Purpose	To identify new users or groups of users to the system.
Starting Conditions	System state—multiuser.
sysadm Menu	USER MANAGEMENT
Commands	**sysadm adduser**(1) **sysadm addgroup**(1)
References	"Login Administration" in Chapter 2, "User Services."

This procedure covers two separate functions:

Function 1- Add a user to the system (**sysadm adduser**).

Function 2- Add a group to the system (**sysadm addgroup**).

Function 1: adduser

If you enter the command **sysadm adduser**, you are led through the following sequence:

```
Running subcommand 'adduser' from menu 'usermgmt',
USER MANAGEMENT


Anytime you want to quit, type "q".
If you are not sure how to answer any prompt, type "?" for help,
or see the Owner/Operator Manual.

If a default appears in the question, press <RETURN> for the default.

Enter user's full name [?, q]:  John Q. Public
Enter user's login ID [?, q]:   jqp
Enter user ID number (default 46145) [?, q]:  <CR>
        (Accepting defaults by entering <CR>)
Enter group ID number or group name
(default 1) [?, q]:  <CR>
Enter user's login (home) directory name.
(default '/usr/jqp') [?, q]:  /usr2/jqp <CR>

This is the information for the new login:
        User's name:    John Q. Public
        login ID:       jqp
        user ID:        46145
        group ID:       1       (other)
        home directory: /usr/jqp
Do you want to install, edit, or skip this entry [i, e, s, q]? i
Login installed.
Do you want to give the user a password? [y, n] y
New password:
        (Enter at least six characters, one of them a numeral.)
Re-enter new password:
Do you want to add another login? [y, n, q] n
#
```

Function 2: addgroup

If you enter the command **sysadm addgroup**, you are led through the following sequence:

```
Running subcommand 'addgroup' from menu 'usermgmt',
USER MANAGEMENT


Anytime you want to quit, type "q" .
If you are not sure how to answer any prompt, type "?" for help,
or see the Owner/Operator Manual.

If a default appears in the question, press <RETURN> for the default.

Enter group name [?, q]:   seventy7
Enter group ID number (default 1) [?, q]:  45201

This is the information for the new group:
        Group name:     seventy7
        group ID:       45201
Do you want to install, edit, or skip this entry? [i, e, s, q]: i
Group installed.
Do you want to add another group? [y, n, q] n
#
```

Procedure 2.2: Modify User or Group Information

Purpose	To change stored information about users or groups.
Starting Conditions	System state—multiuser.
sysadm Menu	USER MANAGEMENT
Commands	**sysadm modadduser**(1) **sysadm modgroup**(1) **sysadm moduser**(1) **sysadm chgloginid**(1) **sysadm chgpasswd**(1) **sysadm chgshell**(1) **sysadm chgname**(1)
References	"Login Administration" in Chapter 2, "User Services."

This procedure covers three separate functions:

Function 1- Change the default values that apply to the **adduser** sequence (**modadduser**).

Function 2- Change the name of a group (**modgroup**).

Function 3- Change three of the attributes of user information (**moduser**).

Function 1: modadduser

Step 1: Enter the command:

sysadm modadduser

Step 2: The **sysadm modadduser** command gives you the opportunity to change either or both of the default values for group ID and home (parent) directory that appear on the **adduser** form. The following screen shows an example of changing the default group number from **1** to **45201**.

```
Running subcommand 'modadduser' from menu 'usermgmt',
USER MANAGEMENT


Anytime you want to quit, type "q".
If you are not sure how to answer any prompt, type "?" for help,
or see the Owner/Operator Manual.

Current defaults for adduser:
group ID                1       (other)
parent directory        /usr2
Do you want to change the default group ID? [y, n, ?, q] y
Enter group ID number or group name [?, q]: 45201
Do you want to change the default parent directory? [y, n, ?, q] n

These will be the new defaults:
group ID:               45201   (seventy7)
parent directory:       /usr2
Do you want to keep these values? [y, n, q] y
Defaults installed.

#
```

Function 2: modgroup

To change the value of a group ID name, enter the command:

sysadm modgroup

Function 3: moduser

Step 1: To change the values for an individual user's login, enter the command:

sysadm moduser

The following menu is displayed on your terminal:

```
                    MODIFY USER'S LOGIN

  1 chgloginid      change a user's login ID
  2 chgpasswd       change a user's password
  3 chgshell        change a user's login shell

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  q to QUIT:
```

Step 2: Selecting an item from this menu starts a prompt sequence that helps you make the required change.

When a user is first given a login ID, a default shell (**/bin/sh**) is assigned. The **chgshell** subcommand enables you to assign a different shell.

Step 3: When you select Item 3 **(chgshell)** from the previous menu, or if you entered the command:

sysadm chgshell

the following sequence appears on your terminal:

```
Running subcommand 'chgshell' from menu 'moduser',
MODIFY USER'S LOGIN

Enter user's login ID [?, q]:   jqp
The current shell is /bin/sh.
Enter new shell command [q]:   /bin/rsh
Do you want to change the login shell of another login? [y, n, q] q


Press the RETURN key to see the moduser menu [?, q]: q
#
```

The above sequence assigns a restricted shell to user **jqp**.

Procedure 2.3: Delete Users or Groups

Purpose	To clear the system of an inactive user. To eliminate a group name that is no longer needed.
Starting Conditions	System state—multiuser.
sysadm Menu	USER MANAGEMENT
Commands	**sysadm deluser**(1) **sysadm delgroup**(1)
Caution	When you delete a user's ID, all the files and directories owned by that ID are also deleted.
References	"Login Administration" in Chapter 2, "User Services."

Step 1: Deleting a group ID is done with this command:

sysadm delgroup

Step 2: The prompt sequence is as follows:

```
# sysadm delgroup

Running subcommand 'delgroup' from menu 'usermgmt',
USER MANAGEMENT

Which group name do you wish to delete? [?,q] seventy7
Do you want to delete group name 'seventy7', group ID 45201? [y, n, ?, q] y
seventy7 has been deleted.
Do you want to delete any other groups? [y, n, q] q
```

Note: The **sysadm delgroup** command deletes only the specified group and not the user login(s) assigned to that group. The logins belonging to the group must be deleted separately using **sysadm deluser**.

Step 3: Deleting a user's login ID requires more persistence. The user's home directory and all the files in and below that directory are deleted as well. The sequence is as follows:

sysadm deluser

```
# sysadm deluser

Running subcommand 'deluser' from menu 'usermgmt',
USER MANAGEMENT


This function COMPLETELY REMOVES THE USER, their mail file, home directory
and all files below their home directory from the machine.
Once this is done, there is no way guaranteed to get them all back.
                BE SURE THIS IS WHAT YOU WANT TO DO!

Enter login ID you wish to remove [q]: jqp
        'jqp' belongs to 'John Q. Public'
        whose home directory is /usr/jqp
Do you want to remove login ID 'jqp'? [y, n, ?, q] y

 /usr/jqp and all files under it have been removed.

 jqp has been completely removed.
Enter login ID you wish to remove [q]: q

$
```

Procedure 2.4: List Users or Groups

Purpose	To see what users or groups are known to the system.
Starting Conditions	System state—multiuser.
sysadm Menu	USER MANAGEMENT
Commands	**sysadm lsuser**(1) **sysadm lsgroup**(1)

Step 1: The two **sysadm** subcommands in this procedure enable you to see what groups and what users are in the computer. The command to list groups is:

sysadm lsgroup

This command produces a report with the following column headings:

```
Groups currently in the computer:
(press <RETURN> to start printing each time you hear the bell)

group            group          logins permitted to become
name             number         members using newgrp
-----            ------         -------------------------
adm              4              root,adm,daemon
bin              2              root,bin,daemon
daemon           12             root,daemon
mail             6              root
other            1
root             0              root
sys              3              root,bin,sys,adm
uucp             5              root,uucp
vm               100            vmsys
```

Step 2: If you enter the command:

sysadm lsuser

the following lines appear on your terminal:

```
Running subcommand 'lsuser' from menu 'usermgmt',
USER MANAGEMENT


Users currently in the computer:
(press <RETURN> to start printing each time you hear the bell)
```

When you press <RETURN>, a list in the following form is displayed:

```
login name       user name
----------       ---------
adm              0000-Admin(0000)
bin              0000-Admin(0000)
checkfsys        check diskette file system
daemon           0000-Admin(0000)
jqp              John Q. Public
lp               0000-lp(0000)
makefsys         make diskette file system
mountfsys        mount diskette file system
nuucp            0000-uucp(0000)
oasys            Object Architecture Files
powerdown        general system administration
root             0000-Admin(0000)
setup            general system administration
sys              0000-Admin(0000)
sysadm           general system administration
trouble          trouble(0000)
umountfsys       unmount diskette file system
uucp             0000-uucp(0000)
vmsys            FACE
$
```

Procedure 2.5: Write to All Users

Purpose	To send urgent messages to all logged-in users.
Starting Conditions	System state—multiuser. Login—**root** required to prevent users from blocking messages.
Commands	**wall**(1M)
References	"Write to All Users" in Chapter 2, "User Services."

Step 1: For times when it is necessary to communicate with all users on the system at once, the **wall** command is used. This command reads whatever you type in at your terminal until it reads an end-of-file (shown by typing in a **CTRL-D**).

Step 2: The message you type in is sent immediately to the terminal of all logged in users. It is preceded by:

```
Broadcast Message from ...
```

A typical use of the **wall** command is to warn users that the system is about to be shutdown:

```
Broadcast Message from root:  System coming down in
ten minutes.  Please log off.
```

Processor Operations Procedures

Processor Operations Procedures

The following procedures are covered in this section:

Procedure 3.1 **Powerup**
To power up the system to the multiuser state.

Procedure 3.2 **Powerdown**
To halt the system and turn the power off.

Procedure 3.3 **Shutdown to Single User**
To bring the system to the single-user state to do administrative tasks.

Procedure 3.4 **Return to Multiuser**
To return the system to the multiuser state after it was brought to another state for administrative purposes.

Procedure 3.5 **Run Firmware Programs**
To bring the system to the firmware mode to run special firmware programs.

Procedure 3.6 **Halt and Reboot the Operating System**
To halt and reboot the system from the hard disk.

Procedure 3.7 **Recovery From System Trouble**
To handle system troubles caused by hardware problems and software errors.

Procedure 3.8 **Use the Diagnostic Monitor**
To use the diagnostic monitor.

Procedure 3.9 **Reload the Operating System**
To reload the system from the Operating System Utilities cartridge tape, if the system itself has been severely damaged or to change the disk partitions. Procedures include partial restore, full restore, full restore with default hard disk partitioning, and full restore with user defined hard disk partitioning.

Procedure 3.1: Powerup

Purpose	To turn on the system and make it available for use.
Starting Conditions	System state—power off.
Time	About 5 minutes (depending on configuration).
References	"Power Up" in Chapter 3, "Processor Operations."

To power up the 3B2 computer from a halted state (power is standby), do the following:

Step 1: Turn on the console and wait for the cursor to appear.

Step 2: Turn the latch on the floppy disk drive to the horizontal position to make sure no floppy disk is in the drive, and check the SCSI Cartridge Tape Drive to make sure no cartridge tape is in the drive. Also, make certain the drawer is in the in position (not the out position).

Step 3: Press the power switch. At this point, the standard powerup sequence occurs.

Step 4: Log in to the system when the prompt `Console Login:` appears. You can log in with a system or user login.

The following screen display is a typical response of what appears at the system console.

```
SELF-CHECK
DIAGNOSTICS        PASSED
UNIX (R) System V Release 3.2.2 AT&T 3B2 Version 3
Node unix
Total real memory = 8388608
Available memory  = 5351424
*********************************************************************
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 code.

*********************************************************************
The system is coming up.  Please wait.
```

(File system may be checked before mounting.)

```
mount -f S5  /dev/dsk/c1t1d0s8  /usr2
mount -f S5  /dev/dsk/c1t1d1s2  /usr
AT&T 3B2 SYSTEM CONFIGURATION:

Memory size: 8 Megabytes
System Peripherals:

        Device Name         Subdevices        Extended Subdevices
        SBD
                          Floppy Disk
        SCSI
        (S.E. Bus ID0)
                          SD01 ID1
                                            317 Megabyte Disk ID0
                                            317 Megabyte Disk ID1
                          ST01 ID2
                                            Tape ID0
        EPORTS
        EPORTS
        EPORTS
        MAU
Print services started.
The system is ready.

Console Login:
```

Procedure 3.2: Powerdown

Purpose	To halt the system and turn off the power.
Starting Conditions	System state—multiuser or single user. You must mount **/usr** to run **sysadm** in single-user mode. Login—authorized user or **root**.
sysadm Menu	MACHINE MANAGEMENT
Commands	**sysadm whoson**(1) **sysadm powerdown**(1) **shutdown**(1M)—**root** login only
Cautions	Do not pull the plug until the powerdown procedure is completely finished.
References	"Turn the System Off" in Chapter 3, "Processor Operations."

There are differences in the procedure, depending on whether you are in multiuser or single-user state.

From Multiuser

The best way to turn off the computer while the system is in the multiuser state is to enter the **sysadm powerdown** command. Entering this command causes the system to flush the system buffers, close any open files, stop all user processes and daemons currently running, unmount file systems, and then remove power from the computer.

Step 1: Check who is logged in before taking any action that would affect a logged-in user. Enter:

sysadm whoson

A typical response might be:

```
Running subcommand 'whoson' from menu 'machinemgmt',
MACHINE MANAGEMENT


These users are currently logged in:

   ID   terminal number  sign-on time
------- --------------- ------------
root       console       Apr  11 08:06
jaf        tty22         Apr  11 07:30
```

Step 2: Notify any users that the system is shutting down via the **wall**(1M) command (see Procedure 2.5, "Write to All Users"). For example:

```
# wall<CR>
The system will be coming down in 5 minutes.
Please log off.
<CTRL-D>
Broadcast Message from root (console) on unix Mon Apr 11 15:09:27...
The system will be coming down in 5 minutes.
Please log off.

#
```

Step 3: Enter:

sysadm powerdown

Observe the following output:

```
Running subcommand 'powerdown' from 'machinemgmt',
MACHINE MANAGEMENT

Once started, a powerdown CANNOT BE STOPPED.
Do you want to start an express powerdown? [y, n, ?, q] n
Enter the number of seconds to allow
between the warning messages (default 60): [?, q] 30

Shutdown started.  Mon Apr 11 15:15:54 EDT 1988

Broadcast Message from root (console) on wr3b2d Mon Apr 11 15:15:54
THE SYSTEM IS BEING SHUT DOWN NOW ! ! !
Log off now or risk your files being damaged.

INIT: New run level: 0
The system is coming down.  Please wait.
System services are now being stopped.
Print services stopped.
Stopping job accounting.
cron aborted: SIGTERM

The system is down.

Please flip the power switch to the STANDBY position.
```

At this point, the power is removed. The cabinet lights turn off, the fan turns off, and the hard disk(s) stops spinning.

To protect your system from unauthorized power removal (from a user terminal), assign a password both to the **sysadm** login as well as to the **powerdown**(1M) command (see Procedure 1.5, "Assign Passwords to Administrative and System Logins").

Step 4: Press the power switch to the STANDBY position.

From Single User

If the system is in the single-user state, use the following command to power down the system.

Step 1: To remove power and guarantee file system integrity, use the **shutdown** command as follows:

shutdown -y -i0 -g0

The arguments have the following meanings:

-y assume yes answers all questions
-i0 go to state 0 (off)
-g0 allow grace period of 0 seconds.

Observe the following on the console:

```
Shutdown started.  Thu Mar 31 16:41:40 EDT 1988

Broadcast Message from root (syscon) on unix Thu Mar 31 16:41:41...
THE SYSTEM IS BEING SHUTDOWN NOW ! ! !
Log off now or risk your files being damaged.


INIT: New run level: 0
The system is coming down. Please wait.
System services are now being stopped.
Print services already stopped.
Stopping job accounting.

The system is down.

Please flip the power switch to the STANDBY position.
```

All services are stopped and power is removed from the machine.

Step 2: Press the power switch to the STANDBY position.

Procedure 3.3: Shutdown to Single User

Purpose	To do administrative tasks that should be done when no other users are on the system, such as —software installation —file backup and restore —hard disk formatting —system reconfiguration.
Starting Conditions	System state—multiuser. Login—**root**.
Commands	**shutdown**(1M)
References	"Go to Single-User Mode" in Chapter 3, "Processor Operations."

Shutting the system down to the single-user state should be done as much as possible during off-hours, since only the console has access to the system in the single-user state.

Step 1: Log in as **root** at the console.

Step 2: Enter:

shutdown

By default, **shutdown** prompts you about the various broadcast messages, provides a 60-second grace period between each message, and brings the system to the single-user state. When you arrive in the single-user state, you will see the following:

```
INIT: SINGLE USER MODE
#
```

You may now proceed with your intended tasks.

Procedure 3.4: Return to Multiuser

Purpose	To make the system available to users after administrative duties have been performed in either the single-user state or the firmware mode.
Starting Conditions	System state—single user or firmware. Login—**root**.
Commands	**init**(1M) **/unix** (boot program)
References	"A Look at Entering the Multiuser State" in Chapter 3, "Processor Operations."

There are three system states from which you can return the system to the multiuser state:

- Single-user state
- Firmware mode
- Halt and reboot (see Procedure 3.6, "Halt and Reboot the Operating System").

From Single User

After administrative tasks are finished, you can bring the system back to the multiuser state from the single-user state via the **init** command.

At the console, enter:

init 2

This causes **init** to inspect **/etc/inittab** and execute entries that will initialize the system to the multiuser state. The following is displayed:

```
INIT: New run level: 2
The system is coming up.  Please wait.


Note: The file systems are checked and
the current system configuration is printed out.
After all the file systems are checked the following
messages are displayed:


The system is ready.

Console Login:
```

Now you can log in either as **root** or as a conventional user, since the system is in the multiuser state.

From Firmware

After firmware programs have been run, you can bring the system back from the firmware mode by executing the boot program **/unix** from the hard disk.

Step 1: When you receive the firmware prompt, enter **/unix**:

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

A typical example at the next prompt:

```
     Possible load devices are:

Option Number     Slot      Type       Name
-----------------------------------------
     0            0       INTEGRAL     FD5
     1            1       I/O BUS      SCSI

Enter Load Device Option Number [1 (SCSI)]: <CR>
Possible subdevices are:

Option Number    Subdevice    Name
-----------------------------------------
     0             0          disk
     1             1          tape

Enter Subdevice Option Number [0 (disk)]: <CR>
```

Step 2: After the sanity of the root file system is checked [via **fsstat**(1M)], a file system check is performed if necessary [via **fsck**(1M)], the system configuration is printed out, and the system is placed in the multiuser state. Observe the prompt:

```
Console Login:
```

You may log in with an appropriate system or user login.

Procedure 3.5: Run Firmware Programs

Purpose	To do the following functions: —change the console baud rate —run diagnostics on system hardware —display the equipped device table —print detailed information about an UNEXPECTED FAULT, INTERRUPT, or NMI —enables/disables execution of diagnostics during boot —modify the equipped device table —make a new floppy key floppy disk —change the firmware password —dump the system image —display the system generic version.
Starting Conditions	System state—multiuser or single user. You must mount **/usr** to run this procedure in single user mode. Login—**root**.
sysadm Menu	MACHINE MANAGEMENT
Commands	**sysadm firmware**(1)—may require a password **/etc/shutdown**(1M)—super user only

Firmware Programs	**baud**(8) **edt**(8) **errorinfo**(8) **express**(8) **newkey**(8) **passwd**(8) **sysdump**(8) **version**(8)
Bootable Programs	**dgmon**(8) **filledt**(8) **/unix** **/etc/system**
References	"Run Firmware Programs" in Chapter 3, "Processor Operations." Firmware Programs and Bootable Programs are in the *AT&T 3B2 UNIX System V Release 3 System Administrator's Reference Manual.*

There are many firmware programs that can be run from the hard disk of the 3B2 Computer. Refer to Chapter 3, "Processor Operations," for individual examples of running these programs.

Step 1: To execute any of these programs, you must be in the firmware mode. To enter the firmware mode, log in as **root**, and enter either **sysadm firmware** (or **shutdown -i5**).

```
# sysadm firmware
Running subcommand 'firmware' from menu 'machinemgmt',
MACHINE MANAGEMENT

Once started, this procedure CANNOT BE STOPPED.
Do you want to go to firmware "express" [y, n, ?, q]
```

Step 2: As with shutting down or powering down the system, you have the option to specify a delay between the warning messages. If the system is in the multiuser state, answer **n** (for no) to the question, so there will be a delay in shutting down the system. If you answered no, then you are prompted for the amount of the time (in seconds) to delay the shutdown:

```
Enter the number of seconds to allow
between the warning messages (default 60): [?, q] 30
```

Step 3: Now the shutdown process starts.

```
Shutdown Started
Broadcast Message from root  (console)  Mon Apr 11 17:21:34...
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.
cron aborted: SIGTERM

The system is down.

SELF-CHECK

FIRMWARE MODE
```

Step 4: Enter the firmware password (default is **mcp**):

mcp

Step 5: Enter the particular firmware program name you want to execute in response to the prompt:

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

Select the load device option number from which the program is to be executed [1(SCSI)] and the subdevice option number [0 (hard disk)].

Step 6: Follow the displayed instructions.

Step 7: When the program exits or you exit the program, you will see this prompt:

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

Return to the multiuser state by executing the boot program **/unix** (see Procedure 3.4, "Return to Multiuser").

Procedure 3.6: Halt and Reboot the Operating System

Purpose	To build a new **/unix** automatically on the hard disk after installing a new driver, or executing the **touch**(1) command on the **/etc/system** file (rather than manually booting **/etc/system**).
Starting Conditions	System state—multiuser and single user. You must mount **/usr** to run this procedure in single-user mode. Login—**root**.
sysadm Menu	MACHINE MANAGEMENT
Commands	**sysadm reboot**(1) **shutdown**(1M)
References	"Boot the Operating System" in Chapter 3, "Processor Operations."

This procedure allows you to halt the system and reboot from **/etc/system** on the hard disk only after hardware and/or software changes, or executing the **touch**(1) command on the **/etc/system** file. Its main purpose is to force a reconfiguration of the operating system. See Chapter 6, "Performance Management," for more about reconfiguration. After a new, bootable operating system is created in core, it is written to **/unix**.

To halt and reboot the system from the hard disk, enter **sysadm reboot** (or **shutdown -i6**):

```
# sysadm reboot

Running subcommand 'reboot' from menu 'machinemgmt',
MACHINE MANAGEMENT

Once started, a reboot CANNOT BE STOPPED.
Do you want to start an express reboot? [y, n, ?, q] y

Broadcast Message from root (console) Thu May 16 17:45:09...
THE SYSTEM IS BEING SHUT DOWN NOW ! ! !
Log off now or risk your files being damaged.

INIT: New run level: 6
The system is coming down.  Please wait.
System services are now being stopped.
cron aborted: SIGTERM

The system is down.

The system is being restarted.


SELF-CHECK
DIAGNOSTICS  PASSED
```

At this point, the messages are exactly the same as those that appear when the system is powered up from a halted state. The system is placed in the multiuser state. The following prompt appears:

```
Console Login:
```

Procedure 3.7: Recovery From System Trouble

Purpose	To return the system to a usable state after determining and recording what went wrong.
Starting Conditions	Variable —running, but throughput is degraded —running, after an automatic reboot —halted or in an unknown state. Login—**root**.
sysadm Menu	MACHINE MANAGEMENT
Commands	**errdump**(1M)—if the system is running **sysadm firmware**(1)—to go to firmware **/etc/shutdown**(1M)—to go to firmware **sysdump**(8)—to dump core image **/unix**—boot program
Media	Hard disk crash partition. One SCSI Cartridge Tape. One floppy disk for each 3/4 megabyte of memory.
References	"Run Firmware Programs" in Chapter 3, "Processor Operations." "Clearing Trouble" in the *Owner/Operator Manual* that came with your system.

Determine the System Trouble

Following a system panic error message(s) the current state of the system needs to be determined and recorded. The following procedures present different tools you can use to record and analyze the system status. The procedure that you use depends on the state of the system following the trouble.

For problems that occurred during the first-time setup of the 3B2 computer, consult the section on "Troubleshooting" in the *Owner/Operator Manual* that came with your system. Call your AT&T Service Representative or authorized dealer if you cannot determine what course of action to take.

Step 1: If you have communications between the console terminal and the system (the UNIX operating system is running), go to Step 3. If the system is unable to automatically reboot the UNIX operating system after a problem, you may get the following message displayed:

```
SYSTEM FAILURE:
```

This message shows that your system is in firmware state (operating system is not running). Go to the "Perform a System Dump" section of this procedure.

Step 2: If you do not have communications with the system and the message in Step 1 was not displayed, press the power switch to the STANDBY position. After the hard disks stop spinning, press the power switch to the ON position. The system should establish communications with the console terminal (go to Step 3), or a system failure message will be displayed (go to Procedure 3.8, "Use the Diagnostic Monitor," and run diagnostics).

When you turn the system on, if a line of characters (garbage) is displayed on the system console, the console baud rate may be incorrect. Refer to Chapter 1, "System Identification and Security Procedures," Procedure 1.1, "Check Console Terminal Configuration," or the *Owner/Operator Manual* that came with your system. If you cannot synchronize the console and the system, set the console baud rate to 9600 and do Procedure 1.7, "Forgotten Firmware Password Recovery," in Chapter 1, "System Identification and Security Procedures."

If the system is again in an unknown state, call your AT&T Service Representative or authorized dealer. The following is a list of things that will assist your AT&T Service Representative or authorized dealer when you call:

- System console displays
- DIAGNOSTICS light is on.
- DIAGNOSTICS light is flashing (pattern being flashed should be reported).

Step 3: When the system displays the `Console Login` prompt, log in as **root**.

Step 4: Change directories to **/usr/adm** and examine the errlog file. This will give you some idea of the problems that the system is having.

Step 5: Consult the listing of error messages in Appendix C, "Error Messages," for a description of what happened and what to do about it. Call your AT&T Service Representative or authorized dealer if you cannot determine what course of action to take.

Step 6: Use the **/etc/errdump** command to display the error history file, which includes the contents of various system registers and the last four error messages received. Enter:

errdump

Send the output of **errdump** to a line printer if possible. The next section is an example of an error dump.

Example of errdump

The following command line entries and system responses show how to execute **errdump**. Panic message number [0] is an example of a valid message. Panic message number [1] is valid; however, the pointer to the string is garbled. Reconfiguring the operating system will cause the previously stored panic messages to become garbled. Panic message numbers [2] and [3] show invalid data. When invalid data is read, the contents of memory for the message are dumped in hexadecimal format. When no panics have been recorded, invalid data is normal for all four message slots.

Procedure 3.7: Recovery From System Trouble

```
# errdump
nvram status:   sane

csr:    0xffffff00      (bus timeout) (req reset) (alignment) (led) (floppy)
(unassigned) (inhibit time)

psw:    rsvd CSH_F_D QIE CSH_D OE NZVC TE IPL CM PM R I ISC TM FT
(hex)      3f       1   1     0  0    0  0   7  3  3 1 0   0  0  0

r3:     0x0213f800
r4:     0x0000013f
r5:     0x00000000
r6:     0xc000076c
r7:     0x0011e13f
r8:     0x00000000
oap:    0x40182908
opc:    0x4000c21b
osp:    0x00000000
ofp:    0x4018292c
isp:    0x40180004
pcbp:   0x40160310

fltar:  0x80880f18
fltcr:  reqacc  xlevel  ftype
        0xf     0x0     0x0

        srama           sramb
[0]     0x0213f800      0x0000037f
[1]     0x02141400      0x00000037
[2]     0x00000000      0x00001fff
[3]     0x021415c0      0x00000000

        Panic log

[0]     Mon Apr 11 07:37:11 1988
        Sanity timeout

[1]     Tue Mar 11 07:35:59 1988
         D,LxtV

[2]     (0x83f0d863,0x44601e8b,0xa14e02e1,0x1c11a292)

[3]     (0x3deee77b,0x1261fef3,0x73ef574f,0xfe8f7fb7)
#
```

Perform a System Dump

The **sysdump** (crash) firmware program is used to write the system image [contents of the Random Access Memory (RAM)] to the boot device crash partition (/dev/rdsk/c1t1d0s3 or device specified by DUMPDEV) or to floppy disks. If DUMPDEV is defined in **/etc/system**, then **sysdump** writes the contents of RAM to that device. If the AUTODUMP parameter is set (value=1) the system will automatically dump the mainstore (RAM) to the crash partition in response to a SYSTEM FAILURE (operating system crash). The operating system is delivered with the AUTODUMP parameter set (enabled). If the AUTODUMP parameter is not set (value=0) the **sysdump** command is used to dump the mainstore.

Therefore, as the system is delivered, the contents of the mainstore is automatically written to partition **/dev/rdsk/c1t1d0s3** following a system crash. This default dump device can be redefined by adding a DUMPDEV entry to the **/etc/system** file and rebooting the system. The SCSI Cartridge Tape is defined as the dump device by the entry **DUMPDEV:/dev/rmt/c1t2d0s0**. (Note that automatically dumping the system image to the SCSI Cartridge Tape requires about 2.5-minutes per 4 megabytes.)

Dump Mainstore to Default Devices

The following steps show how to write the system image to the boot device crash partition (/dev/rdsk/c1t1d0s3) and how to then copy the crash partition to the SCSI Cartridge Tape.

Step 1: If the operating system is no longer running (operating system crashed and did not write mainstore to the hard disk), the system will be in the firmware mode. A `SYSTEM FAILURE:` message is displayed.

Step 2: In response to the `SYSTEM FAILURE:` message, enter the firmware password (default password is **mcp**). When you see the prompt:

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

enter **sysdump**. This program dumps the system core image to the crash partition or to floppy disks. Because it runs in firmware mode, it does not depend on the integrity of the **root** file system.

Step 3: In response to the **sysdump** command, the system will display the following:

```
Dump to the boot device crash partition?
```

If you want to write the system dump to this disk partition (and possibly to the SCSI Cartridge Tape later on), enter **y** and go to Step 4. If you want to write the dump to floppy disks or do not want to write the system dump, enter **n** and go to Step 8.

Step 4: After a yes response, the system will display the following:

```
Dumping mainstore
50 100 150 200 250 300 350 400 450 500 550 600 650 600 650 700 750 800 850 900 950
1000 1050 1100 1150 1200 1250
                              .
                              .
                              .
15500 15550 15600 15650 15700 15750 15800 15850 15900 15950 16000 16050 16100
16150 16200 16250 16300 16350
Dump completed
```

The numbers show the amount of memory on the media being written.
This example shows the output for dumping an 8 MB mainstore.

```
SELF-CHECK

FIRMWARE MODE
```

Step 5: When the dump is finished, the program exits and gives you the FIRMWARE MODE prompt. In response, enter the firmware password (default password is **mcp**). You now have the option of running another firmware program.

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

Note: You may run diagnostics after you do the dump. To run diagnostics from the hard disk, however, the root file system (/) must be accessible (undamaged) (see Procedure 3.8, "Use the Diagnostic Monitor").

Step 6: You can return to the multiuser state by executing the boot program (see Procedure 3.4, "Return to Multiuser").

Step 7: After the system is running, you may want to copy the boot device crash partition to a cartridge tape by using the **dd**(1) command. The following is an example of this command.

```
# dd if=/dev/rdsk/c1t1d0s3 of=/dev/rmt/c1t2d0s0 bs=9k
1820+1 records in
1820+1 records out
#
```

You may want to contact your AT&T Service Representative or authorized dealer for help in analyzing the system dump.

Note: The following steps are done **only** if you want a dump to floppy disks.

Step 8: If you enter **n** (no) in response to the question in Step 3, the system will display the following:

```
Dump to the integral floppy disk?
```

If you want to dump the system to floppy disks, enter **y** (yes) and go to the next step. If you do not want to do a system dump, answer the question with **n** (no). The system will display the following:

```
No dump made
```

The program exits and gives you the `FIRMWARE MODE` prompt. In response, enter the firmware password (default password is **mcp**). You now have the option of running another firmware program.

Step 9: Gather the appropriate number of floppy disks for the dump. Use this rule of thumb: you need one floppy disk for each 3/4 megabyte of memory in your configuration.

With...	Use...
4 Megabytes	6 Floppy Disks
8 Megabytes	12 Floppy Disks
12 Megabytes	18 Floppy Disks
16 Megabytes	24 Floppy Disks
20 Megabytes	30 Floppy Disks
24 Megabytes	36 Floppy Disks
28 Megabytes	42 Floppy Disks
32 Megabytes	48 Floppy Disks
36 Megabytes	54 Floppy Disks
40 Megabytes	60 Floppy Disks
48 Megabytes	72 Floppy Disks
52 Megabytes	78 Floppy Disks
64 Megabytes	96 Floppy Disks

Allow about 4.5 minutes per floppy disk used.

Step 10: Follow the displayed instructions which prompt you to insert floppy disks into the floppy disk drive.

Step 11: When the dump is finished, the program exits and gives you the FIRMWARE MODE prompt. In response, enter the firmware password (default password is **mcp**). You now have the option of running another firmware program.

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

Note: You may run diagnostics after you do the dump. To run diagnostics from the hard disk, however, the **root** file system (/) must be accessible (undamaged) (see Procedure 3.8, "Use the Diagnostic Monitor").

Step 12: You can return to the multiuser state by executing the boot program (see Procedure 3.4, "Return to Multiuser").

You may want to contact your AT&T Service Representative or authorized dealer for help in analyzing the system dump.

Example of sysdump

The following command line entries and system responses show how to execute **sysdump** and dump the system image to floppy disks.

```
SYSTEM FAILURE
<mcp>

Enter name of program to execute [ ]: sysdump

Dump to the boot device crash partition? n
Dump to the integral floppy diskette? y
Insert media for first dump volume
Ready to write crash dump? y

Dumping mainstore
50 100 150 200 250 300 350 400 450 500 550 600 650 700 750 800 850 900 950
1000 1050 1100 1150 1200 1250 1300 1350 1400
Write to next dump volume? y
Continuing
50 100 150 200 250 300 350 400 450 500 550 600 650 700 750 800 850 900 950
1000 1050 1100 1150 1200 1250 1300 1350 1400
Write to next dump volume? y
Continuing
50 100 150 200 250 300 350 400 450 500 550 600 650 700 750 800 850 900 950
1000 1050 1100 1150 1200 1250 1300 1350 1400
Write to next dump volume? y
                          .
                          .
                          .
Continuing
50 100 150 200 250 300 350 400 450 500 550 600 650 700 750
Dump completed


SELF-CHECK
FIRMWARE MODE
```

Procedure 3.8: Use the Diagnostic Monitor

Purpose	To determine via a series of test phases what kinds of problems exist with the operating system.
Starting Conditions	System state—5 (firmware).
Commands	**dgmon**(8)—firmware program
References	"Diagnostic Information" in Chapter 3, "Processor Operations."

Diagnostics are run from the firmware mode via the diagnostic monitor program (**dgmon**). Refer to Procedure 3.5, "Run Firmware Programs," to get to the firmware mode. The steps necessary to run diagnostics are as follows:

Step 1: When the prompt:

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

is displayed, enter:

dgmon

This executes the Diagnostic Monitor program. Select the device from which **dgmon** is executed: SCSI (option **1**), and select the subdevice hard disk (option **0**).

Step 2: Run diagnostics as required. Refer to the "Diagnostic Monitor (dgmon)" discussion in Chapter 3, "Processor Operations," for command format.

Step 3: When you are finished running diagnostics, boot the UNIX operating system (**/unix**) from the SCSI hard disk.

Procedure 3.9: Reload the Operating System

Purpose	To partially restore the system as a prerequisite to doing a full restore: —to bring it to a usable state to do a complete backup of any needed files —to remove a forgotten root password. To fully restore the system: —following a partial restore —if there is a new system —if you need to increase the size of the **swap**, **root**, or **usr** partition.
Starting Conditions	System state—multiuser or single user. You must mount **/usr** to run this procedure in single-user mode. Login—root.
sysadm Menu	MACHINE MANAGEMENT FILE MANAGEMENT
Commands	**sysadm firmware**(1) **sysadm backup**(1) **sysadm store**(1)
Media	The Operating System Utilities cartridge tape. SCSI Cartridge Tape or floppy disks to use for backup. Any backup data that has to be reloaded.

Time	The time required to do these tasks will vary with respect to the system size, disk partitioning, amount of backup data to be loaded, and optional utilities to be installed. Partial restore takes about 30 minutes. Full restore takes about 30 minutes. Reload utilities takes about 45 minutes.
Cautions	A partial restore must be followed by a full restore. A full restore erases everything on the disk. If you use the procedure to change partition sizes, do a complete backup first (See Procedure 5.4, "File System Backup and Restore").
References	"Run Firmware Programs" in Chapter 3, "Processor Operations." "Format and Partitions" in Chapter 4, "Disk/Tape Management." "File System Backup and Restore" in Chapter 5, "File System Administration."

Just as file systems have a backup version, so does the operating system: the Operating System Utilities cartridge tape delivered with the system originally. There are two ways to reload the operating system if it is necessary.

- A partial restore allows you to back up the "user" files before you do a full restore. The "user" files are not affected by a partial restore. You must do a full restore after you get a backup copy of the "user" files.
- A full restore erases everything in root and /usr partitions and then loads the operating system files. usr2-type partitions may be saved depending on the partitioning selected. See full restore procedure for more information.

Read through these procedures for reloading the operating system before you begin the procedure. If you understand all the steps, then reload the operating system. If you do not understand the reload procedure, contact your AT&T Service Representative or authorized dealer for help.

Partial System Restore

The partial restore procedure allows you to save some (those that are not corrupted) of the "user" files after system trouble. Certain system files are overwritten during a partial restore. For example, the terminal configuration and password files are overwritten. The **adm**, **root**, and **sys** crontab files are overwritten. The **at.allow** and **cron.allow** files are also overwritten. To make the job of restoring the system configuration easier, these system files are automatically saved in this procedure. Following the partial restore, you can copy these files along with any other "user" files to a removable media before doing a full restore. Your system will not be configured properly if you do a partial restore only; following a partial restore you must do a full restore to properly configure the system. Some of the drivers are disassociated with the kernel when the system is rebuilt using a partial restore.

Step 1: Take the system to the firmware mode (run-level 5). (See Procedure 3.5, "Run Firmware Programs.")

```
# sysadm firmware
```

Step 2: Check that the Operating System Utilities cartridge tape is write-protected (see the *Owner/ Operator Manual* that came with your computer). Then insert the Operating System Utilities cartridge tape into the SCSI Cartridge Tape Drive.

Step 3: Follow the displayed instructions to boot the operating system (**/unix**) from the SCSI Cartridge Tape Drive [default load device option **1** (SCSI), load subdevice option **1** (tape), this is NOT the default answer]. If you have changed your system configuration, the load device option and subdevice option may have changed. There is a discussion of the way that these load devices and subdevices are displayed in the "SCSI Bus Addresses" section of Chapter 4, "Disk/Tape Management." You may also need to refer to this if the system equipment device table (edt) is removed or modified during a restore.

```
FIRMWARE MODE
<mcp>    (The default password for firmware mode is mcp.)
         (Password is not echoed.)
Enter name of program to execute [ ]: /unix
         Possible load devices are:

Option Number     Slot      Type          Name
-------------------------------------------------
       0            0      INTEGRAL      FD5
       1            1      I/O BUS       SCSI
       2            3      I/O BUS
       3            4      I/O BUS
       4            5      I/O BUS
```

There are devices in slots 3, 4, and 5 that correspond to option numbers 2, 3, and 4; however, the system does not know or have drivers for these devices so they do not appear in the name column. Since they could be a bootable device they are listed here.

```
Enter Load Device Option Number [1 (SCSI)]: <CR>
Possible subdevices are:

Option Number    Subdevice     Name
-------------------------------------
       0            0         disk
       1            1         tape
       2            2         disk
       3            3         disk

Enter Subdevice Option Number [0 (disk)]: 1
```

You want to restore from the SCSI cartridge tape drive.

```
ESSENTIAL UTILITIES

Rebooting.
```

A series of messages reporting unknown hardware and software drivers are output when booting the operating system from cartridge tape.

Step 4: Follow the displayed instructions and select the Partial Restore procedure (Option Number 2).

```
UNIX System V Release 3.2.2 Installation

        1)  Full Restore
        2)  Partial Restore

When responding to a question, you may use the "backspace" key
to erase the last character typed or the "@" key to erase the
entire line.  Enter "help" for additional information.

Selection? [ 1 2 quit help ] 2
```

Continued

Continued from previous screen display

```
        -  PARTIAL RESTORE selected  -

The possible ROOT disks configured on the system are:
        1)  /dev/dsk/c1t1d0s6
        2)  /dev/dsk/c1t3d0s6
        3)  /dev/dsk/c1t4d0s6

Select disk to hold the ROOT file system [<1 - 3> quit help] <default 1><CR>

Checking the hard disk file systems.
```

The system will do file system checks on all the file systems that you have set up on your system.

```
Saving system configuration files in /usr/old.
```

A list of /mnt/usr/old contents is output.

```
See /usr/old/README regarding changes to system configuration files.

Installing the initial core system files.
8610 blocks

You may remove the tape.


Performing the "Finishing Touches"...

        Making /dev/root & /dev/swap nodes.
        Making hard disk bootable.

Installation is now complete. The system is restarting itself from
the hard disk. It will be ready to use when you receive the "Console
Login" prompt. This should take about 10 minutes depending on your
configuration.
```

A series of messages are output ending with the following prompt when the system reboots.

```
Console Login:
```

Step 5: Do a backup to SCSI Cartridge Tape or to floppy disks (**sysadm backup**, or **sysadm store**) of the applicable system files in **/usr/old** and "user" files as necessary. (See Procedure 5.4, "File System Backup and Restore.")

The system files saved in the **/usr/old** directory during a typical Partial Restore procedure are as follows:

```
./README
./bin/ed
./bin/red
./dev/contty
./dgn/.edt_swapp
./dgn/edt_data
./edt/SCSI
./edt/SCSI/edt_data
./etc/TIMEZONE
./etc/bupsched
./etc/checklist
./etc/fstab
./etc/gettydefs
./etc/group
./etc/init.d/ANNOUNCE
./etc/init.d/MOUNTFSYS
./etc/init.d/PRESERVE
./etc/init.d/README
./etc/init.d/RMTMPFILES
./etc/init.d/acct
./etc/init.d/adv
./etc/init.d/autoconfig
./etc/init.d/cron
./etc/init.d/disks
./etc/init.d/firstcheck
./etc/init.d/fumounts
./etc/init.d/lock
./etc/init.d/lp
./etc/init.d/mirdisk
./etc/init.d/perf
./etc/init.d/rfs
./etc/init.d/rumounts
./etc/init.d/scsi
./etc/init.d/sysetup
./etc/init.d/unlock
./etc/init.d/uucp
./etc/inittab
./etc/master.d/disp
./etc/master.d/gentty
./etc/master.d/hdelog
./etc/master.d/idisk
./etc/master.d/iuart
./etc/master.d/kernel
./etc/master.d/mem
./etc/master.d/ports
./etc/master.d/s5
./etc/master.d/stubs
./etc/motd
./etc/passwd
./etc/profile
./etc/rc.d/setup
./etc/rc0.d/K00ANNOUNCE
./etc/rc0.d/K20lp
./etc/rc0.d/K22acct
./etc/rc0.d/K50fumounts
./etc/rc0.d/K60fumounts
./etc/rc0.d/K65rfs
./etc/rc0.d/K70cron
./etc/rc0.d/K30fumounts
./etc/rc0.d/K40fumounts
./etc/rc0.d/K50rfs
./etc/rc2.d/S00firstcheck
./etc/rc2.d/S00scsi
./etc/rc2.d/S01MOUNTFSYS
./etc/rc2.d/S02PRESERVE
./etc/rc2.d/S05RMTMPFILES
./etc/rc2.d/S10disks
./etc/rc2.d/S15autoconfig
./etc/rc2.d/S20sysetup
./etc/rc2.d/S21perf
./etc/rc2.d/S22acct
./etc/rc2.d/S70uucp
./etc/rc2.d/S75cron
./etc/rc2.d/S80errstart
./etc/rc2.d/S80lp
./etc/rc2.d/S80restore
./etc/rc3.d/S21rfs
./etc/shadow
./etc/shutdown.d/*
./etc/system
./usr/lib/cron/.proto
./usr/lib/cron/at.allow
./usr/lib/cron/cron.allow
./usr/lib/cron/queuedefs
./usr/spool/cron/crontabs/adm
./usr/spool/cron/crontabs/root
./usr/spool/cron/crontabs/sys
```

If you are doing this restore because of a forgotten **root** password, you should copy all the above files onto SCSI Cartridge Tape or onto floppy disks, and after doing a full restore, copy them back into their proper places.

If you are restoring because of some corruption in the system, you should examine each of these files carefully before putting them back in your system after the full restore. One of these files may have been the cause of the corruption in your system.

Step 6: After you have done a backup of your file systems, you must now do a full system restore.

Full System Restore (Default Partition Size)

Use this procedure to do a full system restore if you are maintaining the SVR 3.2.2 default disk partition sizes. SVR 3.2.2 default disk partition sizes can be found in Appendix A - SCSI Hard Disk Default Partitions. Skip to the next procedure to do a full restore if you are changing the disk partition sizes or adding file systems to your disks.

In the past, full restore erased everything on the root and on the /usr hard disks and then loaded the operating system files. These files are the Essential Utilities. In SVR 3.2.2, full restore has been enhanced to allow /usr2-type file systems (file systems other than root and /usr) to be retained. To take advantage of this new feature, the following conditions must exist:

1. The disk(s) must not be reformatted.
2. The desired block size and placement of partitions 0-3 must be identical to the current block sizes and placement.

On a dual disk system, block sizes of partitions 0 (root), 1 (swap), and 3 (dump) must be identical to the current block sizes for disk1 /usr2-type partitions to be saved. The block size of partition 2 (/usr) on disk 2 must be identical to the current block size for disk2 /usr2-type partitions to be saved.

For a system, disk connected to target controller t1 that is part of the SCSI bus served by the Host Adapter in Slot 1, the current block sizes for the partitions can be found by executing:

```
prtvtoc /dev/rdsk/c1t1d0s6
```

In previous releases, the full restore procedure allocated the rest of the disk(s) as one /usr2-type partition. In SVR 3.2.2, the default block size for /usr and /usr2-type file systems is approximately 120 megabytes depending on the particular disk size. If there is not enough space on the disk for /usr to be 120 megabytes, default partitioning will allocate the remainder of the disk as the default block size for /usr.

If there is enough space to create a new /usr2-type partition(s), full restore will prompt the usr to select a logical file system block size [either 1K (1024 bytes) or 2K [2048 bytes)]. The size selected here will be applied to all /usr2-type partitions made by default partitioning. Full restore will also label the file system. The default file system names used begin with /usr2, /usr3, /usr4, etc. If you already have a saved file system with the same name you will need to relabel (using the labelit command) the newly-created file system so confusion does not result.

Newly-created file systems will be labeled, entered into /etc/fstab, and mounted when the system comes up. Usr2-type partitions saved (untouched) by full restore must be entered into /etc/fstab manually in order to have these partitions mounted during reboot.

When full restore saves an existing file system, you will see the following message:

```
File systems in the following partitions are untouched on the [root/usr] disk:

        Partitions          Size
        X                   YYYYYY
```

Caution: A full restore should not be done until you copy any files you want to keep to a floppy disk(s) or SCSI tape.

Step 1: Take the system to the firmware mode (run-level 5). (See Procedure 3.5, "Run Firmware Programs.")

```
# sysadm firmware
```

Step 2: Check that the Operating System Utilities cartridge tape is write-protected (see the *Owner/ Operator Manual* that came with your computer). Then insert the Operating System Utilities cartridge tape into the SCSI Cartridge Tape Drive. (Remember to enter **mcp** and then **/unix** after you have inserted the type.)

Step 3: Follow the displayed instructions to boot the operating system (**/unix**) from the SCSI Cartridge Tape Drive [default load device option **1** (SCSI), load subdevice option **1** (tape), this is NOT the default device]. If you have changed your system configuration, the load device option and subdevice option may have changed. There is a discussion of the way that these load devices and subdevices are displayed in the "SCSI Bus Addresses" section of Chapter 4, "Disk/Tape Management." You may also need to refer to this if the system **edt** is removed or modified.

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

The system is down

SELF-CHECK

FIRMWARE MODE

# <mcp> (The default password for firmware mode is mcp.)
        (Password is not echoed.)

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

Option Number     Slot      Type          Name
-------------------------------------------------
       0          0         INTEGRAL      FD5
       1          1         I/O BUS       SCSI
       2          2         I/O BUS
       3          3         I/O BUS
       4          4         I/O BUS
       5          5         I/O BUS
       6          0
```

There are devices in slots 3, 4, and 5 that correspond to option numbers 2, 3, and 4. The system does not know or have drivers for these devices so they are not labeled (name column). Since they could be a bootable device they are listed here.

```
Enter Load Device Option Number [1 (SCSI)]: <CR>
Possible subdevices are:

Option Number    Subdevice     Name
-------------------------------------
       0          0           disk
       1          1           tape
       2          2           disk
       3          3           disk

Enter Subdevice Option Number [0 (disk)]: 1

ESSENTIAL UTILITIES

Rebooting.
```

Step 4: Follow the displayed instructions and select the Full Restore procedure (Option Number 1) and select on which hard disk to load the **/usr** file system (default is disk 2).

```
UNIX System V Release 3.2.2 Installation

        1)  Full Restore
        2)  Partial Restore

When responding to a question, you may use the "backspace" key
to erase the last character typed or the "@" key to erase the
entire line.  Enter "help" for additional information.

Selection? [ 1 2 quit help ] 1

        - FULL RESTORE selected -

The possible ROOT disks configured on the system are:
        1)  /dev/dsk/c1t1d0s6
        2)  /dev/dsk/c1t3d0s6
        3)  /dev/dsk/c1t4d0s6

Select disk to hold the ROOT file system [(1-3) quit help] (default 1) 1


The possible USR disks configured on the system are:
        1)  /dev/dsk/c1t1d0s6
        2)  /dev/dsk/c1t3d1s6
        3)  /dev/dsk/c1t4d1s6

NOTE:  System performance will be greater if the ROOT disk is NOT
       the same device as the USR disk.


Select disk to hold the USR file system [(1-3) quit help] (default 2) 1 <CR>
```

Step 5: The system will now allow you to reformat the hard disks. If you reformat the hard disks, any detected bad blocks will be mapped. As the hard disk is reformatted, the Volume Table of Contents (VTOC) will be erased. As the system verifies the hard disk, it reports the bad blocks mapped.

Note: Do not reformat the disks if you want to save /usr2-type partitions.

If you do not format the hard disk (use the default answer by pressing the RETURN key), go to Step 6. The following is what the system displays for formatting.

```
The disk (/dev/dsk/c1t1d0s0) is formatted.
Re-formatting it may remove undetected bad-blocks.

WARNING: Reformatting will destroy the data on the entire disk!

Re-format /dev/dsk/c1t1d0s0? [yes no] (default no) yes
Formatting /dev/rdsk/c1t1d0s0. . .
Format /dev/rdsk/c1t1d0s0:
(DEL if wrong)
Begin Format (No more than 20 minutes)
Begin Verify (No more than 22 minutes)



The disk (/dev/dsk/c1t1d1s2) is formatted.
Re-formatting it may remove undetected bad-blocks.

WARNING: Reformatting will destroy the data on the entire disk!

Re-format /dev/dsk/c1t1d1s2? [yes no] (default no) yes
Formatting /dev/rdsk/c1t1d1s2. . .
Format /dev/rdsk/c1t1d1s2:
Begin Format (No more than 20 minutes)
Begin Verify (No more than 22 minutes)
Mapping bad block 0x1683 (0x18)
Mapping bad block 0xB3DA (0x11)
Mapping bad block 0x3DAF3 (0x17)
```

Step 6: The system asks if you want to use the default hard disk partitioning.

> **Note:** Once default partitioning is selected, the partition sizes cannot be changed as is true in interactive partitioning.

If you want to change the partitions, use the "Full System Restore (Change Partition Size)" section of this procedure. If you want to use the default partitions, follow the displayed instructions. When the tape operating system has been loaded, the system will restart from the hard disk.

The following is displayed on the screen for full restore with default partitioning where /usr2-type partitions are not saved.

```
Default hard disk partitioning assumes certain defaults for the dump
partition.  If this system has more than 16 megabytes (MB) of memory,
the dump partition will be set to the size of memory.  You may not want
to devote this much disk space to the dump partition.  Also, if you
anticipate adding more memory to the system and the total memory at
that time will exceed 16 MB, a full restore will be necessary to
increase the size of the dump partition.  See the documentation
supplied with your computer for more information regarding hard disk
partitioning.

Use the default hard disk partitioning? [yes no quit help] (default yes) yes

Setting up the initial system with default partition sizes.

The following partitions are available on the root disk:
        Partition       Size
        8               245760
        9               7296

Select file system block size for usr2-type partitions<CR>
[ 1024 2048 help ] (default 1024)

Installing the initial core system files.
8610 blocks

You may remove the tape.

Performing the "Finishing Touches"...

        Making /dev/root & /dev/swap nodes.
        Making the hard disk bootable.
        Linking files.

Installation is now complete.  The system is restarting itself from
the hard disk.  It will be ready to use when you receive the "Console
Login" prompt.  This should take about 10 minutes depending on your
configuration.
```

The following is displayed on the screen for full restore with default partitioning where /usr2-type partitions are saved.

```
WARNING: Reformatting will destroy the data on the entire disk!

Reformat /dev/dsk/c12t4d2s2? [ yes no ] (default no)

Default hard disk partitioning assumes certain defaults for the dump
partition.  If this system has more than 16 megabytes (MB) of memory,
the dump partition will be set to the size of memory.  You may not want
to devote this much disk space to the dump partition.  Also, if you
anticipate adding more memory to the system and the total memory at
that time will exceed 16 MB, a full restore will be necessary to
increase the size of the dump partition.  See the documentation
supplied with your computer for more information regarding hard disk
partitioning.

Use the default hard disk partitioning? [ yes no ] (default yes)

Setting up the initial system with default partition sizes.

File systems in the following partitions are saved on the root disk:
        Partition       Size
        8               245745
        9               245745
        a               15480

Installing the initial core system files.
8600 blocks

You may remove the tape.


Performing the "Finishing Touches"...

        Making /dev/root & /dev/swap nodes.
        Making hard disk bootable.
        Linking files.

Installation is now complete.  The system is restarting itself from
the hard disk. It will be ready to use when you receive the "Console
Login" prompt. This should take about 10 minutes depending on your
configuration.
```

Step 7: When the system is ready, follow the displayed instructions to do the System Administration Menu setup procedure and reload the other utilities you want to use refer to the *Owner/Operator Manual* that came with your system for more detail on setting up the 3B2 computer). After this, you can restore the "user" and applicable system files that you backed up before this full restore to ensure that the system configuration is properly restored.

Full System Restore (Change Partition Size)

This procedure is used when partition sizes other than default sizes are desired. In the past, full restore erased everything on the root and on the /usr hard disks and then loaded the operating system files. These files are the Essential Utilities. In SVR 3.2.2, full restore has been enhanced to allow /usr2-type file systems (file systems other than root and /usr) to be retained. To take advantage of this new feature, the following conditions must exist:

1. The disk(s) must not be reformatted.
2. The desired block size and placement of partitions 0-3 must be identical to the current block sizes and placement.

On a dual disk system, block sizes of partitions 0 (root), 1 (swap), and 3 (dump) must be identical to the current block sizes for disk1 /usr2-type partitions to be saved. The block size of partition 2 (/usr) on disk2 must be identical to the current block size for disk2 /usr2-type partitions to be saved.

To determine the block sizes of the current partitions execute the "prtvtoc" command as detailed earlier in this Full Restore procedure.

If there is enough space to create a new /usr2-type partition(s), full restore will prompt the usr to select a logical file system block size [either 1K (1024 bytes) or 2K (2048 bytes)] for each /usr2-type partition made. You will also be asked to select a file system name upon which to mount the partition. The default file system names given here begin with /usr2, /usr3, /usr4, etc. Avoid selecting the default name if you already have a saved file system by the same name. If duplicate file system names occur, the labelit command can be used to change the file system name.

Newly-created file systems will be labeled, entered into /etc/fstab, and mounted when the system comes up. Usr2-type partitions saved (untouched) by full restore must be entered into /etc/fstab manually in order to have these partitions mounted during reboot.

When full restore saves an existing file system, you will see the following message:

> File systems in the following partitions are untouched on the (root/usr) disk:
>
> | Partitions | Size |
> |---|---|
> | X | YYYYYY |

If you are expecting to save a /usr2-type partition(s), verify that full restore saved it by checking the above display. If the partition is not shown as being "untouched", the partition sizes you entered did not match the previous partition sizes. You will be asked to enter "go" to continue or "again" to select new partition sizes.

Caution: A full restore should not be done until you copy any files you want to keep to a floppy disk(s) or SCSI tape.

Before you begin, be certain you have the figures at hand for the number of blocks to be allocated to the various partitions.

Before you overwrite your system, you should type the following commands and note the information that results:

df -t
prtvtoc /dev/rdsk/c1t1d0s6
or
prtvtoc /dev/rdsk/c1t3d0s6

prtvtoc /dev/rdsk/c1t4d0s6

If you have changed your configuration [added disks and/or moved / (root)], you need to print the VTOC on these disks also. The response will be how much space is left on each disk and the partition sizes. Partition sizes are always rounded up to cylinder boundaries.

Step 1: Do Steps 1 through 5 in the "Full System Restore (Default Partition Size)" section of this procedure.

Step 2: (Interactive Positioning)
After you finish the hard disk formatting portion of the Full Restore procedure, you are asked about the partition sizes. At this point, you define the sizes of hard disk partitions to tailor the computer to your own application. Use the figures you worked out beforehand to complete the repartitioning. Remember, if you increase the number of blocks in one partition, there are fewer blocks available for the remaining partitions (partitions are rounded up to the next cylinder boundary). Also, if there is any change in the block size of partitions 0-3, /usr2-type partitions will not be saved.

After completing interactive partitioning, check the displays to verify that file system(s) that you expect to be saved are "untouched" by full restore. If new disk partition sizes are chosen, the vtoc of the disk(s) is displayed. You should then verify that the partition sizes are what you wanted. Enter "go" if partition sizes are correct; enter "again" to specify different partition sizes.

The following is displayed on the screen for full restore with interactive partitioning where /usr2-type file systems are not saved.

```
Default hard disk partitioning assumes certain defaults for the dump
partition.  If this system has more than 16 megabytes (MB) of memory,
the dump partition will be set to the size of memory.  You may not want
to devote this much disk space to the dump partition.  Also, if you
anticipate adding more memory to the system and the total memory at
that time will exceed 16 MB, a full restore will be necessary to
increase the size of the dump partition.  See the documentation
supplied with your computer for more information regarding hard disk
partitioning.

Use the default hard disk partitioning? [yes no help] (default yes)no

How many blocks for the "sysdump" partition?
[ (0 - 131072) quit again help ] (default 32768) <CR>
[ (0 - 131072) quit again help ] (default 32768) <CR>

How many blocks for the "swap" partition?
[ (10106 - 591708) quit again help ] (default 43581) <CR>

How many blocks for the "root" partition?
[ (12636 - 561150) quit again help ] (default 54180) <CR>

How many blocks for the "usr" partition?
[ (126684 - 235200) quit again help ] (default 235200) 200000

Making file systems on the hard disk(s)....      (please wait)

There are 506970 blocks remaining on disk /dev/dsk/c1t3d0s6.

How many blocks for partition /dev/dsk/c1t3d0s8?
[ (0 - 506970) again quit help ] (default 245745)

Select file system block size for partition /dev/dsk/c1t3d0s8
[ 1024 2048 again quit help ] (default 1024)
```

(continued)

Continued from previous screen display

```
Upon what directory should the file system within partition 8
be mounted? [ (pathname) again quit help ] (default /usr2)

How many blocks for partition /dev/dsk/c1t3d0s9?
[ (0 - 261225) again quit help ] (default 245745)

Select file system block size for partition /dev/dsk/c1t3d0s9
[ 1024 2048 again quit help ] (default 1024)

Upon what directory should the file system within partitions 9
be mounted [ (pathname) again quit helpl ] (default /usr3)

How many blocks for partition /dev/dsk/c1t3d0s10?
[ (0 - 15480) again quit help ] (default 15480)

Select file system block size for partition /dev/dsk/c1t3d0s10
[ 1024 2048 again quit help ] (default 1024)

Upon what directory should the file system within partitions 10
be mounted? [ (pathname) again quit help ] (default /usr4)

There are 35035 blocks remaining on disk /dev/dsk/c12t4d2s6.

How many blocks for partition /dev/dsk/c12t4d2s8?
[ (0 - 35035) again quit help ] (default 35035)

Select file system block size for partition /dev/dsk/c12t4d2s8
[ 1024 2048 again quit help ] (default 1024)

Upon what directory should the file system within partition 8
be mounted? [ (pathname) again quit help ] (default /usr5)
```

(continued)

Continued from previous screen display

```
ROOT disk partitioning:

*                          First     Sector     Last
* Partition   Tag   Flags   Sector    Count     Sector     Mount Directory
       0       2     00     43731     54180     97910      /mnt
       1       3     01       150     43581     43730
       3       0     01    606556     32768    639323
       4       0     00    604881      1675    606555
       6       0     01         0    639324    639323
       7       0     01         0       150       149
       8       0     00     97911    245745    343655      /mnt/usr2
       9       0     00    343656    245745    589400      /mnt/usr3
       a       0     00    589401     15480    604880      /mnt/usr4

USR disk partitioning

*                          First     Sector     Last
* Partition   Tag   Flags   Sector    Count     Sector     Mount Directory
       2       0     00       245    200165    200409      /mnt/usr
       6       0     01         0    235445    235444
       7       0     01         0       245       244
       8       0     00    200410     35035    235444      /mnt/usr5

This completes the interactive partitioning of your core system. Enter
"go" if you are ready to proceed.  Enter "again" to specify
different partitioning.

Type "go" to proceed, "again" to start over [ go again ] go

Installing the initial core system files.
8600 blocks

You may remove the tape.

Performing the "Finishing Touches"...

        Making /dev/root & /dev/swap nodes.
        Making hard disk bootable.
        Linking files.

Installation is now complete. The system is restarting itself from
the hard disk. It will be ready to use when you receive the "Console
Login" prompt. This should take about 10 minutes depending on your
configuration.
```

The following is displayed on the screen for full restore with interactive partitioning where /usr2-type file systems are saved.

```
Reformat /dev/dsk/c12t4d2s2? [ yes no ] (default no)

Default hard disk partitioning assumes certain defaults for the dump
partition.  If this system has more than 16 megabytes (MB) of memory,
the dump partition will be set to the size of memory.  You may not want,
to devote this much disk space to the dump partition.  Also, if you
anticipate adding more memory to the system and the total memory at
that time will exceed 16 MB, a full restore will be necessary to
increase the size of the dump partition.  See the documentation
supplied with your computer for more information regarding hard disk
partitioning.

Use the default hard disk partitioning? [ yes no help ] (default yes) no

How many blocks for the "sysdump" partition?
[ (0 - 131072) quit again help ] (default 32768)

How many blocks for the "swap" partition?
[ (10106 - 591708) quit again help ] (default 43581)

How many blocks for the "root" partition?
[ (12636 - 561150) quit again help ] (default 54180)

How many blocks for the "usr" partition?
[ (126684 - 235200) quit again help ] (default 235200) 200000

Making file systems on the hard disk(s)....    (please wait)
```

```
File systems in the following partitions are untouched on the root disk:
        Partition       Size
        8               245745
        9               245745
        a               15480

File systems in the following partitions are untouched on the usr disk:
        Partition       Size
        8               35035

This completes the interactive partitioning of your core system.  Enter
"go" if you are ready to proceed.  Enter "again" to specify
different partitioning.

Type "go" to proceed, "again" to start over [ go again ] go

Installing the initial core system files.
8600 blocks

You may remove the tape.


Performing the "Finishing Touches"...


        making /dev/root & /dev/swap nodes.
        Making the hard disk bootable.
        Linking files.

Installation is now complete.  The system is restarting itself from
the hard disk.  It will be ready to use when you receive the "Console
Login" prompt.  This should take about 10 minutes depending on your
configuration.
```

Step 3: When the system is ready, follow the displayed instructions to do the System Administration Menu setup procedure and reload the other utilities you want to use. Refer to the *Owner/Operator Manual* that came with your system for more detail on setting up the 3B2 computer.

After this, you can restore the "user" and applicable system files that you backed up before this full restore to ensure that the system configuration is properly restored.

Disk/Tape Management Procedures

Disk/Tape Management Procedures

The following procedures are covered in this section:

Procedure 4.1 **Format Floppy Disks**
To prepare floppy disks for use and verify they are usable.

Procedure 4.2 **Duplicate Floppy Disks**
To make exact copies of floppy disks.

Procedure 4.3 **Check for Hard-Disk Errors**
To see if disk errors have been logged.

Procedure 4.4 **Assign Default Boot Program and Device**
To set the default boot program name and device.

Note: Some variation in these procedures may occur depending on the configuration of your system. If you have an additional SCSI Cartridge Tape or a second floppy disk drive, the **sysadm** menus and prompts will differ.

Duplicate Cartridge Tape — There is a method to copy the entire cartridge tape described in the "Duplicate SCSI Cartridge Tape" section of Chapter 4, "Disk/Tape Management."

Disk Mirroring — "The Disk Mirroring Feature" and associated commands are described in Chapter 4, "Disk/Tape Management."

Procedure 4.1: Format Floppy Disks

Purpose	To format a floppy disk so it can be used by the system.
Starting Conditions	System state—multiuser or single user. You must mount **/usr** to run this procedure in single-user mode. You must be at the computer to insert and remove the floppy disks. Login—an authorized user.
sysadm Menu	DISK MANAGEMENT
Commands	**sysadm format**(1)
Media	Floppy disks to be formatted.
Time	About 1 minute per floppy disk. About 3.5 minutes per floppy disk with verification.
References	"Format Disks and Floppy Disks" in Chapter 4 of Part 2."

Step 1: Enter the following command:

```
# sysadm format

More than one subcommand or submenu name matches 'format'.
    1    diskmgmt/format
    2    diskmgmt/harddisk/format
Select one:  [?, q]  1

Running subcommand 'format' from menu 'diskmgmt',
DISK MANAGEMENT
```

Step 2: You are prompted to insert the medium to be formatted and to say whether the format should be verified; for example:

```
Do you want each format verified? (default: yes)[y, n, ?, q]:y
Insert the medium in the diskettel drive. Press <RETURN> when ready.[q]<CR>

Formatting in progress
```

Step 3: At the conclusion of the formatting process, the following message is displayed on your terminal:

```
The medium in the diskettel drive is now formatted and verified;
it may be removed.

Insert another medium in the diskettel drive.
Press <RETURN> when ready.  Type q to quit.
```

Procedure 4.2: Duplicate Floppy Disks

Purpose	To make a copy of the contents of a floppy disk.
Starting Conditions	System state—multiuser or single user. You must mount **/usr** to run this procedure in single-user mode. You must be at the computer to insert and remove floppy disk. Login—an authorized user.
sysadm Menu	DISK MANAGEMENT
Commands	**sysadm cpdisk**(1)
Media	A new formatted floppy disk for each one to be copied.
Time	About 6 minutes per copy (3 minutes to read to hard disk; 3 minutes to write to new floppy disk)
References	"Duplicate Disks" in Chapter 4 of Part 2.

With a single floppy disk drive, the technique for duplicating floppy disks is to read the contents into a temporary file from the source floppy disk and then write it out to a new floppy disk. If your 3B2 computer is equipped with two floppy disk drives, duplication is done drive-to-drive.

Step 1: From the console, enter the following command:

sysadm cpdisk

Step 2: Follow the displayed instructions to make a copy of a floppy disk.

```
Running subcommand 'cpdisk' from menu 'diskmgmt',
DISK MANAGEMENT

Insert the ORIGINAL medium to be COPIED IN the diskettel drive.
It is recommended that you write-protect the original.
Press <RETURN> when ready [q] <CR>
The original is being copied in.
Copy in complete.
You may now remove the medium from the diskettel drive.

To make a COPY of the original
insert the medium TO BE WRITTEN into the diskettel drive.
Press <RETURN> when ready. Type q to quit. <CR>
The original is being copied out onto the duplicate medium.
Copy out complete.
You may now remove the medium from the diskettel drive.

To make ANOTHER COPY of the original
insert the medium TO BE WRITTEN into the diskettel drive.
Press <RETURN> when ready.  Type q to quit. q
```

Step 3: At this point you may take the floppy disk out of the drive. You will receive the following prompt:

```
To make ANOTHER COPY of the original,
insert the medium TO BE WRITTEN into the diskettel drive.
Press <RETURN> when ready.  Type q to quit.
```

If you want another copy of what is already on the hard disk, you can insert a new floppy disk into the drive.

Note: You do not have to put the first floppy disk back in the drive. The copy that you put on the hard disk stays there until you quit this procedure. You can make as many copies to as many floppy disks as you want, all from the one copy on the hard disk.

If you are through duplicating, you must type **q** to stop the process.

Procedure 4.3: Check for Hard-Disk Errors

Purpose	To check hard disk error report.
Starting Conditions	System state—multiuser or single user. You must **mount /usr** to run this procedure in single-user mode. Login—an authorized user.
sysadm Menu	SYSTEM DIAGNOSTICS
Commands	**sysadm diskreport**(1)
References	"Bad Block Handling Feature" in Chapter 4 of Part 2.

While new disk errors occur rarely, you should check the disk report regularly. The damage caused by uncorrected errors can be considerable.

Step 1: Enter the command:

```
# sysadm diskreport
Password:

Running subcommand 'diskreport' from menu 'diagnostics',
SYSTEM DIAGNOSTICS

        WARNING:  This report is provided to advise you if your machine
        needs the built-in disk repaired.  Only qualified repair people
        should attempt to do the repair.
        NOTE:  If disk errors are reported it probably means that files
        and/or data have been damaged.  It may be necessary to restore the
        repaired disk from backup copies.

Select "full" or "summary" disk error report [?,q]; f
```

Step 2: The following report is displayed on your terminal:

```
Disk Error Log: Full Report for maj=121 min=6
        log created:  Fri Aug  6 01:44:34 1987
        last changed: Tue Aug 21 05:57:35 1987
        entry count: 0
        no errors logged

Disk Error Log: Full Report for maj=121 min=22
        log created:  Fri Aug  6 09:39:42 1987
        last changed: Tue Oct 20 12:33:32 1987
        entry count: 0
        no errors logged
```

The above log report was a reading in Oct. 1987. If there are errors reported, see "The Bad Block Handling Feature" in Chapter 4 of Part 2.

Procedure 4.4: Assign Default Boot Program and Device

Purpose	To set the default boot program name and device.
Starting Conditions	System state—multiuser or single user. You must mount **/usr** to run **sysadm** in single-user mode. Login—an authorized login.
sysadm Menu	MACHINE MANAGEMENT
Commands	**sysadm autold**(1) **fltboot**(1M)
Caution	Before changing the default boot device from option 1 subdevice option 0 (the first hard disk off the first target controller) to another device option, make the other device bootable. This helps to avoid the possibility of autobooting from a device that is not properly configured.
References	"Assignment of Default Boot Program and Device" in Chapter 4 of Part 2.

There are two ways to assign the default boot program name and device: **sysadm autold** and **fltboot**. The **sysadm autold** calls **fltboot**.

System Administration Menu—autold

The following command line entries and system responses show the setting of the default boot program name and device to boot **/etc/system** from the first SCSI hard disk drive.

```
# sysadm autold

Running subcommand 'autold' from menu 'machinemgmt',
MACHINE MANAGEMENT


You may specify the default file for manual load, the device for auto load,
or both.

Typical files to be loaded are /unix, a fully configured UNIX, or /etc/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.

Change the manual load program or auto load device? [y, n, q] y

Enter name of default program for manual load [ /unix ]: /etc/system

Possible load devices are:

Option Number    Slot     Name
-------------------------------------
       0          0          FD5
       1          1          SCSI
enter number corresponding to autoload device desired [ 1 ]:<CR>
NULL response detected, current value will be maintained
```

Continued

```
Continued from previous screen display
Possible subdevices are:

Option Number   Subdevice    Name
-------------------------------------------
       0           0              disk
       1           1              tape

Enter Subdevice Option Number [0(disk)]: <CR>
NULL response detected, current value will be maintained

LOAD PARAMETER UPDATE COMPLETE

Select what to do next:
        continue this session
        firmware
        powerdown
        reboot
[c, f, p, r, ?]: c
#
```

Command—fltboot

The following command line entries and system responses show the setting of the default boot program name and device to boot **/etc/system** from the first SCSI hard disk drive using the **fltboot** command.

```
# fltboot

Enter name of default program for manual load [ /unix ]: /etc/system

Possible load devices are:

Option Number    Slot     Name
-------------------------------------
       0          0          FD5
       1          1          SCSI
enter number corresponding to autoload device desired [ 1 ]:<CR>
NULL response detected, current value will be maintained

Possible subdevices are:

Option Number   Subdevice    Name
-----------------------------------------
       0           0              disk
       1           1              tape

Enter Subdevice Option Number [0(disk)]: <CR>
NULL response detected, current value will be maintained

LOAD PARAMETER UPDATE COMPLETE
#
```

File System Administration Procedures

File System Administration Procedures

The following procedures are covered in this section:

Procedure 5.1 **Create File System on Floppy Disk**
To define a file system on a floppy disk.

Procedure 5.2 **Create File Systems on Hard Disk**
To define additional file systems when more than one disk device is available.

Procedure 5.3 **Maintain File Systems**
To check and possibly repair file systems.
To monitor disk space usage.
To reorganize disk space.

Procedure 5.4 **File System Backup and Restore**
To provide a storage copy of active files.
To archive unneeded files.
To bring files and file systems back from storage.

Note: Some variation in these procedures may occur depending on the configuration of your system. If you have a second floppy disk drive, this will be reflected in the **sysadm** menus and prompts.

Procedure 5.1: Create File System on Floppy Disk

Purpose	To define file systems that are removable for reasons of privacy or security. To write identifying labels on the magnetic medium so the system can know what is brought on-line. To bring a file system under UNIX system control (mount), or to release it so it can be removed from the system (unmount).
Starting Conditions	System state—multiuser or single user. You must mount **/usr** to run this procedure in single-user mode. You must be at the computer to insert and remove the floppy disks. Login—an authorized login.
sysadm Menu	DISK MANAGEMENT
Commands	**sysadm diskmgmt**(1) **sysadm makefsys**(1) **makefsys**(1M) **sysadm mountfsys**(1) **sysadm umountfsys**(1)
Media	One formatted floppy disk for each file system to be created. The floppy disk must be mounted to be used as a file system.
Time	About 3.5 minutes per floppy disk.

Cautions	If you use **sysadm** commands to make or mount file systems, you cannot use **umount**(1) to unmount the file system. Also, if **mount**(1) is used to mount the file system, you cannot use **sysadm** to unmount the file system.
References	Chapter 5, "File System Administration."

Before doing this procedure, make sure that the floppy disks you plan to use have been formatted and are not write-protected (see Procedure 4.1, "Format Floppy Disks").

In this procedure you are prompted to name a directory that will be the mount point for a file system. Users should be informed that they should not keep other files in that directory.

Step 1: Enter one of these two commands:

```
$ sysadm makefsys
Password:
    or
$ makefsys
Password:

Running subcommand 'makefsys' from menu 'diskmgmt',
DISK MANAGEMENT
```

Step 2: You will be prompted for further information as follows:

```
Insert the medium in the diskettel drive. Press <RETURN> when ready. [q]<CR>
        NOTE: Making a file system involves labeling the medium
              and then mounting the medium.
Enter the label to be put on the medium[?, q]? fsys01
```

(This writes a label on the magnetic medium. The name should be six characters or less. A paper label with the same file system name should be affixed to the front of the floppy disk and/or protective envelope.)

```
Enter the file system name[?, q]? dir01
```

(This makes a directory that will be used as the mount point for the file system. The name should be six characters or less.)

```
Enter the maximum number of files and directories on this medium
(default 200)[q]: <CR>
```

(The range is 1—711.)

```
Building 'dir01' file system on 'fsys01'.
Initializing 'dir01' file system.
Do you want to leave 'dir01' mounted?[y, n, ?, q]: y
Mounted.  DO NOT REMOVE THE MEDIUM UNTIL IT IS UNMOUNTED!
```

The names and other responses used in the above scenario are arbitrary. Use names and responses appropriate for your application.

Step 3: If you want to mount a file system that has already been created, select **mountfsys** from the **sysadm diskmgmt** menu. The procedure is as follows:

```
# sysadm mountfsys

Running subcommand 'mountfsys' from menu 'diskmgmt',
DISK MANAGEMENT

Do you want to mount the file system read-only? [y, n] n
Insert the medium in the diskettel drive. Press <RETURN> when ready, [q]<CR>
Disk 'fsys01', file system 'dir01', mounted.
DO NOT REMOVE THE MEDIUM UNTIL IT IS UNMOUNTED!
#
```

Step 4: If you want to unmount a file system that is mounted on the system, you may select **umountfsys** from the **sysadm diskmgmt** menu. Do not use **umount**(1M) if the file system was mounted with **mountfsys**.

Procedure 5.2: Create File Systems on Hard Disk

Purpose	To define additional file systems on a second hard disk device —to give a user group a dedicated portion of the disk space —to balance the distribution of data on the disk. To write identifying labels on the magnetic medium so the system can know what is being brought on-line. To bring a file system under UNIX system control (mount) or to release it so it can be removed from the system (unmount).
Starting Conditions	System state—single user. You must mount **/usr** to run this procedure in single-user mode. Login—**root**.
sysadm Menu	HARD DISK MANAGEMENT
Commands	**sysadm partitioning**(1) **mkfs**(1M) **mkdir**(1) **mount**(1M) **umount**(1M)
Media	A second hard disk device.
Time	About 3 minutes for **mkfs**.
References	"Use **mkfs**" in Chapter 5, "File System Administration."

Normally, it is suggested that the **usr** and **root** file systems be placed on separate hard disks (on a dual disk system). The following **sysadm** procedure will work only if you put both the **usr** and **root** file systems on the first hard disk by doing a full restore and defining the partitions. See the "Full System Restore (Change Partition Size)" section of Procedure 3.9, "Reload the Operating System."

Use sysadm to Make File Systems (Partition the Second Hard Disk)

Note: Displays in this procedure are examples for a 155-megabyte hard disk.

The commands for partitioning a hard disk are located in the **harddisk** submenu of the DISK MANAGEMENT Menu. The procedure for repartitioning a hard disk is as follows:

Step 1: You need to determine the number of file systems and the number of blocks that each file system requires.

Step 2: Display the VTOC for the second hard disk. This will give you the number of blocks that can be used for your file systems. The following display shows the **sysadm display** command line and the associated VTOC for the second hard disk.

```
# sysadm display<CR>

Running subcommand 'display' from menu 'harddisk',
HARD DISK MANAGEMENT

Select which drive to use:
        1 disk1          2 disk2
Enter a number, a name, the initial part of a name, or
? for HELP, q to QUIT: 2
Displaying disk2 drive partitioning (hardware slot 1,
target controller 1, drive 1):

* /dev/rSA/disk2 partition map
*
* Dimensions:
*     512 bytes/sector
*      35 sectors/track
*       9 tracks/cylinder
*     315 sectors/cylinder
*     962 cylinders
*     960 accessible cylinders
*
* Flags:
*   1: unmountable
*  10: read-only
*
*                          First      Sector     Last
* Partition  Tag  Flags   Sector     Count      Sector   Mount Directory
       6      0    01         0    629760     629759
       7      0    01         0       984        383
#
```

From the VTOC you see that there are 302,085 blocks that can be used by your file systems.

Step 3: Use the **sysadm partitioning** command to define the partitions and mount points (file systems) on the second hard disk. The following shows the command line entries and the system responses for the **sysadm partitioning** command. In the following example, two partitions of 200,000 blocks are made. The remaining blocks are defaulted into a third partition. The system will round up to the next cylinder boundary.

```
# sysadm partitioning

Running subcommand 'partitioning' from menu 'harddisk',
HARD DISK MANAGEMENT

Select which drive to use:
    1 disk1        4 disk12       7 disk4       10 disk7
    2 disk10       5 disk2        8 disk5       11 disk8
    3 disk11       6 disk3        9 disk6       12 disk9
Enter a number, a name, the initial part of a name, or
? for HELP, q to QUIT: 2

There are 302085 blocks remaining on disk 10.

How many blocks for disk 10 partition 8?
[ (0 - 302085) again quit help ] (default 245700) (depends on size of disks)

Select file system block size for disk 13 partition 8
[ 1024 2048 again quit help ] (default 1024)

Upon what directory should the file system within disk 10 partition 8
be mounted? [ (pathname) again quit help ] (default /usr3)

How many blocks for disk 10 partition 9?
[ (0 - 56385) again quit help ] (default 56385)

Select file system block size for disk 10 partition 9
[ 1024 2048 again quit help ] (default 1024)

Upon what directory should the file system within disk 10 partition 9
be mounted? [ (pathname) again quit help ] (default /usr4)

The disk3 drive is now partitioned.
```

Step 4: You now need to display the VTOC for the newly partitioned second disk. This will give you the partition sizes for the new file systems. The following screen shows the **sysadm display** command displaying the VTOC of the hard disk that was just partitioned.

```
# sysadm display<CR>

Running subcommand 'display' from menu 'harddisk',
HARD DISK MANAGEMENT

Select which drive to use:
        1 disk1           2 disk2           3 disk3
Enter a number, a name, the initial part of a name, or
? for HELP, q to QUIT: 3
Displaying disk3 drive partitioning (hardware slot 1,
target controller 4, drive 0):

* /dev/rSA/disk3 partition map
*
* Dimensions:
*     512 bytes/sector
*      35 sectors/track
*      12 tracks/cylinder
*     384 sectors/cylinder
*    1642 cylinders
*    1640 accessible cylinders
*
* Flags:
*   1: unmountable
*  10: read-only
*
*                          First     Sector    Last
* Partition  Tag  Flags   Sector    Count     Sector   Mount Directory
       6      0    01        0     629760    629759
       7      0    01        0       384       383
       8      0    00      384     200064    200447   /usr4
       9      0    00   200488     245760    446207   /usr5
       a      0    00   446208     183552    629759   /usr6
#
```

Note: The system rounded the values up to the cylinder boundary.

Use mkfs to Create File Systems

If the circumstances do arise where you need to use this procedure, you will need the following information before you begin:

- The name of the special device file [see **mknod**(1M) in the *User's and System Administrator's Reference Manual*] that is to contain the file system.
- The number of blocks (and optionally, i-nodes) the file system is to have.

You may package some of this information, plus details of directories and files to be copied to the new file system, in a prototype file. See **mkfs**(1M) in the *User's and System Administrator's Reference Manual* for details.

The procedure includes the **mount** and **umount** UNIX system commands, which are used to make file systems available for use or to remove them from use. Because most standard file systems are automatically mounted and unmounted during the startup and shutdown procedures, you seldom have to use these commands.

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

Step 2: Make a directory to use as the mount point for the new file system.

```
# mkdir /usr7
        (/usr3 is the name of the directory where
        the new file system will be mounted.)
```

Step 3: Use the **mkfs** command like this:

```
# mkfs  /dev/dsk/c1t1d1s8  100170:12512
```

The arguments on the command line are:

/dev/dsk/c1t1d1s8 *(the name of the device on which the file system resides)*
100170:12512 *(the number of blocks and i-nodes)*

Step 4: When the command has been entered, you receive a prompt like this:

```
Mkfs: /dev/dsk/c1t1d0s8?
(DEL if wrong)
```

The command waits for 10 seconds before proceeding. If anything on the command line looks incorrect, you have a chance to cancel the command by pressing the DELETE key.

Step 5: The **mkfs** command reports some of the attributes of the file system.

```
bytes per logical block = 2048
total logical blocks = 25042
total inodes = 12512
gap (physical blocks) = 7
cyl size (physical blocks) = 400
mkfs: Available blocks = 24648
#   (root prompt)
```

The figures that appear on your screen may not correspond exactly to those shown.

Step 6: Assign a label for the file system using the **labelit**(1M) command, as shown in the following display:

```
# labelit /dev/dsk/c1t1d0s8 usr7 3.2.2
Current fsname: , Current volname: , Blocks: 100168, Inodes: 12512
FS Units: 2Kb, Date last modified: Wed Mar 29 14:23:43 1989
NEW fsname = usr7, NEW volname = 3.2.2 - - DEL if wrong!!
#
```

Step 7: Mount the new file system with the following command:

```
# mount /dev/dsk/c1t1d1s8 /usr3
mount /dev/dsk/c1t1d1s8 /usr3
```

(The arguments on the command line are **/dev/dsk/c1t1d1s8**, *the name of the device on which the file system resides, and* **/usr3**, *the mount point directory.)*

Step 8: Make a directory in the new file system called **lost+found**; mount file system. The **mklost+found** command automatically creates lost and found directory, creates files, and removes files. If successful, there is no output.

```
mklost+found/usr3 /dev/dsk/c1k1d1s8
```

The **lost+found** directory is used by the file system checking utility, **fsck**. Adding and removing files sets up some available blocks to which **fsck** can assign lost files. The recommended number of empty i-nodes for the **lost+found** directory is roughly one-quarter of the i-node count for the file system. A script like the following can be helpful:

```
#       pop -- populate (and remove)
#       usage: pop pfx [ number ]
#       makes 'number' (default = 5) files: pfx1, pfx2, ...
number=${2-5}
i=1
while test "$i" -le "$number"
do
        > $1$i
        i="`expr $i + 1`"
done
rm $1*
```

Step 9: The file system is now available for use. This procedure may also be used to define file systems on floppy disks.

Procedure 5.3: Maintain File Systems

Purpose	To check and possibly repair **removable** file systems so the integrity of the file system is assured. To be informed on how disk space is being used so adequate resources can be provided to users. **Note:** For hard disk file system information, refer to Part 2, Chapter 5.
When Performed	Before mounting the file system. On a schedule appropriate for your circumstances.
Starting Conditions	System state—multiuser or single user. For checking, the file system must NOT be mounted. For other maintenance, the file system must be mounted. To run this procedure in single-user mode, you must mount **/usr**. You must be at the computer to insert and remove the media.
sysadm Menu	FILE MANAGEMENT DISK MANAGEMENT
Commands	**sysadm checkfsys**(1) **sysadm diskuse**(1) **sysadm fileage**(1) **sysadm filesize**(1) **fsck**(1M)
Media	Floppy disk that contains file system to be checked.
Time	About 3.5 minutes per file system.
References	"Maintain File Systems" in Chapter 5, "File System Administration."

File System Checking for Floppy Disk

Step 1: Insert the floppy disk into its drive and close the latch.

Step 2: Enter the command:

sysadm checkfsys

Step 3: Next, you are prompted to select the appropriate drive and press **RETURN** when ready.

Step 4: You are then asked to choose a type of checking from the following display:

```
Disk 'fs1.v1', file system '/fsys01'
Select:
   check              (Any errors detected are reported, but not fixed.)
   interactive        [You are asked to approve(disapprove) fixes.]
   automatic          (Any errors detected are automatically fixed.)
[c, i, a, q, ?]:
```

Step 5: Following your response, file checking proceeds. If you selected **interactive** and if an error is detected, a message appears on your screen that describes the error and asks for a **yes** or **no** response.

Note: The error messages are in section "Run fsck" in Chapter 5 of Part 2.

Step 6: When the check is completed, the following message appears:

```
27 files 94 blocks 1266 free
You may now remove the medium from the diskette drive.
```

The operation is finished.

Monitor Disk Usage on Hard Disk

The second part of the file system maintenance procedure involves various ways of making sure that enough space is available on hard disk to accommodate the users' needs.

Step 1: Enter the command **sysadm diskuse**.

Step 2: The following appears on your screen:

```
Running subcommand 'diskuse' from menu 'filemgmt',
FILE MANAGEMENT


FILE SYSTEM USAGE AS OF 03/30/88 12:43:25

File            Free    Total   Percent
System          Blocks  Blocks  Full
------          ------  ------  -------
/               26600   36854   27%
/usr            277572  302084  8%
/usr2           39368   40000   1%
```

Step 3: If you judge that a more detailed look at files is needed, there are two commands you can use: **sysadm fileage** and **sysadm filesize**. The **sysadm fileage** command prints the names for all files older than the date you specify. The **sysadm filesize** command prints the names of the *n* largest files (number of files) in the directory you specify.

Step 4: The **fileage** command prompts you for two pieces of information:

1. The full path name of the directory to search:

   It is important to be specific in your response. If you select a high-level directory, such as **/usr**, you will get a great deal more information than you want.

2. The number of days to go back (default is 90 days).

Step 5: The **filesize** command displays information on the *n* largest files (default is 10) in a directory named by you.

The heading for the information displayed is:

```
                File size        Date
Owner           (characters)     last access        File name
-----           ------------     -----------        ---------
```

The information in the display depends on your application.

Procedure 5.4: File System Backup and Restore

Purpose	To store file systems or parts of file systems: —to guard against loss of data —to free up space on the disk.
When Performed	On a schedule developed to fit the needs of your system.
Starting Conditions	System state—single user. You must mount **/usr** to run this procedure in single-user mode. You must be at the computer to insert and remove the media. Login—**root**.
sysadm Menu	FILE MANAGEMENT
Commands	**sysadm backup**(1) **sysadm store**(1) **sysadm restore**(1) **sysadm bupsched**(1) **sysadm hsbackup**(1) **sysadm hsrestore**(1) **mount**(1M)
Media	Formatted floppy disks or cartridge tapes in enough quantity to hold the files or file systems you are backing up. Floppy disks hold 1,422 512-byte blocks. 60–MB SCSI Cartridge Tapes hold 125,604 512-byte blocks. 120–MB SCSI Cartridge Tapes hold 266,004 512-byte blocks.

References | "File System Backup and Restore" in Chapter 5, "File System Administration."

When doing a backup, you have a choice of media to use: floppy disk, cartridge tape, or 9-track tape. Using floppy disks for a complete backup is a time consuming process; therefore, we recommend using cartridge tape or 9-track tape. Most of the procedures that you will find in this section use cartridge tape as the backup media.

If you select tape as the backup media, you have a choice of doing a "standard backup" or using the multiple save set feature to do the backup. The multiple save set feature allows you to append data to the same tape; a standard backup only allows one backup per tape.

Complete Backup

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

Step 2: Take the system to the single-user mode (run-level S or 1). See Procedure 3.3, "Shutdown to Single User."

Step 3: The System Administration Menu package resides in **/usr**, so you must mount that file system plus any other file systems that you want to back up. You can mount individual file systems using the **mount** command, for example, **mount /usr**, or you can mount all file systems with the following command:

mountall

Step 4: Enter the command:

sysadm backup

```
Running subcommand 'backup' from menu 'filemgmt',

FILE MANAGEMENT

Available file systems:
/         /usr          /usr2         ALL
Enter file system(s) you want to backup [?, q]: /usr2
Select complete or incremental backup [c, i, ?, q]: c
Print each file name as it is copied? [y, n, ?, q]: n
Select which drive to use:
   1 diskettel     2 qtapel
Enter a number, a name, the initial part of a name, or
? for HELP, q to QUIT: 2

Select multiple save set or standard backup [m, s, ?, q] (default s):
```

If you select a standard backup, the following messages will be displayed:

```
Select multiple save set or standard backup [m, s, ?, q] (default s): s

         Before inserting the first tape into the qtapel drive, mark it:

         Standard
         Complete Backup of: /usr2 ,
            120MB Cartridge Tape
         Wed. 03/09/88, 08:08:31 AM
                part 1
Insert the tape in the qtapel drive.  Press <RETURN> when ready. [q]<CR>
Retensioning tape.

.....(If you asked for file names to
be printed, they will appear here in place of the dots.)
12200 blocks

Complete backup of  /usr2  on qtapel drive finished.
You may remove the tape when the rewind completes.
#
```

If you select multiple save set, the following messages will be displayed:

```
Select multiple save set or standard backup [m, s, ?, q] (default s): m
Insert the tape in the qtapel drive.  Press <RETURN> when ready. [q]<CR>

NOTICE: Selection of complete backup will overwrite contents of tape.
Do you wish to continue with the backup? [y, n, q] y

        Mark the tape in qtapel drive after the backup has completed.

        Multiple Save Set
        Complete Backup of: /usr2 ,
           120MB Cartridge Tape
        Wed. 03/09/88, 08:08:37 AM
               part 1

Retensioning tape.
Signals will be ignored.

..........(If you asked for file names to
be printed, they will appear here.)

12200 blocks

Complete backup of  /usr2  on qtapel drive finished.
You may remove the tape when the rewind completes.
#
```

Step 5: Label each tape (or floppy disk) used for the backup. Include a sequence number as part of the label (Part 1, Part 2, etc.).

Step 6: Return the system to the normal operating state. (See Procedure 3.4, "Return to Multiuser.")

Incremental Backup

A complete backup of the file system must be performed before an incremental backup of that file system can be done. See the discussion on backup strategies in Chapter 5, "File System Administration," under "File System Backup and Restore."

When doing an incremental backup, you have a choice of media to use: floppy disk, cartridge tape, or 9-track tape. If you only have a small amount of data to backup, floppy disks may be the right choice; for large amounts of data, select tape.

The following procedure uses tape as the backup media. When you do an incremental backup to tape, you have a choice of doing a "standard backup" or using the multiple save set feature to do the backup. The multiple save set feature allows you to append data to the same tape; a standard backup only allows one incremental backup per tape.

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

Step 2: Take the system to the single-user mode (run-level S or 1). See Procedure 3.3, "Shutdown to Single-User."

Step 3: The System Administration Menu package resides in **/usr**, so you must mount that file system plus any other file systems that you want to back up. You can mount individual file systems using the **mount** command, for example, **mount /usr**, or you can mount all file systems with the following command:

mountall

Step 4: Execute the System Administration backup command (**sysadm backup**) and follow the displayed instructions.

```
# sysadm backup
Running subcommand 'backup' from menu 'filemgmt',
FILE MANAGEMENT

Available file systems:
/           /usr        /usr2       ALL
Enter file system(s) you want to backup [?, q]: /usr2
Select complete or incremental backup [c, i, ?, q]: i
Print each file name as it is copied? [y, n, ?, q]: n
Select which drive to use:
   1 diskettel          2 qtapel
Enter a number, a name, the initial part of a name, or
? for HELP, q to QUIT: 2
Select multiple save set or standard backup [m, s, ?, q] (default s):
```

If you select standard backup, the following messages will be displayed:

```
Select multiple save set or standard backup [m, s, ?, q] (default s):s

          Before inserting the first tape into the qtapel drive, mark it:

          Standard
          Incremental Backup of /usr2,
          Sat. 03/05/88, 08:08:31 AM to
          Sat. 03/12/88, 09:04:21 AM
                     part 1

Insert the tape in the qtapel drive.  Press <RETURN> when ready. [q]<CR>
Retensioning tape.

180 files to save since Sat. 03/05/88, 08:08:31 AM on /usr2
.....(If you asked for filenames to
be printed, they will appear here.)
8300 blocks

Incremental backup of  /usr2  on qtapel drive finished.
You may remove the tape when the rewind completes.
#
```

If you select multiple save set, the following messages will be displayed:

```
Select multiple save set or standard backup [m, s, ?, q] (default s):m
Insert the tape in the qtapel drive.  Press <RETURN> when ready. [q]<CR>

          Mark the tape in qtapel drive after the backup has completed.

          Multiple Save Set
          Incremental Backup of: /usr2 ,
          Sat. 03/05/88, 08:08:31 AM to
          Wed. 03/09/88, 09:04:21 AM
                     part 1

180 files to save since Sat. 03/05/88, 08:08:31 AM on /usr2
Signals will be ignored.

.....(If you asked for filenames to
be printed, they will appear here.)
8300 blocks


Incremental backup of  /usr2  on qtapel drive finished.
You may remove the tape when the rewind completes.
#
```

Step 5: Label the medium as shown in the displayed instructions.

Step 6: Return the system to the normal operating state. See Procedure 3.4, "Return to Multiuser."

Selective Backup Using Floppy Disk

The **sysadm store** command can be used to quickly save specific directories and files. This method of storing data is often referred to as "selective backups."

If you are only copying a few files, floppy disks are ideal for most selective backup applications. Cartridge tapes or 9-track tapes, however, can also be used for selective backups. The example used in the following procedure shows a floppy disk being used. The next section covers a selective backup using tape.

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

Step 2: Take the system to the single-user mode, run-level S or 1. See Procedure 3.3, "Shutdown to Single User."

Step 3: The System Administration Menu package resides in **/usr**, so you must mount that file system plus any other file systems that contain the files that you want to back up. You can mount individual file systems using the **mount** command, for example, **mount /usr**, or you can mount all file systems with the following command:

```
mountall
```

Step 4: Label the media to show the directory or file.

Step 5: Insert the media in the appropriate drive.

Step 6: Enter the command:

sysadm store

The display on your terminal is as follows:

```
Running subcommand 'store' from menu 'filemgmt',
FILE MANAGEMENT

Select which drive to use:
    1 diskettel       2 qtapel
Enter a number, a name, the initial part of a name, or
? for HELP, q to QUIT:1

       1. Select a single file for storing.
       2. Select all files under a directory for storing.
Enter a number [?, q]: 1
Enter full path name of file to be stored [q]:
      /usr/abc/file1

       1. Select a single file for storing.
       2. Select all files under a directory for storing.
       3. List files selected so far.
       4. Store selected files.
Enter a number [?, q]: 4

  1 files selected.

Print each file name as it is being stored? [y, n, ?, q] y
```

Step 7: Each time you provide a file or directory name, you are prompted to enter more names, to review what has been entered, or to proceed with the storing process. When you have entered all the names of files to be stored, enter a **4**. You will see the following display:

```
      Before inserting the first part into the diskettel drive, mark it:

            Files stored on:
            Fri. 03/04/88, 04:18:53 PM
                  part 1

Insert the medium in the diskettel drive.  Press <RETURN> when ready. [q]
/usr/abc/file1
.....(If you asked for filenames to
be printed, they will appear here in place of the dots.)
28 blocks
Store complete.
Do you want to verify that your file(s) were stored properly [y,n,?,q] y
PLEASE NOTE:
To verify that the store worked properly, you must re-insert
all parts that were just written to, starting with "part 1"

Insert the medium in the diskette drive. Press <RETURN> when ready.[q]<CR>
28 blocks
Verification complete.
Should the stored files be removed from the built-in disk [y, n, ?, q] n
You may now remove the medium.
```

This last question gives you a chance to remove the files just written to the floppy disk. You would probably elect to do that if you were in the process of freeing up storage space.

Step 8: Return the system to the normal operating state.

Selective Backup Using Tape

When you choose to do a multiple save set store, you are asked to select the "create" or "append" mode. If you select the create mode, a multiple save set type of store is performed and the new store is written at the beginning-of-tape. (Any data stored on the tape is lost.) If you select the append mode, the files to be stored are appended to a current multiple save set.

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

Step 2: Take the system to the single-user mode, run-level S or 1. See Procedure 3.3, "Shutdown to Single-User."

Step 3: The System Administration Menu package resides in **/usr**, so you must **mount** that file system plus any other file systems that contain the files that you want to back up. You can mount individual file systems using the **mount** command, for example, **mount /usr**, or you can mount all file systems with the following command:

mountall

Step 4: Label the tape to shown the directory or file.

Step 5: Insert the tape in the appropriate drive.

Step 6: Enter the command

sysadm store

The following messages are displayed:

```
Running subcommand 'store' from menu 'filemgmt',
FILE MANAGEMENT

Select which drive to use:
    1 diskettel        2 qtapel
Enter a number, a name, the initial part of a name, or
? for HELP, q to QUIT:2
Select multiple save set or standard backup [m, s, ?, q] (default s)
```

The following messages are displayed if you select a standard backup:

```
Select multiple save set or standard backup [m, s, ?, q] (default s) s

       1. Select a single file for storing.
       2. Select all files under a directory for storing.
Enter a number [?, q]: 1
Enter full path name of file to be stored [q]:
      /usr/abc/file1

       1. Select a single file for storing.
       2. Select all files under a directory for storing.
       3. List files selected so far.
       4. Store selected files.
Enter a number [?, q]: 4

 6 files selected.

Print each file name as it is being stored? [y, n, ?, q] n

          Before inserting the first tape into the qtape1 drive, mark it:

          Standard
          6 files stored on:
          Wed. 03/09/88, 03:05:25 PM
          120MB Cartridge Tape
                 part1

Insert the tape in the qtape1 drive.  Press <RETURN> when ready. [q]
Retensioning tape.
.....(If you asked for filenames to
be printed, they will appear here.)
45 blocks

Store complete.
Do you want to verify that your file(s) were stored properly? [y,n,?,q]n
Should the stored files be removed from the built-in disk? [y,n,?,q]n
```

The following messages are displayed if you select multiple save set:

```
Insert the tape in the qtapel drive. Press <RETURN> when ready. [q]
Select create or append mode [c, a, ?, q]: c

    Note: If you select the append mode, you must be appending to a tape
    previously written by a multiple save set store.

Select multiple save set or standard backup [m, s, ?, q]
(default s) m

       1. Select a single file for storing.
       2. Select all files under a directory for storing.
Enter a number [?, q]: 1
Enter full path name of file to be stored [q]:
      /usr/abc/file1

       1. Select a single file for storing.
       2. Select all files under a directory for storing.
       3. List files selected so far.
       4. Store selected files.
Enter a number [?, q]: 4

  5 files selected.

Print each file name as it is being stored? [y, n, ?, q] n

          Mark the tape in qtapel drive after the backup has completed.

          Multiple Save Set
          5 files stored on:
          Wed. 03/09/88, 03:05:25 PM
          120MB Cartridge Tape
                 partl

Retensioning tape.
.....(If you asked for filenames to
be printed, they will appear here.)
43 blocks

Store complete.
Should the stored files be removed from the built-in disk? [y,n,?,q]n
```

Restore

File restorals can be done from floppy disk or tape. The same command (**sysadm restore**) is used for either medium. The following procedure is an example of a file restoral from a floppy disk. A restoral from tape is similar.

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

Step 2: Take the system to the single-user mode (run-level S or 1). See Procedure 3.3, "Shutdown to Single User."

Step 3: The System Administration Menu package resides in **/usr**, so you must mount the **/usr** file system plus any other file systems that contain files you want to restore. You can mount individual file systems using the **mount** command, for example, **mount /usr**, or you can mount all file systems with the following command:

mountall

Step 4: Enter the command

sysadm restore

and follow the displayed instructions. When restoring from a complete or incremental backup, all media of that series must be loaded. Even if you intend to restore only a single file, all media of the backup series must be loaded in sequence.

```
# sysadm restore

Running subcommand 'restore' from menu 'filemgmt',
FILE MANAGEMENT

Select which drive to use
    1 diskettel       2 qtapel
Enter a number, a name, the initial part of a name, or
? for HELP, q to QUIT: 1

Select:
        1. restore a single file
        2. restore a directory of files
        3. restore all files
        4. list all the files
Enter a number [q,?]: 1
Insert the medium in the diskettel drive. Press <RETURN> when ready. [q]<CR>
Do you want to see the file status information, too? [y, n] n
Enter full path name of file(s) to be restored [q, ?]:
         /usr/abc/file1
Do you want to rename the file as it is copied in? [y, n, q]: y
WARNING:
        Be very careful when you rename a file.  Files incorrectly named
        by typing errors are difficult to find and repair.
        Remember that only the first 14 characters of each part of the
        file name (i.e. the characters between the "/"s) are significant.
You will be asked to rename each file in turn.  An empty response (<CR>)
skips that file.  An answer of period "." restores the file with its original
name.  If you do not specify a full pathname, the file will be restored under
/tmp of /usr/tmp (a message will tell you where to find it).

Rename </usr/abc/file1>
/usr/abc/datafile3
.........................
83 blocks

Restoration complete.
You may now remove the medium from the diskette drive.
Select:
        1. restore a single file
        2. restore a directory of files
        3. restore all files
        4. list all the files
Enter a number [q,?]: q

#
```

Step 5: Return the system to the normal operating state. See Procedure 3.4, "Return to Multiuser."

Step 6: Store the backup diskettes in a safe place.

High-Speed Backup

A high-speed backup can only be done on a file system that is unmounted; however, to do a high-speed backup the **/usr** and **root** file systems must be mounted. Therefore, you cannot do a high-speed backup of the **/usr** or **root** file systems.

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

Step 2: Take the system to the single-user mode (run-level S or 1). See Procedure 3.3, "Shutdown to Single User."

Step 3: The System Administration Menu package resides in **/usr**, so you must mount that file system.

```
mount /usr
```

Step 4: Enter the command:

```
sysadm hsbackup
```

```
Running subcommand 'hsbackup' from menu 'filemgmt',
FILE MANAGEMENT

Which drive contains the file system that you want to backup?
        1 disk1          2 disk2
Enter a number, a name, the initial part of a name, or
? for HELP, q to QUIT: 1
Select the file system you wish to backup
[ 4(fs:instal) 8(fs:usr2) (q for quit) ] 8

        The selected partition contains:
                File System Name"usr2"; Label Name "3.2.2"; Size = "201170"

Select which drive the file system will be copied to.
    1 disk1          2 disk2          3 qtape1
Enter a number, a name, the initial part of a name, or
? for HELP, q to QUIT: 3

Before inserting the first tape into qtape1 drive, mark it:

        Volume Backup of usr2,
        sysadm hsbackup - 120M cartridge
        Sat. 05/21/88, 05:06:40 AM
        unix unix 3.2.2 3 3B2

Insert the tape in the qtape1 drive. Press <RETURN> when ready. [q]<CR>

*******************************************************************************
NOTE!!!
Before the hsbackup begins, you may receive warning messages from the "volcopy"
command. There messages will require a response of "y" to continue.

The source is disk1 drive partition 8.
This drive is known as /dev/rdsk/c1t1d0s8 by volcopy.

The destination is qtape1 drive known as /dev/rmt/c1t2d0s0 by volcopy.

The number of blocks copied will be displayed at the end.
The volcopy supports the multi-cartridge/multi-drive feature.
Please refer to the manual page "volcopy(1m)" with questions.
*******************************************************************************
Starting backup of usr2:
```

Continued

```
Continued from previous screen display
Tape 3.2.2, 969 feet, 6250 BPI
You will need 1 tape(s).
(       The same size and density is expected for all tapes)
From: /dev/rdsk/c1t1d0s8, to: /dev/rmt/c1t2d0s0? (DEL if wrong)
Retensioning tape. Please wait.

Writing tape 1 of 1, VOL = 3.2.2
   END: 201107 blocks.

backup of  usr2  onto /dev/rmt/c1t2d0s0 finished.
You may remove the tape when the rewind completes.
#
```

Step 5: You can now remove the SCSI Cartridge Tape. Ensure that the tape is correctly labeled.

Step 6: Return the system to the normal operating state.

High-Speed Restore

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

Step 2: Take the system to the single-user mode (run-level S or 1). See Procedure 3.3, "Shutdown to Single User."

Step 3: The System Administration Menu package resides in **/usr**, so you must mount that file system.

mount /usr

Step 4: Enter the command:

sysadm hsrestore

```
Running subcommand 'hsrestore' from menu 'filemgmt',
FILE MANAGEMENT

From which drive will the file system be restored?
    1 disk1          2 disk2          3 qtape1
Enter a number, a name, the initial part of a name, or
? for HELP, q to QUIT: 3

Insert the tape in the qtape1 drive. Press <RETURN> when ready. [q]<CR>

To which drive will the file system be restored?
    1 disk1          2 disk2
Enter a number, a name, the initial part of a name, or
? for HELP, q to QUIT: 1

Which partition on disk1 drive will the file system be restored to?
[ 4(fs:instal) 8(fs:usr2) (q for quit) ] 8

*******************************************************************************
NOTE!!!
Before the hsrestore begins, you may receive warning messages from the "volcopy"
command. There messages will require a response of "y" to continue.

The last message will be:
        From: (source), to : (destination)? (DEL if wrong)
The source is qtape1 drive known as /dev/rmt/c1t2d0s0 by volcopy.

The destination is disk1 drive partition 8.
This drive is known as /dev/rdsk/c1t1d0s8 by volcopy.

The number of blocks copied will be displayed at the end.
Please refer to the manual page "volcopy(1m)" with questions.
*******************************************************************************
Starting restore of usr2:
```

Continued

```
Continued from previous screen display
Tape 3.2.2, 969 feet, 6250 BPI
/dev/rdsk/c1t1d0s8 less than 48 hours older than /dev/rmt/c1t2d0s0
To filesystem dated:  Tue Mar 8 13:32:04 1988
type 'y' to override:      y
From /dev/rmt/c1t2d0s0, to: /dev/rdsk/c1t1d0s8? (DEL if wrong)
Retensioning tape. Please wait.

Reading tape 1 of 1, VOL = 3.2.2
  END: 201107 blocks.

restore of  usr2  onto /dev/rdsk/c1t1d0s8 finished.
You may remove the tape when the rewind completes.

The file system on partition 8 on disk1 drive must be file
system checked before it can be mounted. To check the file system
type fsck /dev/rdsk/c1t1d0s8 after exiting from the sysadm command.
#
```

Step 5: Return the system to the normal operating state. See Procedure 3.4, "Return to Multiuser." The file system will be automatically checked when the system is returned to the normal operating state.

Step 6: Store the backup SCSI Cartridge Tape in a safe place.

Backup Schedule Reminders

You may do this procedure with any authorized login.

Step 1: Enter the command:

sysadm bupsched

Step 2: The following display will appear on your terminal:

```
                    BACKUP REMINDER SCHEDULING

  1 schedcheck  schedule backup reminder checks
  2 schedmsg    schedule backup reminder message

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP, q for QUIT:
```

Step 3: To schedule a reminder message to be sent to the system console during shutdown, enter a **2**. The following message is displayed on your terminal.

```
Enter the command you wish to execute [p, r, a, m, w, ?, q]:
```

The choices for this prompt (and what is displayed if you enter a question mark) are the following:

```
p - Print lines of the file
r - Remove a line or group of lines
a - Add a line
m - Modify a line
w - Write the changes into the file
q - Quit - Leave bupsched
```

Note: The referenced file contains reminder messages and the schedule for sending the messages to the system console.

Step 4: Enter an **a** to add a line, and the following sequence of prompts occurs:

```
Enter time intervals in which backup reminder messages are to be printed
[?, q]: 16:00-18:00

        (The time interval must be entered with no blanks)

Enter time intervals in which backup reminder messages are to be printed
[?, q]: q

        (Many prompts in this sequence are repeated to let you
        enter additional information. When you have entered as
        much as is appropriate for the particular prompt, a
        q takes you to the next prompt.)

Enter the day of the week [0, 1, ... 5, 6, *, ?, q]: *

        [The asterisk (*) means you want to schedule the message
        to appear every day. Day 0 is Sunday.]

Enter the day of the month [1, 2, ..., 30, 31, *, ?, q]: 15

        (The 15 means you want to schedule the message to appear
        on the 15th of the month. This modifies the Day-of-the-Week
        schedule to mean any day that is the 15th of the month.)

Enter the day of the month [1, 2, ..., 30, 31, *, ?, q]: q

        (The prompt is repeated to let you enter additional days
        for this message to be scheduled. q ends the sequence.)

Enter the month [1, 2, ..., 11, 12, *, ?, q]: *

        [The asterisk (*) means you want to schedule the message
        to appear every month.]
```

Continued

Continued from previous screen display

```
Enter  the file system you wish to backup [?, q]: /usr2
```

[The file system name must start with a slash (/).
The prompt will re-appear until you enter **q**.*]*

```
Enter the command you wish to execute [p, r, a, m, w, ?, q]:  w
```

(The **w** *means you want to add what you just entered*
to the file of reminder messages.)

```
Enter the command you wish to execute [p, r, a, m, w, ?, q]:
```

The sequence is over. You can exit with a **q** or choose another function.

Step 5: To see all the scheduled reminder messages, enter a **p**. The following display appears on your terminal:

NOTE: In this display the pound sign (#) means the line is interpreted as a comment when the file is executed.

```
#
#       Format of lines
#
#time       day       month       list
#
#       time   - time(s) of day (24hr or am/pm)
#       day    - day(s) of week (mon, tue, etc)
#                day(s) of month (1,2,...first, last)
#       month  - month(s) of the year (jan, feb,...)
#       list   - list of file systems to be backed up
#                or command to be executed ( !command line)
#
#       Example:
#
#4:00-18:00   mon   *   /usr
#
#       If ckbupscd is invoked between 4:00 and 6:00 in the
#       evening on Mondays during any month of the year,
#       display /usr as the name of a file system that needs
#       to be backed up.
#
#================================================
#
#       Default backup schedule calls for daily backups of /usr
#       and monthly backups of root (/) on the 15th of each month.
#
   1   0:00-8:00,16:00-23.59 mon,tue,wed,thu,fri   *   /usr
   2   4pm-11pm              15        *          /
   3   16:00-18:00           15        *          /usr2
```

(Line 3 in the list is the line you just added.)

Step 6: Once there are reminder messages in the file, you can use **schedcheck**, selection 1 from the **bupsched** menu, to schedule checks for reminder messages. Reminder messages are sent to the console if a shutdown occurs during the interval specified in the file.

Schedule check is an added protection if a shutdown does not take place during the specified interval. It looks for messages that would have been sent had a shutdown occurred. The prompt sequence is similar to the one shown previously.

System Reconfiguration Procedures

System Reconfiguration Procedures

The following procedures are covered in this section:

Procedure 6.1 **Reconfigure the System**
To rebuild the operating system after tuning resulting from changes to hardware or operating system software.
To tune your system performance for your application.

Procedure 6.2 **Unbootable Operating System Recovery**
To recover if you create an unbootable operating system after attempting a system reconfiguration.

Procedure 6.3 **Display System Parameter Definitions**
To display current system parameter definitions.

Procedure 6.1: Reconfigure the System

Purpose	To make a new **/unix**. To incorporate tunable changes resulting from hardware and software changes to the system. To tune system performance for your application.
When Performed	Only when the system must be tuned.
Starting Conditions	System state—multiuser. Login—**root**.
sysadm Menu	MACHINE MANAGEMENT
Commands	**/etc/mkboot**(1M) **sysadm firmware**(1) **sysadm reboot**(1) **/etc/mkunix**(1M)
Cautions	1. Always copy existing **/unix** to **/oldunix**. 2. Execute a separate **mkboot** for each modified driver or module. 3. Do not change **nodename** during reconfiguration without coordinating it with interfacing systems.
References	Chapter 6, "Performance Management of Part 2." "Run Firmware Programs" in Chapter 3 of Part 2.

System reconfiguration is necessary whenever the physical configuration of the computer or the software configuration of the operating system itself changes. This happens when you upgrade your hardware (such as add more memory or disks), or add a software driver, or when you edit tunable parameters to improve performance. The only way the system can understand such changes is by rebuilding the system from its (partly modified) source files. This procedure allows you to modify and reconfigure (rebuild) the system residing in **/unix**.

As shown in the table at the beginning of this procedure, make sure of these three things:

1. Always copy the existing bootable **/unix** to **/oldunix** so that you will have a bootable operating system if you create an unbootable **/unix**. If you create an unbootable **/unix** and do not have a bootable version of the operating system on the hard disk, you will have to do a partial restore using the Operating System Utilities cartridge tape.
2. For each driver or module that you modified in **/etc/master.d**, execute a separate **mkboot** command. If you execute **mkboot** with more than one module or driver specified, an invalid boot file may be created.
3. When reconfiguring the operating system, do not arbitrarily change the node name (NODE) of the 3B2 computer. Once Basic Networking Utilities has been established, a change in node name must be coordinated with all interfacing systems. If not properly coordinated, a calling system will fail to make a connection because the returned name does not match the name stored in **/usr/lib/uucp/Systems**.

There are two major steps in reconfiguring the operating system: (1) edit the **/etc/master.d** files to change the tunable parameters, and (2) rebuild the operating system.

Change the Tunable Parameters

The major steps of incorporating changes to the tunable parameters are as follows:

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

Step 2: Copy the existing **/unix** to **/oldunix** (see Caution 1).

```
cp /unix /oldunix
```

Step 3: Change the present working directory to **/etc/master.d**.

```
cd /etc/master.d
```

Step 4: Edit the applicable files in the **/etc/master.d** directory to change (increasing or decreasing the value of) the tunable parameters. Refer to Figure 6-5 in Chapter 6, "Performance Management," for a complete listing of recommended values for the tunable parameters available with your system.

Rebuild the Operating System

The major steps in rebuilding the operating system are as follows (note you must be logged in as **root**):

Step 5: Change the present working directory to **/boot**.

```
cd /boot
```

Step 6: Execute the **/etc/mkboot** command to create a bootable object file for each of the files modified in **/etc/master.d** (see Caution 2). For example, if some of the tunable parameters in the **/etc/master.d/kernel** were modified, you would enter:

```
/etc/mkboot -k KERNEL
```

If the tunables that you changed were in another file (for example, **/etc/master.d/sem**), you do not need the **-k** option.

```
/etc/mkboot SEM
```

Step 7: Take the system to the firmware mode and boot **/etc/system**. This step creates a new **/unix**.

```
# sysadm firmware
.
.
.
SELF CHECK
FIRMWARE MODE
        (Enter the firmware password)
Enter name of program to execute [   ]: /etc/system
```

A new **/unix** will be generated automatically during the boot procedure.

Procedure 6.2: Unbootable Operating System Recovery

Purpose	To recover from an unbootable **/unix**. To get a good version of the system running after an unsuccessful attempt at reconfiguring the system.
Starting Conditions	System state—variable. Login—**root**.
sysadm Menu	MACHINE MANAGEMENT
Commands	**sysadm firmware**(1)—to get to firmware
Bootable Programs	**/oldunix**—boot program (from hard disk)
References	Chapter 6, "Performance Management."

If you create a **/unix** that is unbootable or is operating so poorly that recovery while operating in that version is impossible, return the system to firmware mode by doing one of the following:

1. If the system comes up to the point where you get the `Console login:` prompt, log in as root and take the system to firmware mode.

   sysadm firmware

2. If the system fails to come up, the system is unable to automatically reboot the new UNIX operating oystem. You may then get the following message displayed:

   `SYSTEM FAILURE:`

   This message shows that your system is already in firmware state (operating system is not running).

3. If you do not have communications with the system and the `SYSTEM FAILURE:` message was not displayed, press the power switch to the STANDBY position. After the hard disks stop spinning, press the

power switch to the ON position. At the point where the word DIAGNOSTICS appears, press the reset switch (described in the *Read Me First* document). This takes the system directly to firmware mode.

Once you are in firmware, you may recover the system by doing the following:

Step 1: Enter the firmware password. When the following prompt appears, enter **/oldunix**:

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

The system boots the **/oldunix** that you made before reconfiguring the system.

Note: If you did not make this copy, reload the operating system using the Operating System Utilities cartridge tape via the Partial Restore procedure (see Procedure 3.9, "Reload the Operating System").

Step 2: After the system has rebooted and you have logged in again, move **/oldunix** back to **/unix**:

```
mv /oldunix /unix
```

This will protect you from having to do this procedure again if the system crashes.

Step 3: Finally, try to determine what went wrong. Consider first that you may have placed an incorrect value in the parameter(s) with which you were working.

Step 4: Repeat Procedure 6.1, "Reconfigure the System," to correct errors in the previous reconfiguration. If you cannot determine what went wrong, document what happened as thoroughly as you can, and then contact your AT&T Service Representative or authorized dealer.

Procedure 6.3: Display System Parameter Definitions

Purpose	To display current system parameter definitions.
Starting Conditions	System state—multiuser or single user. Login—**root**.
Commands	**/etc/sysdef**(1M) **[** *system_namelist* **] [** *master.d* **]**
References	Chapter 6, "Performance Management."

The **/etc/sysdef**(1M) command is used to display the current system parameter values for the operating system specified by *system_namelist* associated with the parameters defined in the specified */etc/master.d* directory. The default *name_list* is **/unix** and the default *master.d* is the **/etc/master.d** directory. The following display shows an example of the output of this command for the default arguments.

```
# sysdef
*
* 3B2 Configuration
*

  Boot program: /boot/KERNEL
  Time stamp:   Tue May  3 12:57:35 1989
*
* Devices
*
  scsi  board=1
  eports        board=2
  eports        board=3
  eports        board=4
  conlog
  gentty
  hdelog
  idisk
  iuart
  mem
  osm
  sd01
  st01
  sxt
  xt
  mpb
  osm
  mirror
  prf
  clone
  log
  sp
*
* Loadable Objects
*
  dufst
  du
  shm
  sem
  msg
```

Continued

Procedure 6.3: Display System Parameter Definitions

Continued from previous screen display

```
  ipc
  vcache
  s5
  pdi_
  disp
*
* System Configuration
*
  rootdev         sd01(121)       minor=0
  swapdev         sd01(121)       minor=1 swplo=0 nswap=20640
  pipedev         sd01(121)       minor=0
*
* Tunable Parameters
*
  1100  buffers in buffer cache (NBUF)
    60  entries in callout table (NCALL)
  1300  inodes  (NINODE)
  1300  s5inodes  (NS5INODE)
  1300  entries in file table (NFILE)
    25  entries in mount table (NMOUNT)
   400  entries in proc table (NPROC)
  1200  entries in shared region table (NREGION)
   604  clist buffers (NCLIST)
    30  processes per user id (MAXUP)
  1024  hash slots for buffer cache (NHBUF)
    20  buffers for physical I/O (NPBUF)
   150  size of system virtual space map (SPTMAP)
    16  fraction of memory for vhandlow (VHNDFRAC)
     0  maximum physical memory to use (MAXPMEM)
    45  auto update time limit in seconds (NAUTOUP)
    20  maximum number of open files per process (NOFILES)
   432  number of streams queues (NQUEUE)
    36  number of streams head structures (NSTREAM)
     0  number of 4096 bytes stream buffers (NBLK4096)
    28  number of 2048 bytes stream buffers (NBLK2048)
    20  number of 1024 bytes stream buffers (NBLK1024)
    16  number of 512 bytes stream buffers (NBLK512)
    32  number of 256 bytes stream buffers (NBLK256)
    80  number of 128 bytes stream buffers (NBLK128)
   512  number of 64 bytes stream buffers (NBLK64)
   384  number of 16 bytes stream buffers (NBLK16)
```

Continued

Continued from previous screen display

```
   384  number of 4 bytes stream buffers (NBLK4)
  8192  maximum size of user's virtual address space in pages (MAXUMEM)
  8192  for package compatibility equal to MAXUMEM (MAXMEM)
    25  page stealing low water mark (GPGSLO)
    40  page stealing high water mark (GPGSHI)
     1  vhand wake up rate (VHANDR)
   393  awaken vhand if free memory less than vhandl (VHANDL)
    10  bdflush run rate (BDFLUSHR)
    40  minimum resident memory for avoiding deadlock (MINARMEM)
    40  minimum swapable memory for avoiding deadlock (MINASMEM)
     1  maximum number of pages swapped out (MAXSC)
     1  maximum number of pages saved (MAXFC)
*
* Utsname Tunables
*
   3.2.2  release (REL)
    unix  node name (NODE)
    unix  system name (SYS)
       3  version (VER)
*
* Streams Tunables
*
    32  number of multiplexor links (NMUXLINK)
     9  maximum number of pushes allowed (NSTRPUSH)
   288  initial number of stream event calls (NSTREVENT)
     1  page limit for event cell allocation (MAXSEPGCNT)
  4096  maximum stream message size (STRMSGSZ)
  1024  max size of ctl part of message (STRCTLSZ)
    80  max low priority block usage (STRLOFRAC)
    90  max medium priority block usage (STRMEDFRAC)
```

Continued

Procedure 6.3: Display System Parameter Definitions

Continued from previous screen display

```
*
* RFS Tunables
*
    25  entries in advertise table (NADVERTISE)
   150  receive descriptors (NRCVD)
   250  maximum number of rd_user structures (NRDUSER)
   100  send descriptors (NSNDD)
     3  minimum number of server processes (MINSERV)
     6  maximum number of server processes (MAXSERV)
    24  maximum number of remote systems with active mounts (MAXGDP)
  3072  size of static RFS administrative storage area (RFHEAP)
     1  latest compatible RFS version (RFS_VHIGH)
     1  earliest compatible RFS version (RFS_VLOW)
    50  entries in server mount table (NSRMOUNT)
    10  max interval for turning off RFS caching (RCACHE_TIME)
     0  minimum number of RFS buffers (NREMOTE)
     0  minimum number of local buffers (NLOCAL)
*
* IPC Messages
*
   100  entries in msg map (MSGMAP)
  2048  max message size (MSGMAX)
  4096  max bytes on queue (MSGMNB)
    50  message queue identifiers (MSGMNI)
     8  message segment size (MSGSSZ)
    40  system message headers (MSGTQL)
  1024  message segments (MSGSEG)
*
* IPC Semaphores
*
    10  entries in semaphore map (SEMMAP)
    10  semaphore identifiers (SEMMNI)
    60  semaphores in system (SEMMNS)
    30  undo structures in system (SEMMNU)
    25  max semaphores per id (SEMMSL)
```

Continued

Continued from previous screen display

```
    10  max operations per semop call (SEMOPM)
    10  max undo entries per process (SEMUME)
 32767  semaphore maximum value (SEMVMX)
 16384  adjust on exit max value (SEMAEM)
*
* IPC Shared Memory
*
131072  max shared memory segment size (SHMMAX)
     1  min shared memory segment size (SHMMIN)
   100  shared memory identifiers (SHMMNI)
     6  max attached shm segments per process (SHMSEG)
   512  max in use shared memory (SHMALL)
*
* File and Record Locking
*
   100  records configured on system (FLCKREC)
#
```

LP Spooling Administration Procedures

LP Spooling Administration Procedures

This chapter briefly describes the procedures needed to administer the Line Printer (LP) print service provided by the LP Spooling Utilities. The LP Spooling Utilities are installed with the System Administration menus using the **sysadm** command. If more detailed information is needed, see Chapter 7, "LP Spooling Administration." This section provides the following procedures:

Procedure 7.1 **Install the LP Spooling Utilities**
Instructions for installing the LP Spooling Utilities and references to documents that can help you install printers and other hardware.

Procedure 7.2 **Stop the LP Print Service**
Instructions for stopping the LP print service.

Procedure 7.3 **Restart the LP Print Service**
Instructions for restarting the LP print service.

Procedure 7.4 **Set Up the LP Print Service**
Instructions for adding or deleting a printer from the current configuration, or for changing the configuration of a printer.

Procedure 7.5 **Set Up Forms**
Instructions for adding, changing, or deleting a form from the LP print service.

Procedure 7.6 **Set Up Filters**
Instructions for adding, changing, or deleting a filter from the LP print service.

Procedure 7.1: Install the LP Spooling Utilities

Purpose	To install the LP Spooling Utilities.
Starting Conditions	Multiuser or single-user state. (If you are in single-user state, you must run the command **mount /usr** before following this procedure.) Login—**root**.
References	Installation *Owner/Operator Manual* —Chapter 5, "Software Utilities Packages." Printer manuals —*AT&T 3B2 Computer Installation Manual for AT&T Printers* —*Dot Matrix Printer Manual* —*Letter Quality Printer Manual* Manuals for optional hardware —*Expanded Input/Output Capability Manual* —*Enhanced Ports Manual.*

Step 1: To set up your printer and any optional hardware you may have, follow the instructions in the appropriate documents (see "References" in table).

Step 2: To install LP Spooling Utilities, enter the following command:

```
$ sysadm tapepkg
```

The **tapepkg** command will prompt you to insert the cartridge tape (labeled "Operating System Utilities") into the cartridge tape drive at the appropriate time. You will also be prompted to remove the cartridge tape when appropriate.

Step 3: Specify the install option (**i**) and select the Terminal Information Utilities (items 7 and 8) and LP Spooling Utilities (item 10) for installation. Chapter 5 of the *Owner/Operator Manual* that came with your system contains the general procedure for installing optional utilities.

Procedure 7.2: Stop the LP Print Service

Purpose	To stop the LP print service.
Starting Conditions	Multiuser or single user state. (If you are in single-user state, you must run the command **mount /usr** before following this procedure.) Login—**root**.
Commands	**sysadm packagemgmt/lpmgmt/service/stop**
References	"Summary of Administrative Commands" in Chapter 7, "LP Spooling Administration."

Step 1: Enter the following command to stop the LP print service:

```
$ sysadm stop
```

This will stop the LP print service completely. You will be notified by a screen message when the printer has stopped; no further requests for printing will be acted on.

Procedure 7.3: Restart the LP Print Service

Purpose	To make the LP system available again after having been stopped.
Starting Conditions	Multiuser or single-user state. (If you are in single-user state, you must run the command **mount /usr** before following this procedure.) Login—**root**.
Commands	**sysadm packagemgmt/lpmgmt/service/start**
References	"Administrative Commands" in Chapter 7, "LP Spooling Administration."

After you have completed those administrative tasks for which you stopped the LP print service, restart the LP print service by entering the following command:

```
$ sysadm start
```

A screen message will notify you when the LP print service is available again.

Procedure 7.4: Set Up the LP Print Service

Purpose	To add or delete a printer from your system, or to change the configuration of a printer in your system.
Starting Conditions	Multiuser or single-user state. (If you are in single-user state, you must run the command **mount /usr** before following this procedure.) Login—**root**. The printer in question must be connected to an available port.
Commands	**sysadm packagemgmt/lpmgmt/printers/add_p** **sysadm packagemgmt/lpmgmt/printers/change_p** **sysadm packagemgmt/lpmgmt/printers/accept_p** **sysadm packagemgmt/lpmgmt/printers/enable_p** **sysadm packagemgmt/lpmgmt/service/default** **sysadm packagemgmt/lpmgmt/printers/reject_p** **sysadm packagemgmt/lpmgmt/printers/disable_p** **sysadm packagemgmt/lpmgmt/printers/delete_p**
References	"Summary of Administrative Commands" in Chapter 7, "LP Spooling Administration."

Add a Printer

Step 1: To add a printer, enter the following command:

```
$ sysadm printers/add_p
```

If other printers have already been added to the system, you will be asked if the printer you are now adding is similar to any of them. If it is, then the configuration of the similar printer will be used as a template for the configuration of the new printer. If not, standard defaults will be given in a template. In either case, the system will prompt you to define the configuration for the new printer, as shown in the following example.

If you defined the alert-type for the printer to be **mail** or **write** and did not specify a login, the login **sysadm** will be assumed. In addition, the alert-type **quiet** can only be used to terminate an active alert and, therefore, cannot be used as the alert-type for a new printer.

Note: In the following example, it is assumed that the printer is connected to a port on the Enhanced Ports (EPORTS) card. For more information on how to identify ports, see Chapter 7, "LP Spooling Administration."

```
Enter the name of the new printer:  printer2

Printer class to add it to:  (default: none)  letterquality
Enter one line that describes the printer for your users.
(default:  none)  AT&T Model 455 daisy-wheel printer

Type of printer: (default:  unknown)  455

Is the printer an AT&T 455; qume; sprint 11?
(default: yes) [y, n, ?, q] <CR>
Types of files printable without filtering:
(default:simple) simple, nroff

Printer interface to use: (default:  standard)  /usr/kol/interface
Can a user skip the banner page?  (default:  no) [y, n, ?, q] y

Use the default page size and printing pitch?  (default:  yes) [y, n, ?, q] <CR>

Enter a command to run to alert you when the printer faults.
(default:  mail lp)  write
How often should you be alerted (minutes, 0 = once)?  (default:  0)  3
How should printing restart after you fix a printer fault?
(default:  continue) <CR>

Is the printer directly connected?  (default:  yes) [y, n, ?, q] <CR>
Printer port to use:  /dev/tty11
Is the printer also a login terminal?  (default:  no) [y, n, ?, q] <CR>

The default port settings will be:
        9600
        cs8 cstopb -parenb -parodd
        ixon -ixany
        opost -olcuc onlcr -ocrnl -onocr -onlret
        -ofill nl0 cr0 tab0 bs0 vt0 ff0

Enter any changes or additional settings that must be in
effect: <CR>
```

Continued

```
Continued from previous screen display
List the print wheels you have available for this printer:
(default: none) elite, courier
List the users who are denied access to this printer.
(default:  none) <CR>
List the users who are allowed access to this printer.
(default:  all)  merlyn ehr3 glg
List the forms that can be used with this printer.
(default:  none) payroll_check, order_form
List the forms that can't be used with this printer.
(default:  all) <CR>

Install, edit, or skip this printer entry:  (default:  install) <CR>
#
```

If, while setting up the LP print service, you have trouble defining any of the fields for which you must supply values, be sure to read the help messages and look for further details in Chapter 7, "LP Spooling Administration." Also, see "How to Define Printer Ports and Printer Port Characteristics" in Chapter 7.

When you have finished this session, the printer you have defined will be added to the LP print service for your machine. Next, you must specify certain operational parameters for this printer.

Step 2: To tell the LP print service to accept requests for print jobs on the printer you are adding, enter the following command:

```
$ sysadm printers/accept_p
```

You will be asked to list the printers or printer classes that can start accepting print requests.

Step 3: When you are ready to start printing, be sure that the printer is ready to receive output. For several printers, this means that the top of the form has been adjusted and that the printer is on-line. To enable printing to occur on the printer, enter the following command:

```
$ sysadm printers/enable_p
```

You will be asked to list the printers that can start printing requests.

Step 4: (This step is optional.) Set the destination of print requests to a default printer by entering the following command:

```
$ sysadm default
```

You will be asked to name a printer or printer class to be used as the default destination for print requests.

Step 5: You can verify that the new printer has been added according to your specifications and that your final configuration is correct by entering the following command:

```
$ sysadm printers/list_p
```

The command will prompt you for the names of printers for which you want to see a complete configuration; it will give you a configuration listing of those printers. If you specify **all**, you will receive a configuration listing of all available printers, including the one you have just added.

Change the Configuration of an LP Printer

If you change your system by changing a printer configuration or by deleting a printer, two things may change as a result: (1) the appearance of the text of printed files, and (2) the appropriateness of the changed or deleted printer for jobs already in the queue. To safeguard requests for printing, consider taking the following preliminary actions for a printer that you are going to drop from the system or for which you are going to change the configuration: (1) stop the printer from taking requests for printing, and/or (2) disable the printer.

Once you are satisfied that no jobs are in danger of being ruined and no new jobs will be queued for the affected printer, you are ready to add or delete a printer to your system, or to change the configuration of an existing printer. The procedures for these precautionary measures are given in Steps 1 and 2.

Step 1: (This step is optional.) To safeguard against losing print jobs requested from a printer that you are reconfiguring, you may want to stop that printer from accepting print requests. Enter the following command:

```
$ sysadm printers/reject_p
```

The **printers** menu will ask you to specify targeted printers or printer classes and to provide a reason for turning off their ability to accept requests.

Step 2: (This step is optional.) Disable your printer by entering the following command:

```
$ sysadm printers/disable_p
```

The **printers** menu then gives you the option of preserving or deleting current print requests before the specified printer is disabled.

```
Enter the names of the printers that should stop printing requests.
Separate multiple names with a space or comma.
(default:  printer2) <CR>
Cancel any requests currently printing?  (default:  no)  [y, n, ?, q] <CR>
Wait for any requests currently printing?  (default:  no) [y, n, ?, q] y
Reason for disabling:
(default:  none)  to reconfigure printer2
```

The reason that the default printer listed in Step 2 is printer2 is because it was the last printer added to the system with the **add** command from the **printers** menu. If no printers have been added in this way, the default value is "none."

Step 3: Now you are ready to change the configuration of the appropriate printer. Enter the following command:

```
$ sysadm printers/change_p
```

First, you will be prompted for the name of the printer. Then you will be asked to provide details about the new configuration, such as page dimensions.

If you defined the alert-type for the printer to be **mail** or **write** and did not specify a login, the login **sysadm** will be assumed. In addition, the alert-type **quiet** can only be used to terminate an active alert.

```
Enter the name of the printer:  (default:  printer2) <CR>
Printer class to add it to:  (default:  none) letterquality
Enter one line that describes the printer for your users.
(default:  AT&T Model 455 daisy-wheel printer) <CR>

Type of printer: (default:  455) <CR>

Is the printer an AT&T 455; qume; sprint 11?
(default: yes) [y, n, ?, q] <CR>
Types of files printable without filtering:
(default: simple,nroff) <CR>

Printer interface to use: (default: standard) <CR>
Can a user skip the banner page?  (default: yes) [y, n, ?, q] <CR>

Use the default page size and printing pitch?  (default: yes) [y, n, ?, q] <CR>

Enter a command to run to alert you when the printer faults.
(default:  write root) <CR>
How often should you be alerted (minutes, 0=once)?  (default: 3) <CR>
How should printing restart after you fix a printer fault?
(default: continue) <CR>

Is the printer directly connected?  (default: yes) [y, n, ?, q] <CR>
Printer port to use:  (default: /usr) <CR>
Is the printer also a login terminal?  (default: no) [y, n, ?, q] <CR>

The default port settings will be:
        9600
        cs8 cstopb -parenb -parodd
        ixon -ixany
        opost -olcuc onlcr -ocrnl -onocr -onlret
        -ofill nl0 cr0 tab0 bs0 vt0 ff0

Enter any changes or additional settings that must be in
effect: <CR>
```

Continued

```
Continued from previous screen display
List the print wheels you have available for this printer:
(default: elite,courier) <CR>

List the users who are denied access to this printer.
(default:  none) <CR>
List the only users who are allowed access to this printer.
(default: merlyn,ehr3,glg) <CR>
List the forms that can be used with this printer.
(default:  none) <CR>
List the only forms that can't be used with this printer.
(default:  all) <CR>

Install, edit, or skip this printer entry:  (default:  install) <CR>
#
```

Step 4: Ensure that only **"lp"** can write to the device file. This avoids unwanted output from non-LP processes. Enter the following commands:

```
$ chown lp /dev/ttyxx
$ chmod 600 /dev/ttyxx
```

xx is the port number to which the printer is connected.

Step 5: "Authorize" the newly configured printer to accept requests for print jobs by entering the following command:

```
$ sysadm printers/accept_p
```

Step 6: Reenable the newly configured printer to print by entering the following command:

```
$ sysadm printers/enable_p
```

The menu will prompt you to provide the names of the printers that can start printing requests.

Step 7: (This step is optional.) To define a default destination for print requests, enter the following command:

```
$ sysadm default
```

You will be prompted for the name of the printer or printer class to be used as the default destination for print requests.

Step 8: The reconfiguration of your printer is now complete. If you want to verify that your final configuration is correct, enter the following command:

```
$ sysadm printers/list_p
```

The command will prompt you for the names of printers for which you want to see a complete configuration; it will give you a configuration listing of those printers. If you specify **all**, you will receive a configuration listing of all available printers, including the one you have just changed.

Delete a Printer

Step 1: Enter the following command to delete a printer:

```
$ sysadm printers/delete_p
```

You will be asked to name the printer to be deleted. The command will also give you an opportunity to reconsider your request by asking, immediately, for verification of it. The message `Are you sure?` will be displayed.

Step 2: To verify that the appropriate printer has been deleted, enter the following command:

```
$ sysadm printers/list_p
```

The command will prompt you for the names of printers for which you want to see a complete configuration; it will give you a configuration listing of those printers. If you specify **all**, you will receive a configuration listing of all available printers, minus the one you have just deleted.

Procedure 7.5: Set Up Forms

Purpose	To add, change, or delete a form from the LP print service.
Starting Conditions	Multiuser or single-user state. (If you are in single-user state, you must run the command **mount /usr** before following this procedure.) Login—**root**.
Commands	**sysadm packagemgmt/lpmgmt/forms/add_f** **sysadm packagemgmt/lpmgmt/forms/change_f** **sysadm packagemgmt/lpmgmt/forms/delete_f** **sysadm packagemgmt/lpmgmt/forms/mount_f** **sysadm packagemgmt/lpmgmt/forms/list_f**
References	"Summary of Administrative Commands" in Chapter 7, "LP Spooling Administration."

About Using Forms

Your printer can print on a variety of pre-printed forms, such as checks and invoices. The LP print service can help you take advantage of this capability, but you must supply additional software, as well as the forms. Once you have all the necessary components in place, the LP print service can help you use your forms and the software you have to support them in the following ways:

- The LP print service can schedule the work of one or more printers, allowing you to assign the task of printing particular forms to various printers at particular time periods. (If you have only one printer, the scheduling service allows you to allocate its time among multiple forms.)
- The LP print service can keep track of forms currently mounted and alert you to mount forms as needed.

The various steps involved in setting up a system and using it to print on pre-printed forms are shown in the following procedures. This procedure leads you through a sample case, showing the steps taken by a company that wants to automate the printing of payroll checks for its employees. (The payroll data base mentioned in Step 1, which contains the information that will appear on the checks, such as the payee's name and the amount, is assumed to exist on the company's computer.)

Step 1: The company buys (or asks a programmer to write) application software (a program) that takes information from the payroll data base to print a standard check.

Step 2: The System Administrator installs the application on the system.

Step 3: The System Administrator notifies users (in this case, a company clerk responsible for preparing payroll checks) that the application program is available to prepare the checks.

Step 4: The System Administrator uses the **sysadm** menu for LP forms to identify to the LP print service the available forms including checks, invoices, and time cards.

Step 5: The payroll clerk issues a request to the LP print service to print checks.

Step 6: The LP print service notifies the administrator of the payroll clerk's request and alerts the System Administrator to change the paper in the printer from the current stock to checks.

Step 7: The System Administrator loads checks into the printer.

Step 8: The printer prints the payroll checks.

As you can see, the LP print service does not provide any application software such as that described in Step 1 (a program that converts a file to a format suitable for a particular form, such as an invoice). As mentioned in Step 1, you can purchase application software separately or you can write your own applications.

Add a Form

Step 1: Enter the following command to add a form from the LP print service:

```
$ sysadm forms/add_f
```

You will be asked to name the desired form and provide specifications for it, as shown in the following screen.

```
Number of pages in the form: (default:  1)  2
Character set to use with the form:  (default:  any) <CR>
Ribbon color to use with the form:  (default:  any)  black
Enter a one-line description of this form for your users:
payroll check for non-supervisory staff
Enter the full pathname of a file containing an alignment pattern:
/usr/kol/align/align.check

Enter a command to run to alert you when the form needs mounting.
(default:  none)  write
How many print requests should be waiting before you are alerted?
(default:  1)  4
How often should you be alerted (minutes; 0=once)? <CR>
(default:  0)  3

List the users who are denied access to this form.
(default:  none) <CR>
List the only users who are allowed access to this form.
(default:  all)  merlyn ehr3 glg

Install, edit, or skip this form:  (default:  install) <CR>
```

Step 2: (This step is optional.) If you would like to mount this form, enter the following command:

```
$ sysadm forms/mount_f
```

You will be prompted to answer the following questions:

```
Enter the name of the printer:  (default:  printer2)  printer3
Enter the name of the form you are mounting:  (default:  payroll) <CR>
Enter the name of the print wheel you are mounting:  (default:  none) <CR>
Print an alignment pattern?  (default: no)  yes

Press the return key to print a copy of the pattern: <CR>
```

You will see the third prompt (`Enter the name of the print wheel you are mounting:`) only if your printer can take print wheels. If it does, and you are prompted for the names of both a form and a print wheel, be sure that you assign a value other than `"none"` to one of them.

The default form is **payroll** because that was the last form added to the system with the **add** command from the **forms** menu. If the form can be mounted, and if an alignment pattern is to be printed, the prompt `Press return to print an alignment pattern [q to quit]:` will appear repeatedly until the user indicates that he or she has finished. Each time the return key is pressed, an alignment pattern will be printed on the printer named. (Enter **q** to stop printing the alignment pattern.)

Note: An alternative way to mount a new form is through the Printer Management Menu. The following command line is equivalent to the mount command: **sysadm mount_p**.

Step 3: To verify that the new form has been added, enter the following command:

```
$ sysadm forms/list_f
```

The command will prompt you to name the forms for which you want to see a complete description; it will give you a list of those forms. If you specify **all**, you will receive a list of all available forms, including the one you have just added.

Change a Form

Step 1: Enter the following command to change a form from the LP print service:

```
$ sysadm forms/change_f
```

You will be prompted for the name of the form you want to change. (By default, the form to be changed is considered the form you added last by selecting the menu item **add form**.) Then you will be asked, through a series of questions, to describe the form and list users who are or are not allowed to use it.

Notice that the default values for various specifications are the same as the values for the corresponding specifications of the last form added. See the section "Add a Form".

```
Number of pages in the form: (default:  2) <CR>
Character set to use with the form:  (default:  any) <CR>
Ribbon color to use with the form:  (default:  black) <CR>
Enter a one-line description of this form for your users:
(default:  payroll check for non-supervisory staff) <CR>
Enter the full pathname of a file containing an alignment pattern:
(default:  align.check) <CR>

Enter a command to run to alert you when the form needs mounting.
(default:  write root)  mail
How many print requests should be waiting before you are alerted?
(default:  4)  10
How often should you be alerted (minutes)?
(default:  33) <CR>

List the users who are denied access to this form.
(default:  none) <CR>
List the only users who are allowed access to this form.
(default:  merlyn ehr3 glg) <CR>

Install, edit, or skip this form:  (default:  install) <CR>
```

(These are the same questions you are asked when adding a new form.)

Step 2: Now the changes to your form are complete. If you want to verify that these changes are correct, enter the following command:

```
$ sysadm forms/list_f
```

The command will prompt you for the names of forms for which you want to see a complete description; it will then give you a list of those forms. If you specify **all**, you will receive a list of all available forms, including the one you have just changed.

Delete a Form

Step 1: Enter the following command to delete a form from the LP print service:

```
$ sysadm forms/delete_f
```

You will be asked to name the forms you want to remove. Then, before the forms are deleted, you will have a chance to reconsider your request. The message `Are you sure?` will be displayed.

Step 2: The appropriate form has now been deleted. If you want to verify this, enter the following command:

```
$ sysadm forms/list_f
```

The command will prompt you for the names of forms for which you want to see a complete description; it will then give you a list of those forms. If you specify **all**, you will receive a list of all available forms, minus the one you have just deleted.

Procedure 7.6: Set Up Filters

Purpose	To add, change, or delete a filter from the LP print service.
Starting Conditions	Multiuser or single user state (If you are in single-user state, you must run the command **mount /usr** before following this procedure.) Login—**root**.
Commands	**sysadm packagemgmt/lpmgmt/filters/add_f** **sysadm packagemgmt/lpmgmt/filters/change_f** **sysadm packagemgmt/lpmgmt/filters/delete_f** **sysadm packagemgmt/lpmgmt/filters/list_f**
References	"Summary of Administrative Commands" in Chapter 7, "LP Spooling Administration."

About Using Filters

In addition to allowing you to print a variety of documents, the LP print service enables you to have data that appears in your input file in one format printed on paper in another format. The device used by the LP print service to transform formats is a program called a filter. You must provide any filters you want to use on your system (either by buying or writing them); the LP print service does not provide filters.

The LP print service helps you manage filters that you have already installed on your system. Specifically, it oversees the use of filters, checking to find out when they are needed for print jobs, and then matching the appropriate filter with a user's file and a printer.

The following procedure is a sample scenario of how filters may be used with the help of the LP print service.

Step 1: A company identifies the filters it needs, based on the types of files it typically prints.

Step 2: The company buys or asks programmers to write those filters.

Step 3: The System Administrator adds the filters to the filter table.

Step 4: Users issue print requests, specifying the type of file and the printer.

Step 5: A filter is selected by the LP print service.

Step 6: The filter transforms the data into the desired format.

Step 7: The printer prints the document.

Add a Filter

Step 1: Enter the following command to add a filter from the LP print service:

```
$ sysadm filters/add_f
```

The system will prompt you for the name of the desired filter. Then it will present a series of questions through which you can submit specifications for it.

```
Filter name: 450
Input types it can convert:  (default: any)  nroff
Output types it can produce:  (default: any )  450
Printer types it is restricted to:  (default: any)  455
Printers it is restricted to:  (default:  any) <CR>
Is this a slow filter?  (default:  yes) [y, n, ?, q] <CR>
Enter the filter command and any fixed options:  450
        .
        .
        .
        .
Keyword:  MODES
Pattern:  landscape
Template:  -l

Keyword:  done

Install, edit, or skip this filter:  (default:  install) <CR>
```

Step 2: To verify that the new filter has been added, enter the following command:

```
$ sysadm filters/list_f
```

The system will prompt you for a list of the filters for which you want to see a complete description; it will then give you a list of those filters. If you specify **all**, you will receive a list of all available filters, including the one you have just added.

Change a Filter

Step 1: Enter the following command to change a filter from the LP print service:

```
$ sysadm filters/change_f
```

The system will prompt you for the name of the filter you want to change.

If the filter was delivered with the LP print service, you will then be asked whether you want to restore the filter to the "factory setting." If you answer no, the system will then ask you, through a series of questions, to provide a new specification for the filter.

In the following example, we will change the specification for the filter we added in the section "Add a Filter". (Because we are not dealing with a filter that was delivered with the LP print service, we are not given the choice of restoring the filter to the "factory setting.") We will make two changes: We will add another type of printer to which the filter will be restricted (qume), and we will specify that this filter is not to be a "slow" filter (as specified by default).

```
Filter name:  (default: 450) <CR>
Input types it can convert:  (default: nroff) <CR>
Output types it can produce:  (default:  450) <CR>
Printer types it is restricted to:  (default:  455)  455, qume
Printers it is restricted to:  (default:  any) <CR>
Is this a slow filter?  (default: yes)  [y, n, ?, q] no
Enter the filter command and any fixed options:
(default:  450) <CR>
        .
        .
        .
        .
Keyword:  (default:  MODES) <CR>
Pattern:  (default:  landscape) <CR>
Template:  (default:  -l) <CR>

Keyword:   done

Install, edit, or skip this filter:  (default:  install) <CR>
```

Step 2: To verify that the appropriate filter has been changed, enter the following command:

```
$ sysadm filters/list_f
```

The command will prompt you for the names of filters for which you want to see a complete description; it will then give you a list of those filters. If you specify **all**, you will receive a list of all available filters, including the one you have just changed.

Delete a Filter

Step 1: Enter the following command to delete a filter from the LP print service:

```
$ sysadm filters/delete_f
```

The system will then prompt you for the name of the filter you want to remove and give you an opportunity to reconsider your request. The message **Are you sure?** will be displayed.

Step 2: To verify that the appropriate filter has been deleted, enter the following command:

```
$ sysadm filters/list_f
```

The command will prompt you for the names of filters for which you want to see a complete description; it will then give you a list of those filters. If you specify **all**, you will receive a list of all available filters, minus the one you have just deleted.

TTY Management Procedures

TTY Management Procedures

Procedure 8.1: Check TTY Line Settings

Purpose	To tell what line settings are defined.
Starting Conditions	System state—multiuser or single user. You must mount **/usr** to run this procedure in single-user mode. Login—an authorized login.
sysadm Menu	TTY MANAGEMENT
Command	**sysadm lineset**(1)
References	"How the TTY System Works" in Chapter 8, "TTY Management."

Step 1: Enter the following command to go directly to the **lineset** display:

sysadm lineset

Step 2: The following display appears on your terminal.

```
Running subcommand 'lineset' from menu 'ttymgmt',
TTY MANAGEMENT


TTY Line Settings and Sequences

console1  console2  console3  console4  console5  console6
contty1   contty2   contty3   contty4   contty5   contty
contty1H  contty2H  contty3H  contty4H  contty5H  conttyH
pty   (does not sequence)
300     38400    19200     9600      4800      2400      1200
300H    4800H    9600H     19200H    38400H    2400H     1200H
```

Each of the line settings is just a name used to identify set of tty line characteristics. During the **login** process, the line settings on one line "hunt" from left to right, moving from one setting to the next when receiving a **BREAK** signal. The rightmost setting on each line hunts to the first one again, forming a circular hunt sequence.

Note that the "pty" setting does not sequence. Sending a **BREAK** will not make it change in any way.

Step 3: The following displays a line setting in detail.

```
Select one line setting to see it in detail [?, q]: 1200

Line Setting:              1200
       Initial Flags:      B1200 HUPCL
       Final Flags:        B1200 SANE IXANY TAB3 HUPCL
       Login Prompt:       login:
       Next Setting:       300

B1200       1200 Baud
HUPCL       Hang Up on Last Close
IXANY       Enable Any Character to Restart Output
SANE        Set All Modes To "Traditionally Reasonable" Values
TAB3        Expand Horizontal-tab To Spaces
```

Note: Any entry can be specified.

```
Select another line setting or
<RETURN> to see the original list [?, q]: 300

Line Setting:               300
        Initial Flags:      B300 HUPCL
        Final Flags:        B300 SANE IXANY TAB3 HUPCL
        Login Prompt:       login:
        Next Setting:       19200

B300          300 Baud
HUPCL         Hang Up on Last Close
IXANY         Enable Any Character to Restart Output
SANE          Set All Modes To "Traditionally Reasonable" Values
TAB3          Expand Horizontal-tab To Spaces

Select another line setting or
<RETURN> to see the original list [?, q]: q

Press the RETURN key to see the ttymgmt menu [?, q]: q
```

Procedure 8.2: Make TTY Line Settings

Purpose	To create new line settings and hunt sequences.
Starting Conditions	System state—multiuser or single user. You must mount **/usr** to run this procedure in single-user mode. Login—an authorized user.
sysadm Menu	TTY MANAGEMENT
Commands	**sysadm mklineset**(1)
References	"How to Create New Line Settings and Hunt Sequences" in Chapter 8, "TTY Management."

In this procedure, we are undertaking to connect a dual-speed modem to the computer to handle 300 baud and 1200 baud. There is a 1200 setting already in the table, but if the users somehow miss the speed, they will have to press the BREAK key several times to hunt for 1200 again. Since we are interested in only the two speeds, let's create a new 1200-300 sequence.

Step 1: Enter the following command to go directly to the **mklineset** display:

sysadm mklineset

Step 2: The following sequence of prompts appears on your terminal.

Procedure 8.2: Make TTY Line Settings

```
Running subcommand 'mklineset' from menu 'ttymgmt',
TTY MANAGEMENT

Enter the name of the new tty line setting [?, q]: modemfast
Select a baud rate [?, q]: ?  (To ask for HELP)

Available baud rates:
    50      134     200      600     1800     4800     19200
    75      150     300      1200    2400     9600     38400
    110

Select a baud rate [?, q]: 1200

Enter the login prompt you want (default = "login: ") [?, q]: <CR>

      (Accepting the default)

Do you want to add another tty line setting to the sequence? [y, n, q] y

Enter the name of the new tty line setting [?, q]: modemslow
Select a baud rate [?, q]: 300
Enter the login prompt you want (default = "login: ") [?, q]: <CR>
Do you want to add another tty line setting to the sequence? [y, n, q] n

Here is the tty line setting sequence you created:

modemfast modemslow
Line Setting:          modemfast
    Initial Flags:     B1200 HUPCL
    Final Flags:       B1200 SANE IXANY HUPCL TAB3
    Login Prompt:      login:
    Next Setting:      modemslow
Line Setting:          modemslow
    Initial Flags:     B300 HUPCL
    Final Flags:       B300 SANE IXANY HUPCL TAB3
    Login Prompt:      login:
    Next Setting:      modemfast
B1200     1200 Baud
B 300     300 Baud
HUPCL     Hang Up on Last Close
IXANY     Enable Any Character to Restart Output
SANE      Set All Modes To "Traditionally Reasonable" Values
TAB3      Expand Horizontal-tab To Spaces
Do you want to install this sequence? [y, n, q] y
Installed.

Press the RETURN key to see the ttymgmt menu [?, q]: q
```

Procedure 8.3: Modify TTY Line Characteristics

Purpose	To modify TTY line settings or turn line on or off.
Starting Conditions	System state—multiuser or single user. You must mount **/usr** to run this procedure in single-user mode. Login—an authorized login user.
sysadm Menu	TTY MANAGEMENT
Commands	**sysadm modtty**(1)
References	"How to Modify TTY Line Characteristics" in Chapter 8, "TTY Management."

The objective here is to tell the computer the port to use with the line settings defined in Procedure 8.2, "Make TTY Line Settings."

Step 1: Enter this command to go directly to the **modtty** display:

sysadm modtty

Step 2: The following sequence of prompts appears on your terminal.

Procedure 8.3: Modify TTY Line Characteristics

```
Running subcommand 'modtty' from menu 'ttymgmt',
TTY MANAGEMENT

Changeable tty lines:
     contty     tty26      tty34      tty42      tty47      tty54
     tty21      tty27      tty35      tty43      tty48      tty55
     tty22      tty28      tty36      tty44      tty51      tty56
     tty23      tty31      tty37      tty45      tty52      tty57
     tty24      tty32      tty38      tty46      tty53      tty58
     tty25      tty33      tty41
Select the tty you wish to modify,
or enter ALL to see a report of all ttys [?, q]: ALL

Changeable tty lines:
                 Hangup    Line
Tty      State   Delay     Setting     Description
---      -----   -----     -------     -----------
contty   on       60        4800
tty21    off      off       9600      38400 baud rate is available
tty22    off      off       9600      38400 baud rate is available
tty23    off      off       9600      38400 baud rate is available
tty24    off      off       9600      38400 baud rate is available
tty25    off      off       9600      38400 baud rate is available
tty26    off      off       9600      38400 baud rate is available
tty27    off      off       9600      38400 baud rate is available
tty28    off      off       9600      38400 baud rate is available
tty31    off      off       9600      38400 baud rate is available
tty32    off      off       9600      38400 baud rate is available
tty33    off      off       9600      38400 baud rate is available
tty34    off      off       9600      38400 baud rate is available
tty35    off      off       9600      38400 baud rate is available
tty36    off      off       9600      38400 baud rate is available
tty37    off      off       9600      38400 baud rate is available
tty38    off      off       9600      38400 baud rate is available
tty41    off      off       9600      38400 baud rate is available
tty42    off      off       9600      38400 baud rate is available
tty43    off      off       9600      38400 baud rate is available
tty44    off      off       9600      38400 baud rate is available
tty45    off      off       9600      38400 baud rate is available
tty46    off      off       9600      38400 baud rate is available
tty47    off      off       9600      38400 baud rate is available
tty48    off      off       9600      38400 baud rate is available
```

Continued

Continued from previous screen display

```
tty51    off       off       9600      38400 baud rate is available
tty52    off       off       9600      38400 baud rate is available
tty53    off       off       9600      38400 baud rate is available
tty54    off       off       9600      38400 baud rate is available
tty55    off       off       9600      38400 baud rate is available
tty56    off       off       9600      38400 baud rate is available
tty57    off       off       9600      38400 baud rate is available
tty58    off       off       9600      38400 baud rate is available

Continue (default: y)? [y, n, q] <CR>

Changeable tty lines:
     contty     tty26      tty34      tty42      tty47      tty54
     tty21      tty27      tty35      tty43      tty48      tty55
     tty22      tty28      tty36      tty44      tty51      tty56
     tty23      tty31      tty37      tty45      tty52      tty57
     tty24      tty32      tty38      tty46      tty53      tty58
     tty25      tty33      tty41
Select the tty you wish to modify,
or enter ALL to see a report of all ttys [?, q]: tty34
tty34:  current characteristics:
     State              off
     Hangup Delay       off
     Line Setting       9600
     Description

Available states:
     off      on
Select a state (default: off) [?, q]:  on

Enter a hangup delay, in seconds, or 'off' (default: off) [?, q]:  45

          (Because this is a dial-up line, we want to specify a timeout figure.)
Available line settings:
  console    console4 contty2  pty     1200H    4800H    19200    38400H
  console1   console5 contty3  300     2400     9600     19200H   modemfast
  console2   contty   contty4  300H    2400H    9600H    38400    modemslow
  console3   conttyl  contty5  1200    4800
Select a line setting (default: 9600) [?, q]:  modemfast
```

Continued

Procedure 8.3: Modify TTY Line Characteristics

Continued from previous screen display

```
Current description:

Enter a new description (default: current description) [?, q]:
     1200/300 baud dial in line

tty34: new characteristics:
     State          on
     Hangup Delay   45
     Line Setting   modemfast
     Description    1200/300 baud dial in line
Do you want to install these new characteristics? [y, n, q] y
tty34 now has new characteristics.

Changeable tty lines:
     contty     tty26      tty34      tty42      tty47      tty54
     tty21      tty27      tty35      tty43      tty48      tty55
     tty22      tty28      tty36      tty44      tty51      tty56
     tty23      tty31      tty37      tty45      tty52      tty57
     tty24      tty32      tty38      tty46      tty53      tty58
     tty25      tty33      tty41
Select the tty you wish to modify,
or enter ALL to see a report of all ttys [?, q]: ALL

Changeable tty lines:
                Hangup    Line
Tty     State   Delay     Setting     Description
---     -----   -----     -------     -----------
contty   on       60        4800
tty21    off      off       9600      38400 baud rate is available
tty22    off      off       9600      38400 baud rate is available
tty23    off      off       9600      38400 baud rate is available
tty24    off      off       9600      38400 baud rate is available
tty25    off      off       9600      38400 baud rate is available
tty26    off      off       9600      38400 baud rate is available
tty27    off      off       9600      38400 baud rate is available
tty28    off      off       9600      38400 baud rate is available
tty31    off      off       9600      38400 baud rate is available
tty32    off      off       9600      38400 baud rate is available
tty33    off      off       9600      38400 baud rate is available
tty34    on       45      modemfast   1200/300 baud dial in line
```

Continued

Continued from previous screen display

```
tty35     off       off       9600      38400 baud rate is available
tty36     off       off       9600      38400 baud rate is available
tty37     off       off       9600      38400 baud rate is available
tty38     off       off       9600      38400 baud rate is available
tty41     off       off       9600      38400 baud rate is available
tty42     off       off       9600      38400 baud rate is available
tty43     off       off       9600      38400 baud rate is available
tty44     off       off       9600      38400 baud rate is available
tty45     off       off       9600      38400 baud rate is available
tty46     off       off       9600      38400 baud rate is available
tty47     off       off       9600      38400 baud rate is available
tty48     off       off       9600      38400 baud rate is available
tty51     off       off       9600      38400 baud rate is available
tty52     off       off       9600      38400 baud rate is available
tty53     off       off       9600      38400 baud rate is available
tty54     off       off       9600      38400 baud rate is available
tty55     off       off       9600      38400 baud rate is available
tty56     off       off       9600      38400 baud rate is available
tty57     off       off       9600      38400 baud rate is available
tty58     off       off       9600      38400 baud rate is available

Continue (default: y)? [y, n, q] q
```

Basic Networking Procedures

Basic Networking Procedures

The following procedures are covered in this section:

Procedure 9.1 **Install Basic Networking Utilities (BNU) Software**
To place basic networking software on the hard disk.

Procedure 9.2 **Set Up Basic Networking Files**
To configure basic networking files.

Procedure 9.3 **Basic Networking Maintenance**
To maintain basic networking files and operations.

Procedure 9.4 **Basic Networking Debugging**
To track down problems in basic networking.

Procedure 9.5 **Remove BNU Software**
To remove basic networking software from the hard disk.

Procedure 9.6 **Set Up BNU STREAMS-Based Network (Basic)**
To show how BNU files are created for a STREAMS-based network.

Procedure 9.7 **Set Up BNU STREAMS-Based Network (Special)**
To show how BNU files are created for a STREAMS-based network if **cu** and **uucico** services are to be handled differently.

Procedure 9.1: Install Basic Networking Utilities Software

Purpose	Give a checklist of essential information needed to install the Basic Networking Utilities software on the hard disk.
References	Installation *Owner/Operator Manual* —Chapter 5, "Software Utilities Packages."

Chapter 5 of the *Owner/Operator Manual* that came with your system contains the installation procedure for the basic networking software.

Procedure 9.2: Set Up Basic Networking Files

Purpose	To configure basic networking files. To ensure proper communication links.
When Performed	Initial setup and when adding new devices or remote systems.
Starting Conditions	System state—multiuser or single user. You must mount **/usr** to run this procedure in single-user mode. Login—an authorized login user.
sysadm Menu	PACKAGE MANAGEMENT
Commands	**sysadm uucpmgmt**(1) **sysadm devicemgmt**(1) **sysadm portmgmt**(1) **sysadm systemmgmt**(1) **sysadm pollmgmt**(1)
References	*UNIX System V User's Guide.*

The procedure that follows provides instructions for setting up the Basic Networking facility and putting it into operation. This is done using **sysadm** subcommands and a text editor.

The following steps provide instructions on adding entries to three of the necessary support files: **Devices**, **Systems**, and **Permissions**. Instructions are also given to change existing entries in the **/etc/inittab** file for use with basic networking. Finally, the setup of several optional files is described.

Display the **uucpmgmt** System Administration submenu by entering:

```
# sysadm uucpmgmt

                        BASIC NETWORKING UTILITIES MANAGEMENT

  1 devicemgmt      manage devices (list, add, delete)
  2 pollmgmt        manage poll entries (list, add, delete)
  3 portmgmt        manage I/O ports (list, modify)
  4 systemmgmt      manage remote systems entries (list, add, delete, call)

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  q to QUIT:
```

Set Up Devices File - devicemgmt

The **Devices** file (**/usr/lib/uucp/Devices**) contains information about the devices used to call other machines. For details on this file, refer to Chapter 9, "Basic Networking."

Step 1: To add entries to the **Devices** file, type **sysadm devicemgmt**, then select **2** (add).

```
# sysadm devicemgmt

Running subcommand 'devicemgmt' from menu 'uucpmgmt',
BASIC NETWORKING UTILITIES MANAGEMENT

This procedure is used to list, add, and delete entries
in the Basic Networking Utilities '/usr/lib/uucp/Devices' file.
This file contains information about devices
available for calling out using the commands: uucp, cu, and ct.

Type 'q' at any time to quit the present operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <RETURN> for the default.

Enter the operation you want to perform:

        1 list
        2 add
        3 delete

(default list)[q]: 2
```

The subcommand will prompt you for information on the devices used by basic networking.

port name	The name of the port to which the device will be connected.
device name	The name of the device that is being connected to the above port. Pick the one you are using from the list that is displayed. The default is "ATT2212C." If ACU (Automatic Calling Unit) is specified as the device type to be connected to the port, two entries are created: one for 300 baud and one for 1200 baud.

Step 2: After you have entered the requested information, it will be displayed to you before it is entered into the **Devices** file.

The **/etc/inittab** file may not contain a correct entry for the port just assigned. You can change the port now or later using the **portmgmt** subcommand in the next procedure.

Set Up /etc/inittab - portmgmt

The **inittab** file (**/etc/inittab**) contains information on the ports to which the devices are connected. For further information on this file, refer to Chapter 3, "Processor Operations."

To add BNU entries to the **inittab** file, type **sysadm portmgmt** and select **2** (modify).

```
# sysadm portmgmt

Running subcommand 'portmgmt' from menu 'uucpmgmt',
BASIC NETWORKING UTILITIES MANAGEMENT

This procedure is used to list and modify
the entries that control the direction of traffic
on the Basic Networking Utilities I/O ports used by uucp, cu, and ct commands.

Type 'q' at any time to quit the present operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <RETURN> for the default.


Enter the operation you want to perform:

        1 list
        2 modify


(default list)[q]:  2
```

The subcommand lists the ports available to be used by Basic Networking, and then it asks you to choose the one you want to change. It prompts you for the following information:

port name	Name of the port you want to change (must be a port shown in the list).
traffic direction	The direction of the traffic on the port. You must specify whether the traffic will be **incoming** only, **outgoing** only, or **bidirectional**.
baud rate	Enter the speed (baud rate) of the selected port.

After you have entered the requested information, it will be displayed to you before it is entered into the **/etc/inittab** file.

Note: Since adding a device (**sysadm devicemgmt**) automatically creates a port entry in **/etc/inittab**, you may only need to use **sysadm portmgmt** for changes.

Set Up Systems File - systemmgmt

The **Systems** file (**/usr/lib/uucp/Systems**) contains the information needed by **uucp** to call and log on to a remote machine or a remote machine to call your machine. Each entry represents one remote machine that can be called by your basic networking programs.

Note: If your **remote.unknown** file is not executable, any machine will be able to call your machine.

If the **Systems** entry is to be used to contact a machine that is hardwired to your 3B2 Computer, refer to Chapter 9, "Basic Networking," for special instructions on setting up the **Systems** file.

Step 1: To add other machines to your **Systems** file, type **sysadm systemmgmt** and then select **2.**

```
# sysadm systemmgmt

Running subcommand 'systemmgmt' from menu 'uucpmgmt',
BASIC NETWORKING UTILITIES MANAGEMENT


This procedure is used to list, add, and delete entries in the
Basic Networking Utilities '/usr/lib/uucp/Systems' file.
The '/usr/lib/uucp/Systems' file contains information about the remote
systems that can be called by cu and uucp commands.

You can also use this procedure to try to call (via uucp)
any remote system that appears in the '/usr/lib/uucp/Systems' file.

Type 'q' at any time to quit the current operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <RETURN> for the default.


Enter the operation you want to perform:

        1 list
        2 add
        3 delete
        4 call

(default list)[q]:   2
```

After you select **2** (add), the subcommand will prompt you for the following information.

node name	Node name of the system you want to call.
device type	Type of device used to establish connection (for example, ACU).
baud rate	The speed at which the device will place the call.
phone number	The telephone number of the remote machine. Special symbols can be embedded in the phone number, including abbreviations from the **Dialcodes** file (**/usr/lib/uucp/Dialcodes**).

remote device	Type of equipment you are dialing into at the remote site. The types that are commonly used are: dialup (default), develcon, micom, or none (used if the machine is on the same switch or for a direct line).
login ID	Used by **uucp** to log in on the remote machine.
password	The password associated with the above login.

Step 2: After you have entered the requested information, it will be displayed to you before it is entered into the **Systems** file.

Set Up Poll File - pollmgmt

The **Poll** file (**/usr/lib/uucp/Poll**) contains a list of the machines that are to be called (polled) by your 3B2 computer to see if they have anything to transmit to you. It also contains the times they are to be polled.

Step 1: To add entries to the **Poll** file, type **sysadm pollmgmt** and then select **2** (add).

```
# sysadm pollmgmt

Running subcommand 'pollmgmt' from menu 'uucpmgmt',
BASIC NETWORKING UTILITIES MANAGEMENT

This procedure is used to list, add, and delete entries in the
Basic Networking Utilities '/usr/lib/uucp/Poll' file.
This file contains information about what systems and
the times (hours) the systems should be polled.

Type 'q' at any time to quit the current operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <RETURN> for the default.


Enter the operation you want to perform:

        1 list
        2 add
        3 delete

(default list)[q]:
```

The **pollmgmt** subcommand prompts you for the following information:

system name	Name of the system you want to poll.
polling hours	The hours you want to poll the system; must be an integer number between 0 and 23 (for example, 0 4 8 12 16 20 is every 4 hours).

Step 2: After you have entered the requested information, it will be displayed to you before it is entered into the **Poll** file.

Set Up Permissions File

The default **/usr/lib/uucp/Permissions** file provides the maximum amount of security for your 3B2 computer. The file, as delivered, contains the following entry:

```
LOGNAME=nuucp
```

You can set additional parameters for each machine to define:

- The ways it can receive files from your machine
- The directories it can read and write in
- The commands it can use for remote execution.

See Chapter 9, "Basic Networking," for information on how to set up this file. If you want to change the contents of this file, you must edit it to modify the file and make the entries you desire.

Set Up Devconfig File

The **/usr/lib/uucp/Devconfig** file is only needed if you are using Basic Networking Utilities (BNU) over a STREAMS-based transport provider that conforms to the AT&T Transport Interface (TI). If, for example, 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:ld0
service=uucico  device=STARLAN  push=ntty:tirdwr:ld0
```

Remove the comment character (#) in front of each of these lines to activate them. You must also create an entry for STARLAN in your **Devices** file. Descriptions in the **Devices** file tell how to define Transport Interface devices. (See Procedure 9.6 for a complete example of setting up BNU on a STREAMS-based transport provider.)

Devconfig entries define the STREAMS modules that are used for a particular TI 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 **uucp** services. If you want to change the contents of this

file, you must use an editor (**ed** or **vi**) to modify the file and make the entries you desire.

Set Up Sysfiles File

The **/usr/lib/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 **cu** login services can be made to addresses other than **uucp** services.
- You may want different **Dialers** files to use different 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 Chapter 9, "Basic Networking." 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
```

If you want to change the contents of this file, you must use an editor (**ed** or **vi**) to modify the file and make the entries you desire.

Other Networking Files

There are three other files that affect the use of basic networking facilities. Usually, the default values are fine and no changes are needed. If you want to change them, however, use any standard UNIX system editor (**ed** or **vi**).

Maxuuxqts	This file defines the maximum number of **uuxqt** programs that can run at once.
Maxuuscheds	This file defines the maximum number of **uusched** programs that can run at once.
remote.unknown	This file is a shell script that executes when an unknown machine starts a conversation. It will log the conversation attempt and fail to make a connection. (If you change the permissions of this file so it cannot execute, your system will accept any conversation requests.)

Procedure 9.3: Basic Networking Maintenance

Purpose	To keep files related to basic networking from consuming too much disk space.
When Performed	Automatically with **cron**(1M) or as needed.
Starting Conditions	System state—multiuser or single user.

Basic Networking Utilities come with four shell scripts that will poll remote machines, reschedule transmissions, and clean up old log files and unsuccessful transmissions. These shell scripts should be executed regularly to keep your basic networking running smoothly. Normally, they are run automatically with **cron**(1M), though they can also be run manually. The few areas needing clean up that are not handled by these shell scripts must be maintained manually.

Automated Networking Maintenance (cron)

The Basic Networking Utilities are delivered with entries for **uudemon** shell scripts in the **/usr/spool/cron/crontabs/root** file. These entries will automatically handle some BNU administrative tasks for you. Each of these shell scripts is in **/usr/lib/uucp**.

When the 3B2 computer is in run state 2 (multiuser), **cron** scans the **/usr/spool/cron/crontabs/root** file every minute for entries scheduled to execute at that time. As the system administrator, you should become familiar with **cron** and the four **uudemon** shell scripts.

uudemon.poll

The **uudemon.poll** shell script, as delivered, does the following:

- Reads the **Poll** file (**/usr/lib/uucp/Poll**) twice an hour.
- If any of the machines in the **Poll** file are scheduled to be polled, a work file (**C.sysnxxxx**) is placed in the **/usr/spool/uucp/***nodename* directory, where *nodename* is replaced by the name of the machine.

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 3B2 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 (**/usr/spool/locks**).
- Sends resulting status information to the **uucp** administrative login via mail.

There is no default entry **/usr/spool/cron/crontabs/root** for **uudemon.admin**. The following is recommended:

```
48 8,12,16 * * * /bin/su uucp -c /usr/lib/uucp/uudemon.admin > /dev/null
```

uudemon.cleanup

The delivered **uudemon.cleanup** shell script does the following:

- Takes log files for individual machines from the **/usr/spool/uucp/.Log** directory, merges them, and places them in the **/usr/spool/uucp/.Old** directory with other old log information. If log files get large, the ulimit 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.) 2 days old or older from the spool files.
- 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 (**uucp**).

No default root crontab entry for **uudemon.cleanup** is delivered. This is a recommended entry which would appear on one line.

```
45 23 * * * ulimit 5000; /bin/su uucp -c /usr/lib/uucp/uudemon.cleanup >
/dev/null 2>&1
```

Manual Maintenance

Some files may grow indirectly from **uucp** and other basic networking activities. Here are two files you should check and truncate 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/spool/cron/crontabs/root** file use the **su** command, the **sulog** will grow over time. You should truncate 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 multiuser state.

Procedure 9.4: Basic Networking Debugging

Purpose	To use available monitoring tools to solve basic networking problems.
Starting Conditions	System state— multiuser or 1 single user.
Commands	**uustat**(1) **cu**(1) **Uutry**(1) **uuname**(1M) **uulog**(1) **uucheck**(1)

These procedures describe how to go about solving common problems that may be encountered with Basic Networking Utilities.

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 line and print debugging information on the attempt. The line must be defined as Direct in the devices file. (You must add a telephone number to the end of the command line if the line is connected to an autodialer or the device must be set up as **direct**.)

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 the following:

- 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: To simply 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 (**tail -f**). Press **BREAK** to end output.

You can copy the output from **/tmp/***machine* if you want to save it.

Step 2: If **Uutry** does not 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 your AT&T Service Representative or authorized dealer. Save the debugging output; it will help diagnose the problem.

Check Error Messages

There are two types of error messages for Basic Networking Utilities: ASSERT and STATUS. See Appendix C, "Error Messages," for a listing of these messages.

ASSERT Error Messages

When a process is aborted, ASSERT error messages are recorded in **/usr/spool/uucp/.Admin/errors**. These messages include the file name, **sccsid**, line number, and text. These messages usually result from system problems.

STATUS Error Messages

STATUS error messages are stored in the **/usr/spool/uucp/.Status** directory. The directory contains a separate file for each remote machine with which your 3B2 computer attempts to communicate. These files contain status information on the attempted communication and whether it was successful.

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 outputs information on the permissions you have set up.

Procedure 9.5: Remove BNU Software

Purpose	To remove the software from the hard disk
Starting Conditions	System state— multiuser or 1 single user. You must mount **/usr** to run this procedure in single-user mode. You must be at the computer to insert and remove the Operating System Utilities cartridge tape. Login—**root**.
sysadm Menu	SOFTWARE MANAGEMENT
Commands	**sysadm tapepkg**(1)
Media	The Operating System Utilities cartridge tape.

This procedure describes how to remove Basic Networking Utilities software from your 3B2 computer. Since removing Basic Networking Utilities leaves you without a means of communicating with other computers, this should be a last resort for freeing up disk space.

Note: Removing BNU will turn off the **inittab** entry for every device in **/usr/lib/uucp/Devices** and for all devices in **inittab** that are running **uugetty**.

Prerequisites

Before you remove the Basic Networking Utilities, be sure to do the following:

Step 1: When the software is removed, many support files will be lost. If you reinstall BNU, be sure to save the information in the following files on paper, cartridge tape, or floppy disk.

/usr/lib/uucp/Devices
/usr/lib/uucp/Dialcodes
/usr/lib/uucp/Dialers
/usr/lib/uucp/Permissions
/usr/lib/uucp/Poll
/usr/lib/uucp/Systems
/usr/lib/uucp/Sysfiles
/usr/lib/uucp/Devconfig
/usr/spool/uucppublic/*

Step 2: Before you remove Basic Networking Utilities, you should be logged in as **root** at the console terminal.

Run sysadm tapepkg

Step 1: Take the system to single-user mode (run level S or 1). (See Procedure 3.3.)

Step 2: The following command will let you use System Administration Menu subcommands while in the single-user mode:

mount /usr

Step 3: To remove the Basic Networking Utilities, use the direct access method of the System Administration Menu, and follow the displayed instructions to remove Basic Networking Utilities.

```
# sysadm tapepkg

Running subcommand 'tapepkg' from menu 'softwaremgmt',
SOFTWARE MANAGEMENT



        -- Package installation/removal from SCSI tape --


 Do you wish to install or remove packages?
 [ install remove quit i r q ]  r



Insert the removable medium for the package(s) you want to remove
into the qtapel drive.
Press <RETURN> when ready.  Type  q  to quit.
```

Insert the Operating System Utilities cartridge tape into the SCSI tape drive. Then enter **<CR>**

```
Packages available:

    1   Directory and File Management      13   AT&T Form & Menu Interpreter
    2   User Environment                   14   AT&T FACE
    3   Interprocess Communications        15   Enhanced Ports
    4   System Administration              16   Multiprocessor Enhancement Util
    5   System Header Files                17   SCSI Host Adapter Utilities
    6   SPELL                              18   SCSI Cartridge Tape
    7   Terminal Information- part 1       19   SCSI Mirroring Utilities
    8   Terminal Information- part 2       20   Job Accounting
    9   Editing                            21   System Performance Analysis
   10   Line Printer Spooling              22   Network Software Utilities
   11   Basic Networking                   23   Remote File Sharing
   12   Windowing                          24   3.2.2 Release Upgrade

Enter selection(s) to remove [all help quit]: 11
```

Screen continues on the next page.

```
Selection complete--- removepkg starting.


Removing the Basic Networking Utilities.
The following files are being removed:
Note that uninstalling UUCP has turned off the inittab entries for these
devices:
contty tty21 (Your networking devices are listed here.)
/usr/bin/cu
/usr/bin/ct
/usr/bin/uulog
/usr/bin/uuname
/usr/bin/uucp
/usr/bin/uux
/usr/bin/uustat
/usr/bin/uupick
/usr/bin/uuto
/usr/lib/uucp
/usr/spool/uucppublic
/usr/spool/locks
/usr/spool/uucp
/usr/admin/menu/packagemgmt/uucpmgmt
/usr/options/uucp.name
The Basic Networking Utilities has been removed.




The removepkg from SCSI tape has completed-
You may remove the tape after the rewind is complete.


#
```

Return to Multiuser Mode

Step 1: Unmount the **/usr** file system, leaving only the root file system mounted.

```
umount /usr
```

Step 2: To return to the normal operating state, enter:

```
init 2
```

Your 3B2 computer is now available for use except for Basic Networking Utilities.

Procedure 9.6: Set Up BNU STREAMS-Based Network (Basic)

Purpose	To show how to set up BNU files for a STREAMS-based network using **sysadm**.
Starting Conditions	System state— multiuser or 1 single user. You must mount **/usr** to run this procedure in single user mode. Login—**root**.
Commands	**sysadm uucpmgmt**(1)

This procedure contains an example of setting up BNU files for a STREAMS-based network. Though the example is particular to setting up BNU on an AT&T STARLAN NETWORK, the concepts could be applied to setting up BNU to run on a transport provider that is compatible with the AT&T Transport Interface.

The first part of this procedure shows how to set up BNU files so that **uucp** and **cu** requests both go through login to connect to machines on the network. The second part shows what you must change so **uucico** services can connect to hosts without going through login. Since the second part could present security problems, you should only make this type of service available when your STREAMS-based network consists of trusted machines.

Prerequisites

Before running this procedure to set up BNU to run on a STREAMS-based network, such as STARLAN, you must install these packages in the order shown:

- Networking Support Utilities (optional)
- STARLAN NETWORK (or other optional STREAMS-based network software)
- Basic Networking Utilities (if you have installed this package, you will need to remove it, and install the preceding packages first).

Create STREAMS-Based Network Systems Entry

Repeat the following steps for each machine (node name) on the network with which you want to communicate.

Step 1: Type **sysadm uucpmgmt** to display the Basic Networking Utilities Management Menu.

Step 2: Select **systemmgmt** to display the list of operations you can do on the **Systems** file.

Step 3: Type the number required to choose the **add** function.

Step 4: Type the node name for the remote machine you want to communicate with over the STARLAN NETWORK. (The convention is to use node name. However, some networks use some other means of identifying another machine.)

Step 5: Type number required to select Transport Layer Interface (TLI) as the device type.

Step 6: Type **STARLAN** to choose that name as the network name. (The word **STARLAN** need not be used. You can choose any name to represent the network. The key is that this name must match the network name that goes in the caller type field in the **Devices** file. For consistency, STARLAN will be used when the **Devices** file entry is made in this procedure.)

Step 7: Type the network address used for connection to the remote machine. (The recommended convention is to use node name as the STARLAN NETWORK address of the remote machine for login services.)

Step 8: Type **nuucp** as the login ID to use when calling the remote machine. The (**nuucp** is a conventional login, but others can be used.)

Step 9: Type a password if one is required by the remote machine.

Step 10: Type **y** to place the entry in the **Systems** file.

Step 11: Type **y** if you want to add another system, then repeat Steps 4 through 11. Type **n** if you do not want to add any more systems.

Step 12: Type **q**<**CR**> three times to return to the shell.

Create STREAMS-Based Network Devices Entry

Step 1: Type **sysadm uucpmgmt** to display the Basic Networking Utilities Management Menu.

Step 2: Select **devicemgmt** to display the list of operations you can perform on the **Devices** file.

Step 3: Type the number required to choose the **add** function.

Step 4: Type **starlan** as the name of the port. (Here you must use the literal **starlan**. It shows the device name, relative to the **/dev** directory.)

Step 5: Select TLI network to show that the port is connected to this type of network.

Step 6: Type **STARLAN** as the network name. (This is the Caller Type field. By using different names here you can have multiple TLI networks on your system. This name must match the network name added to the **Systems** file entry.)

Step 7: Type **y** to add the entry to the **Devices** file.

Step 8: Type **q**<**CR**> three times to return to the shell.

Create STREAMS-Based Network Devconfig Entries

The **/usr/lib/uucp/Devconfig** file entries must be created to define the **STREAMS** modules to **push** for **cu** and **uucico** services. There is no **sysadm** menu for this file, so you should edit the file to make sure the following two entries are included.

```
service=cu      device=STARLAN  push=ntty:tirdwr:ld0
service=uucico  device=STARLAN  push=ntty:tirdwr:ld0
```

The two lines are already in the **Devconfig** file. To activate them, remove the comment character (#) in front of each line.

Set Up STREAMS-Based Network Listener

STARLAN and other TLI networks running on the UNIX system each has a separate network listener process associated with it. The listener process "listens" to the network for service requests, accepts requests when they arrive, and spawns servers in response to those service requests.

The **nlsadmin** administers the network listener process(es) on a machine. The **nlsadmin** can establish a listener process for a given network, configure the specific attributes of that listener, and start and kill the listener process for that network. [See **nlsadmin**(1M) for more information.]

Chances are, you already started a listener that was configured to provide standard login services when you installed the STARLAN NETWORK. The following steps show you how to check that the STARLAN NETWORK is set up properly to support BNU and what to enter in case it is not.

Step 1: Type:

nlsadmin -v starlan

You should see the following:

```
  1   ENABLED     NOMODULES     /usr/slan/lib/ttysrv     comment
102   ENABLED     NOMODULES     /usr/slan/lib/ttysrv     comment
```

If you do not see the above, then type:

```
# nlsadmin -a 1 -c "/usr/slan/lib/ttysrv" -y "tty login service" -m starlan
# nlsadmin -a 102 -c "/usr/slan/lib/ttysrv" -y "tty login service" starlan
```

Step 2: Type:

nlsadmin -x

You should see:

`starlan ACTIVE`

If you see:

`starlan INACTIVE`

then type:

nlsadmin -s starlan

to start it up. If you do not see either `starlan` ACTIVE or `starlan` INACTIVE, then the STARLAN NETWORK was not properly installed.

STREAMS-Based Network Dialers Entry

You do not need to create an entry in the **Dialers** file for standard BNU services over STREAMS-based network the previous procedure shows.

BNU is now ready to communicate over a STREAMS-based network.

Procedure 9.7: Set Up BNU STREAMS-Based Network (Special)

Purpose	To show how to set up BNU files for a STREAMS-based network to handle **uucico** and **cu** services differently.
Starting Conditions	System state— multiuser or 1 single user. Login—**root**.
Commands	Any text editor [**ed**(1), **vi**(1)]

In the previous procedure, both **uucico** and **cu** services go through a login procedure to connect to other computers on an AT&T STARLAN NETWORK. To improve performance of **uucico** services, you can have **uucico** requests connect directly to remote machines on a STREAMS-based network without going through login.

You should only set up **uucico** as described above if you trust all machines on your STREAMS-based network. The reason is that there is a potential security breach since you would not be requesting a password when a machine tries to transfer files to your machine.

Before you run this procedure, you should run the previous procedure so **cu** services are handled correctly.

Step 1: You must define separate **Systems**, **Devices** and **Dialers** files to handle **uucico** services. This is done in the **Sysfiles** file. The following is an example.

```
service=cu      systems=Systems \
                devices=Devices \
                dialers=Dialers
service=uucico  systems=Systems.cico:Systems \
                devices=Devices.cico:Devices \
                dialers=Dialers.cico:Dialers
```

In the example, **cu** services would use the standard BNU files. The **uucico** services, however, would use the standard BNU files only after checking **Systems.cico**, **Devices.cico**, and **Dialers.cico**. You can use any file names you want. The ones shown above are only given as an example.

Step 2: The systems you want to transfer files to on your STREAMS-based network without going through login must be listed in the new **Systems** file you defined in the **Sysfiles** file.

If you used the file names defined above in the **Sysfiles** file, you would edit **/usr/lib/uucp/Systems.cico** and add entries similar to the following entry for each system.

```
3b2abc Any ListenCico - 3b2abc.serve
```

In this example, **3b2abc** is the system name, **Any** says use the highest baud rate possible, **ListenCico** points to a device type in the **Devices.cico** file, the dash is a field place holder, and **3b2abc.serve** is the STREAMS-based network address of **3b2abc**.

Step 3: Continuing with the example above, from the entry in the **Systems.cico** file, you must add an entry to the **/usr/lib/uucp/Devices.cico** file called **ListenCico**. This entry should look like the following:

```
ListenCico,eg starlan - - TLI \D listencico
```

In this example, **ListenCico** is the device name, **starlan** identifies the network, the two dashes are field place holders, **TLI** is the network type, **\D** says read in the network address (**3b2abc**), and **listencico** is the dialer type.

Note: You do not need an entry in the **Devconfig** file for **uucico** since you will not be pushing any STREAMS modules.

Step 4: From the entry in the **Devices.cico** file, you must add an entry to the **/usr/lib/uucp/Dialers.cico** file called **listencico**. The following is an example of what you should enter.

```
listencico "" "" NLPS:000:001:101\N\c
```

Note: There are no spaces between the pairs of double quotation marks.

Step 5: You must register the **uucico** service with the network listener. The command shown below must all be typed on one line.

 # nlsadmin -a 101 -c "/usr/lib/uucp/uucico -r 0 -i TLI -u \ nuucp" -y "uucico server with NO login checking" starlan

This entry says to answer requests for service code 101 with **uucico** directly as it can be used for a **TLI** network. This same service code must be added to all the systems that you wish to communicate with in the way. Notice that the default **nuucp** login ID is used; you can use a different one if you choose.

Remote File Sharing Procedures

Remote File Sharing Procedures

Procedure 10.1	**Set Up Remote File Sharing (setuprfs)** To set up all basic information needed to run Remote File Sharing (RFS).
Procedure 10.2	**Start/Stop Remote File Sharing (startstop)** To start and stop Remote File Sharing, check if it is currently running, and set up RFS to start automatically at system boot time.
Procedure 10.3	**Local Resource Advertising (advmgmt)** To manage the local resources that you make available to other machines.
Procedure 10.4	**Remote Resource Mounting (mountmgmt)** To manage remote resources that are made available to your machine.
Procedure 10.5	**Change RFS Configuration (confgmgmt)** To change your ID mapping, show your current RFS configuration, or update the domain member list.

These procedures are designed to help you set up and maintain Remote File Sharing (RFS) Utilities on your computer. Before you run any of these procedures, you should do the following:

> Read the *Remote File Sharing Release Notes*. This will tell you how to install RFS and help you with some special problems you may run into.

The first time you set up RFS you should use the **sysadm** interface available with RFS. The interface not only lets you add all basic RFS configuration information, but it also acts as a tutorial by introducing and explaining key RFS concepts.

The following are the two recommended ways you can use the **sysadm** interface for RFS.

- Type **sysadm rfsmgmt**. This will bring you to the top Remote File Sharing Menu. From there you can select the function or submenu you want, such as **setuprfs** to set up RFS for your system. Browsing through the **sysadm rfsmgmt** menus is a good way to familiarize yourself with the functions available.

- Type **sysadm** *subcommand,* where *subcommand* is replaced by the specific **sysadm** subcommand you want to use. Procedures 10.1 through 10.5 describe how to go directly to the **sysadm** subcommand you need for different RFS administrative tasks.

A diagram of the **sysadm rfsmgmt** subcommand tree is shown below. The diagram also notes the section of this document where each subcommand is described.

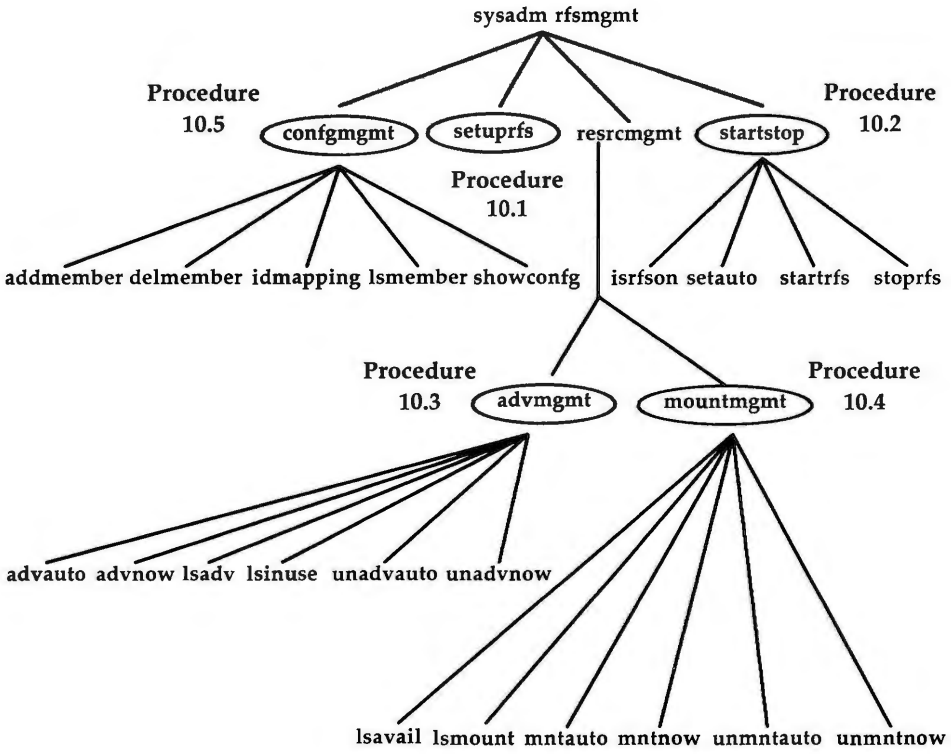


Figure P10-1: The **sysadm rfsmgmt** Subcommands

Should you need more information as you are setting up RFS, there are several things you can do:

- Type **?**. A message will be printed at your terminal that will further describe what you need to know to complete the step.
- Check the "Glossary" section. A glossary of terms is provided at the end of this section to explain important RFS terms.
- Read Chapter 10. The Remote File Sharing chapter provides the most complete description of the RFS software. The first-time RFS administrator will be particularly interested in the "Overview" section.

The **sysadm** interface lets you do everything necessary to set up and run RFS in a basic configuration. However, there are several optional features that are not available through the **sysadm** interface.

The optional features not available using the **sysadm** interface are described at the end of the "Setting Up RFS" section of Chapter 10. The word "Optional" is placed in the heading of each optional feature. The features include the following:

Remote Computer Verification.
: By default, when a machine requests the use of one of your resources, your machine will process the request without verifying the remote machine's password. This procedure describes how to restrict access of all your resources to a limited group of remote machines whose names and passwords match those in lists you set up.

Complex user ID/group ID mapping.
: ID mapping defines the permissions remote users will have to your resources. The choices of ID mapping schemes are limited when you use the **sysadm** interface to set up mapping. This procedure gives you more flexibility in setting up permissions for remote users.

Multiple Domain Resource Sharing.
: The **sysadm** interface assumes that you are only sharing resources within one domain. However, it is possible to have more than one domain on a network. This procedure describes how to share resources among multiple domains on the same network.

Multiple Domain Name Service.

When you define the primary and secondary name servers using **sysadm**, you are defining them to serve a single domain. You can, however, define the same set of machines to be the name server for several domains using this procedure.

More experienced RFS administrators will also be interested in the "Monitoring" and "Parameter Tuning" sections of Chapter 10. Information in these sections will help you fine tune your system so RFS can make the most efficient use of your system's resources.

RFS Glossary

Advertise

To make a local resource available to other computers using Remote File Sharing. The **adv**(1M) (or **sysadm advnow**) command is used by administrators to advertise a resource.

Advertise Table

An internal list of available resources. An advertise table on each computer running RFS has the name of each resource the computer has made available.

Automatic Advertise List

A list of local resources that are automatically offered to other computers when Remote File Sharing is started. The list consists of full **adv** command lines placed inside the **/etc/rstab** file. The command lines are added to **/etc/rstab** using the **sysadm advauto** command or by any standard file editor.

Automatic Mount List

A list of remote resources that are mounted on the local system when Remote File Sharing is started. The list is contained in the **/etc/fstab** file. (See the **fstab**(4) manual page in the *System Administrator's Reference Manual* for the format of the file.) Automatic mount information is added to **/etc/fstab** using the **sysadm mntauto** command or by any standard file editor.

Client

A Remote File Sharing computer that is using a remote resource.

Client Caching

The ability of an RFS computer that is using a remote resource to store remote data blocks in its local buffer pools. This technique improves RFS performance by reducing the number of times data must be read across the network.

Client List

When an RFS administrator advertises a resource, the administrator can restrict the resource so only certain remote machines can use it. This list of machines is added to the **adv**(1M) command line when a resource is advertised.

Client Permissions

When an RFS administrator advertises a resource, the administrator can set permissions for the resource. The permissions are assigned on the **adv**(1M) command line. If the permissions are read-only, the client computers can only mount the resource with read permissions. If they are read/write, a client can mount the resource read/write or read only.

Current Name Server

When a domain is set up, a primary and zero or more secondary domain name servers are assigned. Only one of those machines is actually handling domain name server responsibilities at a time. That machine is referred to as the current name server. Normally, the primary will be the current name server. However, if the secondary has taken over temporarily, it is the responsibility of the secondary's administrator to pass the responsibility back to the primary whenever the primary resumes running RFS. (See also **Domain**, **Primary Name Server**, and **Secondary Name Server**.)

Directory Path Name

RFS will ask you for a full path name to a directory in two instances. When you advertise a local resource, you will need the full path name of the directory you are advertising. When you mount a remote resource, you will need the full path name of the directory where the remote resource should be attached.

Domain

A logical grouping of computers in an RFS environment. A domain name is like a telephone area code, acting as an addressing prefix to attach to a computer name or a resource. Assigning a domain name

server for a domain provides a central location where lists of resources and network addresses for the group of computers can be stored. Domains also provide a level of security.

Domain Information (rfmaster)
The primary and secondary name server assignments for a domain are stored in the **/usr/nserve/rfmaster** file. The primary keeps the definitive copy of this file and distributes it automatically to each computer in the domain when each starts RFS. This file also contains the network address of each name server.

Domain Member List
The list of the computers that make up an RFS domain. This list is stored in the **/usr/nserve/auth.info/***domain***/passwd** file on the primary name server, where *domain* is replaced by the name of the domain. Members are added on the primary using the **rfadmin -a** or **sysadm addmember** commands.

Forced Unmount
To unmount one of your local resources from all remote machines that have mounted it. This has the effect of killing all processes that are currently using the resource on all client machines.

ID Mapping
To define the permissions remote users and groups have to your advertised resources. The tools available for mapping let you set permissions on a per-computer basis and on a global basis. You can then map individual users or groups by ID name or number. When you map IDs for Remote File Sharing, it is easiest to do so with ID numbers since mapping by name requires that you have copies of the remote machines' **/etc/passwd** and **/etc/group** files.

Local Resource
A directory that resides on your machine that you have made available for other computers running RFS to use. You must advertise the directory (**adv**(1M) command) to offer it to other computers. If a remote machine mounts your resource, it could have access to all subdirectories, files, named pipes, and devices within your directory (depending on file permissions you set up).

Mount
The special use of **mount**(1M) in RFS is to attach a remote resource to a directory on your system so local users can access the remote

resource. Once mounted, the remote resource appears to be just another part of the local UNIX file system tree. See the **mount -d**(1M) command and the **sysadm mountmgmt** procedures.

Network Specification

The name that identifies a networking product that is compatible with the AT&T Transport Interface (TI). This is also referred to as the transport provider. RFS requires a transport provider to communicate with other machines. The network specification is used to tell RFS the exact device to use for communications. For example, you would enter **starlan** to tell RFS to use the **/dev/starlan** device if the AT&T STARLAN NETWORK is the transport provider.

Network Listener

The process used by a transport provider to wait for any type of incoming requests from the network. Once a request comes in, the listener directs it to one of the processes registered with the listener. The process represents a service, such as **uucp** or RFS.

Networking Support Utilities

A software package that contains the network listener. This package must be installed in order to use Remote File Sharing.

Network Address

The address by which a computer is known to a particular network. An RFS administrator needs to know the network address of the primary to start RFS for the first time. Address information of other machines is handled internally by RFS. For the AT&T STARLAN NETWORK, the network address is in the form *nodename*.**serve**, where *nodename* is the machine's communications node name.

Node Name

The name you assign to your computer to use for communications needs (use **sysadm nodename** to change it). Networking software, such as Basic Networking Utilities and Remote File Sharing, use this name to identify your machine. A full RFS computer name is *domain.nodename*, where *domain* is the name of the computer's RFS domain.

Primary Name Server

The computer that is assigned to provide a central location for addressing and information collection for an RFS domain.

Information includes a list of domain members, resources offered by domain members, and optional user ID mapping information. Secondary name servers can be assigned to continue limited name service when the primary is down. For example, a secondary cannot add or delete domain members.

RFS Automatic Startup

Remote File Sharing can be set to start automatically when your machine is booted. This is done by changing the **initdefault** line in the **/etc/inittab** file from **2** to **3**. The **sysadm setauto** command does this for you automatically. (The **init** state **3** is the Remote File Sharing/Multiuser state.)

RFS Daemon (rfudaemon)

A daemon process that runs when RFS is running. When network connections to remote resources are broken, **rfudaemon** sends a message to **rfuadmin**, which then continuously tries to remount the resource. (See **rfudaemon**(1M) and **rfuadmin**(1M) for further information.)

RFS Password

A password assigned by the primary name server for every computer in its domain. Each computer must enter its password the first time it starts RFS. After that the password is stored locally in **/usr/nserve/loc.passwd**. By copying the domain password file from the primary (**/usr/nserve/auth.info/***domain*/**passwd**), a computer can verify that a remote machine trying to mount its resource is the machine it claims to be.

Remote File Sharing State (init 3)

The special initialization state used to start RFS. When you type **init 3** or set the **initdefault** line in **/etc/inittab** to **3**, your system will start RFS, advertise all resources in your automatic advertise list, and mount all resources in your automatic mount list.

Remote Resource

A directory that resides on a remote machine that is available for you to connect to using RFS. You must mount the resource [**mount**(1M) or **sysadm mntnow** commands] to make it available to users on your system. Once you mount the remote resource, your users could have access to all subdirectories, files, named pipes and devices related to your directory (depending on file permissions the remote machine set up).

Resource

See **Remote Resource** and **Local Resource**.

Resource Identifier

The name assigned to a resource when it is advertised. The name is limited to 14 printable ASCII characters. Slash (/), period (.), and white space may not be used.

Secondary Name Server

A computer designated to take over name server responsibilities temporarily should the primary domain name server fail. The secondary cannot change any domain information. It can, and should, only pass name server responsibility back to the primary when RFS is running on the primary again.

Server

A Remote File Sharing computer that offers a resource to others.

Transport Provider

The software that provides a path through which network applications can communicate. RFS can communicate over any transport provider that meets the AT&T Transport Interface Specification. (STARLAN NETWORK is one of AT&T's Transport Interface-compatible network.)

User/Group Name

The names associated with each local user and group that is allowed access to your computer. This information can be found in the first field of the **/etc/passwd** or **/etc/group** files, respectively. Remote users and groups can be assigned the same permissions as the local users and groups by using RFS ID mapping.

User/Group ID Number

Every user and group name has a corresponding number that is used by the UNIX operating system to handle permissions to files, directories, devices, etc. These numbers are defined in the third field of the **/etc/passwd** or **/etc/group** files, respectively. Remote users and groups can be assigned the same permissions as the local users and groups by using RFS ID mapping.

Procedure 10.1: Set Up Remote File Sharing (setuprfs)

Purpose	To set up everything needed to run Remote File Sharing Utilities.
Starting Conditions	System state—2 multiuser. You must mount **/usr** to run this procedure in single-user mode. Login—**root**.
Commands	**sysadm setuprfs**
References	Chapter 10, "Remote File Sharing."

This procedure is used to set up Remote File Sharing on your machine. When the procedure is done, you will have completed everything needed to run Remote File Sharing on your system.

Note: When you complete this procedure, **sysadm** will create the file **/usr/nserve/rfmaster**. If you had previously created that file, the old version will be placed in **/usr/nserve/Orfmaster**.

Prerequisites

Before you begin setting up RFS, you must do the following.

- Install software and hardware as described in the *Remote File Sharing Release Notes*. (The order of software installation in the Release Notes is very important!)
- Choose one or more computers on the network to act as domain name servers. Exactly one primary is required. All domain administration is done from the primary. You can choose zero or more secondary domain name servers. These are defined simply to keep the name server running temporarily, should the primary fail. (You can configure RFS on your machine before the primary is configured and running RFS. However, you cannot start RFS until the primary begins running RFS.)
- Log in as **root**.

Set Up RFS

Step 1: Type **sysadm setuprfs**. You will see the following information:

```
                        SET UP RFS

If you have not yet set up RFS on your machine, this subcommand will walk you
through all the steps necessary to set up RFS.  These steps are:

        * entering the name of the transport provider
        * designating your machine as a primary name server or a non-primary
        * entering the domain name
        * primary only: adding members to domain (optional)
        * primary only: designating secondary name servers (optional)
        * setting up ID mappings (optional)
        * putting local resources on the Automatic Advertise List (optional)
        * putting remote resources on the Automatic Mount List (optional)
        * setting up RFS to start automatically (optional)
        * starting RFS (optional)

If these terms are unfamiliar do not be concerned.  All new terms and
concepts will be explained.  Explanations of RFS terms and concepts can
also be found in your System Administrator's Guide.

In most cases, the "setuprfs" subcommand will call other RFS sysadm menus
to do the specific tasks.  Later, when you want to do these tasks as part of
your ongoing RFS administration you will use these specific RFS sysadm menus.

There are a few tasks (that you won't need to do very often) that can not
be done through the specific menus.  To do these you will have to stop RFS
and rerun this "setuprfs" subcommand.  These tasks are:

        - changing the name of the transport provider
        - changing the name of the domain
        - recognizing a new primary name server
        - primary only: adding secondary name servers
                                                        continued
```

```
(Continue from previous screen display)


If you are running this command to change something in your domain
configuration, any old domain configuration information will be saved
for you in /usr/nserve/Orfmaster.

This "setuprfs" subcommand assumes you have already installed a
transport provider and that the network listener is properly set up.
You will be unable to set up RFS until these assumptions are met.
SET THE TRANSPORT PROVIDER ...

To share resources between machines, the machines must be connected in some
way.  Because RFS is media independent, it is not tied to any one type of
connection medium.  The connection between machines, the TRANSPORT PROVIDER,
may be any network that is compatible to the AT&T Transport Interface.  The
System Administrator's Guide explains the Network Support Utilities and how
they work together with RFS.

RFS assumes that this transport provider has been installed.

Enter the name of your transport provider [?, q]:
```

Step 2: Answer each question as it is asked. If you are setting up RFS for the first time, you should answer all questions. If you make a mistake or want to change something later, you can run **sysadm setuprfs** again.

The following is a list of information you will be asked:

Transport provider - This is the name of the transport provider that RFS will use to communicate across the network. If you are told that the listener is not running, see the "Set Up Network Listener" section of Chapter 10. (This value will be **starlan** if the transport provider is the AT&T STARLAN NETWORK.)

Machine type - This will tell your machine whether or not it is the primary name server for your domain.

Domain name - This is the name of your domain. It can be up to 14 characters in any combination of letters, digits, hyphens, and underscores. This name will be the same for all computers in your domain.

Nonprimary - If your machine IS NOT the primary name server, you will be asked to enter the following information:

> **Primary's node name** - You must enter the primary's node name, then its network address. Address formats are different for different networks. (The AT&T STARLAN NETWORK uses the format *nodename*.**serve**, where *nodename* is replaced by the node name of the primary name server.)
>
> **RFS password** - The RFS password for your machine. This must match the password entered for your machine by the primary domain name server.

Primary - If your machine IS the primary, you will be asked for the following information.

> **RFS password** - You will be asked twice to enter a password for your machine. Enter the same password both times.
>
> **Add domain members** - You will be asked to enter the machine names of every machine in your domain. Don't worry if you don't know all of the members now, you can always add and delete later.
>
> **Define secondaries** - You will be asked to enter the name and network address of the secondary name server(s) for your domain. This is optional. Secondaries are only there to take over temporarily when the primary is not available. The network address you add is in the same format as that of the primary.

User and group ID mapping - Since user ID and group ID mapping can be complex, this interface presents you with several standard options. Choice **a** (all remote IDs into a guest ID) is the best choice in most cases since it provides the maximum security

and the minimum complexity. Choices **b** or **c** (each remote ID number to the corresponding local ID number) are only valuable if you have identical **/etc/password** and **/etc/group** files among machines with which you share resources. The difference between **b** and **c** is that **b** protects all administrative logins and **c** only protects **root** login. (See Chapter 10 for information on more complex ID mapping schemes.)

Auto advertise list - If there are some local directories you are ready to share, you can add them to a list of those that are automatically offered when you start RFS. You must know the path name to that directory, make up a name to identify it (resource identifier), decide if you want to restrict it to read-only access, enter a short description, and enter a list of client machines (if you only want certain machines to access the resource).

Auto mount list - If there are remote resources that will be available from other machines in your domain, you can add them to your automatic mount list. Those resources will be automatically mounted on your machine when RFS is started. Usually, **/mnt** is a safe place to use as a mount point directory for the first remote resource you try with RFS. (Read the mount guidelines section in Chapter 10 for information on where you should not mount remote resources.)

Automatic RFS start - If you select the automatic start feature, your machine will automatically start Remote File Sharing when it is booted. This will place your machine in **init** state **3**. (See Chapter 10 for a detailed description of **init 3** processing.)

Start RFS now - If the primary name server is up and running RFS and if everything was done correctly in the previous steps, you should be able to start RFS immediately. Be patient. It may take several minutes.

CONFIGURATION IS COMPLETE!

To make sure RFS is working, you could return to the shell and do the following:

- Type **rfadmin -q** (or **sysadm isrfson**) to see if RFS is running.
- Type **adv** (or **sysadm lsadv**) to list the local resource you have advertised.
- Type **mount** (or **sysadm lsmount**) to list the remote resources you have mounted. (The **mount** will also list locally mounted devices.)

If you want more information on some of the more complex security and resource sharing facilities available, see Chapter 10. The following chart describes the **sysadm setuprfs** functions and lists dependent commands and files of the procedure you just completed. This information should help you relate the **sysadm** processing to the descriptions of RFS components in Chapter 10.

sysadm Commands	Dependent Commands	Files	Description
setuprfs		**rfmaster** *	Set up basic Remote File Sharing information.
	dname -N		Define the Transport Provider used by RFS.
	dname -D		Define the domain your machine is a member of.
	idload	**uid.rules** † **gid.rules** †	Choose mapping scheme. (See **sysadm idmapping**.)
	adv	**/etc/rstab**	Advertise automatically. (See **sysadm advauto**.)
	adv **nsquery** **mount -d**	**/etc/fstab**	Mount automatically. (See **sysadm mntauto**.)
		/etc/inittab	Set RFS to start automatically. (See **sysadm setauto**.)
	rfstart	**/etc/rstab** **/etc/fstab**	Start RFS now. (See **sysadm startrfs**.)
‡	**rfadmin -a**		Add machines to domain. (See **sysadm addmember**.)
‡		**rfmaster** *	Define secondary machine(s).

* File appears in the directory **/usr/nserve**.

† File appears in the directory **/usr/nserve/auth.info**.

‡ You will be prompted for this information only if you are on the primary name server.

Figure P10-2: The **sysadm setuprfs** Description

Procedure 10.2: Start/Stop Remote File Sharing (startstop)

Purpose	Start and stop Remote File Sharing.
Starting Conditions	System state—2 multiuser or 3 (if RFS is running). Login—**root**.
Commands	**sysadm startstop**
References	Chapter 10, "Remote File Sharing."

The **sysadm startstop** command presents a menu of subcommands that can be used to start RFS, stop RFS, check to see if RFS is currently running, and set RFS to start automatically at boot time.

Prerequisites

Before you run **sysadm startstop** subcommands, do the following.

Set up RFS. Use **sysadm setuprfs** to enter basic RFS information.

The following chart describes the **sysadm startstop** subcommands and provides information on dependent commands and files. This will help you relate **sysadm** processing with the RFS components described in Chapter 10.

sysadm Commands	Dependent Commands	Files	Description
isrfson			Reports whether or not RFS is running.
setauto		**/etc/inittab**	Set up your system so that when it is booted Remote File Sharing is automatically started. This command also lets you set up your system to NOT start RFS automatically when it is booted.
startrfs	**rfstart**(1M) **rmountall**(1M)	**/etc/rstab** **/etc/fstab**	Start RFS immediately. When you start RFS with this command, the command will also advertise and mount all resources in your Automatic Advertise List and Automatic Mount List, respectively.
stoprfs	**rfstop**(1M) **fumount**(1M) **unadv**(1M) **fuser**(1M)		Stop RFS immediately. Unmount any resources mounted on your machine. Unadvertise all of your resources and force them to be unmounted from other machines.

Figure P10-3: The **sysadm startstop** Subcommands

Check If RFS Is Running

Type **sysadm isrfson** to see if RFS is currently running. If it is, you will see the following message:

RFS is running.

Set RFS to Start Automatically

Step 1: Type **sysadm setauto**. You will then see the following:

```
SET UP / TURN OFF AUTOMATIC START OF RFS

You can set up your system so that any time you start your machine the
system automatically starts RFS.  When RFS is automatically started,
all the resources you have set up to be automatically advertised and
mounted are advertised and mounted just as if you had manually started
RFS by using the "startrfs" subcommand.

You also use this "setauto" subcommand to turn off automatic starting
of RFS.  When you turn off the automatic start option the machine will
be set to come up in multiuser mode when the system starts.
```

Step 2: If RFS automatic start-up was off, type **y** to turn it on. If RFS automatic start-up was on, type **y** to turn it off.

The **sysadm setauto** command sets up automatic start-up by changing the **initdefault** line in the **/etc/inittab** file to **3** (Remote File Sharing state). When you turn it off, **initdefault** is set back to **2**. See Chapter 10 for other information on the Remote File Sharing state (**init 3**).

Start RFS Now

Step 1: Type **sysadm startrfs** to start Remote File Sharing. If RFS starts successfully, you will see the following:

```
                START REMOTE FILE SHARING

Attempting to start RFS.
This will take a few moments.  Please wait...

        The system will now ATTEMPT to mount the resources you have
        set up to be automatically mounted.  There are several conditions
        that may exist that would prevent a resource from being
        successfully mounted.  These are:
          - The machine owning the resource is not running RFS,
          - The machine owning the resource does not have it advertised,
          - The resource identifier in the Automatic Mount List is incorrect,
          - The local directory you chose as a mount point does not exist,
          - The local directory you chose as a mount point is busy.
        Your system will continue to try to mount a resource if it
        cannot be mounted when RFS is first started.

                    RFS has been started.
```

If RFS does not start, see the "Starting/Stopping Remote File Sharing" section of Chapter 10 for a list of possible problems.

Stop RFS Now

Step 1: Type **sysadm stoprfs** to stop Remote File Sharing and answer **y** to the first prompt. If RFS was running and could be stopped successfully, you will see the following:

```
                    STOP REMOTE FILE SHARING

Stopping RFS will make remote resources unavailable to your local users
and will make your local resources unavailable to remote users.  Processes
using these resources will be killed.  Remote users currently in one of
your directories will be logged off.  Be sure to consider the affect to
local and remote users before stopping RFS.

Do you want to continue to stop RFS? [y, n, q]
```

Procedure 10.3: Local Resource Advertising (advmgmt)

Purpose	Share your resources with other machines.
Starting Conditions	System state—2 multiuser or 3 (RFS state). Login—**root**.
Commands and Subcommands	**sysadm advmgmt** **sysadm advauto** **sysadm advnow** **sysadm lsadv** **sysadm lsinuse** **sysadm unadvauto**
References	Chapter 10, "Remote File Sharing"

You can selectively share parts of your computer's file system with other machines on your RFS network using the **sysadm advmgmt** subcommands.

Prerequisites

Before you run **sysadm advmgmt** subcommands, you should do the following:

- Set up RFS. Use the **sysadm setuprfs** command to enter the basic information needed to run RFS.
- Start RFS. To use **advnow**, **unadvnow**, or **lsinuse** subcommands, RFS should be running. The other subcommands do not require that RFS be running.

The following chart describes the **sysadm advmgmt** subcommands and lists dependent commands and files of the procedures in this section. This information should help you relate the **sysadm** processing to the descriptions of RFS components in Chapter 10.

sysadm Commands	Dependent Commands	Files	Description
advauto	**adv**(1M)	**/etc/rstab**	Set up a resource to be advertised each time RFS is started. The information you enter for each resource will be added to the Automatic Advertise List.
advnow	**adv**(1M) **nsquery**(1M)		Advertise a resource immediately.
unadvauto		**/etc/rstab**	Remove a resource from the Automatic Advertise List. Once removed, the resource will not be advertised the next time RFS is started.
unadvnow	**unadv**(1M) **fumount**(1M) **fuser -k**		Immediately unadvertise a resource, then (optional) unmount it from all client machines.
lsinuse	**rmntstat**(1M)		Print a list of all the machines that have mounted your advertised resources. For each resource the list includes: the Resource Identifier, the path name to the resource, and the name of each client machine.
lsadv	**adv**(1M)	**/etc/rstab**	Print a list of all your resources that are currently advertised and those that are on your automatic advertise list.

Figure P10-4: The **sysadm advmgmt** Subcommands

Advertise Automatically

Step 1: Type **sysadm advauto** to add an entry to the Automatic Advertise List. You will then see the following:

```
        ADD RESOURCES TO THE AUTOMATIC ADVERTISE LIST

You share your local resources with other machines by advertising
them.  The machines that can use the resources you advertise are
called clients.

This subcommand lets you add resources to the Automatic Advertise List
to be advertised every time RFS is started.  You will be asked for all
the information needed.

NOTE:  Adding resources to the Automatic Advertise List does NOT
make the resources immediately available to clients.  If RFS is
running when you access this command you may be able to immediately
advertise the resources.  The next time RFS is started the resources
will be automatically advertised for you.

Enter the full path name of the local directory to be added to
the automatic advertise list [?, q]:
```

Step 2: Enter each item of information as it is asked. This information will include the following:

Directory path name - The local directory you want to share.
Resource identifier - The name you assign to the resource.
Remote permissions - Read-only or read/write permission.
Description - Up to 32-character description of the resource.
Client list - The machines that can use the resource.

You will also be asked if you want to advertise the resource immediately if RFS is running, the path name exists, and the resource is not being used.

The **sysadm advauto** command sets up a resource to be automatically advertised by adding a complete **adv**(1M) command line in the **/etc/rstab** file. See Chapter 10 for information on the **adv** command.

Remove Automatic Advertises

Step 1: Type **sysadm unadvauto** to remove an entry from the Automatic Advertise List. You will see the following:

```
        REMOVE LOCAL RESOURCES FROM AUTOMATIC ADVERTISE LIST

When you no longer want to share a resource with client machines
you unadvertise it.  Resources that have been set up to be advertised
automatically, however, will be readvertised by the system the next
time RFS is started unless they are removed from the Automatic
Advertise List.

This subcommand lets you remove resources from the Automatic Advertise List.
NOTE: This subcommand does NOT unadvertise immediately.  To unadvertise
immediately use "unadvnow".

Enter the resource identifier of the resource to be removed from the
Automatic Advertise List [?, q]:
```

Step 2: Type the resource identifier of the resource you want to remove.

The **sysadm unadvauto** command removes the automatic advertise by deleting an **adv**(1M) command line from the **/etc/rstab** file.

Advertise Immediately

Step 1: Type **sysadm advnow** to advertise a local resource immediately. You will see the following:

```
                ADVERTISE LOCAL RESOURCES IMMEDIATELY

You share your local resources with other machines by advertising them.
The machines that can use the resources are called clients.

This subcommand lets you immediately advertise resources.  As soon
as a resource is advertised it is available to client machines.
You will be asked for all the information needed.

NOTE:  This subcommand does NOT add resources to the Automatic
Advertise List.  Use the subcommand "advauto" to add resources
to this list.

Enter the full path name of the local directory to
advertise [?, q]:
```

Step 2: Enter each item of information as it is asked. This information will include the following:

Directory path name - The local directory you want to share.
Resource identifier - The name you assign to the resource.
Remote permissions - Read-only or read/write permission.
Description - Up to 32-character description of the resource.
Client list - The machines that can use the resource.

Unadvertise Immediately

Step 1: Type **sysadm unadvnow** to unadvertise a local resource immediately. You will see the following:

```
                UNADVERTISE LOCAL RESOURCES IMMEDIATELY

When you no longer want to share a resource with client machines you
unadvertise the resource.  This subcommand lets you unadvertise currently
advertised resources.

When a resource is unadvertised, no new clients may access it.
Unadvertising does not take a resource away from client machines that
have already mounted it.  If any clients have mounted the resource you are
unadvertising, the subcommand lets you force the resource to be unmounted.

NOTE:  This subcommand does NOT remove resources from the Automatic
Advertise List.  Use the subcommand "unadvauto" to remove resources
from this list.

Enter the resource identifier of the resource to be
unadvertised [?, q]:
```

Step 2: Type the name of the resource you want to unadvertise.

List Remotely Mounted Resources

Step 1: Type **sysadm lsinuse** to list your resources that are currently mounted and in use on remote machines. The following is an example of the output.

```
        LOCAL RESOURCES CURRENTLY MOUNTED BY CLIENT MACHINES

The fields in this list are in the following order:
Resource identifier, Local path name, Client name.
------------------------------------------------------------------------------

CROOT    /       peanuts.snoopy peanuts.linus
CDEV    /dev     peanuts.lucy peanuts.patti
```

List Locally Advertised Resources

Step 1: Type **sysadm lsadv** to list the local resources that are currently or automatically advertised. The following is an example of the output.

```
                ADVERTISED LOCAL RESOURCES

The fields in this list are in the following order:
Resource Identifier, Local Pathname, Client Access Permissions,
Resource Status, Resource Description, Client List.

The status of a resource is "C" (Current) if RFS is running and the
resource is now available to client machines.  The status is "A" (Automatic)
if the resource is in the Automatic Advertise List.  A resource may be both
"A" and "C".  Please wait for the list ...
------------------------------------------------------------------------


CROOT /    read/write A/C "Charlie root file system" peanuts.linus peanuts.snoopy
CDEV  /dev       read/write C "Charlie device directory"    unrestricted
```

Procedure 10.4: Remote Resource Mounting (mountmgmt)

Purpose	Mount remote resources on your machine.
Starting Conditions	System state—2 multiuser or 3 (RFS state). Login—**root**.
Commands	**sysadm mountmgmt** **sysadm mntauto** **sysadm unmntauto** **sysadm mntnow** **sysadm unmntnow** **sysadm lsavail** **sysadm lsmount**
References	Chapter 10, "Remote File Sharing."

You can attach another machine's advertised resource to your file system using the **sysadm mountmgmt** subcommands.

Prerequisites

Before you run **sysadm mountmgmt** subcommands, you should do the following:

- Set up RFS. Use the **sysadm setuprfs** to enter the basic information needed to run RFS.
- Start RFS. To use **mntnow**, **unmntnow**, or **lsavail** subcommands, RFS should be running. The other subcommands do not require that RFS be running to provide useful information.

The following chart describes the **sysadm mountmgmt** subcommands and lists dependent commands and files of the procedures in this section. This information should help you relate the **sysadm** processing to the descriptions of RFS components in Chapter 10.

sysadm Commands	Dependent Commands	Files	Description
mntauto	**mount -d**(1M) **nsquery**	**/etc/fstab**	Set up a remote resource to be mounted each time RFS is started. The information you enter for each resource will be added to the Automatic Mount List.
mntnow	**mount**(1M) **nsquery**(1)		Mount a resource immediately.
unmntauto		**/etc/fstab**	Remove a remote resource from the Automatic Mount List. Once removed, the resource will no longer be mounted when RFS is started.
unmntnow	**umount**(1M) **fuser -k**(1M) **mount**(1M)		Immediately unmount a resource and kill all local processes associated with it.
lsavail	**nsquery**(1M)		Print a list of all remote resources available for you to mount.
lsmount	**mount -d**(1M)	**/etc/fstab**	Print a list of all your currently-mounted and automatically-mounted resources.

Figure P10-5: The **sysadm mountmgmt** Subcommands

Mount Automatically

Step 1: Type **sysadm mntauto** to add an entry to the Automatic Mount List. You will then see the following:

```
ADD REMOTE RESOURCES TO AUTOMATIC MOUNT LIST

To use remote resources that have been made available to you from
remote machines, you mount the resources on local directories.
Users on your machine can then access the resources as if they were local.

This subcommand lets you add remote resources to the Automatic Mount List.
Your system will attempt to mount these resources each time
RFS is started.  If RFS is running on your machine, you may also be able
to immediately mount the resources you have added to the Automatic Mount
List.

Enter the full path name of the local directory to be
used as the mount point [?, q]:
```

Step 2: Enter each item of information as it is asked. This information will include the following:

Local mount point - The local directory to use as a mount point.
Resource identifier - The name of the remote resource.
Local permissions - The permissions (read-only or read/write).

The **sysadm mntauto** command sets up the automatic mount by adding information to the **/etc/fstab** file that is needed with the **mount**(1M) command. See Chapter 10 for information on the **mount** command.

Once the resource is added, you will be asked if you want to mount the resource immediately if the following are true: RFS is on, the mount point exists, the mount point is not busy, and the resource is not already in use.

Note: When using the **sysadm** interface, you can only mount resources available from your own domain. To mount resources available from other domains, see the "Remote Resource Mounting" section of Chapter 10 for information.

Remove Automatic Mounts

Step 1: Type **sysadm unmntauto** to remove an entry from the Automatic Mount List. You will see the following:

```
REMOVE REMOTE RESOURCES FROM THE AUTOMATIC MOUNT LIST

When you no longer want to share remote resources you can unmount them
from your file system.  If the resources have been added to the Automatic
Mount List and you do not want the system to try to mount them the next
time RFS is started, you use this subcommand to remove the resources
from the Automatic Mount List.

NOTE: This subcommand does not immediately unmount resources.
Use the subcommand "unmntnow" to immediately unmount resources.

Enter the resource identifier of the remote resource to be removed
from the Automatic Mount List [?, q]:
```

Step 2: Type the resource identifier you want to remove.

The **sysadm unmntauto** command removes the automatic mount by deleting the mount information for the resource from the **/etc/fstab** file.

Note: Using the **sysadm** interface, you can only remove entries for resources within your own domain.

Mount Immediately

Step 1: Type **sysadm mntnow** to mount a remote resource immediately. You will see the following:

```
                MOUNT REMOTE RESOURCES IMMEDIATELY

To use remote resources that have been made available to you from
other machines you mount the resources on local directories.
Users on your machine can then access the resources as if they were local.

This subcommand lets you immediately mount remote resources when RFS is
running on your machine.

NOTE:  This subcommand will not add resources to the Automatic Mount List.
Use the subcommand "mntauto" to add resources to this list.

Now checking for available remote resources.  Please wait ...

Enter the full path name of the local directory to be
used as the mount point [?, q]:
```

Step 2: Enter each item of information as it is asked. This information will include the following:

Local mount point - The local directory to use as a mount point.
Resource identifier - The name of the remote resource.
Local permissions - The permissions (read-only or read/write).

Unmount Immediately

Step 1: Type **sysadm unmntnow** to unmount a remote resource immediately. You will see the following:

```
                UNMOUNT REMOTE RESOURCES IMMEDIATELY

When you no longer want to use remote resources you can unmount them from
your file system.  If RFS is running and the resources are currently mounted
you use this subcommand to immediately unmount the resources.

NOTE: This subcommand does not remove the resources from the Automatic Mount
List. Use the subcommand "unmntauto" to remove resources from this list.

Enter the resource identifier of the remote resource
to be unmounted [?, q]:
```

Step 2: Type the name of the resource you want to unmount. The system will kill all local processes currently accessing the resource and then unmount the resource. (If your current directory is within the resource, you will be logged off.)

List Available Remote Resources

Step 1: Type **sysadm lsavail** to list Remote Resources that are available for you to mount from other members of your domain. The following is an example of the output.

```
                REMOTE RESOURCES CURRENTLY AVAILABLE

Remote resources are available if they have been advertised by other
machines.  Resources can only be available when RFS is running on your
machine.

The fields on this list are in the following order:
Resource identifier, Access permissions, Advertising machine, Description.
Please wait for the list...
-------------------------------------------------------------------------

LROOT           read/write  peanuts.linus            Linus root file system
```

List Locally Mounted Resources

Type **sysadm lsmount** to list the remote resources that are currently or automatically mounted on your system. The following is an example of the output.

```
                REMOTE RESOURCES MOUNTED

This list shows remote resources that are mounted or set up to be mounted
on your machine.  The order of the fields is:  Resource identifier,
Local mount-point, Local access permissions, Resource status.

The status of a resource is "C" (Current) if RFS is running and the remote
resource is mounted on a local directory.  The status is "A" (Auto) if
the remote resource is on the Automatic Mount List.  Resources may be both
"C" and "A".   Please wait for the list...
--------------------------------------------------------------------------

LROOT /usr/Lroot read-write A/C
```

Procedure 10.5: Change RFS Configuration (confgmgmt)

Purpose	Change ID mapping, show current RFS configuration, or update domain member list.
Starting Conditions	System state—2 multiuser or 3 (RFS state). Login—**root**.
Commands	**sysadm confgmgmt** **sysadm showconfg** **sysadm idmapping** **sysadm lsmember** (primary only) **sysadm addmember** (primary only) **sysadm delmember** (primary only)
References	Chapter 10, "Remote File Sharing."

The **sysadm confgmgmt** subcommands let you perform several separate RFS configuration tasks. The tasks include listing basic RFS configuration information and setting up basic ID mapping strategies. If your machine is a primary name server, commands are also available to maintain the domain member list.

Each of these commands is also available through the **sysadm setuprfs** command. The advantage of using them separately is that you don't have to go through the whole **setuprfs** procedure if you are only interested in one function.

Prerequisites

Before you run **sysadm confgmgmt** subcommands, you should do the following:

Set up RFS. Use the **sysadm setuprfs** to enter the basic information needed.

The following chart describes the **sysadm confgmgmt** subcommands and lists dependent commands and files. This information should help you relate the **sysadm** processing to the descriptions of RFS components in Chapter 10.

sysadm Commands	Dependent Commands	Files	Description
showconfg	**uname**(1) **dname**(1M) **dname -n**(1M)	**rfmaster** *	List the following RFS information for your domain: Your machine's node name. The name of your machine's domain. The Transport Provider used by RFS. The primary and secondary name servers' names and network addresses.
idmapping	**idload**(1M)	**uid.rules** † **gid.rules** †	Choose one of three basic strategies for defining the permissions remote users will have to your resources.
addmember	**rfadmin -a** (1M)	*dom*/**passwd** †	Add a machine and its password to the domain member list. This is only used on the primary name server machine. (*dom* is replaced by the domain name.)
delmember	**rfadmin -r** (1M)	*dom*/**passwd** †	Delete a machine and its password from the domain member list. This is only used on the primary name server machine. (*dom* is replaced by the domain name.)
lsmember		*dom*/**passwd** †	List all machines in the domain member list, noting those that are primaries and secondaries. This is only used on the primary name server machine. (*dom* is replaced by the domain name.)

* File appears in the directory **/usr/nserve**.
† File appears in the directory **/usr/nserve/auth.info**.

Figure P10-6: The **sysadm confgmgmt** Subcommands

Show RFS Configuration

Type **sysadm showconfg** to show RFS configuration information for your machine. The following is an example of the output.

```
        CURRENT RFS CONFIGURATION

Node name:                      charlie
Domain name:                    peanuts
Transport provider:             starlan
Primary name server:            charlie
Address of primary:             charlie.serve
Secondary name server:          linus
Address of linus:               linus.serve
```

Choose ID Mapping Scheme

Note: The **sysadm idmapping** command provides you with the choice of three of the most common ways of defining remote users' permissions to your resources. For more complex methods of ID mapping, such as mapping by name or mapping by individual machines, Chapter 10 describes how to manually map IDs.

Caution: **Do not use the ID mapping available through the *sysadm* interface if you want to keep any ID mapping you enter manually from descriptions in Chapter 10.**

Step 1: Type **sysadm idmapping** to change or list the user and group ID mapping for your machine. You will see the following:

```
            USER ID AND GROUP ID MAPPING

Every machine in a domain defines how remote users will be allowed
to access its local resources.  You define this access by setting
up a mapping table of how remote user IDs and remote group IDs will
be mapped to local user and group IDs.

Since setting up mapping can be a complex procedure, this subcommand
allows you to choose from three pre-defined mappings.
Selecting option a, b, or c will replace any current mapping that
you may have previously set up.  To set up mappings not defined
by this subcommand see your System Administrator's Guide.

Do you need a more detailed explanation of ID mapping? [y, n, q]
```

Step 2: Type **y** if you want more details, **n** if you don't. After the detailed explanation (if requested) you will see the following:

```
Choose option desired for uid/gid mappings:
  a) all remote IDs map to guest ID with "other" permissions.
  b) remote IDs map directly to same local IDs except uids 0-99, gids 0-10.
  c) remote IDs map directly to same local IDs except uid 0, gid 0 (root).
  d) show current ID mapping.
[a, b, c, d, ?, q]:
```

Step 3: Since user ID and group ID mapping can be complex, this interface presents you with several standard options. All of these options represent global mapping schemes. This means that the same mapping rules will be applied to users of all machines that use your resources. For information on other mapping schemes, see the "Mapping Remote Users" section of Chapter 10.

Type a **?** if you need more information on one of the three basic mapping types. Then type one of the three mapping choices (**a**, **b**, or **c**) or type **d** to show the current ID mapping.

The following paragraphs describe the three mapping choices.

a - This is usually the best choice since all remote users on all machines that share your resources will be assigned to a special guest login ID. It provides the maximum security and the minimum complexity.

b or **c** - These options are only valuable if you have identical **/etc/password** and **/etc/group** files among machines with which you share resources. The difference between **b** and **c** is that **b** protects all administrative logins and **c** only protects the **root** login.

Add Domain Members

Each time a machine is added to an RFS network, the machine's node name and password must be entered on the primary name server machine.

Step 1: Type **sysadm addmember** to add a machine to the domain member list. You will see the following:

```
                ADD MEMBER MACHINES TO YOUR DOMAIN

Every machine must be a member of a domain. The primary name server machine
for a domain maintains the list of machines that are members of its domain.

This subcommand allows the primary name server to add machines to the
list of domain members. The addition takes effect immediately. When
a machine is added as a member of a domain it may share resources
available in that domain.

Enter the node name of a machine to be added as a member of your
domain [?, q]:
```

Step 2: Type the node name of the machine you want to add to the domain.

Step 3: Type a password for the machine you want to add to the domain. This password should match the password entered by the machine when it first starts RFS. This is the password the machine will use when it wants to enter your domain. (The password may simply be a carriage return.)

Delete Domain Members

Step 1: From the primary name server machine, type **sysadm delmember** to delete a machine from the domain member list. You cannot delete primary or secondary name servers. You will see the following:

```
DELETE MEMBER MACHINES FROM YOUR DOMAIN

The primary name server machine for a domain maintains the list of
machines that are members of its domain.

This subcommand allows the primary name server to delete
machines from the list of domain members.  The deletion takes
effect immediately.  When a machine is deleted from a domain it may
no longer access resources available in that domain.

Enter the node name of a machine to be added as a member of your
domain [?, q]:
```

Step 2: Type the name of the machine you want to delete from the domain member list.

List Domain Members

Step 1: From the primary name server machine, type **sysadm lsmember** to list machines in the domain member list. The following is an example of the output you will see.

```
MEMBERS OF DOMAIN: peanuts

Node name: charlie          Primary name server
Node name: linus            Secondary name server
Node name: lucy
Node name: patti
Node name: snoopy
```

Part 2: Support

Chapter 1: System Identification and Security

Introduction

This chapter has information on the security of your 3B2 computer system.

- Important security guidelines

  Guidelines for setting passwords and permissions; protecting the system from unauthorized access.

- Logins and passwords

  Aging passwords to control the amount of time passwords may be kept for logins.

  Locking logins to prevent their being used.

  Protecting administrative commands and logins with passwords.

  Implementing dial-up passwords.

  Implementing unsuccessful log in logging.

- Console Logger

  Logging all Input/Output (I/O) at the console.

- Set-User Identification (set-UID) and set-Group Identification (set-GID)

  Preventing unauthorized use of programs conditioned to execute via an administrative login.

Important Security Guidelines

The security of the system is eventually the responsibility of all who have access to the system. No system is totally secure. The system is not tamperproof. Some of the items to consider are as follows:

- Especially with a small computer, physical access to the machine must be considered.
- Set the access permissions to directories and files to allow only the necessary permissions for owner, group, and others.
- All logins should have passwords. Change passwords regularly. Do not pick obvious passwords. Six-to-eight character nonsense strings using letters and numbers are recommended over standard names. Logins that are not needed should be either removed or blocked.
- Dial-up ports that do not have logins usually cause trouble. Use dial-up passwords to provide increased access security.
- Any system with dial-up ports is not really secure. Sensitive information should not be kept on a system with dial-up ports.
- Users who make frequent use of the **su** command can compromise the security of your system by accessing files belonging to someone else without the other person's knowledge. The **su** command is also dangerous since you must know another user's login and password to use it. The more people who know a given login and password, the less secure access is to the system. Therefore, a log is kept on the use of the command. Check the file **/usr/adm/sulog** to monitor use of the **su** command. The format of **/usr/adm/sulog** is described in Appendix B, "Directories and Files."
- Login directories, **.profile** files, and files in **/bin**, **/usr/bin**, **/lbin**, 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**), provides protection for sensitive information. The **crypt** command is installed by the Security Administration Utilities package (domestic customers only).
- Log off the system if you must be away from the data terminal. Do not leave a logged-in terminal unattended, especially if you are logged in as **root**.

Logins and Passwords

The discussion of logins and passwords covers:

- Shadow password feature
- Displaying password status and aging information
- Password aging
- Locking unused logins
- Special administrative logins
- Dial-up passwords
- Logging unsuccessful log-in attempts.

Shadow Password Feature

The shadow password feature provides increased security by masking the encrypted password field in the **/etc/passwd** and moving the password information to an access-restricted file called the shadow password file (**/etc/shadow**). With the shadow password feature enabled, unfriendly users cannot obtain the encrypted password field to see what logins have no password or to try to "crack" the passwords by using password generation programs.

The shadow password feature is enabled by the **pwconv**(1M) command and is disabled by the **pwunconv**(1M) command. When converting between the shadow password environment and the single password file environment, note that the password aging information is based on days in the shadow password environment and on weeks in the single password environment. This means that a value of 30 days in the shadow password environment is rounded up to 35 days (5 weeks) in the single password environment. Therefore, converting from the shadow password environment to the single password file environment and back again will round all intervals up to modulo 7 days (weeks). Under these conditions, the maximum interval is 63 weeks or 441 days.

The consequences of disabling, enabling, and then disabling shadow password is shown in the following screen display.

```
# pwconv
# passwd -n8 -x30 rar
# passwd -sa rar
rar  PS  03/31/88  8  30
# pwunconv
# passwd -s rar
rar  PS  03/31/88  14  35
# pwconv
# passwd -s rar
rar  PS  03/31/88  14  35
#
```

Displaying Password Status and Aging Information

The **passwd -sa** command outputs password status and password aging information for all logins. The **passwd -s** *name* command outputs password status and password aging for the *name* login. The format of the password status and aging information output by these commands is as follows.

name status last_changed minimum maximum

name	The *name* field is the login name.
status	The *status* of a password can be **LK** (lock string), **NP** (no password), or **PS** (password string).
last_changed	This field contains the date when the password was last modified. A date of 00/00/00 means that the user has yet to log in and change the password for which password aging has been applied for the first time.
minimum	The number of days required between password changes.
maximum	The number of days the password is valid. This is also the number of days between the password expiration date and the *last_changed* field.

Password Aging

In general, password aging controls the amount of time passwords may be kept for logins. The password aging mechanism forces users to change their password on a periodic basis. Realistically, password aging forces a user to adopt at least two passwords for a login. Provisions can be made to prevent a user from changing a new password before a specified interval or to prevent a user from ever changing the password.

The **passwd**(1) command is used to selectively apply password aging to logins and to display password status and password aging information.

The relationship of the *last_changed, minimum,* and *maximum* fields determine how password aging is applied to the login.

- When the *maximum* field is greater than the *minimum* field, the password will expire on the *maximum* days from the *last_changed* date. Also, the user cannot change the password for the *minimum* days from the *last_changed* date.
- When *maximum* and *minimum* fields are equal to zero, the user is forced to change the password at the next login. No further password aging is then applied to the login.
- When *minimum* field is greater than *maximum* field, only **root** is able to change the password for the login.
- When *maximum* field is set to a **-1**, password aging is turned off for the login.

Locking Unused Logins

If a login is not used or needed, you should do one of two things:

- Remove the login from system
- Disable (lock) the login.

A login is removed from the password file(s) **/etc/passwd** and **/etc/shadow** by using the **passmgmt -d** *name* command. The home directory structure or any files owned by the *name* user are not removed by the **passmgmt -d** command. To remove a login and the associated login directory structure from the system, use the **sysadm deluser** command.

A login is locked by using the **passwd -l** *name* command.

Special Administrative Passwords

There are two familiar ways to access the system: either via 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 special system logins and administrative commands that can be password-protected (see Procedure 1.5, "Assign Passwords to Administrative and System Logins," for information on doing this). The administrative commands, which are also logins, do functions that might be needed by several users on your 3B2 computer.

The administrative commands that can be password-protected are listed below:

Function	Use
setup	This command is used to set up the 3B2 computer. Once the machine has been set up, you do not want anyone doing it again without your knowledge.
sysadm	This command allows access to many useful administrative functions that do not require a user to log in as root.
powerdown	This command powers the computer down.
checkfsys	This command starts a file system check on the specified file systems.
makefsys	This command makes a new file system on the specified media.
mountfsys	This command mounts the specified removable media file system for use.
umountfsys	This command unmounts the specified, previously mounted file system.

The administrative commands 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 commands, the system will execute the command after login and exit to the **login** prompt once you quit or complete the function performed by the command.

If an administrative command is assigned a password, any user who attempts to log in using one of these commands or tries to execute one of these commands from the shell is prompted for the password.

The system and other important logins are listed in the table below:

Login	Use
root	This login has no restrictions on it and it overrides all other logins, protections, and permissions. It allows the user access to the entire operating system. The password for the **root** login should be carefully protected.
sys	This login has the power of a normal user login over the files it owns.
bin	This login has the power of a normal user login over the files it owns, which are in **/bin**.
adm	This login has the power of a normal user login over the object files it owns, which are in **/usr/adm**.
uucp	This login owns the object and spooled data files in **/usr/lib/uucp**.
nuucp	This login is used by remote machines to log into the system and start file transfers via **/usr/lib/uucp/uucico**.
daemon	This is the login of the system daemon, which controls background processing.
trouble	This login has the power of a normal user login over the files it owns, which are in **/usr/lib/trouble**.
lp	This login owns the object and spooled data files in **/usr/spool/lp**.
vmsys	This login owns the **/usr/vmsys** directories and files. It is used to administer the AT&T FACE Utilities.
oasys	This login owns the **/usr/oasys** directories and files. It is used to administer the AT&T FACE Utilities.

Most of the administrative commands and system logins allow a user access to critical portions of the operating system. Therefore, it is recommended that you assign passwords to these commands/logins and that only a few people know the passwords.

Dial-Up Passwords

Additional machine access security can be applied to any port (CONSOLE, CONTTY or tty) by requiring a second password under certain conditions. The second password is called a dial-up password. Dial-up passwords are assigned to specific ports on the basis of what shell is called by the login process. Dial-up passwords are therefore assigned to a login shell process. The dial-up password is prompted for when a login process running on one of the ports identified in the **/etc/dialups** file calls a shell identified in the **/etc/d_passwd** file.

The **/etc/dialups** and **/etc/d_passwd** files must be created to establish dial-up passwords. These files should have restricted access permissions. Others should not be able to access these files. Two examples of restricted permissions are shown in the following screen display.

```
# ls -la /etc/dialups /etc/d_passwd
-rw-r-----   1 root     sys           70 Mar 19 12:17 /etc/d_passwd
-rw-r-----   1 root     sys           36 Mar 19 12:17 /etc/dialups
                OR
# ls -la /etc/dialups /etc/d_passwd
-r--------   1 root     sys           70 Mar 19 12:17 /etc/d_passwd
-r--------   1 root     sys           36 Mar 19 12:17 /etc/dialups
#
```

/etc/dialups File

The **/etc/dialups** file contains a list of the ports that require additional security. Comments can be added to entries in the file using a pound sign (#). All characters on a line following a pound sign are ignored. The following sample **/etc/dialups** file specifies the **contty** and **tty21** ports as dial-up ports. A dial-up password is NOT applied to port **/dev/tty31** because of the pound sign.

```
#  cat /etc/dialups
/dev/contty      #COMMENT
/dev/tty21       #COMMENT
#/dev/tty31
#
```

/etc/d_passwd File

The **/etc/d_passwd** file identifies the shells and their associated encrypted passwords. The following sample **/etc/d_passwd** file shows a password assigned to **/bin/sh** and no password assigned to **/usr/lib/uucp/uucico** login shell processes.

```
# cat /etc/d_passwd
/bin/sh:yzPD4VPGeJk4U:
/usr/lib/uucp/uucico::
#
```

d_passwd Entry Creation Program—dpass

The following C Language program source code is provided to simplify the creation of the entries for the **/etc/d_passwd** file. The program is called **dpass**. After entering the source code into a file called **dpass.c**, the program is compiled using the following command:

```
cc -O dpass.c -o dpass
```

```
# cat dpass.c
#include <stdio.h>
#include <errno.h>
#include <pwd.h>

#define PWDSZ   10                          /*size of the passwd*/
#define PROMPT  "Enter Dialup passwd: "     /*first prompt for passwd*/
#define PROMPT2 "Enter again to verify: "   /*verification prompt*/
#define TPASS   "/etc/d_pass.add"           /*file to hold new passwd*/
#define SEED    "yz"                        /*seed for crypt()*/
#define EOS     '\0'                        /*end of string*/

main(argc,argv)
char *argv[];
{
    FILE    *tpass;                         /*temp d_passwd file*/
    char    *cpass,                         /*points at encrypted password*/
            pass[PWDSZ],                    /*holds password typed in*/
            pass2[PWDSZ],                   /*holds verification password*/
            *crypt();                       /*encrypts password*/

    /*      Usage is "dpass loginsh" where loginsh is a shell that
    **      will need a password.  For example:
    **
    **              $ dpass /bin/sh
    **              $ dpass /usr/lib/uucp/uucico
    **              $ dpass /bin/ksh
    **
    **      if no loginsh is specified, display usage and exit.
    */
    if((strcmp(argv[1],EOS))==0)
    {
        fprintf(stderr,"\nusage: dpass loginsh\n\n");
        exit(1);
    }
```

Continued

Figure 1-1: dpass.c—d_passwd Entry Creation Program (Sheet 1 of 2)

Continued from previous screen display

```
    strcpy(pass,getpass(PROMPT));           /*prompt for password*/
    strcpy(pass2,getpass(PROMPT2));         /*prompt again - verify*/
    /*
    **      if they don't match, notify, and loop until they do.  This
    **      can be broken out of by typing <BREAK> or <DELETE>.
    */

    while(strcmp(pass,pass2)!=0)
    {
        fprintf(stderr,"\nPasswords don't match, try again ....\n\n");
        strcpy(pass,getpass(PROMPT));
        strcpy(pass2,getpass(PROMPT2));
    }
    cpass=crypt(pass,SEED);                 /*encrypt passwd*/
    if((tpass=fopen(TPASS,"w"))==NULL)
    {
        perror("error opening temp dpass file!");
        fprintf(stderr,"Errno: %d\n",errno);
        exit(1);
    }
    fprintf(tpass,"%s:%s:\n",argv[1],cpass);   /*write out to file*/
    fclose(tpass);
}
```

Figure 1-1: dpass.c—d_passwd Entry Creation Program (Sheet 2 of 2)

Sample /etc/d_passwd Entry Creation

The following screen display shows how to create an entry for use in the **/etc/d_passwd** file that defines a dial-up password for the **/bin/sh** login shell.

```
# dpass /bin/sh
Enter Dialup passwd:
Enter again to verify:
# cat /etc/d_pass.add
/bin/sh:yzhEOvHMuxlEI:
#
```

The **dpass** program puts its output in the **/etc/d_pass.add** file. The **/etc/d_passwd** is then edited to include the information in the **/etc/d_pass.add** file.

Logging Unsuccessful Login Attempts

As the system is delivered, five consecutive unsuccessful login attempts on a given port cause the **login** process to sleep for 20 seconds and then drop the line.

The occurrence of repetitive unsuccessful attempts to access the system (log in) can be tracked (logged). Fewer than five consecutive unsuccessful login attempts on a given port (CONSOLE, CONTTY, or tty) are not logged. As the system is delivered, this logging mechanism is turned off. To turn on the mechanism that logs unsuccessful attempts to access the system, the **/usr/adm/loginlog** file must be created. This file should have read and write permissions for only **root**. To turn off unsuccessful login logging, remove the **/usr/adm/loginlog** file.

The format of the entries in this file are in the following form.

login_name:port:time

The following screen display shows the contents of a typical **/usr/adm/loginlog** file resulting from one occurrence of five consecutive unsuccessful login attempts on the **/dev/contty** port.

```
# cat /usr/adm/loginlog
rar:/dev/contty:Thu Mar 31 05:19:50 1988
rar:/dev/contty:Thu Mar 31 05:19:57 1988
rar:/dev/contty:Thu Mar 31 05:20:05 1988
cms:/dev/contty:Thu Mar 31 05:20:16 1988
cms:/dev/contty:Thu Mar 31 05:20:24 1988
#
```

Console Logger

The console logger records all console I/O and saves it in a file. All **stdin**, **stdout**, and **stderr** are sent to a buffer. When the UNIX System V operating system is booted, the console logger is not active. A daemon will read the buffer and write the contents to a file called **conlog**. The **conlog** file is in **/usr/adm**.

The console logger is a method of recording all the activity that goes on at the console. The logger is not a replacement for the console printer. The local procedures that you have established determine if you need a hard copy of the console activities. If the console terminal is available for any user, it is suggested that the console logger be running.

The console logger is started by the **conslog**(1M) command, using the **-a (conslog -a)** option. To deactivate the console logger, use the **-d** option of the **conslog** command. To read the contents of the console log, use the **-r (conslog -r)** option. You should be logged in at the console to read the **conlog** file. Every time the console logger is started, the old log file **(/usr/adm/conlog)** is moved to **/usr/adm/conlog***mmddhhmm*. If you do not need this old console log file, you should remove it.

The console log file will grow until the UNIX system file size limit is reached, at which time an error message is displayed. Because of this the log needs to be moved periodically. You can do this by simply deactivating the logger and then activating the logger.

Caution: Deactivate the console logger before executing commands that put the console in raw mode (for example, *pg, vi*), or commands that are required to be attached to a tty device when the console logger is activated. Do not *cat* the conlog file from the console with the console logger active. Use the *conslog -r* command to read or look at the file when your at the console terminal with the console logger active.

Set-UID and Set-GID

The set-UID and set-GID bits must be used carefully if any security is to be maintained. 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 of the executable file.

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**. The following paragraphs provide a few examples of command lines that can be used to identify the files with a set-UID.

For more information about the set-UID and set-GID bits, see **chmod**(1) and **chmod**(2).

Check Set-UIDs Owned By Root

The following command line lists all set-UID programs owned by **root**.

```
# find / -user root -perm -4100 -exec ls -l {} \; | mail root&
```

The results of running this command are mailed to **root**. All files are checked by this command starting at **/**. Any surprises in the mail of **root** should be investigated.

```
you have mail
# mail
From root Mon Mar 28 13:17 EDT 1988
-r-sr-xr-x   1 root     root       18252 Mar 11 09:01 /etc/prtconf.d/scsi
-r-sr-xr-x   1 root     sys        40108 Feb 24 09:05 /etc/scsi/mirror
-r-sr-xr-x   1 root     bin         8952 Mar 11 09:01 /etc/devnm
-r-sr-xr-x   1 root     bin        45664 Jan 19 11:03 /usr/bin/at
-r-sr-xr-x   1 root     bin        22168 Jan 19 11:03 /usr/bin/crontab
-r-sr-xr-x   1 root     bin        29308 Jan 19 11:03 /usr/bin/shl
---s--x--x   1 root     uucp       49424 Jan 19 12:29 /usr/bin/ct
-r-sr-xr-x   1 root     bin        20998 Feb  2  1987 /usr/bin/layers
-r-sr-xr-x   1 root     sys        14368 Mar 11 09:01 /usr/lib/mv_dir
-r-sr-xr-x   1 root     bin       205412 Jan 20 14:10 /usr/lib/lpsched
---s--x--x   1 root     sys        20048 Jan 19 12:33 /usr/lib/uucp/nttysrv
-r-sr-xr-x   1 root     bin         5616 Feb  2  1987 /usr/lib/layersys/relogin
---s--x---   1 root     rar        45376 Mar 13 15:11 /usr/rar/bin/sh
-r-sr-xr-x   1 root     bin         8952 Mar 11 09:01 /bin/df
-r-sr-xr-x   1 root     bin        27668 Mar 11 09:01 /bin/login
-rwsr-xr-x   1 root     sys        11242 Mar 11 09:01 /bin/newgrp
-r-sr-sr-x   1 root     sys        25876 Mar 11 09:01 /bin/passwd
-r-sr-xr-x   1 root     sys        28284 Mar 11 09:01 /bin/su

? d
#
```

In this example, an unauthorized user (**rar**) has made a personal copy of **/bin/sh** and has made it set-UID to **root**. This means that **rar** can execute **/usr/rar/bin/sh** and become the super user.

Check Set-UIDs in the Root File System

The following command line reports all files with a set-UID in the root file system. The **ncheck**(1M) command, by itself, can be used on a mounted or unmounted file system. The normal output of the **ncheck -s** command includes special files. Here, the **grep** command is used to remove device files from the output. The filtering done in this example to remove the device files is applicable only for the root file system (**/dev/dsk/c1t1d0s0**). The output of the modified **ncheck** is used as an argument to the **ls** command. The use of the **ls** command is possible only if the file system is mounted.

```
# ls -l `ncheck -s /dev/dsk/c1t1d0s0 | cut -f2 | grep -v dev`
-r-sr-xr-x   1 root     bin         8952 Mar 11 09:01 /bin/df
-rwxr-sr-x   1 root     sys        37250 Jan 19 12:17 /bin/ipcs
-r-sr-xr-x   1 root     bin        27668 Mar 11 09:01 /bin/login
-r-xr-sr-x   1 bin      mail       41334 Mar 11 09:01 /bin/mail
-rwsr-xr-x   1 root     sys        11242 Mar 11 09:01 /bin/newgrp
-r-sr-sr-x   1 root     sys        25876 Mar 11 09:01 /bin/passwd
-r-xr-sr-x   1 bin      sys        21564 Mar 11 09:01 /bin/ps
-r-xr-sr-x   1 bin      mail       41334 Mar 11 09:01 /bin/rmail
-r-sr-xr-x   1 root     sys        28284 Mar 11 09:01 /bin/su
-r-sr-xr-x   1 root     root       18252 Mar 11 09:01 /etc/prtconf.d/scsi
-r-sr-xr-x   1 root     sys        40108 Feb 24 09:05 /etc/scsi/mirror
-r-xr-sr-x   1 bin      sys        12938 Jan 19 10:58 /etc/whodo
#
```

In this example, nothing looks suspicious.

Check Set-UIDs in Other File Systems

The following command line entry shows the use of the **ncheck** command to examine the **usr** file system (**/dev/dsk/c1t1d1s2**, assuming a system with default partitioning) for files with a set-UID. In this example, the complete path names for the files start with **usr**. The **usr** is not part of the **ncheck** output.

```
# ncheck -s /dev/dsk/c1t1d1s2 | cut -f2
/dev/dsk/c1t1d1s2:
/bin/mailx
/bin/at
/bin/crontab
/bin/shl
/bin/disable
/bin/enable
/bin/ct
/bin/cu
/bin/uucp
/bin/uuname
/bin/uustat
/bin/uux
/bin/layers
/bin/sadp
/lib/mv_dir
/lib/lpsched
/lib/mailx/rmmail
/lib/sa/sadc
/old/dev/contty
/lib/uucp/remote.unknown
/lib/uucp/nttysrv
/lib/uucp/uucico
/lib/uucp/uusched
/lib/uucp/uuxqt
/lib/layersys/relogin
/rar/bin/sh
#
```

In this example, **/usr/rar/bin/sh** should be investigated.

Chapter 2: User Services

Introduction

This chapter defines a variety of services to the users of your 3B2 computer system.

- User login administration

  Assigning user and group IDs to persons authorized to be users of your system. Maintaining the **/etc/passwd** and **/etc/group** files.

- User environments

  Setting up a master profile and helping users develop individual profiles. Establishing environment variables.

- User communications services

  Establishing and maintaining such services as message-of-the-day, news, mail.

- User requests

  Developing an organized plan for responding to user problems.

Login Administration

Add Users

Before users are permitted to log in to your system, they must be listed in the **/etc/passwd** file. The **adduser** selection from the **sysadm usermgmt**(1) menu leads you through a series of prompts that create an entry in the **/etc/passwd** file (see Procedure 2.1, "Add Users or Groups"). If you prefer, you can make the necessary changes to the **/etc/passwd** file using the **passmgmt**(1M) and **passwd**(1) commands. You need to log in as **root** to do this; **/etc/passwd** is generally installed as a read-only file. The **passmgmt** and **passwd** commands handle the processing of both the **/etc/passwd** and **/etc/shadow** password files. Do not use an editor to make changes to the password files. It is strongly recommended that the password files be changed using only **sysadm**, **passwd**, and **passmgmt** commands.

An entry in the **passwd** file consists of a single line with seven colon-separated fields. The colons are delimiters for the different fields, and the final colon must be there if there is nothing in the seventh field.

```
jqp:x:103:1:John Q. Public:/usr2/jqp:
```

The fields are as follows:

login name	A valid name for logging onto the system. A login name is from three to six characters; the first character must be alphabetic. It is usually chosen by the user.
password	The encrypted form of the password, if any, associated with the login name in the first field. All encrypted passwords occupy 13 bytes. The password can be a maximum of 8 characters. At least one character must be numeric. This is to discourage users from choosing ordinary words as passwords. When you add a user to the file you may use a default password, such as **passwd9**, and instruct the user to change it at the first login. Following the encrypted password, separated by a comma, there may be a field that controls password aging. See "Password Aging" in Chapter 1, "System Identification and Security." When the shadow password feature is enabled, the password field in the **/etc/password** file is mask and contains a lowercase

	letter "x" for all line entries. See "Shadow Password Feature" in Chapter 1, "System Identification and Security."
user id	The user-ID number (**uid**) is between 0 and 50,000. The number must not include a comma. Numbers below 100 are reserved. User-ID 0 is reserved for the super user. The System Administration Menu package does not permit you to specify a number below 100 when adding a user.
group id	The same conditions apply to the group-ID (**gid**) number as those stated for the **uid**, except that the group-ID 1 is reserved for the "other" group.
account	The account is the field for optional additional information about the user. There is no required format for this field.
home directory	The home directory is where the user is placed after logging in. The name is usually the same as the login name, preceded by a parent directory such as **usr2**. It is recommended to put home directories in file systems other than **usr.** This will make system management (back-up, upgrade etc.) easier by separating user files from the /usr file system. The **modadduser** command of the System Administration Menu allows you to specify the default parent directory. The home directory is the origination point of the user's directory tree.
program	This is the name of a program invoked at the time the user logs in. If the field is empty, the default program is **/bin/sh**. This field is most commonly used to invoke a special shell, such as **/bin/rsh** (restricted shell).

As noted above, the password field may contain a subfield that controls the aging of passwords. A description of how the process works can be found in Chapter 1, "System Identification and Security," and in **passwd**(4) in the *User's and System Administrator's Reference Manual.* The effect is to force users periodically to select a new password. If password aging is not implemented, a user can keep the same password indefinitely.

If you inspect the **/etc/passwd** file on your 3B2 computer, you will see several commands listed among the user login names. These are commands, such as **makefsys**(1M) and **sysadm**(1), that can have passwords assigned to them.

Change or Delete Password Entries

As with adding users, there are **sysadm usermgmt** menu selections for changing or deleting user entries from the **/etc/passwd** file (see Procedure 2.2, "Modify User or Group Information," and Procedure 2.3, "Delete Users or Groups"). You have the option, however, of using the **passmgmt**(1M) and **passwd**(1) commands to make the changes.

Occasionally a user will forget his or her password. When that happens (let's say to user **abc**), the user can log in as **root** and enter the following command:

```
# passwd abc     (The # prompt shows you are root.)
New password: passwd9    (The password entered is not echoed.)
Re-enter new password: passwd9
```

Since you did this as the super user (**root**), you were not prompted for the old password. The command changes **abc**'s password to **passwd9**. You should make sure the user changes the password immediately.

When you delete a login from **/etc/passwd** using the **sysadm usermgmt** menu, all the user's files and directories are removed. If you remove an entry from the **passwd** file using an editor, you have only removed the entry. The user's files remain.

Group IDs

Group IDs are a means of establishing another level of ownership and access to files and directories. Users with some community of interest can be identified as members of the same group. Any file created by a member of the group carries the group-ID as a secondary identification. By manipulating the permissions field of the file, the owner (or someone with the effective user-ID of the owner) can grant read, write, or execute privileges to other group members.

Information about groups is kept in the **/etc/group** file. A sample entry from this file is shown and explained below:

```
prog::123:jqp,abc
```

Each entry is one line; each line has the following fields:

group name	The group name can be from three to eight characters, the first of which must be alphabetic.
password	The password field should not be used.
group id	The group id is a number from 0 to 50,000. The number must not include a comma. Numbers below 100 are reserved.
login names	The login names of group members are in a comma-separated list. A login name should be a member of no more than one group. There is nothing to prevent a user from having more than one login name, however, as long as each is unique within the system. A user does not have to be in this file to be in the group.

The User's Environment

The key element in establishing an environment in which users can successfully communicate with the computer is the profile. Profiles are of two types:

1. The system profile.

   This is an ASCII text file, **/etc/profile**, that contains commands, shell procedures, and environment variables. Whenever a user logs in, the **login** process executes this file.

2. An individual user's profile.

   This is an executable commands file, **.profile**, that may reside in a user's home directory. The individual profile can contain additional commands and variables that further customize a user's environment. If one exists, it too is executed at login time, after the execution of **/etc/profile**.

A sample **/etc/profile** is shown in Figure 2-1.

```
#       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
        ;;
-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

        if [ ${LOGNAME} != root ]
        then
                news -n
        fi
        ;;
esac

umask 022
trap  2 3
```

Figure 2-1: A Default **/etc/profile**

Several interesting items are contained in the profile:

- Some environment variables are exported (see "Environment Variables" discussed later in this chapter).

- A file named **/etc/motd** is **cat**-ted (see "Message of the Day" discussed later in this chapter).
- If the user is not **root**, the names of news items are displayed **news -n** (see "news" discussed later in this chapter).
- If the user has mail (**mail -e**), a message about it is displayed (see "Mail" discussed later in this chapter).

For information on the shell programming commands used in Figure 2-1, see **sh**(1) in the *User's and System Administrator's Reference Manual.*

Environment Variables

An array of strings called the environment is made available by **exec**(2) when a process begins. Since **login** is a process that executes the individual's **.profile**, the array of environment strings is made available to it. An example of a typical array of strings is shown in Figure 2-2.

```
PS1=$
LOGNAME=abc
PWD=/usr/abc
HOME=/usr/abc
PATH=:/bin:/usr/bin:/usr/lbin
SHELL=/bin/sh
MAIL=/usr/mail/abc
TERM=4425
PAGER=pg
TZ=EST5EDT
TERMINFO=/usr/lib/terminfo
EDITOR=vi
```

Figure 2-2: Environment Array for a Typical User

The environment variables shown in Figure 2-2 give values to 12 names for user **abc**. Other programs make use of the information. For example, the user's terminal is defined as a 4425 (TERM=4425). When the user invokes the editor **vi**(1), **vi** checks the file referenced by TERMINFO (**/usr/lib/terminfo**) where it learns the characteristics of a 4425 terminal

(such as the 24-line screen). New strings can be defined at any time. By convention they are defined with the variable in uppercase, followed by an equal sign, followed by the value. Once defined, an environment variable can be made global for the user through the **export** statement. The individual **.profile** file can contain whatever the user wants.

umask

A system default controls the permissions mode of any files or directories created by a user. The 3B2 computer has default values of 644 for files and 755 for directories because of the umask specified in the **/etc/profile** file. That means everyone creating files automatically gets read and write permission. For directories, everyone gets read, write, and execute permission. (Execute permission on a directory means the ability to **cd** to the directory and to copy files from it.)

Users frequently set up a user mask in their **.profile** using the **umask**(1) command. The **umask** specifies the permissions to be denied. For example,

```
umask 027
```

denies group-write and other-read/write/execute permissions. Since the system default permissions are 666 for files and 777 for directories, a **022** user mask changes the file creation permissions to 644 (-rw-r--r--) and directory creation permissions 755 (drwxr-xr-x).

The execution of a **umask** command is always in reference to the system default creation permissions for files (666) and directories (777). The creation permissions established by the **umask** command in **/etc/profile** are therefore redefined by the execution of a subsequent **umask** command. Typically, users redefine the file and directory creation permissions in their **.profile** file.

Default Shell and Restricted Shell

Generally, when a user logs in, the default program that is started is **/bin/sh**. There may be cases, however, where a user needs to be given a restricted shell.

A restricted shell is one where the user is not allowed to do the following:

- Change directories.
- Change the value of $PATH.
- Specify path names or command names containing a slash (/). That is, the user of a restricted shell may not access files or directories other than the present working directory or those included in $PATH.
- Redirect output.

The restrictions are enforced after **.profile** has been executed.

The administrator can use a restricted shell strategy to limit certain users to the execution of a few commands or programs. By setting up a special directory for executables (**/usr/rbin**, for example), and controlling PATH so it only references that directory, the administrator can restrict the user's activity in whatever way is appropriate.

User Communications Services

Several ways of communicating with and among users are available in the UNIX System. Some of the most frequently used are described in this section.

Message-of-the-Day

Items of broad interest that you want to make available to all users can be put in the **/etc/motd** file. The contents of **/etc/motd** are displayed on the user's terminal as part of the login process. The login process executes a file called **/etc/profile**, which is an executable shell script that, among other things, commonly contains the command

cat /etc/motd

Any text contained in **/etc/motd** is displayed for each user each time the user logs in. For this information to be noticed by users, you must take care to use it sparingly and to clean out outdated announcements. A typical use for the Message-of-the-Day facility might be as follows:

```
5/30:  The system will be unavailable from 6—11pm
Thursday, 5/30 - preventive maintenance.
```

Part of the preventive maintenance should be to remove the notice from **/etc/motd**.

news

Another electronic bulletin board facility is the **/usr/news** directory and the **news**(1) command. The directory is used to store announcements in text files, the names of which are usually used to provide a clue to the content of the news item. The **news** command is used to print the items on your terminal.

The **/etc/profile** file is also used to inform users about news items. A typical **/etc/profile** contains the following line:

news -n

The **-n** argument causes the names of files in the **/usr/news** directory to be printed on a user's terminal as the user logs in. Item names are displayed only for current items, that is, items added to the **/usr/news** directory since

the user last looked at the news. The idea of currency is implemented like this: when you read a news item, an empty file named **.news_time** is written in your login directory. As with any other file, **.news_time** carries a time stamp indicating the date and time the file was created. When you log in, a comparison is made between the time stamp of your **.news_time** file and time stamp of items in **/usr/news**.

Unlike the Message-of-the-Day where users have no ability to turn the message off, **news** gives your users a choice of several possible actions:

read everything	If the user enters the command, **news** with no arguments, all news items posted since the last time the user typed in the command are printed on the user's terminal.
select some items	If the **news** command is entered with the names of one or more items as arguments, only those items selected are printed.
read and delete	After the **news** command has been entered, the user can stop any item from printing by pressing the DELETE key. Pressing the DELETE key twice in succession stops the program.
ignore everything	If the user is too busy to read announcements, they can safely be ignored. Items remain in **/usr/news** until removed. The item names will continue to be displayed each time the user logs in.
flush all items	If the user simply wants to eliminate the display of item names without looking at the items, a couple of techniques will work. However, these may not be appropriate to give out to users: **touch .news_time** updates the time-accessed and time-modified fields of the **.news_time** file thus, the currency mechanism of **news** sees that the time fields of **.news_time** are current. **news > /dev/null** prints the news items on the null device.

write to All Users

The ability to write to all logged-in users, via the **wall**(1M) command, is an extension of the **write**(1) command. It is fully effective only when used by the super user. While **wall** is a useful device for getting urgent information out quickly, users tend to find it annoying to have messages print out on their terminal right in the middle of whatever else is going on. The effect is not destructive, but is somewhat irritating. Many users guard against this distraction by including the following command:

mesg n

in their **.profile**. This blocks other ordinary users from interjecting a message into your **stdout**. The **wall** command, when used by the super user, overrides the **mesg n** command. It is best to reserve this for those times when you (as the system administrator) need to ask users to get off the system. The **wall** command is described in Procedure 2.5, "Write to All Users."

mail and mailx

The UNIX System offers two electronic mail utilities through which users can communicate among themselves. If your system is connected to others by networking facilities, **mail**(1) and **mailx**(1) can be used to communicate with persons on other systems as well as users on your system.

mail is the basic utility for sending messages. **mailx** uses **mail** to send and receive messages but adds to it a multitude of extras that are useful for organizing messages into storage files, adding headers, and many other functions.

When **mailx** is used, a setup file is helpful. You can find a description of how to use a **.mailrc** setup file in the **mailx**(1) pages of the *User's and System Administrator's Reference Manual.*

User Requests

As the system administrator for your 3B2 computer you can expect users to look to you to help solve their problems. In addition to the system log described in Chapter 3, "Processor Operations," you will find it helpful to keep a users' trouble log. The problems that users run into fall into patterns. If you keep a record of how problems were resolved, you will not have to start from scratch when a problem recurs.

Trouble Reports

Another technique that is strongly recommended is an organized way for users to report problems. Figure 2-3 shows a sample Trouble Report that can be used to record and keep track of system problems.

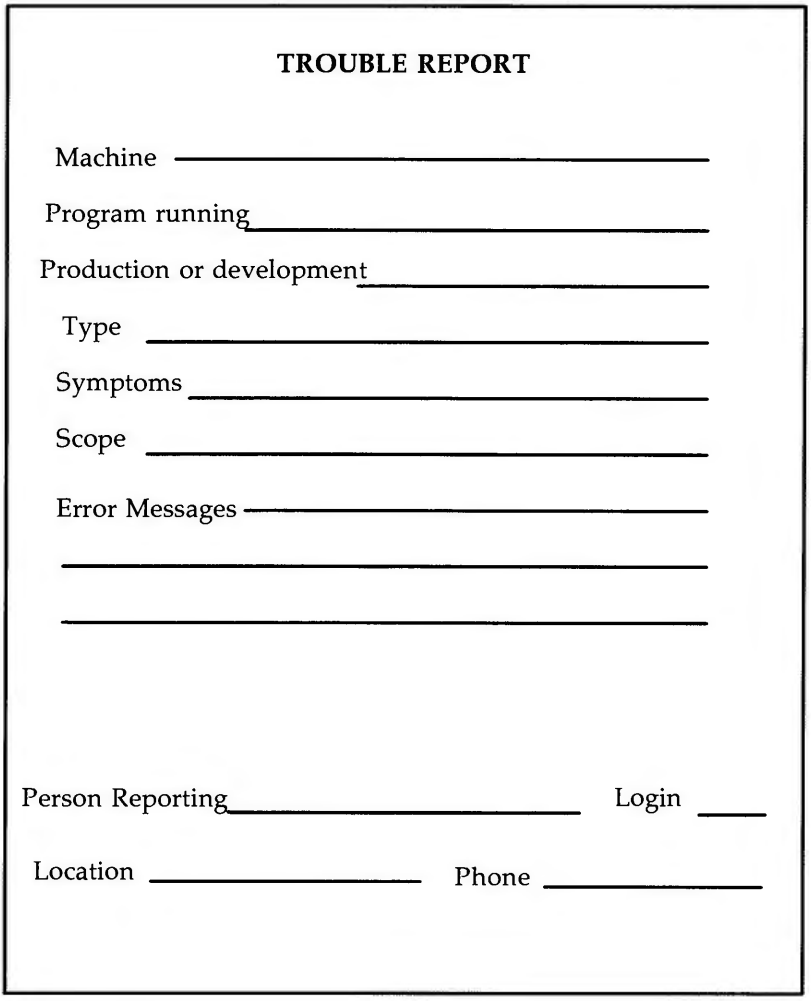


Figure 2-3: Sample Trouble Report

Chapter 3: Processor Operations

Introduction

This chapter describes the day-to-day operations of your 3B2 computer system.

- General policy

  Guidelines for balancing the needs of system maintenance and the interests of your user community; suggestions for record keeping; lists of important administrative directories and files

- Operating levels

  Definition of the operating levels of the system; how they are controlled

- Running firmware programs

  Running firmware-resident and bootable programs from the system firmware

- Hardware diagnostic information

  Using the hardware diagnostic monitor to locate system troubles.

General Policy

Many administrative tasks require the system to be shut down to a run level other than the multiuser state (see the discussion on "Operating Levels"). This means that conventional users cannot access the system. When the machine is taken out of the multiuser state, the users on the machine at the time are requested to log off. You should do these types of tasks when they will interfere least with the activities of the user community.

Sometimes problems arise that require the system to be taken down with little or no notice to the users. Try to give the user community as much notice as possible about events affecting the use of the machine. When the system must be taken out of service, tell the users when to expect the system to be available. Use the news (**/etc/news/***headline*) and the Message-of-the-Day (**/etc/motd**) to keep users informed about changes in hardware, software, policies, and procedures.

At your discretion, the following items should be done as prerequisites for any task that requires the system to leave the multiuser state.

1. When possible, schedule service-affecting tasks to be done during periods of low system use. For scheduled actions, use the Message-of-the-Day (**/etc/motd**) to inform users of future actions.
2. Check to see who is logged in before taking any actions that would affect a logged-in user. The **/etc/whodo** and **/bin/who** commands can be used to see who is on the system.
3. If the system is in use, give the users advanced warning about changes in system states or pending maintenance actions. For immediate actions, use the **/etc/wall** command to send a broadcast message announcing that the system will be taken down at a given time. Give the users a reasonable amount of time to end their activities and log off before taking the system down.

Maintain a System Log

In a multiuser environment, it is strongly recommended that a complete set of records be maintained. A system log book can be a valuable tool when troubleshooting transient problems or when trying to establish system operating characteristics over a period of time. Some of the things that you should consider entering into the log book follow:

- Maintenance records (dates and actions)
- Printouts of error messages and diagnostic phases
- Equipment and system configuration changes (dates and actions).

The format of the system log and the types of items noted in the log should follow a logical structure. Think of the log as a diary that you update on a periodic basis. How you use your system will dictate the form and importance of maintaining a system log.

Administrative Directories and Files

This section briefly describes the directories and files that are frequently used by a System Administrator. For more details about the purpose and contents of these directories and files, see Appendix B, "Directories and Files." For additional information on the formats of the system files, refer to the UNIX System V manual pages (Section 4) in the *User's and System Administrator's Reference Manual.*

Root Directories

The following directories are of the **root** file system (**/**):

bck	Directory used to mount a backup file system for restoring files.
bin	Directory that contains public commands.
boot	Directory that contains configurable object files created by the **/etc/mkboot**(1M) program.
dev	Directory containing special files that define all the devices on the system.
dgn	Directory that contains diagnostic programs.
edt	Directory that contains equipped device table data.
etc	Directory that contains administrative programs and tables.
install	Directory used by the System Administration Menu package to mount utilities packages for installation and removal (**/install** file system).
lib	Directory that contains public libraries.
lost+found	Directory used by **fsck**(1M) to save disconnected files.
mnt	Directory used to temporarily mount file systems.
save	Directory used by the System Administration Menu package for saving data on floppy disks.

shlib	Directory that contains shared libraries.
tmp	Directory used for temporary files.
usr	Directory used to mount the **/usr** file system. (See Chapter 5, "File System Administration," for a description of this file system.)

Important System Files

The following files and directories are important in the administration of the 3B2 computer.

/etc/checklist	File used to define a default list of file system devices to be checked by **/etc/fsck**.
/etc/fstab	File used to specify the file system(s) to be mounted by **/etc/mountall** and remote file systems to be mounted by **/etc/rmountall**.
/etc/gettydefs	File containing information used by **/etc/getty** to set the speed and terminal settings for a line.
/etc/group	File describing each group to the system.
/etc/init.d	Directory containing executable files used in upward and downward transitions to all system run levels. These files are linked to files beginning with **S** (start) or **K** (stop) in **/etc/rc*n*.d**, where *n* is replaced by the appropriate run level.
/etc/inittab	File containing the instructions to define the processes created or stopped by **/etc/init** for each initialization state.
/etc/master.d	Directory containing files that define the configuration of hardware devices, software drivers, system parameters, and aliases.
/etc/motd	File containing a brief Message of the Day, output by **/etc/profile**.

/etc/passwd	File identifying each user to the system.
/etc/profile	File containing the standard (default) environment for all users.
/etc/rc0	File executed by **/etc/shutdown** that executes shell scripts in **/etc/rc0.d** and **/etc/shutdown.d** directories for transitions to system run levels 0, 5, and 6.
/etc/rc0.d	Directory containing files executed by **/etc/rc0** for transitions to system run levels 0, 5, and 6. Files in this directory are linked from files in 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 run levels 0, 5, or 6.
/etc/rc2	File executed by **/etc/init** that executes shell scripts in **/etc/rc2.d** and **/etc/rc.d** on transitions to system run level 2.
/etc/rc2.d	Directory containing files executed by **/etc/rc2** for transitions to system run levels 2 and 3. Files in this directory are linked from files in 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 run levels 2 or 3.
/etc/rc.d	Directory containing executable files that do the various functions needed to initialize the system to run level 2; they are executed when **/etc/rc2** is run. (Files contained in this directory before UNIX System V Release 3.0 were moved to **/etc/rc2.d**. This directory is only maintained for compatibility.)
/etc/rc3	File executed by **/etc/init** that executes shell scripts in **/etc/rc3.d** on transitions to system run level 3 (Remote File Sharing state).

/etc/rc3.d	Directory containing files executed by **/etc/rc3** for transitions to system run level 3 (Remote File Sharing 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 run level 3.
/etc/save.d	Directory containing files that are used by the System Administration Menu commands associated with backing up data on media.
/etc/shadow	File containing encrypted password and password aging information for the corresponding logins in the **/etc/passwd** file when the shadow password feature is enabled.
/etc/shutdown	File containing a shell script that gracefully shuts down the system in preparation for system backup or for scheduled downtime.
/etc/shutdown.d	Directory containing executable files that do the various functions needed to transition the system to the single-user state (run levels 1, s, or S). (Files contained in this directory before UNIX System V Release 3.0 were moved to **/etc/rc0.d**. This directory is only maintained for compatibility.)
/etc/TIMEZONE	File used to set the time zone shell variable TZ.
/etc/utmp	File containing the information on the current run-state of the system.
/etc/wtmp	File containing a history of system logins. This file should be checked periodically for size.
/usr/adm/conlog	File containing a history of all the Input/Output (I/O) of the console terminal. This file should be checked periodically for size.

/usr/adm/errlog File containing a history of all the driver-produced error messages in the same format that was displayed to the console. The growth of this file is controlled by a cron task that copies the error log file into **/usr/adm/Oerrlog**.

/usr/adm/loginlog File used to record repetitive unsuccessful login attempts. If created to turn on the logging mechanism, the file should be checked periodically for size.

/usr/adm/sulog File containing a history of **su** command usage. This file should be checked periodically for size.

/usr/lib/cron/log File containing a history of all the actions taken by **/etc/cron**. This file should be checked periodically for size.

/usr/lib/help/HELPLOG
File containing a history of all the actions taken by the **/usr/bin/help** (if it is enabled on the system).

/usr/lib/spell/spellhist
File containing a history of all words that **spell** fails to match (if the Spell Utilities are installed on the system).

/usr/news Directory containing news files. This directory should be checked periodically, and old files should be discarded.

/usr/options Directory containing files that identify the utilities that are installed on the system.

/usr/spool/cron/crontabs
Directory containing crontab files for the **adm**, **root**, and **sys** logins and ordinary users listed in **cron.allow**.

Each of these files is described in more detail in Appendix B, "Directories and Files."

Levels of Operation

General

After you have set up your 3B2 computer for the first time (plugging it in, hooking all the hardware together, installing the system software, booting it, running the setup programs, making the "floppy key") as documented in the *Owner/Operator Manual*, you and other users can use the system. As controlled by the contents of the **/etc/inittab** file, when you turn on the computer (including the first time), the system comes up in a multiuser environment in which the following actions are taken:

- The file systems are mounted.
- The **cron** daemon is started for scheduled tasks.
- The basic networking functions of **uucp** (if added to the system) are available for use.
- The spooling and scheduling functions of the Line Printer Spooling Utilities (if added to the system) are available for use.
- Users can log in. The **getty**s are spawned on all connected terminal lines listed in **/etc/inittab** to have **getty**s respawned. (**getty**s are not on when the system is installed. You have to turn them on yourself.)

This is defined as the multiuser state. It is also referred to as **init** state 2 because all the activities of initializing the system are under the control of the **init** process. The "2" refers to entries in the special table **/etc/inittab** used by **init** to initialize the system to the multiuser state.

Not all activities, however, can be performed in the multiuser state. For example, if you were able to unmount a file system while users were accessing it, you would cause much data to be lost. Hence, for unmounting and other system administration tasks, another state (the single-user state) is needed.

The single-user state is an environment in which only the console has access to the system and the root file system alone is mounted. You are free to do tasks that affect the file systems and the system configuration because you are the only one on the system.

There are other system states (see Figure 3-1), but first a note of clarification. In discussions of system states, many similar terms are used to identify the same thing: the particular operating level of the system.

Here is a list of frequently encountered synonyms:

- Run state
- Run level
- Run mode
- Init state
- System state.

Likewise, each system state may be referred to in many ways, for example:

- Single user
- Single-user mode
- Run level 1, and so on.

In any case, each state or run level clearly defines the operation of the computer. Figure 3-1 defines each of them as they pertain to the 3B2 computer.

Run Level	Description
0	Powerdown state.
1, s, or S	Single-user mode is used to install/remove software utilities, run file system backups/restores, and to check file systems. **s**, **S**, or **1** can be used to go to single-user state. Using **1** unmounts everything except root and kills all user processes, except those that relate to the console. Using **s** and **S**, however, only kills processes spawned by **init** and does not unmount file systems.
2	Multiuser mode is the normal operating mode for the system. The default is that the **root** (**/**) and user (**/usr**) file systems are mounted in this mode. When the system is powered up, it is put in multiuser mode.
3	Multiuser/Remote File Sharing (RFS) (optional package) mode is used to start Remote File Sharing, connect your computer to an RFS network, mount remote resources, and offer your resources automatically.
4	User defined run state.
5	Firmware mode is used both to access programs that reside in ROM and to run programs in the **root** file system under the control of ROM. An example of the former is making a floppy key. An example of the latter is executing **/unix** to reboot the system.
6	Halt and reboot the operating system. By default, the system comes up in multiuser mode (run level 2).

Figure 3-1: System States

How init Controls the System State

The actions that cause the various states to exist are under the control of the **init** process, which is the first general process created by the system at boot time. It reads the file **/etc/inittab**, which defines exactly the processes that exist for which run level.

In multiuser state (run level 2), **init** scans the file for entries that have a tag for the run level (the tag is a 2) and executes everything after the last colon (:) on the line containing the tag. These tags represent the run levels in the table in Figure 3-2.

If you look at your **/etc/inittab**, you will see something that looks like the following display. (It is most unlikely that yours will look exactly like this one; **/etc/inittab** changes from one configuration to another.)

Note: If **/etc/inittab** was removed by mistake and is missing during shutdown, **init** will enter the single-user state (**init s**). While entering single-user state, **/usr** will remain mounted and processes not spawned by **init** will continue to run. You should replace **/etc/inittab** before changing states again.

```
zu::sysinit:/etc/bzapunix </dev/console >/dev/console 2>&1
fs::sysinit:/etc/bcheckrc </dev/console >/dev/console 2>&1
sd::sysinit:sh -c /etc/rc2.d/S00scsi </dev/console >/dev/console 2>&1
xdc::sysinit:sh -c 'if [ -x /etc/rc.d/0xdc ] ; then /etc/rc.d/0xdc ; fi' >/dev/console 2>&1
mt:23:bootwait:/etc/brc </dev/console >/dev/console 2>&1
pt:23:bootwait:/etc/ports </dev/console >/dev/console 2>&1
is:2:initdefault:
pl:s1234:powerfail:/etc/led -f                  # start green LED flashing
p3:s1234:powerfail:uadmin 2 0
fl:056:wait:/etc/led -f                         # start green LED flashing
s0:056:wait:/etc/rc0 >/dev/console 2>&1 </dev/console
s1:1:wait:/etc/shutdown -y -iS -g0 >/dev/console 2>&1 </dev/console
s2:23:wait:/etc/rc2 >/dev/console 2>&1 </dev/console
s3:3:wait:/etc/rc3 >/dev/console 2>&1 </dev/console
OF:0:wait:echo "\nPlease flip the power switch to the STANDBY position." >/dev/console 2>&1
of:0:wait:/etc/uadmin 2 0 >/dev/console 2>&1 </dev/console
un:56:wait:/etc/init.d/unlock > /dev/console 2>&1 < /dev/console
fw:5:wait:/etc/uadmin 2 2 >/dev/console 2>&1 </dev/console
RB:6:wait:echo "\nThe system is being restarted." >/dev/console 2>&1
rb:6:wait:/etc/uadmin 2 1 >/dev/console 2>&1 </dev/console
co:234:respawn:/etc/getty console console
ct:234:respawn:/etc/getty contty contty
he:234:respawn:sh -c 'sleep 20 ; exec /etc/hdelogger >/dev/console 2>&1'
mi::sysinit:/etc/init.d/mirdisk </dev/console >/dev/console 2>&1
lo::sysinit:/etc/init.d/lock < /dev/console > /dev/console 2>&1
21:234:respawn:/etc/getty tty21 9600 #rar
22:234:off:/etc/getty tty22 9600 #38400 baud rate is available
23:234:off:/etc/getty tty23 9600 #38400 baud rate is available
24:234:off:/etc/getty tty24 9600 #38400 baud rate is available
25:234:off:/etc/getty tty25 9600 #38400 baud rate is available
26:234:off:/etc/getty tty26 9600 #38400 baud rate is available
27:234:off:/etc/getty tty27 9600 #38400 baud rate is available
28:234:off:/etc/getty tty28 9600 #38400 baud rate is available
31:234:respawn:/usr/lib/uucp/uugetty -r -t 60 tty31 9600 #wr3b2a Direct Link
32:234:off:/etc/getty tty32 9600 #38400 baud rate is available
33:234:off:/etc/getty tty33 9600 #38400 baud rate is available
34:234:off:/etc/getty tty34 9600 #38400 baud rate is available
35:234:off:/etc/getty tty35 9600 #38400 baud rate is available
36:234:off:/etc/getty tty36 9600 #38400 baud rate is available
37:234:off:/etc/getty tty37 9600 #38400 baud rate is available
38:234:off:/etc/getty tty38 9600 #38400 baud rate is available
```

Continued

Continued from previous screen display

```
41:234:respawn:/etc/getty tty41 9600 #cms
42:234:off:/etc/getty tty42 9600 #38400 baud rate is available
43:234:off:/etc/getty tty43 9600 #38400 baud rate is available
44:234:off:/etc/getty tty44 9600 #38400 baud rate is available
45:234:off:/etc/getty tty45 9600 #38400 baud rate is available
46:234:off:/etc/getty tty46 9600 #38400 baud rate is available
47:234:off:/etc/getty tty47 9600 #38400 baud rate is available
48:234:off:/etc/getty tty48 9600 #38400 baud rate is available
```

The format of each line follows:

id:level:action:process

- The *id* is one or two characters that singularly identify an entry.
- The *level* is zero or more numbers and letters (**0** through **6**, **s**, **S**, **a**, **b**, and **c**) that determine what *level*(s) in which The *action* is to take place. If *level* is null, the *action* is valid in all levels.
- The *action* can be one of the following:

sysinit	Run *process* before **init** sends anything to the system console (`Console Login:`).
bootwait	Start *process* the first time **init** goes from single user to multiuser state after the system is booted. (If **initdefault** is set to **2**, the process will run right after the boot.) *init* starts the process, waits for its termination and, when it dies, does not restart the process.
wait	When going to *level*, start *process* and wait until it is finished.
initdefault	When **init** starts, it will enter *level*; the *process* field for this *action* has no meaning.
once	Run *process* once and do not start it again if it finishes.

powerfail	Tells **init** to run *process* whenever a direct powerdown of the computer is requested.
respawn	If process does not exist, start it; wait for it to finish, and then start another.
ondemand	Synonymous with **respawn**, but used only with *level* **a**, **b**, or **c**.
off	When in *level*, kill process or ignore it.

- The *process* is any executable program, including shell procedures.
- The # can be used to add a comment to the end of a line. Everything after a # on a line will be ignored by init.

When changing levels, **init** kills all processes not specified for that level. We will go through more manageable pieces of this table to get a clearer idea of how the system is controlled by **init**.

A Look at Entering the Multiuser State

Power Up

When you power up your system, it will enter multiuser state by default. (You can change the default by changing the **initdefault** line in your **inittab** file.) In effect, going to the multiuser state follows these broad lines (see Figure 3-2):

1. Turn on the computer.
2. The operating system is loaded and the early system initializations are started by **init**.
3. The run level change is prepared by the **/etc/rc2** procedure.
4. Finally the system is made public via the spawning of **getty**s along the terminal lines.

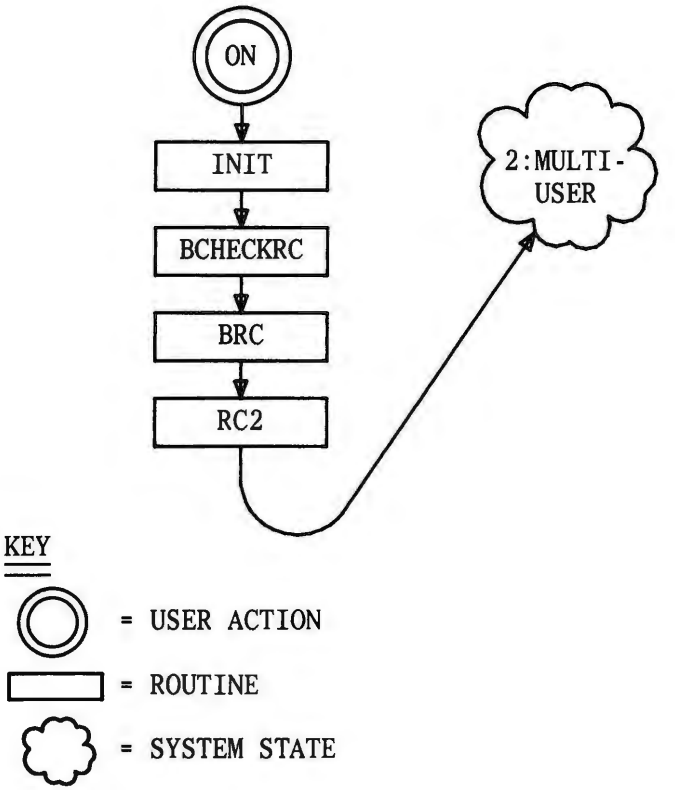


Figure 3-2: A Look at System Initialization

Early Initialization

Just after the operating system is first loaded into main memory via the specialized boot programs [see **newboot**(1M) and **3b2boot**(8) in the *User's and System Administrator's Reference Manual* for more detail], the **init** process is created. It immediately scans **/etc/inittab** for entries of the type **sysinit**.

```
zu::sysinit:/etc/bzapunix </dev/console >/dev/console 2>&1
fs::sysinit:/etc/bcheckrc </dev/console >/dev/console 2>&1
sd::sysinit:sh -c /etc/rc2.d/S00scsi </dev/console >/dev/console 2>&1
xdc::sysinit:sh -c 'if [ -x /etc/rc.d/0xdc ] ;
                then /etc/rc.d/0xdc ; fi' >/dev/console 2>&1
```

The **sysinit** entries are executed in sequence and do the necessary early initializations of the system. Note that each entry shows a standard input/output relationship with **/dev/console**. This is the way communication is established with the system console before the system has been brought to the multiuser state.

Prepare the Run Level Change

Now the system must be placed in a particular run level. First, **init** scans the table to find an entry that specifies an *action* of the type **initdefault**. If it finds one, it uses the run level of that entry as the tag it will use to select the next entries to be executed. In our sample **/etc/inittab**, the **initdefault** entry specifies run level 2 (the multiuser state) as the level to select and execute other entries.

```
is:2:initdefault:
s2:23:wait:/etc/rc2 >/dev/console 2>&1 </dev/console
co:234:respawn:/etc/getty console console
ct::respawn:/usr/lib/uucp/uugetty -r -t 60 contty 1200H
he:234:respawn:sh -c 'sleep 20 ; exec /etc/hdelogger >/dev/console 2>&1'
21:234:respawn:/etc/getty -t 60 tty21 9600
24:234:respawn:/etc/getty -t 60 tty24 9600
25:234:respawn:/etc/getty -t 60 tty25 9600
28:234:respawn:/etc/getty -t 60 tty28 9600
31:234:respawn:/etc/getty -t 60 tty31 1200
32:234:respawn:/etc/getty -t 60 tty32 1200
35:234:respawn:/etc/getty -t 60 tty35 9600
38:234:respawn:/etc/getty -t 60 tty38 9600
41:234:respawn:/usr/lib/uucp/uugetty -r -t 60 tty41 1200
42:234:respawn:/usr/lib/uucp/uugetty -r -t 60 tty42 1200
43:234:respawn:/usr/lib/uucp/uugetty -r -t 60 tty43 1200
46:234:respawn:/etc/getty -t 60 tty46 1200
47:234:respawn:/etc/getty -t 60 tty47 9600
48:234:respawn:/etc/getty -t 60 tty48 9600
```

The other entries shown above specify the actions necessary to prepare the system to change to the multiuser run level. First, **/etc/rc2** is executed. It executes all files in **/etc/rc2.d** that begin with the letter **S**. It then executes all the files in the **/etc/rc.d** directory, accomplishing (among other things) the following:

- Sets up and mounts the file systems
- Starts the **cron** daemon
- Displays the current system hardware configuration
- Makes **uucp** available for use, if installed
- Makes line printer (lp) system available for use, if installed
- Starts a **getty** for the console
- Starts the hard disk error logging daemon
- Starts **getty** on the lines connected to the ports that are active.

At this moment, the full multiuser environment is established, and your system is available for users to log in (see Procedure 3.1, "Powerup," and Procedure 3.4, "Return to Multiuser").

A Look at the System Life Cycle

Change Run Levels

In effect, changing run levels follows these broad lines (see Figure 3-3):

1. The system administrator enters a command that directs **init** to execute entries in **/etc/inittab** for a new run level.
2. Key procedures, such as **/etc/shutdown**, **/etc/rc0**, **/etc/rc2**, and **/etc/rc3** (whichever are applicable), are run to initialize the new state.
3. The new state is reached.

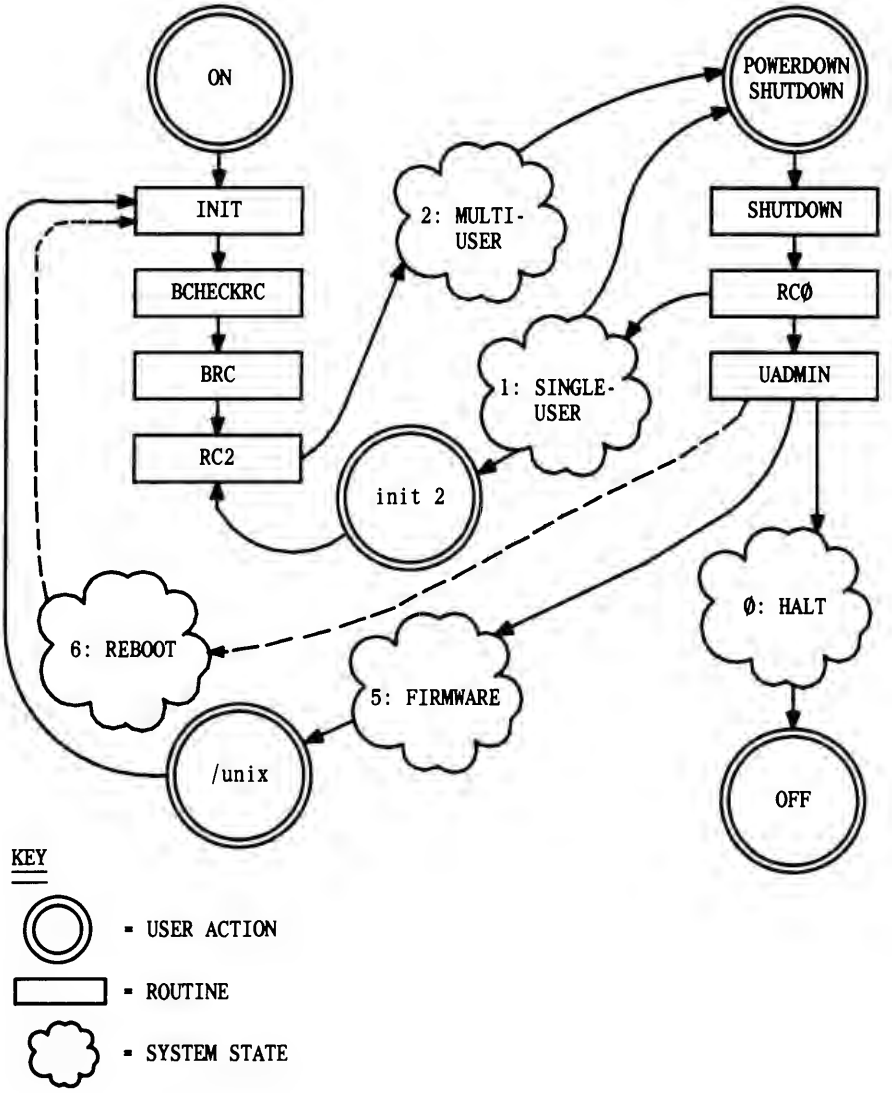


Figure 3-3: A Look at the System Life Cycle

Run Level Directories

Run levels 0, 2, and 3 have a directory of files that are executed in transitions to and from that level. These directories are **rc0.d**, **rc2.d**, and **rc3.d**, respectively. All files in these directories are linked to files in **/etc/init.d**. The run level file names look like this:

S*00name*

or

K*00name*

The file names can be split into three parts:

S or **K**	The first letter defines whether the process should be started (**S**) or stopped (**K**) on entering the new run level.
00	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, etc.) or stopped (K00, K01, K02, etc.).
name	The rest of the file name is the **/etc/init.d** file name to which this file is linked.

For example, the **init.d** file **cron** is linked to the **rc2.d** file, **s75cron** file, and **rc0.d** file **K70cron**. When you enter **init 2**, this file is executed with the **start** option: **sh S75cron start**. When you enter **init 0**, this file is executed with the **stop** option: **sh K70cron stop**. This particular shell script will execute **/usr/bin/cron** when run with the **start** option and will **kill** the **cron** process when run with the **stop** option.

Because these files are shell scripts, you can read them to see what they do. You can change the files, though it is preferable to add your own since the delivered scripts may change in future releases. To create your own scripts, you should follow these rules:

- Place the file in **/etc/init.d**.
- Link the file to files in appropriate run state directories using the naming convention described above.
- Have the file accept the **start** and/or **stop** options.

Go to Single User Mode

At times in a given work week, you will need to do some administrative functions in the single-user mode (see Procedure 3.3, "Shutdown to Single-User"). The normal way to go to single-user mode is through the **/etc/shutdown** command. This procedure executes all the files in the **/etc/rc0.d** directory by calling the **/etc/rc0** procedure, accomplishing, among other things, the following:

- Closing all open files and stopping all user processes
- Stopping all daemons and services
- Writing all system buffers out to the disk
- Unmounting all file systems except root.

The entries for single-user processing in the sample **/etc/inittab** follow:

```
s1:1:wait:/etc/shutdown -y -iS -g0 >/dev/console 2>&1 </dev/console
p1:s1234:powerfail:/etc/led -f   # start green LED flashing
p3:s1234:powerfail:uadmin 2 0
```

There are two major ways to start the shutdown processing:

1. Enter the **shutdown -i1** command (recommended).
2. Enter the **init 1** command, which forces the **init** process to scan the table. The first entry it finds is the **s1** entry, and it starts the shutdown processing; however, this does not advise the users that the system is going down.

Now the system is in the single-user environment, and you can do the appropriate administrative tasks.

Run Level 3 (Optional Remote File Sharing Utilities)

The **init 3** is used to enter the Remote File Sharing state (run level 3). This procedure executes **/etc/rc2** and **/etc/rc3** to run processes in all directories associated with those two states. On top of the multiuser state (state 2) processes, the processes in **/etc/rc3.d** will:

- Start Remote File Sharing and connect you to the 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 **/etc/inittab** from **2** to **3**, so you automatically come up in Remote File Sharing mode when you reboot. More information is available in the documentation delivered with the Remote File Sharing Utilities.

Run Levels 5 and 6

To go to the firmware mode or to reboot the system because of reconfiguration, use the **shutdown** command with the **-i5** or the **-i6** option. Note that the pertinent entries in **/etc/inittab** are similar, and sometimes the same.

```
fl:056:wait:/etc/led -f                          # start green LED flashing
s0:056:wait:/etc/rc0 >/dev/console 2>&1 </dev/console
fw:5:wait:/etc/uadmin 2 2 >/dev/console 2>&1 </dev/console
RB:6:wait:echo "\nThe system is being restarted." >/dev/console 2>&1
rb:6:wait:/etc/uadmin 2 1 >/dev/console 2>&1 </dev/console
```

For both system states, the cabinet light is caused to flash and the **/etc/rc0** procedure is called. Then, the **/etc/uadmin** procedure with different arguments for each state is called. This procedure invokes the **uadmin**(2) system call, which directly prepares either system state.

While the computer stays as long as you want it to in state 5, it stays only briefly in state 6 to restart the entire system initialization. Both states are similar, however, since they exist when the computer is under firmware control.

Turn the System Off

The final step in the life cycle of the system is turning it off. There are 3 ways to turn off the 3B2 computer if it is in the multiuser state:

1. Using the power switch
2. Executing the **powerdown** command
3. Executing the **shutdown** command.

The following entries in **/etc/inittab** apply to powering down the system:

```
p1:s1234:powerfail:/etc/led -f                    # start green LED flashing
p3:s1234:powerfail:uadmin 2 0
f1:056:wait:/etc/led -f                           # start green LED flashing
s0:056:wait:/etc/rc0 >/dev/console 2>&1 </dev/console
OF:0:wait:echo "\nPlease flip the power switch to the STANDBY position." >/dev/console 2>&1
of:0:wait:/etc/uadmin 2 0 >/dev/console 2>&1 </dev/console
```

For more information on each of these methods of turning the system off, see the *Owner/Operator Manual* that came with your computer.

Caution: **NEVER PULL THE POWER PLUG UNTIL THE 3B2 COMPUTER HAS COMPLETED THE POWERDOWN PROCEDURE. If you pull the power plug or externally remove power before the computer has completed this step, your files may be damaged or destroyed.**

Error Logger

To assist in finding problems with your system, there is an error logger that logs all the driver error messages. This includes the messages that are reported to *hdelogger*. These are the same error messages that get displayed on the console terminal. This logger is a complete history of all the error messages that have occurred on the system.

The messages are placed in a circular buffer in memory. This data is extracted from memory and placed into a file by a user level process. The data in the circular buffer could be lost if the buffer writes begin to wrap around before the user level process has a chance to read the buffer.

The file that the data is written to is **/usr/adm/errlog**. There is a cron task that controls the growth of the error log file. This cron task copies the error log file into **/usr/adm/Oerrlog** once a week. If you find the **errlog** file growing too large in the 1-week interval, you may need to change the cron task.

A time stamp is placed in the buffer with each message. With this you can determine when each error occurred. The time stamp is in the form *time*: where *time* is the number of seconds since January 1, 1970.

To aid you in the observation of error data, there are two commands: **errcrash**(8) and **errint**(8). The **errcrash** command allows error data to be extracted from the memory buffer after a system dump [**sysdump**(8)] is made. If any information is found that is not in the log file, it is appended to the end of the file. The **errint** command allows errors that have occurred in a certain interval to be copied from the log file. These commands will help in diagnosing system trouble.

Run Firmware Programs

Firmware procedures include the use of the firmware-resident programs and the use of the bootable programs that are part of the Essential Utilities. The following are firmware-level programs:

baud	Change the firmware baud rate
edt	Display the equipped device table data
errorinfo	Display the expanded firmware error message information
express	Set diagnostic mode during reboot
newkey	Make a new floppy key
passwd	Change the firmware password
sysdump	Dump the system image (RAM) to dump device
version	Display firmware version information.

The bootable programs are **dgmon**, **filledt**, **/etc/system**, and **/unix**. All eight firmware programs are accessible in response to the firmware prompt

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

When the prompt

```
Enter path name:
```

is displayed, only bootable programs on the defined boot device are accessible. (See Procedure 3.5, "Run the Firmware Programs.")

Note: In the firmware mode, the erase character is the backspace key, and the kill character is the "at sign," **@**.

Display Firmware Program Menu

To display the firmware program menu, enter a question mark (?) in response to the message

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

This message is displayed when the firmware mode is initiated. The following command line entries and system responses show how to display the firmware program menu. At the start of the example, the system has been taken to the firmware mode (run level 5).

Note: As a security measure, echo is turned off by programs requesting a password entry. Therefore, the characters entered for the firmware password are not displayed. Echo is enabled after the password is entered. The default firmware password is **mcp**.

```
FIRMWARE MODE

password

Enter name of program to execute [    ]: ?

Enter an executable or system file, a directory name,
or one of the possible firmware program names:

baud
edt
errorinfo
express
newkey
passwd
sysdump
version
(q to quit)

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

Change the Firmware Baud Rate

The **baud**(8) command is used to display/change the console baud rate. The system will change the console baud rate to the requested value and save the value in NVRAM. The valid baud rates follow:

1200 4800 9600

Caution: If a baud rate is selected that is not compatible with the console terminal, the computer will not communicate with the console.

If you cannot establish communication between the console and the system, set the console terminal baud rate to 9600 and boot the system from the floppy key.

The following command line entries and system responses show how to change the console baud rate.

```
FIRMWARE MODE

password

Enter name of program to execute [    ]: baud
Enter new rate [9600]: 4800
Change baud rate to 4800
```

Display Equipped Device Table

The current Equipped Device Table (EDT) is displayed using the firmware program, **edt**(8). The following command line entries and system responses show how to display the equipped device table. The system is in the firmware mode (run level 5) at the start of the example. A password is required to enter the firmware mode. The default firmware password is **mcp**.

```
FIRMWARE MODE

password

Enter name of program to execute [    ]: edt

Current System Configuration

System Board memory size: 8 megabyte(s)
#0 - 4 megabyte(s),  #1 - 4 megabyte(s),

00 - device name = SBD      , occurrence = 0, slot = 00, ID code = 0x01
     boot device = y, board width = double, word width = 2 byte(s),
     req Q size = 0x00, comp Q size = 0x00
     subdevice(s)
     #00 = FD5     , ID code = 0x01,

Press any key to continue <CR>

01 - device name = SCSI     , occurrence = 0, slot = 01, ID code = 0x100
     boot device = y, board width = single, word width = 2 byte(s),
     req Q size = 0x38, comp Q size = 0x38, indirect edt
     subdevice(s)
     #00 = disk     , ID code = 0x100,  #01 = tape    , ID code = 0x101

Press any key to continue <CR>
```

Continued

Continued from previous screen display

```
02 - device name = EPORTS    , occurrence = 0, slot = 02, ID code = 0x102
     type = integral i/o bus
     boot device = n, board width = single, word width = 2 byte(s),
     req Q size = 0x21, comp Q size = 0x46

Press any key to continue <CR>

03 - device name = EPORTS    , occurrence = 0, slot = 03, ID code = 0x102
     type = integral i/o bus
     boot device = n, board width = single, word width = 2 byte(s),
     req Q size = 0x21, comp Q size = 0x46

Press any key to continue <CR>

04 - device name = EPORTS    , occurrence = 0, slot = 04, ID code = 0x102
     type = integral i/o bus
     boot device = n, board width = single, word width = 2 byte(s),
     req Q size = 0x21, comp Q size = 0x46

Press any key to continue <CR>

05 - device name = MAU, occurrence = 0, slot = 00, ID code = 0xfd00
     type = co-processor

Press any key to continue <CR>



DONE

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

Display Expanded Firmware Error Message Information

The **errorinfo** command is a firmware-level command used to output more detailed, fault-specific information about firmware error messages. Executing the **errorinfo** command outputs the expanded error information and clears the data from NVRAM. Executing the command a second time or executing the command when no expanded error data is stored in NVRAM results in a NONE output message.

Expanded firmware error message information is available for the following:

- Interrupt Messages
- Exception Messages
- Abort Messages
- Thermal Shutdown.

Caution: Executing the firmware-level command `errorinfo` outputs and clears the expanded error information stored in NVRAM. Be sure to either copy the displayed output or have a printer enabled when the command is first executed.

Interrupt Messages

Interrupt message expansion provides values for the Program Counter (PC), Program Status Word (PSW), the Control Status Error Register (CSER), Fault Latches 1 and 2, and priority level (LVL) at the time the system was interrupted. The format of the message is as follows.

```
FIRMWARE MODE
password
Enter name of program to execute [   ]:errorinfo
INTERRUPT,    LVL=13
PC=0xnnnnnnnn
PSW=0xnnnnnnnn
CSER=0xnnnnnnnn
FL1=0xnnnnnnnn
FL2=0xnnnnnnnn
```

Exception Messages

Exception message expansion provides values for the Program Counter (PC), Program Status Word (PSW), the Control Status Error Register (CSER), and Fault Latches 1 and 2. The format of the message is as follows:

```
FIRMWARE MODE
password
Enter name of program to execute [   ]:errorinfo
EXCEPTION
PC=0xnnnnnnnn
PSW=0xnnnnnnnn
CSER=0xnnnnnnnn
FL1=0xnnnnnnnn
FL2=0xnnnnnnnn
```

Abort Messages

Abort message expansion provides values for the Program Counter (PC), Program Status Word (PSW), the Control Status Error Register (CSER), and Fault Latches 1 and 2. The format of the message is as follows:

```
ABORT
PC=0xnnnnnnnn
PSW=0xnnnnnnnn
CSER=0xnnnnnnnn
FL1=0xnnnnnnnn
FL2=0xnnnnnnnn
```

Thermal Shutdown

The thermal shutdown message does not provide for message expansion and is shown as follows:

```
FIRMWARE MODE

password

Enter name of program to execute [ ]:errorinfo

THERMAL SHUTDOWN
```

Execute Diagnostics During Reboot

The **express**(8) command will enable or disable the execution of diagnostics during a reboot sequence. When diagnostics are disabled, the boot executes more quickly. The normal automatic boot sequence checks NVRAM to determine whether to execute diagnostics. The value stored in NVRAM is changed by the **express** command. The default is for the diagnostics to execute.

The **express** command will display the current value (enabled or disabled) and ask if you want to change (toggle) to the other value. The default answer is no with **<CR>**. The following command line entries and system responses show how to toggle the execution of diagnostics during a reboot.

Caution: You should run diagnostics. By disabling diagnostics you run the risk of corrupting file systems along with other dangers.

```
FIRMWARE MODE

password

Enter name of program to execute [   ]:express

Automatic diagnostics are enabled
        toggle? (n) y
```

Make a Floppy Key

A floppy key is a floppy disk used to reset certain system parameters stored in NVRAM. These parameters include the system name, node name, speed of the console terminal, and the firmware password. A floppy key is used to reset these parameters to known values. For example, the floppy key is used to recover from a forgotten firmware password. The **newkey**(8) program is used to make a floppy key. The following command line entries and system responses show how to make a floppy key. The default firmware password is **mcp**.

```
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 [    ]:
```

See Procedure 1.7, "Forgotten Firmware Password Recovery," for a description of how to use the floppy key.

Change the Firmware Password

The **passwd**(8) program is used to change the firmware password. You must know the existing firmware password to change the password. The following command line entries and system responses show how to change the firmware password. As a security measure, echo is turned off by programs requesting a password entry; therefore, the characters for the firmware password are not displayed. Echo is enabled after the password is entered. You must enter the new password twice for confirmation.

```
FIRMWARE MODE

password

Enter name of program to execute [   ]: passwd

enter old password: old password
enter new password: new password
confirmation: new password


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

Dump System Image

Following a crash (SYSTEM FAILURE), the system will automatically dump the system image to the boot device crash partition (if the system can write to this partition and AUTODUMP=1) and automatically reboot the UNIX operating system (if AUTOBOOT=1). If the crash partition is unaccessible (automatic dump failed), you may dump to floppy disks.

The **sysdump** firmware program is used to write the system image (contents of the RAM) to the boot device crash partition (/dev/rdsk/c1t1d0s3 or device the specified by DUMPDEV) or to floppy disks. If DUMPDEV is defined in **/etc/system**, then **sysdump** writes the contents of RAM to that device. If the AUTODUMP parameter is set (value=1) the system will automatically dump the mainstore (RAM) to the crash partition in response to a SYSTEM FAILURE (crash). The operating system is delivered with the

AUTODUMP and AUTOBOOT parameters set (enabled). If the AUTODUMP parameter is not set (value=0) the **sysdump** command must be used to dump the mainstore.

Therefore, as the system is delivered, the contents of the mainstore is automatically written to partition **/dev/rdsk/c1t1d0s3** following a system crash. This default dump device can be redefined by adding a DUMPDEV entry to the **/etc/system** file and rebooting the system. The Small Computer Systems Interface (SCSI) Cartridge Tape is defined as the dump device by the entry **DUMPDEV:/dev/rmt/c1t2d0s0**. (Note that dumping the system image to the SCSI Cartridge Tape requires about 45 seconds per megabyte.)

If a system dump is to be executed following a crash, execution of the **sysdump** program must be the first action taken. Rebooting the system or executing another program will overwrite the system image, making further crash analysis a futile effort. For an example of how to do a system dump, see "Perform a System Dump" section of Procedure 3.7, "Recovery From System Trouble."

Display Firmware Version

The **version**(8) program is used to display the firmware version. The following command line entries and system responses show how to display the firmware version.

```
FIRMWARE MODE

password

Enter name of program to execute [    ]: version

Created: 1/31/89
Issue: 0b0b0b0b
Release: 3.2.2
Load: 3FG9
Serial Number: 0bffffff


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

Fill Equipped Device Table (Boot filledt)

The bootable **filledt**(8) program is executed to build the equipped device table from data in the **/dgn/edt_data**, and the **/edt/SCSI/edt_data** files. The following command line entries and system responses show how to manually build the equipped device table. Refer to Appendix C, "Error Messages," for descriptions of the **filledt** error messages.

```
FIRMWARE MODE

password

Enter name of program to execute [    ]: filledt
            Possible load devices are:

Option Number    Slot     Name
-----------------------------------
      0           0      FD5
      1           0      SCSI

Enter Load Device Option Number [1 (SCSI)]: <CR>

Possible subdevices are:
Option Number    Subdevice   Name
---------------------------------------
      0             0        disk
      1             1        tape

Enter Subdevice Option Number [0 (disk)]: <CR>

BEGIN FILLING EDT

EDT SUCCESSFULLY COMPLETED

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

Boot the Operating System

The UNIX operating system can be booted in either of two ways: **/unix** or **/etc/system**. Booting **/unix** loads and runs the **/unix** file on the boot device. Booting **/etc/system** causes a new **/unix** to be generated. The new system configuration and load map are displayed when **/etc/system** is booted.

Diagnostic Information

Diagnostics are intended to be used as tools for locating hardware problems in the 3B2 computer. By running the diagnostic phases, you should be able to isolate the source of the problem to a specific area of hardware or possibly to a specific card. This will help your AT&T Service Representative or authorized dealer determine the problem and what needs to be done to solve it.

If you have a system failure and you are not sure you can run the diagnostic phases properly, contact your AT&T Service Representative or authorized dealer for help. Procedure 3.8, "Use the Diagnostic Monitor," gives you help in getting started.

After powering up the system, the system administrator should periodically run all diagnostics. The diagnostic types are described below. Since only the NORMAL diagnostics are run during a powerup sequence, diagnostics should be periodically run to execute the DEMAND and INTERACTIVE phases. In certain applications, the system may be powered down only for maintenance. In this application, it is important that a schedule be established to run demand and interactive diagnostics periodically.

Types of Diagnostics

There are three types of diagnostic phases: normal, demand, and interactive. They are defined as follows:

NORMAL Normal diagnostics are automatically run each time the system is powered up. The normal diagnostics are run manually via the diagnostic monitor (**dgmon**).

DEMAND Demand diagnostics are diagnostics that run only on a manual request basis via the diagnostic monitor. These diagnostic phases DO NOT run automatically as part of a powerup sequence.

INTERACTIVE Interactive diagnostics are diagnostics that are manually run via the diagnostic monitor, and they require operator intervention. The operator intervention usually consists of inserting a floppy disk or SCSI Cartridge Tape into the floppy disk drive or the SCSI Tape Drive and/or entering data via the keyboard.

Diagnostic Monitor (dgmon)

Diagnostics are run from the firmware mode via the diagnostic monitor program (**dgmon**). To get to the firmware mode from the multiuser mode, you must be logged in at the console as **root**. The steps necessary to run diagnostics are as follows:

Step 1: At the console terminal, take the system to the firmware mode (run level 5).

shutdown -y -i5

If you are the only one logged in, you can use an express shutdown (**-g0**) where a grace period of zero seconds is used.

Step 2: Enter the firmware password (**mcp** is the default). What you type is not echoed to the terminal. The system will respond with the following message:

```
Enter program to execute [ /unix ]:
```

Step 3: Execute the **dgmon** program from the SCSI hard disk (load device option **1** SCSI, subdevice option **0** disk).

Step 4: Run diagnostics as required. Refer to the "Examples of Diagnostic Commands" discussion for command format.

Step 5: When you have finished running diagnostics, type **quit**, then boot the UNIX operating system (**/unix**) from the SCSI hard disk (load device option **1** SCSI, subdevice option **0** disk).

dgmon Commands

When the **dgmon** program is executed, it expects you to enter commands. There are six commands: **h** or **?**, **l**, **q**, **s**, **errorinfo** and **dgn**.

h **dgmon** provides a help menu, which describes the other commands. The **h** command is used to display the help menu. To see the help menu, enter: **h** <**CR**>.

q The **q** command is used to exit from the **dgmon** program. To do this, enter: **q** <**CR**>.

s The devices that can be checked by running the diagnostic phases are listed in the Equipped Device Table (EDT). The **s** command is used to show the EDT. To see the list of devices, enter: **s** <**CR**>.

l The diagnostic phases for a given device are listed in the Diagnostic Phase Tables. The **l** command is used to display these tables. To display the diagnostic phases for a specific device, enter: **l** *device_type* <**CR**>.

errorinfo

The **errorinfo** command toggles the error flag. This enables the augmented diagnostic interrupt and exception messages to be printed at the console. To do this enter **errorinfo**<**CR**>

```
ERROR FLAG IS OFF
TOGGLE [n] ? y
```

dgn The **dgn** command is used to run the diagnostic phases. The optional arguments of the **dgn** command are as follows:

device_type Represents the option of running diagnostic phases on a certain type of device. For example, the command **dgn sbd** runs all the NORMAL diagnostic phases on the system board.

device_type # Represents the option of running diagnostic phases on a certain device. For example, the command **dgn eports 0** runs all the NORMAL diagnostic phases on the first EPORTS card.

Whereas, the command **dgn eports** would run all the NORMAL diagnostic phases on all EPORTS cards.

rep=*?* Represents the number of times you want the phases to run. Valid numbers are between 1 and 65536.

ph=*?*[**-***?*] Represents the option to run a specific phase or string of phases. When running specific phases, be sure you know the phase you want or you could cause some problems. INTERACTIVE phases are run if they are included in a string of phases. When possible, run INTERACTIVE phases separately.

ucl Represents the option to run the phases in the unconditional mode. In this mode, testing continues when a phase fails. The results of each phase are displayed as it is completed. This mode cannot be used with the **soak** option.

soak Represents the option of running the phases continuously and storing all 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 NORMAL and DEMAND phases in sequence within the requested range of phases. Stop **soak** by entering a character at the console or using the **rep** option. The **soak** option cannot be used with the INTERACTIVE phases.

print = yes or no Allows you to display or suppress the individual phase outputs regardless of the displayed operating mode.

Examples of dgn Commands

The following are some examples of valid **dgn** commands using the various options. They should all be followed by a carriage return.

dgn	Runs all NORMAL phases once on the devices in the Equipped Device Table. The results of each phase are displayed as it completes. If any of the phases fail, testing stops, and a failure message is displayed.
dgn sbd 0	Runs all NORMAL phases once on the system board. The results of each phase are displayed as it completes. If any of the phases fail, testing stops and a failure message is displayed.
dgn scsi ph=17	Runs phase 17 (SCSI Reset Test) on all the SCSI devices. It shows the configuration and results when testing completes.
dgn scsi ph=24	Runs phase 24 (SCSI Configuration Status) on all the SCSI devices. It shows the configuration and results when testing completes.
dgn eports	Runs all NORMAL phases once on all the EPORTS cards. If any of the phases fail, testing stops and a failure message is displayed.
dgn eports 1	Runs all NORMAL phases once on EPORTS card 1. The results of each phase are displayed as it completes. If any of the phases fail, testing stops and a failure message is displayed.
dgn eports 2 ucl	Runs all NORMAL phases once on EPORTS card 2. The results of each phase are displayed as it completes. Testing continues if a phase fails.
dgn sbd ph=3	Runs phase 3 (CPU #4 Normal DGN) once on the system board. The results of the phase are displayed as it completes.
dgn sbd ph=1-4	Runs phases 1 (CPU #2 Normal DGN), 2 (CPU #3 Normal DGN), 3 (CPU #4 Normal DGN), and 4 (Memory Management #1 Normal DGN) once on the system board or until a failure occurs. The results of each phase are displayed as it completes.

dgn sbd rep=3 ph=3

Runs phase 3 (CPU #4 Normal DGN) three times. Testing stops if any part of the phase fails, and a failure message is displayed.

dgn ucl

Runs all NORMAL phases once on every device in the Equipped Device Table. Results of each phase are displayed as it runs. Testing continues if a phase fails.

dgn ucl rep=3

Runs all NORMAL phases three times. Testing continues if a phase fails. The results of each phase are displayed as it runs.

dgn soak

Runs all NORMAL and DEMAND phases on the system board and all cards until a character is entered on the console terminal. Testing stops when a character is entered, and the results are displayed.

dgn ports 1 soak

Runs all NORMAL and DEMAND phases on PORTS card 1 until a character is entered on the console. Testing stops when a character is entered, and the results are displayed.

dgn sbd soak ph=1-3

Runs phases 1 (CPU #2 Demand DGN), 2 (CPU #3 Demand DGN), and 3 (CPU #4 Demand DGN) until a character is entered on the console terminal. Testing results are displayed when testing stops.

dgn sbd soak rep=10 ph=11

Runs phase 11 (Control Status Register Normal DGN) 10 times and then displays a summary of the results. Testing continues when a phase fails.

dgn soak rep=25

Runs all NORMAL and DEMAND phases on the system board and all cards 25 times. The results are displayed when testing is completed. Testing continues when a phase fails.

dgn print

Prints individual or phase outputs regardless of displayed operating mode.

Note: When specific phases are requested, the device(s) to be tested must be named.

Suggested Sequence for Running Phases

While in the Diagnostic Monitor, you can run diagnostic phases. Decide what phases you want to run before starting. The following sequence is a guideline to follow when running diagnostic phases. Random running of phases is not suggested.

Note: Be sure to record the results of all phases so your AT&T Service Representative or authorized dealer will have an idea of the trouble before making the service call.

1. Use the **s** command to identify devices in the Equipped Device Table.
2. Use the **dgn** command to locate the device that is causing the trouble.
3. If one device returns a `DIAGNOSTICS FAILED` message, run all the NORMAL phases on that device.
4. If any of the NORMAL phases failed, you may want to repeat those phases with the **soak** or **ucl** option and a specific number of repetitions. Note that the phase number for the individual diagnostic phases is found by using the **l** command.
5. If you are testing because of a system failure or intermittent trouble, but all the NORMAL phases passed, the next logical step is to run some of the DEMAND phases.
6. If any of the DEMAND phases fail, you may want to run those phases again with the **soak** or **ucl** option and a specific number of repetitions.
7. After running any phases that failed with the **soak** or **ucl** option, omit those phases and continue running the other phases.
8. When a phase fails, the phases that follow may not be executed. If any of the devices in the Equipped Device Table were not tested because of a premature test termination, those devices should be tested by using the specific phase number.

9. After running all the NORMAL and all the DEMAND phases, you may want to run the INTERACTIVE phases.
10. Once you have completed running diagnostic phases, check to make sure you have a complete record of the results. If the AT&T Service Representative or authorized dealer cannot understand your results, the diagnostics must be run again.

Sample Diagnostic Execution

The following examples are provided to show you what to expect when running diagnostics. The responses will vary according to the command you execute.

NORMAL Diagnostic Phase

Phase 1 (CPU #2) is a NORMAL type phase run on the system board; it takes about 1 second to execute. The following command line entry and system responses show the successful execution of phase 1 on the system board.

```
DGMON > dgn sbd ph=1

    <<<  DIAGNOSTIC MODE  >>>

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

     SBD 0 (IN I/O BUS SLOT 0) DIAGNOSTICS PASSED

DGMON >
```

DEMAND Diagnostic Phase

Phase 5, Extended Dynamic Random Access Memory (DRAM) diagnostic, is a DEMAND type phase run on the system board DRAM. This phase takes about 5 minutes to execute. The following command line entry and system responses show the successful execution of the Extended Dynamic Random Access Memory diagnostic phase.

```
DGMON > dgn sbd ph=5

    <<<  DIAGNOSTIC MODE  >>>

SBD Phase: 5  Test: Extended Dynamic Random Access Memory  Type:  DEMAND
Time Taken = ~5 minutes
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46
47 48 49 50 51 52 53 54 55 56 57 58 59 60 61
*** SBD Phase 5 Diagnostic PASSED ***


     SBD 0 (IN I/O BUS SLOT 0) DIAGNOSTICS PASSED
```

INTERACTIVE Diagnostic Phase

Phase 17, Time-of-Day Clock diagnostic, is an INTERACTIVE type phase run on the system board. This phase takes about 15 seconds to execute. The following command line entry and system responses show the successful execution of this diagnostic phase.

```
DGMON > dgn sbd ph=17

    <<<  DIAGNOSTIC MODE  >>>

SBD Phase 17  Name: Time of Day Clock Test  Type: INTERACTIVE
Time Taken ~15 seconds

WARNING: This test will destroy Time Of Day Clock setting.
Do you wish to continue? [y/n]y
1 2 3 4 5
*** SBD Phase 17 Diagnostic PASSED ***

     SBD 0 (IN I/O BUS SLOT 0) DIAGNOSTICS PASSED

DGMON >
```

How to Leave the Diagnostic Monitor

There are two procedures available for leaving the Diagnostic Monitor when you receive the prompt (DGMON >).

- If you are in the Diagnostic Monitor because of a system failure message or an intermittent problem or if any of the phases failed during testing, go to "Procedure for Shutdown When a Problem Is Present."
- If you are in the Diagnostic Monitor to do nontrouble testing and all the phases passed, go to "Procedure for Rebooting the UNIX System."

Procedure for Shutdown When a Problem Is Present

If you entered the Diagnostic Monitor because of the system failure message or if any of the diagnostic phases failed, do the following:

1. Press the power switch to STANDBY. This causes the computer to immediately shutdown.
2. Contact your AT&T Service Representative or authorized dealer.

Procedure for Rebooting the UNIX System

The following procedure should only be used if all the diagnostic phases passed and the system failure message was not the reason for entering the Diagnostic Monitor.

1. Quit the Diagnostic Monitor. Enter **q**.
2. Execute **/unix** from the SCSI hard disk (load device option **1** SCSI, subdevice option **0** disk) to reboot the system to the multiuser state.

Chapter 4: Disk/Tape Management

Introduction

This chapter discusses the Small Computer System Interface (SCSI) disk and SCSI Cartridge Tape devices on your 3B2 computer. The topics are listed below:

- SCSI bus
- Disk device types and sizes
- Tape device sizes
- Making devices known to the operating system
- Formatting disks
- Making a Bootable Device
- Duplicating disks
- Duplicating cartridge tapes
- Verifying disk usability
- Using the bad block handling feature
- Using the disk mirroring feature.

This chapter does not discuss file systems or the information stored on disk devices. Those subjects are covered in Chapter 5, "File System Administration."

Device Types

The 3B2 computer has SCSI Cartridge Tapes and two types of disk devices: sealed, hard disk units and drives for removable floppy disks. Possible SCSI configurations permit you to have single or multiple hard disk units, cartridge tape units, and 9-track tape units. Additional SCSI Cartridge Tape Drives, SCSI hard disks, and SCSI 9-Track Tape Drives are available.

All system software and user files are kept on the hard disk device(s). The floppy disk device is used primarily as a means of getting user files into the system where they can be used, or out of the system for storage. The SCSI Cartridge Tape device is used primarily for file system backups.

SCSI Bus Addresses

There can be up to eight controllers on each SCSI bus including the Host Adapter. Each controller has a different SCSI address (0 through 7) which is set by a SCSI ID switch located on the controller. The SCSI address establishes the bus priority with device "7" having the highest priority. The SCSI ID switch for the Host Adapter is on the feature card and is factory preset to "0." The SCSI ID switches for the internal peripheral controllers are also factory preset. The disk controller is preset to ID No. 1, and the SCSI Cartridge Tape is preset to ID No. 2.

SCSI Host Adapter

The 3B2 SCSI Host Adapter (HA) gives the 3B2 computer an interface to the SCSI bus. The SCSI HA is equipped with two 50-pin connectors and can support up to seven peripheral controllers via the interface bus cables. One of these two connectors is used to connect the hard disk(s) and the SCSI Cartridge Tape Drive internally.

SCSI Host Adapter Cabling

Cabling to the Host Adapter involves two SCSI bus connectors that are located on the Host Adapter card inside the 3B2 computer cabinet. Typically, the internal SCSI bus cable connects to one of these two connectors, while a bus terminator connects to the other.

Warning: **Do not tamper with the SCSI HA connectors. If a SCSI cable or terminator is connected incorrectly, irreparable damage may occur to the SCSI hardware.**

SCSI Hard Disk Devices

Your 3B2 computer is equipped with either 155–megabyte or 300–megabyte SCSI hard disks:

The units are sealed to protect them from dust, smoke, and other contaminants in the air. Sealing has both advantages and disadvantages. It means that the 3B2 computer can operate without special dust-free climate control, but it also means that disk storage packs cannot easily be swapped in and out of the computer.

SCSI Cartridge Tape Drive

Note: This SCSI Cartridge Tape Drive should in no way be confused with the AT&T 3B2 Cartridge Tape Controller (CTC) drive currently available with some 3B2 computers. The tapes written on these two cartridge tape drives are not compatible and should be kept separate.

Warning: **If you attempt to write a cartridge tape written on a noncompatible tape drive, you may destroy the information on the tape.**

The primary purpose of the SCSI Cartridge Tape Drive is for backing up and restoring the 3B2 computer file systems.

Because you can dump a lot of data to a tape quickly, having a SCSI Cartridge Tape Drive may influence the way you do file system backups. Complete backups may work better for you than a combination of complete and incremental backups. See Chapter 5, "File System Administration," for a discussion of backups.

Floppy Disk Drives

A 3B2 computer comes with one integral floppy disk drive. The drive accommodates 5¼-inch floppy disks. (You sometimes hear these floppy disks called "diskettes.") As a raw device, a floppy disk holds 1422 blocks (512-byte blocks) of data. Most microcomputer software is distributed on floppy disks. The standard procedure is to read the software in from a floppy disk and store it on the hard disk device. The reverse path is used to make backup copies of data stored on the hard disk.

It is possible to use the floppy disk drive(s) as the way to get programs and data into main memory for processing. On smaller microcomputers that do not have hard disk units, this is not only possible, but is a must. This is not, however, the optimal way of working if hard disks are available. The larger storage capacity and faster access time of the hard disk make it much more desirable. In addition, using the floppy disk drive in this way means that it is unavailable for its more normal use.

Identify Devices to the Operating System

Before a disk or tape device can be used on a computer running the UNIX operating oystem, it must be made known to the system. For equipment that comes with your computer, the process of identifying devices is part of configuration and is done automatically as the system is booted.

The traditional way of handling the identification is through an entry in the **/dev** directory of the **root** file system. Of course, an entry in a directory is a file (or another directory), and conceptually a disk device is treated as if it were a file. There is a difference, however, which leads to the practice of referring to devices as "special" files. In the place where a regular file would show the character count for the file, for a special file you find two decimal numbers called the major and minor numbers. Figure 4-1 shows excerpts from the output of **ls -l**(1) command on a user's directory and the **/dev** directory structure.

```
                    (a regular file)

-rw-r-----   1 abc      dsg          1050 Apr 23 08:14 dm.o1

                    (SCSI hard disk device files)

brw-------   2 root     root      121,  0 Apr 15 10:59 /dev/dsk/c1t1d0s0
brw-------   2 root     root      121,  1 Apr 12 13:51 /dev/dsk/c1t1d0s1

crw-------   2 root     root      121,  0 Apr 15 10:58 /dev/rdsk/c1t1d0s0
crw-------   2 root     root      121,  1 Apr 12 13:51 /dev/rdsk/c1t1d0s1

                    (floppy disk device files)

brw-------   2 root     sys        47,128 Apr 12 13:51 /dev/dsk/c0d0s0
brw-------   7 root     sys        47,134 Apr 22 17:56 /dev/dsk/c0d0s6

crw-------   2 root     sys        47,128 Apr 12 13:51 /dev/rdsk/c0d0s0
crw-------   7 root     sys        47,134 Apr 12 13:51 /dev/rdsk/c0d0s6

brw-------   7 root     sys        47,134 Apr 22 17:56 /dev/SA/diskette1
crw-------   7 root     sys        47,134 Apr 12 13:51 /dev/rSA/diskette1

                    (SCSI cartridge tape device files)

crw-rw-rw-   2 root     root      122, 10 May 17 12:01 /dev/rSA/qtape1
crw-rw-rw-   2 root     root      122, 10 May 10 19:03 /dev/rmt/c1t2d0s0
crw-rw-rw-   1 root     root      122, 11 May 15 01:24 /dev/rmt/c1t2d0s0n
```

Figure 4-1: Directory Listing Extracts: Regular and Device Files

The extracts from directory listings in Figure 4-1 show a regular file [specified by the dash (-) in the first position of the line] with these characteristics.

- It has 1050 characters.
- The file name is **dm.o1**.
- It is owned by user **abc** who is a member of **dsg** group.

- The owner has read/write permission; group members have read permission; other users have no permissions.

There are also device files with these characteristics:

- Major and minor numbers appear in place of the character count.

  Major is the number of the device controller or driver (actually, an offset into a table of devices in the kernel); minor is the identifying number of the specific device.

- There are devices that have identical major and minor numbers, but they are designated in one entry as a block device (**b** in the first column) and in another entry as a character device (**c** in the first column).

  Notice that such pairs of files have different file names or are in different directories (for example, **/dev/SA/diskette1** and **/dev/rSA/diskette1**).

- There are alias names, for example, file **diskette1** is an alias for **c0d0s6**.

- The files are owned by **root**, and no group or other user has any permission to use them. This means that only processes with the **root** ID can read from and write to the device files. However, there are several exceptions to this rule. (The SCSI Cartridge Tape Drive is one exception to this rule.)

Block and Character Devices

The identification as a block device or a character device has more to do with how the device is accessed than with any physical characteristics. A block device name is used when the intent is to read from or write to the device in logical block sizes (512-, 1024-, or 2048-byte blocks). In the UNIX system, standard C language subroutines for handling file I/O work with blocks.

A character device name is used to read from or write to the device one character at a time. A character device is also referred to as a "raw" device. This is reflected in the device names or directory names shown in Figure 4-1, where the character device version of the floppy disk drive is in directory **/dev/rdsk**. The character-at-a-time method is used by some file maintenance

utilities. The SCSI Cartridge Tape Drive is a character device only; it cannot be accessed as a block device.

Define a New Special File

The need to define new special device files occurs infrequently. If you add devices, the autoboot process takes care of defining the files. When the need does occur, however, there is a UNIX system command, **mknod**(1M), available to do it.

The general format of the **mknod** command follows:

mknod *name* **b** | **c** *major minor*

mknod *name* **p**

The options of **mknod** listed below:

name	Specifies the *name* of the special file.
b	Specifies a block device.
c	Specifies a character device. The "or"sign (|) means you must specify one or the other.
major	The *major* number is the slot number.
minor	The *minor* number is the physical device.
p	Specifies the special file as a first-in, first-out device. This is also known as a named pipe. (For more on pipes, see the *Programmer's Guide*.)

Format and Partitions

Format means to establish addressable areas on a medium. Partition means to assign file systems or other logical units to an addressable area. This section covers the formatting and partitioning of disks. Since SCSI Cartridge Tapes are formatted as data is written to them, they do not have to be formatted by the user.

Format Disks and Floppy Disks

Before a disk can be used for the storage of information, it must be formatted. Until the medium is formatted, its surfaces are simply uncharted areas treated with a substance that accepts and holds magnetic charges. Formatting imposes an addressing scheme onto these magnetic surfaces. 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 Volume Table Of Contents (VTOC) resides in that area. The VTOC shows how the partitions on the disk are allocated. On a hard disk, an additional use of the reserved area is to map 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.

You will have much more need to format floppy disks than hard disks. It is a good practice to format an entire box of floppy disks at the time the box is first opened. By formatting floppy disks on a box basis, you avoid the problem of keeping track of which ones are or are not formatted (that is, if the box is open, it means all the floppy disks are formatted). Floppy disks are formatted by the UNIX system **fmtflop**(1M) command or the **sysadm format**(1) subcommand.

The **fmthard**(1M) command, which sounds as though it might be used to format hard disks, is used on devices other than the root device to allocate partitions on a hard disk and to install a VTOC.

Hard disks are shipped from the factory already formatted. You can use the **sysadm format**(1) command to format a SCSI disk, provided the disk partitions are unmirrored and unmounted. "The Disk Mirroring Feature" is described later in this chapter. (Also see the *AT&T 3B2 Computer SCSI Release 3.0, SCSI Operations Manual* for information on disk mirroring and shared

peripherals.) When you enter the **sysadm format** command, the system responds with the following:

```
More than one subcommand or submenu matches 'format'.
  1  diskmgmt/format  2  diskmgmt/harddisk/format
Select one: [?, q] 2
```

The system will prompt you to select the drive to format.

The file systems that are on that drive will be destroyed if you continue, and the system notifies you of what will be destroyed. After the disk is formatted the system will try to access it, and fail. The system will generate the following error message:

```
Warning: SD00: Bad sanity word in the VTOC on disk 2, tc 3, slot 1.
```

This message only means that there is no VTOC on the hard disk. When you partition this disk, a VTOC will be written onto the disk.

SCSI Hard Disk Partitions

Partitions on the hard disk devices on your 3B2 computer are allocated in a standard arrangement. Appendix A, "Device Names and Designators," describes the default partitioning.

The 3B2 computer's single disk system has **usr** and **root** on the same disk, and a dual disk system has **usr** on the second disk while **root** and **usr2** share the first. Hard disks are partitioned using the **sysadm partition** command under the Disk Management Menu. Partitions do not have to be the same size. The total number of blocks assigned (all assigned partitions) cannot be greater than the number of blocks available.

There can be up to 8 user–defined partitions (8 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. Partitions 8 through 15 can be assigned by the Administrator.

The default partitions have been defined for optimal use of the average UNIX system. After your system has been in operation for a few months, you may come to feel that a different arrangement would better serve the needs of your users.

Plans to Change Hard Disk Partitions

The basic question in reaching a decision about repartitioning your hard disk devices is whether you would be better off having many smaller file systems or staying with **usr** and **usr2** (for which the default is the entire disk). Here are some additional questions that need to be considered.

- What group IDs are defined? Do we have the right number of groups and are users assigned to them appropriately?
- What processing is done by the user 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 the others?
- Have we added, or are we planning to add, system software that changes our thinking about space requirements?
- Will disk partitions be mirrored?

Helpful information about the performance of your existing file system arrangement can be obtained with the **sar**(1M) command. The use of **sar** is described in Chapter 6, "Performance Management."

If you decide that repartitioning is needed, it can be done with a full system restore (see Procedure 3.9, "Reload the Operating System"). The **sysadm partitioning** or **fmthard** commands can be used to redefine partitions on disks other than the **root** and **usr** disks.

Change Partitions to Increase Swap Space

If you frequently get console messages warning of insufficient memory, it may mean that the system's current configuration of main memory and swap area is insufficient to support user demands. Before adding more main memory, an alternate solution is to expand the swap area (on either single or multiple hard disk systems).

The swap space can be increased by either repartitioning the root disk (full restore) or by designating additional disk partitions for the swap space using the **/etc/swap**(1M) command.

Using the Full Restore

The Full Restore procedure is described in Procedure 3.9, "Reload the Operating System." Before you run the procedure, there are three things you should do.

1. Find out about your present partitions. (Check Appendix A, "Device Names and Designators," for default partitioning, or if your system has been previously repartitioned so the defaults no longer apply, check your records or use the **sysadm display** menu to get the information.)
2. Decide what the new partition sizes should be. (The disk is already fully allocated, so increasing the size of the swap partition means you have to reduce the size of one other partition.)
3. Do a complete backup (see Procedure 5.4, "File System Backup and Restore"). (The process of changing partitions is going to erase everything that is now there.)

You are now ready to go ahead with Procedure 3.9, "Reload the Operating System."

Using the swap Command

Adding Swap Space

Using the **/etc/swap**(1M) command to designate additional swap disk space requires the following.

1. The disk partition to be used for additional swap space must be unmounted and not listed in the **/etc/fstab** file.

2. A file containing the **/etc/swap** command must be put in the **/etc/rc2.d** directory.

Refer to the **swap**(1M) manual page for more information. An example of using the **/etc/swap** command to add a 21105–block disk partition **/dev/dsk/c1t1d0sa** to the swap area(s) is as follows:

```
# cat /etc/rc2.d/S03swap
/etc/swap -a /dev/dsk/c1t1d0sa 0 21105
```

Report Swap Area Status

The status of the swap area(s) is reported by the **/etc/swap -l** command.

```
# swap -l
path               dev  swaplo blocks   free
/dev/dsk/c1t1d0s1 121,1       0  20640  20640
/dev/dsk/c1t1d0sa 121,10      0  21104  21104
#
```

Delete a Swap Partition

A disk partition being used as part of the swap area can be removed from the swap area using the **/etc/swap -d** command. The swap area status is displayed before and after the delete option to show the result of the **swap -d** command.

```
# swap -l
path                dev  swaplo blocks   free
/dev/dsk/c1t1d0s1 121,1       0  20640  20640
/dev/dsk/c1t1d0sa 121,10      0  21104  21104
# swap -d /dev/dsk/c1t1d0sa 0
# swap -l
path                dev  swaplo blocks   free
/dev/dsk/c1t1d0s1 121,1       0  20640  20640
#
```

Make a Bootable Device

The devices that can be used as a boot device are the integral floppy disk (diskette1), or the first hard disk of any target controller.

Note: Although the Operating Systems Utilities is delivered on a cartridge tape, you cannot use other cartridge tapes to boot the system.

A device is made bootable by copying the applicable boot programs to the boot partition (7) of the device. The **/etc/newboot** command is used to write these programs to the boot partition of the device. Except for a floppy disk, the boot partition must be at least 100 blocks (512-byte blocks). The floppy disk boot partition must be at least 18 blocks.

The major steps in making a device bootable are as follows:

1. Format/partition the media to include a boot partition.
2. Copy the applicable boot programs to the boot partition, using the **/etc/newboot** command.
3. Make file system(s) on the applicable device partitions.
4. Label the file systems.
5. Mount the applicable partition(s) and copy the program(s) to be booted to the applicable file system(s). Also copy any supporting files required by the programs.
6. Unmount the file system(s) when you have finished copying programs and data.
7. If the bootable device is a removable media (floppy disk), clearly label the floppy disk.

Make a Bootable Floppy Disk

The important points to remember in making a bootable floppy disk are as follows:

- The boot partition for the integral floppy disk is c0d0s7.
- The file system partition is c0d0s5 and has a maximum of 1404 blocks.
- The **newboot** command must specify **/lib/olboot** and **/lib/mboot**.

The following shows how to make a bootable floppy disk for running diagnostics. The floppy disk is first formatted using the **/etc/fmtflop** command. The boot programs are copied to the boot partition (/dev/rdsk/c0d0s7) using the **/etc/newboot** command. After making a file system on partition 5 (/dev/dsk/c0d0s5), the partition is labeled to identify the floppy disk. Partition 5 (c0d0s5) is then mounted as **/mnt**, and the applicable programs are copied to the **/mnt** file system. The floppy disk file system (c0d0s5) is then unmounted using the **/etc/umount** command. The system is then shut down to the firmware mode, and the diagnostic monitor (**dgmon**) is booted from the floppy disk.

```
# fmtflop -v /dev/rdsk/c0d0s6
# newboot /lib/olboot /lib/mboot /dev/rdsk/c0d0s7
newboot: confirm request to write boot programs to /dev/rdsk/c0d0s7: y
# mkfs /dev/dsk/c0d0s5 1404 1 18 -b 1024
Mkfs: /dev/dsk/c0d0s5?
(DEL if wrong)
bytes per logical block = 1024
total logical blocks = 702
total inodes = 160
gap (physical blocks) = 1
cylinder size (physical blocks) = 18
mkfs: Available blocks = 689
# labelit /dev/rdsk/c0d0s5 dgn 3223
Current fsname: , Current volname: , Blocks: 1404, Inodes: 160
FS Units: 1Kb, Date last mounted: Thu Mar 28 14:19:08 1989
NEW fsname = dgn, NEW volname = 3223 -- DEL if wrong!!
# mount /dev/dsk/c0d0s5 /mnt
mount: warning! <dgn> mounted as </mnt>
# find /dgmon /dgn /filledt -print | cpio -pdumv /mnt
/mnt/dgmon
/mnt/dgn/.edt_swapp
/mnt/dgn/MAU
/mnt/dgn/PORTS
/mnt/dgn/SBD
/mnt/dgn/SCSI
/mnt/dgn/VCACHE
/mnt/dgn/X.MAU
/mnt/dgn/X.PORTS
/mnt/dgn/X.SBD
/mnt/dgn/X.SCSI
/mnt/dgn/X.VCACHE
/mnt/dgn/edt_data
/mnt/dgn/EPORTS
/mnt/dgn/X.EPORTS
/mnt/filledt
864 blocks
# umount /dev/dsk/c0d0s5
```

Continued

```
Continued from previous screen display
# shutdown -y -g0 -i5

        series of messages are displayed
        ending with the following

FIRMWARE MODE
<mcp><CR>

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

Option Number    Slot      Type         Name
-----------------------------------------
       0          0      INTEGRAL     FD5
       1          1      I/O BUS      SCSI

Enter Load Device Option Number [1 (SCSI)]: 0

        3B2 DIAGNOSTIC MONITOR
DGMON >
```

Make a Second Target Controller's Hard Disk Bootable

The important points to remember in making a second target controller's hard disk bootable are as follows:

- The hard disk must be the first hard disk of a target (disk) controller.
- The boot partition is c1t?d0s7 (? is the target controller number) and must be 100 blocks, minimum.
- The **newboot** command must specify **/lib/lboot** and **/lib/mboot**.
- The swap partition is c1t?d0s1 and is sized to make the next partition fall on a cylinder boundary.

- The **root** file system partition is c1t?d0s0 and has a block size of whatever you assigned when you partitioned the disk. File system partitions should begin on cylinder boundaries. The **/etc/prtvtoc** command is used to display the current partition values.
- Back up any existing file systems before repartitioning an existing second hard disk.
- If configuring a second hard disk to boot the UNIX operating system, remember to edit the **/etc/fstab** file on the second disk to define the appropriate file system mount information.
- The first time the UNIX operating system is booted from another device, **/etc/system** must be booted.
- The first time the UNIX operating system is booted from another device, the swap partition and the swap device are a mismatch. An error message is output when the operating system is booted for the first time from the new device. The swap device is changed automatically.
- The Full Restore process of the UNIX system supports only the configuration of root and swap being on the first hard disk drive.

The following example shows how to make a second hard disk bootable for running the UNIX operating system. The second hard disk is first partitioned using the **/etc/fmthard** command. In this example, the default volume table of contents for the first hard disk is used for the partition values by using the output of the **/etc/prtvtoc** command to create a VTOC file for use with the **/etc/fmthard** command. The second hard disk must be the same size as the first hard disk for this to work. The boot programs are copied to the boot partition (/dev/rdsk/c1t4d0s7) of the second hard disk using the **/etc/newboot** command. The root (/) file system (c1t1d0s0) is copied to partition 0 (c1t4d0s0) using the **/etc/volcopy** command. The root file system on the second disk is then checked using **/etc/fsck**. The system is then shut down to the firmware mode and **/etc/system** is booted from the second disk.

```
# prtvtoc -s /dev/rdsk/c1t1d0s0 | tee /etc/vtoc/newvtoc
*                          First    Sector    Last
* Partition    Tag  Flags  Sector    Count    Sector   Mount Directory
       0        2    00     20790    36855     57664   /
       1        3    01       150    20640     20789
       3        0    01    269632    32768    302399
       4        0    00    268065     1567    269631
       6        0    01         0   302400    302399
       7        0    01         0      150       149
       8        0    00     57645   201420    268064   /usr2
# fmthard -m -s /etc/vtoc/newvtoc -n root -v /dev/rdsk/c1t4d0s0
fmthard:  New volume table of contents now in place.
+(fmthard) mkfs /dev/rdsk/c1t4d0s0 36855 3 315
Mkfs: /dev/rdsk/c1t4d0s0?
(DEL if wrong)
bytes per logical block = 2048
total logical blocks = 9213
total inodes = 4592
gap (physical blocks) = 3
cylinder size (physical blocks) = 315
mkfs: Available blocks = 9086
+(fmthard) mkfs /dev/rdsk/c1t4d0s4 1567 3 315
Mkfs: /dev/rdsk/c1t4d0s4?
(DEL if wrong)
bytes per logical block = 2048
total logical blocks = 391
total inodes = 192
gap (physical blocks) = 3
cylinder size (physical blocks) = 315
mkfs: Available blocks = 384
+(fmthard) mkfs /dev/rdsk/c1t4d0s8 210420 3 315
Mkfs: /dev/rdsk/c1t4d0s8?
(DEL if wrong)
bytes per logical block = 2048
total logical blocks = 52605
total inodes = 26288
gap (physical blocks) = 3
cylinder size (physical blocks) = 315
mkfs: Available blocks = 51782
```

Continued

Continued from previous screen display

```
# prtvtoc -s /dev/rdsk/c1t4d0s0
*                            First    Sector    Last
* Partition    Tag  Flags    Sector    Count    Sector   Mount Directory
       0        2    00      20790    36855     57664   /
       1        3    01        150    20640     20789
       3        0    01     269632    32768    302399
       4        0    00     268065     1567    269631
       6        0    01          0   302400    302399
       7        0    01          0      150       149
       8        0    00      57645   201420    268064   /usr2

# newboot /lib/1boot /lib/mboot /dev/rdsk/c1t4d0s7
newboot: confirm request to write boot programs to /dev/rdsk/c1t4d0s7: y
# volcopy root /dev/rdsk/c1t1d0s0 - /dev/rdsk/c1t4d0s0 -
/dev/rdsk/c1t4d0s0 less than 48 hours older than /dev/rdsk/c1t1d0s0
To filesystem dated:  Wed Mar 30 06:22:44 1988
Type 'y' to override:     y
warning! from fs(root) differs from to fs()
Type 'y' to override:     y
From: /dev/rdsk/c1t1d0s0, to: /dev/rdsk/c1t4d0s0? (DEL if wrong)
  END: 36850 blocks.
# fsck -D /dev/rdsk/c1t4d0s0

  /dev/rdsk/c1t4d0s0
  File System: root Volume:

  ** Phase 1 - Check Blocks and Sizes
  ** Phase 2 - Check Pathnames
  ** Phase 3 - Check Connectivity
  ** Phase 4 - Check Reference Counts
  ** Phase 5 - Check Free List
  FILE SYSTEM STATE SET TO OKAY
  645 files 9356 blocks 26920 free
  *** FILE SYSTEM WAS MODIFIED ***
```

Caution

/dev/root and /dev/swap need to be recreated on the target with correct major numbers.

Continued

Continued from previous screen display

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

series of messages are displayed
ending with the following

FIRMWARE MODE

```
<mcp><CR>

Enter name of program to execute [   ]: /etc/system
             Possible load devices are:

Option Number     Slot      Type          Name
-------------------------------------------
       0           0      INTEGRAL      FD5
       1           1      I/O BUS       SCSI

Enter Load Device Option Number [1 (SCSI)]: 1
Possible subdevices are:

Option Number    Subdevice     Name
-------------------------------------------
       0            0          disk
       1            1          tape
       2            2          disk
       3            3          disk

Enter Subdevice Option Number [0 (disk)]: 2

CONFIGURATION SUMMARY
=====================
```

series of messages are displayed
ending with the following

Continued

Continued from previous screen display

```
UNIX System V Release 3.2.2 3B2 Version 3
Node unix
Total real memory   = 12582912
Available memory    = 8912896

********************************************************************************

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.
```

series of messages are displayed
ending with the following

```
Console Login:
```

Assignment of Default Boot Program and Device

Caution: Before changing the default boot device from option 1 subdevice option 0 (the first hard disk off the first target controller) to another device option, make the other device bootable. This helps to avoid the possibility of autobooting from a device that is not properly configured.

General

The assignment of a default program name and device to be used when the system is manually booted are assigned using the **sysadm autold** command. The term "boot" is used to mean loading and executing a program from the firmware mode. When the system is delivered, the boot program name is null and the boot device is the first hard disk off the first SCSI target controller (option 1, subdevice option 0). When the boot program name is null, the auto boot program name defined in Nonvolatile Random Access Memory (NVRAM) is passed to the boot process. This name is **/unix**. Only **/unix** or **/etc/system** can be used as auto boot programs. Other program names must be manually booted. Booting **/unix** or **/etc/system** from a particular device can be done either manually or automatically. Any equipped and properly configured subdevice hard disk (disk 0 off a target controller), or floppy disk (diskette1) can be used as a boot device. If the **/unix** file on a given device was not generated after booting the device, the operating system parameters will not be properly set and the operating system will not run. Therefore, when booting the UNIX operating system from a device for the first time, or when in doubt, boot **/etc/system**. Booting **/etc/system** remakes the operating system core image. Auto-configuration (**/etc/rc.2d/autoconfig**) then copies the core image to the **/unix** file on the boot device. Refer to the "Making a Bootable Device" description for information on conditioning a particular device to be used as a boot device.

Set Automatic Boot Device Procedure

The **sysadm autold** command is used to output and/or change the current default boot program name and device. The command can be executed in any run level in which the UNIX operating system is running (run levels 1, 2, 3, 4, s, or S) and the **/usr** file system is mounted. Note that the program name can be changed to a null by entering a space for the program name.

The following shows the use of the **sysadm autold** command to display the current values.

```
# sysadm autold

Running subcommand 'autold' from menu 'machinemgmt',
MACHINE MANAGEMENT

You may specify the default file for manual load, the device for auto load,
or both.

Typical files to be loaded are /unix, a fully configured UNIX, or /etc/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.

Change the manual load program or auto load device? [y, n, q] y

Enter name of default program for manual load [ /unix ]: <CR>
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          1          SCSI
enter number corresponding to autoload device desired [ 1 ]:<CR>
NULL response detected, current value will be maintained
```

Continued

```
Continued from previous screen display

Possible subdevices are:

Option Number    Subdevice      Name
--------------------------------------
       0            0              disk
       1            1              tape

Enter Subdevice Option Number [0(disk)]: <CR>
NULL response detected, current value will be maintained

LOAD PARAMETER UPDATE COMPLETE

Select what to do next:
                continue this session
                firmware
                powerdown
                reboot
[c, f, p, r, ?]: c
#
```

The following shows the use of the **sysadm autold** command to change the current values. The default program name is changed to **/etc/system**; the default device option subdevice is changed to a 1 (the SCSI Cartridge Tape).

```
# sysadm autold

Running subcommand 'autold' from menu 'machinemgmt',
MACHINE MANAGEMENT

You may specify the default file for manual load, the device for auto load,
or both.

Typical files to be loaded are /unix, a fully configured UNIX, or /etc/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.

Change the manual load program or auto load device? [y, n, q] y

Enter name of default program for manual load [ /unix ]: /etc/system

Possible load devices are:

Option Number    Slot    Name
-----------------------------------------
       0          0          FD5
       1          0         SCSI
enter number corresponding to autoload device desired [ 1 ]:<CR>
NULL response detected, current value will be maintained
```

Continued

Continued from previous screen display

```
Possible subdevices are:

Option Number   Subdevice   Name
--------------------------------------
       0           0          disk
       1           1          tape

Enter Subdevice Option Number [0(disk)]: 1

LOAD PARAMETER UPDATE COMPLETE

Select what to do next:
                continue this session
                firmware
                powerdown
                reboot
[c, f, p, r, ?]: c
#
```

Other Disk/Tape Operations

Additional operations you may need to do from time to time are duplicating disks and verifying the usability of a disk or tape.

Duplicate Disks

The contents of an existing floppy disk are copied to another floppy disk by using the **dd**(1) command or the **sysadm cpdisk**(1) subcommand. If your computer has a single floppy disk drive, the contents of the floppy disk to be duplicated are first copied to a temporary file on the hard disk. When **dd** is used, either the character or the block device can be specified, but the same device (character or block) must be specified for the entire procedure. The source floppy disk is then replaced and the temporary file copied to the destination floppy disk.

While any file system can be used for the temporary file space, it is better to use space in either **/tmp** or **/usr/tmp**. Files in those two directories are automatically deleted during the transition to the multi-user mode (run level 2). No matter which directory is specified, a minimum of 1422 blocks must be available (free) in that file system to use as the temporary file space. If you have more than one floppy disk drive, the temporary file on hard disk is not needed.

Verify Usability

An option, **-v**, of the **fmtflop**(1M) command does a verification check to see if the formatting was done without error. When a floppy disk contains data, however, a different procedure must be used to find out if the floppy disk is usable (that is, can be read from accurately).

The integrity of the storage medium of a formatted floppy disk can be verified without changing the data on the disk by using the **dd**(1) command. The technique is to copy the data on the disk to **/dev/null**. Either the character (**/dev/rSA/diskette1**) or the block (**/dev/SA/diskette1**) device can be specified as the source. On completion, the **dd** command reports the number of whole and partial data blocks processed (input and output). A "good" floppy disk provides 158 blocks (4608-byte blocks). The block size of 4608 [must be specified (bs=4608)] is the number of bytes per track (9 times 512 bytes). Specifying a block size that matches the number of bytes per track

is the most efficient way to do the verify procedure. By directing the output to a null file (**/dev/null**), this procedure is independent of the amount of free disk space, and it requires no file cleanup at the end of the copy.

Duplicate SCSI Cartridge Tape

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. To copy the records into a single file, the ULIMIT of the system may need to be changed. After the contents of the Cartridge Tape have been written to the hard disk, they must be written sequentially out to the new cartridge tape.

Figure 4-2 shows an example of a shell script to copy a SCSI Cartridge Tape. This example can only be executed by root, because it changes the ULIMIT and uses the **mknod**(1M) command to create character special files. The example shell script uses a temporary directory as the destination of the records that are copied. You need to be sure that there is room on your system for the copy. The example script supports both 60– and 120–MB cartridge tape drives. Remember that a 120–MB drive can read both 60– and 120–MB tapes, but can write only in 120–MB format. Therefore, a copy of a 60–MB tape made on a 120–MB drive can be read only on a 120–MB drive.

```
PATH=/usr/lbin:$PATH; export PATH
tpl=`selectdevice -q "Select which tape device:" -c $$ /dev/rSA qtape`
tp2=`ls -l $tpl | sed 's/.*\(...\),.*$/\1/'`
no_rewind=/tmp/no_rewind$$
rewind=/tmp/rewind$$
mknod $rewind c $tp2 0
mknod $no_rewind c $tp2 1
< $rewind
< $rewind
echo "\n\n\nEnter directory to be used for temporary storage: \c"; read name
node="$name"
count=0
case $1 in
    "in") if test -d $name; then
            echo "\nDirectory $name already exists!  Continue? (y or n):  \c"
            read answer
            if test $answer = "n"; then exit; fi
        else
            if `mkdir $name`; then echo "\nMaking $name directory"
            else exit
            fi
        fi
        LIMIT=`ulimit`
        ulimit 2048000
        echo "\nCopying records from $tpl to $name"
        echo "\n       The copy is now in progress.  Be patient!!!\n"
        while `dd if=$no_rewind of=$node/$count bs=10k 2> /dev/null`
            do
                count=`expr $count + 1`
                echo ".\c"
            done
        ulimit $LIMIT;;
```

Figure 4-2: Duplicate Cartridge Tape Shell Script (Sheet 1 of 2)

```
    "out") if test -d $name; then
            stop=`ls $node | sort -n | tail -1`
            echo "\nWriting $stop records to $tpl"
            echo "                    Be patient!!!\n"
            while test $stop -gt $count
                do
                    echo ".\c"
                    dd if=$node/$count of=$no_rewind bs=10k 2> /dev/null
                    count=`expr $count + 1`
                done
            echo "\n\nAfer all copies are made, remember to"
            echo "clean up the $node directory.\n"
        else
            echo "Directory $name does not exist!"
        fi;;
    *) echo Usage: copy in  - reads load tape into $node directory.
        echo "       copy out - generates load tape from" $node "directory.\n"
        exit;;
esac
< $rewind
rm $rewind $no_rewind
echo "\n\nCopy $1 Complete!!!\n"
```

Figure 4-2: Duplicate Cartridge Tape Shell Script (Sheet 2 of 2)

The executable file is named "copytape." To copy data from a SCSI Cartridge Tape to temporary files on the hard disk, enter **copytape in**. To copy the temporary files created by the **copytape in** process to another cartridge tape, replace the source cartridge tape with the cartridge tape on which you are making the copy and enter **copytape out**. If you want additional copies, you need to make them before you remove the files in the temporary directory. After you finish, the files in the temporary directory and the directory need to be removed.

The Bad Block Handling Feature

Note: This feature applies only to hard disk device(s). There is no comparable feature for floppy disks or tapes.

The 3B2 computer has a software feature called bad block handling. The purpose of this feature is to extend the useful life of the hard disks by providing mechanisms for the following:

- Detecting and remembering blocks that are no longer usable
- Reminding you that you need to "fix" some remembered bad blocks
- Restoring the usability of the disk in spite of the bad blocks that exist.

It should be pointed out that new bad blocks seldom occur, particularly with the sealed environment of the hard disk, as long as you take reasonable precautions against movement or vibration of the computer while the disk is still spinning. But when a new bad block does occur, the data stored in the bad block is lost and the disk may be unusable in its current state.

The bad block handling feature addresses the problem of restoring the usability of the disk. However, you must address the data loss yourself with whatever backup procedures you deem appropriate for your system. Backup procedures are also needed to protect against operational errors and other types of hardware failures. These other problems, particularly operational errors, are the dominant cause of data loss on a system. System backups provide protection against operational errors as well as protection against lost data because of new bad blocks. A discussion of backup procedures can be found in Chapter 5, "File System Administration."

When Is a Block Bad?

A block is bad when it cannot reliably store data. This is discovered only when an attempt is made to read the data and the read fails. A read failure does not always guarantee a bad block. A read failure might also mean problems in the format of the disk or a failure in the controller or the hardware.

A write failure generally signals a problem with the format of the disk or a more basic failure in the disk or disk controller hardware. While all failures are reported, the bad block handling feature does not distinguish genuine bad

blocks from format problems or hardware problems. To fix problems of those types you will need to reformat the disk or get the hardware repaired. In either case, you should call your AT&T Service Representative or authorized dealer. Several distinct failures occurring about the same time should also prompt you to contact your AT&T Service Representative or authorized dealer to check them out.

What Makes a Block Unreliable?

A disk is an analog medium used to store digital data. The analog phenomena involve the magnetic properties of the film coating on disk surfaces. The data is recorded with a high-bit density to get millions of bits in a small space. Because of the density, the significance of small variations in the magnetic properties of the recording medium become magnified. The variations mean that a given portion of the medium may prefer to represent some bit patterns and dislike representing others. Normally, these preferences are insignificant, compared to the signal level thresholds. When variations pass the point of being insignificant, the disk has a bad block. If the data pattern in a block matches the preferences in the recording medium, the bad block may escape recognition for a while. If the disk is active, however, the block will eventually be judged unreliable.

How Are Bad Blocks Fixed?

It is not really so much that a bad block is fixed, but the system finds a way to live with it. A small portion of the disk is set aside from the normally accessible portion of the disk. This portion, call it the media-specific data area, cannot be reached by normal UNIX system commands and system calls. This reserved portion of the disk contains a description of the properties of the disk and other media-specific data.

The media-specific data portion of the disk includes a set of blocks called the surrogate image region. The mechanism for preserving the apparent accessibility of most disk blocks is to use surrogate image blocks to contain the data that were to have been stored in the bad blocks. The media-specific data also includes a mapping table that maps bad blocks on the disk to these surrogate image blocks. The disk driver software in the operating system translates disk accesses so the data is read from or written to the surrogate image block. This disk address translation is transparent to the calling software.

Most disks come with a few manufacturing defects. Bad blocks detected in the manufacturer's quality control checks are identified on a label when the unit is delivered. The bad block handling feature provides special software for remembering bad blocks that have been found and for mapping any additional ones that are found. If a surrogate block becomes bad, the software even remaps the original bad block to a new surrogate block.

A Few Blocks Cannot Be Mapped

A few special blocks, all in the media-specific data portion of the disk, cannot be mapped:

- The disk block containing the physical description of the disk
- The disk block(s) containing the mapping table.

All other blocks, including surrogate image blocks, can be mapped.

When Are Bad Blocks Detected?

Bad blocks are detected when input/output disk operations fail for several successive attempts. This means that data being input or output is lost, but the system can restore use of the disk by mapping bad blocks to surrogate blocks that are readable.

Often Asked Questions

Why doesn't the system try to discover that a given block is bad while the system still has the data in memory?

Aside from the undesirable increase in system size and complexity, severe performance degradation would result. Also, a block can become a bad block after the copy in memory no longer exists.

Why doesn't the system periodically test the disk for bad blocks?

Reading blocks with their current contents may not show a bad block to be bad. A thorough bit pattern test would take so long that you would never run it, even assuming a thorough test could be devised using ordinary write/read operations. The disk manufacturer already has

tested the disk using extensive bit pattern tests and special hardware. All manufacturing defects have been dealt with already.

Why are disks with manufacturing defects used?

Allowing the disks to contain a modest number of manufacturing defects greatly increases the yield, thereby considerably reducing the cost. Many systems, including this one, take advantage of this cost reduction to provide a more powerful system at lower cost.

How Bad Block Handling Works

The bad block handling feature provides the mechanisms to detect, remember (where feasible), and add new bad blocks to the existing map. In normal operations the mechanisms are automated. Some cases do occur that require special handling, and even the automated cases have special properties at some stages of the processing. The remainder of this section describes:

- Normal operation
- Handling exceptional cases
- Fixing bad blocks manually.

Bad Block Handling: Normal Operation

The automated mechanisms are part of the normal UNIX system execution environment. The error logging daemon is running when the 3B2 computer is operating in multiuser mode (run level 2). The first phase of automated mechanisms involves detecting new bad blocks and remembering them for later mechanisms.

An understanding of the ways of referring to disk blocks helps in understanding the examples that follow.

- A block number is an integer counter.
- A physical block number is the integer counter that describes how the sectors are numbered on the disk. (For example, physical block 3 is sector 3 of head 0 of cylinder 0.)

- A logical block number is the counter for sectors, starting with 0 at the portion of the disk where disk partitions begin.
- A partition block number is the counter for sectors, starting with 0 at the beginning of the partition.
- A file system block number is the counter for file system blocks, starting with 0 at the beginning of the first partition in the file system.

A Bad Block Handling Scenario

In this scenario you are the administrator of a 3B2 computer with a 155–MB disk. On the disk, physical disk block 3 is a surrogate block for physical block X in the **/usr** file system.

Block X is in a text file. Block 3 has become a bad block sometime after the last time the file was read. Now the file needs to be read again. When block X is reached, the driver for the hard disk sees that block X is mapped to block 3 and attempts to read block 3. But block 3 is now bad and cannot be read. When the hard disk driver determines that block 3 is unreadable, the following messages are output to the system console.

```
WARNING: SB00: Cannot access block 3 on slot 1, tc 1, drive 0, err (0x11).

WARNING: unreadable ECC hard disk error: maj/min = 121/0

        block # = 3

Disk Error Daemon: successfully logged error for block 3 on disk maj=121 min=0
```

Note: ECC stands for Error Correction Code, an error checking method.

The attempt to read the text file has failed. As the System Administrator, you notice the message on the console and run **shutdown**(1M) to go to single-user mode. While shutdown is running, the following message is output to the system console.

```
Disk Error Daemon: Disk maj=121 min=0: 1 errors logged
```

The following sections discuss what was happening in the scenario.

Disk Identification

These mechanisms support both single-disk and multidisk models of the 3B2 computer. To avoid confusion and to support all possible configurations, disks are identified by their major/minor device numbers. Messages printed out by bad block handling use the major/minor numbers rather than any other name. The utilities of this feature can be given these names as arguments when more specialized operations must be used.

Detection of New Bad Blocks

The disk driver software detects the new bad blocks for the disk in question. The disk driver determines that a block is not accessible by attempting several accesses. The disk driver also repositions the disk read/write heads between some of the retries to be sure that the problem is not a head positioning error.

Report and Log New Bad Blocks

When a block is determined to be inaccessible, the disk driver tells the bad block logging mechanism about the bad block. The logging mechanism reports the problem on the system console.

The logging mechanism then attempts to remember (write) the error in the disk error log. The disk error log is in the media-specific portion of the disk. If the error is successfully logged, a message similar to the following is output to the system console.

```
Disk Error Daemon: successfully logged error for block 3 on disk
maj=121 min=0
```

Normally, the sequence of events previously described is what happens when the automated mechanisms can handle the identification, reporting, and logging of new bad blocks. Most of the cases of new bad blocks are handled automatically.

The logging mechanism is implemented using a special driver (**hdelog**-Hard Disk Error Log driver) in the operating system and a disk error daemon process [**hdelogger**(1M)] run by the **/etc/init**(1M) process. The special driver

provides access to the reserved disk areas needed by this feature, as well as providing the mechanisms for reporting and queuing up reports until they get logged. The disk error daemon gets reports from this queue and attempts to add each report to the disk error log.

The disk error daemon also has another reporting role. When the system changes its run level (for example, when you turn on the 3B2 computer, shut it off, or shut down to the single-user mode), the daemon checks the error log. If the daemon finds outstanding bad block reports in a log, it outputs a message on the system console. In the previous example, that message was as follows:

```
Disk Error Daemon: Disk maj=121 min=0: 1 errors logged
```

The normal run level for the system is the multiuser mode (run level 2). The **hdelogger** daemon is running in run level 2. The daemon also runs in run levels 3 and 4. Run levels 5 and 6 are used for returning to firmware and for rebooting the operating system, respectively. Run level 1, s, or S is the single-user mode. The bad block handling daemon is not running in run levels 1, 5, or 6.

Unusual Cases and How to Handle Them

Errors in Single-User Mode

If errors happen while the system is in the single-user mode, the errors stay queued until the system is returned to a run level where the bad block handling daemon is running. However, if you shut your system off or reboot it without going to another run level, error reports in the queue are lost. When errors occur while in the single-user mode, only the messages from the logging mechanism and disk driver are output to the system console.

If you get errors while in the single-user mode and you are not ready to fix them for some reason (the mechanism for fixing them takes error reports from the queue as well as from the disk error logs), you can switch to multiuser run level to get them logged. You will get a "successfully logged" message for each error that occurred. When all are logged, you can switch back to the single-user mode. When you do that, a reminder message from the disk error daemon will print on the console.

The Special Case of a Bad Error Log Block

Though unlikely, the new bad block could be the block for the disk error log. Obviously, if you cannot access it, you cannot log that fact in it. But another auxiliary mechanism is provided as part of the **/etc/hdefix** command that adds new bad blocks to the defect map. That command is discussed next.

Fix Bad Blocks

To fix a bad block a quiescent machine is needed. Specifically, the machine must be shut down to the single-user run mode. You must see that all extra processes have died and that only the **root** file system is mounted. After all these conditions are met, run the **/etc/hdefix -a** command. This command checks to see if you are in the single-user run level before proceeding, though it does not check for the other conditions.

If run with just the **-a** option, the **/etc/hdefix** command scans the disk error log. When run this way, the command also looks in the disk error queue for unlogged error reports. If it finds any queued reports, it processes them when it is processing the disk. Thus, if an error is reported while you are in the single-user mode for other reasons, you need not switch run levels to get it processed.

The **hdefix** command updates the map as appropriate (remember that bad surrogate blocks get replaced, not mapped) and removes any reports for the block (there may be more than one) from the disk error log. What the block is used for is also identified. If the block seems to be 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 was processed, the **hdefix** command forces an immediate reboot. The style of reboot that is forced causes the root file system to be checked while coming back up. In addition, any other file system that was marked bad will have to be checked before being mountable.

If you need manually to specify blocks to be fixed, there are additional arguments that can be given to this command for specifying the disk and

block(s) to fix. For instance, in the swap PANIC example, the **hdefix** command could have been used directly when first in the single-user mode. The command line would have been as follows.

```
hdefix -a -D 121 0 -b 463
```

However, when a block number is specified on the command line, **hdefix** ignores the current contents of the error log and the error queue. If there happened to be a report in the log for the block being fixed, the report would still be in the log when you were finished. If the fix list is taken from the log and queue, the log is cleaned up.

Data Loss

Although the useful life of disk hardware has been greatly extended through the bad block handling feature, keep in mind that when a bad block has been logged, any data in the block has been lost. You must be prepared to restore files or file systems that are important to you. Under rare circumstances, you might also need to have your disk reformatted if you lose the Volume Table of Contents (VTOC) block.

The Disk Mirroring Feature

Note: This feature applies only to SCSI hard disk device(s). There is no comparable feature for floppy disks or tapes.

Mirroring SCSI Hard Disks

The AT&T 3B2 SCSI Release 3.0 provides an added high reliability and data integrity feature for SCSI disks called mirroring. Since hard disks are the most vulnerable part of a system, mirroring helps solve any reliability problems by allowing one driver to write to two separate disk sections at once. This is done by having the operating system provide a single virtual mirror partition to the user while internally maintaining two physical disk partitions. If an error occurs while accessing either one of these disk partitions, the operating system can remove the offending disk from service and continue to operate using the other partition. The disk partitions bound to a mirrored partition can be in either the **ACTIVE** or **OUT-OF-DATE** state. If a disk partition is **ACTIVE**, this means that reads and writes to the mirrored partition are also being made to the disk partition. If the disk partition is **OUT-OF-DATE**, this means that the accesses are not being made to the disk partition.

The mirroring feature provides the 3B2 computer with data storage that has a greater reliability than a single disk, and with a greater overall fault tolerance if the **boot** device containing the */(root)* file system is mirrored. When any disk is taken out of service, you can have an exact duplicate of a specific partition, or of the entire disk, already on line and immediately accessible. When the damaged disk is brought back on-line, you can use the **sysadm mirrestore** command to update the **OUT-OF-DATE** disk. The states of the mirrored partitions are maintained even when the system is rebooted.

This section provides the instructions on the following areas:

- Mirroring SCSI disk partitions through **sysadm** commands
- Mirroring SCSI disk partitions through UNIX system level commands
- Mirroring the *root*, *usr*, and *swap* partitions on a SCSI boot device.

The following commands are discussed in this chapter:

Procedure	Command
Display System Administration Mirror Management Menu	**sysadm mirrormgmt**
Partition a SCSI hard disk for mirroring	**sysadm mirpartition**
Configure a processor to support mirrored disk partitions	**sysadm mirsetup**
Verify that two mirrored disk partitions are identical	**sysadm mirverify**
Mirror two disk partitions on a processor	**sysadm mirror** *or* **mirror**(1M)
Copy the contents of a disk partition to another or the mirrored pair	**sysadm mirrestore** *or* **mirrestore**(1M)
Unmirror partitions of a SCSI disk	**sysadm unmirror** *or* **umirror**(1M)
Remove mirroring	**sysadm mirremove**
Display the current mirrored partitions for a SCSI disk	**sysadm mirdisp** *or* **mirror**(1M)
Unmirror the *root*, */usr*, and/or *swap* partitions on a SCSI boot device	**sysadm rootremove**
Mirror the *root*, */usr*, and/or *swap* partitions on a SCSI boot device	**sysadm rootsetup**

Mirroring Components

Actual mirroring is performed by a mirror driver residing in the */boot* directory. The mirror driver interfaces at one end with the user via the normal UNIX operating system calls (**open**, **close**, **read**, **write**, and **ioctl**) and on the other end, with the SCSI Release 3.0 disk driver. The mirror driver is a software module that is responsible for maintaining information about all

mirrored partitions and for accessing the disk partitions that are mirrored. It also provides the **ioctl** function to do such things as restore, verify, mirror, and unmirror disk partitions.

This feature allows the user to bind two normal disk partitions, of equal size, to a mirror special device file name. Thus, each pair of mirrored partitions has one mirror special file. Once the two disk partitions are bound to the mirror special file, their separate disk special files are ignored. So, direct access to the partitions is no longer allowed.

Note: Some commands will not work on mirrored partitions because they require direct access to the device file (**hsbackup**, for instance). In this case, the partition should be unmirrored, backed up, and then remirrored. Other commands can be performed using similar work-arounds.

All writes are performed to both partitions through their mirror special file. Reads are alternated between the two partitions. Read performance of mirrored disks will be equivalent to that for nonmirrored disks. For small configurations, the write performance for a mirrored disk configuration is equivalent to that of a nonmirrored configuration.

Mirroring is supported for two partitions on the same disk (however; if you lose that disk, you have lost both partitions), the same bus, or two different buses on the same machine. If, for instance, you want to create and mount a file system on a currently mirrored disk partition, you would use the mirror special file name for the two partitions. The file system would then be created on both partitions at the same time.

If one of the disks should then fail, the disk could be removed, repaired, and brought back on-line without interrupting access to the currently mounted file system. The following restrictions apply when mirroring disk partitions:

- The disk partitions must be the same size.
- The disk partitions cannot contain mounted file systems.
- The disk partitions cannot contain the Volume Table of Contents (VTOC).

The number of possible mirrored partitions in your system is determined by a tunable parameter called **MPARTS** located in the */etc/master.d/mirror* file. This parameter determines the maximum amount of mirrored partitions.

There are three UNIX system level commands used to set up and maintain mirrored devices: **mirror**, **umirror**, and **mirrestore**. Each of these commands has a System Administration interface under the Mirror Management Menu. Before any of these commands can be used, however, you must make sure that the **MIRROR** driver is included in the */boot* directory. The */etc/system* file should contain the following line:

```
INCLUDE:MIRROR
```

The **mirror** command substitutes one mirror device file for two separate device files on separate disks. These mirror device files are contained in the */dev/dsk* and */dev/rdsk* directories. If you do not use the **sysadm mirsetup** command to create these files, you must create a device file for each mirrored pair. A file named */etc/scsi/mirrortab* contains a table of each currently mirrored device file.

Mirror device files are named consecutively using a **mir***XXX* format. The *XXX* values show the sequence in which the mirror device file was created, beginning with **mir000**. For example, **mir000** is the name of the first mirror device file created; **mir001** is the name of the second, **mir002** is the name of the third, and so on. You can use the **sysadm mirror** command to display a current list of all mirror device files for each processor.

Note: For SCSI boot devices, **mir000**, **mir001**, and **mir002** are reserved for the *root*, */usr*, and *swap* partitions, respectively.

The **umirror** command unbinds one or both special disk files from the mirror special file. Once unbound, the two disk partitions can again be accessed through their special disk file names.

The **mirrestore** command is used to update an **OUT-OF-DATE** partition with the data from an up-to-date partition. Whenever you are setting up two partitions to be mirrored for the first time, you must use the **mirrestore** command to make sure both partitions are in the same state. You must also use the **mirrestore** command to update a disk that has been taken out of service and is brought back on-line.

The following sections discuss the format for each separate command, and their System Administration interfaces.

Using System Administration to Mirror SCSI Disks

The System Administration Mirror Management commands provide a facility for enabling and disabling the mirroring feature.

To reach the Mirror Management Menu, enter the **sysadm mirrormgmt** command or enter **mirrormgmt** from the Disk Management Menu.

The following menu is displayed:

```
            MIRROR DISK PARTITION MANAGEMENT

   1 mirdisp         display all mirrored partitions
   2 mirpartition    partition a hard disk for mirroring
   3 mirremove       remove the mirroring capability
   4 mirrestore      copy a disk partition to its mirrored pair
   5 mirror          mirror two disk partitions
   6 mirsetup        configure the system to support mirroring
   7 mirverify       verify two mirrored partitions are identical
   8 rootremove      unmirror root, /usr, and swap partitions
   9 rootsetup       mirror root, /usr, and swap partitions
  10 unmirror        unmirror one or two disk partitions

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  q to QUIT:
```

Use the Mirror Management commands in the following order when setting up your mirrored devices for the first time:

mirsetup The **sysadm mirsetup** command includes the mirror driver line in the */etc/system*, creates a specified number of mirror device files or a default of 16 in the */dev/dsk* and */dev/rdsk* directories, and specifies the value for the **MPARTS** parameter in the */etc/master.d* mirror file.

mirpartition The **sysadm mirpartition** command interactively configures a SCSI hard disk for mirroring. Once partitioned, the drive can be mirrored by using the **sysadm rootsetup** command or the **sysadm mirror** command. This command does not configure the drive into the system automatically.

mirror The **sysadm mirror** command binds one or two disk special files to one mirror device file. The mirror device file is then added to the */etc/scsi/mirrortab* file. Currently, mounted file systems must be unmounted before the disk partitions can be bound to the mirror special file.

mirrestore The **sysadm mirrestore** command copies the contents of an **ACTIVE** partition to an **OUT-OF-DATE** partition, bringing both partitions into the **ACTIVE** state.

Once you have set up your mirrored devices, the following commands can be used to maintain mirroring:

mirdisp The **sysadm mirdisp** command lists all currently mirrored partitions.

mirremove The **sysadm mirremove** command undoes the setup done by **sysadm mirsetup**, removing the mirroring capability from the system.

mirverify The **sysadm mirverify** command verifies that two disk partitions that are mirrored together are identical.

rootremove The **sysadm rootremove** command performs the steps necessary to remove mirroring of *root*, *usr*, and *swap*. It edits the necessary files (*/etc/system*, */etc/scsi/mirlist*, etc.) to remove the mirror information associated with *root*, *swap*, or */usr*.

rootsetup The **sysadm rootsetup** command performs the steps necessary to support mirroring for *root*, */usr*, and *swap*. It does things like edit the */etc/system* file, edit the */etc/scsi/mirlist* file, etc. The ASCII file, */etc/scsi/mirlist*, lists all disk partitions which should automatically be mirrored when the system comes up. A "sysinit" entry in */etc/inittab* executes the shell script */etc/init.d/mirdisk*. When invoked, the **mirdisk** shell script executes **setmirror** and then mirrors all partitions listed in the */etc/scsi/mirlist* file. When the system goes to multiuser state, the */etc/rc2.d/S80restore* shell script examines all mirrored partitions and starts restores for any mirrored partitions with one disk **ACTIVE** and the other **OUT-OF-DATE**.

unmirror The **sysadm unmirror** command is used to unbind a mirrored partition from a mirror device file. The **sysadm mirdisp** command can be used to access the */etc/scsi/mirrortab* file and display all currently **ACTIVE** mirror device files and the states of their partitions.

The following sections cover each of these commands in more detail.

sysadm mirsetup Command

The **sysadm mirsetup** command is used to initially configure or reconfigure the number of mirror device files for a specific processor. The **sysadm mirsetup** command performs the following functions:

- Includes the following mirror driver line in the */etc/system* file:

  INCLUDE:MIRROR

- Updates the */dev/dsk* and */dev/rdsk* directories to include the new device files. It prompts you for a specific number of mirror device files. If you do not choose a specific number, a default value of 16 mirror device files is used.

- Updates the **MPARTS** parameter residing in the */etc/master.d/mirror* file. The **MPARTS** parameter determines the total number of mirrored device files supported by a processor at one time. This number can be from 1 to 255.

To set up mirroring, follow these steps:

1. Install the 3B2 SCSI Disk Mirroring Utilities Release 1.0 (See instructions for installing packages.) The installation process will invoke the **sysadm mirsetup** command.

2. Enter the number of mirror devices to make. This step creates a specified or default number of mirror device files in the */dev/dsk* and */dev/rdsk* directories and updates the **MPARTS** parameter. Up to 255 mirror device files can be created. Pressing RETURN creates a default 16 mirror device files. On a SCSI boot device, the first three mirror device files (**mir000, mir001,** and **mir002**) are automatically reserved for the *root, /usr,* and *swap* partitions. (See "Mirroring the SCSI Boot Device" covered later in this chapter.)

   Note: Executing **sysadm mirsetup** later on can be used to change the number of mirror device files.

When your selections have been made, **sysadm mirsetup** displays a message indicating that the setup is complete. The following is a sample mirror setup procedure.

```
# sysadm mirsetup

Running subcommand 'mirsetup' from menu 'mirrormgmt',
MIRROR DISK PARTITION MANAGEMENT

The System has already been setup for mirroring and has 20
mirror device files.

Do you want to re-configure the mirror device files? [y, n, ?]  y

The first three mirror device files (mir000, mir001, and mir002)
will be reserved for the sysadm rootsetup command to mirror root (/),
usr (/usr), and the swap (/dev/swap) device.

Enter the number of mirror devices you want to make (default=16) [q, ?]: <CR>

Mirror setup on the System is complete. The System must
be rebooted in order to mirror disk partitions on all of
the 16 mirror devices.

Execute "shutdown -i6 -g0 -y" and
wait for the "Console Login:" prompt.
#
```

sysadm mirpartition Command

The **sysadm mirpartition** command is used to partition a SCSI hard disk to be used for mirroring. The procedure for partitioning a SCSI hard disk drive for mirroring is basically the same procedure as for all hard disk drives. You need to have a complete plan for partitioning a SCSI hard disk for mirroring before you start the actual partitioning procedure. Your plan should include the following:

- Type of partitioning (bootable or nonbootable)
- Number of partitions to be mirrored (up to a maximum of 9 if nonbootable or 14 if bootable)
- Size of each partition to be mirrored (in blocks).

The type of partitioning is important when you are determining which file systems will be mirrored. For example, if you are going to mirror the *root* file system, it is suggested that the *root* file system be mirrored to a bootable device; then, the system can be booted from the mirrored device. User file systems and other system partitions can be allocated on either a bootable or nonbootable device.

The procedure for partitioning a hard disk for mirroring is as follows:

1. Before you partition a disk for mirroring, you will need to look at the "VTOC" of the partitioned hard disk drive you want to mirror. Invoke the **sysadm display** command on the disk drive that contains the *root (/), swap,* and */usr* partitions (normally disk 1).
2. The next step is to partition the hard disk drive by using the **sysadm mirpartition** command.

   Note: As mirror partitions are allocated, make sure you write these names on the summary sheet inside the front cover (Disk Module) or on the rear panel (Expansion Module).

By now, you should know the size of each partition you are mirroring.

The following shows the command line entries and system responses for the **sysadm mirpartition** command. In this example, three partitions of different block sizes are formed to mirror the *root, swap,* and */usr* partitions. The remaining unallocated blocks, if any, will be displayed after all partitions have been selected. Note that partition designations can be skipped by answering "0" to the "How many blocks for disk ... partition ..." questions.

```
# sysadm mirpartition

Running subcommand 'mirpartition' from menu 'mirrormgmt',
MIRROR DISK PARTITION MANAGEMENT

Select which drive to use:
    1 disk1          3 disk3          4 disk4          5 disk5
    2 disk2
Enter a number, a name, the initial part of a name, or
? for HELP, q to QUIT: 3<CR>

Do you want a bootable partition to mirror root?
[ yes no quit help ] help

        Enter "yes" to allocate partitions 0 - 5 and 8 - 15 on disk 3. The TAG
        and FLAG fields in the VTOC will be assigned for partitions 0, 1 and 2.
        Enter "no" to allocate partitions 2 and 8 - 15 on disk 3.
        You may also enter "quit" to quit.

        See the "SCSI Operations Manual" for more information on setting
        up mirrored partitions.

Press RETURN to continue <CR>

Do you want a bootable partition to mirror root?
[ yes no quit help ] yes

There are 276660 blocks remaining on disk 3.

How many blocks for disk 3 partition 1 (swap)?
[ (0 - 276660) again quit help ] (default 276660) 10440

        Note: This command starts all partitions (except 1) on a
        cylinder boundary to improve file system performance.
        Therefore, you may have some unallocated blocks between
        partitions.

How many blocks for disk 3 partition 0 (root)?
[ (0 - 266220) again quit help ] (default 266220) 18360

How many blocks for disk 3 partition 2 (/usr)?
[ (0 - 247860) again quit help ] (default 247860) <CR>

The disk3 drive is now partitioned.
#
```

sysadm mirror Command

The **sysadm mirror** command binds a single disk partition or a pair of disk partitions to a single mirror device file. No more than two device files can be bound to a mirror device file at once. Once the partition or partitions have been bound to the mirror device file, all reads and writes to those partitions must go through the mirror special file.

Note: Some commands will not work on mirrored partitions because they require direct access to the device file (**hsbackup**, for instance). In this case, the partition should be unmirrored, backed up, and then remirrored. Other commands can be performed using similar work-arounds.

The mirror file can then be treated like a regular disk partition. One of two **ACTIVE** disk partitions can also be unbound from the mirror device file for repair and then rebound without affecting a mounted file system on the mirror device file, as long as the remaining partition is in an **ACTIVE** state.

If you are setting up mirrored devices for the first time, make sure you run the **sysadm mirsetup** command BEFORE running **sysadm mirror**. Also, if you currently have file systems mounted on the partitions you want to mirror, you must unmount them before the partitions can be bound to the mirror special file.

To bind a single partition or a pair of partitions to a single mirror device file, follow these steps:

1. Enter the **sysadm mirror** command or enter **mirror** from the Mirror Management Menu.
2. Decide whether to bind one or two partitions to a mirror device file.
3. Select the disk that has the first partition to mirror.
4. Select the partition on the disk to mirror.
5. Enter the disk that has the second partition to mirror.
6. Enter the second partition to bind to the mirror device file.

   The two partitions selected must be the same size. If they are not, an error message is displayed informing you of both partition sizes and indicating that they do not match.

A warning message is also displayed if the mirrored partitions are on the same disk. Mirroring partitions on the same disk reduces both the high reliability and data integrity capabilities of the system. If the disk fails, both partitions become unavailable.

7. Enter the partition that is to be in the up-to-date or **ACTIVE** state, or let the system attempt to determine the **ACTIVE** partition.

The following example shows the procedure.

```
# sysadm mirror

Running subcommand 'mirror' from menu 'mirrormgmt',
MIRROR DISK PARTITION MANAGEMENT

Only two partitions can be mirrored to a mirror device file at once.
The mirror device file is then used for all reads and writes to those
partitions; that is, the mirror device file acts like the disk special
device files.  It can be mounted and treated like a regular disk file
system.  A disk partition can be unmirrored (unbound) from the mirror
device file (as long as its not the only ACTIVE partition), for repair
and then re-mirrored without affecting the users interface to a mounted
file system on the mirror device file.

Enter which function you want to perform.

   1.  mirror two disk partitions to an available mirror device
   2.  mirror a single disk partition to an already mirrored disk partition
[1,2,?,q]:  1

Select which disk on the System has the first partition to mirror.
    1 disk1          3 disk3          4 disk4          5 disk5
    2 disk2
Enter a number, a name, the initial part of a name, or
? for HELP,  q to QUIT: 2
```

Continued

```
Continued from previous screen display

Select which partition on the disk2 drive to mirror.
Possibilities are: 0 1 2 6 7 8 9 a b c [?,q]: c

Select which disk on the System has the second partition to mirror.
    1 disk1          3 disk3          4 disk4          5 disk5
    2 disk2
Enter a number, a name, the initial part of a name, or
? for HELP,  q to QUIT: 5

Select the second partition on the disk5 drive to mirror.
Possibilities are: 0 1 2 6 7 8 9 a b c [?,q]: c

Select which partition to initially be up-to-date (labeled ACTIVE).  By
default, the other partition will be initialized as out-of-date (labeled
OUT_OF_DATE).  The partition labeled ACTIVE contains the data that will
initially be seen through the mirror device.

The third choice will allow the system to determine if a partition is
up-to-date, and will then label the partitions accordingly.  If the system
cannot find an up-to-date partition then you MUST choose one.

Enter which function you want to perform.

   1.  initialize partition  c of disk disk2 to be up-to-date
   2.  initialize partition  c of disk disk5 to be up-to-date
   3.  let the system try to determine the up-to-date partition
   [1,2,3,?,q]:  3
```

Note: The system determines which partition is **ACTIVE** by checking the time stamps that were created by a previous mirror. If no activity has occurred on either partition and the system cannot determine which is the most up-to-date, an error message is displayed along with a prompt asking you to choose a partition.

```
ERROR: The system cannot determine which partition is up-to-date.

Enter which partition to initialize as up-to-date.

   1  partition  c of disk disk2
   2  partition  c of disk disk5
   [1,2,?,q]:  1

The mir003 device has been used to mirror these disk partitions.

Would you like this entry to be mirrored
automatically when the system comes up.
Enter [yes, no, ?]:  yes

This entry has been added to the system mirror list.

Do you want to mirror another disk? [y, n, q] n
#
```

When the **sysadm mirror** command is complete, it is a good idea to use the **sysadm mirdisp** command to display the current table of mirrored partitions contained in the */etc/scsi/mirrortab* file. The table shows the mirror device name and both disk partitions, including their states. At the end of the **sysadm mirdisp** command, one of the disk partitions should be listed as the **ACTIVE** partition, and the other as the **OUT-OF-DATE** partition.

sysadm mirrestore Command

The **sysadm mirrestore** command brings an **OUT-OF-DATE** partition to the **ACTIVE** state by copying the contents of the **ACTIVE** partition to the **OUT-OF-DATE** partition. The **sysadm mirrestore** command is an uncomplicated command, but it must be performed in order to bring both partitions to the **ACTIVE** state.

To bring an **OUT-OF-DATE** partition to the **ACTIVE** state, follow these steps:

1. Enter the **sysadm mirrestore** command or enter **mirrestore** from the Mirror Management Menu.
2. Select the mirror device file to restore.

   The **mirrestore** command then determines which partition is **ACTIVE** and copies its contents to the **OUT-OF-DATE** partition. The following example shows the procedure.

```
# sysadm mirrestore

Running subcommand 'mirrestore' from menu 'mirrormgmt',
MIRROR DISK PARTITION MANAGEMENT

Select the mirror device file to restore.
    1 mir001          2 mir003
Enter a number, a name, the initial part of a name, or
? for HELP,  q to QUIT:  2

Restoring the mir003 device file is in progress.

mir003 has been restored.

Do you want to restore another device file? [y, n, q]  n

#
```

The **mirdisp** command should now list both partitions as **ACTIVE**.

sysadm mirverify Command

The **sysadm mirverify** command is used to compare the data on two mirrored disk partitions and verify that they are identical.

To verify that disk partitions are identical, follow these steps:

1. Enter the **sysadm mirverify** command or select option **6** from the Mirror Management Menu.
2. Select the mirror device file to verify.

   The **mirverify** command then verifies that both mirrored disk partitions are identical. If they are not, the command will list the disk blocks that are different. The following example shows the procedure:

```
# sysadm mirverify

Running subcommand 'mirverify' from menu 'mirrormgmt',
MIRROR DISK PARTITION MANAGEMENT


Select the mirror device file to verify.
        1 mir003        2 mir004
Enter a number, a name, the initial part of a name, or
? for HELP,  q to QUIT: 2

Verification of the mir004 device file is in progress.

The verification of mir004 is complete.

Do you want to verify another device file? [y, n, q]  n
#
```

sysadm mirdisp Command

The **sysadm mirdisp** command displays the contents of the */etc/scsi/mirrortab* file, including all currently **ACTIVE** and **OUT-OF-DATE** partitions for each mirror device file. Note that you may have created many mirror device files, but have only initialized a few through the **sysadm mirror** command. Only those mirror device files that have disk partitions bound to them are in the */etc/scsi/mirrortab* file.

To display the contents of the */etc/scsi/mirrortab* file, do the following:

- Enter the **sysadm mirdisp** command or enter **mirdisp** on the Mirror Management Menu.

The following example shows the procedure and table:

```
# sysadm mirdisp

Running subcommand 'mirdisp' from menu 'mirrormgmt',
MIRROR DISK PARTITION MANAGEMENT


MIRROR DEVICE    DISK PART #1 (STATE)    DISK PART #2 (STATE)
=============================================================

mir002           c1t4d0s1 (ACTIVE)       c2t3d1s1 (ACTIVE)
mir000           c1t4d0s0 (ACTIVE)       c2t3d1s0 (ACTIVE)
mir001           c1t4d0s2 (ACTIVE)       c2t3d1s2 (ACTIVE)
mir003           c2t3d0sc (ACTIVE)       none (-)

#
```

sysadm unmirror Command

The **sysadm unmirror** command unbinds a specified partition from its mirror special file, and removes its entry from the */etc/scsi/mirrortab* file. Once the partition has been unbound, it can only be accessed through its disk special file.

Note: It is suggested to only access an unmirrored partition if that partition will not be remirrored in the future. If the disk partition is unmirrored, then any accesses to it are not known by the mirror driver. If the partition is remirrored, the mirror driver will consider it to be in the same state as when it was unmirrored.

To unmirror a partition, follow these steps:

1. Enter the **sysadm unmirror** command or enter **umirror** from the Mirror Management Menu.
2. Select the mirror device file binding the disk partition(s).
3. Select the partition you would like unbound.

The **sysadm unmirror** command then unbinds the partition from the mirror special file. If no disk partitions are left bound to the mirror special file, **sysadm unmirror** then removes its entry from the */etc/scsi/mirrortab* file. When these functions have been completed, a message is displayed.

This command can be used to unbind one or both disk partitions from all mirrored partitions, except *root*, */usr*, and *swap* (mir000, mir001, and mir002). This command can be used on *root*, */usr*, or *swap*, only to remove one of the two disk partitions if the other disk partition is ACTIVE. To totally unmirror *root*, */usr*, and *swap*, use **sysadm rootremove**.

The following example shows the command line entries and system responses for the **unmirror** procedure:

```
# sysadm unmirror

Running subcommand 'unmirror' from menu 'mirrormgmt',
MIRROR DISK PARTITION MANAGEMENT


Select the mirror device which binds the disk partitions to unmirror.
    1 mir000          2 mir002          3 mir001          4 mir003
Enter a number, ? for HELP, or  q to QUIT:  4

The mir003 device file is currently binding two disk partitions.

Enter which partition to unmirror:

   1. partition a of the disk5 drive
   2. partition a of the disk3 drive
   3. both partitions
[1,2,3,?,q]:   3

The unmirroring of mir003 completed successfully.

Enter which entry to substitute for the mirror entry in
/etc/fstab:

   1. partition a of the disk5 drive
   2. partition a of the disk3 drive
[1,2,?,q]:   1

Would you like to remove this entry from the system mirror list.
Enter [yes, no, ?]:  y

The mir003 entry has been removed from the system mirror list.

Do you want to unmirror another disk? [y, n, q] n
#
```

To verify that the mirror partition was removed, the **sysadm mirdisp** command can be used. A **sysadm mirdisp** table for the previous example would not have a **mir003** entry listed in the output, but would still have **mir000, mir001**, and **mir002**.

sysadm mirremove Command

The **sysadm mirremove** command disables mirroring by removing the **MIRROR** driver and setting the **MPART** tunable parameter. Before you can use **sysadm mirremove**, you must first unmirror any mirrored partitions, except for the **root**, **usr**, and **swap** partitions.

To remove mirroring, follow these steps:

- Enter the **sysadm mirremove** command or enter **mirremove** from the Mirror Management Menu.

The following example shows the procedure:

```
# sysadm mirremove

Running subcommand 'mirremove' from menu 'mirrormgmt',
MIRROR DISK PARTITION MANAGEMENT

The root, usr and/or swap partitions are mirrored on the system.
Do you want to continue with this request? [y, n, q] y

The system must now be rebooted to remove
the mirror capability from the system.

Execute shutdown -i6 -g0 -y and
wait for the Console Login: prompt.
#
```

Caution: **Do not remove IPC package while the 3B2 SCSI Mirroring package is installed.**

Note: Prior to removing 3B2 SCSI Mirroring Utilities, **sysadm mirremove** must be executed.

Mirroring the SCSI Root Device sysadm rootsetup Command

Mirroring the system boot device significantly increases the reliability of your system. If one disk containing the *root*, */usr*, and *swap* partitions becomes disabled, the system can be supported from the second device without interruption.

Mirroring the root device can either be performed manually, or through the **sysadm rootsetup** command under the Mirror Management Menu. Because mirroring the root device is a complicated procedure, it is highly recommended that you use the **sysadm rootsetup** command. It is also recommended that you mirror *root*, */usr*, and *swap* since this command automatically reserves mirror partitions **mir000**, **mir001**, and **mir002** for the *root*, */usr*, and *swap* partitions.

The **sysadm rootsetup** command provides an easy method of performing the complicated steps involved in mirroring the root device. To mirror a root device using System Administration, perform the following steps:

- If not previously installed, install the mirroring package to automatically invoke **sysadm mirsetup** or enter the **sysadm mirsetup** command.

  This command creates the necessary mirror device files in the */dev/dsk* and */dev/rdsk* directories, and includes the **MIRROR** driver in the */etc/system* file (**mirsetup** coverage can be found earlier in this chapter).

- Enter the **sysadm rootsetup** command.
- Press [RETURN] for the default of mirroring *root*, */usr*, and *swap*.
- Select the disk that has the second partition to mirror.
- Select the second partition to mirror.

  The same series of prompts is provided for each of the partitions.

- When you have completed **sysadm rootsetup**, reboot the 3B2 computer from the*/etc/system* file.

  This rebuilds the */unix* file from the modified */etc/system* file, and mounts the *root* and */usr* file systems by their mirror device file specifications.

The following example shows the command line entries and system responses for mirroring a root device.

```
# sysadm rootsetup

Running subcommand 'rootsetup' from menu 'mirrormgmt',
MIRROR DISK PARTITION MANAGEMENT

Enter the file systems that are to be mirrored.
Possibilities are: root, usr, swap, all [(default all) ?, q]:  all

Interrupt signals will be ignored.

The setup for mirroring root is in progress.

It is strongly recommended that the mirrored root partition
be on a bootable drive (partition 0). Then in the event the
primary root partition fails, the system can be rebooted from
the mirrored backup drive. A bootable drive must be disk 0 on
a target controller. Use the sysadm mirpartition command to
create the bootable drive.

The root partition is on the disk3 drive partition 0.

Select the second disk on which you want to mirror root:
    1 disk3          2 disk4          3 disk5          4 disk6
Enter a number, a name, the initial part of a name, or
? for HELP,  q to QUIT: 3<CR>

Select the second partition on the disk5 drive to mirror root.
Possibilities are: 0 1 2 8 9 a b c [?, q]: 0
```

Continued

Continued from previous screen display

```
The setup for mirroring usr is in progress.

The usr partition is on the disk3 drive partition 2.

Select the second disk on which you want to mirror usr:
    1 disk3          2 disk4          3 disk5          4 disk6
Enter a number, a name, the initial part of a name, or
? for HELP,  q to QUIT: 3<CR>
Select the second partition on the disk5 drive to mirror usr.
Possibilities are: 0 1 2 8 9 a b c [?, q]: 2<CR>

The setup for mirroring swap is in progress.

The swap device(s) is/are:
        disk3 drive partition 1

Select the first disk you want to contain the swap partition:
    1 disk3          2 disk4          3 disk5          4 disk6
Enter a number, a name, the initial part of a name, or
? for HELP,  q to QUIT:  1

Select the partition on the disk3 drive used for swap.
Possibilities are: 0 1 2 8 9 a b c [?, q]:  1

Select the second disk on which you want to mirror swap:
    1 disk3          2 disk4          3 disk5          4 disk6
Enter a number, a name, the initial part of a name, or
? for HELP,  q to QUIT:  3

Select the second partition on the disk5 drive to mirror swap.
Possibilities are: 0 1 2 8 9 a b c [?, q]: 1

/etc/system has been modified and should be used during a
reboot of the system.

The setup for mirroring root, usr, and swap partitions is
now complete.
In order for the mirroring to become effective,
you must reboot the system now.

Execute shutdown -i6 -g0 -y and
wait for the Console Login: prompt.
#
```

sysadm rootremove Command

The **sysadm rootremove** command is used to unmirror the *root, /usr,* and/or *swap* partitions. Mirroring the root device is a complicated procedure, but root mirroring can be done easily through the **sysadm rootsetup** command. The **rootremove** command works backward through the complicated procedure to "undo" the **rootsetup** command.

If it is desired to remove mirroring from the system, this command does not need to be executed. The **sysadm mirremove** command can be executed without unmirroring the *root, /usr,* and/or *swap* partitions.

After the **rootremove** process is complete, you must reboot the system. This will complete the unmirroring of the root device.

The following shows the command line entries and system responses associated with the **sysadm rootremove** command.

```
# sysadm rootremove

Running subcommand 'rootremove' from menu 'mirrormgmt',
MIRROR DISK PARTITION MANAGEMENT

Enter the file systems that are to be unmirrored.
Possibilities are: root, usr, swap, all [(default all) ?,q]: all

Interrupt signals will be ignored.

Unmirroring of the root partition is in progress.

Unmirroring of the usr partition is in progress.

Unmirroring of the swap partition is in progress.

The unmirroring of root, usr, and swap partitions is
now complete.  In order for the unmirroring to become
effective, you must reboot the system now.

Execute shutdown -i6 -g0 -y and
wait for the Console Login: prompt.

#
```

Using UNIX System Commands to Mirror SCSI Disks

The following sections describe the UNIX system commands used to mirror SCSI disks. These commands are located in the */etc/scsi* directory. In order to execute these commands, you must either place */etc/scsi* into your search path (PATH variable in the *.profile*) or enter the full path name for each command. The examples in this manual assume that */etc/scsi* has been added to the login search path.

mirror Command

The UNIX system **mirror**(1M) command substitutes two separate disk device files for one mirror device file, and inserts an entry in the */etc/scsi/mirrortab* file for the mirror device. The **mirror** command only links the two special device files to a mirror special file. In order to make both partitions **ACTIVE**, you must use either the **mirrestore** or the **sysadm**

mirrestore command to copy the contents of the **ACTIVE** partition to the **OUT-OF-DATE** partition.

The **mirror** command is entered in the following format:

mirror *special1 special2 special3*

Where:

special1 The mirror special file. This is the special file used to identify the two mirrored partitions.

special2 The disk special device file for the first partition to be mirrored. When used with the **-f** option, this partition will be made the up-to-date or **ACTIVE** partition.

special3 The disk special device file for the second partition to be mirrored. When used with the **-f** option, this partition will be made the **OUT-OF-DATE** partition.

For example, the following command line binds partitions **c0t6d0s1** and **c0t7d0s8** to the **mir005** mirror device file:

```
$ mirror /dev/rdsk/mir005 /dev/rdsk/c0t6d0s1 /dev/rdsk/c0t7d0s8
```

This next example binds partitions **c0t7d0sa** and **c0t6d0s8** to the **mir010** mirror device file. The **-f** is included to force the command to use the first partition special device file as the **ACTIVE** partition and the second partition special device file as the **OUT-OF-DATE** partition:

```
$ mirror -f /dev/rdsk/mir010 /dev/rdsk/c0t7d0sa /dev/rdsk/c0t6d0s8
```

The **mirror** command determines which partition is up to date by checking the time stamps on both partitions and determining which partition was most recently accessed. Time stamps are updated each time a special file is closed. In the event of a system crash, the time stamps for currently open files will not be updated. In this case, the mirror command will not be able to find an up-to-date partition and will fail. You must then force *special2* into the **UP-TO-DATE** condition by using the **-f** option. If you use the **-f** option, the position of the special files on the command line becomes important. The **-f** option forces *special2* into the **ACTIVE** state and *special3* into the **OUT-OF-DATE** state.

You can use the **mirror**(1M) command to bind just one disk special device file to a mirror special file. If a mirror special file is bound to just one disk special device file and you want to bind it to the second disk special device file, you must use both disk special device files as arguments in the command.

Entering the **mirror** command without specifying any special file names displays a list of all the mirrored disk partitions contained in the */etc/scsi/mirrortab* file. The */etc/scsi/mirrortab* file consists of a table of all currently mirrored partitions and their states. Entering the **mirror** command with one currently active mirror device file displays the partitions associated with that mirror file.

Here is an example of the **mirror**(1M) without arguments.

```
# mirror<CR>

Entry Date:  Mon May 16 15:51:52 1988

Mirror partition : /dev/rdsk/mir005
Partition Size   : 90396  blocks
Disk Partition 1 : /dev/rdsk/c0t6d0s1  ACTIVE
Disk Partition 2 : /dev/rdsk/c0t7d0s8  OUT_OF_DATE

*********

Entry Date:  Mon May 16 15:51:55 1988

Mirror partition : /dev/rdsk/mir010
Partition Size   : 90396  blocks
Disk Partition 1 : /dev/rdsk/c0t7d0sa  ACTIVE
Disk Partition 2 : /dev/rdsk/c0t6d0s8  OUT_OF_DATE
#
```

The **mirror** command binds two special disk files to one mirror special file, but it may only make one active. To make them both active use the **mirrestore** command, which is discussed in the next section.

mirrestore Command

The **mirrestore**(1M) command copies the contents of the **ACTIVE** partition to the **OUT-OF-DATE** partition and sets both flags in the mirror driver to an **ACTIVE** state. This command must be used whenever partitions are mirrored for the first time through the **mirror**(1M) command to make the mirrored partitions active. This needs to be done whether or not the up-to-date partition has any data, as it sets the state-of-the-mirror driver flags.

The **mirrestore** command is entered in the following format:

mirrestore *special1*

Where:

special1 The mirror special file.

The **mirrestore** command checks the mirror driver to determine which of the mirrored pair is **OUT-OF-DATE**. When it has determined the **OUT-OF-DATE** partition, **mirrestore** sends a command to the mirror driver to copy the contents of the **ACTIVE** partition to the **OUT-OF-DATE** partition.

The following is a typical example of a **mirrestore** command line. A mirror special file (**mir004**) is first bound to two disk partitions through the **mirror** command. The two disk partitions are then brought to the **ACTIVE** state through the **mirrestore** command.

```
$ mirror /dev/rdsk/mir004 /dev/rdsk/c0t7d0s8 /dev/rdsk/c0t6d0s8
$ mirrestore /dev/rdsk/mir004
```

umirror Command

The **umirror**(1M) command unbinds the mirrored partitions from the mirror special file. You can unbind both partitions, or you can unbind one, provided the other partition is **ACTIVE**. The **umirror** command is entered in the following format:

umirror *special1 special2*

Where:

special1 The mirror special file. If only the mirror special file name is specified, the disk special files bound to that mirror special file are unmirrored and the mirror special file entry in the */etc/scsi/mirrortab* file is removed. After the command completes, the disk partitions can only be accessed through the disk special file name and not the mirror special file name.

special2 A disk special file for a specific partition. If a specific disk special file name is given, only that partition will be unmirrored. The second partition is still bound to the mirror special file and the mirror special file remains in the */etc/scsi/mirrortab* file. However, the **umirror** command will not unbind the last **ACTIVE** partition of a mirrored pair.

There may be times when you would like to unmirror one disk partition from a mirror special file in order to service the disk, then rebind that partition to the same mirror special file when the disk is back in service. As long as the remaining partition is in the **ACTIVE** state, this can be done simply by specifying the disk special file as the argument for the **umirror** command:

```
$ umirror /dev/rdsk/mir004 /dev/rdsk/c0t6d0s8
```

Note: It is suggested to only access an unmirrored partition if that partition will not be remirrored in the future. If the disk partition is unmirrored, then any accesses to it are not known by the mirror driver. If the partition is remirrored, the mirror driver will consider it to be in the same state as when it was unmirrored.

If the partition you want to mirror is the only **ACTIVE** partition, you must first make the other partition **ACTIVE**. For example, the following portion of a **sysadm mirdisp** command output shows partition **c0t6d0s8** to be an **OUT-OF-DATE** partition and **c0t7d0s8** to be an **ACTIVE** partition:

```
MIRROR DEVICE    DISK PART#1 (STATE)    DISK PART #2 (STATE)
=============================================================

mir003           c0t1d0sa (ACTIVE)      c0t1d1s8 (ACTIVE)
mir004           c0t1d0s8 (ACTIVE)      none (-)
mir005           c0t7d0s8 (ACTIVE)      c0t6d0s8 (OUT-OF-DATE)
```

In order to unbind **c0t7d0s8** from the mirror special file, you must first run the **mirrestore** command to bring both partitions up to date so as to leave one **ACTIVE** partition bound to the mirror device file when the **umirror** command is executed:

```
$ mirrestore /dev/rdsk/mir005
$ umirror /dev/rdsk/mir005 /dev/rdsk/c0t7d0s8
```

Similarly, if the mirror special file has only one partition currently mirrored to it, you must first use the **mirror** command to bind another partition to the mirror special file, then use the **mirrestore** restore command to bring the new mirrored partition to the **ACTIVE** state.

Other UNIX System Mirroring Commands

If the mirror table (*/etc/scsi/mirrortab*) becomes out of sync with the mirror driver (looses an entry for a currently mirrored partition), you can use the **setmirror**(1M) command to insert an entry into the mirror table.

The **mverify**(1M) command can then be used to verify that the data on two mirrored partitions is identical.

Mirroring the SCSI Boot Device Manually

Mirroring the system boot device significantly increases the reliability of your system. If one disk containing the *root*, */usr*, and *swap* partitions becomes disabled, the system can be supported from the second device without interruption.

Mirroring the root device can either be performed manually or through the **sysadm rootsetup** command under the Mirror Management Menu. Because mirroring the root device is a complicated procedure, it is highly recommended that you use the **sysadm rootsetup** command. It is also recommended that you mirror *root*, */usr*, and *swap* since this command automatically reserves mirror partitions **mir000**, **mir001**, and **mir002** for the *root*, */usr*, and *swap* partitions. However, the manual procedure is as follows:

Steps for Mirroring *root*

Mirroring the *root* disk partition requires that the */etc/system* file be edited to include a new device specification called **MIRRORDEV** and to change the existing **ROOTDEV** and **PIPEDEV** entries (the */etc/system* file includes **ROOTDEV** and **PIPEDEV** entries, but comments them out). Normally, the location of the *root* file system is determined from the VTOC on the SCSI boot disk, and not from the */etc/system* file. Adding new **MIRRORDEV** and **ROOTDEV** entries to the */etc/system* file overrides the original *root* partition specification in the VTOC and specifies the mirror device file on which the *root* file system is mounted. You do not need to change the boot disk VTOC.

Mirroring the *root* partition involves the following steps. The **sysadm rootsetup** command performs these steps automatically. If you are mirroring the *root* partition manually, complete the following steps using disk partitions appropriate for your system:

1. Edit the */etc/system* file to include the **MIRRORDEV** device specification.

   This device specification includes only the two partition device files in the following format:

   MIRRORDEV:/dev/rdsk/c1t4d0s0 /dev/rdsk/c2t3d1s0

2. Modify the **ROOTDEV** and **PIPEDEV** entries in the */etc/system* file to reflect the mirror device file.

   This step redirects the root and pipe devices to the mirror device file instead of a specific disk device file. For example, after the above two steps, the */etc/system* file should be modified to include entries like the following:

   MIRRORDEV:/dev/rdsk/c1t4d0s0 /dev/rdsk/c2t3d1s0
   ROOTDEV:/dev/dsk/mir000
   PIPEDEV:/dev/dsk/mir000

3. Ensure the first partition used to mirror *root* is in the **ACTIVE** state, and that the second partition is in the **OUT-OF-DATE** state.

   The mirror device driver determines which partition is in the **ACTIVE** state by reading the time stamp on the partition's device file. The one that has received the most current activity is the one determined to be **ACTIVE**.

You can manually ensure that the second partition is determined to be **OUT-OF-DATE** by performing the following set of commands:

```
# prtvtoc /dev/rdsk/c2t3d1s0 > vtoc
# fmthard -s vtoc /dev/rdsk/c2t3d1s0
# rm vtoc
```

The above commands first redirect the contents of the second disk's VTOC

into a file called **vtoc**. The disk's VTOC is then recreated using the same information. Rewriting the VTOC clears the time stamps on all partitions. With the second partition's time stamp set to 0, the mirror driver will automatically determine it to be **OUT-OF-DATE**.

The **/etc/fmthard** command is used to zero out the time stamp on disk partitions. The mirror feature uses the time stamp information to determine which disk partition in a mirrored pair should be made **ACTIVE**. If one time stamp is zero and the other is nonzero, the nonzero one will be made **ACTIVE** (unless the force option is used).

Steps for Mirroring */usr*

The next step is to mirror the */usr* disk partition.

To mirror the */usr* disk partition, perform the following steps:

1. Manually edit the */etc/scsi/mirlist* file to include the */usr* mirror device file and its corresponding partition device files.

   This will ensure that the */usr* disk partition will be mirrored whenever the system is rebooted.

2. Change **fstab** so it mounts */dev/dsk/mir001* as */usr*.
3. Ensure the first partition used to mirror */usr* is in the **ACTIVE** state, and that the second partition is in the **OUT-OF-DATE** state.

You can use the same procedure to force the second partition into the **OUT-OF-DATE** state that you used with the *root* mirror device:

```
# prtvtoc /dev/rdsk/c1t4d0s2 > vtoc
# fmthard -s vtoc /dev/rdsk/c2t3d1s2
# rm vtoc
```

As with the *root* procedure, rewriting the VTOC on the second partition clears the partition's time stamp. With the time stamp set to 0, the mirror driver will automatically establish the second partition as **OUT-OF-DATE**.

Steps for Mirroring *swap*

The final step in mirroring the entire boot device is to establish a mirror device file as the boot disk *swap* device. The original location of the *swap* device is determined by the VTOC on the boot disk. The default VTOC for a SCSI boot device reserves partition 1 as the *swap* device.

The following procedure will create a mirror device file for *swap* and set up your system to insert the mirror device file as the *swap* device specification each time you boot.

1. Choose the two disk partitions you want to bind to the mirrored *swap* partition.
2. Add to the file */etc/scsi/mirlist* the following:

   /dev/rdsk/mir002 disk part #1 disk part #2

3. Add to the file */etc/system* the following directive:

   SWAPDEV:/dev/dsk/mir002 0 size_of_swap_device

The Reboot Step of Mirroring a SCSI Boot Device

When the setup has been completed to mirror any combination of *root*, */usr*, and *swap*, the system needs to be rebooted. When the system comes up, these partitions will automatically be mirrored. The system will also make sure they are automatically put in the **ACTIVE-ACTIVE** state.

For example, if all three partitions (*root*, */usr*, and *swap*) were set up for mirroring, after the system is rebooted, the **sysadm mirdisp** command would display the following:

```
# sysadm mirdisp

Running subcommand 'mirdisp' from menu 'mirrormgmt',
MIRROR DISK PARTITION MANAGEMENT

MIRROR DEVICE    DISK PARTITION #1 (STATE)    DISK PARTITION #2 (STATE)
=========================================================================

mir000           c1t4d0s0 (ACTIVE)            c2t3d1s0 (ACTIVE)
mir001           c1t4d0s2 (ACTIVE)            c2t3d1s2 (ACTIVE)
mir002           c1t4d0s1 (ACTIVE)            c2t3d1s1 (ACTIVE)
```

Maintenance for Mirrored Disks

When you want to perform service or maintenance on disks that are mirrored, you can work only on one disk at a time. There must be a disk in the **ACTIVE** state at all times. For example, the following shows the sequence of procedures that would have to occur for performing maintenance on two disks that have been mirrored and are currently **ACTIVE**.

Disk 1 Procedure	Disk 1 Partition	Disk 2 Partition
Beginning state	ACTIVE	ACTIVE
Unmirror Disk 1 Partition	UNMIRRORED	ACTIVE
Perform maintenance on Disk 1	DISABLED	ACTIVE
Mirror Disk 1 Partition to Disk 2 Partition	OUT-OF-DATE	ACTIVE
Restore the Partitions	ACTIVE	ACTIVE

Disk 2 Procedure	Disk 1 Partition	Disk 2 Partition
Beginning state	ACTIVE	ACTIVE
Unmirror Disk 2 Partition	ACTIVE	UNMIRRORED
Perform maintenance on Disk 2	ACTIVE	DISABLED
Mirror Disk 2 Partition to Disk 1 Partition	ACTIVE	OUT-OF-DATE
Restore the Partitions	ACTIVE	ACTIVE

Chapter 5: File System Administration

Introduction

How the 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 one-dimensional array 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; the operating system itself retains the responsibility for writing directories. The combination of directories and files make up a file system. Figure 5-1 shows the relationship between directories and files in a UNIX system file system. The circles represent directories.

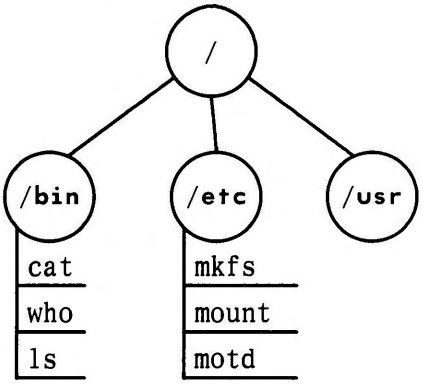


Figure 5-1: A UNIX System File System

The starting point of any UNIX system file system is a directory that serves as the root. In the UNIX operating system there is always one file system that is referred to by that name, **root**. Traditionally, the root directory of the **root** file system is represented by a single slash (/). The file system diagrammed in Figure 5-1, then, is a **root** file system. If we graft another file system onto **root** at a directory called, **usr**, the result can be illustrated by the diagram in Figure 5-2.

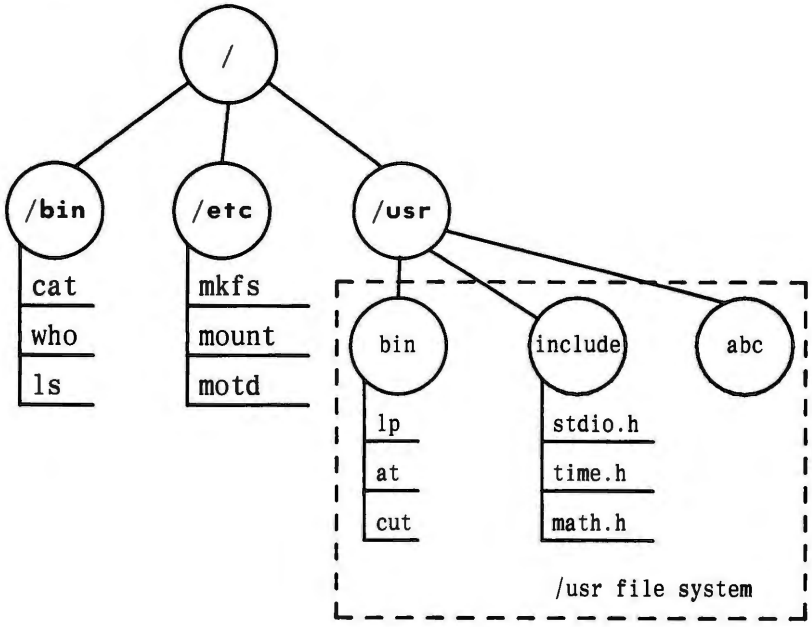


Figure 5-2: Adding the **/usr** File System

A directory such as **usr** is referred to in various ways. You sometimes see the terms "leaf" and "mount point" used to describe a directory that is used to form the connection between the **root** file system and another mountable file system. Regardless of the terms used, such a directory is the root of the file system that descends from it. The name of that file system is, coincidentally, the name of the directory. In our example, the file system is **usr**.

The diagrams in Figures 5-1 and 5-2 may be a convenient representation of the file and directory structure of file systems, but not a particularly accurate, or helpful, way of illustrating how a file system is known to the UNIX operating system.

The operating system views a file system as an arrangement of addressable blocks of disk space that can be classified in four categories:

- Block 0
- Block 1: the super-block
- A variable number of blocks comprising the i-list
- A variable number of storage blocks: most contain data, some contain the freelist and indirect addresses

This scheme is illustrated in Figure 5-3.

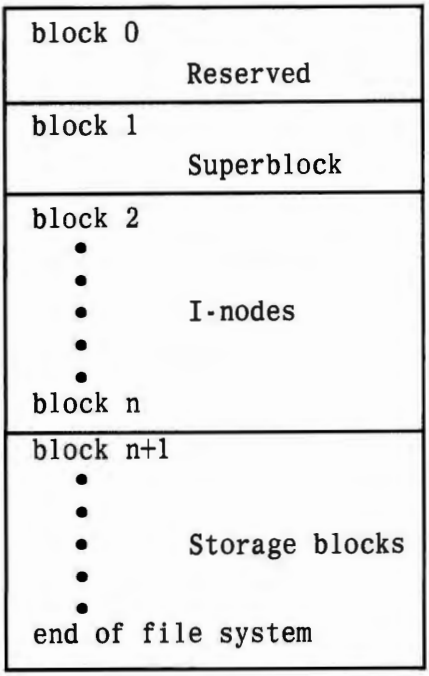


Figure 5-3: The UNIX System View of a File System

Block 0

Block 0, although considered to be part of the file system, is actually not used by it. It is reserved for storing the boot procedures. Not all file systems are involved in booting. For a file system that is not involved in booting, block 0 is left unused.

Block 1: The 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 physical and logical blocks
  - Read-only flag
  - Super-block modified flag
  - Date and time of last update
- I-nodes
  - Total number of i-nodes allocated
  - Number of free i-nodes
  - Array containing 100 free i-node numbers
  - An index into the free i-node number array
- Storage blocks
  - Total number of free blocks
  - Array containing 50 free-block numbers
  - An index into the free-block number array.

Note that the super-block does not maintain complete lists of free i-nodes 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 i-nodes and storage blocks, there is sure to be more free i-nodes and blocks than are listed in the super-block. The information about them is kept in one storage block.

I-Nodes

The term "i-node" stands for information node or index node. (You will often see it spelled with no hyphen: inode.) The same formulation is used in other references to things associated with i-nodes. For example, the list of i-nodes is referred to as the i-list (or ilist); an i-number is the position of an i-node in the i-list.

The i-node contains all the information about a file except for its name, which is kept in a directory. An i-node is 64 bytes long, so there are 8 i-nodes to a physical block. There is no set number of blocks occupied by the i-node list; it depends on how many i-nodes are specified at the time the file system is created. An i-node contains:

- The type and mode of file: type is regular (-), directory (d), block (b), character (c), or FIFO, also known as named pipe, (p); mode is the set of read-write-execute permissions
- The number of links to the file
- The owner's user-id number
- The group-id number to which the file belongs
- The number of bytes in the file
- An array of 13 disk block addresses
- The date and time last accessed
- The date and time last modified
- The date and time created.

The array of 13 disk block addresses is the heart of the i-node. The first 10 are direct addresses; that is, they point directly to the first 10 storage blocks of the contents of the file. If the file is larger than 10240 characters, the 11th address points to an indirect block that contains 256 more block addresses; the 12th address points to a double indirect block that contains the addresses of another 256 indirect blocks, each of which contains the addresses of 256 storage blocks. If the file is larger than 67,381,248 bytes, the 13th address in the array points to the address of a triple indirect block that contains the addresses of 256 double indirect blocks, and so on. The theoretical maximum size of a UNIX system file is well beyond the limits of the amount of disk storage on your 3B2 computer. Figure 5-4 illustrates this chaining of address blocks stemming from the i-node.

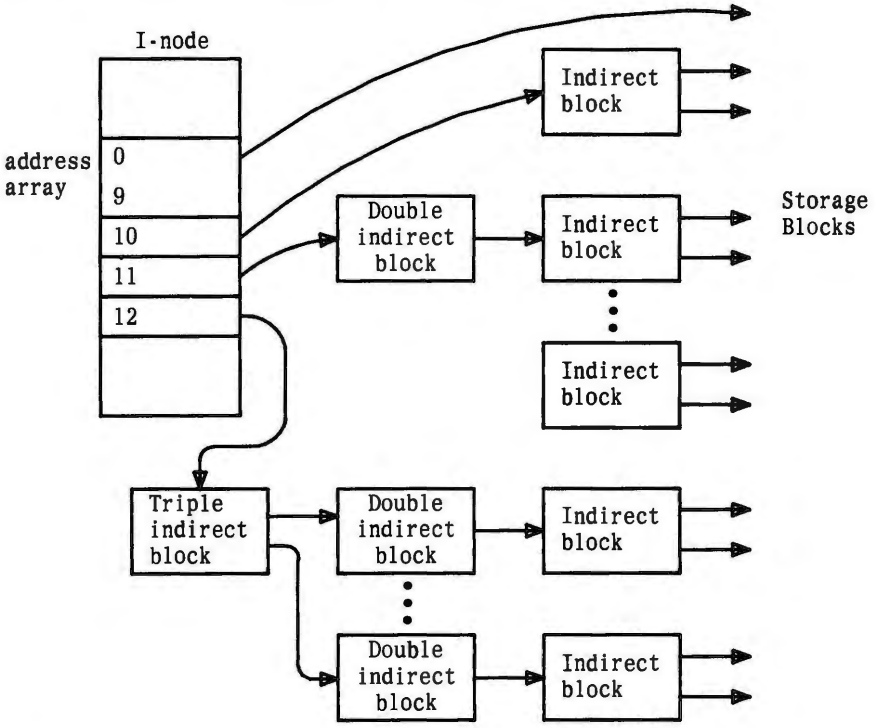


Figure 5-4: The File System Address Chain

The following table shows the number of bytes addressable by the different levels of indirection in the i-node address array. These numbers are calculated using the logical block size of the file system and the number of bytes used to hold an address (4).

Logical Block Size	MAXIMUM NUMBER OF BYTES ADDRESSABLE BY			
	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 a UNIX system file system 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 i-node. This is a 32-bit field, so file sizes are limited to 4 gigabytes. Moreover, the theoretical maximum sizes shown in the table are beyond the limits of the disk storage on the 3B2 computer.

Storage Blocks

The remainder of the space allocated to the file system is taken up by storage blocks, also called data blocks. For a regular file, the storage blocks contain the contents of the file. The contents are undefined. For a directory, the storage blocks contain 16-byte entries (64 to a block). 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 file name of the member file or subdirectory.

Free Blocks

Blocks not currently being used as i-nodes, as indirect address blocks, or as a storage block are chained together in a linked list. Each block in the list carries the address of the next block in the chain.

Summary

What we have described thus far is an abstract view of a UNIX system file system, the components of a file system, and the way they relate to each other. In later sections of this chapter we will see how file systems are stored on disks and what happens to a file system when it is in use.

The Relationship Between the File System and the Storage Device

In the UNIX system, file systems reside on random-access disk devices. (You can back up a file system on tape that is for backup security rather than live access.) Before you can install a file system on a disk there are some preliminaries that must be taken care of. The material in this part of the chapter is a summary of material described in more detail in Chapter 4, "Disk/Tape Management."

Disk Format

Before a disk can be used by the UNIX system, the disk 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.

Note: Hard disk units on the 3B2 computer are formatted when installed. The only time they might need to be reformatted would be after a catastrophic hardware failure. We recommend that you contact your AT&T Service Representative or authorized dealer if such an event occurs.

The 3B2 computer's dual disk system has **usr** on the second disk while **root** and **usr2** share the first. The 3B2 computer's single disk system has **usr** and **root** on the same disk. Hard disks are partitioned using the **sysadm partitioning** command under the Disk Management Menu. Partitions do not have to be the same size. The total number of blocks assigned (all assigned partitions) must equal the number of blocks available.

Floppy disks are made to be usable in more than one machine. Manufacturers produce floppy disks unformatted, leaving it to customers to format them for the particular machine on which they are to be used. The commands to use follow:

fmtflop(1M)	To format a floppy disk
sysadm format(1)	To format a floppy disk using the System Administration DISK MANAGEMENT Menu

A command that sounds related, **fmthard**(1M), is used to define a Volume Table Of Contents (VTOC) for a formatted hard disk (see "Partitions").

Partitions

The next level in disk formatting is partitions. On 3B2 computers, up to 16 partitions can be defined on a hard disk device, up to 8 on a floppy disk. Partitioning of a floppy disk is hardcoded into the driver and may not be changed. On a hard disk, the **fmthard**(1M) command is used to tie the starting points of partitions to sector numbers (sectors are numbered from 0 to the number the disk holds). The number of sectors allocated to a partition is specified, and a tag that is a hex code tells the intended use of the partition. Partition tags 0 through 8 are reserved. The list below shows how the tags may be used on the 3B2 computer.

Name	Number
UNASSIGNED	0
BOOT	1 or 0
ROOT	2
SWAP	3
USR	4

When you first get your 3B2 computer, the primary hard disk devices have already been formatted, and the partitions assigned are something like those shown in Figure 5-5. (The example shown is for the first 155-megabyte disk device in a dual disk system.)

155-Megabyte Hard Disk Drive (blocks per cylinder = 315) (rotational gap = 12)				
Disk Partition	Use	First Sector	Size*	I-Nodes
c1t1d0s0	root	43008	53376	2917
c1t1d0s1	swap	150	42858	—
c1t1d0s2	unassigned			
c1t1d0s3	sysdump	269632	32768	—
c1t1d0s4	install	268065	1567	200
c1t1d0s5	unassigned			
c1t1d0s6	entire disk	0	302400	—
c1t1d0s7	boot	0	150	—
c1t1d0s8	usr2	46620	221445	27680
c1t1d0s9	-			
c1t1d0sa	-			
c1t1d0sb	unassigned			
c1t1d0sc	unassigned			
c1t1d0sd	unassigned			
c1t1d0se	unassigned			
c1t1d0sf	unassigned			

* Size in 512-byte blocks.

Figure 5-5: Disk Partitions, 155-Megabyte Drive

The illustration shows two file systems defined on this drive: the **root** file system and one called **usr2**. Space has also been set aside for the boot file, install file, crash space, and for swap space. Tables showing the default partitioning for all supported devices are in Appendix A, "Device Names and Designators."

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. However, you cannot make a file system larger than a partition size. The maximum number of blocks depends on how the disk was formatted (size, number of partitions), not just the size of the disk.

In UNIX System V Release 3.2.2 on the 3B2 computer the default size of a logical block is 2048 bytes. Subroutines that handle file Input/Output (I/O) work with logical blocks of 2048-bytes (2KB) rather than with the 512-byte physical block size on a disk.

How the File System Works

What we have discussed so far has been the organization of a UNIX system file system on paper and on the physical storage disk. We now want to describe what the UNIX system does with a file system when it is being used.

Tables in Memory

When a file system is identified to the UNIX system through a **mount**(1M) command, an entry is made in the mount table, and the super-block is read into an internal buffer maintained by the kernel. Parts of the super-block that are most needed in memory are the lists of free i-nodes and free storage blocks, and the flags and time fields that are constantly being modified.

The System I-Node Table

The UNIX system maintains a structure known as the system i-node table. Whenever a file is opened, its i-node is copied from the secondary storage disk into the system i-node table. If two or more processes have the same file open, they share the same i-node table entry. The entry includes, among other things, the following:

- The major and minor number of the device from which the i-node was copied
- The i-node number or i-number
- A reference count of the number of pointers to this entry. (A file can be open for more than one process.)

A diagram of the system i-node table is shown in Figure 5-6.

I-node chain pointers
Free-list chain
Flag
Waiting count for i-node
Reference count
Device where i-node resides
I-number
Mode
Number of links
User ID of owner
Group ID of owner
Size of file

Figure 5-6: The System I-Node Table

The System File Table

The system maintains another table called the system file table. Because files may be shared among related processes, a table is needed to keep track of which files are accessible by which process. For each file descriptor, an entry in the system file table contains:

- A flag to tell how the file was opened (read/write)
- A count of the processes pointing to this entry
  (When the count drops to zero, the system drops the entry.)
- A pointer to the system i-node table
- A pointer that tells where in the file the next I/O operation will take place.

The Open File Table

The last table that is used to access files is the open file table. It is located in the user area portion of memory. There is a user area for each process and, consequently, an open file table for each process. An entry in the open file table contains a pointer to the appropriate system file table entry. Figure 5-7 shows how these tables point to each other.

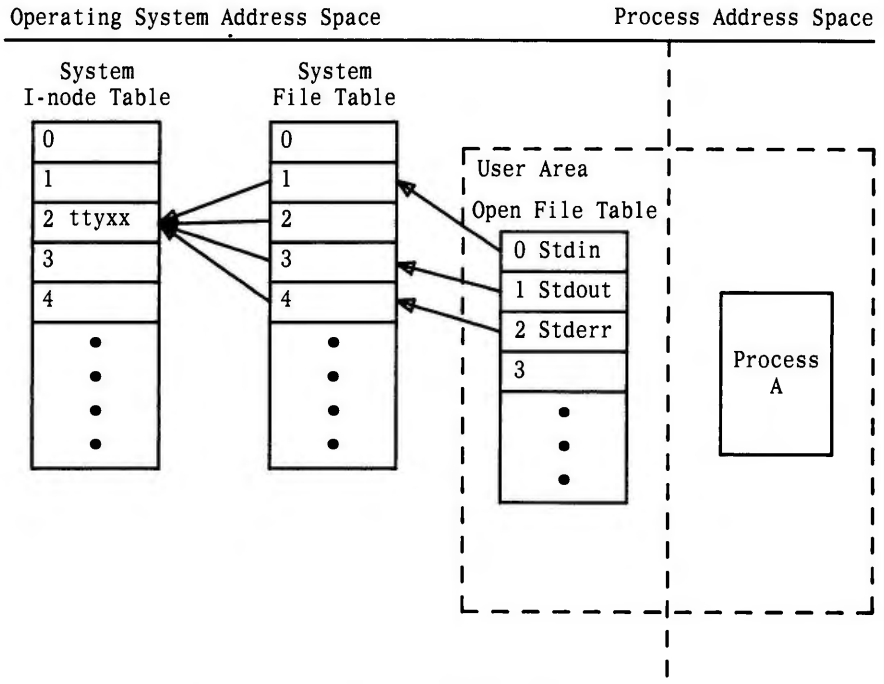


Figure 5-7: File System Tables and Their Pointers

System Steps in Accessing a File

In the next few paragraphs we describe steps followed by the operating system in opening, creating, reading, or writing a file.

Open

Suppose we give the path name **/a/b** to the **open**(2) system call. [Our program probably uses the **fopen**(3) subroutine from the standard I/O library, that in turn invokes the system call.]

1. The operating system sees that the path name starts with a slash, so the root i-node is obtained from the i-node table.
2. Using the root i-node, the system does a linear scan of the root directory file looking for an entry `"a."` When `"a"` is found, the operating system picks up the i-number associated with `"a."`
3. The i-number gives the offset into the i-node list at which the i-node for `"a"` is located. At that location, the system determines that `"a"` is a directory.
4. Directory `"a"` is searched linearly until an entry `"b"` is found.
5. When `"b"` is found, its i-number is picked up and used as an index into the i-list to find the i-node for `"b."`
6. The i-node for `"b"` is determined to be a file and is copied to the system i-node table (assuming it is not already there), and the reference count is initialized.
7. The system file table entry is allocated, the pointer to the system i-node table is set, the offset for the I/O pointer is set to zero to show the beginning of the file, and the reference count is initialized.
8. The user area file descriptor table entry is allocated with a pointer set to the entry in the system file table.
9. The number of the file descriptor slot is returned to the program.

The linear scan algorithm for locating the i-node of a file illustrates why it is advisable to keep directories small. Search time is also speeded up by keeping subdirectory names near the beginning of a directory file. [Use **dcopy**(1M) to do this.]

Create

Creating a file [the **creat**(2) system call] has these additional steps at the beginning.

1. The super-block is referenced for a free i-node number.
2. The mode of the file is established [possibly **and**-ed with the complement of a **umask** entry; see **umask**(2)] and entered in the i-node.
3. Using the i-number, the system goes through a directory search similar to that used in the **open** system call. The difference is that in **creat**, the system writes the last portion of the path name into the directory that is the next to last portion of the path name. The i-number is stored with it.

Read and Write

Both the **read**(2) and **write**(2) system call follow these steps:

1. Using the file descriptor supplied with the call as an index, the user's open file table is read and the pointer to the system file table obtained.
2. The user buffer address and number of bytes to read (write) are supplied as arguments to the call. The correct offset into the file is read from the system file table entry.
3. (Reading) The i-node is found by following the pointer from the system file table entry to the system i-node table. The operating system copies the data from storage to the user's buffer.
4. (Writing) The same pointer chain is followed, but the system writes into the data blocks. If new direct or indirect blocks are needed, they are allocated from the list of free blocks of the file system.
5. Before the system call returns to the user, the number of bytes read (written) is added to the offset in the system file table.
6. The number of bytes read or written is returned to the user.

Files Used by More Than One Process

If related processes are sharing a file descriptor [as happens after a **fork**(1)], they also share the same entry in the system file table. Unrelated processes that access the same file have separate entries in the system file table because they may be reading from or writing to different places in the file. In both cases, the entry in the i-node table is shared; the correct offset at which the read or write should take place is tracked by the offset entry in the system file table.

Path Name Conversion

The directory search and path name conversion takes place only once as long as the file remains open. For later access of the file, the system supplies a file descriptor that is an index into the open file table in your user process area. The open file table points to the system file table entry where the pointer to the system i-node table is picked up. Given the i-node, the system can find the data blocks that make up the file.

Synchronization

The above description, while complex, may seem neat and orderly. The procedure is complicated, moreover, because the UNIX system is a multi-tasking system. To give some tasks prompt attention, the system may make the decision that other tasks are less urgent. In addition, the system keeps a buffer cache and a cache of free blocks and i-nodes in memory, together with the super-block to give more responsive service to users. The stability that comes from having every byte of data in a file immediately written to the storage disk is traded for the gain of being able to give more service to more users.

In normal processing, disk buffers are flushed periodically to the disk devices. This is a system process that is not related directly to any reads or writes of user processes. The process is called "synchronization." It includes writing out the super-blocks in addition to the disk buffers. The **sync** command can be used to cause the writing of super-blocks and updated i-nodes and the flushing of buffers. It is worth noting, however, that the return from the command simply means that the writing was scheduled, not necessarily completed. Therefore, many people enter the command twice in succession to introduce delay.

Search Time

There are two things that have a bearing on the amount of time the system needs to spend looking for and reading in a file:

- The size of the directories being searched
- The size of the file.

As described above, when the UNIX system is locating a file to be opened, it searches linearly through all the directories in the path name. Search time can be reduced in two ways:

1. Keep the number of entries in a directory low. Unless compressed with **dcopy**(1M), a directory retains its largest size. If you have a directory that has had more than 640 entries, you have reached the point of indirect addresses.
2. Move subdirectory names to the start of the directory. The **dcopy**(1M) utility does this by default. When subdirectory names are grouped at the start of a directory, the system finds a match with less searching.

Large files are slower to read because of the need to chain through indirect or double indirect addresses.

Holes in Files

When a file is created by a program rather than by a person using an editor, the program may seek to locations in the file that are blocks away from where previous data in the file was stored. Imagine a program that has written to blocks 1 and 2 of a file. The i-node points to those storage blocks. If the program seeks to block 5 and writes data there, the fifth address pointer in the i-node will hold a live address, but the third and fourth will still be zeros pointing to nothing. The file is now said to have a hole in it. There is nothing really serious about this; the blocks used by the file are the ones that have been written to. If the file system is reorganized [for example, by **dcopy**(1M)], storage blocks will be assigned to blocks 3 and 4; addresses will be added to the i-node; the storage blocks will be taken off the free list and initialized to zeros.

Summary

We have tried in this section to give you an understanding of how the UNIX operating system controls file systems. Seeing how things are supposed to work can give us an appreciation of the steps that must be taken to keep a file system consistent.

Administer the File System

Create a File System and Make It Available

Once a disk is formatted, the next step is to define the file system. The **mkfs**(1M) command is used for this purpose. The **sysadm makefsys**(1) command can be used to define a file system on floppy disk.

Use mkfs

The **mkfs** command has two formats:

mkfs *special_file blocks*[*:i-nodes*] [*gap blocks/cyl*] [**-b** *blocksize*]
mkfs *special_file prototype* [*gap blocks/cyl*] [**-b** *blocksize*]

Notice that in neither format is the file system actually given a name; it is identified by the file name of the special device file on which it will reside. The special device file, traditionally located in the directory **/dev**, is tied to the identifying controller and unit numbers (major and minor, respectively) for the physical device. The format of the minor is feature-dependent.

In the first format 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 format 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 formats 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. The recommended value for the gap is 12, as used during full restore. However, **mkfs** will default to 7. The gap size will not normally affect disk performance. See appendix for default values. Figure 5-8 shows the recommended values for use with the **mkfs** command for devices supported on the 3B2 computer. In the first **mkfs** format, even though the number of blocks in the file is required, the number of i-nodes may be omitted. If the number of i-nodes is omitted, the command will use a default value of one i-node for every four logical storage blocks, rounding up, if necessary, to fill the final i-node block.

DEVICE	GAP SIZE
Hard Disks	12
Floppy Disk	1

Figure 5-8: Interrecord Gap Recommendations

If you use the first format of **mkfs**, the file system is created with a single directory. If you use a prototype file, as noted above, it can include information that causes the command to build and initialize a directory and file structure for the file system. The format of a prototype file is described in the **mkfs**(1M) pages of the *User's and System Administrator's Reference Manual.* Note that the **sysadm makefsys** subcommand has no provision for the use of prototype files.

The final option to **mkfs** lets you specify the logical block size to be used for the file system. By default, the file system has a logical block size of 2048 bytes. With the **-b** option, you can specify a logical block size of 512 bytes, 1024 bytes or 2048 bytes.

Choosing Logical Block Size

Logical block size is the size of the chunks the UNIX system 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 chunk that the disk controller can read or write, usually 512 bytes.

An administrator who uses the **mkfs** (1M) command to make a file system may specify the logical block size of the file system. By default, the logical block size is 2048 bytes (2KB). The **root** and **usr** file systems are delivered as 2KB file systems. Besides 2KB file systems, the UNIX system also supports 512-byte file systems and 1024-byte file systems.

To choose a reasonable logical block size for your system, you must consider performance and space. For information on disk performance, see the section "Improving Disk Utilization" in Chapter 6, "Performance Management." For information on file system space requirements, use the file system block analyzer, **fsba** (1M). For some systems, a 1KB file system

is a good compromise between disk performance and use of space in primary memory and on disk. For special applications (such as file servers) that use many large executable files and data files, a 2KB file system may be a better choice. See Chapter 6, "Performance Management," for more information.

Summary: Creating and Converting 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 where an old file system resides, back up the old file system. For information, see the section "File System Backup and Restore" in this chapter; also see Procedure 5.4, "File System Backup and Restore." Typically, you would use **sysadm** (1) to back up systems with one hard disk; on systems with more than one hard disk, you can also 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 and Partitioning" in Chapter 4, "Disk/Tape Management," and Procedure 3.9, "Reloading the Operating System."
4. Use the **mkfs** (1M) command with the **-b** option to make the new file system with the appropriate logical block size. The **mkfs** (1M) command is described in the section "Administering the File System" in this chapter. Also see Procedure 5.2, "Create File Systems on Hard Disk."

5. Run the command to restore the file system and volume names.
6. If the new file system is not the default file system type, change **fstab** (4) to specify its type.
7. Populate the new file system—for example, use **sysadm** (1) to 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. Procedure 5.4, "File System Backup and Restore," shows an example of a file system restore. [The **volcopy** (1M) and **dd** (1M) commands copy a file system image; they cannot convert logical block size.]

Relating the File System Device to a File System Name

A UNIX system file system is generally referred to by the name of the highest level directory in its hierarchy. The file system shown in Figure 5-2 at the beginning of this chapter is called **/usr** because it is tied to that directory. Similarly, the root file system is called that because its first directory is "root" [represented in UNIX system parlance by a slash (/)]. But we saw above that when the file system was created, the only name on the command line (other than the name of a prototype file, if you used that option) was the name of a special device file. There are a couple of ways in which the file system name and the directory name can be tied together.

The first, and most explicit, is through the **labelit**(1M) command. The **labelit** makes the connection between the device special file and the mounted name of the file system. It writes the name of the file system, that is, its highest level directory, into a field in the super-block. When **labelit** is used for removable file systems, such as those on floppy disk, one command line argument can be the identifying number of a volume. This number, too, is stored in a field in the super-block, but it is common practice to write it on a self-adhesive label that is attached to the floppy disk or tape that holds the file system.

The connection between the device and the file system name is also made by the **mount**(1M) command. This step is mandatory if the file system is to be available to users.

Mount and Unmount File Systems

For a file system to be available to users, the UNIX system has to be told to "mount" it. The **root** file system is always mounted as part of the boot procedure. The **usr** file system, which may be on the same disk device as **root**, is also automatically mounted as the system is being brought up to multiuser mode. The **usr2** file system, which is on the same disk device as **root** on a dual disk system (default), is also automatically mounted as the system is being brought up to multiuser mode.

The issuing of the **mount** command that brings these file systems on-line is hidden in start-up shell procedures. Regardless of whether the **mount** command is hidden or not, its execution causes the file system on a specified disk device to be listed in an internal UNIX system table [called the mount table (information is also contained in the **/etc/mnttab** file)] paired with the directory that is specified. For example, the command:

mount /dev/dsk/c1t1d1s2 /usr

tells the system that **/dev/dsk/c1t1d1s2** contains a file system that begins in the directory **usr**.

The **usr2** file system is in the **/etc/fstab** file to be automatically mounted (along with **usr** on a dual disk system), you may enter the following variation on the command:

mount /usr2

The **mount** command searches **/etc/fstab** for the *special* device that is associated with the **usr2** file system.

Note: The **mount** command has other arguments. See the *User's and System Administrator's Reference Manual* for complete details.

If you try to change directories [**cd**(1)] to a directory in **usr** 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 beneath **usr**. True, there is a directory **usr** (it must exist at the time the **mount** command is issued), but the structure of files and directories below that remain hidden from the UNIX system until the **mount**.

It is common practice for small file systems to be contained completely on one floppy disk. A floppy disk can hold as much as 1422 (512-byte) blocks. That is close to ¾ of a megabyte. You can define file systems on floppy disk and use them either for storage or for live access. Using a floppy disk for live access has the following two disadvantages:

1. The access time is not as fast as the hard disk.
2. The floppy disk ties up the floppy disk device.

It is more common for users to copy in such a file system to a directory on the 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 floppy disk 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. Let's say you have created the directories. The mount point is named **/hk**, and the root is named **/myfs**. You could bring a file system from floppy disk to hard disk with the following command sequence:

mount /dev/diskette /hk -r

(*The* **-r** *means read-only.*)

cd /hk

find . -print | cpio -pdm /myfs

[*See the* **find**(1) *and* **cpio**(1) *manual pages for an explanation of the options used.]*

The command for unmounting a file system requires only the name of the special device. After you have copied in a file system from a floppy disk, for example, you would issue the commands:

cd /
umount /dev/diskette or **/hk**

to free up the floppy disk drive.

Unmounting is frequently a first step before using other commands that operate on file systems. For example, **fsck**(1M), which checks and repairs a file system, and **dcopy**(1M), which copies and compresses a file system, work on unmounted file systems. Unmounting is also an important part of the process of shutting down the system.

Summary

Thus far we have looked at file systems in the abstract. We have seen something of the way they are created, stored on disks or floppy disks, made available to the system, or removed from the system. In the next portion of this chapter we will see how to maintain the integrity of an active file system.

Maintain File Systems

Once a file system is created and made available to users, it is always necessary to monitor how it is being used by the people in the organization. The distinction between a file system and its files may result in some confusion. The administrator's view is more likely to be of the file system, while users tend to think and work in terms of files. When we begin to talk about the tasks involved in keeping file systems working smoothly for users, we have to be ready to deal with users' individual files as well as with the entire system.

Once a file system has been created and made available, there are several tasks routinely done to make certain that the file systems in regular use on a 3B2 computer are providing the level of service and stability they should. They can be grouped into procedures for the following:

- Checking for file system consistency
- Monitoring disk usage
- Compressing and reorganizing file systems
- Backing up and restoring file systems.

The Need for Policies

As with most other aspects of administering a 3B2 computer, file system administration should be based on establishing a set of policies that are appropriate for your organization. There can be no hard-and-fast rules for such things as the size of file systems, the number of users in a file system, the way in which backups are done, the extent to which users can be allowed to keep inactive files in the system, or the amount of disk space a single user is entitled to occupy. These questions can only be resolved within the context of the organization. The number of users, the type of work they are doing, the number of files needed—all are variables. The responsible administrator must determine what best meets the needs of the organization.

Shell Scripts for File System Administration

Once policies have been agreed on, many of the routine tasks connected with file system administration can be incorporated in shell scripts. Monitoring disk usage, for example, can be handled through shell scripts that monitor for you and transmit messages to the system console when exceptions are detected. Here are a few ideas.

- Use a shell script running under **cron**(1M) control to investigate free blocks and free i-nodes and to report on file systems that fall below a given threshold.
- Use a shell script to do automatic clean-ups of files that grow (log files).
- Use a shell script to highlight cases of excessive use of disk space.

Check for File System Consistency

There is a separate section later in this chapter that describes **fsck**(1M), the file system checking utility. However, we want to include this mention of it here because file system checking is central to the whole problem of normal file system maintenance.

Monitor Disk Usage

You need to monitor the level of usage of a file system for the following reasons:

- If not watched regularly, the percentage of disk space used increases until the allocated space is used up.
- When the allocated space is used up, processes run slowly or not at all; the system spends its time putting out a message about being out of file space.
- There is a natural tendency for users to forget about files they no longer use; therefore, files use space needlessly.
- Some files grow larger as a result of perfectly normal use of the system. It is an administrative responsibility to keep them under control.

- Some directories, notably **/tmp**, accumulate files during the day. When the system is first brought up, **/tmp** needs to have enough free blocks to carry it through to **shutdown**(1M).

There are four tasks that are part of keeping disk space uncluttered:

1. Monitoring percent of disk space used
2. Monitoring files and directories that grow
3. Identifying and removing inactive files
4. Identifying large space users.

The tasks mentioned here are shown in Procedure 5, "File System Administration Procedures." In the procedures, the tasks are done through the System Administration menus. This section supplements the procedures in Part 1 and shows some UNIX system commands that you may use.

Monitor 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. In this example, the **df**(1M) command is used.

```
df -t
```

The **-t** option causes the total allocated blocks and i-nodes to be displayed, as well as free blocks and i-nodes. When no file systems are named, information about all mounted file systems is displayed.

Monitor 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
/etc/wtmp	history of system logins
/usr/adm/conlog	history of console terminal I/O
/usr/adm/errlog	history of driver error messages
/usr/adm/sulog	history of **su** commands
/usr/lib/cron/log	history of actions of **/etc/cron**
/usr/lib/help/HELPLOG	actions of **/usr/bin/help**
/usr/lib/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 technique for keeping them down to a reasonable size uses a combination of **tail**(1) and **mv**(1):

```
tail -50 /usr/adm/sulog > /tmp/sulog

mv /tmp/sulog /usr/adm/sulog
```

This sequence puts the last 50 lines of **/usr/adm/sulog** into a temporary file, and then it moves the temporary file to **usr/adm/sulog**, thus effectively truncating the file to the 50 most recent entries.

The error log has a daemon in place to move it once a week, but this may need to be done more frequently, depending on your system. The console log should not be moved while the logger is active.

Identify and Remove 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 commands you might use to do this work are shown below; the policy decisions involved follow:

- How long should a file remain unused before it becomes a candidate for removal?
- Should users be warned that old files are about to be purged?
- Should the files be permanently removed or archived?

The **find**(1) command locates files that have not been accessed recently. The **find** command searches a directory tree beginning at a point named on the command line. It looks for file names that match a given set of expressions, and when a match is found, performs a specified action on the file. This example hardly begins to suggest the full power of **find**.

```
find /usr -type f -mtime +60 -print > /tmp/deadfiles &
```

Here is what the example shows:

/usr	Specifies the path name 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 path name to be printed.
> /tmp/deadfiles &	Directs the output to a temporary file and shows 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.

The **sysadm fileage**(1) command can be used to produce similar information (see Procedure 5.3, "Maintain File Systems").

Identify Large Space Users

Again, the most important questions are not what commands to use to learn who is occupying excessive amounts of disk space, but rather policy questions concerned with deciding what the limits should be. Policy questions include:

- On our system, what constitutes a reasonable amount of disk space for a user to need?
- If a user exceeds the normal amount by 25%, is it possible that the user's job requires extraordinary amounts of disk space?
- Is our system as a whole running short of space? Do our existing limits need to be reviewed?

Two commands produce useful information in this area: **du**(1) and, once again, **find**(1).

The **du** command produces a summary of the block counts for files or directories named in the command line. For example:

```
du /usr
```

displays the block count for all directories in the **usr** 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 /usr -size +10 -print
```

This example produces a display of the path names of all files (and directories) in the **usr** file system that are larger than 10 (512-byte) blocks. Similar information can be produced by the **sysadm filesize**(1) command (see Procedure 5.3, "Maintain File Systems").

File System Backup and Restore

Creating a backup (making a copy) of information stored in your computer system provides the security of knowing you will not have to start all over again from the beginning. You will only lose the data that has been input since the last backup. The importance of establishing and following a file system backup plan is too often unappreciated until data is lost and cannot be recovered. Backing up a file system takes time. Trying to recover lost or damaged data from paper records and best-guess-work, however, takes significantly more time.

Performing backup and restore operations involves handling data in the following forms:

- Entire file systems
- Partial or incremental backup of file systems
- Individual directories and files.

A complete backup copies an entire file system, while an incremental backup only copies files that were "changed" since the last backup.

System Administration menus provide the following commands for backing up and restoring files:

- **hsbackup**—High-speed backup of an entire file system
- **hsrestore**—High-speed restoral of an entire file system
- **backup**—Complete or partial backup of a file system
- **restore**—Restores individual files or a complete file system
- **store**—Backs up individual files or directories.

System Administration menus help you make complete and incremental file system backups to tape or floppy disk and help you restore backup data to the hard disk (see Procedure 5.4, "File System Backup and Restore"). You can copy selected directories and files to floppy disks by using the **sysadm store**(1) command or the **find**(1) command with **cpio**(1). The directories and files are read back to the hard disk by using **sysadm restore**(1) or the appropriate **cpio** command.

Another method of backing up information stored on a 3B2 computer is to copy file systems to another computer system over a high-speed data link. The link between the machines must be a high-speed data link so that data transfers are done in a timely fashion. The other machine should be a larger system with mass storage capability. AT&T Local Area Network products provide the speed necessary to transfer large amounts of data between systems.

The backup plan that you use can include any or all of these methods. The important consideration is that you evaluate the need for system backup and form a backup plan. This plan should be re-evaluated as the use of the machine changes. A System Administration facility for establishing a check of your backup schedule is available; it is described under "Backup Schedule Reminder" later in this section.

Special Precautions

Remember this important fact about the backup and restore commands: when you do backup with a particular method, you are restricted to use a corresponding method to restore that data. For example, if you back up a user file system with **sysadm backup**, you can only restore that file system using **sysadm restore**.

Many of the backup procedures require you to shut down the 3B2 computer to the "single-user" mode. If you attempt to do a backup procedure and the system will not let you, it may mean that you are not logged in properly. Also, shutting down to single user disrupts all user processes and, therefore, should be done with discretion.

Schedule and Plan Backups

There is no standard method for planning and performing backups. Each application and system configuration has different aspects that must be considered. Personal preference also plays a part in establishing backup procedures. The main issue is that you must evaluate your particular system needs for backup and form a backup plan accordingly. The backup plan should be re-evaluated as the use and configuration of the machine changes.

To assist in getting started with a schedule, a basic schedule is provided under **sysadm bupsched**. Change can be made according to your system requirements.

There are many things to consider when formulating a backup schedule. Some of the main considerations include the following:

- The method in which the lost or damaged data will be restored (recovered)
- The importance of the data
- The downtime of the machine needed to do backup
- The use of the system; that is, how the 3B2 computer is being used
- The size of the file system to be saved
- The type(s) of backup to do
- The device(s) used to store the backup data; the storage device could be floppy disk(s), 9-track tape, hard disk(s), or cartridge tape(s).

Restoral/Recovery

The main consideration in formulating a backup plan is the method of recovery for lost or damaged data. The recovery process should be as quick and easy as possible. The speed of recovery depends on many things. The device used to store the data is important. For example, an individual file can be restored faster from a floppy disk than from a cartridge tape. However, if a complete file system needs to be restored, the tape will be faster than inserting and removing several floppy disks.

Backing up to a SCSI hard disk may be the easiest method of all the backup alternatives as far as recovery of data. These devices have fast access times and require no removable media storage. However, it may not be cost-effective for you to use one of these devices as a backup media.

Organization of Data

The ease of recovery depends greatly on how the data is organized and how the file systems are set up on the machine. It is better if all files and file systems are structured logically and efficiently.

Organization also refers to how the backup media are physically stored. Rummaging through a drawer of floppy disks for the one disk labeled "john's files" would be ridiculous. Each time a backup of a file or file system is performed, the storage medium (cartridge tape, 9-track tape, or floppy disk) should be labeled with the following information and then properly stored.

1. Identify the type of backup performed (complete, incremental, etc.).
2. Record the complete name of the file or file system.
3. Record the date and time.
4. Specify the order or sequence number for the storage medium (for example: 1 of 5, 2 of 5). If more than one medium is required, you will need to know which medium to load first.
5. Provide some instructional information for someone else who may have to restore the data from the storage medium.
6. Specify the device used as the backup media. Use the System Administration device name such as qtape1 (SCSI cartridge tape), 9track1 (SCSI 9-track tape), etc.

The following is an example of what a storage medium label might look like:

Complete Backup of all files in: /usr
Fri. 04/22/88 5:38:04 pm
PART 1 of 2
TO LOAD: sysadm restore
Backed up on: qtape1

This information should be used to organize the storage media into a library of stored data. In this way, obsolete data can be removed or written over instead of building a historical library of numerous floppy disks and tapes. Also, you do not want to create a great deal of work for yourself by having to look through all the saved data to find the one file that was lost or damaged.

Importance of Data

Important data should be saved more often than data of average importance. For something critical, the data should be saved every time there is an update. This will ensure that the data is safe if an unexpected system crash occurs before the planned backup can be performed. For example, if your building has an unreliable power source, you should naturally do backups more often than you would in most other environments.

Downtime of System

The 3B2 computer should be in the single-user mode when a backup is being performed or when a complete restore is taking place. Therefore, the amount of backup downtime (time that the machine is unavailable for use) should be considered. Therefore, you may want to schedule backups during "nonworking" hours.

The time in which a file may need to be recovered is not predictable. Recovery downtime is another reason you should organize the stored data appropriately with the configuration of the system. If you know where a stored file is located, the file will be easier to find and can be more quickly replaced. Also, if there are just a few files, you can restore them in multiuser mode and notify the user that needs them without disturbing other users.

System Application

The way the 3B2 computer is being used influences the planning of backup schedules. A system with several users needs to be backed up more often than a system with one or two users. The more users that are on a system automatically increases the chances of files being lost or damaged, thus, the need for more frequent backups.

Another example of a system that needs to be backed up frequently would be a data base that is accessed by many customers. This data base must be kept as current as possible so customers will not be conducting business using data from last week. If there is a system failure, you will want the restored data to match the latest accessed data as accurately as possible.

File Size

The sizes of files and file systems of data you wish to save are important considerations when formulating a backup schedule. You will probably want to store large file systems on a cartridge tape, 9-track tape, or hard disk drive so you do not have to load several floppy disks just to restore one file system. A complete backup of a typical file system could be done on one or two cartridge tapes as opposed to 30 or more floppy disks.

On the other hand, frequent storing of small files could be done rapidly on floppy disks. If you have a small file system that contains critical data, this data can be backed up on floppy disks at the end of each day.

Types of Backup

In developing a file system backup plan, you have to decide what type of backup procedure is best for your application. You can do frequent backups using the "high-speed" **hsbackup** command to a SCSI device. Or, you can do less frequent complete backups using **backup** and do routine incremental backups. Or, you may want to do a combination of "high-speed" and regular backups with incrementals depending on the size and frequency of change of individual file systems. The final decision is yours. However, the following list gives some things to consider about the different backup procedures that may help you in making your decision:

- You cannot mix backup and restore procedurers. File systems backed up using **hsbackup** must be restored using **hsrestore**. You cannot use **sysadm restore** in this case.
- **hsbackup/hsrestore**
  - Complete backups and restorals using these commands are fast, usually taking only a matter of minutes.
  - The SCSI devices (cartridge tape, 9-track tape, and hard disks) are the primary backup media.
  - **hsbackup** backs up one entire file system per operation. This means only one file system per backup media. However, if your file systems have been partitioned to match available backup media, you will get the maximum benefits from both **hsbackup** and your backup media. Also, it makes keeping track of what is on each backup media much easier.
  - **hsbackup** does not provide for incremental backups. However, because complete backups using **hsbackup** do not require much time, they can be done more often than complete backups using **sysadm backup**. More frequent complete backups mean there is less of a need for incremental backups.
  - **hsrestore** restorals are normally much easier to do than incremental restorals. Because you are restoring from a complete backup, you only have to reload from one previous backup. Incremental restorals require you to reload, in sequence, all backups from the desired backup to the most recent.

- **hsbackup** does not provide for backups to floppy disk. However, complete backups to floppy disk are generally time-consuming and not cost-effective.
- Restorals using **hsrestore** mean the entire file system will be rewritten. This may retrieve the desired data, but if the file system is restored to the original file system, it may cause other users to lose recently entered data. This can be resolved by restoring the file system to an unused file system on hard disk. Then, you can mount that file system, retrieve the data, and mail to the appropriate user.
- Cannot backup the **root (/)** or **usr** file systems.

- **backup/restore**
  - Complete or incremental backups of selected file systems can be done in one operation to the same device. This includes the **root(/)** and **usr** file systems.
  - Single files, directories, or entire file systems can be restored.
  - Complete or incremental backups to floppy disks are possible.
  - **backup** copies only the data that is written in a file system. It does not duplicate the entire file system like **hsbackup**. In cases where there is a small amount of data written in a large file system, a complete backup may be done quicker using **backup**. Usually, **backup** takes much longer than **hsbackup**.
  - **backup** only provides for backing up to removable media devices, such as SCSI cartridge tape. You cannot back up to a hard disk using **backup**.

Storage Device

The device chosen to store the backup data falls in the personal preference category. The storage device can be floppy disk(s), cartridge tape(s), 9-track tape reels, hard disk drives, or combinations of the above. Each device has advantages and disadvantages. Some of the advantages and disadvantages are pointed out in the following discussion:

- In general, the SCSI cartridge tape seems to be the best choice for backup devices. SCSI cartridge tapes have storage capacities up to 120 megabytes, are inexpensive removable media, and are a convenient size for storage.
- A SCSI disk drive makes an excellent backup device as far as recovery performance. For example, the second hard disk has fast data-access time, 155 or 317 megabytes of storage area, and can accommodate more than one file system. However, it has fixed media that makes it expensive when compared to the other backup devices, and using the second disk as a backup device will not enhance your system's performance.
- The optional SCSI 9-track tape, like the SCSI cartridge tape, has large storage capacity and removable media. The 9-track tape can accommodate only one file system.
- The integral floppy disk has good access time and inexpensive, removable media; however, the floppy disk has limited capacity and should only be considered in small storage operations.

The next section discusses the backup and corresponding restore commands.

Backup and Restore Commands

The System Administration File Management Menu gives you the simplest methods to do the many backup and restore procedures. The following is an example of the File Management Menu.

```
                        FILE MANAGEMENT
 1 backup    backup files from built-in disk to removable media
 2 bupsched  backup reminder scheduling menu
 3 diskuse   display how much of the hard disk is being used
 4 fileage   list files older than a particular date
 5 filesize  list the largest files in a particular directory
 6 hsbackup  high-speed backup of a file system
 7 hsrestore high-speed restore of a file system
 8 restore   restore files from "backup" & "store" media to built-in disk
 9 store     store files and directories of files onto removable media

Enter number, a name, the initial part of a name, or
? or <number>? for HELP, ^ to GO BACK, q to QUIT:
```

Complete Backup and Restore

Complete system backups can be done with either the **sysadm backup** command or the **sysadm hsbackup** command. Generally, the **sysadm hsbackup** command is used to do complete file system backups and the **sysadm backup** command is used to do incremental backups. Complete backups of the **root(/)** and **/usr** file systems must be done with the **backup** command since they cannot be backed up with the **hsbackup** command.

The **hsbackup** command uses the **volcopy** command as its copy vehicle.

Use the **sysadm hsrestore** command to restore file systems that were backed up using the **hsbackup** command.

The complete backup used by the System Administration backup facilities copies directories and files of a specified mounted file system to floppy disks or tape. Complete backups plus incremental backups combine to form a unified backup strategy. Complete backups can be done without intervening incremental backups, but incremental backups must be preceded by at least one complete backup to establish the base.

Incremental Backup and Restore

A complete backup (using **sysadm backup**) can be done at any time and must be done at least once before an incremental backup can be done on the file system.

The **sysadm store** command copies selected files or directories of files from a specified file system.

Use the **sysadm restore** command to restore files or file systems that have been copied using the **backup** or **store** commands.

The **backup**, **restore**, and **store** commands use a version of the **cpio** command as their copying vehicle.

The incremental backup used by the System Administration backup facilities copies files that have changed since the last backup to floppy disks or tape. An incremental backup is fast in comparison to the time required to do a complete backup because of the obvious fact that only files that have changed since the last prior backup (either complete or incremental) are being collected.

Note: Not all changed directories and files are copied in an incremental backup. The contents of **/etc/save.d/except** specifies files and directories that ARE NOT copied. A typical **/etc/save.d/except** file is shown in Figure 5-9.

```
#  Patterns of file names to be excluded from saving by savefiles.
#  These are ed(1) regular expressions.
.news_time$
/a.out$
/core$
/dead.letter$
/ed.hup$
/nohup.out$
/tmp/
^/etc/mnttab$
^/etc/save.d/timestamp/
^/etc/utmp$
^/etc/wtmp$
^/usr/adm/
^/usr/asp/
^/usr/at/
^/usr/crash/
^/usr/games/
^/usr/news/
^/usr/rje/
^/usr/spool/
^/usr/tmp/
```

Figure 5-9: Sample **/etc/save.d/except** File

Selective Backup

Specific directories and files can be quickly saved on a single floppy disk by using the **find**(1) and **cpio**(1) commands. The amount of data copied to a single floppy disk cannot exceed 1422 blocks (512 bytes per block). Before you copy the selected directory(ies) or file(s), check the number of blocks involved.

The **find** and **cpio** combination may also be used to back up directories and files on tape. A cartridge tape can hold 60 or 120 megabytes of data.

Multiple Save Sets

The multiple save set (MSS) feature allows you to create and store multiple **cpio**(1) type backups on either of the following types of media:

- Cartridge tape
- 9-track tape.

The capability to keep multiple backups on tapes (cartridge or 9-track) allows you to use the tapes entire capacity for more effective and efficient backups. This means, for example, that you can store incremental backups from Monday, Tuesday, and Wednesday, etc., on the same tape.

The MSS feature is provided for both the cartridge tape and 9-track drives via System Administration file management. The MSS feature is implemented as options to the System Administration **backup**, **store**, and **restore** commands.

This section covers the following topics:

- The two types of multiple backups provided by the MSS feature
- Enhancements to the System Administration facility (e.g., backup, store, and restore)
- Differences in compatibility between an MSS backup and a standard backup
- Manual method for reading MSS backup tapes.

Backups

After a complete backup of a file system has been made to tape, incremental backups of that file system can be appended to the complete backup on the same tape. This simplifies incremental backups and provides a more efficient use of the entire tape medium.

Backups can consist of multiple unrelated save sets (e.g., copies of independent files, directories, and file systems such as */*, */usr*, and */usr2*) on a tape medium by using **sysadm store**. Also, backups can consist of a single complete backup of a file system such as */*, */usr*, or */usr2*, followed by any number of incremental backups of that initial complete backup on the same medium, or subsequent media, by using **sysadm backup**.

The Restore Feature for MSS Backups

This feature allows you to restore any number of (or all) backups residing on cartridge or 9-track media. The **restore** command reads the label information and prints it on the screen, in a convenient format, for each backup you may want to restore. Using this information, you have the capability to do the following:

- Position the medium at a specific save set.
- List the contents of a particular save set.
- Restore any files, directories, or the entire save set.
- Restore the latest version of specific files, directories, or the entire media.

When you position the medium to a specific save set, the label information is printed. This allows you to search for a particular save set that you need to restore.

Note: Although the MSS feature provides a mechanism to locate specific save sets, it is recommended that you maintain a log of save sets written to the tape. This log should include the save set name, the save set position on the tape, the date that the save set was made, and the save set size. A log is useful when a portion of a tape becomes corrupted, and you have to manually read the tape.

Also, you should include the list of save sets and their sequence on the tape as part of the external tape labeling information.

Compatibility Between MSS Backup and Standard Backup

The MSS feature has been implemented while still maintaining compatibility with older backup facilities. This has been done by having the system prompt you to select MSS or standard backup. A <CR> response to the prompt puts you in the older backup facility. This prompt is provided in the **sysadm backup** and **sysadm store** commands. Determination of MSS or standard backup is done transparently in the **sysadm restore** command.

Automatic Density Selection

Once a density has been selected for a tape, that tape can only be used at that density. During subsequent backups, the software reads information from the tape and automatically selects the appropriate density to be written. Therefore, the density prompts are not displayed for additional backups on the same medium.

Introduction to Manual Method of Reading MSS Backup Tapes

A tape created by an MSS backup is not compatible with a tape created by the standard backup because of the enhanced logical backup format. A manual method can be used to read backup tapes made using the MSS feature. This method is to position over each label (**</dev/rmt/c1t1d0hn**) and then read the **cpio** copy.

Backup Format for MSS Backups

A new logical backup format for the tape medium has been adopted to manage multiple **cpio** save sets. The new format requires that a label be written at the beginning of each **cpio** save set. This label is a single 512-byte block of information that identifies what is in the **cpio** save set. All single backup MSS tapes and the initial tape of a multiple MSS tape backup has a label written at the Beginning Of Tape (BOT). Backups that extend across a tape have two labels written after the completions of the **cpio** copy that crossed the tape. Multiple tapes containing cross-over data are ended with an End-Of-Tape (EOT) mark. The last multiple tape is ended with an End-Of-Data (EOD) mark. The medium can be positioned at the EOD for appending.

An example tape format is shown in Figure 5-10.

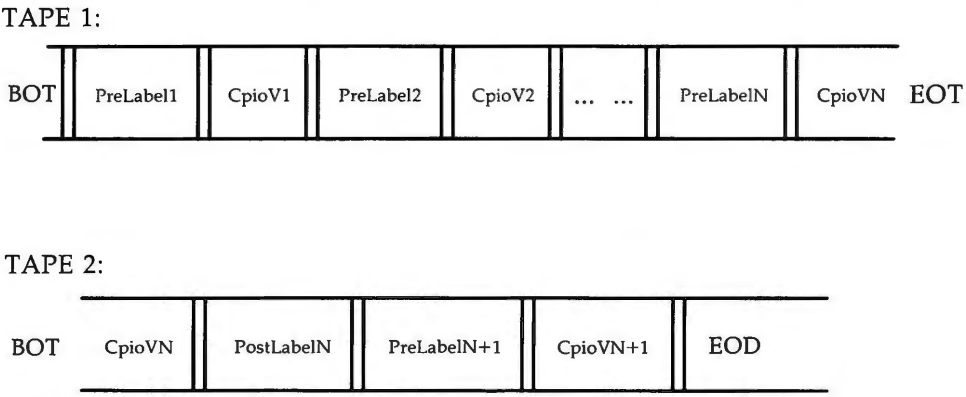


Figure 5-10: Backup Format for Multiple Backups

Manual Recovery of User Data From MSS Tapes

Manual methods of reading MSS tapes are described in the following sections. These methods of recovering data from MSS tapes are most useful when a portion of a tape has been corrupted, and you desire to salvage data from areas not affected by the corruption. Because the MSS feature allows you to put multiple backups on a single tape, there may be backups that can be recovered from a damaged tape.

Figure 5-10 shows the format of MSS tapes. After the Beginning-Of-Tape (BOT) mark, notice that the first tape written in a multiple or single tape backup has a "filemark" which distinguishes it from successive tapes. Additional tapes in a multiple tape backup do not contain the BOT filemark.

Filemarks also separate the data written to the tape by the "Copy File Archives In and Out" (**cpio**) command.

The position of the filemarks and the constant size, 512 bytes, of labels allows you to sequence through the contents of the tape until the desired data can be retrieved.

Recovering User Data From the First Tape

Once the tape is loaded, a "Convert and Copy a File" command (**/bin/dd**) is used to read the tape. The **dd** command is used to position the tape to the desired save set and to read the label before recovering user data.

Note: *Tape Device***n** in the following procedures refers to the input device file name. The format of *Tape Device***n** for a 9-track tape, could be:

/dev/rmt/c?t?d?hn.

where "h" stands for high density and "n" is for "no rewind." Refer to the **dd** manual page for more details.

The following procedure can be used to recover user data from the MSS backup tape. Refer to Figure 5-10 for the backup format.

1. Load tape
2. Position the tape after the first filemark by entering:

   dd if=*Tape Device***n of=/tmp/label bs=512**<CR>

   or

   < *Tape Device*n<CR>
3. Read the label at the current tape position by entering:

   dd if=*Tape Device***n of=/tmp/label bs=512**<CR>
4. Read the displayed contents (of file **/tmp/label**) and determine if this is the save set to be recovered.

5. If this is the save set to be recovered, skip this step. If this is not the save set to be recovered, enter:

   dd if=*Tape Device***n of=/dev/null bs=512**<CR>

   and repeat the previous Steps 3 and 4.

6. The **cpio** command restores the save set contents into the current directory. Make sure you are in the appropriate directory before performing the restore.

7. When positioned at the correct save set and appropriate directory, recover the data by entering:

   cpio -icdvB < *Tape Device*n<CR>

Recovering User Data That Crosses Tapes

To recover data that crosses tapes, perform the following steps:

1. Load the tape with the initial part of data that crosses to another tape.
2. Position to the last save set on the tape. (described previously in "Recovering User Data From the First Tape").
3. Restore the save set contents into the current directory by entering:

   cpio -icdvB -I*Tape Device*<CR>

4. A message will be displayed when the end of media is reached.
5. Load the next tape and follow the instructions displayed.

Recovering User Data From the Second and Successive Tapes

User data can be recovered from the second or following tapes of multiple tape backups without starting from the first tape. The recovery method is slightly different because the second and successive tapes do not contain the identifying BOT filemark. Refer to Figure 5-10 for the tape format.

Perform the following steps to recover user data from the second and successive MSS tapes.

1. Load tape.
2. Position to the end of the cross-over data on the tape by entering:

   dd if=*Tape Device***n of=/dev/null bs=512**<CR>

3. Position to the first save set on the tape by entering:

   dd if=*Tape Device***n of=/dev/null bs=512**<CR>

4. Perform the procedures from "Recovering User Data From the First Tape," beginning with Step 3.

For an alternate positioning method, refer to the **tapecntl**(1M) manual page.

Backup Schedule Reminder

A feature of the System Administration FILE MANAGEMENT Menu is a way to create reminders for yourself about your backup schedule. The backup schedule reminder feature lets you do the following:

1. Schedule reminder messages that say, in effect, "If the machine is shut down within this time range, send a message to the console to do a backup of this or that file system."
2. Schedule reminder checks that say, "If it gets to be this time and the machine has not been shut down, take a look to see if any reminder messages would have been sent had the machine been shut down."

This reminder mechanism does not relieve you of the job of working out a reasonable schedule of backups. It merely nudges you about the schedule you have set up. Procedure 5.4, "File System Backup and Restore," shows the steps you should follow to use the reminder feature. You can, if you prefer, set up reminder notices using **cron**(1M) and **crontab**(1).

What Can Go Wrong With a File System

Most of the things that can corrupt a file system have to do with the failure of the correct pointer and count information to make it out to the storage medium. This can be caused by the following:

- Hardware failure
- Program interrupts
- Human error
- Combination of hardware/program failures and incorrect procedures.

Hardware Failure

There is no effective way of predicting when hardware failure will occur. The best way of dealing with it is to be sure that recommended diagnostic and maintenance procedures are followed conscientiously. For the 3B2 computer there is a utility that flags bad blocks on hard disk and uses a substitute area for blocks the system attempts to write to a flagged block (see Chapter 4, "Disk/Tape Management").

Program Interrupts

It is possible that errors that cause a program to fail might result in the loss of some data. It is not easy to generalize about this because the range of possibilities is so large. Perhaps the best thing to be said is that programs should be exhaustively tested before they are put into production with valuable data.

Human Error

While it may be painful to admit it, probably the greatest cause of file system corruption falls under this heading. We are recommending here four rules that should be followed by anyone who manages file systems.

1. ALWAYS check a file system before mounting it. Nothing complicates the problem of cleaning up a corrupt file system so much as allowing it to be used when it is bad.
2. NEVER remove a file system physically without first unmounting it.
3. ALWAYS use the **sync** command before shutting down the system and before unmounting a file system.
4. NEVER physically write-protect a mounted file system, unless it is mounted "read only."

The random nature of all these mishaps simply underscores the importance of establishing and observing good backup practices. It is the most effective form of insurance against data loss.

How to Check a File System for Consistency

When the UNIX operating system is brought up, a consistency check of the file systems should always be done. On the 3B2 computer this check is automatically done as part of the powerup process. Included as part of the process is the command **fsstat**(1M). The **fsstat** command returns a code for each file system on the hard disk indicating whether the consistency checking and repair program, **fsck**(1M), should be run.

These same commands or **sysadm checkfsys**(1) should be used to check file systems not mounted routinely as part of the powerup 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 noted 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 go undetected, the final loss can be substantial.

The fsck Utility

The file system check **(fsck)** utility is an interactive file system check and repair program. Procedure 5.3, "Maintain File System," shows the steps required to run **fsck** with the **sysadm checkfsys** command. The **fsck** command uses the information carried in the file system itself to do consistency checks. If an inconsistency is detected, a message describing the inconsistency is displayed. You may elect to have **fsck** make the repair. The reason you might choose to have **fsck** ignore an inconsistency is that you judge the problem to be so severe that you want either to fix it yourself using the **fsdb(1M)** utility, or you plan to go back to an earlier version of the file system. The decision to have **fsck** ignore inconsistencies and then do nothing about them yourself is a bad decision. File system inconsistencies do not repair themselves. If ignored, they only get worse.

The fsck Command

The **fsck** command is used to check and repair inconsistencies in a file system. Except for the **root** file system, a file system should be unmounted while it is being checked. 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 following is the general format of the **fsck** command:

fsck [**-y**][**-n**][**-b**][**-l**][**-s***X*][**-S***X*][**-t***file*][**-q**][**-D**][**-f**] [*fsdevice*]

The options of the **fsck** command 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.

-b Reboots the system if the file system being checked is the root (/) file system and changes have been made by **fsck**. If only minor, fixable damage is found, the file system is remounted.

-l Lists the path names of corrupted files, otherwise, you would have to record the inode numbers and use the utility *ff* (fast find) to identify the files and path names.

-s*X* Specifies an unconditional reconstruction of the free list. Following the reconstruction of the free list, the system should be rebooted so that the in-core copy of the super-block is updated (see the **-b** option if root is file system). 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 1KB file systems. See the "Using mkfs" section of this chapter for more information.

Device	*-sblocks/cylinder:gap*
155 MB Hard Disk	**-s315:12**
317 MB Hard Disk	**-s510:12**
Floppy Disk	**-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 if the file system check requires additional memory. If this option is not specified, the process asks for a file name when more memory is needed.

-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.

fsdevice Names the special block device file associated with a file system. If no device name is specified, **fsck** checks all file systems named in **/etc/checklist**.

Sample Command Use

The command line below shows **fsck** being entered to check the **root** 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 (i-nodes), blocks, and free blocks.

```
# fsck /dev/dsk/c1t1d0s0

  /dev/dsk/c1t1d0s0
  File System: root Volume: root

  ** 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
#
```

File System Components Checked by fsck

Before getting into a discussion of the **fsck** phases and the messages that may appear in each, it is important to review the components of a UNIX system file system and to describe the kinds of consistency checks that are applied to them.

Super-Block

The super-block is vulnerable because every change to the file system blocks or i-nodes 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 is almost certainly corrupted. The super-block can be checked for inconsistencies involving the following:

- File system size
- I-node list size
- Free-block list
- Free-block count
- Free i-node count.

File System Size and I-Node List Size

Total file system size must be greater than the number of blocks used by the super-block plus the blocks used by the list of i-nodes. The number of i-nodes must be less than 65,532. While there is no way to check these sizes, **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 the following:

- 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 of 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 blocks in the file system. Each block number is compared to a list of already 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 i-nodes, 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 found free within the file system. If the counts do not agree, **fsck** can replace the count in the super-block by the free-block count.

Free I-Node Count

The super-block contains a count of the number of free i-nodes within the file system. The **fsck** program compares this count to the number of i-nodes it found free within the file system. If the counts do not agree, **fsck** can replace the count in the super-block by the free i-node count.

I-Nodes

The list of i-nodes is checked sequentially starting with i-node 1 (there is no i-node 0). Each i-node is checked for inconsistencies involving the following:

- Format and type
- Link count
- Duplicate blocks
- Bad block numbers
- I-node size.

Format and Type

Each i-node contains a mode word. This mode word describes the type and state of the i-node. I-nodes may be one of five types:

- Regular
- Directory
- Block special
- Character special
- First in, first out (FIFO) (named pipe).

If an i-node is not one of these types, it is illegal. I-nodes may be in one of three states: unallocated, allocated, and partially allocated. This last state means an incorrectly formatted i-node. An i-node can reach this state if, for example, bad data is written into the i-node list through a hardware failure. The only corrective action **fsck** can take is to clear the i-node.

Link Count

Each i-node contains a count of the number of directory entries linked to it. The **fsck** program verifies the link count of each i-node by examining the total directory structure, starting from the root directory and calculating a link count for each i-node.

If the link count stored in the i-node and the link count determined by **fsck** do not agree, the reason may be as follows:

- Stored count not 0, actual count 0

  No directory entry appears for the i-node.
  fsck can link the disconnected file to the lost+found directory.

- Stored count not 0, actual count not 0, counts unequal

  Directory entry possibly removed without i-node update.
  fsck can replace the stored link count with the actual link count.

Duplicate Blocks

Each i-node contains a list of all blocks claimed by the i-node. The **fsck** program compares each block number claimed by an i-node to a list of allocated blocks. If a block number claimed by an i-node is on the allocated-blocks list, it is put on a duplicate-blocks list. If the block number is not on the allocated-blocks list, it is put there. If a duplicate-blocks list develops, **fsck** makes a second pass of the i-node list to find the other i-node that claims the duplicated block. While there is not enough information available to determine which i-node is in error, most of the time the i-node with the latest modification time is correct. This condition can occur by using a file system with blocks claimed by both the free-block list and by other parts of the file system. A large number of duplicate blocks in an i-node may be caused by an indirect block not being written to the file system. The **fsck** program prompts the user to clear both i-nodes.

Bad Block Numbers

The **fsck** program checks each block number claimed by an i-node for a value lower than that of the first data block or greater than the last block in

the file system. If the block number is outside this range, the block number is bad. If there are many bad block numbers in an i-node, it may be caused by an indirect block not being written to the file system. The **fsck** program prompts the user to clear the i-node.

Note: A certain amount of semantic confusion is possible here. A bad block number in a file system is not the same as a bad (unreadable) block on a hard disk.

I-Node Size

Each i-node contains a 32-bit (4-byte) size field. This size shows the number of characters in the file associated with the i-node. A directory i-node within the file system has the directory bit set in the i-node mode word. The directory size must be a multiple of 16, because a directory entry contains 16 bytes (2 bytes for the i-node number and 14 bytes for the file or directory name).

If the directory size is not a multiple of 16, **fsck** warns of directory misalignment. This is only a warning because not enough information can be gathered to correct the misalignment.

For a regular file, a rough check of the consistency of the size field of an i-node can be performed by using the number of characters shown in the size field to calculate how many blocks should be associated with the i-node, and comparing that to the actual number of blocks claimed by the i-node.

The Algorithm

The **fsck** program calculates the number of blocks that should be in a file by dividing the number of characters in an i-node by 512 (the number of characters per block) and rounding up. One block is added for each indirect block associated with the i-node.

If the actual number of blocks does not match the computed number of blocks, **fsck** warns of a possible file-size error. This is only a warning. Logical blocks can be created in random order, and the UNIX system does not fill them in. A check of the file would be required to tell if the error is real (see the section on "Holes in Files" discussed earlier in this chapter).

Indirect Blocks

Indirect blocks are owned by an i-node. Therefore, inconsistencies in an indirect block directly affect the i-node that owns it. Inconsistencies that can be checked are:

- Blocks already claimed by another i-node
- Block numbers outside the range of the file system.

The consistency checks described under "Duplicate Blocks" and "Bad Block Numbers" are performed for indirect blocks as well as for the direct blocks of an i-node.

Directory Data Blocks

Directories are distinguished from regular files by an entry in the mode field of the i-node. Data blocks associated with a directory contain the directory entries. Directory data blocks are checked for inconsistencies involving the following:

- Directory i-node numbers pointing to unallocated i-nodes
- Directory i-node numbers greater than the number of i-nodes in the file system
- Incorrect directory i-node numbers for "." and ".." directories
- Directories disconnected from the file system.

Directory Unallocated

If a directory entry i-node number points to an unallocated i-node, **fsck** can remove that directory entry. This condition occurs if the data blocks containing the directory entries are modified and written out while the i-node is not yet written out.

Bad I-Node Number

If a directory entry i-node number is pointing beyond the end of the i-node list, **fsck** can remove that directory entry. This condition occurs if bad data is written into a directory data block.

Incorrect "." and ".." Entries

The directory i-node number entry for "." should be the first entry in the directory data block. Its value should be equal to the i-node number for the directory data block. The directory i-node number entry for ".." should be the second entry in the directory data block. Its value should be equal to the i-node number for the parent of the directory entry (or the i-node number of the directory data block if the directory is the root directory). If the directory i-node 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 directories are found not to be linked into the file system, **fsck** links the directory back into the file system in the **lost+found** directory. This condition can be caused by i-nodes being written to the file system with the corresponding directory data blocks not being written to the file system. When a file is linked into the **lost+found** directory, the owner of the file needs to be told about it.

Regular Data Blocks

Data blocks associated with a regular file hold the contents of the file. The **fsck** program does not attempt to check the validity of the contents of the data blocks of a regular file.

Run fsck

The **fsck** program runs in phases. Each phase reports errors it detects. If an 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 5-11 identifies the abbreviations used in the **fsck** error messages.

Abbreviation	Definition
BLK	Block number
DUP	Duplicate block number
DIR	Directory name
MTIME	Time file was last modified
UNREF	Unreferenced

Figure 5-11: Error Message Abbreviations in **fsck**

The following single-letter abbreviations, used in the messages shown in the pages that follow, are replaced by a value when the message appears on your screen.

Abbreviation	Definition
B	Block number
F	File (or directory) name
I	I-node number
M	File mode
O	User-id of a file's owner
S	File size
T	Time file was last modified
X	Link count Number of BAD, DUP, or MISSING blocks Number of files (depending on context)
Y	Corrected link count number 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 quits (exits) on initialization errors.

General Errors

Three error messages may appear in any phase. While they seem to offer the option to continue, it is generally best to regard them as fatal, end the run, and investigate what may have caused the problem.

CANNOT SEEK: BLK B (CONTINUE?)
The request to move to a specified block number *B* in the file system failed. The occurrence of this error condition shows a serious problem (probably a hardware failure) that may require additional help.

CANNOT READ: BLK B (CONTINUE?)
The request for reading a specified block number *B* in the file system failed. The occurrence of this error condition shows a serious problem (probably a hardware failure) that may require additional help.

CANNOT WRITE: BLK B (CONTINUE?)
The request for writing 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 says:
  - Terminate program.
    (This is the recommended response.)
- A y(yes) response to the CONTINUE? prompt says:
  - Attempt to continue to run file system check.
    Often, however, the problem persists. The error condition does not allow a complete check of the file system. A second run of **fsck** should be made to recheck this file system.

Phase 1: Check Blocks and Sizes

This phase checks the i-node list. It reports error conditions resulting from the following:

- Checking i-node types
- Setting up the zero-link-count table
- Examining i-node block numbers for bad or duplicate blocks
- Checking i-node size
- Checking i-node format.

Types of Error Messages—Phase 1

Phase 1 has three types of error messages:

1. Information messages
2. Messages with a CONTINUE? prompt
3. Messages with a CLEAR? prompt.

There is a connection between some information messages and messages with a CONTINUE? prompt. The meaning of the CONTINUE? prompt generally is that some limit of tolerance has been reached.

Meaning of Yes/No Responses—Phase 1

- In Phase 1, an n(no) response to the CONTINUE? prompt says:
  - Terminate the program.
- In Phase 1, a y(yes) response to the CONTINUE? prompt says:
  - Continue with the program.
  - This error condition means that a complete check of the file system is not possible. A second run of **fsck** should be made to recheck this file system.
- In Phase 1, an n(no) response to the CLEAR? prompt says:
  - Ignore the error condition.

- A NO response is only appropriate if the user intends to take other measures to fix the problem.

- In Phase 1, a y(yes) response to the CLEAR? prompt says:
  - Deallocate i-node *I* by zeroing its contents.
  - This may invoke the UNALLOCATED error condition in Phase 2 for each directory entry pointing to this i-node.

Phase 1 Error Messages

UNKNOWN FILE TYPE I=I (CLEAR?)
The mode word of the i-node *I* suggests that the i-node is not a pipe, special character i-node, regular i-node, or directory i-node.

LINK COUNT TABLE OVERFLOW (CONTINUE?)
An internal table for **fsck** containing allocated i-nodes with a link count of zero has no more room.

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 invoke the EXCESSIVE BAD BLKS error condition in Phase 1 if i-node *I* has too many block numbers outside the file system range. This error condition invokes the BAD/DUP error condition in Phase 2 and Phase 4.

EXCESSIVE BAD BLOCKS I=I (CONTINUE?)
There is more than a tolerable number (usually 10) of 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 i-node *I*.

B DUP I=I
I-node *I* contains block number *B*, which is already claimed by another i-node. This error condition may invoke the EXCESSIVE DUP BLKS error condition in Phase 1 if i-node *I* has too many block numbers claimed by other i-nodes. This error condition invokes Phase 1B and the BAD/DUP error condition in Phase 2 and Phase 4.

EXCESSIVE DUP BLKS I=I (CONTINUE?)
There is more than a tolerable number (usually 10) of blocks claimed by other i-nodes.

DUP TABLE OVERFLOW (CONTINUE?)
An internal table in **fsck** containing duplicate block numbers has no more room.

POSSIBLE FILE SIZE ERROR I=I
The i-node *I* size does not match the actual number of blocks used by the i-node. This is only a warning. If the **-q** option is used, this message is not printed.

DIRECTORY MISALIGNED I=I
The size of a directory i-node is not a multiple of 16. This is only a warning. If the **-q** option is used, this message is not printed.

PARTIALLY ALLOCATED INODE I=I (CLEAR?)
I-node *I* is neither allocated nor unallocated.

Phase 1B: Rescan for More DUPS

When a duplicate block is found in the file system, the file system is rescanned to find the i-node that previously claimed that block. When the duplicate block is found, the following information message is printed:

B DUP I=I
I-node *I* contains block number *B*, which is already claimed by another i-node. This error condition invokes the BAD/DUP error condition in Phase 2. I-nodes with overlapping blocks may be determined by examining this error condition and the DUP error condition in Phase 1.

Phase 2: Check Path Names

This phase removes directory entries pointing to bad i-nodes found in Phase 1 and Phase 1B. It reports error conditions resulting from the following:

- Root i-node mode and status
- Directory i-node pointers out of range
- Directory entries pointing to bad i-nodes

Types of Error Messages—Phase 2

Phase 2 has 4 types of error messages:

- Information messages
- Messages with a FIX? prompt
- Messages with a CONTINUE? prompt
- Messages with a REMOVE? prompt

Meaning of Yes/No Responses—Phase 2

- In Phase 2, an n(no) response to the FIX? prompt says:
  - Terminate the program since **fsck** will be unable to continue.
- In Phase 2, a y(yes) response to the FIX? prompt says:
  - Change the root i-node type to "directory."
  - If the root i-node data blocks are not directory blocks, a large number of error conditions are produced.
- In Phase 2, an n(no) response to the CONTINUE? prompt says:
  - Terminate the program.
- In Phase 2, a y(yes) response to the CONTINUE? prompt says:
  - Ignore DUPS/BAD error condition in root i-node and attempt to continue to run the file system check.

- If root i-node is not correct, then this may result in a large number of other error conditions.

- In Phase 2, an n(no) response to the REMOVE? prompt says:
  - Ignore the error condition.
  - A NO response is only appropriate if the user intends to take other measures to fix the problem.
- In Phase 2, a y(yes) response to the REMOVE? prompt says:
  - Remove duplicate or unallocated blocks.

Phase 2 Error Messages

ROOT INODE UNALLOCATED. TERMINATING
The root i-node (always i-node number 2) has no allocate mode bits. The occurrence of this error condition shows a serious problem. The program stops.

ROOT INODE NOT DIRECTORY (FIX?)
The root i-node (usually i-node number 2) is not a directory i-node type.

DUPS/BAD IN ROOT INODE (CONTINUE?)
Phase 1 or Phase 1B found duplicate blocks or bad blocks in the root -- node (usually i-node number 2) for the file system.

I OUT OF RANGE I=I NAME=F (REMOVE?)
A directory entry *F* has an i-node number *I* that is greater than the end of the i-node list.

UNALLOCATED I=I OWNER=O MODE=M SIZE=S MTIME=T NAME=F (REMOVE?)
A directory entry *F* has an i-node *I* without allocate mode bits. The owner *O*, mode *M*, size *S*, modification time *T*, and file name *F* are printed. If the file system is not mounted and the **-n** option was not specified, the entry is removed automatically if the i-node it points to is character size 0.

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 i-node *I*. The owner *O*, mode *M*, size *S*, modification time *T*, and directory name *F* are printed.

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*, i-node *I*. The owner *O*, mode *M*, size *S*, modification time *T*, and file name *F* are printed.

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 bad block was found in DIR i-node *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 i-node if the entire block looks bad or change (or remove) those directory entries that look bad.

Phase 3: Check Connectivity

This phase concerns itself with the directory connectivity seen in Phase 2. It reports error conditions resulting from the following:

- Unreferenced directories
- Missing or full **lost+found** directories.

Types of Error Messages—Phase 3

Phase 3 has 2 types of error messages:

1. Information messages
2. Messages with a RECONNECT? prompt.

Meaning of Yes/No Responses—Phase 3

- In Phase 3, an n(no) response to the RECONNECT? prompt says:
  - Ignore the error condition.
  - This invokes the UNREF error condition in Phase 4.

- A NO response is only appropriate if the user intends to take other measures to fix the problem.

- In Phase 3, a y(yes) response to the RECONNECT? prompt says:
  - Reconnect directory i-node *I* to the file system in directory for lost files (usually **lost+found**).
  - This may invoke a **lost+found** error condition if there are problems connecting directory i-node *I* to **lost+found**. This invokes CONNECTED information message if link was successful.

Phase 3 Error Messages

UNREF DIR I=I OWNER=O MODE=M SIZE=S MTIME=T (RECONNECT?)
The directory i-node *I* was not connected to a directory entry when the file system was traversed. The owner *O*, mode *M*, size *S*, and modification time *T* of directory i-node *I* are printed. The **fsck** program forces the reconnection of a nonempty directory.

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 in **lost+found**. This invokes the UNREF error condition in Phase 4. Possible problem with access modes of **lost+found**.

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 in **lost+found**. This invokes the UNREF error condition in Phase 4. Clean out unnecessary entries in **lost+found** or make **lost+found** larger (see Procedure 5.2, "Create File Systems on Hard Disk").

DIR I=I1 CONNECTED. PARENT WAS I=I2
This is an advisory message indicating a directory i-node *I1* was successfully connected to the **lost+found** directory. The parent i-node *I2* of the directory i-node *I1* is replaced by the i-node number of the **lost+found** directory.

Phase 4: Check Reference Counts

This phase checks the link count information seen in Phases 2 and 3. It reports error conditions resulting from the following:

- Unreferenced files
- Missing or full **lost+found** directory
- Incorrect link counts for files, directories, or special files
- Unreferenced files and directories
- Bad and duplicate blocks in files and directories
- Incorrect total free i-node counts.

Types of Error Messages—Phase 4

Phase 4 has 5 types of error messages:

1. Information 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

- In Phase 4, an n(no) response to the RECONNECT? prompt says:
  - Ignore this error condition.
  - This invokes a CLEAR error condition later in Phase 4.
- In Phase 4, a y(yes) response to the RECONNECT? prompt says:
  - Reconnect i-node *I* to file system in the directory for lost files (usually **lost+found**).
  - This can cause a **lost+found** error condition in this phase if there are problems connecting i-node *I* to **lost+found**.

- In Phase 4, an n(no) response to the CLEAR? prompt says:
  - Ignore the error condition.
  - A NO response is only appropriate if the user intends to take other measures to fix the problem.
- In Phase 4, a y(yes) response to the CLEAR? prompt says:
  - Deallocate the i-node by zeroing its contents.
- In Phase 4, an n(no) response to the ADJUST? prompt says:
  - Ignore the error condition.
  - A NO response is only appropriate if the user intends to take other measures to fix the problem.
- In Phase 4, a y(yes) response to the ADJUST? prompt says:
  - Replace link count of file i-node *I* with *Y*.
- In Phase 4, an n(no) response to the FIX? prompt says:
  - Ignore the error condition.
  - A NO response is only appropriate if the user intends to take other measures to fix the problem.
- In Phase 4, a y(yes) response to the FIX? prompt says:
  - Replace count in super-block by actual count.

Phase 4 Error Messages

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 modification time *T* of i-node *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.

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 in **lost+found**. This invokes the CLEAR error condition later in Phase 4. Possible problem with access modes of **lost+found**.

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 in **lost+found**. This invokes the CLEAR error condition later in Phase 4. Check size and contents of **lost+found**.

(CLEAR)

The i-node mentioned in the immediately previous UNREF error condition cannot be reconnected.

LINK COUNT FILE I=I OWNER=O MODE=M SIZE=S MTIME=T COUNT=X SHOULD BE Y (ADJUST?)

The link count for i-node *I*, which is a file, is *X* but should be *Y*. The owner *O*, mode *M*, size *S*, and modification time *T* are printed.

LINK COUNT DIR I=I OWNER=O MODE=M SIZE=S MTIME=T COUNT=X SHOULD BE Y (ADJUST?)

The link count for i-node *I*, which is a directory, is *X* but should be *Y*. The owner *O*, mode *M*, size *S*, and modification time *T* of directory i-node *I* are printed.

LINK COUNT F I=I OWNER=O MODE=M SIZE=S MTIME=T COUNT=X SHOULD BE Y (ADJUST?)

The link count for *F* i-node *I* is *X* but should be *Y*. The file name *F*, owner *O*, mode *M*, size *S*, and modification time *T* are printed.

UNREF FILE I=I OWNER=O MODE=M SIZE=S MTIME=T (CLEAR?)

I-node *I*, which is a file, was not connected to a directory entry when the file system was traversed. The owner *O*, mode *M*, size *S*, and modification time *T* of i-node *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.

UNREF DIR I=I OWNER=O MODE=M SIZE=S MTIME=T (CLEAR?)
I-node *I*, which is a directory, was not connected to a directory entry when the file system was traversed. The owner *O*, mode *M*, size *S*, and modification time *T* of i-node *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.

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 i-node *I*. The owner *O*, mode *M*, size *S*, and modification time *T* of i-node *I* are printed.

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 i-node *I*. The owner *O*, mode *M*, size *S*, and modification time *T* of i-node *I* are printed.

FREE INODE COUNT WRONG IN SUPERBLK (FIX?)
The actual count of the free i-nodes does not match the count in the super-block of the file system. If the **-q** option is specified, the count will be fixed automatically in the super-block.

Phase 5: Check Free List

This phase checks the free-block list. It reports error conditions resulting from the following:

- Bad blocks in the free-block list
- Bad free-block count
- Duplicate blocks in the free-block list
- Unused blocks from the file system not in the free-block list
- Total free-block count incorrect.

Types of Error Messages—Phase 5

Phase 5 has 4 types of error messages:

1. Information 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

- In Phase 5, an n(no) response to the CONTINUE? prompt says:
  - Terminate the program.
- In Phase 5, a y(yes) response to the CONTINUE? prompt says:
  - Ignore rest of the free-block list and continue execution of **fsck**.
  - This error condition will always invoke BAD BLKS IN FREE LIST error condition later in Phase 5.
- In Phase 5, an n(no) response to the FIX? prompt says:
  - Ignore the error condition.
  - A NO response is only appropriate if the user intends to take other measures to fix the problem.
- In Phase 5, a y(yes) response to the FIX? prompt says:
  - Replace count in super-block by actual count.
- In Phase 5, an n(no) response to the SALVAGE? prompt says:
  - Ignore the error condition.
  - A NO response is only appropriate if the user intends to take other measures to fix the problem.
- In Phase 5, a y(yes) response to the SALVAGE? prompt says:
  - Replace actual free-block list with a new free-block list.

- The new free-block list will be ordered according to the gap and cylinder specs of the **-s** or **-S** option to reduce time spent waiting for the disk to rotate into position.

Phase 5 Error Messages

EXCESSIVE BAD BLKS IN FREE LIST (CONTINUE?)
The free-block list contains more than a tolerable number (usually 10) of blocks with a value less than the first data block in the file system or greater than the last block in the file system.

EXCESSIVE DUP BLKS IN FREE LIST (CONTINUE?)
The free-block list contains more than a tolerable number (usually 10) of blocks claimed by i-nodes or earlier parts of the free-block list.

BAD FREEBLK COUNT
The count of free blocks in a free-list block is greater than 50 or less than 0. This error condition will always invoke the BAD FREE LIST condition later in Phase 5.

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 error condition will always invoke the BAD FREE LIST condition later in Phase 5.

X DUP BLKS IN FREE LIST
X blocks claimed by i-nodes or earlier parts of the free-list block were found in the free-block list. This error condition will always invoke the BAD FREE LIST condition later in Phase 5.

X BLK(S) MISSING
X blocks unused by the file system were not found in the free-block list. This error condition will always invoke the BAD FREE LIST condition later in Phase 5.

FREE BLK COUNT WRONG IN SUPERBLOCK (FIX?)
The actual count of free blocks does not match the count in the superblock of the file system.

BAD FREE LIST (SALVAGE?)

This message is always preceded by one or more of the Phase 5 information messages. If the **-q** option is specified, the free-block list will be salvaged automatically.

Phase 6: Salvage Free List

This phase reconstructs the free-block list. It has one possible error condition that results from bad blocks-per-cylinder and gap values.

Phase 6 Error Messages

DEFAULT FREE-BLOCK LIST SPACING ASSUMED

This is an advisory message indicating 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 values of 7 blocks-to-skip and 400 blocks-per-cylinder are used. Care must be taken to specify correct values with the **-sX** 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 status of the file system.

Cleanup Phase Messages

X files Y blocks Z free

This is an advisory message indicating that the file system checked contained *X* files using *Y* blocks leaving *Z* blocks free in the file system.

***** BOOT UNIX (NO SYNC!) *****

This is an advisory message indicating that a mounted file system or the root file system has been modified by **fsck**. If the UNIX System is not rebooted immediately without **sync**, the work done by **fsck** may be undone by the in-core copies of tables the UNIX System keeps. If the **-b** option of the **fsck** command was specified and the file system is **root**, a reboot is automatically done.

***** FILE SYSTEM WAS MODIFIED *****

This is an advisory message indicating that the current file system was modified by **fsck**.

Chapter 6: Performance Management

Introduction

This chapter describes ways to monitor and enhance the performance of your 3B2 computer system. The topics covered are listed below:

- General approach to performance management
- Ways to improve system performance
- Samples of how to locate the cause of poor system performance
- A sample system reconfiguration
- A description of the tools (commands) that are used to analyze system performance
- Descriptions of the tunable parameters.

General Approach to Performance Management

When you bring the computer up for the first time, the system is automatically set to a basic configuration that is satisfactory for most applications. This configuration, however, cannot take into account the usage patterns and the behavior of your particular applications. Therefore, you have the ability to reconfigure the system to ensure maximum performance for your particular application.

You will probably have to reconfigure the system only when you add more memory or peripherals. The possibility does exist, however, that as system usage increases you may start experiencing a deterioration in system performance. One sure indicator of poor system performance is slow response time on the console terminal. To determine precisely how your system is performing, you need to use the System Performance Analysis Tools which are described later in this chapter (most notably the **sar** command).

Note 1: The performance analysis tools are in the System Performance Analysis Utilities and optional utilities. You must install this package before you can use these tools.

Note 2: The date/time must be set for the System Performance Analysis Utilities to correctly operate.

If the performance analysis shows that your system needs tuning, there are several actions you can take:

- Reconfigure the operating system.

  This is usually referred to as tuning the kernel because you are adjusting the essential control structures in the kernel. There are two majors steps in reconfiguring the operating system:

  1. Change the tunable parameters.
  2. Rebuild the operating system.

- Uninstall optional kernel packages that are not needed by your applications.

  This procedure makes disk and memory space available for user programs and can benefit performance.

- Improve disk utilization.

  In addition to the allocation of system memory space defined by the tunable parameters, you have some control over how your file systems are organized on the disk and policies to cache frequently used programs in memory.

- Define the best system usage patterns.

  Finally, you can establish the model for best system usage, such as encouraging users to run large noninteractive programs at night.

These areas are discussed in more detail in the remaining sections of the chapter.

Improving Performance

Modifying the Tunable Parameters

The setting of the core system tunable parameters is done by editing the parameter entries in the **/etc/master.d/kernel** file, or other files in **/etc/master.d**, such as **shm**, **msg**, **sem**, or **disp**. For complete definitions of the individual tunable parameters and suggestions about setting them, refer to the section "Tunable Parameters." (See Figure 6-5 for the recommended initial values for the tunable parameters relative to a particular memory size.) Use the **/etc/sysdef** command to determine the current values of the tunable parameters are in the present configuration of your system.

Generally, the default parameters for your configuration will result in acceptable performance. (Default parameters are defined in Figure 6-5.) If, however, you are running an application that has special performance needs, you can use the tools described in the section "Performance Tools" to measure system load and determine the parameters that might be changed to improve performance. For example, two key parameters having a strong affect on the efficiency of disk use are NBUF (system buffers) and NHBUF (hash buffers). These buffers are discussed in the next section, "Improving Disk Utilization."

See Procedure 6.1, "Reconfiguring the System," for examples of editing the tunables and reconfiguring the system. Also, look at the end of this section for a sample of a typical reconfiguration.

Improving Disk Utilization

Disk input/output may cause a bottleneck in system performance. Here are some ways to improve the performance of the disk system.

- Allocate a larger buffer cache
- Set the text-bit (sticky-bit) for frequently used executables
- Organize the file systems to minimize disk activity
- Set the logical block size to suit the application
- Expand hardware configuration.

Size of the Buffer Cache

The NBUF parameter specifies the number of 2KB buffers in the system buffer cache. These buffers are used for file systems whose logical block size is 512 bytes, 1024 bytes or 2048 bytes. These buffers, which are in main memory, hold recently-used data on the chance that it will be needed again. If a read or a write can be satisfied using the buffer cache instead of the disk, system performance improves, because memory operations are much faster than disk operations. NHBUF specifies the number of hashing buckets in the buffer cache. The more buffers, the greater chance that data can be found in the buffers without the system having to do a time-consuming disk read. The read and write buffer cache hit ratios listed by the **sar -b** command shows how effective the system buffers are. If the value for NBUF is left null, the system calculates values based on the memory size shown in Figure 6-5. The value for NHBUF is a power of 2 that is at least 50% of the value of NBUF; they can be specified in **/etc/master.d/kernel**.

The values of NBUF and NHBUF given in Figure 6-5 are a good starting place. These values come close to optimum for most system workloads. Increasing NBUF and NHBUF, up to a point, improves system performance.

A general "rule of thumb" is that the size of the buffer cache should be 12 to 15 percent of main memory. Therefore, about 600 KB of memory can be devoted to buffers on a 4-MB system and 3.4 MB on a 16-MB system. If the buffer cache becomes too large, there may not be enough memory space for efficient operation of user processes, and the amount of swapping done by the system will increase. The swapping activity usually costs more in system efficiency than is gained by having a large amount of buffer space. If the

sar -w command shows that your system has **swpot/s** greater than 1.0, adding buffers may not be beneficial.

After the UNIX system has run for a day or so, check for excessive swapping activity. If excessive activity is found, there are two ways to correct the problem:

1. Reduce the number of buffers (NBUF and NHBUF).
2. Increase your system memory.

The preferred method is to increase system memory.

Set Text-Bit (Sticky-Bits)

Setting the text-bit can reduce the disk traffic of a select group of commands. Text pages of sticky commands are kept resident in memory, even when the process ends. Once loaded into memory, such pages will usually remain. One exception is if the pages are reclaimed by the paging daemon in a tight memory situation (that is, when the number of available pages falls below the low-water mark as defined by the tunable parameter GPGSLO). Finding sticky pages already in memory can reduce considerably the loading time for the text pages of a process.

On systems that usually run a light to medium workload and are seldom short of memory, setting the text-bit can cause a significant improvement in performance. Be selective in setting the text-bit on a system with limited memory or a heavy workload. If you are in doubt about using the text-bit,

use the following guideline: use the text-bit if the average amount of free memory is greater than the high-water mark (GPGSHI) plus 100.

The average amount of free memory should be determined by setting the text-bit on some test commands with the **chmod**(1) command, and by using the **-r** option of **sar**(1) to determine the average FREEMEM count over a typical interval of system activity. If the **sar** report shows that it is safe to set the text-bit on for some commands, logical candidates would be frequently used, preferably, small commands. If the average free memory for the interval is less than the high-water mark plus 100, then performance is not likely to be significantly improved and may be hurt by setting the text-bit.

File System Organization

This section describes several actions that can be taken to reduce the overhead of file access. As file systems are used, the blocks of individual member files tend to become physically scattered around the disk(s) and I/O becomes less efficient. This scattering yields poor ordering of blocks with files and poor directory structure.

Organization of File System Free List

The 3B2 computer file systems are set up to allocate free blocks in a way that allows the files to be read or written with efficiency. A free list array is created when a file system is created with **/etc/mkfs**(1M). The free list is set up with the rotational gap specified by **mkfs** options. On the 3B2 computer the rotational gap can be explicitly specified on the **mkfs** command line. See the "Using mkfs" section of Chapter 5, "File System Administration," for recommended values for the rotational gap. The difference between successive block numbers in the free list is the rotational gap. For example, a file created on a system with a rotational gap of 10 may consist of blocks 510, 520, and 530. When the file is read, I/O requests are sent to the disk drive to read blocks 510, 520, and 530. As soon as the drive finishes reading block 510 and starts to process the second request, block 520 will have passed beyond the read/write head just as the drive is ready to read that request. This method makes for efficient I/O operation.

However, as you start changing files (changing size or removing), the efficiency starts to decrease. When several files are being created at once, they will be contending for blocks from the free list. Some of the blocks allocated to the files will be out of sequence. As you can see, the free list also becomes scattered about the disk as blocks are allocated and freed.

Directory Organization

Directory organization also affects input/output performance. The problems show up when files are removed by users. When a file is removed from a directory, the i-node number is freed. This leaves an unused slot for that i-node; over a period of time the number of empty slots may become large. 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 (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.

Restore Good File System Organization

There is no automatic way to solve these problems; however, you can manually rearrange the file system. Here are a variety of ways to do this. Note that the file system(s) must always be unmounted.

1. To reorganize the free list, run **fsck**(1M) **-s**.
2. To reorganize particular directory structures, use **cpio**(1) **-pdm** to copy them to a temporary location; remove the original structure; then use **cpio -pdm** to copy them back to their original location.
3. To reorganize the file system, run **dcopy**(1M) **-s**. This is probably the most comprehensive way to handle file system reorganization. The **dcopy** program performs Steps 1 and 2 above, and it also provides the opportunity to change the file system and i-node list sizes.
4. Run **sysadm compress**(1) or **cmpress**(1M) to reorganize the file system. This procedure copies the file system temporarily to the cartridge tape and then back to its original location.

5. If you have directories that require file system indirection, consider dividing them into smaller directories. Directories require indirection if they take more than 10 logical blocks. The following command finds such directories:

   find / -type d -size +10 -print

   A directory entry takes 16 bytes, so a directory in a 1KB file system requires indirection if it has more than 640 entries ((1024/16) *10).

6. If you have more than one disk, balance heavily used file systems across the disks. You can determine how often each disk is being used with the **sar -d** command. Use the output of this command to decide whether to relocate file systems.

Logical Block Size

If you have a file system with many large files, you may want to use a larger logical block size on that file system. Logical block size is the size of the chunks the UNIX system kernel uses to read or write files.

As logical block size goes up, disk performance tends to improve, because when the system does large I/O operations, it does fewer of them—CPU overhead and disk seek time are reduced. On the other hand, as logical block size goes up, space tends to be wasted, both in primary memory and on disk. The extra space is lost to "fragmentation." Fragmentation is the space wasted by a data storage method. For example, if files are stored on disk and in memory in chunks of 1024 bytes, then a 24-byte file leaves 1000 bytes unused on the disk. When the file is accessed, 2024 bytes are unused in the kernel buffer space in primary memory. If a 24-byte file is stored in 2048-byte chunks, 2024 bytes are unused.

Use the **sar -u** command, described later in this chapter, to determine if you have a disk bottleneck. If you also have a CPU bottleneck, a larger logical block size is not likely to help much. But if you do have some idle CPU time, and the **sar** command reports that your system is spending too much time waiting for I/O to complete, you may improve system performance by converting some file systems to a larger block size. See the section "Administering the File System" in Chapter 5, "File System Administration," for more information.

Defining Best System Usage Patterns

After the kernel and the system activities are tuned and the file systems organized, the next step for improving system performance is to do some housekeeping activities and to check whether prime time load can be reduced. The person responsible for administering the system should check for the following:

- Less important jobs interfering with more important jobs
- Unnecessary activities being carried out
- Scheduling selected jobs for when the system is not so busy
- Efficiency of user-defined features, such as **.profile** and $PATH.

ps Command

The **ps**(1) command is used to obtain information about active processes. The command gives a "snapshot" picture of what is going on, which is useful when you are trying to identify what processes are loading the system. Things will probably change by the time the output appears; however, the entries that you should be interested in are TIME (minutes and seconds of CPU time used by processes) and STIME (time when process first started).

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 stopped immediately via the **kill**(1) **-9** 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 long time to execute, you should consider using **cron**(1M) or **at**(1) to execute the job during off-hours.

User $PATH Variables

$PATH is searched on each command execution. Before outputting "not found," the system must search every directory in $PATH. These searches require both processor and disk time. If there is a disk or processor bottleneck, changes here can help performance.

Some things that you should check for in user $PATH variables are:

- Path Efficiency

  $PATH is read left to right, so the most likely places to find the command should be first in the path (**/bin** and **/usr/bin**). Make sure that a directory is not searched more than once for a command.

- Convenience and Human Factors

  Users may prefer to have the current directory listed first in the path (**:/bin**).

- Path Length

  In general, $PATH should have the least number of required entries.

- Large Directory Searches

  Searches of large directories (ones with many files) should be avoided if possible. Put any large directories other than **/bin** and **/usr/bin** at the end of $PATH.

Samples of General Procedures

This section depicts typical approaches to performance management. First, it describes a general procedure for troubleshooting performance problems. Then, it shows a sample of a typical system reconfiguration.

Note: The performance analysis tools discussed in this section—**sar**, **sag**, **sadp**, and **timex**—are in the System Performance Analysis Utilities. You must install this package before you can use these tools.

Sample Procedure for Investigating Performance Problems

Locating the source of the problem can require some careful detective work. Hence, what follows is not a canned procedure, but rather a sample of a typical approach, covering basic areas where problems usually surface, and suggesting some of the actions to take that will alleviate the problem. The most common symptom that a problem exists is consistently poor response time. Refer to Figure 6-1 for an outline of the approach. Note that if you have identified a familiar problem area, see Procedure 6.1, "Reconfiguring the System," to make the necessary system parameter changes.

Check for Excess Page Swapping

Since the swapping of pages is costly in both disk and CPU overhead, the first thing to look at is swapping activity. Get the **sar**(1) **-qw** report. Look at the percentage that the swap queue is occupied (%swpocc) for values greater than 5. Then look at the swap-out rate (swpot/s) for values greater than 1.00.

Check whether the "freemem" count (number of pages available to user programs), shown by **sar -r**, is consistently less than the value of the tunable parameter GPGSHI (high-water mark).

If any or all of these are happening frequently, increase your system memory or check if you can reduce memory allocated for system buffers. (Increasing system memory is preferable.) If a large number of buffers have been configured and the buffer cache hit ratios (**sar -b**) are 90 percent or more, try decreasing the number of buffers (stored in the tunable parameter NBUF). It is possible that returning memory space occupied by some buffers will solve the swapping problem by leaving more space for user programs. Additional memory for user programs can also be obtained by uninstalling optional utilities that are not needed by your applications.

Check for Disk Bottleneck

If the value of %wio (from the **sar -u** report) 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 bottleneck. Some ways to alleviate a disk bottleneck are listed below:

1. Increase the number of buffers.
2. Organize the file system to minimize disk activity.
3. Consider distributing file systems across two or more disks for a more balanced load. Balancing the load may require adding more disks to your system.
4. Set the text-bit for frequently used files. See the earlier discussion of this subject under "Setting Text-Bit (Sticky-Bit)."
5. Consider adding more memory if the situation persists. Additional memory reduces swapping/paging traffic, allows you to add more buffers, and allows you to be more liberal in your use of the text-bit.

Check for Modem Interrupts

Run **sar -y** to get a report of 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. Repair it.

Check for Potential 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, file, or i-node tables. Overflows in these tables are avoided by increasing NPROC (only gets reported if all slots are full, the last slot is for root), NFILE, and NINODE/NS5INODE (in the **/etc/master.d/kernel** file).

Shift Workload to Off-Peak Hours

Examine **/usr/spool/crontab** to determine 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**(1) or **troff**(1)] at off-peak hours. You may also want to run such commands with a low priority by using the **nice**(1) or **batch**(1) commands.

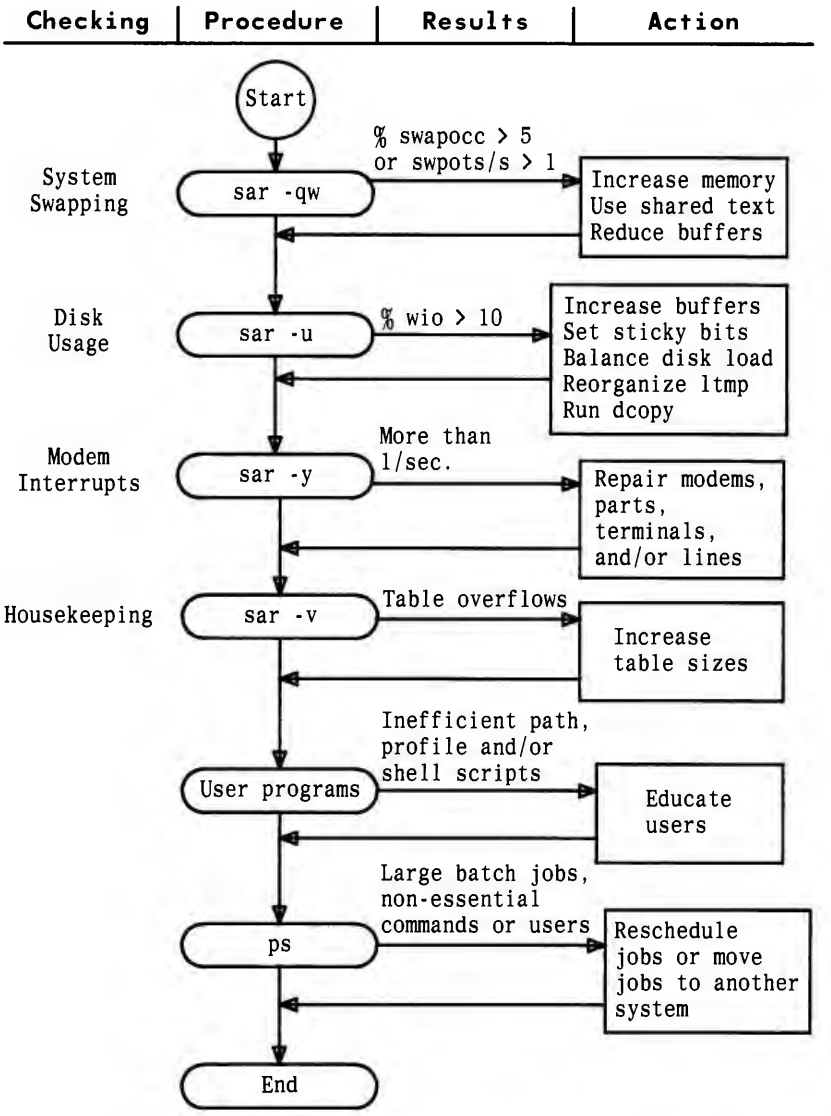


Figure 6-1: Outline of Typical Troubleshooting Procedure

Sample System Reconfiguration

The following is a typical scenario for reconfiguring the system because of a hardware upgrade from 8 MBs to 12 MBs of memory.

Because of the additional memory, many tunable parameters may be increased. (See Figure 6-5 for parameter values recommended for your system.)

Note: Critical tunable parameters are automatically changed by the system (autotuned) based on equipped RAM or I/O cards.

Many of the parameters are specified in the **/etc/master.d/kernel** file. The command line entries and system responses in the display below show the reconfiguration and rebooting of the operating system to support these new parameters. The editing of the **/etc/master.d/kernel** file is not shown.

The illustration shows that tunable parameters for semaphores are also being modified; the **/etc/master.d/sem** file is edited and **mkboot** is run to convert SEM into a bootable object file.

```
# cp 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 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.*

```
# cd
# sysadm firmware
```

(A series of messages are displayed ending with the following:)

```
SELF-CHECK

FIRMWARE MODE
<mcp><CR>       (Enter firmware password.)

Enter name of program to execute [ ]: /etc/system
            Possible load devices are:

Option Number    Slot   Type         Name
-------------------------------------
       0          0    INTEGRAL     FD5
       1          1    I/O BUS      SCSI
```

Continued

Continued from previous screen display

```
Enter Load Device Option Number [1 (SCSI)]: <CR>
Possible subdevices are:

Option Number     Subdevice      Name
-----------------------------------------
       0             0          disk
       1             1          tape

Enter Subdevice Option Number [0 (disk)]: <CR>

CONFIGURATION SUMMARY
=====================
     ----driver---- #devices major
     PRF                 1     33
     MIRROR              1     32
     XT                  1     31
     SXT                 1     30
     EPORTS              3      2,   3,    4
     ST01                1    122
     SD00                1    121
     SCSI                1      1
     PIR                 1      0
     OSM                 1     44
     MEM                 1     19
     MAU                 1
     IUART               1      0
     IDISK               1     17
     HDELOG              1     16
     GENTTY              1     20
     CONLOG              1     45
```

Continued

Continued from previous screen display

Note: Not all of the configuration summary and load map are shown.

```
UNIX(R) System V Release 3.2.2 AT&T 3B2 Version 3
Node unix
Total real memory   = 16777216
Available memory    = 12120064
*******************************************************************************

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.
```

File systems are checked.

```
mount /dev/dsk/c1t1d0s8  /usr2
mount /dev/dsk/c1t1d1s2  /usr
Generating a new /unix
AT&T 3B2 SYSTEM CONFIGURATION:

Memory size: 16 Megabytes
System Peripherals:
        Device Name         Subdevices        Extended Subdevices

        SBD
                          Floppy Disk
        SCSI
        (S.E. Bus ID0)
                          SD01 ID1
                                            317 Megabyte Disk
                                            317 Megabyte Disk
                          ST01 ID2
                                            Tape ID0
        EPORTS
        EPORTS
        EPORTS
        MAU

The system is ready.

Console Login:
```

Performance Tools

Internal activity is measured by a number of counters contained in the UNIX system kernel. Each time an operation is performed, an associated counter is incremented. The **sar** command and the other performance tools allow you to monitor the values of these counters. The functions monitored by **sar** are discussed in the following sections. The performance tools for the UNIX system internal activities described in this section are listed below:

sar	Samples cumulative activity counters internal to the UNIX system and reports on various systemwide activities.
sag	Graphically displays the information collected by **sar**.
sadp	Produces profiles of disk access location and seek distance.
timex	Reports both system-wide and per-process activity during the execution of a command or program.

These tools are part of the System Performance Analysis Utilities. You must install this package before you can use the tools. Examples for these tools are given in the following sections. The examples are from a system with 4 MBs of main memory and two 155-MB integral hard disks. Command outputs are typical values observed for user workloads on the UNIX operating system. Values you receive may be different from values in the examples, depending on your application, configuration, or benchmark. When tuning your system, it is recommended that you use a benchmark or have the system under normal load for your application to allow you to tune directly toward your specific application.

sar Command

Throughout this section, **sar** options are described with an analysis of sample outputs of the options. The **sar** options can be used either to gather system activity data or to extract what has been collected in data files created by **sa1** and **sa2**. Data files **sa1** and **sa2** are started by entries put in the **/usr/spool/cron/crontab/sys** file.

sar -a Command

The **sar -a** option reports the use of file access operations. The UNIX operating system routines reported are as follows:

iget/s Number of files located by i-node entry per second.

namei/s Number of file system path searches per second. The **namei** routine calls **iget**, so **iget/s** is always larger than **namei/s**.

dirbk/s Number of directory block reads issued per second.

An example of **sar -a** output, with a 30-second sampling interval, follows:

```
unix unix 3.2.2 3 3B2    04/03/88

12:41:40  iget/s namei/s dirbk/s
12:42:10     44     34       1
12:42:40     42     34       1
12:43:10     43     34       1
12:43:40     46     36       1

Average      44     34       1
```

The larger the values reported, the more time the UNIX system kernel is spending to access user files. This shows how heavily programs and applications are using the file system(s). The **-a** option is helpful for understanding how disk-dependent the application system is; it is not used for any specific tuning step.

sar -b Command

The **-b** option reports the following buffer activity.

bread/s Average number of physical blocks read into the system buffers from the disk (or other block devices) per second.

lread/s Average number of logical blocks read from system buffers per second.

%rcache	Fraction of logical reads found in buffer cache (100 percent minus the ratio of **breads** to **lreads**).
bwrit/s	Average number of physical blocks written from the system buffers to disk (or other block devices) per second.
lwrit/s	Average number of logical blocks written to system buffers per second.
%wcache	Fraction of logical writes found in buffer cache (100 percent minus the ratio of **bwrit/s** to **lwrit/s**).
pread/s	Average number of physical read requests per second.
pwrit/s	Average number of physical write requests per second.

The entries that you should be most interested in are the cache hit ratios, *%rcache* and *%wcache* which measure the effectiveness of system buffering. If *%rcache* falls below 90 or *%wcache* falls below 65, it may be possible to improve performance by increasing the number of buffers.

An example of **sar -b** output follows:

```
unix unix 3.2.2 3 3B2    04/03/88

16:32:57 bread/s lread/s %rcache bwrit/s lwrit/s %wcache pread/s pwrit/s
16:33:07       2      97      98       2      22      90       0       0
16:33:17       1      91      98       2      23      90       0       0
16:33:27       2      97      98       2      23      90       0       0
16:33:37       1     100      99       2      25      90       0       0

Average        2      96      98       2      23      90       0       0
```

This example shows that the buffers are not causing any bottlenecks, because all data is within acceptable limits.

sar -c Command

The **-c** option reports system calls in the following categories:

scall/s	All types of system calls per second, generally about 30 per second on a busy 25 to 30 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 10 to 20 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 $PATHs.]
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 this varies greatly with the activities that are being performed by the system.

The following is an example of **sar -c** output:

```
unix unix 3.2.2 3 3B2    04/03/88

18:33:04 scall/s sread/s swrit/s fork/s exec/s rchar/s wchar/s
18:33:14      59      17       4   0.42   0.70    4466     275
18:33:24      85      29      10   0.59   0.82    2146     710
18:33:34      69      35      10   0.59   0.97    8133    1135
18:33:44      85      28      11   0.49   0.82    6828    2077

Average       75      27       9   0.52   0.83    5393    1049
```

A parallel option (**sar -Dc**) is available for systems that have Remote File Sharing installed.

sar -d Command

The **sar -d** option reports the activity of block devices.

device	Name of the block device(s) that **sar** is monitoring.
%busy	Percent of time the device was servicing a transfer request.
avque	The average number of requests outstanding during the period of time (measured only when the queue is 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 idle 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 is an example of a **sar -d**:

```
unix unix 3.2.2 3 3B2    04/03/88

13:46:28  device %busy avque r+w/s blks/s  avwait  avserv

13:46:38 sd01-0     2   1.0     1      3     0.0    17.9
         sd01-1     6   1.1     3      5     2.0    23.9

13:46:48 sd01-0     2   1.0     1      2     0.0    19.6
         sd01-1     6   1.0     3      5     0.2    24.3

13:47:08 sd01-0     3   1.0     1      3     0.3    18.3
         sd01-1     7   1.1     3      5     1.3    25.4

13:47:18 sd01-0     3   1.0     1      3     0.0    17.2
         sd01-1     5   1.0     2      5     0.0    21.6

Average  sd01-0     2   1.0     1      3     0.1    18.2
         sd01-1     6   1.0     3      5     0.9    23.8
```

Note that queue lengths and wait times are measured while the queue had something on it. 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.

sar -m Command

The **sar -m** option reports on Inter-Process Communication (IPC) 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.

The following is an example of a **sar -m** output:

```
unix unix 3.2.2 3 3B2    04/03/88

15:16:04    msg/s  sema/s
15:16:14    0.00    0.00
15:16:24    0.00    0.00
15:16:34    0.00    0.00
15:16:44    0.00    0.00

Average     0.00    0.00
```

These figures will usually be zero (0.00) unless you are running applications that use the message or semaphore features.

sar -q Command

The **sar -q** option reports the average queue length while the queue is occupied and the percent of time it is occupied.

runq-sz	Run queue of processes in memory; typically, this should be less than two. 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	Swap queue of processes to be swapped out; the smaller this number, the better.
%swpocc	The percentage of time the swap queue is occupied; the smaller this value, the better.

The following is an example of a **sar -q** output:

```
unix unix 3.2.2 3 3B2    04/03/88

11:01:04 runq-sz %runocc swpq-sz %swpocc
11:01:14    1.4      49
11:01:24    1.6      55
11:01:34    2.3      61
11:01:44    1.9      61

Average     1.8      57
```

In this example, the processor use (%runocc) varies between 58 percent and 98 percent, while the fraction of time the swap queue is not empty (%swpocc) is 31 percent to 49 percent. This means that memory is not causing a major bottleneck in the system throughput, but more memory would help reduce the swapping/paging activity.

If %runocc is greater than 90 and runq-sz is greater than 10, the CPU is heavily loaded and response is degraded. In this case, additional CPU capacity may be required to obtain acceptable system response. If %swpocc is greater than 20, more memory or fewer buffers would help reduce swapping/paging activity.

sar -u command

The CPU usage is listed by **sar -u** (default). At any given moment the processor will be either busy or idle. When busy, the processor will be in either user or system mode. When idle, the processor will either be waiting for input/output completion or has no work to do. The **-u** option of **sar** lists the percent of time that the processor is in system mode (%sys), user mode (%usr), waiting for input/output completion (%wio), and idle time (%idle).

In typical timesharing use, %sys and *%usr* are about the same value. In special applications, either of these may be larger than the other without anything being abnormal. A high *%wio* (greater than 10 percent) generally means a disk bottleneck. A high *%idle*, with degraded response time, may mean memory constraints; time spent waiting for memory is attributed to %idle.

The following is an example of a **sar -u** output:

```
unix unix 3.2.2 3 3B2     04/03/88

09:20:08        %usr      %sys      %wio     %idle
09:20:18          40        20         2        38
09:20:28          43        20         1        36
09:20:38          43        21         3        33
09:20:48          46        24         2        28

Average           43        21         2        34
```

If your 3B2 computer is equipped with the Multiprocessor Enhancement feature, **sar -u** produces additional output for each additional processor, including an extra column showing the number of system calls per second being executed on the second processor. The %wio column is empty for these processors because they do not do I/O. The following **sar** data was collected on a system equipped with the Multiprocessor Enhancement feature at intervals of 20 seconds.

```
sf600 sf600 3.2.2 3 3B2    04/12/88

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 (**sar -Du**) is available for systems that have Remote File Sharing installed.

sar -v Command

The **-v** option reports the status of process, i-node, file, shared memory record, and shared memory file tables. From this report you know when the system tables need to be modified.

proc-sz — Number of process table entries now being used/allocated in the kernel.

inod-sz — Number of i-node table entries now being used/allocated in the kernel.

file-sz — Number of file table entries now being used/allocated in the kernel.

ov — Number of times an overflow occurred. (One column for each of the above three items.)

lock-sz — The number of shared memory record table entries now being used/allocated in the kernel.

The values are given as level/table size. An example of **sar -v** follows:

```
unix unix 3.2.2 3 3B2    04/03/88

17:37:05 proc-sz ov   inod-sz ov   file-sz ov   lock-sz
17:37:15  41/120  0    88/300  0    72/300  0    0/100
17:37:25  41/120  0    89/300  0    69/300  0    0/100
17:37:35  41/120  0    88/300  0    67/300  0    0/100
17:37:45  41/120  0    89/300  0    70/300  0    0/100
```

This example shows that all tables are large enough to have no overflows. Sizes could be reduced to save main memory space if these are the highest values ever recorded.

sar -w Command

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, memory may need to be increased or buffers decreased.
bswot/s	Number of blocks transferred for swap-outs per second.
pswch/s	Process switches per second. This should be 30 to 50 on a busy 25- to 30-user system.

An example of **sar -w** output follows:

```
unix unix 3.2.2 3 3B2    04/03/88

19:53:04 swpin/s bswin/s swpot/s bswot/s pswch/s
19:53:14    0.0     0.0     0.0     0.0      32
19:53:34    0.0     0.0     0.0     0.0      31
19:53:44    0.0     0.0     0.0     0.0      28
19:53:54    0.0     0.0     0.0     0.0      27

Average     0.0     0.0     0.0     0.0      30
```

This example shows that there is enough memory for the currently active users, since no swapping is occurring.

sar -p Command

The **-p** option reports paging activity. The following page rates are recorded.

vflt/s Number of address translation page faults per second (valid page not present in memory).

pflt/s Number of page faults from protection errors per second (illegal access to page) or "copy-on-writes." **pflt/s** generally consists entirely of "copy-on-writes."

pgfil/s Number of **vflt/s** per second satisfied by a page-in from the file system. (Each **pgfil** causes two **lreads**; see **sar -b**).

rclm/s Number of valid pages per second that the system has reclaimed (added to list of free pages).

An example of **sar -p** output follows:

```
unix unix 3.2.2 3 3B2   04/03/88

12:56:04 vflt/s  pflt/s  pgfil/s  rclm/s
12:56:14  23.57   33.92    1.27     0.0
12:56:24  21.55   32.05    1.40     0.0
12:56:34  24.70   32.35    1.70     0.0
12:56:44  22.85   34.20    1.45     0.0

Average   23.16   33.12    1.46     0.0
```

sar -r Command

The **-r** option records the number of memory pages and swap file disk blocks that are currently unused. The following are recorded.

freemem Average number of 2KB pages of memory available to user processes over the intervals sampled by the command.

freeswp Number of 512-byte disk blocks available for process swapping.

An example of **sar -r** output follows:

```
unix unix 3.2.2 3 3B2   04/03/88

12:56:04 freemem freeswp
12:56:14    2192    20640
12:56:24    2204    20640
12:56:34    2197    20640
12:56:44    2169    20640

Average     2190    20640
```

sar -y Command

The **-y** option monitors terminal device activities. If you have much terminal I/O, you can use this report to determine if there are any bad lines. 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 unix 3.2.2 3 3B2    04/03/88

16:50:11 rawch/s canch/s outch/s rcvin/s xmtin/s mdmin/s
16:50:21     0       0       4       0       0       0
16:50:31     0       0       2       0       0       0
16:50:41     0       0       2       0       0       0
16:50:51     0       0       2       0       0       0

Average      0       0       3       0       0       0
```

sar -A Command

The **sar -A** option is equivalent to **sar -udbycwaqvmprCDS**. The report includes RFS operations (the **-D** and **-S** options). The **-A** option gives 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** follows:

```
unix unix 3.2.2 3 3B2    04/03/88

09:30:29    %usr    %sys    %sys    %wio   %idle
                    local  remote
09:30:59      44      49       0       7       0
09:31:29      37      40       0       9      14
09:31:59      37      40       0       7      16
09:32:29      21      23       0       5      50
Average       35      38       0       7      20
```

Continued

Continued from previous screen display

```
09:30:29   device   %busy   avque   r+w/s  blks/s  avwait  avserv
09:30:59   sd01-0     15     2.7       6      13    40.2    23.0
09:31:29   sd01-0     14     1.9       6      12    20.4    23.1
09:31:59   sd01-0     12     1.5       5      11    10.5    22.3
09:32:29   sd01-0      9     1.3       4       8     7.6    22.2
Average    sd01-0     12     1.9       5      11    21.4    22.7

09:30:29  runq-sz  %runocc  swpq-sz  %swpocc
09:30:59      1.9       37
09:31:29      1.8       20
09:31:59      1.0       13
09:32:29      1.2       17
Average       1.6       22

09:30:29 bread/s lread/s %rcache bwrit/s lwrit/s %wcache pread/s pwrit/s
09:30:59
   local        3      265      99       4      30      87       0       0
   remote       0        0       0       0       0       0       0       0
09:31:29
   local        3      217      99       3      27      88       0       0
   remote       0        0       0       0       0       0       0       0
09:31:59
   local        3      218      99       3      26      89       0       0
   remote       0        0       0       0       0       0       0       0
09:32:29
   local        2      126      99       2      15      86       0       0
   remote       0        0       0       0       0       0       0       0
Average
   local        2      206      99       3      24      88       0       0
   remote       0        0       0       0       0       0       0       0

09:30:29  swpin/s bswin/s swpot/s bswot/s pswch/s
09:30:59     0.00     0.0    0.00     0.0      14
09:31:29     0.00     0.0    0.00     0.0      13
09:31:59     0.00     0.0    0.00     0.0      12
09:32:29     0.00     0.0    0.00     0.0       9
Average      0.00     0.0    0.00     0.0      12
```

Continued

Continued from previous screen display

```
09:30:29 scall/s sread/s swrit/s  fork/s  exec/s rchar/s wchar/s
09:30:59
   in          0       0       0              0.00       0       0
   out         0       0       0              0.00       0       0
   local     489     196      10    1.86      1.86   55308    8256
09:31:29
   in          0       0       0              0.00       0       0
   out         0       0       0              0.00       0       0
   local     413     173       8    1.83      1.83   48373    6648
09:31:59
   in          0       0       0              0.00       0       0
   out         0       0       0              0.00       0       0
   local     400     159       8    1.67      1.67   45418    6733
09:32:29
   in          0       0       0              0.00       0       0
   out         0       0       0              0.00       0       0
   local     238      99       5    1.17      1.17   28681    3636
Average
   in          0       0       0              0.00       0       0
   out         0       0       0              0.00       0       0
   local     384     156       8    1.63      1.63   44403    6311

09:30:29  iget/s namei/s dirbk/s
09:30:59     280     146       1
09:31:29     214     112       1
09:31:59     226     119       1
09:31:29     123      65       1
Average      211     111       1

09:30:29 rawch/s canch/s outch/s rcvin/s xmtin/s mdmin/s
09:30:59       0       0      13       0       1       0
09:31:29       0       1      11       0       1       0
09:31:59       0       1      12       0       1       0
09:32:29       0       0       7       0       1       0
Average        0       0      11       0       1       0
```

Continued

Continued from previous screen display

```
09:30:29 proc-sz ov inod-sz ov file-sz ov lock-sz
09:30:59  14/200  0  38/600  0  27/600  0   0/100
09:31:29  14/200  0  35/600  0  29/600  0   0/100
09:31:59  14/200  0  38/600  0  27/600  0   0/100
09:32:29  13/200  0  33/600  0  23/600  0   0/100

09:30:29    msg/s  sema/s
09:30:59     0.00    0.00
09:31:29     0.00    0.00
09:31:59     0.00    0.00
09:32:29     0.00    0.00
Average      0.00    0.00

09:30:29  vflt/s  pflt/s pgfil/s  rclm/s
09:30:59   34.62   25.28    1.52    0.00
09:31:29   30.17   24.67    1.70    0.00
09:31:59   29.23   21.97    1.43    0.00
09:32:29   17.76   15.29    0.97    0.00
Average    27.92   21.79    1.41    0.00

09:30:29 freemem freeswp
09:30:59    2577   20640
09:31:29    2625   20640
09:31:59    2631   20640
09:32:29    2674   20640
Average     2627   20640

09:30:29 snd-inv/s snd-msg/s rcv-inv/s rcv-msg/s dis-bread/s blk-inv/s
09:30:59       0.0        0.0        0.0        0.0         0.0       0.0
09:31:29       0.0        0.0        0.0        0.0         0.0       0.0
09:31:59       0.0        0.0        0.0        0.0         0.0       0.0
09:32:29       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

09:30:29 serv/lo-hi    request   request    server    server
            3 - 6       %busy  avg lgth   %avail avg avail
01:00:00        0         0.0         0      0.0         0
01:00:00        0         0.0         0      0.0         0
01:00:00        0         0.0         0      0.0         0
01:00:00        0         0.0         0      0.0         0
Average         0           0         0        0         0
```

sag Command

The **sag** command graphically displays the system activity data stored in a binary data file by a previous **sar** run. Any of the **sar** data items may be plotted separately or in combination. The **sag** command invokes **sar** and matches strings in the data column header. Figure 6-2 shows a typical **sag** display. Running **sar** will show what data is available.

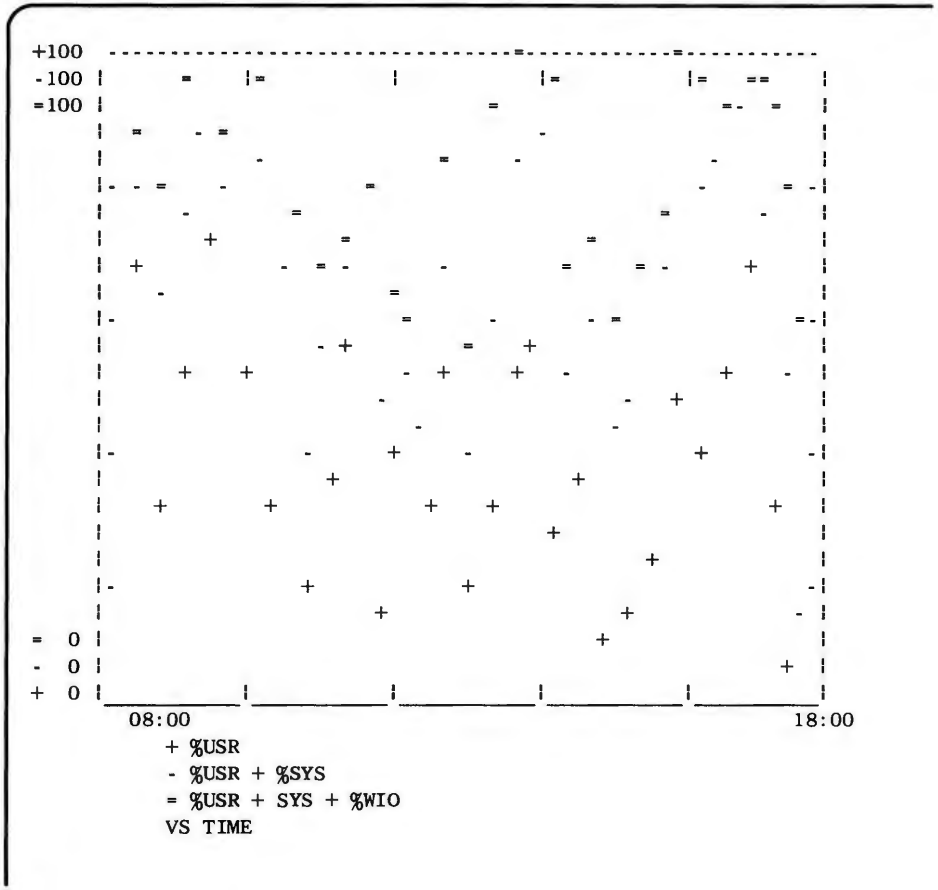


Figure 6-2: Example of **sag** Output

In Figure 6-2, 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 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. From this example, the first place to look to reduce any bottleneck is in reducing processor load.

The **sag** command is useful only if you have a standard output device that can read plotting instructions. Refer to **tplot**(1G) in the *User's and System Administrator's Reference Manual* for a list of terminals with this capability.

timex Command

The **timex** command times a command and reports the system activities that occurred during the time the command was executing. If no other programs are running, then **timex** can give you a good idea of which resources a specific command uses during its execution. System consumption can be collected for each application program and used for tuning the heavily loaded resources. For our example, the **cal** command is used. Enter the following:

```
$ timex cal
   April 1988
 S  M Tu  W Th  F  S
                1  2
 3  4  5  6  7  8  9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30


real        0.10
user        0.00
sys         0.07
```

While **cal**, for its simplicity, was used for the preceding demonstration, it is not the best example since it is not a major user of system resources.

timex can be used in the following way:

timex -s *application program*

Your application program will operate normally. When you finish and exit, the **timex** result will be printed on your screen. You get a clear picture of system resources used by your program.

sadp Command

The **sadp** command

sadp [**-th**] [**-d** *device*[*-drive*]] **s** [*n*]

reports disk access locations and seek distance in tabular (**-t**) or histogram (**-h**) form. Disk activity is sampled once every second during a specified interval of *s* seconds. The number of reports to be generated in the sampling interval is specified by the optional *n* argument. The default value of *n* is 1.

The **-d** argument specifies the *device* and *drive* to be profiled. Valid names for the *device* are **hdsk** for non-SCSI disks, **sdsk** for SCSI disks, and **fdsk** for floppy disk drives.

An example of **sadp** output for SCSI hard disk drive 0 follows. In this example, one report is generated during the sampling interval of 3600 seconds (1 hour).

```
$ sadp -h -d sdsk-0 3600<CR>

Mon Apr  4 03:18:54 1988

unix unix 3.2.2 3 3B2

CYLINDER ACCESS HISTOGRAM
sdsk-0:
Total transfers = 33291
```

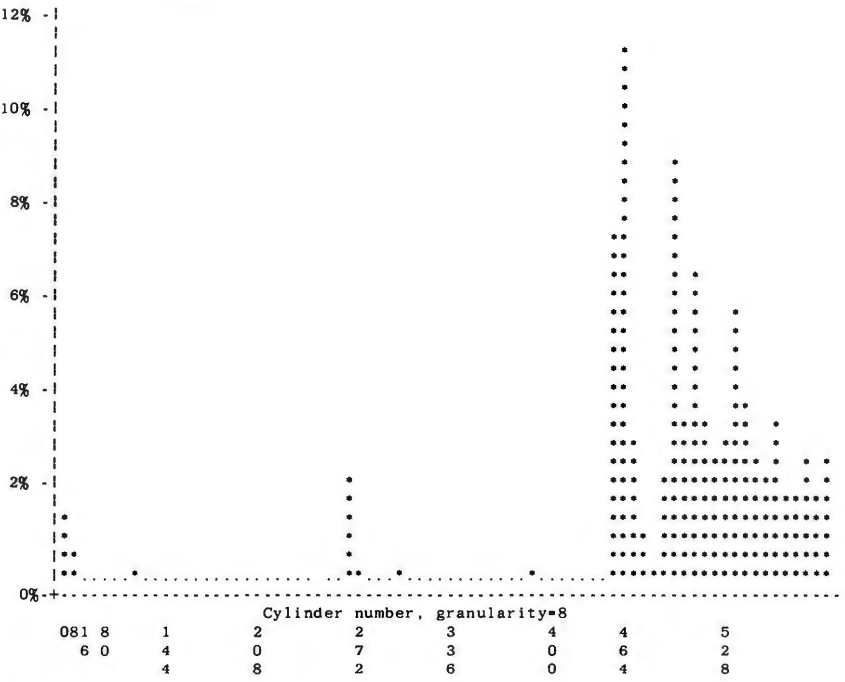


Figure 6-3: Output From **sadp**: Cylinder Access Histogram

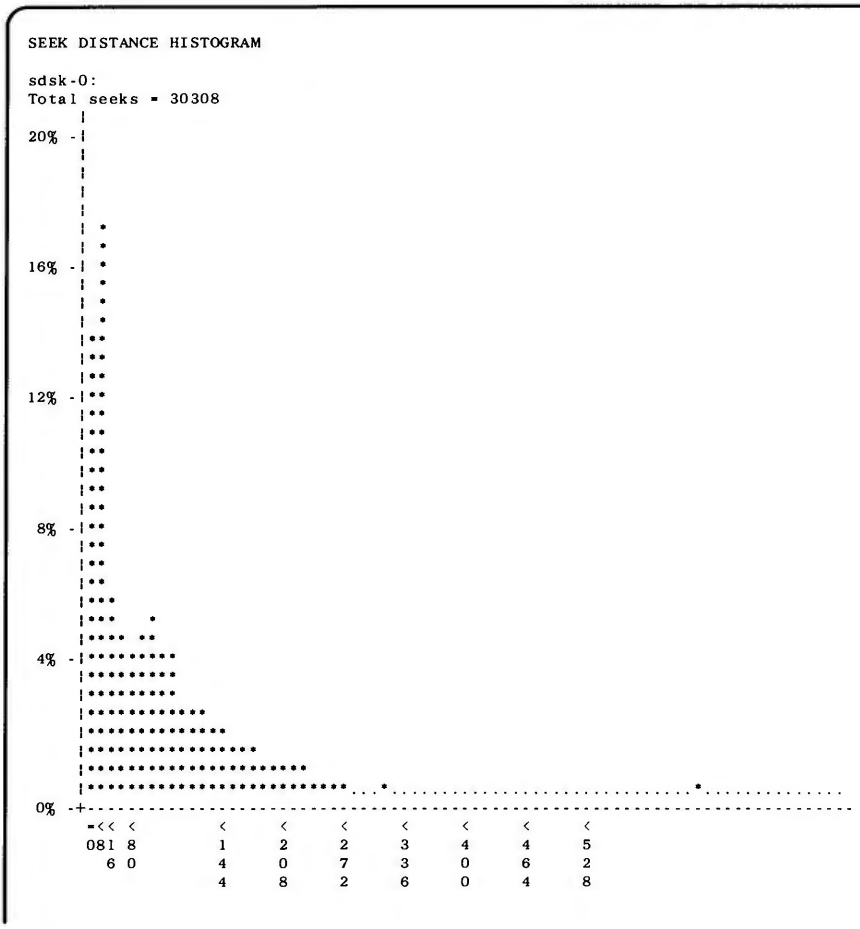


Figure 6-4: Output From **sadp**: Seek Distance Histogram

Using the **sadp** output, along with the output of **/etc/mount**(1M) or **/etc/prtvtoc**(1M), and a table of disk sections [see Appendix A, "Device Names and Designators," 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 areas of high activity close together. This will reduce the number of seeks over large distances.

The first graph (Figure 6-3) shows excellent disk cylinder locality of references. This means that every time the location of a block was referenced by the disk head (for reading or writing), it most often occurred in the same general region of the disk. In the example, most references (as shown by percent times referenced) occur for files near cylinders 450 to 600, with a few for files around cylinder 250. There are a few references to other files on the disk, but they occur only a small percent 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 6-4) 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 are under 10 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 about one-third of the disk activity, the disk head was forced to move no more that 16 cylinders to reference a given block, and the more to the left the stars are grouped, the better.

These two graphs give an idea of 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 4, "Disk/Tape Management," and Chapter 5, "File System Administration," for more information.)

Tunable Parameters

Tunable system parameters are used to set various table sizes and system thresholds to handle the expected system load. Some of these parameters are automatically set by the system. 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.

Figure 6-5 shows the autotuned and default tunable parameter values for a Release 3.2.2 System. The parameters shown in the figure are defined in the following files:

- **/etc/master.d/disp**
- **/etc/master.d/eports**
- **/etc/master.d/idisk**
- **/etc/master.d/iuart**
- **/etc/master.d/kernel**
- **/etc/master.d/mirror**
- **/etc/master.d/msg**
- **/etc/master.d/ports**
- **/etc/master.d/prf**
- **/etc/master.d/s5**
- **/etc/master.d/scsi**
- **/etc/master.d/sd01**
- **/etc/master.d/sem**
- **/etc/master.d/shm**

The default parameter settings defined in the **/etc/master.d/kernel** and **/etc/master.d/disp** files are delivered with the Operating System Utilities cartridge tape. Parameters for Remote File Sharing (RFS) and Networking Support Utilities (NSU) are always present, but should be left at their default setting of 0; nonzero settings are automatically provided when the RFS and NSU packages are installed. The message, semaphore, and shared memory

parameters are distributed with the Inter-Process Communications Utilities in **/etc/master.d/msg**, **/etc/master.d/sem**, and **/etc/master.d/shm**, respectively.

The current system parameter definitions are displayed using the **/etc/sysdef**(1M) command. See Procedure 6.3, "Display System Paramter Definitions," for an example of this command.

The following notes apply to Figure 6-5:

- The value of a few parameters is calculated each time a new kernel (**/unix**) is generated, unless the value is manually overridden (see note).
- All the other 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 Figure 6-5.
- A dash (—) is used in the size information to show parameters that set flags in the kernel. Parameters that set flags do not affect the size of the kernel when their values are changed; only the values of the specific flags are changed.

Note: A calculated (autotuned) parameter value is overridden by adding the parameter name and value to the appropriate **/etc/master.d** file. Overriding the calculated value causes the parameter to be set to the new value each time a new kernel (**/unix**) is generated. When calculated (autotuned) parameter values are overridden, any further changes in the hardware configuration (adding new memory for example) require another manual change of the value. This allows you to improve the performance of the new configuration.

AUTOTUNED PARAMETERS				
Parameter	4M	6M	8M	Size per Entry in bytes
Est. Users	16	24	32	—
NAUTOUP	15	15	30	—
NBUF	300	400	500	1080
NCLIST *	316	460	604	72
NFILE	400	500	600	12
NHBUF	256	256	512	16
NINODE	400	500	600	76
NPROC	120	150	200	184
NREGION	360	450	600	36
NS5INODE	400	500	600	68

* NCLIST is autotuned, based on the number of EPORTS and PORTS.

Figure 6-5: Default Parameter Values: Release 3.2.2 Systems (Sheet 1 of 7)

AUTOTUNED PARAMETERS					
Parameter	10M	12M	14M	16M 64M	Size per Entry in Bytes
Est. Users	40	48	56	64	
NAUTOUP	30	45	45	45	—
NBUF	600	700	800	1100	1080
NCLIST	748	892	1036	1276	72
NFILE	800	1000	1100	1300	12
NHBUF	512	1024	1024	1024	16
NINODE	800	1000	1100	1300	76
NPROC	240	300	350	400	184
NREGION	720	900	1050	1200	36
NS5INODE	800	1000	1100	1300	68

Figure 6-5: Default Parameter Values: Release 3.2.2 Systems (Sheet 2 of 7)

TUNABLE PARAMETER			
/etc/master.d File	Parameter	Default Value	Size per Entry in Bytes
disp	MAXSLICE	100	—
eports	ESAVEXP	24	12
iuart	IUQSIZ	12	16
kernel	AUTOBOOT	1	—
	AUTODUMP	1	—
	BDFLUSHMAX	10	—
	BDFLUSHR	1	—
	CACHESTACK	1	—
	CONBUFSZ	2048	—
	FLCKREC	100	28
	GPGSHI	40	—
	GPGSLO	25	—
	GPGSMSK	0x00000220	—
	M64BUF *	50	2048
	M64MAP *	100	8
	M64PDE *	100	—
	MAXFC	1	—
	MAXPMEM	0	—
	MAXSC	1	—
	MAXSEPGCNT	1	2048
	MAXUMEM	8192	—
	MAXUP	30	—
	MINARMEM	40	—
	MINASMEM	40	—

* M64BUF, M64MAP, and M64PDE parameters apply only to systems equipped with more than 16 MB of RAM.

Figure 6-5: Default Parameter Values: Release 3.2.2 Systems (Sheet 4 of 7)

TUNABLE PARAMETER			
/etc/master.d File	Parameter	Default Value	Size per Entry in Bytes
kernel	NBLK4	384	50
	NBLK16	384	61
	NBLK64	512	110
	NBLK128	280	174
	NBLK256	32	302
	NBLK512	16	558
	NBLK1024	20	1070
	NBLK2048	28	2094
	NBLK4096	0	4142
	NCALL	60	16
	NMOUNT	25	36
	NMUXLINK	32	12
	NODE	"unix"	—
	NOFILES	20	—
	NPBUF	20	52
	NQUEUE	432	256
	NSRMOUNT	50	28
	NSTREAM	36	32
	NSTREVENT	288	12
	NSTRPUSH	9	—
	PIRCOUNT	100	12
	PUTBUFSZ	2000	—
	REL	"3.2.2"	—
	SBEDELAY	30	
	SHLBMAX	2	12 × NPROC
	SPTMAP	50	8
	STRCTLSZ	1024	—
	STRLOFRAC	80	—
	STRMEDFRAC	90	—
	STRMSGSZ	4096	—

Figure 6-5: Default Parameter Values: Release 3.2.2 Systems (Sheet 5 of 7)

TUNABLE PARAMETER			
/etc/master.d File	Parameter	Default Value	Size per Entry in Bytes
kernel	SYS	"unix"	—
	SYSSEGSZ	1024	
	ULIMIT	2048	
	VER	"3"	—
	VHANDL	10	—
	VHANDR	1	—
	VHNDFRAC	16	—
mirror	MPART	16	
msg	MSGMAP	100	8
	MSGMAX	2048	—
	MSGMNB	4096	—
	MSGMNI	50	53
	MSGSEG	1024	8
	MSGSSZ	8	1024
	MSGTQL	40	12
ports	SAVEXP	5	
prf	PRFMAX	2048	8
s5	FLAGS5	0	—
	NAMES5	"S5"	—
	NOTFYS5	0x4	—
	S5_BUCKETS	311	4
	S5_ENTRIES	2000	36

Figure 6-5: Default Parameter Values: Release 3.2.2 Systems (Sheet 6 of 7)

TUNABLE PARAMETER			
/etc/master.d File	Parameter	Default Value	Size per Entry in Bytes
scsi	N_SCB	8	
sd01	JOBS	100	40
sem	NBPW	4	
	SEMAEM	16384	—
	SEMMAP	10	8
	SEMMNI	10	32
	SEMMNS	60	8
	SEMMNU	30	8 × (SEMUME + 2)
	SEMMSL	25	—
	SEMOPM	10	8
	SEMUME	10	8 × SEMMNU
	SEMVMX	32767	—
shm	SHMALL	512	—
	SHMMAX	131072	—
	SHMMIN	1	—
	SHMMNI	100	52
	SHMSEG	6	12 × NPROC

Figure 6-5: Default Parameter Values: Release 3.2.2 Systems (Sheet 7 of 7)

Kernel Parameters

The kernel and kernel-related parameters are defined in the following files:

- /etc/master.d/disp
- /etc/master.d/eports
- /etc/master.d/kernel
- /etc/master.d/ports
- /etc/master.d/prf.LI **/etc/master.d/s5**
- **/etc/master.d/sd01**

Unless otherwise stated in these descriptions, the parameter is in the **kernel** file.

AUTOBOOT The value should be 0 or 1. If set to 1, then the system will automatically initiate a boot after a system panic. If set to 0, the system will return to firmware mode. The default value is 1.

AUTODUMP The value should be 0 or 1. If set to 1, the system will automatically initiate a memory dump to the dump device if the system panics. The default value is 1.

BDFLUSHR The BDFLUSHR parameter specifies the rate in seconds for checking the need to write the file system buffers to disk. The default is 1 second.

BDFLUSHMAX

The BDFLUSHMAX parameter specifies the amount of time between buffer flushes. The values can be set between 0 and 10. Buffer flushing uses considerable CPU time when set to 0.

CACHESTACK

For performance reasons, the value should be 1. The default value is 1. If the CACHESTACK tunable is non-zero, the CACHEABLE bit is set ON in the MMU Segment Descriptors for each process's stack. This will tend to increase performance.

CONBUFSZ The CONBUFSZ parameter specifies the size of a circular buffer, **consbuf**, used to contain all console I/O (stdin, stdout, and stderr). The **conslog**(1M) command is used to activate/deactivate the logger. A daemon process is used to read the buffer and to write its contents to a file located under **/usr/adm** called **conlog**. The initial size of the buffer is 2048 bytes.

ESAVEXP The ESAVEXP parameter is specified in the **/etc/master.d/eports** file. The ESAVEXP parameter specifies the number of common I/O bus express queue entries. The default value of this parameter (24) should not be changed.

FLAGS5 The FLAGS5 parameter is specified in the **/etc/master.d/s5** file. This flag is specific to the file system type. It is not tunable and should not be changed.

FLCKREC The FLCKREC parameter specifies the number of records that can be locked by the system. The default value is 100. Each entry contains 28 bytes.

GPGSHI See "Paging Parameters" section of this chapter.

GPGSLO See "Paging Parameters" section of this chapter.

GPGSMSK See "Paging Parameters" section of this chapter.

ILOGSIZE The ILOGSIZE parameter is used to track igets and iputs of i-nodes. It is used for debugging only and should be set to zero on a production system.

JOBS The JOBS parameter is defined in the **/etc/master.d/sd01** file. JOBS is equal to the number of job structures allocated per disk target controller. The default value is 100. It should be increased if the driver reports that it is out of jobs.

M64BUF For each raw I/O buffer (read/write system call or page swap) there may be some pages within the buffer whose physical address is greater than 16 MBs. The contents of these pages must be copied to an area below 16 MBs before the I/O can take place. Therefore, the kernel

reserves a buffer pool containing M64BUF pages.

The minimum value of M64BUF is 1; however, this will likely incur a severe performance penalty because raw I/O jobs with data over 16 MBs will contend for the use of a single page. Also, any raw I/O jobs containing more than 1 page over 16 MBs will fail.

The maximum value of M64BUF is the expected sum at any given instant of one for each process undergoing a page swap, and the total number of pages in all raw I/O buffers.

If M64BUF is too small, the kernel will wait for buffers to become available as each raw I/O job completes. A message is issued to the console log file if this occurs. Therefore, the practical value to use for M64BUF is the upper limit of an estimate of the total number of pages above 16 MBs involved in raw I/O at any given instant.

M64MAP Each raw I/O request (read/write system call or page swap) involves a transfer of data from a contiguous block of virtual memory. For buffers containing pages whose physical address is above 16 MBs, the kernel must copy the contents of those pages above 16 MBs into a page allocated from the pool (see M64BUF). At the same time, the pages of the I/O buffer are remapped to a kernel virtual address, thereby creating a new contiguous block of virtual memory, all pages of which are below 16 MBs.

This new contiguous virtual memory is allocated from the SPTMAP, whose number of elements is increased by the value of M64MAP.

M64PDE The M64PDE parameter is the number of page descriptors needed for the remapping of raw I/O jobs. The value of M64PDE should be the upper limit of the number of pages involved in all raw I/O jobs that contain data physically above 16 MBs.

If this number is too small, a message is issued to the

console log file, and the kernel waits for space to become available.

MAXFC See "Paging Parameters" section of this chapter.

MAXPMEM See "Paging Parameters" section of this chapter.

MAXSC See "Paging Parameters" section of this chapter.

MAXSEPGCNT
See "STREAMS Parameters" section of this chapter.

MAXSLICE The MAXSLICE parameter specifies in clock ticks the maximum time slice for user processes. After a process executes for its allocated time slice, that process is suspended. The operating system then dispatches the highest priority process and allocates to it MAXSLICE clock ticks. MAXSLICE, defined in **/etc/master.d/disp**, is normally 1 second (100 clock ticks on the 3B2 computer).

MAXUMEM See "Paging Parameters" section of this chapter.

MAXUP The MAXUP parameter specifies how many concurrent processes a nonsuper user is allowed to run. The entry is normally between 15 and 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 12 people are logged in on the same user identification, the default limit would be reached quickly.

NAMES5 The NAMES5 parameter is specified in the **/etc/master.d/s5** file. This parameter is the name of the file system type. It is not tunable and should not be changed.

NAUTOUP The NAUTOUP entry specifies the buffer age in seconds for automatic file system updates. A system buffer is written to the hard disk when it has been memory-resident for the interval specified by the NAUTOUP parameter. Specifying a smaller limit increases system reliability by writing the buffers to disk more frequently and decreases system performance. Specifying a larger

limit increases system performance at the expense of reliability. The NAUTOUP parameter is autotuned based on the size of RAM.

NBLK*n* See "STREAMS Parameters" section of this chapter.

NBUF The NBUF entry specifies how many 2KB system buffers to allocate. The UNIX system buffers form a data cache. The data cache is an area of kernel data space used to hold data being transferred to and from user space and the physical disk storage. Improvement in the hit rate of this cache increases with the number of buffers. Cache hits reduce the number of disk accesses and thus improve overall performance. The entries are normally between 300 and 1100. (Be careful not to allocate too many buffers because BDFLUSH and SYNC can take a long time.) The NBUF parameter is autotuned based on the size of RAM.

Each buffer contains 2104 bytes. Hash buffers (NHBUF) should be increased along with system buffers (NBUF) for optimal performance. The value for NBUF is calculated automatically by the system, unless specifically set to a value by adding a line (NBUF = *n*) to the **/etc/master.d/kernel** file.

NCALL The NCALL parameter specifies how many callout 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 between 10 and 70. The default value is 60. Each entry contains 16 bytes.

Software drivers may use call entries to check hardware device status. When the callout table overflows, the system crashes and outputs the following message on the system console:

```
PANIC: Timeout table overflow
```

NCLIST The NCLIST parameter specifies how many character list buffers to allocate. Each buffer contains up to 64 bytes. The buffers are dynamically linked to form input and

output queues for the terminal lines and other slow speed devices. The average number of buffers needed per terminal is between 5 and 10. Each entry (buffer space plus header) contains 72 bytes. When full, input and output characters dealing with terminals are lost, although echoing continues. The NCLIST parameter is autotuned based on the number of PORTS and EPORTS cards.

```
28 + (80)(number of ports) + (144)(number of
EPORTS)
```

NFILE

The NFILE parameter specifies how many open file table entries to allocate. Each entry represents an open file. The entry is normally between 400 and 1300. Each entry contains 12 bytes. The NFILE entry relates directly to the NINODE entry. Normally, NINODE is equal to or greater than NFILE, however; some applications allow NFILE to be greater than NINODE. The NFILE control structure operates similar to the NINODE structure. When the file table overflows, the following message is output on the system console.

```
NOTICE: file table overflow
```

As a reminder, this parameter does not affect the number of open files per process (see the NOFILES parameter). The NFILE parameter is autotuned based on the size of RAM.

NHBUF

The NHBUF parameter specifies how many "hash buckets" to allocate. These are used to search for a buffer given a device number and block number rather than a linear search through the entire list of buffers. **This value must be a power of 2.** Each entry contains 12 bytes. The value of NHBUF is autotuned based on the size of RAM.

NINODE The NINODE parameter specifies how many i-node table entries to allocate. Each table entry represents an in-core i-node that is an active file. For example, an active file might be a current directory, an open file, or a mount point. The file control structure is modified when changing this variable. The number of entries used depends on the number of opened files. The entries are normally between 400 and 1300. The value for NINODE pertains directly to the NFILE value. Normally, NINODE is equal to or greater than NFILE; however, some applications allow NFILE to be greater than NINODE. NINODE must always be less than or equal to NS5INODE. NINODE greater than NS5INODE results in an unusable system. The NINODE parameter is autotuned based on the size of RAM. When the i-node table overflows, the following warning message is output on the system console:

```
WARNING: i-node table overflow
```

NMOUNT The NMOUNT parameter specifies how many mount table entries to allocate. Each entry represents a mounted file system. The root (/) file system is always the first entry. When full, the **mount**(2) system call returns the error EBUSY. Since the mount table is searched linearly, this value should be as low as possible.

NMUXLINK See "STREAMS Parameters" section of this chapter.

NODE The NODE parameter specifies the node name of the system. The default node name is **unix** (see Procedure 1.4, "Establish or Change System and Node Names").

NOFILES The NOFILES parameter specifies the maximum number of open files per process. Default is 20. Values higher than 20 are accessible only to processes using system calls [**open**(2), **creat**(2); for example]. Processes using standard I/O subroutines are limited to 20, independent of the value of NOFILES. Unless an application package recommends that NOFILES be changed, the default setting of 20 should be left as is.

/bin/sh uses 3 file table entries: standard input, standard output, and standard error (0, 1, and 2 are normally reserved for stdin, stdout, and stderr, respectively). This leaves the value of NOFILES minus 3 as the number of other open files available per process. If a process requires up to 3 more than this number, then the standard files must be closed. This practice is NOT recommended, and must be used with caution, if at all.

If the configured value of NOFILES is greater than the maximum (100) or less than the minimum (20), the configured value is set to the default (20), and a NOTICE message is sent to the console.

NOTFYS5 The NOTFYS5 flag is specified in the **/etc/master.d/s5** file. This is a flag used to determine when the file system switch entry point "fs_notify" should be called. This flag should not be changed.

NPART NPART specifies the number of mirror devices on the system. A separate mirror device is needed for every pair of disk partitions that are mirrored together. Increasing NPART by one will grow the mirror driver data size by 36 bytes.

NPBUF The NPBUF parameter specifies how many physical input/output buffers to allocate. One input/output buffer is needed for each physical read or write active. Each entry contains 52 bytes. The default value is 20.

NPROC The NPROC parameter specifies how many process table entries to allocate. Each table entry represents an active process. The swapper is always the first entry and **/etc/init** is always the second entry. The number of entries depends on the number of terminal lines available and the number of processes spawned by each user.

The average number of processes per user is between 2 and 5 (also see MAXUP, default value 30). When full, the **fork**(2) system call returns the error EAGAIN. The NPROC entry is between 120 and 400. The NPROC parameter is autotuned based on the size of RAM.

NQUEUE See "STREAMS Parameters" section of this chapter.

NREGION The NREGION parameter specifies how many region table entries to allocate. Each NREGION entry contains 36 bytes. Most processes have 3 regions: text, data, and stack. Additional regions are needed for each shared memory segment and shared library (text and data) attached. However, the region table entry for the text of a "shared text" program will be shared by all processes executing that program. Each shared memory segment attached to one or more processes uses another region table entry. A good starting value for this parameter is from 3 to 3.5 times NPROC. The NREGION parameter is autotuned based on the size of RAM. If the system runs out of region table entries, the following message is output on the system console.

```
Region table overflow
```

NS5INODE The NS5INODE must always be equal to or greater than NINODE. The NS5NINODE parameter is autotuned based on the size of RAM.

NSRMOUNT See "Remote File Sharing Parameters" section of this chapter.

NSTREAM See "STREAMS Parameters" section of this chapter.

NSTREVENT See "STREAMS Parameters" section of this chapter.

NSTRPUSH See "STREAMS Parameters" section of this chapter.

PIRCOUNT The PIRCOUNT parameter specifies the size of the programmable interrupt request queue entries. The default value is 100. If this number is too small, the system will panic. Since each queue entry is only 12 bytes, it does no harm to set this tunable parameter slightly higher than required. If the system runs out of queue entries, it will panic with the message:

```
PANIC: pir queue overflow
```

PRFMAX The PRFMAX parameter is specified in the **/etc/master.d/prf** file. The PRFMAX parameter specifies the maximum number of symbols the kernel profiler can reference.

PUTBUFSZ The PUTBUFSZ parameter specifies the size of a circular buffer, **putbuf**, that is used to contain a copy of the last PUTBUFSZ characters written to the console by the operating system. The contents of **putbuf** can be viewed using **crash**(1M).

REL The REL parameter specifies the UNIX system release.

SAVEXP The SAVEXP parameter is specified in the **/etc/master.d/ports** file. The SAVEXP parameter specifies the number of common I/O bus express queue entries. The default value of this parameter (5) should not be changed.

SBEDELAY After a single-bit memory error is reported, these messages are disabled for SBEDELAY occurrences. Silenced single-bit errors are still corrected by the hardware while error reporting is disabled. The default value is 30. A value of zero disables all messages after the first occurrence.

SHLBMAX The SHLBMAX parameter specifies the maximum number of shared libraries that can be attached to a process at one time.

SPTMAP The SPTMAP parameter specifies the number of entries in kernel page table map that manage free virtual memory. The SPTMAP default value is 50. The SPTMAP number of elements is increased by the value of M64MAP. Therefore, the SPTMAP value reported in a **sysdef** output is equal to (SPTMAP + M64MAP). See the M64MAP description for more information.

STRCTLSZ See "STREAMS Parameters" section of this chapter.

STRLOFRAC See "STREAMS Parameters" section of this chapter.

STRMEDFRAC See "STREAMS Parameters" section of this chapter.

STRMSGSZ See "STREAMS Parameters" section of this chapter.

SYS The SYS parameter specifies the system name. The default system name is **unix**. (See Procedure 1.4, "Establish or Change System and Node Names.")

ULIMIT The ULIMIT parameter specifies, in 512-byte blocks, the size of the largest file that an ordinary user may write. The default value is 2048; that is, the largest file an ordinary user may write is 1,048,576 bytes. The super-user may write a file as large as the file system can hold. The ULIMIT parameter does not apply to reads; any user may read a file of any size.

VER The VER parameter specifies the hardware version.

VHANDL See "Paging Parameters" section of this chapter.

VHANDR See "Paging Parameters" section of this chapter.

VHNDFRAC See "Paging Parameters" section of this chapter.

Cache Parameters

The configurable parameters for cache are found in the file **/etc/master.d/s5**. They are:

S5_BUCKETS The S5_BUCKETS parameter specifies the number of hash buckets used for hashing entries in the directory cache. The default is 311.

S5_ENTRIES The S5_ENTRIES parameter specifies the number of most recently accessed directory entries to cache for a more efficient path name resolution. The cache is a hash table in main memory. Entries in this table are flushed whenever a mount or unmount is done. Each table entry requires 36 bytes. The default is 2000.

Paging Parameters

There exists in the system a paging daemon, **vhand**, whose sole responsibility is to free up memory as the need arises. It uses a "least recently used" algorithm to approximate process working sets, and it writes those pages out to disks that have not been touched for some time. The page size is 2048 bytes. When memory is exceptionally tight, the working sets of entire processes may be swapped out.

The following tunable parameters determine how often **vhand** runs and under what conditions. The default values in **/etc/master.d/kernel** should be adequate for most applications.

GPGSHI The GPGSHI parameter specifies the high-water mark of free memory in pages for **vhand** to stop stealing pages from processes. The default is 40. Increase the value to make the daemon more active; decrease the value to make the daemon less active (The value must be an integer > 0, > GPGSLO, and < 25 percent of the number of pages of available memory).

GPGSLO The GPGSLO parameter specifies the low-water mark of free memory in pages for **vhand** to start stealing pages from processes. The default is 25. Increase the value to make the daemon more active; decrease the value to make the daemon less active (must be an integer ≥ 0 and < GPGSHI).

GPGSMSK The GPGSMSK parameter specifies the mask used by the paging daemon to determine the required number of aging passes before stealing a page. The default is 0x00000220. This value should not be changed.

MAXFC The MAXFC parameter specifies the maximum number of pages that can be added to the freelist in a single operation. The default value is 1.

MAXPMEM The MAXPMEM parameter specifies the maximum amount of physical memory to use in pages. The default value of 0 specifies that all available physical memory be used.

MAXSC The MAXSC parameter specifies the maximum number of pages that can be swapped out in a single operation. The default value is 1.

MAXUMEM The MAXUMEM parameter specifies the maximum size of a user's virtual address space in pages. This value cannot be greater than 8192. The default is 8192.

MINARMEM The MINARMEM parameter specifies the minimum number of memory pages reserved for the text and data segments of user processes.

MINASMEM The MINASMEM parameter is a 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).

VHANDL The value of VHANDL determines when the paging daemon **vhand** runs. The amount of available free memory is compared with the value of VHANDL every VHANDR seconds. If free memory is less than VHANDL, then the paging daemon **vhand** is awakened. VHANDL is autotuned based on the size of memory. Increase

VHANDL to make the daemon more active; decrease VHANDL to make the daemon less active.

VHANDR The VHANDR parameter specifies in seconds the maximum rate at which **vhand** parameter can run. The **vhand** parameter will only run at this rate if free memory is less than VHANDL, as explained previously for VHNDFRAC. The default is 1. Increase the value to make the daemon less active (must be an integer > 0 and ≤ 300). If you have set the value higher, decreasing it makes the daemon more active.

VHNDFRAC The VHNDFRAC parameter is used to determine the initial value for the system variable VHANDL. VHANDL is set to the maximum user-available memory divided by VHNDFRAC or the value of GPGSHI, whichever is larger.

The default for VHNDFRAC is 16. Decrease this value to make the daemon more active; increase this value to make the daemon less active (must be > 0 and < 25 percent of available memory).

STREAMS Parameters

The following tunable parameters are associated with STREAMS processing. These parameters are defined in the **/etc/master.d/kernel** file. The values should not be changed unless the optional Network Support Utilities package has been installed.

MAXSEPGCNT

The MAXSEPGCNT parameter specifies the number of additional pages of memory that can be dynamically allocated for event cells. If this value is 0, only the allocation defined by NSTREVENT is available for use. If the value is not 0 and if the kernel runs out of event cells, it will, under some circumstances, attempt to allocate an extra page of memory from which new event cells can be created. MAXSEPGCNT places a limit on the number of pages that can be allocated for this purpose. On a 3B2

computer, each new page can provide 166 event cells. Once a page has been allocated for event cells, however, it cannot be recovered later for use elsewhere. It is recommended that the NSTREVENT value be set to accommodate most load conditions, and that MAXSEPGCNT be set to 1 to handle exceptional load cases should they arise.

NBLK*n* The NBLK*n* parameters specify the number of STREAMS data blocks and buffers to be allocated for each size class. Message block headers are also allocated based on these numbers: the number of message blocks is 1.25 times the total of all data block allocations. This gives a message block for each data block, plus some extras for duplicating messages [kernel functions **dupb**(), **dupmsg**()]. The optimal configuration depends on both the amount of primary memory available and the intended application. The default values in the NSU package are intended to support a moderately loaded configuration using RFS and UUCP/CU over a STREAMS-based network.

NMUXLINK the NMUXLINK parameter specifies the maximum number of multiplexer links to be configured. One link structure is required for each active multiplexor link (STREAMS I_LINK **ioctl**). This number is application-dependent; the default allocation equal to the number of STREAMS (NSTREAM) guarantees availability of links.

NQUEUE The NQUEUE parameter specifies the number of STREAMS queues to be configured. Queues are always allocated in pairs, so this number should be even. A minimal STREAM contains four queues (two for the STREAM head, two for the driver). Each module pushed on a STREAM requires an additional two queues. A typical configuration value is 8*NSTREAM.

NSTREAM The NSTREAM parameter specifies the number of "STREAM-head" (stdata) structures to be configured. One is needed for each STREAM opened, including both STREAMS currently open from user processes and STREAMS linked under multiplexers. The recommended configuration value is highly application-dependent, but a

value of 32—40 usually suffices on a 3B2 computer for running a single transport provider with moderate traffic.

NSTREVENT The NSTREVENT parameter specifies the initial number of STREAM event cells to be configured. STREAM event cells are used for recording process-specific information in the **poll**(2) system call. They are also used in the implementation of the STREAMS I_SETSIG **ioctl** and in the kernel **bufcall**() mechanism. A rough minimum value to configure would be the expected number of processes to be simultaneously using **poll**(2) times the expected number of STREAMS being polled per process, plus the expected number of processes expected to be using STREAMS concurrently. The default is 256. Note that this number is not necessarily a hard upper limit on the number of event cells that will be available on the system (see MAXSEPGCNT).

NSTRPUSH The NSTRPUSH parameter specifies the maximum number of modules that may be pushed onto a STREAM. This is used to prevent an errant user process from consuming all the available queues on a single STREAM. The default value is 9, but in practice, existing applications have pushed at most four modules on a STREAM.

STRCTLSZ The STRCTLSZ parameter specifies the maximum allowable size of the control portion of any STREAMS message. The control portion of a **putmsg**(2) message is not subject to the constraints of the min/max 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 enough for existing applications.

STRLOFRAC The percentage of data blocks of a given class at which low-priority block allocation requests are automatically failed. For example, if STRLOFRAC is 80 and there are 48 256-byte blocks, a low-priority allocation request will fail when more than thirty-eight 256-byte blocks are already allocated. The parameter is used to help prevent deadlock situations by starving out low-priority activity. The recommended value of 80 works well for current

applications. STRLOFRAC must always be in the range 0 <= STRLOFRAC <= STRMEDFRAC.

STRMEDFRAC

The STRMEDFRAC parameter specifies the percentage cutoff at which medium priority block allocations are failed (see STRLOFRAC discussion above). The recommended value of 90 works well for current applications. STRMEDFRAC must always be in the range STRLOFRAC <= STRMEDFRAC <= 100.

Note: There is no cutoff fraction for high-priority allocation requests; it is effectively 100.

STRMSGSZ The STRMSGSZ parameter specifies 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**(2) or **putmsg**(2) can consume an inordinate number of message blocks. The recommended value of 4096 is enough for existing applications.

Message Parameters

The following tunable parameters are associated with Inter-Process Communication (IPC) messages. These parameters are defined in the **/etc/master.d/msg** file.

MSGMAP The MSGMAP parameter specifies the size of the control map used to manage message segments. Default value is 100. Each entry contains 8 bytes.

MSGMAX The MSGMAX parameter specifies the maximum size of a message. The default value is 2048. The maximum size is 64 kilobytes -1.

MSGMNB The MSGMNB parameter specifies the maximum length of a message queue. The default value is 4096.

MSGMNI The MSGMNI parameter specifies the maximum number of message queues systemwide (id structure). The default value is 50.

MSGSSZ The MSGSSZ parameter 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 The MSGTQL parameter 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 The MSGSEG parameter 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).

Semaphore Parameters

The following tunable parameters are associated with Inter-Process Communication (IPC) semaphores. These parameters are defined in the **/etc/master.d/sem** file.

SEMMAP Specifies the size of the control map used to manage semaphore sets. The default value is 10. Each entry contains 8 bytes.

SEMMNI The SEMMNI parameter specifies the number of semaphore identifiers in the kernel. This is the number of unique semaphore sets that can be active at any given time. The default value is 10. Each entry contains 32 bytes.

SEMMNS The SEMMNS parameter specifies the number of semaphores in the system. The default value is 60. Each entry contains 8 bytes.

SEMMNU The SEMMNU parameter specifies the number of undo structures in the system. The default value is 30. The size is equal to [8 × (SEMUME + 2)] bytes.

SEMMSL The SEMMSL parameter specifies the maximum number of semaphores per semaphore identifier. The default value is 25.

SEMOPM The SEMOPM parameter 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 The SEMUME parameter specifies the maximum number of undo entries per undo structure. The default value is 10. The size is equal to 8*(SEMMNU) bytes.

SEMVMX The SEMVMX parameter specifies the maximum value a semaphore can have. The default value is 32767. The default value is the maximum value for this parameter.

SEMAEM The SEMAEM parameter 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 16384. The default value is the maximum value for this parameter.

Shared Memory Parameters

The following tunable parameters are associated with Inter-Process Communication (IPC) shared memory. These parameters are defined in the **/etc/master.d/shm** file.

SHMMAX The SHMMAX parameter specifies the maximum shared memory segment size. The default value is 131072.

SHMMIN The SHMMIN parameter specifies the minimum shared memory segment size. The default value is 1.

SHMMNI The SHMMNI parameter specifies the maximum number of shared memory identifiers system wide. The default value is 100. Each entry contains 52 bytes.

SHMSEG The SHMSEG parameter specifies the number of attached shared memory segments per process. The default value is 6. The maximum value is 15.

SHMALL The SHMALL parameter specifies the maximum number of in-use shared memory text segments. The default value is 512.

Remote File Sharing Parameters

Remote File Sharing (RFS) parameters are discussed in the documentation that is included with the optional *Remote File Sharing Utilities* package. NSRMOUNT is included in Figure 6-5 because it is located in the **/etc/master.d/kernel** file. The values associated with NSRMOUNT are 50 unless the RFS package is installed.

Chapter 7: LP Spooling Administration

Introduction

This chapter tells you:

- How the LP print service works
- How to find instructions for installing the LP print service
- The commands used to administer the LP print service
- How to stop and start the LP print service
- How to configure the LP print service:
  - How to set up printer configurations
  - How to manage the printing load
  - How to set job priority limits for users
  - How to manage pre-printed forms
  - How to define filters.
- LP print service files and directories
- How to write customized filters and interface programs.

This chapter describes in detail the procedures needed to administer this feature. Error messages issued by the LP print service are listed in Appendix C, "Error Messages."

How the LP Print Service Works

The LP print service, originally called the LP spooler, is a mechanism that allows you 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 LP print service system is software provided by the LP Spooling Utilities that:

- Handles the task of receiving files users want printed
- Filters the files (if needed) so they can print properly
- Schedules the work of one or more printers
- Starts programs that interface with the printer(s)
- Keeps track of the status of jobs
- Alerts you to printer problems
- Keeps track of forms currently mounted and alerts you to mount needed forms
- Issues error messages when problems arise.

Installation Information

Instructions for installing the LP Spooling Utilities are in Procedure 7.1, "Install the LP Spooling Utilities." The following two manuals can be of specific help in the area of printer operation:

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 this book.

Summary of User Commands

The LP print service provides three regular user commands. These are shown in Figure 7-1.

Command	Description
cancel(1)	Cancel a request for a file to be printed.
lp(1)	Send a file or files to a printer.
lpstat(1)	Report the status of the LP system.

Figure 7-1: User Commands for the LP Print Service

In addition to being able to send requests to the LP print service system, check the status of requests, and cancel requests, users may be given the ability to disable and enable a printer. The idea is that if a user finds a printer is malfunctioning in some way, it should not be necessary to call the administrator to turn the printer off. On the other hand, it may not be reasonable in your printing environment to allow regular users to disable a printer. You can control whether other users have access to the two commands shown in Figure 7-2.

Command	Description
disable(1)	Deactivate the named printer(s).
enable(1)	Activate the named printer(s).

Figure 7-2: Privileged User Commands for the LP Print Service

Summary of Administrative Commands

A separate set of commands available for the LP Administrator is shown in Figure 7-3. 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**." The "**lp**" login is a system login.

You will also probably need to use the commands for disabling and enabling a printer, and the remaining commands described under the "Summary of User Commands" above.

Command	Description
`/usr/lib/accept`(1M)	Permit job requests to be queued for a specified destination.
`/usr/lib/reject`(1M)	Prevent jobs from being queued for a specified destination. Described on the same manual page as `accept`(1M).
`/usr/lib/lpadmin`(1M)	Set up or change printer configurations.
`/usr/lib/lpfilter`(1M)	Set up or change filter definitions.
`/usr/lib/lpforms`(1M)	Set up or change preprinted forms. (Use `/usr/lib/lpadmin`(1M) to mount a form.)
`/usr/lib/lpmove`(1M)	Move output requests from one destination to another. Described on the same manual page as `lpsched`(1M).

Figure 7-3: Administrative Commands for the LP Print Service (Sheet 1 of 2)

Command	Description
/usr/lib/lpsched(1M)	Start the LP print service.
/usr/lib/lpshut(1M)	Stop the LP print service. Described on the same manual page as lpsched(1M).
/usr/lib/lpusers(1M)	Set or change the default priority and priority limits the users of the LP print service can request.

Figure 7-3: Administrative Commands for the LP Print Service (Sheet 2 of 2)

In Figure 7-3 the administrative commands are listed in the order in which they appear in the *UNIX System V User's* and *System Administrator's Reference Manual*. In the sections that follow, the commands are described in the order that they are typically used to handle the tasks you will face as you set up the LP print service to meet your needs.

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. However, if 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 manually start and stop the LP print service you must be logged in as either the super-user **root** or the user **lp**.

Manually Stopping the Print Service

To manually stop the LP print service, enter the following command:

/usr/lib/lpshut

The message

```
Printservices
```

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

```
Printservices
```

Manually Starting the Print Service

To manually restart the LP print service, enter the following command:

/usr/lib/lpsched

The message

```
Printservices
```

will appear. It may take a minute or two for the printer configurations, forms, and filters to be re-established before any saved print requests start printing. If you try to restart the LP print service when it is already running, you will see the message

```
Printservices
```

Note: The LP print service does not have to be stopped to change printer configurations or to add forms or filters.

Printer Management

Before the LP print service can start accepting print requests, you will have to define the configuration of each printer you have. This section describes how to do that.

Defining the Configuration of a Printer

The following table lists the information that can be given to define the configuration of each printer.

Printer Configuration Information

printer name
interface program
printer type
content types
connection method
printer port characteristics
character sets or print wheels
forms allowed
fault alerting
fault recovery
use restrictions
banner necessary
description
default printing attributes

You need to give 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 be managed for you and the better it will be represented to the people using the LP print service.

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 have read all the sections before adding a printer, so that you can do it in one step.

Printer Name

The printer name and the connection method (described next) are the only items you must specify to define a new printer. The name is used to identify the printer, both by you when you want to change the printer configuration or manage the printer, and by people who want to use the printer to print a file. The name may contain no more than 14 characters, and can include numbers as well as letters, but no special characters other than underscore.

You can choose any names you like, but it is good to choose names that mean something to the users of the LP print service. For example, `laser` is a good name for a laser printer, but if you have several laser printers you may want to number them, such as `laser1`, `laser2`, and so on.

You do not have to try to fit a lot of descriptive information into the name; there is a better place for this information (see the "Description" section below). You also do not have to make the name precisely identify the type of printer—people who need to use a particular type of printer can specify it by type, not 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 so on. Thus, the first thing you must do to add a printer is identify its name. You will do this as shown below, but do not do it yet because you will also need to specify the connection method.

/usr/lib/lpadmin -p *printer-name*

There are no default names; you must name every printer.

Connection Method

The LP print service allows you to connect your printers in a variety of ways. The simplest way is to connect your printer directly to the computer. However, you may want to connect printers via a network or through a dialed modem, where they can be shared with other computers or workstations. Once you have connected the printer to the computer or to a network and you have connected the network to the computer, you should then describe the connection method for the LP print service.

The default method by which printers are connected to the computer is the direct connection method. If you have used this method to connect your printer to your computer, you generally need to do only one other thing,

name the connecting port. Some directly connected printers, however, can also be used as terminals for login sessions. If you want to use a printer as a terminal, you will have to arrange for the LP print service to handle it as such. To do so, use the **-l** option to the **lpadmin** command, as described below.

There are two methods of making nondirect connections: through a dial-up modem or over any other type of network. The LP print service uses the Basic Networking Utilities to handle both methods of nondirect connections. When a dial-out modem is used, three prerequisites must be satisfied: the printer must be connected via a dialed modem; a dial-out modem must be connected to the computer; and the Basic Networking Utilities must know about this modem.

Printers connected via any other type of network require that a "system name" be given for each printer. This is the name of an entry in the `Systems` file or related file. Although the printer is not a UNIX system, the `Systems` file can be used to record the access method. (No login information will be given.)

Because the **cu** program accesses a printer in the same way the LP print service does, you should set up the files as though preparing access to the printer for **cu**. The **cu** command is not used to access printers, but can serve as a yardstick when setting up files: if **cu** can access a printer, the LP print service will be able to access it, too.

Adding a Directly Connected Printer

To add a directly connected printer:

/usr/lib/lpadmin -p *printer-name* **-v** *path-name*

Path-name is the name of the special file representing the printer port. Typically this is one of the following files:

/dev/contty
/dev/tty11
/dev/tty12
/dev/tty13
/dev/tty14
/dev/tty15
etc.

Adding a Printer to be Used as a Login Terminal

To add a directly connected printer to your system for use as a login terminal:

/usr/lib/lpadmin -p *printer-name* **-v** *path-name* **-l**

As before, *path-name* is the name of the special file representing the printer port. The `-l` indicates that the printer should be automatically disabled when the LP print service is started, to allow people to log in. The printer/terminal will have to be manually enabled before it can be used for printing. See the section "Enabling and Disabling a Printer" in this chapter for information.

Adding a Printer Connected Via a Modem or Network

To add a printer that is connected via a modem or network:

/usr/lib/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.

You must enter an `lpadmin` command with either the `-U` or `-v` option. And, unless you give the `-l` option, the LP print service will assume the printer is not to be used as a login terminal.

A note on dial-out or network printers: 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. The rate is not adjustable. However, you can force an immediate retry by issuing an **enable** command for the printer. If the port or printer is likely to be busy for an extended period, you should issue a **disable** command.

The **lpstat -p** command reports the reason for a failed dial attempt. Also, if you are alerted to a dialing fault (see "Fault Alerting"), 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 called "BNU STATUS Error Messages" in Appendix C, "Error Messages," for an explanation of the reasons for failure.

Interface Program

This is the program the LP print service uses to manage the printer each time a file is printed. It has four main tasks:

- To initialize the printer port (the connection between the computer and the printer)
- To initialize the printer (restore it to a normal state if 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 to print the file.

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 the "Customizing the Print Service" section for details on how to customize an interface program.

If you will be using the standard interface program, you need not specify it when adding a printer. However, if you will be using a different interface program, you can either refer to it by its full path name or by referring to another printer using the same interface program.

Note: You can also specify an interface program by naming a model interface program. The models provided with the old LP print service have been carried over, *but they do not support all the new features.* These models will be phased out in a future release.

To identify a customized interface program by name, give the printer name and the path name of the interface program as follows:

/usr/lib/lpadmin -p *printer-name* **-i** *path-name*

To identify a customized interface program by reference to another printer, give the printer names as follows:

/usr/lib/lpadmin -p *printer-name*[1] **-e** *printer-name*[2]

Printer-name[1] should be replaced with the name of the printer you are adding; *printer-name*[2] should be replaced with the name of the printer already added that is using the customized interface program.

To identify an interface program by reference to a model interface program, give the printer name and model name as follows:

/usr/lib/lpadmin -p *printer-name* **-m** *model-name*

Printer Type

The printer type is important for the proper use of the printer. The LP print service uses the printer type to extract information about the printer from the Terminfo data base. This information describes the capabilities of the printer so that you can be warned if some of the configuration information you provide is not appropriate for the printer. The information also describes the control data to use to initialize the printer before printing a file. While you are not required to specify a printer type, you are urged to specify one so that better LP print services will be provided.

The printer type is the generic name for the printer. Typically, it is derived from the manufacturer's name, such as `495` for the AT&T 495 Laser Printer. The "Acceptable Terminal Names" section of Appendix F in the *UNIX System V User's Guide* provides a description of how to determine a correct `TERM` variable for a user terminal, and can be used as a guide for picking an acceptable name for your printer.

Specify the printer type as follows:

/usr/lib/lpadmin -p *printer-name* **-T** *printer-type*

If you do not define the printer type, the default `unknown` will be used. This will produce empty results when the LP print service looks up information about the printer, so the LP print service will not be able to verify certain requests or initialize the printer.

Content Types

While the printer type information tells the LP print service what type of printer is being added, the content type information tells the LP print service what types of files can be printed. Most printers can print only one type of file; for them, the content type is likely to be identical to the printer type. Some printers, though, can accept several different types of files and print

their content properly. When adding this kind of printer, you should list the names of the content types it accepts.

When a file is submitted to the LP print service for printing, the LP print service searches for a printer capable of handling the job. The LP 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.

Content type names may look a lot like printer type names, but you are free to choose names that mean something to you and the people using the printer. (The names `simple`, `terminfo`, or `any` are recognized as having particular meanings by the LP print service; be sure to use them consistently.) The names must contain no more than 14 characters and may include only letters, digits, and underscores. If the same content type is printable by several different types of 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.

For example, several manufacturers may produce printers that accept PostScript files. While these printers may need different printer types so that each can be properly initialized (assuming the initialization control sequences are different), they may all be capable of handling the same type of input file which you may call `postscript`. Another example would be that several manufacturers may produce printers that accept ANSI X3.64 defined escape sequences. However, the printers may not support all ANSI capabilities, or may support different sets of capabilities. You may want to give different content type names for these printers, to differentiate them.

You do not have to list the content types for a printer. If you do not list them, 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 that the printer can print only files of content type `simple`. This may be sufficient if you will require people to pick the proper printer and make sure the files are properly prepared for the printer before submitting them for printing.

One type of file often encountered on UNIX systems is called `simple`. This file is assumed to contain just 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, it is 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 "How to Define Printer Ports and 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 actions similar to those performed by a carriage apply. If a printer can handle a `simple` type of file, you should include it in the content type list when you add the printer and specify the content type(s) the printer can handle. If you *do not* want a printer to accept files of type **simple**, you must give an alternate list of content types the printer can accept. (The printer type is a good name to use if no other type is appropriate.)

Another content type name is `terminfo`. This does not refer to a particular type of file but instead refers to all the types represented in the Terminfo data base. It is not likely that any printer is capable of handling all the types listed in the data base. However, this name is reserved for describing possible filter capabilities. Likewise, the content type `any` is describing the types of files a filter can accept or produce. These names should not be used as content types when adding a printer.

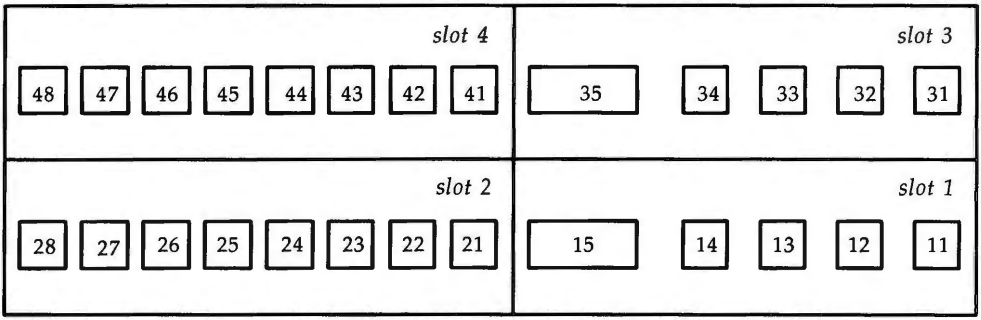
Specify the list of content types as follows:

/usr/lib/lpadmin -p *printer-name* **-I** *content-type-list*

The *content-type-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 `-I`) in quotes. If you do not define the types of files a printer can accept, the LP print service will assume it can take type **simple** and a type with the same name as the printer type (if the printer type is defined).

How to Define Printer Ports and Printer Port Characteristics

In all 3B2 computers, the name of an I/O port assignment is made up of two digits: the number of the slot on the computer in which the I/O card is installed and the number of the port on the card. The following diagram shows a sample of four slots (for Enhanced Ports cards and Expanded I/O Ports cards).



Slot assignments in this example are soley for the purpose of illustrating the relationship of slots to I/O port identification and are not representative of Version 3 hardware slot equipage. In this example, slots 1 and 3 contain Expanded I/O Ports cards and slots 2 and 4 contain Enhanced Ports (EPORTS) cards. 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. Refer to the Owner/Operator Manual for more information about slot numbering. The slot number is the first digit 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 port number as the second digit of the port assignment number.

By counting in this way, you can determine from the example that the first serial port on the EPORTS card in slot number 2 is "tty21." Similarly, the parallel port on the Expanded I/O Ports card in slot number 1 is designated "tty15." (For more details about Expanded I/O Ports cards, see the *AT&T 3B2 Computer Expanded Input/Output Capability Manual.* For more information about EPORTS cards, see the *AT&T 3B2 Computer Enhanced Ports Manual.*)

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 (software) flow control or hardware flow control; 7, 8, or other bits per byte; style of parity; and output post-processing. The standard interface program supports software flow control and uses the **stty**(1) command to initialize the printer port, minimally setting the baud rate and a few other default characteristics. Hardware flow control is supported using EPORTS cards and requires changes to the standard interface program.

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:
-oluc	do not map lowercase to uppercase
onlcr	map linefeed into carriage-return/linefeed
-ocrnl	do not map carriage-return into linefeed
-nocr	output carriage-returns even at column 0
nl0	no delay after linefeeds
cr0	no delay after carriage-returns
tab0	no delay after tabs
bs0	no delay after backspaces
vt0	no delay after vertical tabs
ff0	no delay after 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**(1) command in the *UNIX System V User's and System Administrator'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**(1) 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 can specify an additional list of port characteristics that should be applied when printing each user's file. The list you give will be applied after the default list, so that you do not need to include in your list default items that you do not want to change. Specify the additional list as follows:

/usr/lib/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*. If you do not include alternate printer port characteristics, the default list in the table will be used.

For example, suppose your printer is to be used for printing graphical data, where linefeed characters should be output alone, without an added carriage return. You would enter the following command:

/usr/lib/lpadmin -p *printer-name* **-o "stty=-onlcr"**

Note that the single quotes are omitted because there is only one item in the list.

Another example is suppose your printer requires odd parity for data sent to it. You would enter the following command:

/usr/lib/lpadmin -p *printer-name* **-o "stty='parenb parodd cs7'"**

Character Sets or Print Wheels

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. The LP print service, with your help, can minimize the impact of these differences on the users of the LP print service.

When adding a printer, you can specify what print wheels, font cartridges, or character sets are available with the printer. Only one of these is assumed to apply to each printer. For 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.

With UNIX system releases earlier than UNIX System V Release 3.2, the System Administration menus printer function (for adding and changing a printer configuration) is **NOT** able to distinguish between a printer that takes print wheels and one that has preprogrammed/loadable character sets. The **mount-form** and **umount-form** functions will **NOT** prompt the user for print wheels to be mounted or unmounted when a printer that takes print wheels is used.

If you are using a software release earlier than UNIX System V Release 3.2, use the **lpadmin** command to define or change the list of print wheels or aliases for character sets and to mount or unmount print wheels.

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 you 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 data base. If you are using AT&T UNIX System V Release 3.2.1 or a later release, you can use the following command to determine the names of the character sets listed in the Terminfo data base.

TERM=*printer-type* **tput csnm 0**

Printer-type is the name of the printer type in question. The name of the 0th character set (the character set obtained by default after the printer is initialized) should be printed. Repeat the command, using 1, 2, 3, and so on in place of the 0, to see the names of the other character sets. In general, the Terminfo names should closely match the names used in the user documentation for the printer. However, because not all manufacturers use the same names, the Terminfo names may differ from one printer type to the next.

Note: For the LP print service to be able to find the names in the Terminfo data base, 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:

/usr/lib/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:

/usr/lib/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 data base. 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 "=" sign is the name you choose as an alias of the character set.

Note: You do not have to provide a list of aliases for the character sets if the Terminfo names are adequate. You can refer to a character set by number, by Terminfo name, or by your alias.

For example, suppose your printer has two selectable character sets (#1 and #2) in addition to the standard character set (#0). The printer type is 5310. Enter the following commands to determine the names of the selectable character sets.

```
TERM=5310 tput csnm 1
english
TERM=5310 tput csnm 2
finnish
```

The words **english** and **finnish**, 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 and set #1. You enter the following command to define synonyms.

/usr/lib/lpadmin -p *printer-name* **-S "cs0=american, english=british"**

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 people cannot 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

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 some 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 **mail** command in the *UNIX System V 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 *UNIX System V 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 have not printed because the proper print wheel is not mounted.

In addition to the method of alerting, you can also set the number of requests that must be queued before you are alerted, and you can arrange for repeated alerts every few minutes until the print wheel is mounted. You can choose the rate of repeated alerts, or you can choose to receive only one alert per print wheel.

To arrange for alerting to the need to mount a print wheel, enter one of the following commands:

/usr/lib/lpadmin -S *print-wheel-name* **-A mail -Q** *integer* **-W** *minutes*
/usr/lib/lpadmin -S *print-wheel-name* **-A write -Q** *integer* **-W** *minutes*
/usr/lib/lpadmin -S *print-wheel-name* **-A** *'command'* **-Q** *integer* **-W** *minutes*
/usr/lib/lpadmin -S *print-wheel-name* **-A none**

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 fourth command above directs the LP print service to never send you an alert when the print wheel needs to be mounted. The *integer* is the number of requests that need to be waiting for the print wheel, and the *minutes* is the number of minutes between repeated alerts.

Note: If you want mail sent or a message written to another person when a printer fault occurs, you will have to use the third command listed. Use the option **-A 'mail** *user-name'* or **-A 'write** *user-name'*

Once you start receiving repeated alerts, you can direct the LP print service to stop sending you alerts for the current case only, by giving the following command.

/usr/lib/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 do not define an alert method for a print wheel, you will not receive an alert for it. If you do define a method but do not give the `-W` option, you will be alerted once for each occasion.

Forms Allowed

Note: For a description of forms, see the "Forms" section in this chapter.

You can limit the use of preprinted forms on any printer. 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 the list of forms allowed or denied for a printer to warn you against mounting a denied form on the printer. However, you have the final word on this; the LP print service will not refuse such an attempt. The LP print service will refuse a user's request to print a file on a printer using a form denied on that printer, unless the form is already mounted.

If you try to list a form as allowed on 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 those users allowed or denied access to the `cron` and `at` facilities. See the description of the `crontab` command in the *UNIX System V User's Reference Manual*. Briefly, the rules are as follows:

1. An allow list is a list of forms that you are allowed to use with the printer. A deny list is a list of forms that you have been denied permission to use.
2. If the allow list is not empty, the deny list is ignored. If the allow list is empty, the deny list is used. If both lists are empty, there are no restrictions on which forms can be used.
3. Putting "any" or "all" into the allow list allows all forms; putting "any" or "all" into the deny list denies all forms.

You can add names of forms to either list using one of the following commands:

/usr/lib/lpadmin -p *printer-name* **-f allow:***form-list*
/usr/lib/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 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 add forms to the allow list or deny list, the LP print service will consider that the printer denies the use of all forms. It will, however, allow you to mount any form. It will also provide a warning message if the form is not in the allow list or if you are attempting to mount a form that does not match the capabilities of the printer, as described earlier.

Fault Alerting

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 shows 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: a drop in carrier and an XOFF not followed in reasonable time by an XON. However, you can add filters that can recognize any other printer fault indicators, and rely on the 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 **mail** command in the *UNIX System V User's Reference Manual* for a description of mail on the UNIX system.
- You can receive an alert written to the terminal on which you are logged in (any terminal). See the description of the **write** command in the *UNIX System V 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), will overcome this problem.

To arrange for alerting to a printer fault, enter one of the following commands:

/usr/lib/lpadmin -p *printer-name* **-A mail -W** *minutes*
/usr/lib/lpadmin -p *printer-name* **-A write -W** *minutes*
/usr/lib/lpadmin -p *printer-name* **-A** *'command'* **-W** *minutes*
/usr/lib/lpadmin -p *printer-name* **-A none**

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* is the number of minutes between repeated alerts. The fourth command above directs the LP print service to not send you an alert when a fault occurs.

Note: If you want mail sent or a message written to another person when a printer fault occurs, use the third command. Use the option **-A 'mail** *user-name'* or **-A 'write** *user-name'*.

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 giving the following command.

/usr/lib/lpadmin -p *printer-name* **-A quiet**

If the *printer-name* is `all` in any of the commands above, the alerting condition will apply to all printers.

If you do not define an alert method, you will receive mail once for each printer fault. If you do define a method but do not give the `-W` option, you will be alerted once for each fault.

Fault Recovery

Once a printer fault has been detected and you have been alerted, you will probably fix the fault and get the printer ready for printing. When the printer is ready for printing again, the LP print service will recover in one of three ways:

- Continue printing at the top of the page where printing stopped.
- Restart printing at the beginning of the print request that was active when the fault occurred.
- 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:

/usr/lib/lpadmin -p *printer-name* **-F continue**
/usr/lib/lpadmin -p *printer-name* **-F beginning**
/usr/lib/lpadmin -p *printer-name* **-F wait**

These 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 continue 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.

Restricting User Access

You can limit the use of a printer to a subset of all people on your computer. You may want to do this, for instance, if a printer is being set aside for printing sensitive information and only a subset of the people can print sensitive information, or if use of a high-quality printer incurs expenses not all people are allowed to incur.

The LP print service will use the list of users allowed or denied for a printer to restrict use of the printer. The LP print service will refuse a user's request to print a file on a printer he or she is not allowed to use.

The method of listing the users allowed or denied for a printer is similar to the method used to list users allowed or denied access to the `cron` and `at` facilities, and the method described above in the "Forms Allowed" section. Briefly, the rules are as follows:

1. An allow list is a list of those users allowed to use the printer. A deny list is a list of those users denied access to the printer.
2. If the allow list is not empty, the deny list is ignored. If the allow list is empty, the deny list is used. If both lists are empty, there are no restrictions on who can use the printer.
3. Putting "any" or "all" into the allow list allows everyone to use the printer; putting "any" or "all" into the deny list denies everyone, except the user "**lp**" and the super-user "**root**".

You can add names of users to either list using one of the following commands:

/usr/lib/lpadmin -p *printer-name* **-u allow:***user-list*
/usr/lib/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. The first command adds the names to the allow list and removes them from the deny list. The second command adds the names to the deny list and removes them from the allow list. Using `allow:all` will allow everyone, using `deny:all` will deny everyone.

If you do not add user names to the allow or deny lists, the LP print service will assume that everyone can use the printer.

Banner Necessary

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 can 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 a user to skip the banner page:

/usr/lib/lpadmin -p *printer-name* **-o nobanner**

If you later change your mind, you can reverse this choice by entering the following command.

/usr/lib/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.

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 would 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.

People can see the message when they use the **lpstat -D -p** *printer-name* command.

To add a description when adding a printer, enter the following command.

/usr/lib/lpadmin -p *printer-name* **-D** *'text'*

The *text* is the message. You will need to include the quotes if the message contains blanks or other characters that the shell might interpret if the quotes are left out. Unless you give a printer description, none will be presented to people who ask about it.

Default Printing Attributes

When a user submits a request to print a file, page size, character pitch, and line pitch (i.e., print spacing) are normally determined from the form that will be printed on. If the user does not require a form, he or she can specify the page size and print spacing to be used. However, if he or she specifies neither a form nor the page size and print spacing, defaults will be used.

You can set 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 columns 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:

/usr/lib/lpadmin -p *printer-name* **-o width**=*scaled-number*
/usr/lib/lpadmin -p *printer-name* **-o length**=*scaled-number*
/usr/lib/lpadmin -p *printer-name* **-o cpi**=*scaled-number*
/usr/lib/lpadmin -p *printer-name* **-o lpi**=*scaled-number*

Add 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 do not provide defaults, then the page size and print spacing will be those available when the printer is initialized. You can find out what the defaults will be by first defining the printer configuration without providing your own defaults, then using the `lpstat` program to display the printer configuration. The command

lpstat -p *printer-name* **-l**

will report the default page size and print spacing. If you have not provided the defaults, the reported defaults will be calculated from the Terminfo data base entry for the printer. Obviously, this requires you to have provided a printer type in the printer configuration.

Adding a Printer to a Class

It is occasionally convenient to treat a collection of printers as a single class. The benefit is that a person can submit a file for printing by a member of a class, and the 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 are not needed if the only purpose is to allow a user to submit a print request by type of printer. The **lp -T** *type* command allows a user to submit a file and specify its type. The first available printer that can handle the type of file will be used to print the file. 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. See the "Filter Management" 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 does not belong to one. If you want to do so, use the following command:

/usr/lib/lpadmin -p *printer-name* **-c** *class-name*

If the class *class-name* does not exist yet, it will be created.

Note: Class names and printer names must be unique. If they are, a user can specify the destination for a print request without having to know whether it is a class of printers or a single printer. Thus, you cannot have a class and printer with the same name.

Setting the 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:

/usr/lib/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.

/usr/lib/lpadmin -d

If you do not set a default destination, there will be none. Users will have to explicitly name a printer or class in each print request, or they will have to set the `LPDEST` shell variable with the name of a destination.

Mounting a Form or Print Wheel

Note: See the "Forms" section in this chapter for information about preprinted forms.

Before the LP print service can start printing files that need a preprinted form or print wheel, you will have to mount the form or print wheel on a printer. 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.

When you mount a form you may want to see if it is lined up properly. If an alignment pattern has been registered with the form, you can ask that this be repeatedly printed after you have mounted the form, until you have adjusted the printer so that the alignment pattern looks correct.

Mounting a form or print wheel involves 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 is 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 below.
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.)

When you have loaded the new form or print wheel into the printer, enter the following command to tell the LP print service to mount it. (This command is shown on two lines for readability; it must be entered as one line.)

/usr/lib/lpadmin -p *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 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 registered, you will not be asked to press the key. You can drop the `-a` and **-o filebreak** options if you do not want to bother with the alignment pattern.

The **-o filebreak** option tells the LP print service to add a "formfeed" after each copy of the alignment pattern. The control sequence used for the "formfeed" depends on the printer involved and is obtained from the Terminfo data base. If the alignment pattern already includes a formfeed, omit the **-o filebreak** option.

If you want to unmount a form or print wheel, use the following command:

/usr/lib/lpadmin -p *printer_name* **-M -S none -f none**

Omit the **-S none** if you just want to unmount a form; likewise, omit the **-f none** if you just want to unmount a print wheel.

Until you have mounted a form on a printer, only print requests that do not require a form will be sent to it. Likewise, until you have mounted a print wheel on a printer, only print requests that do not require a particular print wheel will be sent to it.

Removing a Printer or Class

You can remove a printer or class if it has no pending print requests. If there are pending requests, you have to move them to another printer or class (using the `lpmove` command) or remove them (using the `cancel` command).

Removing the last printer of a class automatically removes the class as well. However, removing a class does not cause the removal of printers that were members of the class. If the printer or class removed is also the system default destination, the system will no longer have a default destination.

To remove a printer or class, enter the following command:

/usr/lib/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:

/usr/lib/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` on printer port `/dev/tty13`. 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. (This example is split into two lines for readability.)

```
/usr/lib/lpadmin -p lp1 -v /dev/tty13 -T 455
                 -o "width=90 length=71 stty=-onlcr"
```

Example 2

Add a new printer called `laser` on printer port `/dev/tty41`. It should use a customized interface program, it can handle three file types—**i10**, **i300**, and **impress**—and it may be used only by the users `doceng` and `docpub`.

```
/usr/lib/lpadmin -p laser -v /dev/tty41 -i /usr/doceng/laser_intface
                 -I "i10,i300,impress" -u "allow:doceng,docpub"
```

Example 3

When adding the `lp1` printer in the first example, we forgot to set the alerting. We can do this now. We will have the LP print service alert us every 10 minutes after a fault until we fix the problem.

```
/usr/lib/lpadmin -p lp1 -A write -W 10
```

Accepting Print Requests for a New Printer

Initially, the LP print service will not consider a new printer eligible for printing files. This will give you time to make sure you have defined the printer configuration the way you want. When you are ready to make the printer available to others, you will have to tell the LP print service.

There are two steps in making a printer ready for use after you have defined the printer configuration. First, the LP print service will have to be told to accept print requests for the new printer. Second, the new printer will have to be enabled to print. These are separate tasks because you may have occasion to want to do one but not the other.

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:

/usr/lib/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.

Enabling and Disabling a Printer

When a printer is ready for use and the LP print service is accepting print requests for it, you must enable the printer before anything can be printed. Use the `enable` command to do this. Having the LP print service wait for you instead of automatically starting to print files lets you make sure that the correct form is loaded in the printer, that the correct print wheel or font cartridge is in place, and that the printer is on-line.

When all is ready, you should enter the following command to enable printing on a printer:

enable *printer-name*

Only printers are enabled for printing—not classes. If you want to enable several printers at once, list the printers, separated by spaces, on the same line as the `enable` command. Do not enclose the list in quotes.

At some point you may have to disable a printer. This should be done before you change the form or print wheel or whenever you want to interrupt a print request. Disabling a printer stops further print requests from being printed, but it does not stop the LP print service from accepting new print requests for the printer. Normally, disabling a printer also stops the request that is currently being printed, placing it back in the queue so it can be printed later. However, you can have the LP print service wait until the current request finishes or even cancel the request outright.

Enter one of the following commands to disable a printer:

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 makes the LP print service wait for the current request to finish, while the third cancels the current request. The *reason* is stored and displayed whenever anyone asks the status of the printer. You can omit it (and the `-r`) if you do not want to specify a reason.

Several printers can be disabled at once by listing their names in the same line as the `disable` command.

Allowing Users to Enable and Disable a Printer

You may want to make the `enable` and `disable` commands available for use by other people. This availability is useful, for instance, if you have a small organization where anyone who spots a problem with the printer should 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 LP 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 is 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

Examining a Printer Configuration

Once you have defined a printer configuration, you probably want to review it to determine if it is correct. If after examining the configuration you find you have made a mistake, just re-enter the command that applies to the part that is 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 determine if the printer exists and if it is busy, idle, or disabled. The long form of the command requests 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 .cWwaiting for auto-retry output shows that the LP print service failed in trying to use the printer (because of the *reason* shown), and that the LP print service 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: path-name
On fault: alert-method
After fault: fault-recovery
Users allowed: user-list
Forms allowed: form-list
Banner required Character sets: character-set-list
Default pitch: integer CPI, integer LPI
Default page size: scaled-decimal-number wide,
                   scaled-decimal-number long
Default port settings: stty-option-list
```

Troubleshooting

If you are having difficulty getting your printer to work, here are a few suggestions for what to do.

No Output - Nothing Prints

The printer is sitting idle; nothing happens. First, check the documentation that came with the printer to determine if there is a self-test feature you can invoke; make sure the printer is working before continuing.

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 are not 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. Set up the printer as described in the "Printer Management" section of this chapter. If you receive error messages when doing this, follow the "fixes" suggested in the messages. When you have finished setting up the printer, issue the commands

/usr/lib/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* **-T** *printer-type file-name*

If you have not specified a printer type for the printer, omit the **-T** *printer-type* option.

Is the Baud Rate Correct?

If the baud rate (the rate at which data is transmitted) is not the same for both the computer and the printer, sometimes nothing will print (see below).

Illegible Output

The printer tries printing, but the output is not what you expected; it certainly is not readable.

Is the Baud Rate Correct?

Usually, when the baud rate of the computer does not match that of the printer, you will 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 does not matter for now. If it is not 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 does not matter.

To set a different baud rate for the LP print service to use, enter the following command:

/usr/lib/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 Card?

Flow Control

There are two ways of implementing flow control: software and hardware. The EPORTS card supports hardware flow control and software flow control. However, the hardware and software flow control modes are mutually exclusive for a given port (ttyxx). Before the **epstty**(1) command can be used to enable hardware flow control, the **stty**(1) command must be used to disable **ixon**, **ixoff**, and **ixany**. Refer to the *AT&T 3B2 Computer Enhanced Ports Manual* for information on flow control.

Buffer Overflow

Even with flow control (hardware or software), it is possible to overflow the print buffer. Transmitted characters are stored temporarily by the printer in a buffer. When the buffer is filled to a specified threshold, the printer sends a signal to the EPORTS card, asking it to stop transmission. No more characters are sent until space is available in the buffer; then a signal is sent (from the printer to the EPORTS card) and transmission resumes.

The EPORTS card is more efficient than other types of ports cards; 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 signal to the EPORTS card quickly enough to stop transmission before it receives more characters than its buffer can hold. The result is a character overflow 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 buffer overflow 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 the problem 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 card. To stop buffer overflow, 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 expect to set the parity bit by default. You can change this, however, by entering one of the following commands:

/usr/lib/lpadmin -p *printer-name* **-o stty=oddp**
/usr/lib/lpadmin -p *printer-name* **-o stty=evenp**
/usr/lib/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. Select the command that matches what your printer needs.

If you are also setting a baud rate other than 9600, combine the baud rate setting with the parity settings, as in the sample command below.

/usr/lib/lpadmin -p *printer-name* **-o "stty='evenp 1200'"**

Both double and single quotes are needed.

Tabs Set Correctly?

If the printer does not 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" in this chapter.

Legible Printing, but Wrong Spacing

The output contains all the expected text and is 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.

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 **-tabs** option or **-onlcr** option, respectively.)

/usr/lib/lpadmin -p *printer-name* **-o stty=-tabs**
/usr/lib/lpadmin -p *printer-name* **-o stty=-onlcr**

No Left Margin/Runs Together/Jammed Up

The printer's tab settings are not correct; they should be set every 8 spaces. You can have the LP print service not send tabs by using the **-tabs** option.

/usr/lib/lpadmin -p *printer-name* **-o stty=-tabs**

Zig Zags Down the Page

The **onlcr** option is needed. This is set by default, but you may have cleared it accidentally.

/usr/lib/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.

/usr/lib/lpadmin -p *printer-name* **-o "stty='-onlcr -tabs 2400'"**

Both double and single quotes are needed.

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 a problem described above. Another result may be that the wrong control characters cause the printer to set the wrong character set or font.

If you do not know what printer type to give, try the following to examine the available printer types. First, if you think the printer type has a certain name, try the following command:

TERM=*printer-type* **tput 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 do not know what names to try, you can examine the */usr/lib/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/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:

/usr/lib/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 have not arranged to receive fault alerts, they are mailed, by default, to the user **lp**.) See Appendix C, "Error Messages" for "BNU STATUS Error Messages."

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 determine 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 does not 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 cannot 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*

- Lost "child process." If the UNIX system process controlling the printer is killed (by the UNIX system during periods of extremely heavy load, or by an administrator), the LP print service may not realize it for a few minutes. Disabling the printer and then re-enabling it will force the LP print service to check for the controlling process, and restart one. Make sure the printer is really idle, though, because disabling a printer stops it in the middle of printing a request. Though the request will not be lost, it will have to be reprinted in its entirety.

  disable *printer-name*
  enable *printer-name*

  If the process that is lost is one controlling a slow filter, do not try re-enabling the printer; instead, put the print request (the one at the head of the queue for the printer) on hold and then resume it, as shown below.

  lpstat -o -l
  lp -i *request-id* **-H hold**
  lp -i *request-id* **-H resume**

  Use the first command to list the requests queued.

Managing the Printing Load

Occasionally you may need to stop accepting print requests for a printer or move print requests from one printer to another. There are various reasons 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 have 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 people 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 does not care which printer is to be used. Such requests will not be automatically moved; if you do not 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:

/usr/lib/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 do not 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 denying requests has been corrected or changed, enter the following command to start accepting new requests:

/usr/lib/accept *printer-or-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 have to move requests from one printer or class to another, enter one of the following commands:

/usr/lib/lpmove *request-id printer-name*
/usr/lib/lpmove *printer-name*[1] *printer-name*[2]

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 named. The latter command moves from the first printer to the second printer *all* requests currently queued for the first printer. When the latter command is used, the LP print service also stops accepting requests for the first printer (the same result you would obtain by running the **reject** command).

Examples

Here are some examples of how you might use these three commands:

Example 1

You have decided it is time to change the ribbon and to do some preventive maintenance on printer `lp1`. First, to prevent the loss of print requests, you move all requests for printer `lp1` to printer `lp2`. After the requests are moved, the LP print service no longer accepts requests for `lp1` (the same result you would obtain by running the `reject lp1` command before the **lpmove** command).

```
/usr/lib/lpmove lp1 lp2
```

(At this point you may disable the printer and start working on it.)

Example 2

You have finished changing the ribbon and doing the other work on `lp1`; now it is time to bring it back into service.

```
/usr/lib/accept lp1
```

(At this point, if you had disabled the printer you should re-enable it. See the "Enabling and Disabling a Printer" section under "Managing Printers" 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.

```
/usr/lib/lpmove laser1-23 laser1-46 laser2
/usr/lib/reject -r "too busy--will reopen later" laser1
```

Managing Queue Priorities

The LP print service provides a simple priority mechanism that people can use to adjust the position of a print request in the queue. Each print request can be given a priority level by the person who submits it; this is a number from 0 to 39, with *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 person who decides that his or her print request is of low priority can assign it a larger value when he or she submits the file for printing. Another person who decides that his or her print request is of high priority can assign it a smaller value when he or she 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 of this scheme:

- Each user can be assigned a priority limit. One cannot submit a print request with a priority higher than his or her limit, although one can submit a request with a lower priority.
- A default priority limit can be assigned for the balance of users not assigned a personal limit.
- A default priority can be set. This is the priority given 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 can have it put on "hold" another print request. This will allow the first request to be printed and will delay the latter 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 now discussed in detail.

Setting Priority Limits

To set someone's priority limit, enter the following command:

/usr/lib/lpusers -q *priority-level* **-u** *user-name*

You can set the limit for a group of people 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 *priority-level* is a number from 0 to 39. As mentioned before, the lower the number the higher the priority or, in this case, the priority limit.

If you want to set a priority limit for all other users, enter the following command:

/usr/lib/lpusers -q *priority-level*

This sets the default limit; the default applies to those people 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 re-enter 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:

/usr/lib/lpusers -u *user-name*

Again, you can do this for more than one person at a time by including a list of names. Using the `lpusers` command with just the `-u` option puts the users in the "default limit" category.

Setting a Default Priority

To set the default priority (the priority level assigned to print requests submitted without a priority), use the following command:

/usr/lib/lpusers -d *priority-level*

Do not confuse this default with the "default limit." This default is applied when a user does not specify a priority level; the "default limit" is applied if you have not 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:

/usr/lib/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.
- You 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 *requestid* **-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 can also put his or her own request on hold and then resume it, but cannot 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. Normally, printing begins on page one, but if you prefer, you can have printing begin on a later page. Enter the following command to resume the request on a different page:

lp -i *request-id* **-H resume -P** *starting-page-*

The final dash is needed to specify the starting page and all subsequent pages.

Note: The ability to print a subset of pages requires the presence of a filter that can do so; the default filter used by the LP print service does not. An attempt to resume a request on a later page will be rejected if an appropriate filter is not used.

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 your request to start printing immediately but another request is currently being printed, you can 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.

Forms

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 an old form.
- Remove a form.
- Examine a form.
- Restrict user access to a form.
- Arrange alerting to the need to mount a form.
- Mount a form.

But before getting into the details, let's see what a form means in the context of the LP print service.

What is a Form?

A Form is a preprinted template that provides blank spaces when necessary information is to be filled in with particulars. Common examples of forms include:

- Blank checks
- Vouchers
- Receipts
- Labels
- Company letterhead
- Special paper stock.

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 a file that will be printed on the form, thereby filling it out.

The LP print service helps you manage the use of preprinted forms, but it 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.

Of course, if you do not use special forms for printing, you can skip this section.

Defining a Form

When adding a new form, the first thing you have to do is to 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 multipage 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 columns, inches, or centimeters.

Number of pages
: The number of pages in a multipage 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 can 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 comment you wish to make about the form. This comment is available for people to see so they can 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 examine this sample to see if the printing is lined up properly on the form. If it is not you can adjust the printer to get it lined up.

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 (i.e., the user "**lp**" and the super-user "**root**") can read it.

When you have 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: comment
Alignment pattern: alignment-pattern
```

With two exceptions, the information can 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 do not 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
Character pitch	10
Character set choice	any
Ribbon color	any
Comment	(no default)
Alignment pattern	(no default)

Use one of the following commands to define the form.

/usr/lib/lpforms -f *form-name* **-F** *file-name*
/usr/lib/lpforms -f *form-name* **-**

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 14 or fewer letters, digits, and underscores.

If you need to change a form, just re-enter one of the same commands. You need only provide information for items that must be changed; items for which you do not specify new information will stay the same.

Removing a Form

The LP print service imposes no fixed limit on the number of forms you can define. It is a good idea, however, to remove forms that are no longer appropriate. If you do not, 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 LP print service.

To remove a form, enter the following command:

/usr/lib/lpforms -f *form-name* **-x**

Restricting User Access

If your system has a form that you do not want to make available to everyone, you can limit its availability to a subset of all people on your computer. 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 list 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 refuse his or her request to print a file with it.

The method of listing the users allowed or denied access to a form 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 *UNIX System V User's Reference Manual*.) Briefly, the rules are as follows:

1. An allow list is a list of those users allowed to use the form. A deny list is a list of those users denied access to the form.
2. If the allow list is not empty, the deny list is ignored. If the allow list is empty, the deny list is used. If both lists are empty, there are no restrictions on who can use the form.
3. Putting "any" or "all" into the allow list allows everybody to use the form; putting "any" or "all" into the deny list denies everybody, except the user "**lp**" and the super-user "**root**."

You can add names of users to either list using one of the following commands:

/usr/lib/lpforms -f *form-name* **-u allow:***user-list*
/usr/lib/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. The first command adds the names to the allow list and removes them from the deny list. The second command adds the names to the deny list and removes them from the allow list. Using `allow:all` will allow everybody; using `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 can 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 type of form? One method would be to periodically monitor the number of print requests pending for a particular 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 **mail** command in the *UNIX System V 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 *UNIX System V 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 if any print requests have not printed because the proper form is not 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 you can choose to receive only one alert per form.

To arrange for alerting to the need to mount a form, enter one of the following commands:

/usr/lib/lpforms -f *form-name* **-A mail -Q** *integer* **-W** *minutes*
/usr/lib/lpforms -f *form-name* **-A write -Q** *integer* **-W** *minutes*
/usr/lib/lpforms -f *form-name* **-A** *'command'* **-Q** *integer* **-W** *minutes*
/usr/lib/lpforms -f *form-name* **-A none**

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 current directory. The fourth command above directs the LP print service to never send you an alert when the form needs to be mounted. The *integer* is the number of requests that needs to be waiting for the form and the *minutes* is the number of minutes between repeated alerts.

Note: If you want mail sent or a message written to another person when a printer fault occurs, use the third command listed, specifying the option **-A 'mail** *user-name*' or **-A 'write** *user-name*'.

Once 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:

/usr/lib/lpforms -f *form-name* **-A quiet**

Once the form has been mounted and then unmounted, 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.

If you do not define an alert method for a form, you will not receive an alert to mount it. If you do define a method but do not give the `-W` option, you will be alerted once for each occasion.

Mounting a Form

Refer to the "Mounting a Form" section under "Defining the Configuration of a Printer" in this chapter.

Examining a Form

Once you have added a form definition 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 definition of the form. (The display produced by **lpforms** can be used as input, so 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.

/usr/lib/lpforms -f *form-name* **-l**
/usr/lib/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.

Filter Management

This section explains how you can manage the use of filters with the LP print service. You will see how you can do the following:

- 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 performs one or more of three related roles:

- It may convert a user's file into a data stream that can be printed properly on a given printer.
- It may handle the special modes of printing that people may request with the `-y` option to the `lp` command (such as two-sided printing, landscape printing, draft or letter-quality printing).
- It may detect printer faults and notify the LP print service of them, so that the LP print service can alert you.

Not every filter will do all three roles. 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, 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 roles of a filter more closely.

Role 1: Converting Files

The LP print service allows you to classify each printer you add to the system as a particular "type," and it allows a user to identify each file he or she submits for printing as a "type." This type information is used to match a file with the printer that will best reproduce the file. Because many applications can generate data for various printers, this is often sufficient. However, some of the applications you use may not be able to generate output that can be printed by your printers.

By defining and creating a filter that converts such output into a type that your printers can handle, you can begin to support more applications in the LP print service. The Terminal Filters Utilities provide a small set of simple filters that convert output from applications such as `nroff` (from the DOCUMENTER'S WORKBENCH* software) to data streams that can be printed properly on some printers.

For each filter that is added to the system, the type of input it can accept and the type of output it can produce are listed. Now the LP print service can be more sophisticated in its attempt to match a user's file with a printer. If it cannot make a direct match, it consults the table of filters to find one that can convert the file type into the printer type. Below are some examples.

Example 1

The user Chris has run a spreadsheet program and 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 is 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.

---

* Registered trademark of AT&T

Example 2

The user Marty has run the `nroff` word processing program to produce a large document. Forgetting the fact that the `nroff` program understands how to generate output for several printers, Marty requests the default output type (let's call it type `nroff35`) which cannot be reproduced well on the AT&T 455. Fortunately, you have foreseen this situation and have added the `450` filter from the Terminal Filters Utilities to the filter table, marking it as a filter that takes standard **nroff** output (i.e., **nroff35)** and produces output for the AT&T 455 (let's call it type `455`). Because you have added the printer as a type `455`, the LP print service recognizes that it can use the `450` filter to convert Marty's output before printing it.

Role 2: Handling Special Modes

Another important role of filters 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
Input type
Output type
Printer type
Character pitch
Line pitch
Page length
Page width
Pages to print
Character set
Form name
Number of copies

A filter is required only to handle special modes; the LP print service provides a default handling for all the rest. However, it may be more efficient to have a filter handle these, 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. Another example is that 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 will see below how you can register special printing modes and other aspects of printing with each filter.

Role 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 LP print service can detect, 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 LP print service cannot determine the cause of a fault, so it cannot 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 dropping a carrier or shutting off 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 "Fault Alerting" section under "Printer Management" 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 will see any fault descriptions generated by the filter. If you have 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 filter:

1. Examine the kinds of files users will submit for printing that will require processing by the filter. If they stand alone (that is, if they do not reference other files that the filter will need), the filter is probably okay. Check also to see if the filter expects any files other than those submitted by a user for printing.
2. If referenced files are permitted in the files submitted for printing, or if the filter will need files other than those submitted by a user, then the filter, unable to access the additional files, is likely to fail. We suggest you not 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 path names *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.

Defining a Filter

When adding a new filter, the first thing you must do is define the characteristics of its use. There are seven characteristics you must define: input types, output types, printer types, printers, filter types, command, and options. Each of these is described below.

Input types — This is the list of file types that the filter can process. The LP print service does not 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, subject to a limit of 14 letters, digits, and dashes

(not underscores). Because the LP print service uses these names to match a filter with a file type, you should be consistent in the naming convention. For example, if more than one filter can accept the same input type, use the same name. These names should be advertised to your users so they know how to identify the type of 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 14 letters, digits, 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 to produce a new filter, if it finds that several passes by different filters are needed to convert a file. It is 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, but which can recognize several different types of files. (See the "Printer Type" section under "Printer Management" in this chapter.) 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.

Another example is that you may have two different models of printers that are listed as accepting the same types of files. However, because of 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. Since this filter is only needed for those printer types, you would list it as working only on type B printers. For most printers and filters you can ignore this part of the filter definition.

Printers

You may have some printers that, although they are 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, the 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" either because they incur little overhead in preparing a file for printing, or 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. Slow filters are the opposite. Filters that incur a lot of overhead in preparing a file and that do not require access to the printer should be labeled "slow." The LP print service runs slow filters in the background, without tying up a printer. This allows files that do not 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.

Command

This is the full path name of the program run as the filter. If there are any fixed options that the program always needs, include them here.

Options — Options that the filter program needs, depending on the special modes and other aspects of printing, can be registered with the filter. This is discussed in more detail below.

When you have gathered this information about the filter, enter it as input to the `lpfilter` command. You may want to record this information first in a separate file so you can edit it before entering it with **lpfilter**. 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:

Input types: *input-type-list*
Output types: *output-type-list*
Printer types: *printer-type-list*
Printers: *printer-list*
Filter type: fast *or* **slow**
Command: *simple-command*
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
Input types	any
Output types	any
Printer types	any
Printers	any
Filter type	slow
Command	(no default)
Options	(none)

As you can see, the default values define a 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.

Templates

The *"templates-list"* is a comma-separated list of templates of the following form:

keyword pattern = replacement

A *keyword* labels the template as registering a particular characteristic of the printing. A *pattern* is either a value of the characteristic or an asterisk (*), which serves as a placeholder for a value. Only the *keywords* shown in the following table may be used:

Characteristic	*keyword*	Possible *patterns*
Special modes	**MODES**	*mode*
Content type (input)	**INPUT**	*content-type*
Content type (output)	**OUTPUT**	*content-type*
Printer type	**TERM**	*printer-type*
Character pitch	**CPI**	*integer*
Line pitch	**LPI**	*integer*
Page length	**LENGTH**	*integer*
Page width	**WIDTH**	*integer*
Pages to print	**PAGES**	*page-list*
Character set	**CHARSET**	*character-set*
Form name	**FORM**	*form-name*
Number of copies	**COPIES**	*integer*

The sources of the values for these templates are 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, respectively. They will each be a type registered with the filter.
- The value for the `TERM` template is the printer type.

- 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 (e.g., 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 being printed on, if any.
- 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* 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 following examples show how this works.

Example 1

The filter program is called `/usr/bin/npf`. It takes two input types, `nroff37` and `X`, produces an output type called `TX`, and works with any printer of type `TX`. The program accepts three options:

-Xb	The input type `X`
-l *integer*	The length of the output page
-w *integer*	The width of the output page.

The filter definition would look like this:

```
Input types: X,nroff37
Output types: TX
Printer types: TX
Filter type: fast
Command: /usr/bin/npf
Options: INPUT X = -Xb, LENGTH * = -l*, WIDTH * = -w*
```

Let's say a user submits a file of type **nroff37**, requesting printing by a printer named `lp1` (which is of type **TX**) and a page length of `72`:

```
lp -T nroff37 -d lp1 -o length=72
```

The LP print service will call this filter to convert the file. The filter will be invoked as follows:

```
/usr/bin/npf -l72
```

Example 2

Another user submits a file of type `X` that is to be printed on the same printer, with default length and width. The filter will be invoked as follows:

```
/usr/bin/npf -Xb
```

Example 3

The filter program is called `/usr/bin/x9700`. It takes one input type, `troff`, produces an output type called `9700`, and works with any printer of type `9700`. The program has one fixed option, **-ib**, and accepts four other options:

-l *integer*	The length of the output page.
-s *name*	The character set.
-o portrait	Portrait orientation of the paper.
-o landscape	Landscape orientation of the paper.

You have decided that your users need give only the abbreviations `port` and `land` when they ask for the paper orientation. 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: 9700
Printer types: 9700
Filter type: fast
Command: /usr/bin/x9700 -ib
Options: LENGTH * = -l *, CHARSET * = -s *,
         MODES port = -o portrait, MODES land = -o landscape
```

(The description of **Options** should actually be entered as one line. It is shown on two lines here for readability.)

A user submitting a file of type `troff` for printing on a printer of type `9700`, with requests for landscape orientation and the `gothic` character set, would enter the following command:

```
lp -T troff -S gothic -y land
```

Then this filter would be invoked by the LP print service to convert the file as follows:

/usr/bin/x9700 -ib -s gothic -o landscape

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. A backslash in front of either character is removed when the *pattern* or *replacement* is used; all other backslashes are left alone.

Command to Enter

Once a filter definition is complete, enter one of the following commands to add the filter to the system.

/usr/lib/lpfilter -f *filter-name* **-F** *file-name*
/usr/lib/lpfilter -f *filter-name* **-**

The first command gets the filter definition from a file, and the second command gets the filter definition from you, through the standard input. A *filter-name* can be anything you choose, as long as it contains 14 or fewer letters, digits, and underscores.

If you need to change a filter, just re-enter one of the same commands. You need only provide information for those items that must be changed; items for which you do not 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:

/usr/lib/lpfilter -f *filter-name* **-x**

Examining a Filter

Once you have 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:

/usr/lib/lpfilter -f *filter-name* **-l**
/usr/lib/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.

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 users who submitted them). There can also be delays in the responses to new or changed print requests when filters are changed, because of 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 these possible delays, you may want to make changes to filters during periods when the LP print service is not being used much.

Directories and Files

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 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 section "Cleaning Out the Request Log."

All directories and files are found under the parent directory */usr/spool/lp*. This directory should have the following access permissions and ownership:

Permissions	Owner	Group	Directory or File
drwxrwxr-x	lp	bin	/usr/spool/lp

You can check this by entering the following command:

ls -ld /usr/spool/lp

Under this directory you should see only the directories and files shown in the table that starts on the following page. Those marked with an asterisk (*) may be missing, depending on the state of the LP print service or its configuration.

You can generate a similar table for comparison by entering this command:

ls -lR /usr/spool/lp

Permissions	Owner	Group	Directory or File
-rw-rw-rw-	lp	bin	* SCHEDLOCK
drwxrwxr-x	lp	bin	admins
drwxrwxr-x	lp	bin	bin
-rw-rw-r--	lp†	bin†	* default
drwxrwxr-x	lp	bin	fifos
drwxrwxr-x	lp	bin	logs
drwxrwxr-x	lp	bin	model
drwxrwxr-x	lp	bin	requests
drwxrwxr-x	lp	bin	system
drwxrwxr-x	lp	bin	temp
-rw-rw-r--	lp†	bin†	* users
/usr/spool/lp/admins:			
drwxrwxr-x	lp	bin	lp
/usr/spool/lp/admins/lp:			
drwxrwxr-x	lp	bin	classes
-rw-rw-r--	lp†	bin†	* filter.table
-rw-rw-r--	lp	bin	* filter.table.i
drwxrwxr-x	lp	bin	forms
drwxrwxr-x	lp	bin	interfaces
drwxrwxr-x	lp	bin	logs
drwxrwxr-x	lp	bin	printers
drwxrwxr-x	lp	bin	pwheels
/usr/spool/lp/admins/lp/classes:			
-rw-rw-r--	lp†	bin†	* *class1*
-rw-rw-r--	lp†	bin†	* *class2*
.			
.			
.			
-rw-rw-r--	lp†	bin†	* *classN*

Permissions	Owner	Group	Directory or File
/usr/spool/lp/admins/lp/forms:			
drwxrwxr-x	lp†	bin†	* *form1*
drwxrwxr-x	lp†	bin†	* *form2*
.			
.			
.			
drwxrwxr-x	lp†	bin†	* *formN*
/usr/spool/lp/admins/lp/forms/*formK*:			
-rwxrwx---	lp†	bin†	* alert.sh
-rw-rw----	lp†	bin†	* alert.vars
-rw-rw----	lp†	bin†	* align_ptrn
-rw-rw-r--	lp†	bin†	* allow
-rw-rw-r--	lp†	bin†	* comment
-rw-rw-r--	lp†	bin†	* deny
-rw-rw-r--	lp†	bin†	* describe
/usr/spool/lp/admins/lp/interfaces:			
-rwxrwxr-x	lp†	bin†	* *printer1*
-rwxrwxr-x	lp†	bin†	* *printer2*
.			
.			
.			
-rwxrwxr-x	lp†	bin†	* *printerN*

Permissions	Owner	Group	Directory or File
/usr/spool/lp/admins/lp/printers:			
drwxrwxr-x	lp†	bin†	* *printer1*
drwxrwxr-x	lp†	bin†	* *printer2*
.			
.			
.			
drwxrwxr-x	lp†	bin†	* *printerN*
/usr/spool/lp/admins/lp/printers/*printerK*:			
-rwxrwx---	lp†	bin†	* alert.sh
-rw-rw----	lp†	bin†	* alert.vars
-rw-rw-r--	lp†	bin†	* comment
-rw-rw-r--	lp†	bin†	* configuration
-rw-rw-r--	lp†	bin†	* forms.allow
-rw-rw-r--	lp†	bin†	* forms.deny
-rw-rw-r--	lp†	bin†	* users.allow
-rw-rw-r--	lp†	bin†	* users.deny
/usr/spool/lp/admins/lp/pwheels:			
drwxrwxr-x	lp†	bin†	* *printwheel1*
drwxrwxr-x	lp†	bin†	* *printwheel2*
.			
.			
.			
drwxrwxr-x	lp†	bin†	* *printwheelN*
/usr/spool/lp/admins/lp/pwheels/*printwheelK*			
-rwxrwx---	lp†	bin†	* alert.sh
-rw-rw----	lp†	bin†	* alert.vars

Permissions	Owner	Group	Directory or File
/usr/spool/lp/bin:			
-r--r--r-x	lp	bin	alert.proto
-rwxrwxr-x	lp	bin	drain.output
-rwxrwxr-x	lp	bin	lp.cat
-rwxrwxr-x	lp	bin	lp.page
-rwxrwxr-x	lp	bin	lp.set
-rwxrwxr-x	lp	bin	lp.tell
-rwxrwxr-x	lp	bin	lp.sched.jr
-rwxrwxr-x	lp	bin	slow.filter
/usr/spool/lp/fifos:			
p-w--w--w-	root	bin	*FIFO
drwxrwx--x	lp	bin	private
drwxrwx-wx	lp	bin	public
/usr/spool/lp/fifos/private:			
pr--------	user	group	
.			
.			
.	*mach PID		
/usr/spool/lp/fifos/public:			
pr--------	user	group	
.			
.			
.	*mach PID		
/usr/spool/lp/logs:			
-rw-rw----	lp	bin	* lpsched
-rw-rw----	lp	bin	* requests
-rw-rw----	lp	bin	* requests1
-rw-rw----	lp	bin	* requests2
.			
.			
.			
-rw-rw----	lp	bin	* requests*N*

Permissions	Owner	Group	Directory or File
/usr/spool/lp/model:			
-rwxrwxr-x	bin	bin	1640
-rwxrwxr-x	bin	bin	5310
-rwxrwxr-x	bin	bin	dqp10
-rwxrwxr-x	bin	bin	dumb
-rwxrwxr-x	bin	bin	f450
-rwxrwxr-x	bin	bin	hp
-rwxrwxr-x	bin	bin	lqp40
-rwxrwxr-x	bin	bin	pprx
-rwxrwxr-x	bin	bin	prx
-rwxrwxrwx	bin	bin	standard
/usr/spool/lp/requests:			
-rw-rw----	lp	bin	* *id1*-0
-rw-rw----	lp	bin	* *id2*-0
.			
.			
.			
-rw-rw----	lp	bin	* *idN*-0
/usr/spool/lp/system:			
-rw-rw-r--	lp	bin	* cstatus
-rw-rw-r--	lp	bin	* pstatus
/usr/spool/lp/temp:			
-rw-------	lp	bin	* *idN*-0
-rw-------	lp	bin	* *idN*-1
-rw-------	lp	bin	* *idN*-2
.			
.			
.			
-rw-------	lp	bin	* *idN-M*
-rw-------	lp	bin	* F*idN*-1
-rw-------	lp	bin	* F*idN*-2
.			
.			
.			

Permissions	Owner	Group	Directory or File
/usr/spool/lp/temp: (*Continued*)			
-rw-------	lp	bin	* F*idN-M*
-rw-------	lp	bin	* *idN-M*
-rw-------	lp	bin	*A-K
-rw-------	lp	bin	*F-K
-rw-------	lp	bin	*P-K

The italicized names, *printerN*, *formN*, *classN*, *printwheelN*, and *idN*, are placeholders for a single printer, form, class, print wheel, and request ID, 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 on your system. The italicized letter *K* is a placeholder for an internal number; the **A-***K*, **F-***K*, and **P-***K* files are used to store alert messages.

The ownership and permissions of the *idN-M* request files under the */usr/spool/lp/temp* 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 */usr/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 */usr/spool/lp/temp* and */usr/spool/lp/requests* 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 */usr/spool/lp/requests* directory where it can be kept secure: the request file in the */usr/spool/lp/temp* is safe from all except the user who submitted the request, while the file in */usr/spool/lp/requests* is safe from 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 */usr/spool/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 /usr/spool/lp/logs; if [ -f requests ]; then
  /bin/mv requests xyzzy; /bin/cp xyzzy requests; >xyzzy;
  /usr/lbin/agefile -c2 requests; /bin/mv xyzzy requests; fi
```

(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 2 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 system 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 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. 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).

Letter	Content of Line
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 LP print service, the most important bits are listed below: **0x0004** Slow filtering finished successfully. **0x0010** Printing finished successfully. **0x0040** The request was canceled. **0x0100** The request failed filtering or printing.
T	The title placed on the banner page.
t	The type of content found in the file(s).
U	The name of the user who submitted the print request.
x	The slow filter used for the request.
Y	The list of special modes to give to the filters used to print the request.
y	The fast filter used for the request.
z	The printer used for the request. This 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.

Customizing the Print Service

Although the LP print service has been designed to be flexible enough to handle most printers and printing needs, it does not handle every possible situation. You may buy a printer that does not 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 do not accommodate.

You can customize the LP print service in a few ways. This section tells you how to do the following:

- Adjust the printer port characteristics.
- Adjust the Terminfo data base.
- Write an interface program.
- Write a filter.

The diagram in Figure 7-4 gives an overview of the processing of a print request.

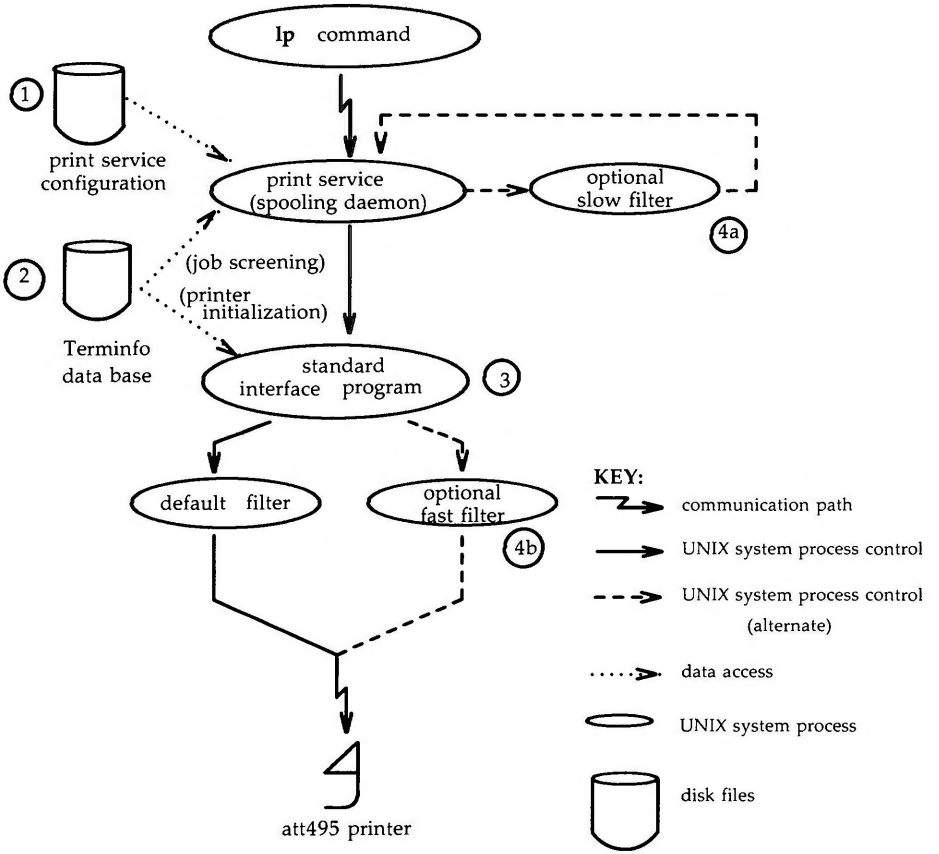


Figure 7-4: How LP Processes Print Request **lp -d att495** File

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 LP print service, adjust or replace some of the pieces shown in Figure 7-4. (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 is dependent on the printer includes the printer port characteristics: baud rate, parity, and so on.

2. For a printer that is not represented in the Terminfo data base, you can add a new entry that describes its capabilities. The Terminfo data base 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 data base 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 will not work with all files and printers. This is not 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 is an important step in getting your printer to work with the LP print service, so it is described in more detail here.

When you add a new printer, read the documentation that comes with it so that you understand what it expects from the host (the LP print service). Then read the manual page for the `stty`(1) command in the *UNIX System V User's Reference Manual*. It summarizes the various characteristics that can be set on a terminal or printer port.

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.

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 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	Do not 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	Do not 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	Do not send a carriage return before every linefeed.
ocrnl	Change carriage returns into linefeeds.
-ocrnl	Do not change carriage returns into linefeeds.
-tabs	Change tabs into an equivalent number of spaces.
tabs	Do not change tabs into spaces.

Adjusting the Terminfo Data Base

The LP print service relies on a standard interface and the Terminfo data base 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 data base to add a new printer to the LP print service. Several entries for AT&T printers and other popular printers are delivered in Terminfo data base entries with the LP print service package.

Each printer is identified in the Terminfo data base with a short name; this name is identical to the 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 F in the *UNIX System V 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 do not, you may still be able to use the printer with the LP print service; however, you will not 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 data base. 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:	
daisy	Printer needs operator to change character set.
Numbers:	
bufsz	Number of bytes buffered before printing.
* **cols**	Number of columns in a line.
* **it**	Tabs initially every # spaces.
* **lines**	Number of lines on a page.
orc	Horizontal resolution in units per character.
orhi	Horizontal resolution in units per inch.
orl	Vertical resolution in units per line.
orvi	Vertical resolution in units per inch.
cps	Average print rate in characters per second.
Strings:	
* **cr**	Carriage return.
cpi	Change number of characters per inch.

Terminfo Item	Meaning
lpi	Change number of lines per inch.
chr	Change horizontal resolution.
cvr	Change vertical resolution.
csnm	List of character set names.
mgc	Clear all margins (top, bottom, and sides).
* **hpa**	Horizontal position absolute.
* **cud1**	Down one line.
* **cuf1**	Carriage right.
swidm	Enable double wide printing.
rwidm	Disable double wide printing.
* **ff**	Page eject.
* **is1**	Printer initialization string.
* **is2**	Printer initialization string.
* **is3**	Printer initialization string.
* **if**	Name of initialization file.
* **iprog**	Path name of initializing program.
* **cud**	Move carriage down # lines.
* **cuf**	Move carriage right # columns.
* **rep**	Repeat a character # times.
* **vpa**	Vertical position absolute.
scs	Select character set.
smgb	Set bottom margin at current line.
smgbp	Set bottom margin.
* **smgl**	Set left margin at current column.
smglp	Set left margin.
* **smgr**	Set right margin at current column.
smgrp	Set right margin.
smgt	Set top margin at current line.
smgtp	Set top margin.
scsd	Start definition of a character set.
* **ht**	Tab to next 8-space tab stop.

The items marked with a leading asterisk (*) are available on all releases of UNIX System V. The other items can be added only if you are using UNIX System V Release 3.2 or a later release.

Note: If you are running the LP print service on UNIX System V Release 3.1 or an earlier operating system, only the Terminfo items marked in the table are available. They are sufficient for initializing the printer, but not for setting page sizes and pitches, or selecting character sets.

To make a data base entry for a new printer, see details about the structure of the Terminfo data base in the **terminfo**(4) manual page (*UNIX System V Programmer's Reference Manual*).

Once you have made the new entry, you need to compile it into the data base 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 data base. 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 Write an 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 data base, 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

/usr/spool/lp/model/standard

What Does an 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**(1) command to do this.
- Initializing the physical printer. The generic interface program uses the Terminfo data base 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. This is done by the LP print service, which calls 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 an 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:

/usr/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	Full path name of a file to be printed.

When the interface program is invoked, its standard input comes from `/dev/null`, its standard output is directed to the printer port, and its standard error output is directed to a file that will be given to the user who submitted the print request.

The standard interface recognizes the following values in the 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 data base 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 data base 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**(1) command. The default list is used to establish a default port configuration; the additional list given to the interface program is used to change the configuration as needed.

The above options 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 data base.

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 data base 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:

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 be 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 nonprintable characters, or the request exceeds the printer capabilities). 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 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 does not 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 is where we set up the $LPTELL program to capture
 # fault messages.

 # Here is 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. 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 must interrupt the printing of a file at any time, it will "kill" the interface program with a signal 15 (see `kill`(1) and `signal`(2) in the *UNIX System V and System Administrator's Reference Manual* and *UNIX System V Programmer's Reference Manual* (optional), respectively). If the interface program dies from receipt of any other signal, the LP print service assumes that future print requests will not 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 **SIGHUP**, **SIGINT**, **SIGQUIT**, and **SIGPIPE** (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 is not 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 finished and the user who submitted the request is notified. Likewise, if a "slow" filter exits with a nonzero exit code, the print request is finished 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.
- The filter 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.
- The filter 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 nonzero 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 under "Filter Management" in this chapter for details.)

Chapter 8: TTY Management

Introduction

This chapter covers the following topics:

- The terms used in discussing TTY management
- How the TTY system works
- How to tell what line settings are defined
- How to create new line settings and hunt sequences
- How to change TTY line characteristics
- How to set terminal options.

Definition of Terms

The following terms are used in this chapter:

TTY	Derived from the near-classic abbreviation for teletypewriter, the term covers the whole area of access between the UNIX system and peripheral devices, including the system console. It shows up in commands such as **getty**(1M) and **stty**(1), in the names of device special files such as **/dev/tty21**, and in the names of files such as **/etc/gettydefs**, which are used by **getty**.
TTY line	The physical equipment through which access to the computer is made.
port	A synonym for TTY line.
line settings	A set of line characteristics.
baud rate	The speed at which data is transmitted over the line. A part of line settings.
mode	The characteristics of the terminal interface. A part of line settings. The TTY line and the terminal must be working in the same mode before communication can take place. Described in **termio**(7).

hunt sequence	A circular series of line settings such as different baud rates. During the login sequence, a user looking for a compatible connection to the computer can go from one setting to the next by sending a BREAK signal.
terminal options	Selectable settings that define the way a given terminal operates. Described in **termio**(7).

The TTY System

The remaining sections in this chapter describe how the TTY system operates and how you can administer it.

How the TTY System Works

A series of four processes (**init**(1M), **getty**(1M), **login**(1), **sh**(1)) connect a user to the UNIX system. **init** is a general process spawner that is invoked as the last step in the boot procedure. It spawns a **getty** process for each line that a user may log in on, guided by instructions in **/etc/inittab**. An argument required by the **getty** command is **line**. The TTY line argument is the name of a special file in the **/dev** directory. For a description of other arguments that may be used with **getty**, see the *User's and System Administrator's Reference Manual.*

A user attempting to make a connection generates a request-to-send signal that is routed by the hardware to the **getty** process for one of the TTY line files in **/dev**. (We are omitting how the signal gets from the user's terminal to the 3B2 computer.) The **getty** process responds by sending an entry from file **/etc/gettydefs** down the line. The **gettydefs** entry used depends on the **speed** argument used with the **getty** command. [In the SYNOPSIS of the **getty**(1M) command the argument name is **speed**, but it is really a pointer to the **label** field of a **gettydefs** entry.] If no **speed** argument is given, **getty** uses the first entry in **gettydefs**. Among the fields in the **gettydefs** entry (described later in this chapter) is the login prompt.

On receiving the login prompt, the user enters a login name and **getty** starts **login**, using the login name as an argument. The **login** issues the prompt for a password, evaluates the user's response, and, assuming the password is acceptable, calls in the user's shell as listed in the **/etc/passwd** entry for the login name. If no shell is named, **/bin/sh** is furnished by default. **login** also executes **/etc/profile**.

The **/bin/sh** executes the user's **.profile**, if it exists. The **.profile** often contains **stty** commands that reset terminal options that differ from the defaults. The connection between the user and the UNIX system has now been made.

How to Tell What Line Settings Are Defined

You have two ways to check line settings:

1. Through the System Administration menus, specifically the **sysadm**(1) **lineset** subcommand. **sysadm lineset** first shows the full range of line settings, then gives you the chance to examine a line in detail. (See Procedure 8.1, "Check TTY Line Settings".)
2. By looking directly in **/etc/gettydefs**.

The **/etc/gettydefs** file contains information used by the **getty**(1M) command to establish the speed and terminal settings for a line. The general format of the **gettydefs** file is:

```
label# initial-flags # final-flags #login-prompt #next-label
```

Figure 8-1 shows a few lines from a **gettydefs** file.

```
38400# B38400 HUPCL # B38400 SANE IXANY TAB3 HUPCL #login: #19200

19200# B19200 HUPCL # B19200 SANE IXANY TAB3 HUPCL #login: #9600

9600# B9600 HUPCL # B9600 SANE IXANY TAB3 HUPCL #login: #4800

4800# B4800 HUPCL # B4800 SANE IXANY TAB3 HUPCL #login: #2400

2400# B2400 HUPCL # B2400 SANE IXANY TAB3 HUPCL #login: #1200

1200# B1200 HUPCL # B1200 SANE IXANY TAB3 HUPCL #login: #300

300# B300 HUPCL # B300 SANE IXANY TAB3 HUPCL #login: #38400
```

Figure 8-1: The **gettydefs** Entries

The entries shown in Figure 8-1 form a single, circular hunt sequence; the last field on each line is the label of the next line. The next-label field for the last line shown points back to the first line in the sequence. The object of the hunt sequence is to link a range of line speeds. If you see garbage characters instead of a clear login prompt, entering a BREAK causes **getty** 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. The flag fields shown have the following meanings:

B300-B38400	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.

For a description of all **getty** flags, see **termio**(7).

How to Create New Line Settings and Hunt Sequences

You have two ways to do this.

1. Use the System Administration menus, specifically the **sysadm mklineset**(1) subcommand. The **sysadm mklineset** leads you through a series of prompts. Your responses make up the information for a new **gettydefs** entry. (See Procedure 8.2, "Make TTY Line Settings".)
2. By using **ed**(1) or **vi**(1) to edit **/etc/gettydefs**.

Create new lines for the **gettydefs** file by following the example shown above. Each entry in the file is followed by a blank line. After editing the file, run the command:

```
# /etc/getty -c /etc/gettydefs
```

This causes **getty** to scan the file and print the results on your terminal. If there are any unrecognized modes or improperly constructed entries, they are reported.

How to Modify TTY Line Characteristics

You have two ways to do change TTY line characteristics.

1. Use the System Administration menus, specifically the **sysadm modtty**(1) subcommand. The **sysadm modtty** subcommand leads you through a series of prompts. Your responses edit a "getty" entry in **/etc/inittab** (see Procedure 8.3, "Modify TTY Line Characteristics").
2. By using **ed**(1) or **vi**(1) to edit **/etc/inittab**.

The **/etc/inittab** file contains instructions for the **/etc/init**(1M) command. The general format of a line entry in the **/etc/inittab** file is as follows:

identification:level:action:process

The four colon-separated fields are as follows:

identification	A unique one- or two-character identifier for the line entry.
level	The run level in which the entry is to be performed.
action	How **/etc/init** treats the process field [refer to the **inittab**(4) manual page for complete information].
process	The shell command to be executed.

The **/etc/inittab** file contains several entries that spawn **getty** processes. Figure 8-2 is a selection of such entries **grep**'ed from an **/etc/inittab** on a 3B2 computer.

```
co:234:respawn:/etc/getty console console
ct::off:/etc/getty contty contty;# Dedicated lp printer port
21:23:respawn:/etc/getty tty21 9600
22:23:respawn:/etc/getty tty22 9600
23:23:respawn:/etc/getty tty23 9600
24:23:off:/etc/getty tty24 9600;#38400 baud rate is available
25:23:respawn:/etc/getty tty25 9600
26:23:respawn:/etc/getty tty26 9600
27:23:respawn:/etc/getty tty27 9600
28:23:off:/etc/getty tty28 9600 #38400 baud rate is available
31:23:respawn:/etc/getty tty31 9600
32:23:respawn:/etc/getty tty32 9600
33:23:respawn:/etc/getty tty33 9600
34:23:off:/etc/getty tty34 9600 #38400 baud rate is available
35:23:respawn:/etc/getty tty35 9600
36:23:respawn:/etc/getty tty36 9600
37:23:respawn:/etc/getty tty37 9600
38:23:off:/etc/getty tty38 9600 #38400 baud rate is available
41:234:off:/etc/getty tty41 9600 #38400 baud rate is available
42:234:off:/etc/getty tty42 9600 #38400 baud rate is available
43:234:off:/etc/getty tty43 9600 #38400 baud rate is available
44:234:off:/etc/getty tty44 9600 #38400 baud rate is available
45:234:off:/etc/getty tty45 9600 #38400 baud rate is available
46:234:off:/etc/getty tty46 9600 #38400 baud rate is available
47:234:off:/etc/getty tty47 9600 #38400 baud rate is available
48:234:off:/etc/getty tty48 9600 #38400 baud rate is available
```

Figure 8-2: The **getty** Entries From **/etc/inittab**

There are at least three things you might want to do to an **inittab** entry for a TTY line:

1. Change the action. Two actions that apply to TTY lines are "respawn" and "off" [see the **inittab**(4) manual page for complete information on this field].

2. Add or change arguments to **/etc/getty** in the process field. A frequently used argument is **-t***nn*. This tells **getty** to hang up if nothing is received within *nn* seconds. It is good practice to use the **-t** argument on dial-up lines.

3. Add or change comments. Comments can be inserted after a semicolon (;) to end the command and a pound sign (#) to start the comments.

How to Set Terminal Options

The TTY system described thus far establishes a basic style of communication between the user's terminal and the UNIX operating system. Once the user has successfully logged in, there may be terminal options that would be preferable to those in the default set.

The command that controls terminal options is **stty**(1). Many users add an **stty** command to their **.profile** so the options they want are automatically set as part of the **login** process. Here is an example of a simple **stty** command.

```
$ stty cr0 nl0 echoe -tabs erase '^H'
```

The options in the example and their meanings follow:

cr0 nl0	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	Erases characters as you backspace.
-tabs	Expands tabs to spaces when printing.
erase '^H'	Changes the character-delete character to a CTRL-H (^H). The default character-delete character is the pound sign (#). Most terminals transmit a CTRL-H (^H) when the **backspace** key is pressed. Specifying this option makes the **backspace** key useful.

Chapter 9: Basic Networking

Introduction

The Basic Networking Utilities allow computers using the UNIX operating system to 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**).

As an administrator, you need to be familiar with the administrative tools, logs, and data base files used by the Basic Networking Utilities. Procedure 9, "Basic Networking," presents instructions to install and maintain these utilities; this chapter goes into greater detail about the Basic Networking Utilities files, directories, daemons, and commands.

Hardware Used for Networking

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 will need depend on how you want to connect the computers: direct links, telephone lines, or local area networks.

Direct Links

You can create a direct link to another computer by running cables between serial ports on the two computers. Direct links are useful where two computers communicate regularly and are physically close—within 50 feet of each other. You can use a limited distance modem to increase this distance somewhat. Transfer rates of up to 19,200 bits per second (bps) are possible when computers are directly linked.

Telephone Lines

Using an Automatic Call Unit (ACU), your computer can communicate with other computers over standard phone lines. The ACU dials the telephone number requested by the Basic Networking Utilities. The computer it is trying to contact must have a telephone modem capable of answering incoming calls.

Local Area Network

A Local Area Network (LAN) can be the communications medium for basic networking. Once your computer is established as a node on a LAN, it will be able to contact any other computer connected to the LAN.

Commands Used for Networking

Basic networking programs can be divided into two categories: user programs and administrative programs. The following paragraphs describe the programs in each category.

User Programs

The user programs for basic networking are in **/usr/bin**. No special permission is needed to use these programs. These commands are all described in the *User's and System Administrator's Reference Manual.*

cu Connects your computer to a remote computer so you can be logged in on both at the same time, allowing you to transfer files or execute commands on either computer without dropping the initial link.

ct Connects your computer to a remote terminal so the user of the remote terminal can log in. The user of a remote terminal can call the computer and request that the computer call it back. In this case, the computer drops the initial link so that the remote terminal of the modem will be available when it is called back.

uucp Lets a user copy a file from one computer to another. It creates work files and data files, queues the job for transfer, and calls the **uucico** daemon, which in turn attempts to contact the remote computer.

uuto Copies files from one computer to a public spool directory on another computer (**/usr/spool/uucppublic/receive**). Unlike **uucp**, which lets you copy a file to any accessible directory on the remote computer, **uuto** places the file in an appropriate spool directory and tells the remote user to pick it up with **uupick**.

uupick Retrieves the files placed under **/usr/spool/uucppublic/receive** when files are transferred to a computer using **uuto**.

uux Creates the work, data, and execute files needed to execute commands on a remote computer. The work file contains the same information as work files created by **uucp** and

uuto. The execute files contain the command string to be executed on the remote computer and a list of the data files. The data files are those files required for the command execution.

uustat Displays the status of requested transfers (**uucp**, **uuto**, or **uux**). It also provides you with a means of controlling queued transfers.

Administrative Programs

Most of the administrative programs are in **/usr/lib/uucp**, along with basic networking data base files and shell scripts. The only exception is **uulog**, which is in **/usr/bin**. These commands are described in the *User's and System Administrator's Reference Manual.*

You should use the **uucp** login ID when you administer the Basic Networking Utilities 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**.)

uulog Displays the contents of log files of a specified computer. Log files are created for each remote computer with which your computer communicates. The log files contain records of each use of **uucp**, **uuto**, and **uux**.

uucleanup Cleans up the spool directory. It is normally executed from a shell script called **uudemon.cleanup**, which is started by **cron**.

Uutry Tests call processing capabilities and does a moderate amount of debugging. It invokes the **uucico** daemon to establish a communications link between your computer and the remote computer you specify.

uucheck Checks for the presence of basic networking directories, programs, and support files. It can also check certain parts of the **Permissions** file for obvious syntactic errors.

Daemons

There are three daemons in the Basic Networking Utilities. A daemon is a routine that runs as a background process and performs a systemwide 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 data and execute files, 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.

The **uucico** daemon is executed by **uucp**, **uuto**, and **uux** programs, after all the required files have been created, to contact the remote computer. It is also executed by the **uusched** and **Uutry** programs.

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 *X.file* file is found, **uuxqt** opens it to get the list of data files that are required for the execution. It then checks to determine 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. The **uuxqt** daemon is executed by the **uudemon.hour** shell script, which is started by **cron**.

uusched Schedules the queued work in the spool directory. Before starting the **uucico** daemon, **uusched** randomizes the order in which remote computers will be called. Then **uusched** is executed by a shell script called **uudemon.hour**, which is started by **cron**.

Internal Programs

uugetty This program is similar to the **getty** program except it permits a line (port) to be used in both directions. A **uugetty** will be assigned to a port in the **/etc/inittab** file if bi-directional is chosen when you change a port using the **sysadm**(1) **portmgmt** command. The **uugetty** program is executed as a function of the **init** program and is described in the *User's and System Administrator's Reference Manual.*

Support Data Base

The Basic Networking Utilities support files are in the **/usr/lib/uucp** directory. Most changes to these files can be made using the System Administration Menu commands described in Procedure 9, "Basic Networking." The descriptions below, however, provide details on the structure of these files so you can edit them manually.

Devices Contains information about the location and line speed of the automatic call unit, direct links, and network devices.

Dialers Contains character strings required to negotiate with network devices (automatic calling devices) in the establishment of connections to remote computers (non-801 dialers).

Systems Contains information needed by the **uucico** daemon and the **cu** program to establish a link to a remote computer. It contains information 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, login ID, and password.

Dialcodes Contains dial-code abbreviations that may be used in the phone number field of **Systems** file entries.

Permissions Defines the level of access that is granted to computers when they attempt to transfer files or remotely execute commands on your computer.

Poll Defines computers that are to be polled by your system and when they are polled.

Devconfig Configures utilities for the Basic Networking Utilities on a STARLAN NETWORK or some other transport provider that conforms to the AT&T Transport Interface.

Sysfiles Assigns different or multiple files to be used by **uucico** and **cu** as **Systems**, **Devices**, and **Dialers** files.

There are several other files that may be considered part of the supporting data base but are not directly related to the process of establishing a link and transferring files. These files—**Maxuuxqts**, **Maxuuscheds**, and **remote.unknown**—are described briefly in Procedure 9, "Basic Networking."

Devices File

General

The **Devices** file (**/usr/lib/uucp/Devices**) contains information for all the devices that may be used to establish a link to a remote computer, devices such as automatic call units, direct links, and network connections. Although provisions are made for several types of devices, only the AT&T Automatic Dial Modem and Direct Links are supported by AT&T.

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 all of them. 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

Each of these fields is defined in the following section.

Type This field may contain one of two keywords (**Direct** or **ACU**), the name of a Local Area Network switch, or a system name.

Direct This keyword specifies a Direct Link to another computer or a switch (for **cu** connections only).

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 value can be replaced by the name of a LAN switch. The **micom** and **develcon** switches, are the only ones for which there are caller scripts in the **Dialers** file. You can add your own LAN switch entries to the **Dialers** file. If you are adding an AT&T Transport Interface-compatible network, such as STARLAN NETWORK, you would use the special dialer types **TLI** or **TLIS**.

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 tty11 - 1200 ATT2212C

Systems: eagle Any ACU 1200 3251 ogin: nuucp \
```

You can name 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/tty11** line, the name entered in this field would be **tty11**.

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 the line(s) that should be used for internal communications and the lines that 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 \`

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** 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). Usually, the last pair contains only a *dialer* portion and the *token* portion is retrieved from the *Phone* field of the **Systems** file entry.

---

* Registered trademark of AT&T.

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 four different ways, depending on the device associated with the entry:

1. 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 ttyl1 - 1200 ventel

   Dialers: ventel =&-% "" \r\p\r\c $ <K\T%%\r>\c ONLINE!
   ```

   Notice that only the *dialer* portion (**Ventel*** modem) 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, see below.) Backslash sequences are described below.

2. If a direct link is established to a particular computer, the *DTP* field of the associated entry would contain the keyword **direct**. This is true for both types of direct link entries, **Direct** and *System-Name* (refer to discussion on the *Type* field).

* Registered trademark of Ven-Tel, Inc.

3. If a computer with which you wish to communicate 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 tty13 - 1200 develcon \D

Dialers: develcon "" "" \pr\ps\c est:\007 \E\D\e \007
```

   As shown, the *token* portion is left blank, which shows 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 phone 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.

4. 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 tty14 - 1200 develcon vent ventel

Dialers:develcon "" "" \pr\ps\c est:\007 \E\D\e \007
Dialers: ventel =&-% "" \r\p\r\c $ <K\T%%\r>\c ONLINE!
```

   In the first pair, **develcon** is the dialer and **vent** is the token that is passed to the Develcon switch to tell it the device (Ventel 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 Ventel modem has been connected, the second pair is accessed, where **ventel** 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 (**Penril*** modem, **Ventel** modem, etc.). Therefore, the translation will not take place until the caller script is accessed.

\D Specifies that the *Phone* (*token*) field should not be translated using the **Dialcodes** file. If no escape character is specified at the end of a **Devices** entry, the **\D** is assumed (default). A **\D** is also used in the **Dialers** file with entries associated with network switches (develcon and micom).

Protocols

You can define the protocol to use with each device. Usually, it is not needed since you can use the default or define the protocol with the particular System you are calling (see **Systems** file, type field). If you specify the protocol, you must specify it in the form *Type,Protocol* (for example, **STARLAN,e**). Available protocols are:

g This protocol is slower and more reliable than **e**. It is good for transmission over noisy telephone lines.

e This protocol is faster than **g**, but it assumes error-free transmission.

For reliable local area networks, you should use the **e** protocol. Here is an example of adding a protocol name to a device entry:

```
STARLAN,e starlan - - TLIS \D
```

This says, for device **Starlan**, use **e** protocol.

Dialers File

The **Dialers** file (**/usr/lib/uucp/Dialers**) specifies the initial conversation that must take place on a line before it can be made available for transferring data. This conversation is usually a sequence of ASCII strings that is transmitted and expected, and it is often used to dial a phone number using an ASCII dialer (such as the Automatic Dial Modem).

In the above examples, the fifth field in a **Devices** file entry is an index into the **Dialers** file or a special dialer type (801, TLI, or TLIS). Here an attempt is made to match the fifth field in the **Devices** file with the first field of each **Dialers** file entry. In addition, each odd numbered **Devices** field starting with the seventh position is used as an index into the **Dialers** file. If the match succeeds, the **Dialers** entry is interpreted to do the dialer negotiations. 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. Below are some character strings distributed with the Basic Networking Utilities in the **Dialers** file.

```
penril =W-P "" \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!
hayes  =,-, "" \dAT\r\c OK\r \EATDT\T\r\c CONNECT
rixon  =&-% "" \d\r\r\c $ s9\c )-W\r\ds9\c-) s\c : \T\r\c $ 9\c LINE
vadiac =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 \007
micom ""     "" \s\c NAME? \D\r\c GO
direct
att2212c         =+-,     "" \r\c :--: atol2=y,T\T\r\c red
att4000 =,-,     "" \033\r\r\c DEM: \033s0401\c \006 \033s0901\c \
         \006 \033s1001\c \006 \033s1102\c \006 \033dT\T\r\c \006
att2224 =+-,     "" \r\c :--: T\T\r\c red
nls      ""      "" NLPS:000:001:1\N\c
```

There are also three AT&T modems that have entries in the **Dialers** file. The meaning of some of the escape characters (those beginning with "\") used in the **Dialers** file are listed below:

\p	Pause (about ¼ to ½ second)
\d	Delay (about 2 seconds)
\D	Phone number or token without **Dialcodes** translation
\T	Phone 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
\n	Send new-line
\nnn	Send octal number.

Additional escape characters that may be used are listed in the section discussing the **Systems** file.

The "Penril" modem entry in the **Dialers** file is executed as follows. First, the phone 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, go to the next thing.)
\d	Delay for 2 seconds.
>	Wait for a >.
s\p9\c	Send an **s**, pause for ½ second, send a **9**, send no terminating new-line.

)-W\p\r\ds\p9\c-)	Wait for a **)**. If it is not received, process the string between the **-** characters as follows. Send a **W**, pause, send a carriage-return, delay, send an **s**, pause, send a **9**, without a new-line, and then wait for the **)**.
y\c	Send a **y**.
:	Wait for a **:**.
\E\TP	Enable echo checking. (From this point on, whenever a character is transmitted, it will wait for the character to be received before doing anything else.) Then, send the phone number. The **\T** means take the phone number passed as an argument and apply the **Dialcodes** translation and the modem function translation specified by field 2 of this entry. Then send a **P**.
>	Wait for a **>**.
9\c	Send a **9** without a new-line.
OK	Waiting for the string **OK**.

Systems File

The **Systems** file (**/usr/lib/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 can be configured to prevent any computer that does not appear in this file from logging in on your computer (Refer to the section "Other Files Used for Networking" 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 management of this file is supported by the System Administration Menu subcommand, **systemmgmt**.

Using the **Sysfiles**, 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

Each of these fields is defined in the following section.

System-name
This field contains the node name of the remote computer.

Time This field is a string that shows 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 weekday (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 call the remote computer. The call must be started by the remote computer. In other words, your computer is in a passive mode in respect to the remote computer (see discussion of **Permissions** file).

Here is an example:

```
Wk1700-0800,Sa,Su
```

This example allows calls from 5:00 p.m. to 8:00 a.m., Monday through Friday, and calls anytime Saturday and Sunday. The example would be an effective way to call only when phone rates are low, if immediate transfer is not critical.

Time (cont'd)

The *time* portion should be a range of times such as 0800-1230. If no *time* portion is specified, anytime 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.

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 \
Devices: ACU ttyll - D1200 att2212c
```

You can define the protocol used to contact the system by adding it to the *Type* field. The example above shows how to attach the protocol **g** to the device type **ACU**. See the information under the "Protocols" section in the description of the **Devices** file for details.

Class This field is used to show 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 on the **Devices** file, *Class* field.) 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 \
Devices: ACU ttyll - D1200 att2212c
```

If information is not required for this field, use a **-** as a place holder for the field.

Phone This field is used to provide the phone number (token) of the remote computer for automatic dialers (LAN switches). The phone number is made up of an optional alphabetic abbreviation and a numeric part. If an abbreviation is used, it must be one that is listed in the **Dialcodes** file. See the following page for an example.

```
Systems:  eagle Any ACU D1200 NY3251 ogin: nuucp \

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 phone number in the *Phone* field. Instead, this field will contain the token that must be passed on to the switch so it will know with which computer your computer wishes to communicate. (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 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 ended 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 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 in UUCP 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 2 seconds before sending or reading more characters.
\p	Pause for about 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.
\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 (**/usr/lib/uucp/Dialcodes**) contains the dial-code abbreviations that can be used in the *Phone* field of the **Systems** file. Each entry has the following 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=847-
```

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=847-7867 would be sent to the dialer if the token in the dialer-token-pair is \T.

Permissions File

The **Permissions** file (**/usr/lib/uucp/Permissions**) specifies the permissions that remote computers have with respect to login, file access, and command execution. There are options that restrict the ability of the remote computer 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.

How Entries Are Structured

Each entry is a logical line with physical lines ended by a \ to specify continuation. 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 `"#"` and they occupy the entire line up to a new-line character. Blank lines are ignored (even within multiline 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.

LOGNAME entries will contain a LOGNAME option, and MACHINE entries will contain a MACHINE option.

Considerations

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 log in for UUCP communications must appear in 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 the **/usr/spool/uucppublic** directory of your computer.
    3. The commands sent by the remote computer for execution on your computer must be one of the default commands; usually **rmail**.

Options

This section describes each option, specifies how it is used, and lists its default values.

REQUEST When a remote computer calls your computer and requests to receive a file, this request can be granted or denied. The REQUEST option specifies whether the remote computer can request to set up file transfers from your computer. The string

```
REQUEST=yes
```

specifies that the remote computer can request to transfer files from your computer. The string

```
REQUEST=no
```

specifies that the remote computer cannot request to receive files 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. A note on security: 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.

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 SENDFILE 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=/usr/spool/uucppublic
WRITE=/usr/spool/uucppublic
```

The strings

```
READ=/ WRITE=/
```

specify permission to access any file that can be accessed by a local user with "other" permissions.

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 the prefix 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=/usr/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=/usr/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 **/usr/spool/uucppublic** directory. NOWRITE works similar to 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 sure 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 rarely used. Note that if two sites have this option set for each other, a conversation will never get started.

COMMANDS The COMMANDS option can be hazardous to 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 the COMMANDS option is not used 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 COMMAND default so that the computers owl, raven, hawk, and dove can now execute **rmail**, **rnews**, and **lp** on your computer.

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 command **rmail** uses the default path. The default paths for your computer are **/bin**, **/usr/bin**, and **/usr/lbin**. 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.

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, and 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 with the COMMANDS option when specifying commands that are potentially dangerous to the security of your computer. 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 someone 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**. As can be seen, 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. 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 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 files put into one of the **eagle**, **owl**, or **hawk** spool directories is put there by one of those computers. If a remote computer logs in and says that it is one of these three computers, its execution files 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 when the command set changes. 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 (**/usr/lib/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 hour ...

For example the entry:

```
eagle 0 4 8 12 16 20
```

will provide polling of computer **eagle** every four hours.

The **uudemon.poll** script does not actually do the poll. It merely sets up a polling work file (always named C.*file*), in the spool directory that will be seen by the scheduler, which is started by **uudemon.hour**.

Devconfig File

The **/usr/lib/uucp/Devconfig** file is used when your computer communicates over some other STREAMS-based transport provider that conforms to the AT&T Transport Interface (TI) such as the STARLAN network.

Devconfig entries define the STREAMS modules that are used for a particular TI 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 TI 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 should most commonly be used in the file:

```
service=cu      device=STARLAN  push=ntty:tirdwr:ld0
service=uucico  device=STARLAN  push=ntty:tirdwr:ld0
```

This example pushes **ntty**, then **tirdwr**, then **ld0**. The **Devconfig** 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.

Sysfiles File

The **/usr/lib/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 addresses different from **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 **/usr/lib/uucp** directory, unless a full path is given. A backslash-carriage return (\<**CR**>) can be used to continue an entry on the next line.

Here is an example of using a local **Systems** file in addition to the usual **Systems** file:

```
service=uucico:cusystems=Systems:Local_Systems
```

If this is in **/usr/lib/uucp/Sysfiles**, then both **uucico** and **cu** will first look in **/usr/lib/uucp/Systems**. If the system they are trying to call does not have an entry in that file or if the entries in the file fail, then they'll look in **/usr/lib/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.

Other Files Used for Networking

There are three other files that affect the use of basic networking facilities. Usually, the default values are fine and no changes are needed. If you want to change them, however, use any standard UNIX system text editor (**ed** or **vi**).

Maxuuxqts This file defines the maximum number of **uuxqt** programs that can run at once.

Maxuuscheds This file defines the maximum number of **uusched** programs that can run at once.

remote.unknown This file is a shell script that executes when a machine that is not in any of the **Systems** starts a conversation. It will log the conversation attempt and fail to make a connection. If you change the permissions of this file so it cannot execute (**chmod 000 remote.unknown**), your system will accept any conversation requests.

Administrative Files

The basic networking administrative files are described below. These files are created in spool directories to lock devices, hold temporary data, or keep information about remote transfers or executions.

TM (temporary data file)

These data files are created by Basic Networking processes under the spool directory (that is, **/usr/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 3-digit number starting at 0.

When the entire file is received, the **TM.***pid.ddd* file is moved to the path name specified in the **C.***sysnxxxx* file 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)

Lock files are created in the **/usr/spool/locks** directory for each device in use. Lock files prevent duplicate conversations and multiple attempts to use the same calling device. The names of lock files have the format

LCK..*str*

where *str* is either a device or computer name. These files may remain in the spool directory if the communications link is unexpectedly dropped (usually on computer crashes). The lock files will be ignored (removed) after the parent process is no longer active. The lock file contains the process ID of the process that created the lock.

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 4-digit job sequence number assigned by UUCP. Work files contain the following information:

- Full path name of the file to be sent or requested.
- Full path name 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 (**D.0**) is used.
- Mode bits of the source file.
- Remote user's login name to be notified on completion of the transfer.

D. (Data file) Data files are created when specified in the command line to copy the source file to the spool directory. The names of data files have the following format

D.*systmxxxxyyy*

where *systm* is the first five characters in the name of the remote computer, *xxxx* is a 4-digit job sequence number assigned by **uucp**. The 4-digit job sequence number may be followed by a subsequence number, *yyy*, that is used when there are several **D.** files created for a work (**C.**) file.

X. (Execute file)

Execute files are created in the spool directory before 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 4-digit sequence number assigned by UUCP. Execute files contain the following information:

- Requester's login and computer name
- Name of file(s) required for execution
- Input to be used as the standard input to the command string
- Computer and file name to receive standard output from the command execution
- Command string
- Option lines for return status requests.

Direct Links

General

This section discusses how RS232 direct links are created between the following:

- Two 3B2 computers
- A 3B2 computer and a 3B5 computer
- A 3B2 computer and a 3B15 computer (same procedure as 3B5)
- A 3B2 computer and a 3B20 computer

Direct links are beneficial only when:

- It is not possible to link the computers together through a Local Area Network (LAN).
- Two computers transfer large amounts of data on a regular basis.
- Two computers are located no more than several hundred cable feet apart.

The distance between two directly linked computers is dependent on the environment in which the cable is run. The standard for RS-232 connections is 50 feet or less with transmission rates as high as 19,200 bits per second (bps). As the cable length is increased, noise on the lines may become a problem, which means that the transmission rate must be decreased or limited distance modems be placed on each end of the line.

Do not use more than 1000 cable feet to connect the two computers, or communications will be unreliable. This link should operate comfortably at 9600 bps in a clean (noise-free) environment. The following sections apply to direct link connection. For additional details on direct link connection, refer to installation guide.

If the link is established using standard AT&T parts, the computers could not be separated by more than 100 cable feet (two 50-foot cables connected together) because the longest cable available is 50 feet.

If the two computers are separated by more than 100 cable feet, a null-modem cable must be constructed as follows:

Pin 1 to 1
Pin 2 to 3
Pin 3 to 2
Strap pin 4 to 5 in the same plug
Pin 6 to 20
Pin 7 to 7
Pin 8 to 20
Pin 20 to 6
Pin 20 to 8.

How the Direct Link Is Connected

3B2 Computer to 3B2 Computer Direct Link

A direct link between two 3B2 computers is easily established with two 8-wire cables and two RS-232 connectors. Follow the instructions listed below to establish a direct link.

Note: Do not establish a direct link between the two computers before **uugetty**s are started on both computers (from **inittab** file). If the ports are running **getty**s, both computers will continuously try to log in to each other and will crash.

Step 1: Connect one end of the first shielded cable to the selected port on your 3B2 computer. Be sure to attach the ground connector.

Step 2: Connect the other end of the first shielded cable to the ACU/Modem Adapter. (If you want to connect the two computers over a distance greater than 100 feet, use a Terminal/Printer Adapter attached to a null modem cable of the appropriate length, instead of an ACU/Modem adapter.)

Step 3: Connect the Terminal/Printer Adapter to the ACU/Modem Adapter.

Step 4: Connect the second shielded cable to the Terminal/Printer Adapter.

Step 5: Connect the other end of the second shielded cable to the appropriate port on the remote 3B2 computer.

3B2/3B5/3B15 Computer or 3B2/3B20 Computer Direct Link

A direct link can also be made between a 3B2 computer and either a 3B5/3B15 computer or a 3B20 computer with two 8-wire cables and two RS-232 connectors. The following instructions guide you in establishing such a direct link.

Step 1: Connect one end of a shielded cable to the selected port on your 3B2 computer. Be sure to attach the ground connector.

Step 2: Connect the other end of the shielded cable to a Terminal/Printer Adapter.

Step 3: Connect a null modem cable to appropriate port on the remote 3B5/3B15 computer or 3B20 computer.

BNU Software and Direct Links

Ideally, systems that have a direct link should be running the same release of the UNIX system to have the full set of capabilities available. (Bi-directional ports, which are supported by the **uugetty** program, were introduced with UNIX System V Release 2.0 Version 1.) However, lack of a common version of the UNIX operating system will not prevent you from using the Basic Networking Utilities.

This section describes the software files that must be modified on your 3B2 computer to accommodate a direct link connection. You may want to consult the documentation provided with your computer if you are linking directly to a remote computer other than a 3B2 computer.

The following support files must be updated to reflect the presence of a Direct Link:

- **/usr/lib/uucp/Devices**
- **/etc/inittab**
- **/usr/lib/uucp/Systems.**

All necessary additions/modifications to these three files can be done using the System Administration menus subcommand **uucpmgmt**. (See the *Owner/Operator Manual* for information on using the System Administration menus.)

Make Devices File Entries

The **Devices** file contains the information about the location (line) and transmission rate of the link. Entries can be added to the **Devices** file using the System Administration menus subcommand **uucpmgmt devicemgmt** and selecting the **add** operation. You will be prompted for the following information:

- Port name (for **/dev/tty21** use **tty21**)
- Device type to call on (**direct**)
- Speed at which you want to call (**9600** or **19200**).

To access the System Administration menus subcommand **devicemgmt**, enter:

 # sysadm devicemgmt<CR>

You should see the following output on your screen:

```
Running subcommand 'devicemgmt' from menu 'uucpmgmt',
BASIC NETWORKING UTILITIES MANAGEMENT
```

Note: After a brief introduction to the **devicemgmt** *subcommand, the procedure interactively prompts you for the information listed above.*

Make Changes to the /etc/inittab File

There are two versions of the Basic Networking Utilities on 3B computers. The differences between them are reflected in the **/etc/inittab** file. The newest version, which you have on all 3B2 computers, allows for bi-directional login capability by respawning **uugetty** instead of **getty**. This means that if two computers (both using **uugetty**) were connected via a direct link, either of these computers could request communication with the other. This would not be true if only one computer was capable of respawning **uugetty**.

If the direct link is connecting your 3B2 computer with a computer that has the new version of basic networking, the **/etc/inittab** files on both computers should be set up to allow "bi-directional" traffic on the associated lines. This means that the lines used must respawn **uugetty** on each end of the link. This would allow either computer to request communication with (call) the other.

If the direct link is connecting your 3B2 computer with a computer that does not have the new version of basic networking, the **/etc/inittab** file would be set up differently on each system. The **/etc/inittab** file on each computer would be set up to allow either "incoming" or "outgoing" traffic on its line. If one computer allows incoming traffic, the other must allow only outgoing traffic. A **uugetty** could not be used on either computer in this case.

A computer's **/etc/inittab** entry would be respawning **getty** for incoming traffic or have respawn turned off for outgoing traffic. In order for this type of link to work, one of the computers must be set up to poll the other. If the remote computer is allowing only incoming traffic, you must set up your 3B2 computer to poll the remote computer. (You can use the System Administration menus subcommand **uucpmgmt pollmgmt** to do this). If the remote computer is allowing only outgoing traffic on the link, it must be set up to poll your 3B2 computer.

Entries in the **/etc/inittab** file can be changed using the System Administration menus subcommand **uucpmgmt portmgmt** and selecting the **modify** operation. The **modify** operation will prompt you for the following information:

- Port name you want to change (for **/dev/tty21** use **tty21**)
- Direction of traffic on port (**bidirectional, incoming**, or **outgoing**)
- Transmission speed of the link (**9600** or **19200**).

The modify procedure will display the ports that are currently dedicated for use by UUCP (listed in **Devices** file). The port name to be modified must be one that is listed.

To access the **portmgmt** subcommand and change entries in the **/etc/inittab** file, enter:

 # sysadm portmgmt<CR>

You should see the following output on your terminal screen:

```
Running subcommand 'portmgmt' from menu 'uucpmgmt',
BASIC NETWORKING UTILITIES MANAGEMENT
```

Note: After a brief introduction to the portmgmt subcommand, the procedure interactively prompts you for the information listed above.

Make Systems File Entries

An entry must be made into the **Systems** file for the computer associated with the direct link. This can be done using the System Administration menus subcommand **systemmgmt** and selecting the add operation. The add operation will prompt you for the following information:

- Node name of system
- Type of device to call on (**direct**)
- Transmission speed of link (**9600** or **19200**)
- Device port used with link (for /dev/tty21 use **tty21**)
- Login ID used to login on system (**nuucp** or some other login you set up)
- Password used by above login.

To prevent possible problems logging on when operating at high speeds, you can insert pauses (**\p**) between the characters being sent out for the login ID and the password. For instance, instead of **nuucp**, you should enter **n\pu\pu\pc\p\pp** when prompted for the login ID. Do **not** select the default by pressing RETURN. The same applies for the password assigned.

To access the **systemmgmt** subcommand and add entries in the **Systems** file, enter the following.

 # sysadm systemmgmt<CR>

You should see the following output on your terminal screen:

```
Running subcommand 'systemmgmt' from menu 'uucpmgmt',
BASIC NETWORKING UTILITIES MANAGEMENT

        Note: After a brief introduction to the systemmgmt
        subcommand, the procedure interactively prompts you for
        the information listed above.
```

On completion of the **add** operation, a new entry is added to the **Systems** file for the remote computer. You must also make an additional entry in the **Devices** file. When you use **devicemgmt** to create an entry for the link, it creates an entry similar to the one shown below.

```
Direct tty21 - 9600 direct
```

Chapter 10: Remote File Sharing

Overview

Remote File Sharing (RFS) allows computers running UNIX System V to selectively share resources (directories containing files, subdirectories, devices, and/or named pipes) across a network. As an administrator of a computer on an RFS network, you can choose directories on your system you want to share and add them to a list of available resources on the network. From this list, you can choose resources on remote computers that you would like to use on your computer.

Resource Sharing

Sharing a resource on a Remote File Sharing system begins with a path name to a UNIX system directory. If there is a directory you want to share, assign it a resource identifier and "advertise" it to other machines, using the **adv**(1M) command. The resource identifier is how other machines reference that directory. Computers that pass the security checks you have set up can then mount your resource as they would mount a file system locally. The **mount** command with the **-d** option is used for mounting remote resources.

Figure 10-1 shows how two computers can share resources. In this example, the administrator of a computer named **fie** on a Remote File Sharing system wants to share all files and directories under **/fs1** on its file system tree. The administrator advertises **/fs1** as a resource called FSLOGS.

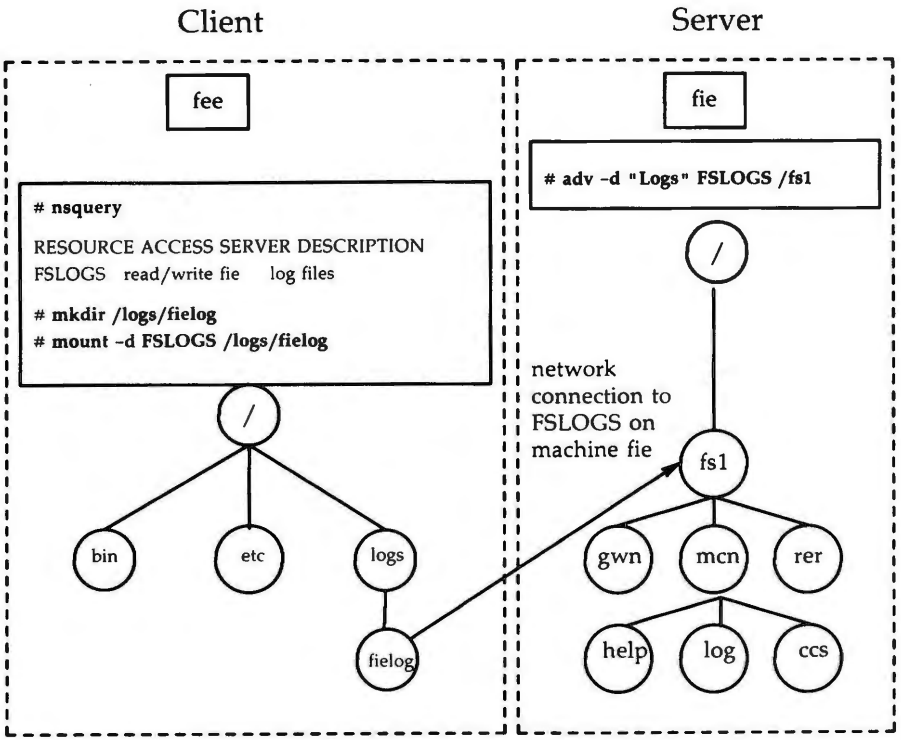


Figure 10-1: Example — Sharing Resources

Another machine in **fie**'s domain is called **fee**. The administrator from **fee** uses the **nsquery** command to see that FSLOGS is available on **fie**. Then **fee**'s administrator creates a directory called **/logs/fielog** on **fee** (**mkdir** command) and **mounts** FSLOGS on **/logs/fielog**.

Files or subdirectories from **/fs1** are now accessible to users on **fee**. Users can **cd** to the remote directory, list the contents, and run a remote program locally. If the resource contained the **/dev** directory, users could direct output to a remote device as though the device were on the local machine.

Domains

Each machine on a Remote File Sharing network must be assigned to a domain. The main reasons for domains are to simplify name service and provide a focal point for security of a group of machines.

Domain names act like telephone area codes. You can address all computers and resources in your domain directly. For outside domains, you simply attach the domain name to the node name or resource identifier. This becomes increasingly valuable as RFS networks expand.

Name Service

Each domain must be assigned a primary and zero or more secondary domain name servers. These machines can share resources, like any other computer in the domain, but they have some special responsibilities.

Primary

The main duty of the primary domain name server is to keep track of all computers and resources within the domain it serves. It ensures that all resource identifiers and machine node names are unique within the domain.

A required task of the primary is to add each computer to the Domain Member List and assign its RFS password.

A list of advertised resources are automatically stored on the primary, so any computer can see a complete list of available resources for the domain. Also, when a computer advertises a resource, it registers its network address with the primary.

When a computer tries to mount another machine's resource, the primary can tell the computer where the resource can be found on the network.

An optional function of the primary is to gather lists of each computer's users (**/etc/passwd**) and groups (**/etc/group**). Each computer in the domain can then use these lists to specifically define the permissions each machine's users will have to its resources.

A primary can also gather names and network addresses of other domains' name servers. Once the primary knows another domain name server's address, machines in its domain have the potential to access resources from any machine in the other domain.

Secondaries

If the primary fails, domain name service functions are automatically assumed by one of the secondary domain name servers. The secondary is intended to take over temporarily, until the primary comes up again.

While the secondary will have information needed to run the domain name server, domain information should not be modified on the secondary. As soon as the primary comes up again, the secondary should be instructed to pass name server responsibility back to the primary (**rfadmin -p** command). Then the primary's administrator can change the domain member list, edit the **rfmaster** file, or gather optional user and group information again on the primary.

Transport Provider

The transport provider provides the pathway used by Remote File Sharing to communicate with other machines. The term transport provider is used to refer to the physical network that connects the machines and the software needed to send messages across the network.

Remote File Sharing can communicate using any transport provider that is compatible with the AT&T Transport Interface Specification. The AT&T STARLAN NETWORK is one transport provider that can be used with RFS.

Although the transport provider is not considered part of the RFS package, RFS will not work if the transport provider is not functioning properly. Also, some information needed to configure RFS varies from one transport provider to another. For example, network addresses of the primary and secondaries and the network specification to identify the transport provider to RFS are dependent on the particular transport provider used.

The following sections describe transport provider information that relates to RFS administration: the Network Listener, Network Specification, and Network Addresses.

Network Listener

The network listener is part of the Networking Support Utilities package. Essentially, the listener's function is to wait for requests from the network. A call from the network will request a particular service code. The service code will tell the listener to direct the call to a particular process.

Service code **105** is used to request RFS services. If all software installation was done as noted in the *Remote File Sharing Release Notes*, the RFS service code should be automatically configured for the listener of every transport provider you installed. Otherwise you will need to use the **nlsadmin**(1M) command to manually configure the listener. (See the "Setting Up RFS" section of this chapter.)

Network Specification

Since you could have several transport providers on one computer, you must tell RFS which transport provider will handle RFS on your machine. The network specification is the name you will use when you initially configure RFS to indicate its transport provider. The STARLAN NETWORK, for example, uses **starlan** as its network specification. This tells RFS that **/dev/starlan** is the device representing the transport provider to use.

Network Addresses

When Remote File Sharing is started on a machine, the machine tries to contact its domain's primary name server. In order to do that, the machine must know the primary's network address.

The form of the network address varies according to the transport provider used. The STARLAN NETWORK convention for network addressing is to use a machine's node name and append the string **.serve** to create the

network address. For example, the network address for a machine whose node name is **charlie** would be **charlie.serve** on a STARLAN NETWORK.

Security

RFS provides several mechanisms for ensuring the security of your resources. Some of these mechanisms, however, require diligence to set up and maintain. This is especially true if the machines, resources, and users are constantly changing on the network.

As a system administrator you can maintain strict control of your resources. No files, directories, or devices in an unshared file system can be accessed by other computers. Standard UNIX system file security measures can be used in combination with special RFS facilities to protect your resources.

Direct access to your computer is controlled because local users still have to log in as they always have. As for remote accessibility, you can set up security to allow only certain remote computers to access your resources.

The major mechanisms in RFS for protecting your resources are described in the sections "Verify Computers," "Restrict Resources," and "Map IDs."

Verify Computers

When a remote computer tries to mount a resource from your computer, and no other resources are mounted, it tries to set up a connection (virtual circuit) across the network to your machine. Once this virtual circuit is set up, the remote machine can mount any resource you have made available to it. This virtual circuit is closed when the last resource is unmounted.

Before this virtual circuit is created, you can verify that the computer is the one it claims to be by checking its RFS password. The following text describes what happens when verification is and is not used.

- No verify: any computer can connect

  If the computer is listed in the *domain*/**passwd** file, your machine will check its password. Otherwise, your computer will accept it as the machine it claims to be.

- Verify: some computers can connect

  If you use the RFS verification feature, you can make sure that only specific machines can use any of your resources. Those machines must be listed in the proper *domain*/**passwd** file and must match the password you have for them. (*domain* is the domain name of the requesting machine.) You can tailor this file if you only want a subset of machines to be allowed to connect. (A description of how to use this feature is contained in the "Setting Up RFS" section of this chapter.)

Restrict Resources

Once a remote computer has established a connection to your computer, the resources it can mount from your machine depend on how you advertised each resource. These are your choices:

- Any machine can mount

  You may have advertised the resource so that any machine that can connect to your machine can mount it.

- Some machines can mount

  You restricted access to the resource to certain machines. The remote computer trying to mount it must be one of those machines.

You may have advertised the resource as read-only. In that case, the remote computer can only mount the resource read-only instead of read/write (default).

Map IDs

Remote users' permissions can be defined to provide another layer of security for a mounted resource. Remote users and groups can be mapped into your computer's user and group list to set permissions they will have to your resources.

You can set these mapping rules on a global or per-machine basis. The global rules set user and group permissions for all remote machines that do not have explicit mapping rules.

Here are the ways you can map remote machines' users into your machine. These rules apply to both global and per-machine mapping.

- No mapping

  If you don't set any special mapping for any remote computer, all users are mapped into your machine as a "special guest" user ID/group ID. This is the easiest approach because you don't need to keep any records for the remote machine, create rules files or run the **idload** command.

- DEfault mapping

  You can set default mapping so that all remote users are mapped into one of these permissions:

  - The local user ID number that matches each remote user's ID (**default transparent**)
  - A single local ID number
  - A single local ID name
  - The local user name that matches each remote user's name (**map all**).

  Group permissions can be mapped in the same way. Users and groups are mapped independently. If there are exceptions to the default mapping, you can **exclude** certain users and groups so they only have special guest permissions (for example, **exclude 0**).

- specific mapping

  You can map any user or group from any remote machine into a specific user or group on your machine. This can be done by user name or numeric ID.

Using these mapping techniques and standard methods for setting file permissions, you can keep strict controls over your resources, even after they are remotely mounted. (See the "Mapping Remote Users" section of this chapter for more details.)

RFS Features

Some Remote File Sharing features that reflect improvements over other distributed file systems are described in the following paragraphs.

Compatibility — Once you mount a remote resource on your system, it will look to your users as though it is part of the local system. You will be able to use most standard UNIX system features on the resource. Standard commands and system calls, as well as features like File and Record Locking, work the same on remote resources as they do locally. Applications should be able to work on remote resources without modification.

Flexibility — Since you can mount a remote resource on any directory on your system, you have a lot of freedom to set up your computer's view of the world. You do not have to open all your files to every machine on the network. Likewise, you do not have to make all files on the network available to your computer's users.

Performance (Client Caching)

The client caching feature of RFS provides substantial performance improvements over noncaching systems by reducing the number of times data must be read across the network. Client refers to the computer that is using a remote resource, while caching refers to the client's ability to store data in local buffer pools.

The first time a client process reads a block of data from a remote resource, it is placed in local buffer pools. Subsequent client processes reading a server file can avoid network access by finding the data already present in local buffers. This generally causes a large reduction in network messages, resulting in improved performance.

In order for client caching to work simply and reliably, the following features were built into it:

- Cache consistency. Checking mechanisms are used to ensure that the cache buffers accurately reflect the contents of the remote file the user is accessing.

- Transparency. The only difference users should see between caching and non-caching systems is improved response time. RFS-based applications do not have to be changed to run on a Remote File Sharing system that caches remote data.
- Administration. By default, client caching is on. However, options are available to turn off caching for an entire system or for a particular resource. (You would probably only do this if you have an application that does its own network buffering.) There are also some tunable parameters available to fine tune your system according to the way you use RFS. (See the "Monitoring" and "Parameter Tuning" sections of this chapter for more information.)

Setting Up RFS

In most cases, you will not need the set of tasks described in this section because the basic RFS configuration and reconfiguration can be handled using the **sysadm setuprfs** command, as described in Procedure 10.1. These tasks are for those who want to go deeper into the workings of RFS or are having problems with particular components.

These tasks are run from the shell. They should be run initially in the order described.

Once these tasks are completed, go to the "Starting/Stopping RFS" section for information on starting RFS.

Prerequisites

Before you begin setting up RFS, the following must be installed and running: UNIX System V Release 3.1 (or later) software, Remote File Sharing Utilities, Networking Support Utilities, and transport provider software. (See the *Remote File Sharing Release Notes* and the transport provider manuals that accompany the product for installation instructions.)

You must also log in as **root**.

Set Node Name

Caution: Changing the node name of your 3B2 computer requires careful coordination with all machines that communicate with yours using Remote File Sharing or other communications packages that rely on node name.

Check to see if your computer's node name is set to the name you want (**uname -n**). If it's not, set it by typing:

sysadm nodename

You will be asked to type in your computer's node name. A node name that is valid for Remote File Sharing can consist of up to 8 letters (uppercase and lowercase), digits, hyphens (-), and underscores (_). Some networks, such as AT&T STARLAN NETWORK, require that every node name in the network be different. Remote File Sharing, however, only requires that every node name in a domain be different.

Set Up Network Listener

If you have installed the Networking Support Utilities, the AT&T STARLAN NETWORK, and Remote File Sharing in the order described in the *Remote File Sharing Release Notes*, you can skip this task. The listener will already be installed and set up to run automatically, and Remote File Sharing will be listed as an available service.

If you are using another transport provider, or suspect that your STARLAN NETWORK listener is improperly set up, this task shows how to manually set up the listener. In the following example the STARLAN NETWORK is used. To set up the listener for other networks compatible with the AT&T Transport Interface, you would replace **starlan** with the name of the network (network specification) you are installing. (For more details, see the **nlsadmin**(1M) manual page.)

To determine if the listener is properly installed and set up for use by RFS, type the following:

nlsadmin -v starlan

If service code 105 is listed, then the listener is configured to be used for Remote File Sharing.

Run the following commands if the listener is not properly set up. If you run any of these commands and they have already been run, you will receive a message telling you so. This won't harm your listener configuration. Type:

nlsadmin -i starlan

to initialize the files needed for the listener process for the network specified, in this case **starlan**.

Next type:

nlsadmin -a 105 -c /usr/net/servers/rfs/rfsetup -y "RFS server" starlan

to add the Remote File Sharing service (**rfsetup**) to the list of services available to the **starlan** listener.

Use the following command line to report the status of the **starlan** listener process installed on this machine (ACTIVE or INACTIVE):

nlsadmin -x

Next type:

nlsadmin -l "*nodename***.serve" -t "***nodename***" starlan**

to register the network addresses of your machine. The listener will listen for requests for these addresses on the network. Only the **-l** address is required by Remote File Sharing. The **-t** address is used only for terminal services and may not be needed on all networks. (Other networks will use other types of network addresses. See the description of **rfmaster**(4) in the *System Administrator's Reference Manual* for the syntax of different address types.)

To start the listener, type:

nlsadmin -s starlan

Normally, it will be started automatically when your machine enters multi-user mode **(init 2)**.

Set the Domain Name

Set the domain name by typing:

dname -D *domain*

where *domain* is replaced by the domain of which your machine will be a member of. The domain name must:

- Contain no more than 14 characters
- Consist of any combination of letters (uppercase or lowercase), digits, hyphens, and underscores
- Be different from the name of any other domain used on the network, if there is more than one domain on your network.

You can check the current domain name by simply typing:

dname

Set the Transport Provider

To identify the network, you must tell Remote File Sharing the network (transport provider) it should use. (In our example, this is **starlan** for the STARLAN NETWORK.)

dname -N starlan

This command indicates the device, relative to the **/dev** directory, that is used for the transport provider.

Create rfmaster File

The **rfmaster** file should only be created manually on the primary. If your machine is not the primary, you should skip this task; the **rfmaster** file for your domain will automatically be placed on your machine the first time you start RFS (**rfstart -p** *primary_addr*).

If you are on the primary, you can create an **rfmaster** file in the **/usr/nserve** directory using any standard file editor. The contents of this file will define:

- The primary name server for your domain
- Secondary name servers for your domain
- Network addresses for each of these machines.

(See the section on "Multiple Domain Name Service" in this chapter for a description of other information you may want to put into the **rfmaster** file.)

Here is an example of an **rfmaster** file for a domain called **peanuts**, whose primary and secondary name servers' node names are **charlie**, **linus**, and **lucy**. Adding each machine's domain name (**peanuts**) to its node name, separated by a period, forms its full Remote File Sharing machine name. Each line of the example translates as follows:

- For domain **peanuts** the primary is **peanuts.charlie**.
- For domain **peanuts** a secondary is **peanuts.linus**.
- For domain **peanuts** another secondary is **peanuts.lucy**.

- For computer **peanuts.charlie** the network address is **charlie.serve**.
- For computer **peanuts.linus** the network address is **linus.serve**.
- For computer **peanuts.lucy** the network address is **lucy.serve**.

(The addresses shown are an example of AT&T STARLAN NETWORK addresses. STARLAN addresses should be in the form *nodename*.**serve**.)

```
peanuts          p          peanuts.charlie
peanuts          s          peanuts.linus
peanuts          s          peanuts.lucy
peanuts.charlie  a          charlie.serve
peanuts.linus    a          linus.serve
peanuts.lucy     a          lucy.serve
```

Each line in the example is an entry. The second field is the *Type* field, which indicates whether the entry defines a primary name server (**p**), secondary name server (**s**), or the network address (**a**) for one of these name servers. Here is the information needed for the first field, *Name*, and the third field, *Rdata*, for each type of entry.

p Primary entry. *Name* is the domain name. *Rdata* is the full RFS machine name of the domain's primary name server (*domain.nodename*).

s Secondary entry. *Name* is the domain name. *Rdata* is the full RFS machine name of the domain's secondary name server (*domain.nodename*).

a Address entry. *Name* is the full RFS machine name (*domain.nodename*) of a name server computer. *Rdata* is the network address of the computer. The manuals that come with your network should describe how to find a computer's network address.

Here are some special considerations when creating the file.

- Fields in each entry must be separated by one blank or one tab.
- The address must be in ASCII text or hexadecimal notation. For hexadecimal, the field must begin with \x and contain an even number of digits. If the address contains tabs or spaces, the field must be surrounded by double quotes (").
- An entry can extend beyond one line if you enter a back slash, then a carriage return to continue to the second line.
- This file should be write protected from all but **root**, but all read permissions should be enabled (644 permissions).
- If you start a line with the # character in column 1, the entire line will be treated as a comment.

Add/Delete Domain Members

If your computer is the current primary name server for the domain, you must add each computer to the domain member list. (If a secondary has temporarily taken over, the secondary must pass name server responsibility back to the primary using the **rfadmin -p** command.) To add members, use the following command:

```
# rfadmin -a domain.nodename
Enter password for nodename:
Re-enter password for nodename:
```

where *nodename* is replaced by the node name of the computer you want to add to your *domain*. (The two names must be connected by a period.)

You will be prompted for an initial password, which will be stored in the **/usr/nserve/auth.info/***domain***/passwd** file for the *domain*. When the computer you added starts Remote File Sharing, the computer's administrator must enter this password. You can simply type a **<CR>** for a null password. Otherwise, the password must conform to the same criteria used with the **passwd**(1) command. Repeat this command for each computer you want to add to the domain.

Note: Adding a primary and secondary to the **rfmaster** file does not automatically add them to the domain. You must do this procedure for each of those machines.

You can also use the **rfadmin** command to delete members from the domain member list, as follows:

rfadmin -r *domain.nodename*

Remote Computer Verification (Optional)

Note: This procedure assumes that you are starting RFS from the shell using **rfstart** with the **v** option or **init 3** either manually or automatically at boot time to start RFS (**sysadm setauto**).

When you start Remote File Sharing, you can indicate that all remote machine passwords be verified when they try to use your computer's resources. The command **rfstart** is run automatically when you go into Remote File Sharing state (**init 3**).

If you use **rfstart** with the **-v** option, any machine that tries to mount your resources must match a name and password you have in the **passwd** file in the **/usr/nserve/auth.info/***domain* directory on your machine, where *domain* is replaced by the name of the remote computer's domain. If the remote computer is not listed in this *domain*/**passwd** file, if it is listed and the password doesn't match, or if no *domain*/**passwd** file exists, the remote mount will fail. (This file is automatically on the primary, but it must be added to other machines, as described in this procedure, to use verification.)

If you don't use the **-v** option, the following validation occurs. If a *domain*/**passwd** exists for the remote computer's domain on your computer and the remote computer is listed, but the password doesn't match, a mount request will fail. If the computer is not listed in the file or if the *domain*/**passwd** file doesn't exist, the computer will be allowed to mount your resources without validation. (Of course, a remote mount could still fail if the resource was advertised to a limited subset of machines or was advertised read-only and the machine tried to mount it read/write.)

The following steps describe how verification is set up:

Step 1: Obtain *domain*/**passwd** file(s). The **/usr/nserve/auth.info**/*domain* directory on the primary will contain a file called **passwd**. (*domain* is replaced by the domain name.) This file will have the name and encrypted password for each machine in the domain.

You must make the *domain*/**passwd** file, plus the *domain*/**passwd** file for any outside domains containing machines you want to verify, accessible to your machine in one of the following ways:

Step 1A: Place a copy of this file(s) in the same directory on your machine. The **passwd** file for each domain must be in the appropriate *domain* subdirectory.

or

Step 1B: Have the primary for each domain advertise the **/usr/nserve/auth.info**/*domain* directory; then have it automatically mounted in the same location on your computer. This way you can automatically pick up any changes in machines or passwords. (See the description of **/etc/fstab** in the "Automatic Remote Mounts" section of this chapter for information on setting up automatic mounts.)

Step 2: **rfstart -v**. You must edit the **/etc/rc3.d/S21rfs** file to automatically run **rfstart** with the **-v** option. You will add the **-v** to about line 61 of this file, after the **rfstart** command, as shown in the following example.

```
'rfstart')
   trap 'rm -f /usr/tmp/rfs$$;exit' 0 1 2 3 15
   stat=1
   retries=0
   while [ ${stat} -eq 1 ]
   do
     /usr/bin/rfstart -v </dev/console >/dev/console 2>/usr/tmp/rfs$$
     stat=$?
     case ${stat} in
```

Step 3: If you want to verify only a limited subset of these computers, you must use manually edited versions of the *domain*/**passwd** files, removing any computers you want to prevent from using your resources. (You cannot edit this file if you are a primary or secondary name server or if you have mounted the file from the primary.)

Resource Sharing with Other Domains (Optional)

For computers in your domain to share resources with computers in other domains on your network, you must do the following:

Step 1: Find out:

- The primary name server for each domain
- The secondary name server(s) for each domain
- The network address for each of the above name servers.

Step 2: You must see that the information in Step 1 is added to your domain's **/usr/nserve/rfmaster** file on the primary. See the description of the **rfmaster**(4) file in the *System Administrator's Reference Manual* for the format of the **rfmaster** file.

The following example shows the information added to contact a domain called **docs**.

```
docs          p          docs.big
docs          s          docs.little
docs.big      a          big.serve
docs.little   a          little.serve
```

Step 3: Stop Remote File Sharing on the primary (**rfstop**, **init 2**, or **sysadm stoprfs**).

Step 4: Restart Remote File Sharing on the primary (**rfstart**, **init 3**, or **sysadm startrfs**). Make sure start up has completed before going to the next step.

Step 5: If a secondary machine took over name service when the primary was stopped, pass name service responsibilities back to the primary by typing the following from the secondary:

rfadmin -p

Step 6: Mount resources from an outside domain. Once the name server machines have picked up the new domain names, you can mount a resource from a remote domain on your own machine. You would use the same method of mounting a resource from an outside domain as you would to mount a resource from your domain, with one exception. When you specify the resource to be mounted, you must prepend the domain name to the resource identifier. For example, the command

mount -d docs.INFO /usr/info

could be used to mount a resource called INFO that is advertised in domain **docs** with read/write permissions.

Multiple Domain Name Service (Optional)

Once you have defined a set of primary and secondary name servers to serve a domain, that set of machines may also be name servers for another domain on the same network. The following procedure describes how this can be configured:

Step 1: Edit **rfmaster** file. You must add the information on the new domain's name servers to the **rfmaster** file on the primary. The following is an example of two sets of name servers that serve domains called **docs** and **peanuts**.

```
docs          p          docs.big
docs          s          docs.little
docs.big      a          big.serve
docs.little   a          little.serve

peanuts       p          docs.big
peanuts       s          docs.little
```

Step 2: Stop and restart RFS. You must stop all machines served by the primary (**rfstop**, **init 2**, or **sysadm stoprfs**). You must then restart the primary (**rfstart**, **init 3**, or **sysadm startrfs**). Then start each machine on the system, starting machines that previously had other machines as domain name servers with the **rfstart -p** *address*, where *address* is replaced by the network address of the new primary domain name server. This will ensure that the new information is picked up by each machine.

Complex User ID/Group ID Mapping (Optional)

ID mapping lets you control the access remote users will have to files and directories that make up your shared resources. This feature lets you assign each remote user the permissions of one of your local users (listed in **/etc/passwd**) or the permissions of a special "guest ID," with respect to your shared resources. The "guest ID" will never overlap with any of your local users. The same mechanism can be used to define group permissions (listed in **/etc/group**).

Use this procedure as a tutorial for ID mapping and as a procedure for setting up mapping. If you have questions about particular mapping components, refer to the "Mapping Remote Users" section of this chapter.

When Not to Map

In most cases, ID mapping is not necessary. If you never set up mapping using this or the **sysadm** procedures, all users will be mapped into a single special guest ID. This special guest ID is represented by an ID number that is one higher than the maximum allowed for your system. By default, the maximum number of users and groups on a system is 60000, so the special guest is ID number 60001.

No mapping, or the default mapping, provides the maximum security for your shared resources. When a remote user lists the permissions of your files (**ls -l**), all files will be owned by 60001 or 60002. The 60001 means the file was created by a remote user and, therefore, is owned by every remote user that can access your resource. The 60002 means the file was created by one of your local users and, therefore, remote users can only access the file if the "other" permissions are set (the last three bits of the **rwx** permissions).

When to Map

Using mapping increases the power and flexibility of RFS. The following are some reasons you may want to use mapping:

Special permissions.

> You may want to map some or all remote users into particular local users' permissions. For example, if you are the administrator of several machines, you may want to map all

root logins together across the machines. This would enable you to modify any remote resources mounted on any machine you are working from.

Transparent mapping.

If you set up a group of computers to have the exact same **/etc/passwd** and **/etc/group** files, mapping transparently can be a very powerful technique. When a user creates a file, the user will maintain sole ownership to the file, whether or not the file resides on a remote resource.

With transparent mapping, you could share many resources that require a consistent view of user ownership. For example, you could share your **/usr/mail** directory, mount it on **/usr/mail** on other computers, and have one mail directory for the entire set of machines. The basic concept is that you can avoid duplication of many files and directories while maintaining consistent user permissions.

Mapping by machine.

You may want to map users from one machine differently than users from another machine. For example, you may want to map all users from one machine into user ID **600**, from another machine into **700**, and from a third into **800**. In that way you could monitor which remote machine's users were creating files within your resources.

Note: The default mapping and transparent mapping can be set up using the **sysadm** interface described in Procedure 10. For other mapping techniques, use the following procedure.

Mapping Tools and Files

The result of this procedure is "mapping translation tables." These tables will be used by your system to process requests from remote users for access to your resources that are mounted on their computers.

The command used to create the translation tables is **idload**. When **idload** is run with no options, it does the following:

- Reads the rules files to determine how you want to set up the mapping
- Reads the **passwd** and **group** files on your computer, and copies of those files from other computers, if needed
- Creates translation tables.

There are two options to **idload** you also may want to use when setting up translation tables.

idload -n Before you run **idload** with no options, the **-n** option lets you do a trial run without actually changing the mapping tables. The result is a listing at your terminal of the tables you would create if you ran **idload** with no options.

idload -k After you run **idload** with no options, the **-k** option lets you read the mapping that is currently in effect on your computer.

Figure 10-2 illustrates the components described in the previous paragraphs.

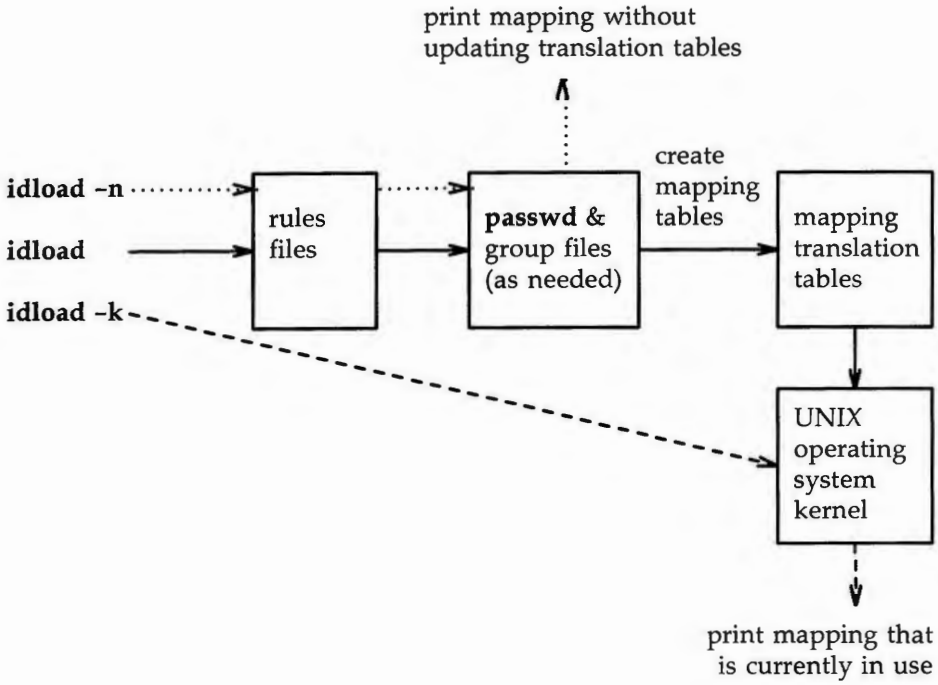


Figure 10-2: ID Mapping Components

The files that are involved in setting up ID mapping are illustrated in Figure 10-3.

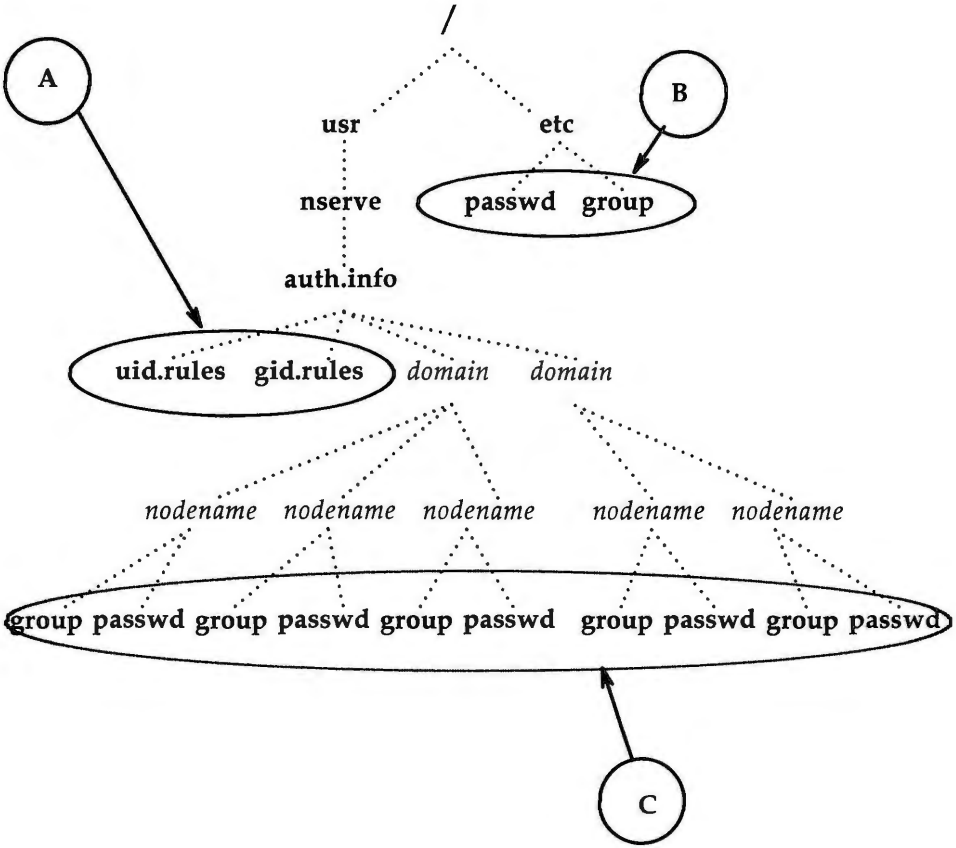


Figure 10-3: ID Mapping Files

The files used for ID mapping are divided into the following three groups, as shown in Figure 10-3.

A. Rules Files

The **uid.rules** and **gid.rules** files are located in the **/usr/nserve/auth.info** directory. The information you add to these files tells the **idload** command how to create the mapping tables.

B. Local **passwd** and **group** Files

The **/etc/passwd** and **/etc/group** files contain lists of the local users on your system. Though you don't modify these files to do ID mapping, you will be interested in the information that is in these files. The first field in each line of your **passwd** and **group** files contains local user and group names, respectively. The third field contains the related ID number. If you map by local name in the rules files, these files are read to translate the names into numbers.

C. Remote **passwd** and **group** Files

Because mapping translation tables are sets of numbers, if you want to map a remote user by name, you must have a copy of the **passwd** and/or **group** files for the remote user's machine. These files should be placed in the **/usr/nserve/auth.info/***domain/nodename* directories, where *domain* and *nodename* are replaced by the remote computer's domain and node names, respectively.

Step 1: Create uid.rules File

The following steps describe how to create the rules used to map remote users.

Using any standard file editor (**ed** or **vi**, for example), create or edit the **uid.rules** file in the **/usr/nserve/auth.info** directory. Steps 1A-1D will help you set up a **global** block of mapping information; steps 1E-1H are for **host** blocks of mapping information. The **global** block defines the permissions that will apply to the users on all computers that do not have specific mapping. Note that all lines within a **global** block are optional.

Step 1A: Add the **global** line. (Only add this line if you want to define a block of global information.) The global block of information must begin with the following keyword on a line by itself:

global

Step 1B: Add a **default** line. (Only add this line if you want to define default information for a global block.) Following the **global** line, you can choose the default permissions that will apply to users from all machines that are not specifically mapped. If this line is not used, the system assumes **default 60001**. (In most

cases, **default 60001** is fine.) The two types of default lines are illustrated below.

The line **default transparent** means that each user will have the permissions of the user with the same ID number on your system. (This strategy is most valuable when the **/etc/passwd** files are identical on the two machines.) In the line **default** *local*, the word *local* can be replaced by a local ID number or ID name. This means that any users that are not specifically mapped will have the permissions of a particular user on your system. (Use only one **default** line in a **global** block.)

default transparent
or
default *local*

Step 1C: Add **exclude** line(s). (Only add this line(s) if you want to exclude certain users.) The **exclude** lines let you exclude certain users from having the permissions defined in the default line. For example, if you used **default transparent**, you may want to use **exclude 0** to make sure that the **root** user doesn't have permission to modify the restricted files owned by **root** in your resources. The two types of exclude lines are illustrated below.

In **exclude** *remoteid, remoteid* is replaced by a remote user ID number. The remote user would then have the permissions of the guest user (UID 60001) to your resources. The **exclude** *remoteid-remoteid* line lets you specify a range of remote IDs to exclude. For example, **exclude 0-100** could be used to exclude all administrative logins from your default mapping.

exclude *remoteid*
or
exclude *remoteid-remoteid*

Step 1D: Add **map** line(s). (Only add this line if you want to map specific users from global machines.) **map** lines let you take specific remote user IDs and map them into the permissions of one of your local users. The two types of map lines are illustrated below.

In **map** *remoteid:local, remoteid* is replaced by a remote user ID number and *local* is replaced by a local user's name or ID number. For example, the line **map 20:root** would map the remote user with

ID number 20 into your machine's **root** permissions (UID 0). The line **map** *remoteid* says give the remote user the permissions of the user with the same ID number on the local system. For example, **map 0** would give **root** from a remote machine the same permissions as **root** on your machine.

map *remoteid:local*
or
map *remoteid*

Once **global** mapping is done, you may want to add **host** mapping information to the **uid.rules** file. A **host** block defines the permissions that will apply to the users on particular remote machines. You can have one **host** block for each remote machine you want to map specifically. Note that all lines within a **host** block are optional.

Step 1E: Add a **host** line. (Only add this line if you want to define a block of host information.) The host block of information must begin with the following keyword on a line by itself:

host *domain.nodename*

where *domain* is replaced by the remote machine's domain name and *nodename* is replaced by the machine's node name.

Step 1F: Add a **default** line. (Only add this line if you want to define default information for a host block.) Following the **host** line, you can choose the default permissions that will apply to all users on the remote machine that are not specifically mapped or excluded. If this line is not used, the system assumes **default 60001**. (In most cases, **default 60001** is fine.) The two types of default lines are illustrated below.

The line **default transparent** means that each user will have the permissions of the user with the same ID number on your system. (This strategy is most valuable when the **/etc/passwd** files are identical on the two machines.) In the line **default** *local*, the word *local* can be replaced by a local ID number or ID name. This means that any users that are not specifically mapped will have the permissions of a particular user on your system.

default transparent
or
default *local*

Step 1G: Add **exclude** line(s). (Only add this line(s) if you want to exclude certain users from default permissions.) The **exclude** lines let you exclude certain users from having the permissions defined in the default line. For example, if you used **default transparent**, you may want to use **exclude 0** to make sure that the **root** user doesn't have permission to modify the restricted files owned by **root** in your resources. The two types of default lines are illustrated below.

In **exclude** *remote, remote* is replaced by a remote user name or UID number. The *remote* user would then have the permissions of the guest user (UID 60001) to your resources. The **exclude** *remoteid-remoteid* line lets you specify a range of remote IDs to exclude. For example, **exclude 0-100** could be used to exclude all administrative logins from your default mapping.

exclude *remote*
or
exclude *remoteid-remoteid*

Step 1H: Add **map** line(s). (Only add this line(s) if you want to map particular users.) The **map** lines let you map specific remote users from specific remote machines into the permissions of one of your local users. The two types of map lines are illustrated below.

The **map all** line says to map all user names into the permissions of the users with the same names on your system. In **map** *remote:local, remote* is replaced by a remote user ID name or number and *local* is replaced by a local user's name or ID number.

For example, the line **map 20:root** would map the remote user with ID number 20 into your machine's **root** permissions (UID **0**). The line **map** *remoteid* says give the remote user the permissions of the user with the same ID number on the local system. For example, **map 0** would give **root** from a remote machine the same permissions as **root** on your machine.

map all
or
map *remote:local*
or
map *remote*

Repeat steps 1E-1H for each specific computer whose users you want to map.

THE **uid.rules** FILE IS NOW COMPLETE!

Figure 10-4 is an example of what your rules file may look like.

```
global
default 1000
exclude 0

host peanuts.snoopy
default transparent
exclude 0

host peanuts.linus
default 60001
map 0:100
```

Figure 10-4: Example **uid.rules** File

Step 2: Create gid.rules File

The following steps describe how to create the rules used to map remote groups.

Create **gid.rules** file. Using any standard file editor (**ed** or **vi** for example), edit the **gid.rules** file in the **/usr/nserve/auth.info** directory. The **gid.rules** file follows the same format as the **uid.rules** file. Therefore, you can use Steps 1A through 1H to set up the **gid.rules** file, replacing any references to users with references to groups.

Note: If you create a **uid.rules** file you should also create a **gid.rules** file. Though **idload** will still work without the **gid.rules** file (**idload** will use defaults for mapping groups) a warning message will be produced.

Step 3: Add passwd and group Files

If, when you edited the **uid.rules** and **gid.rules** files, you referenced any remote users by name, you must have copies of the **passwd** file from the remote users' computers in the **/usr/nserve/auth.info/***domain/nodename* directories on your machine. The same is true of the **group** file for groups referenced by name. (Note that **map all** maps by name.)

The best way to obtain these files is as follows:

Step 3A: Obtain files. Have each machine whose users you want to map by name send you its **/etc/passwd** and **/etc/group** files using any standard file transfer method (such as **uucp**). (The information in the password field can be removed from each entry, if you prefer. The password is made up of the characters between the first and second colon in each entry.)

Step 3B: Create directories. You must create a separate directory on your machine for each computer whose users and groups you map by name. Each directory must be created using the path **/usr/nserve/auth.info/***domain/nodename*, where *domain* is replaced by the remote machine's domain name and *nodename* is replaced by the remote machine's node name. For example, you create the following directory for a machine called **linus** in domain **peanuts**:

/usr/nserve/auth.info/peanuts/linus

Step 3C: Place files. Place the remote machines' **passwd** and **group** files in the directory you created in the previous step.

Step 4: Run idload

Step 4A: Run **idload -n**. This command will print a listing of the mapping rules you set up, without creating translation tables. Figure 10-5 is the output from **idload -n** using the **uid.rules** file shown after Step 1H and a **gid.rules** file with simply **default 60001** in the global block.

```
TYPE  MACHINE          REM_ID    REM_NAME   LOC_ID       LOC_NAME

USR   GLOBAL           DEFAULT   n/a        1000         n/a
USR   GLOBAL           0         n/a        60001        guest_id
USR   peanuts.snoopy   DEFAULT   n/a        transparent  n/a
USR   peanuts.snoopy   0         n/a        60001        guest_id
USR   peanuts.linus    DEFAULT   n/a        60001        n/a
USR   peanuts.linus    0         n/a        100          n/a

GRP   GLOBAL           DEFAULT   n/a        60001        n/a
```

Figure 10-5: Example Output from **idload -n**

Step 4B: Run **idload**. If the output from **idload -n** was acceptable, type the **idload** command with no options to create the translation tables. The **global** rules and **host** rules for any computer that currently has your resources mounted will immediately take effect. Rules for any other computer that you mapped will take effect as soon as that computer mounts one of your resources.

Step 4C: Run **idload -k**. This will print the mapping that is currently in use on your computer. (Remember that rules for any other computer that you mapped will not be in effect until that computer mounts one of your resources.)

ID MAPPING IS NOW COMPLETE!

Once mapping is set up, it can be changed whenever you like. You can edit rules files and run **idload** again at any time. It doesn't matter if resources are mounted or even if RFS is running.

Starting/Stopping RFS

Before a nonprimary machine can start Remote File Sharing, RFS must be configured on the machine and the primary must be up and running RFS.

Is RFS Running?

If you are not sure if RFS is running, type either **sysadm isrfson** or **rfadmin -q**. These will tell you whether RFS is running.

Another way is to check that processes related to RFS are active. To do this, type **ps -e**. These processes should be active:

listen
rfdaemon
nserve
rfudaemon
recovery
server (*optional*)

There may be multiple processes of some of these names running.

Initial RFS Start

The first time you start RFS on a nonprimary machine, if you are not using the **sysadm** interface, you should use the following command. (A primary can start RFS initially by using **init 3**.)

```
# rfstart -p primary_ns_address
rfstart: Please enter machine password:
```

The *primary_ns_address* is replaced by the network address of the primary name server for your domain. (The network addresses for the STARLAN NETWORK are in the form *nodename*.**serve**, where *nodename* is replaced by the machine's node name.)

RFS Password

You will be prompted for a password the first time you start RFS. The password must match the password entered when your machine was added to the domain member list in the primary name server (the **rfadmin -a** command). If password verification succeeds, your computer will save this password automatically so you do not have to enter it again.

Likewise, your machine will save the network address of the primary name server. Therefore, the next time you start up Remote File Sharing, you will be able to do it via **init 3**.

RFS Password Mismatches

Anytime you start RFS (**rfstart**) and your password doesn't match the one on the current domain name server, you will receive a warning, but **rfstart** will NOT fail.

Though Remote File Sharing will be active, you may have a problem if the *domain*/**passwd** file from the primary domain name server is shared with other machines to use for verification. In that case, your remote **mount** requests will fail if the passwords don't match. For this reason, it is recommended that RFS passwords always be kept up to date on each computer and the primary name server. If passwords aren't important to you, you can simply enter a carriage return for the passwords on each computer and the primary.

If you do get warnings that your password is out of sync with the current domain name server and you want to fix it, you should handle it differently if the primary is the current domain name server than if the secondary has temporarily taken over.

First find out which machine is the current name server, and whether it is the primary or the secondary, by doing the following:

```
# rfadmin
the acting name server for domain domain is domain.nodename
# cat /usr/nserve/rfmaster
domain   P   domain.nodename
domain   S   domain.nodename
domain.nodename  A  network_address
domain.nodename  A  network_address
```

Then, depending on which machine is the current name server, do one of the following.

- Secondary is the current name server

  If the primary went down and a secondary took over as domain name server, the secondary may not have a *domain*/**passwd** file or may have one that is out of date. In this case, do not try to correct your password until the primary takes over as domain name server again.

- Primary is the current name server

  Try to correct your password by re-entering it with the **rfpasswd** command. If that doesn't work, follow the sequence shown below, replacing *domain.nodename* with your computer's RFS machine name.

From the primary name server:

```
# rfadmin -r domain.nodename
# rfadmin -a domain.nodename
Enter password for nodename: type password
```

From your computer:

```
# sysadm stoprfs
# rm /usr/nserve/loc.passwd
# sysadm startrfs
rfstart: Please enter machine password: type password
```

You should then make sure that any computer that verifies your computer's password copies the new *domain*/**passwd** file from the primary.

Changing RFS Password

If you want to change your RFS password later, you must use the **rfpasswd** command. This will change your RFS password, both on your computer and on the primary domain name server. Processing of the new password follows the same criteria as **passwd**(1) in the *User's Reference Manual*.

Since changing passwords requires communication with the primary domain name server, Remote File Sharing must be running on both your computer and the primary domain name server. You cannot change your RFS password if the primary is down and a secondary is the current domain name server.

Caution: When you change your password, computers that are authenticating your computer may not automatically receive the change. If you are unable to mount a resource from a remote machine after you change your password, check that the remote machine has copied the latest version of your domain's passwd file from your primary domain name server.

Automatic RFS Startup (init 3)

There are several steps involved in starting up Remote File Sharing and sharing resources. To simplify this procedure, a new Remote File Sharing run level has been defined: run level 3.

When you enter run level 3 using the **init 3** command, Remote File Sharing is automatically started via **/etc/rc3** from shell scripts in your computer's **/etc/rc3.d** directory. These scripts start Remote File Sharing, advertise local resources, and mount remote resources. When you leave run level 3 (using **shutdown** or **init 2**, for example), Remote File Sharing processes will be stopped.

You can add your own shell scripts to those that start run level 3. You can also tailor the run level 3 shell scripts to suit the way you use Remote File Sharing.

See the "Operating Levels" section of Chapter 3, Processor Operations, for details on the run levels. This section will describe those shell scripts used in run level 3 and suggest how to modify or add to them.

Note: Before you can enter Remote File Sharing mode, you must have already installed and configured Remote File Sharing. See Procedure 10.1 for information on setting up Remote File Sharing.

Entering Run Level 3

You can go into **init** level 3 in one of three ways:

1. From single-user mode (run level **s**)

   Remote File Sharing mode is also a multiuser mode. Therefore, when you type **init 3** from single user mode, all multiuser processes (**gettys**, **cron**, etc.) will be started, followed by Remote File Sharing mode processes.

2. From multiuser mode (run level 2)

   When **init 3** is run from run level 2, init checks that all multiuser processes are running, then starts the Remote File Sharing mode processes. (**init 3** will not spawn another process for a level 2 script that is already running.)

3. At boot time

   By default, your system will enter run level 2 at boot time. You can change that to have run level 3 start automatically at boot time by changing the value for **initdefault** in the **/etc/inittab** file so it reads as follows:

```
is:3:initdefault:
```

init 3 Processing

When **init 3** is run, all entries in the **inittab** file that indicate level 3 are started, including **/etc/rc3**. **/etc/rc3** executes all shell scripts in **/etc/rc3.d** that begin with **S**.

RFS places only one file in **/etc/rc3.d**: **S21rfs**. This file is linked to the **rfs** file in **/etc/init.d**. Also, the **rfs** file is linked to **K50rfs** in **/etc/rc2.d** and **K65rfs** in **/etc/rc0.d**.

/etc/rc3 executes **S21rfs** with the **start** option upon entering run level 3. **S21rfs** then does the following:

- Validates that the domain name has been defined for your machine.
- Validates that the **rfmaster** file has been created. (This may have been created automatically the first time you ran **rfstart -p** if your machine is not the primary. The latest copy is then sent to your machine from the primary domain name server.)
- Executes the **rfstart** command continuously, with 60-second sleep intervals, until it succeeds or returns a fatal error.
- Executes **/etc/init.d/adv** to advertise all system resources you set up in your **/etc/rstab** file. (The **/etc/rstab** file contains an entire **adv** command line for each advertised resource.)
- Executes **/etc/rmountall** to mount all remote resources you listed in your **/etc/fstab** file. Any remote mount that does not succeed will be tried continuously until it does via **/etc/rmount**(1M). (See "Automatic Remote Mounts" in this chapter for the format of **/etc/fstab**.)

When you leave run level 3 via **init 1** or **2**, **/etc/init.d/rfs** is executed with the **stop** option. This will execute **rfstop**.

Note: If for some reason RFS fails to terminate the **rfudaemon**, Remote File Sharing may continue to run in the lower run state. You can always bring down Remote File Sharing by running **unadv** and **fumount** for each advertised resource, **umount** for each mounted remote resource, and **rfstop**.

Changing init 3 Processing

Going into **init** state **3** makes some assumptions about how you use your Remote File Sharing system. Here is a description of how to change some of the processing that takes place.

- Retry **rfstart**

  By default, **init 3** will keep trying to start Remote File Sharing (**rfstart**) until it succeeds. If you want it to try a limited number of times, you must edit the **/etc/rc3.d/S21rfs** file. Find the line retries=0 and change the number **0** (try forever) to the number of times you want it to retry.

- Retry mounts

  When you enter **init 3**, the system tries separately, every 60 seconds, to mount each resource listed in **/etc/fstab** until it succeeds or you leave state 3. To change this behavior you can edit **/etc/rmount**. Find the line RETRIES=0 and change **0** (try forever) to the number of times you want to attempt to mount each resource. Find the line TIME=60 and change 60 to the number of seconds you want it to wait between retries.

Adding RFS Mode Scripts

All files in **/etc/rc3.d** and other **/etc/rc?.d** directories are shell scripts, so you can read them to see what they do. You can modify the existing files, though it is preferable to add your own since the delivered scripts may change in future releases. To create your own scripts you should follow these rules:

- Place the file in **/etc/init.d**.

- Link the file to files in appropriate run level directories using the naming convention described below.
- Have the file accept the **start** and/or **stop** options.

You should name the files using the following conventions:

S*00name*
or
K*00name*

The file names can be split into three parts:

S or **K**	The first letter of each file defines whether the process should be started (**S**) or killed (**K**) upon entering the new run level.
00	The next two characters represent a number from 00 to 99. These numbers indicate the order in which the files will be started (S00, S01, S02, etc.) or stopped (K00, K01, K02, etc).
name	The rest of the file name is the **/etc/init.d** file name this file is linked to.

For example, the **init.d** file **rfs** is linked to the **/etc/rc3.d** file **S21rfs** and **rc2.d** file **K50rfs**. When you enter **init 2**, this file is executed with the **start** option: **sh S68netdaemon start**. When you enter **init 0**, this file is executed with the **stop** option: **sh K67netdaemon stop**.

Stopping RFS

If you started RFS using **init 3**, you can stop it by going to a lower run state (**init 2**, **init S**, or **shutdown**). If you started RFS using **rfstart**, you can stop it by simply typing **rfstop**.

Before you can use **rfstop**, you must:

- Unadvertise all your resources (**unadv**).
- Unmount everything you have mounted from remote machines (**umount**).
- Make sure all your advertised resources are unmounted from remote machines. (can be forced by using **fumount**)

These steps will happen automatically when you leave **init** state 3.

If you are the primary name server, you should not stop RFS unless a secondary is up and ready to take over. If the primary goes down and no secondary is available to take over, computers in the domain that are not the primary or a secondary will be able to start RFS. Computers that are already running RFS will continue to run RFS; however, they will not be able to mount or advertise new resources.

Sharing Resources

This section describes how to share your local resources with other computers (advertising) on a Remote File Sharing system and how to use the resources other machines have made available (mounting).

Local Resource Advertising

The **adv** command is used to advertise a local directory (one that physically resides on your machine) so it is accessible to other machines. When you advertise a directory you must assign it a resource name. This resource must have a unique name within your domain.

When **adv** is performed with the options listed below, the resource is registered with your domain name server. Any computer that has access to your domain can find a listing of your resource from your domain's advertise table (**nsquery** command). A remote computer will not know the exact location of the resource on your machine. All the remote computer will know is its resource name, the short description you assign, that it resides on your computer, and the read/write permissions.

You can set up your system so resources are advertised automatically when you enter **init 3**. To do this, place the entire command line for each advertised resource in the **/etc/rstab** file.

The syntax of the **adv** command to advertise a resource is:

adv [**-r**] [**-d** "*description*"] *resource pathname* [*clients...*]

The syntax of **adv** to modify an advertised resource entry is either:

adv -m *resource* **-d** "*description*" [*clients ...*]

or

adv -m *resource* [**-d** "*description*"] *clients ...*

The options are as follows:

-r The **-r** option, for read-only, is used to advertise the resource with read-only access. If it is not used, read/write access is assumed.

-d	The **-d** option indicates that the next argument (*description*) is a description of the resource. The description can be from 0 to 32 characters and should be in quotes.
resource	This is the resource name you assign. The name is limited to 14 printable ASCII characters; slash (/), period (.), and spaces and tabs may not be used. (If you enter more than 14 characters, the name will be accepted and truncated.)
pathname	This is the full path name to the directory you want to share. The directory must be on your local system, and it cannot be already advertised.
clients...	This is an optional list of one or more remote machines or domain names to which you want to restrict this resource. (A domain name must have a period (.) appended to it.) If clients are not included, the resource will be accessible to any Remote File Sharing computer that can connect with your computer. You can also define aliases so a single name can represent a group of computers and/or domains. (See the section "Aliases" below.)
-m *resource*	This option is used to modify the *description* or *client* fields for an advertised resource listing. It cannot be used to change the read/write permissions of a resource.

Below are two examples of **adv** command lines:

adv -r -d "Department news" DNEWS /usr/news peanuts.

adv -d "My devices" MDEV /dev lucy linus doc.comp1

The first example advertises your **/usr/news** directory with read-only permissions under the resource name **DNEWS** to all computers in the **peanuts** domain. The second advertises your **/dev** directory as **MDEV** to computers **lucy** and **linus** in your domain and **comp1** in domain **doc**.

Automatic Advertising

You can set up all your **adv** commands to start automatically when the system enters the Remote File Sharing state (**init 3**). Do this by placing each full **adv** command line in the **/etc/rstab** file. As soon as **init 3** successfully starts Remote File Sharing, all **adv** commands in **/etc/rstab** will be run.

The following is an example of an **/etc/rstab** file to automatically advertise the two resources shown previously.

```
# cat /etc/rstab
adv -r -d "Department news" DNEWS /usr/news peanuts.
adv -d "My devices" MDEV /dev lucy linus compl.doc
#
```

Aliases

The **adv** command reads the **/etc/host.alias** file to find the definitions of any aliases in the *clients* field. The format of the file is as the following:

alias *name client1 client2 client3 ...*

where *name* is replaced by the character string you want to represent the list of clients. Each client can be a machine name, domain name, or an alias name previously defined in the file. The three fields must be separated by blanks or tabs. If you have too many *clients* to fit on one line, you can extend an entry beyond one line by entering a back slash, then a carriage return to continue to the next line.

Resource Security

These are the levels of security that protect your resource once you have advertised it.

Verify computers
: Only those remote computers that pass your security checks can even connect to your machine. You may have indicated that only those machines you have a record of can connect (see **rfstart -v**).

Restrict Resource
: You may have advertised your resource so only selected remote computers can mount it (see the *client* option of the **adv** command).

Map IDs
: The permissions remote users will have to your resources are set on a computer-by-computer basis. In other words, the user and group mappings you set up for a remote computer will apply to any of your resources that computer mounts.

UNIX system security
: Normal UNIX system access security, governing read, write, and execute permissions, will apply to any advertised resource.

Local Advertise Table

All advertised resources for your computer are contained in your computer's local advertise table. Any user can use the **adv** command with no options to display the local advertise table. The output will be a listing like the following:

```
# adv
CUSTOMER   /usr/bin/cust   read-only    "Atlanta customers"     lucy linus doc.tick
SCCS       /sccs           read/write   "Project Y source"      unrestricted
CALENDAR   /usr/bin/cal    read-only    "UNIX System calendar"  peanuts. compgrp
```

The information will match what you entered using the **adv** command, with appropriate options, for each resource. Some of the information shown was implied when the resource was advertised. For example, access is read/write if **-r** is not specified and clients are not limited to certain machines (**unrestricted**) when no clients are identified. The *clients* listed in the last field can be:

- Computer names in your domain (**lucy**)
- Computer names in other domains (**doc.comp1**)
- Domain names (**peanuts.**)
- Aliases listed in **/etc/host.alias** (**compgrp**)
- **unrestricted** if the resource is not restricted to certain machines.

Domain Advertise Table

All advertised resources for your domain are in the domain advertise table on your domain name server. The **nsquery** command is available to all users to list any or all of the advertised resources in a domain. The syntax of the command is as follows:

nsquery [**-h**] [*name*]

where the **-h** option can be used to suppress printing of the heading line and the *name* option can be replaced by one of the following:

nodename	To list the resources advertised by a particular computer in your domain.
domain.	To list all resources advertised by all machines in a domain. (A period at the end of a name causes it to be interpreted as a domain name.)
domain.nodename	To list all resources advertised by a particular computer in a domain outside your own domain.

If the *name* option is not used, **nsquery** will print a list of all advertised resources in your domain. Here is an example of output from an **nsquery** command:

```
# nsquery peanuts.lucy
RESOURCE     ACCESS        SERVER          DESCRIPTION

GRAPHICS     read/write    peanuts.lucy    Domain files
CALENDAR     read/write    peanuts.lucy    Monthly meetings
USERHELP     read-only     peanuts.lucy    System help information
```

For each available resource, **nsquery** will list the resource name (RESOURCE), the permissions (ACCESS), the computer that owns the resource (SERVER), and the description of the resource.

Note: The output from **nsquery** does not indicate whether you have permission to mount the resource.

Advertised Resources in Use

You can use the **rmntstat** command to determine which remote computers have mounted your advertised resources. This command can print output for all your resources or the one you choose. The syntax is as follows:

rmntstat [**-h**] [*resource*]

where **-h** will print the output without the heading and *resource* can be used to restrict output to information for a particular resource. Here is an example of the output from **rmntstat**:

```
# rmntstat
RESOURCE        PATH                    HOSTNAMES
DNEWS           /usr/news               peanuts.linus peanuts.lucy
MDEV            /dev                    peanuts.linus
SPECIAL         unknown                 peanuts.charlie
```

The output shows the resources your machine has advertised, where those resources are located on the local machine, and the computers that have mounted the resource.

Note: If **unknown** appears in the path name field, it means you have unadvertised the resource, but it is still mounted on the listed remote machines.

Unadvertise

You can unadvertise any of your computer's resources using the **unadv** command. It will remove the resource from the advertise tables on your computer and the domain name server.

The domain administrator can use **unadv** to unadvertise any resource within the domain. (You should only use **unadv** when the machine has gone down, otherwise, the domain and your computer's advertise tables will not match.)

Unadvertising does not remove a currently mounted resource from a remote computer [see **umount**(1M)]. It does, however, prevent additional machines from mounting the resource. There are two reasons you may want to use this command.

1. Before you can unmount (**umount** or **fumount**) one of your file systems containing an advertised directory, it must be unadvertised.
2. If you want to restrict a previously shared directory to only local access, you will want to unadvertise it.

Because advertise commands can be set up to run automatically in **init 3**, you may have to remove them if you want them permanently unadvertised. (See the "Advertise Resources" section of this chapter for information on how to modify the **/etc/rstab** file.)

The syntax of the command is:

unadv *resource*
or
unadv *domain.resource*

where *resource* is used to unadvertise one of your machine's resources, and *domain.resource* is used by a domain name server to unadvertise any resource in its domain. In the second case, the resource name is prefixed by the domain name in which it resides and a period (.).

Forced Unmount

You cannot unmount a local file system using the **umount** command if any part of that file system is mounted remotely. Normally, you should tell each administrator whose machine has mounted such a resource to unmount it. In this way, a resource can be removed in an orderly fashion.

When you have to unmount a local file system immediately, however, you can use the **fumount** command. The **fumount** command will remove a remotely-mounted resource from all machines that have mounted it. You should only do this in cases where it is urgent that the resource be removed, because you may be cutting off remote processes that are accessing the resource.

The syntax of the **fumount** command is as follows.

fumount [**-w** *sec*] *resource*

where the **-w** option says to wait *sec* seconds before remotely unmounting the resource and *resource* is replaced by the resource name.

When you execute **fumount**, the following is what happens:

1. The resource is unadvertised.
2. If the **fumount** command is executed with a grace period of several seconds, the following shell script is run on all client machines currently using the resource.

   /usr/nserve/rfuadmin fuwarn *resource sec*

   By default, this shell script will write to all terminals on all client machines:

   ```
   resource will be disconnected from the system in sec seconds.
   ```

   (You can edit **rfuadmin** to tailor the action taken in response to **fumount**.)
3. After the grace period of *sec* seconds, the resource is removed from all remote machines on which it is mounted. The following message is then sent to all terminals:

   ```
   resource has been disconnected from the system.
   ```

4. On each client machine, **rfuadmin** executes **rmount** and **rmount** will try to remount the resource every 60 seconds until it succeeds.

See **rfuadmin**(1M) and **rmount**(1M) for further information on processing these commands.

Remote Resource Mounting

You can attach another computer's advertised resource to your system using the standard **mount** command. Simply choose an existing directory, preferably empty, or create a directory to use as a mount point and **mount** the resource, using the **-d** option.

When you try to **mount** a remote resource, a request is sent to the computer that advertised the resource. If you have permission to mount the resource, the resource will be added to your mount table and connected to the mount point you specified. You can list the remote resources, as well as local file systems, mounted on your computer using the **mount** command without options.

The form of the **mount** command to mount a remote resource is as follows:

mount [**-r**] [**-c**] **-d** *resource directory*

The options are as follows:

-r The **-r** option indicates that the resource should be mounted read-only. If it is not used, the resource is mounted with read/write permissions. (A remote resource can only be mounted read/write if it was advertised that way.)

-d This option is used to indicate that you are mounting a remote resource.

resource This must be replaced by the resource identifier assigned by the computer that advertised the resource.

directory This must be replaced by the full path to the local directory on which you want to mount the resource.

-c This option indicates that remote reads and writes for the remote resource you mount should NOT be cached in the local buffer pool. You will generally want the default (buffer

caching on), since caching will cut down on network access and improve RFS performance. (See the "Monitoring" and "Parameter Tuning" sections of this chapter for more information on monitoring client caching activities.)

Automatic Remote Mounts

You can set up your **mount** commands to run automatically when your computer enters the **init 3** state. Do this by adding mount information to the **/etc/fstab** file. The format of **/etc/fstab** for remote mounts is as follows:

resource directory **-d[r]**

resource is replaced by the resource name, *directory* is replaced by the directory where the resource will be mounted, and **-d** says this is a remote mount. You can use **-dr** instead of **-d** if you want to mount the remote resource read-only.

Mounting Guidelines

Below are some guidelines that apply to resources.

- Once you have advertised a resource from your computer, you can:
  - Mount a local file system on a subdirectory of the advertised resource. The new file system will become part of the advertised resource. (You cannot mount directly on the advertised mount point, however.)
  - Mount a remote resource on subdirectories of your advertised resource. The remote resource you mount will not become part of the resource, however. Only your local users will be able to access it. (Remote users will be able to see this mount point directory, but will get a "multihop" error message if they try to access the directory in any way.)

Note: You cannot mount a remote resource directly on an advertised directory.

- If a resource was advertised with read-only permissions, you must mount it read-only. If it was advertised read/write, you have a choice of mounting it read-only or read/write.

Mounting Rules

There are some rules you must follow to avoid unexpected results when mounting remote resources.

Rule #1 Mounting over basic directories

A directory containing files that define your local machine should not be used as a mount point for a remote resource. This will result in essential local files being inaccessible to your system.

For example, you shouldn't mount a remote **/dev** on your machine's **/dev** directory or you will make your machine's console inaccessible (**/dev/console**). As another example, if you mounted an **/etc** directory on your **etc** directory, you would cover your local **inittab**, **passwd**, and **mnttab** files, to name a few.

Some other directories that fall into this category are: **/**, **/usr**, **/usr/bin**, **/usr/nserve**, **/usr/net**, and **/shlib**.

Rule #2 Mounting spool and work directories

Like Rule #1, Rule #2 has to do with mounting a directory from one computer on the same directory to another. In this case the problem is spool files and workspace directories. Applications such as **uucp**(1) and **lp**(1) can run into problems when multiple machines are trying to create spool files or lock files in the same directory. For example, if you share the **/usr/spool/locks** directory, by using a tty device for **uucp** on one machine, you would prevent use of a device of the same name on another machine. Also, mounting **/tmp** can cause collisions among temporary files.

Rule #3 File systems on remote devices

When a remote machine advertises a directory containing a device and that device contains a file system, you would not be able to mount the file system by simply mounting the resource containing the device. To access the file system on the remote device, the

remote machine would have to mount the device locally, then advertise that mount point. (You can access the remote as a raw device, however, by simply mounting the resource containing the device.)

Rule #4 Using remote sticky bit programs

Mounting remote resources that contain executable files with the sticky bit on can improve performance of those files. When executed on your machine, the text portion of the sticky bit program will remain in main memory on your machine, thereby reducing the network overhead on future executions. From your perspective as a client, you should be careful not to mount too many sticky bit programs or you could unknowingly use a lot of memory.

If your machine is a server sharing sticky bit files, you should be aware that they are treated differently from strictly local sticky bit files. Before removing sticky bit programs from an advertised resource, you must unmount the resource from all client machines [**fumount**(1M)], remove the program, then readvertise the resource. You should do this to prevent out-of-date text for recompiled or deleted files from remaining in memory on client machines. (See Chapter 6 for a discussion of local sticky bit use.)

Local Mount Table

You can list the remote resources that are mounted on your computer as you would list local file systems: the **mount** command with no options. Remote resource output from this command will appear in the following form:

directory **on** *resource permission* **on** *date*

where *directory* is the name of the directory where the remote *resource* is mounted, the *permission* is **read only/remote** or **read/write/remote**, and *date* is the time and date the resource was mounted.

The following is an example of output from the **mount** command, with no options. The last two entries in this example are remote resource mounts.

```
$ mount
/ on /dev/dsk/c1d0s0 read/write on Thu Jan 16 09:07:19 1986
/usr2 on /dev/dsk/c1d0s8 read/write on Thu Jan 16 09:07:32 1986
/usr on /dev/dsk/c1d1s2 read/write on Thu Jan 16 09:07:33 1986
/s/codes on LCODE  read/write/remote on Thu Jan 16 09:10:13 1986
/s/timing on TEMPO read only/remote on Thu Jan 16 09:10:27 1986
$
```

Remote Resource Disconnected

When a machine that shares its resources with you goes down or the network connection is broken, resources that you have mounted from that server will be disconnected.

A Remote File Sharing daemon process (**/usr/nserve/rfudaemon**) runs an administrative shell script (**rfuadmin**) to try to clean up when a resource has been disconnected. It then tries to remount the remote resource as soon as it becomes available again.

rfudaemon

The **rfudaemon** process is run automatically when Remote File Sharing is started (**rfstart**) and continues to run until it is stopped (**rfstop**). The **rfudaemon** process waits for one of the following events to occur, then passes that information to the **rfuadmin** administrative shell script.

disconnect When a link is cut to a remote resource, **rfudaemon** sends a disconnect message and the resource name to the **rfuadmin** shell script.

fumount When a resource is unmounted (**fumount**) by the server, the **rfudaemon** sends a fumount message and the resource name to the **rfuadmin** shell script.

fuwarn When a server sends a message that a resource is about to be unmounted (**fumount**), the **rfudaemon** sends a **fuwarn** message, the resource name, and the number of seconds before the resource will be unmounted to the **rfuadmin** shell script.

rfuadmin

When links to resources are disconnected, the response to the disconnect is handled at the user level by the **/usr/nserve/rfuadmin** shell script. By editing this shell script, you can tailor the response your system makes when the connection to a remote resource is lost. The **rfudaemon** process starts **rfuadmin** with one of the following arguments.

disconnect *resource* When **rfuadmin** is started by **rfudaemon** with these arguments, **rfuadmin** sends this message to all terminals using the **wall**(1) command:

resource `has been disconnected from the system.`

Then it executes **fuser**(1M) to kill all processes using the resource, unmounts the resource [**umount**(1M)] to notify the kernel, and starts **rmount** to try to remount the resource. The assumption is that the link was either broken by mistake or that as soon as the server makes the resource available again, the client will want to mount it.

fumount *resource* When **rfuadmin** is started by **rfudaemon** with these arguments, the processing is similar to a disconnect.

fuwarn *resource seconds*
When **rfuadmin** is started by **rfudaemon** with these arguments, **rfuadmin** sends this message to all terminals:

resource `is being removed from the system in` *sec* `seconds.`

There are many reasons you may want to change the **rfuadmin** shell script. If access to a resource is lost, you may want to respond by trying to mount another resource. You may want to send different messages when a resource is lost.

Note: When a resource is disconnected, **rfuadmin** tries to remount the resource using **/usr/bin/rmount**. This command retries the remount every 60 seconds until it succeeds. To change this behavior, you must either edit **/etc/rfuadmin** so it no longer does an **rmount**, or edit **rmount** so it retries a limited number of times [see **rmount**(1M)].

Unmounting

You can unmount any remote resource you have mounted with the **umount** command. The syntax for using **umount** to unmount a remote resource from your computer is as follows:

umount -d *resource*

where *resource* is replaced by the name of the resource you are unmounting.

Before you run **umount**, you should make sure none of your users are using the resource with the **fuser** command. When the **fuser** command is run, it lists the processes on your computer that are accessing a mounted remote resource. It can then be used to kill all processes relating to a resource.

The form of **fuser** for reporting on remote resources mounted on your machine is:

fuser [**-ku**] *resource ...*

where **-u** will list user names in the report of processes that have files open in any directory or subdirectory relating to the resource, and **-k** will kill all processes that have files open in any directory or subdirectory relating to the resource.

Sharing Printers

One of the common uses of RFS is for sharing printers. The concepts presented here may also apply to sharing other peripherals that use spool files.

On the server machine:

1. Set up **lp** on the server the way you normally would on any machine. (The *server* machine is the one that will do all the spooling). Make sure that the printer works and you are able to print text on this machine.
2. Advertise **/usr/spool/lp** to all the *client(s)* that will be using this printer.
3. In **/usr/nserve/uid.rules**, map the user id of **lp** to itself. Number 71 is usually the user id of **lp**, so the entry in **uid.rules** would be: **map 71:71**.

On the client machines:

1. Do not run the scheduler on the client machines. All you need on the client machines are the **lp** and the **lpstat** commands.
2. Mount the resource that was advertised by the server on the client's **/usr/spool/lp**.

The **-c** option of **lp** should be used for any user file that is not a shared resource. The **lp** commands on the client machines generate the file names for the spooler utilizing the process id. For RFS versions prior to Release 1.2, identical spool file names might be generated by multiple clients. If this happens simultaneously, there might be a collision in the spooling directory. The consequence of this is that one of the files will get lost, but the chances of this happening are very small.

Mapping Remote Users

Note: The "Complex User ID/Group ID Mapping" section in this chapter is designed to act as a tutorial for setting up ID mapping. This section provides further reference information to support that section.

Your computer has a set of users, defined in the **/etc/passwd** file. These users can also be members of groups that are defined in the **/etc/group** file. The user and group ID assignments are used by the system to evaluate requests by the user for access to local files, directories, and devices.

When you share your directories with other computers using Remote File Sharing, you have the ability to define the permissions each remote user will have to your resources. You do this by mapping remote users and groups into the permissions of existing users and groups on your computer. You also have the option of mapping remote users and groups into a special "guest ID" that does not map into permissions of any existing users and groups on your system.

If you do not want to map remote users, you do not have to. The default treats all remote users as the special guest ID, which has the ID number of MAXUID plus one. MAXUID is the maximum ID number defined for the system, so MAXUID+1 is always guaranteed not to overlap with any current or future users (by default, MAXUID+1 is 60001).

When a remote user checks the ownership of one of your resources, the user might see another special ID: MAXUID+2 (or 60002). No files will ever be owned by 60002 on the system where a file resides. MAXUID+2 is simply a way of telling remote users that a file or directory is not owned by them or any other users from their system. For example, if all users on a remote system were mapped to 60001, any files created by one of your local users would appear to be owned by user ID 60002.

How Mapping Works

When you set up your remote user and group mapping for a remote computer, you define how requests from users and groups will be handled. This mapping has an impact on the remote users' access to files and directories on your resources, as well as each remote user's view of ownership.

For example, say you map user ID **101** from machine **abc** into user ID **115** on your machine. When **101** from **abc** tries to create a file in a directory of one of your advertised resources, your machine will translate the request from **abc**'s **101** into a request from **115**. If local ID **115** has permissions to create a file in that directory, then the file will be created.

If you tried to **stat** the file on your machine (**ls -l**, for example) you would see that user ID **115** was the owner. However, if a **stat** comes from machine **abc**, your machine would do inverse mapping. Therefore, the user from **abc** would see the file as being owned by user ID **101**.

Inverse mapping from the machine that owns the resource (the server) provides the most consistent file system view to a remote user. It could potentially cause confusion. Continuing with the example, say that instead of just mapping **101** into **115**, you also mapped **102** from **abc** into **115** on your machine. A file created by **102** would correctly create the file as owned by **115** on your machine. However, when a user from **abc stat**s the file, it would always show ownership by the smaller numeric value: **101** user ID.

Note: This same result would occur if you gave several local user names the same numeric user ID.

If users are confused when files they create do not seem to belong to them, the situation described above could be the reason. This does not cause any problems with each user's ability to access the resource. However, it could break some programs that are dependent on local IDs. The most consistent way to map, however, is one-to-one remote to local IDs.

Mapping Components

You must use the **idload**(1M) command to do the user and group mappings. This command reads the user and group mapping rules you create, reads your computer's **/etc/passwd** and **/etc/group** files, if needed, and maps the remote users into your users' permissions. If you are using remote user and group names to map into your computer, you must have access to user and group lists from the remote computers, so **idload** can read the files and translate those names into the appropriate numeric ID numbers.

Rules Files

The rules files you create will tell **idload** how to map remote users. Both files are in **/usr/nserve/auth.info** under the names **uid.rules**, for user rules, and **gid.rules**, for group rules.

Figure 10-6 shows how the user rules file can be structured. The format of the group rules file is exactly the same. All lines in each file are optional.

global
default *local_id* | **transparent**
exclude [*remote_id-remote_id* ...] | [*remote_id*]
map [*remote_id:local* ...]

host *domain.nodename* ...
default *local* | **transparent**
exclude [*remote_id-remote_id* ...] | [*remote_id* ...] | [*remote_name* ...]
map [*remote:local* ...] | *remote* | **all**

Figure 10-6: Format of **uid.rules** and **gid.rules** Files

The following notation is used in the previous figure:

local_name = a local user name
local_id = a local user ID number
remote_id = a remote user ID number
remote_name = a remote user name
local = a local name or ID number
remote = a remote name or ID number.

A rules file is divided into blocks of information. Each block is either a **global** or **host** block. There is only one **global** block per file, but there can be one **host** block for each computer mapped.

global This line starts the block of global information. Each line of definitions after **global** and before the first **host** line will be applied to all computers that are not explicitly defined in **host** blocks. You can use **default**, **exclude**, and **map** inside **global** blocks.

You cannot map or exclude names in **global** blocks. You must use ID numbers.

host *domain.nodename ...*

This line starts a block of information for a particular computer. Each line of definitions following this line and before the next **host** line will be applied to the *domain.nodename* specified. You can use **default**, **exclude**, and **map** inside **host** blocks.

If you want to map more than one computer from a single set of **passwd** and **group** files, you can put several computer names on one line. In this case, **idload** will read the **passwd** and **group** files for the first computer referenced (if you map by name) and use the information in those files for all computers that are referenced.

A computer can only be mapped once in each rules file.

Each of the following lines of information can appear in either a **host** block or a **global** block. A name or an ID should only be mapped once in each block. If one is mapped more than once, the first reference is in effect and the others will produce warning messages from **idload**.

1. **default** *local* | **transparent**

   One **default** line can be put in each block to indicate how to handle remote users and groups that are not explicitly mapped or excluded.

   A **transparent** means use the same numeric ID on your machine that the user had on the remote machine for undefined users. So if a request comes from remote uid **101**, that request will have the permissions of local uid **101**.

   A *local* is replaced by a local user name or ID number. By default, all remote users will be mapped into the permissions of the local user indicated by name or ID. If a default line does not appear in a block, MAXUID+1 permission will be assigned.

2. **exclude** [*remote_id-remote_id*] | [*remote_id*] | [*remote_name*] ...

   Optional **exclude** lines can go in a block to exclude certain users from the default mapping. Zero or more ranges of ID numbers

(*remote_id-remote_id*), single *remote_names*, or single *remote_id* numbers can be excluded. (The *remote_name* is not available in the global block.)

A user who is excluded will still have access to your resources but will only have permissions of the MAXUID+1 user. All **exclude** lines must go before any **map** lines in a block.

3. **map** [*remote:local*] ¦ *remote* ¦ **all**

   You can use **map** lines in each block to assign local permissions to particular remote users. There are several ways to use the **map** command. You can set any remote user's permissions to any local user's permissions by either local user *id#* or *name;* separate the two with a colon (:). By entering a single *remote_id* or *remote_name*, the remote user who matches will have the permissions of the local user of the same ID or name. For example:

   map mcn

   would give the remote **mcn** the same permissions of the local user **mcn**.

   The literal entry **all** maps all users by user name in the permissions of users with the same name on your computer.

   Multiple **map** lines are valid. You cannot map by remote name in **global** blocks.

Note: The **map all** and mapping by name are not allowed in a global block. The **map all** will usually produce warning messages, since multiple administrative logins will have uid 0, and **idload** will try to map each one 0 to 0. There is no harm in this.

idload Command

Once the rules files are created, use the **idload** command to read your rules files and create mapping translation tables. When you run **idload**, the rules in **global** blocks and any **host** blocks that have resources currently mounted immediately take effect. All other **host** block rules will take effect when the remote machine mounts one of your resources.

The syntax of **idload** is:

idload [**-n**] [**-k**] [**-g** *g_rules*] [**-u** *u_rules*] [*directory*]

The options are as follows:

-n This is the "no update mode" option. When it is used, **idload -n** will print the mapping that would result from the rules files without putting them into effect.

-k This option shows the mapping that is currently active on your machine. (Note that there will be mapping ready to take effect that is not shown as active when you do not currently have a connection to a remote machine.)

-g *g_rules* This option lets you use a group rules file other than **/usr/nserve/auth.info/gid.rules** as input for group mapping rules.

-u *u_rules* This option lets you use a user rules file other than **/usr/nserve/auth.info/uid.rules** as input for user mapping rules.

directory This option indicates that some directory other than **/usr/nserve/auth.info** contains the *domain/nodename* directories where the **passwd** and **group** files for each remote computer reside. If it is not used, **/usr/nserve/auth.info** will be assumed.

Each time you set up or change your rules files, first run **idload** with the **-n** option. The results will show you the mapping that will occur when the command is run to actually load the IDs. You must then run **idload** for the rules to go into effect.

Remote Computer passwd and group Files

If you are mapping remote users by name, you will need lists of these users from each remote computer. These lists should be copies of the **/etc/passwd** and **/etc/group** files from each computer.

If **idload** finds a request for a remote user name in a **host** information block, it will check the directory for that computer for **passwd** and **group** files. The path name to the remote computer's directory will be **/usr/nserve/auth.info**/*domain*/*nodename* on your system, where *domain* and *node name* are replaced by the remote computer's domain and the remote computer's nodename, respectively (unless you overrule this using the **-g** and **-u** options).

Note: Mapping by name can be a very useful feature. However, if you map only by ID number or local name, and avoid mapping by remote names, you will avoid the need to coordinate distributing and updating remote **passwd** and **group** files and rerunning **idload**.

Example Rules Files

This section describes some strategies you can use to map users. It describes the easiest way to deal with remote user permissions and progresses to the most complicated ways. Read through each example to decide what strategy is best for your computer.

No Mapping

If you do not run **idload** to map users, all remote users will have the permissions of the user ID number MAXUID +1, which is the maximum ID number defined on your system plus one. Because there are no users on your system with that user ID number, remote users will only have access to files created by your users that are open to all users.

Mapping Remote IDs

If you map remote users using remote ID numbers and local ID numbers and names, you do not need to get any **passwd** and **group** files from remote computers. The following displays contain some simple examples of mapping that only involve remote ID numbers.

In Figure 10-7, all remote user IDs will be mapped into the same user ID permissions on your computer, except for **root** (ID number **0**), which would only have special guest permissions. This would apply to all remote computers.

Caution: The *exclude 0* line is strongly recommended to prevent possible security breaches from root users on other systems.

```
global
default transparent
exclude 0
```

Figure 10-7: **uid.rules** File: Setting Global Defaults

In Figure 10-8, users have the same permissions as in the previous example, except remote user IDs 0 through 100 will have MAXUID +1 permissions, and any user ID **732** would have the same permission as local user ID **106**.

```
global
default transparent
exclude 0-100
map 732:106
```

Figure 10-8: **uid.rules** File: Global Mapping by Remote ID

In Figure 10-9, the users from computer **lucy** in domain **peanuts** will not be mapped by the global rules. Instead, all users will have the permissions of local user **mpg** except that user IDs 0 through 50 will have MAXUID +1 permissions.

```
global
default transparent
exclude 0-100
map 732:106

host peanuts.lucy
default mpg
exclude 0-50
```

Figure 10-9: **uid.rules** File: Host Mapping by Remote ID

Mapping Remote Names

If you want to use specific remote user names to map into your local users' permissions, you will need to have access to **passwd** and **group** files from those computers on your system. Following are some examples of ways you can map remote user names.

map all

If you have the same set of user names on different machines, but the user IDs differ, you may want to use **map all** as shown in Figure 10-10.

```
global
default transparent
exclude 0

host peanuts.lucy
exclude mary 0 uucp
map all
```

Figure 10-10: **uid.rules** File: Mapping by Name With **map all**

In Figure 10-10, each user name from computer **lucy** in domain **peanuts** will have the same permissions as the same user name on your computer. The only exceptions will be users **mary**, **root**, and **uucp** who will have MAXUID +1 permissions.

map name:name

You can also map particular remote user names into local user names or user IDs on your computer. Figure 10-11 is an example.

```
global
default transparent
exclude 0

host peanuts.lucy
default transparent
exclude 0
map mcn:jcb ral gwn:103
```

Figure 10-11: **uid.rules** File: Mapping Specific Users by Name

In Figure 10-11 all users from the computers will be mapped into their same user ID with the following exceptions. Remote user **mcn** will have the permission of local user **jcb**, remote user **ral** will have permissions of local user **ral**, and remote user **gwn** will have permissions of local user ID **103**.

List Current Mapping

There are two ways to list the mapping you have set up: **idload -n** and **idload -k**. The **-n** option inspects the rules files and prints a listing of what would be in effect were you to load them. The **-k** option prints the mapping that is currently in effect in the kernel.

Figure 10-12 shows the result of **idload -n** used for the example shown in Figure 10-11. (The **gid.rules** file simply has the global block set at **default transparent**.) The **-n** option says to print the mapping that is set up in the rules file. You should do this before you run **idload** without options so you can see the mapping that will take effect.

```
# idload -n

TYPE  MACHINE        REM_ID     REM_NAME     LOC_ID        LOC_NAME

USR   GLOBAL         DEFAULT    n/a          transparent   n/a
USR   GLOBAL         0          n/a          60001         guest_id
USR   peanuts.lucy   DEFAULT    n/a          transparent   n/a
USR   peanuts.lucy   0          n/a          60001         guest_id
USR   peanuts.lucy   100        mcn          105           jcb
USR   peanuts.lucy   102        gwn          103           n/a
USR   peanuts.lucy   191        ral          101           ral

GRP   GLOBAL         DEFAULT    n/a          transparent   n/a
```

Figure 10-12: Output From **idload -n**

If you were to then run **idload**, the mapping shown above would take effect. If you were then to run **idload -k**, and the machine called **peanuts.lucy** did not have a resource mounted, you would see that the output in Figure 10-13 was active.

```
# idload -k

TYPE   MACHINE        REM_ID      REM_NAME     LOC_ID          LOC_NAME

USR    GLOBAL         DEFAULT     n/a          transparent     n/a
USR    GLOBAL         0           n/a          60001           guest_id

GRP    GLOBAL         DEFAULT     n/a          transparent     n/a
```

Figure 10-13: Output From **idload -k**

All mapping to **peanuts.lucy** would be active as soon as you are connected to it.

Note: The output from **idload** with the **n** and **k** options could be different if you have changed the rules files, but not yet run **idload** without options. Also, the **k** option will not show mapping for computers that are not currently mounting a resource from your machine, even though the mapping would be in effect as soon as the remote machine mounted one of your resources.

Domain Name Servers

One machine in each RFS domain must be chosen to be the primary name server, and zero or more can be secondary name servers. The duties of these machines are described briefly under the "Name Service" heading in the "Overview" section of this chapter. This section describes the "how-to" of being a name server.

Before you run any of these tasks, you will want to know which machines are assigned as name servers and which machine is the current name server. To find out the current name server, type:

rfadmin

To find out the name server assignments, type:

cat /usr/nserve/rfmaster

The line in the **rfmaster** file that has a **P** in the second field designates the primary name server. If an **S** is in the second field, the entry designates a secondary name server. (Lines with **A** in the second field designate the network address of a primary or secondary.)

Primary Name Server

If your machine is the primary domain name server, you are responsible for maintaining domain information. The "Setting Up Remote File Sharing" section of this chapter describes primary name server responsibilities as you set up your machine and domain. You may want to refer to the paragraphs in that section if you want to change your RFS configuration after initial configuration.

- Create **rfmaster** File
- Add/Delete Domain Members
- Resource Sharing With Other Domains
- Multiple Domain Name Service

Note: If you want to change the primary and secondary designations in the **rfmaster** file for a domain that is currently running, you must follow this procedure to make sure those changes are properly put in place.

1. Stop Remote File Sharing on all primary and secondary domain name servers for the domain (**rfstop** or **init 2**).
2. Change the **rfmaster** file on the old primary and the new primary.
3. Start the primary designated in the new **rfmaster** file (**rfstart** or **init 3**).
4. Start the secondaries designated in the new **rfmaster** file (**rfstart** or **init 3**).

Once changed on the name servers, each individual computer will pick up the change the next time it starts Remote File Sharing.

Secondary Name Server

Because a secondary name server is only intended to take over domain name service temporarily, its main responsibility is to pass name server responsibility back to the primary as soon as possible. It does not happen automatically! Most domain maintenance (adding new computers or changing the **rfmaster** file) cannot be done while the secondary is acting domain name server. The secondary simply maintains information machines need to mount and advertise resources.

To pass name server responsibility back to the primary once it is again running RFS, type the following from the secondary:

rfadmin -p

The **rfadmin -p** command will pass the domain name server information to the primary, or one of the other computers listed in the domain's **rfmaster** file if it cannot contact the primary. (Note that name service will automatically be passed off when the current name server goes down.)

Recovery

As a domain name server, computers in your domain rely on your machine for information on domain resources and domain member machines. Remote File Sharing is designed to recover quickly when communication is cut between machines and the name server. The following sections describe Remote File Sharing events that can occur and the recovery mechanisms designed to handle them.

Primary Goes Down

All essential domain records are maintained on the primary domain name server. The primary regularly distributes the most critical of these records to secondary domain name servers. (These records do not include files and directories under **/usr/nserve/auth.info**.)

If the primary goes down, domain name server responsibilities are passed to the first secondary name server listed in the **rfmaster** file. The secondary is only intended to take over temporarily. The reason is that a secondary has limited name service capabilities. This is done to maintain the definitive domain records on the primary. Changing the name server does not affect any currently mounted resources.

While a secondary is acting domain name server, these functions cannot be done:

- Maintaining domain member lists

  Computers cannot be added or deleted from domain member lists while a secondary is acting domain name server.

- Changing RFS passwords

  Neither the secondary nor another computer can change RFS authentication passwords while a secondary is acting domain name server.

The secondary will maintain lists of advertised resources for the domain and continue basic name server functions so Remote File Sharing activities can continue. In most cases, the computers in the domain should not be aware the primary is down. When the primary comes back up, the secondary should pass name server responsibilities back to the primary using the **rfadmin -p** command.

Note: When a primary crashes without properly shutting down Remote File Sharing and passing name server responsibilities to a secondary in an orderly fashion, the advertise table on the secondary may contain some errors. Resources from the primary may still be listed as available and recently advertised resources from other computers may not appear on the list. You can fix the domain advertise table using **unadv** and **adv -m** commands from the domain name server.

Primary and Secondaries Go Down

If all primary and secondary name servers go down at once, all information on advertised resources will be lost. Active mounts and links, however, are not disturbed. The problem is that when the primary comes back up, each computer will still think its resources are advertised but the primary will have no record of these advertised resources.

As soon as the primary is running, each computer can make sure its advertised resources are in sync with those listed on the primary in one of two ways:

- Readvertise with **-m**

  This is a less drastic way to update the advertise tables on the primary. Readvertise each resource using the **adv -m** command from the computer where the resource resides. This command will get the primary and remote computer's advertise tables back in sync.

- Restart Remote File Sharing

  You can bring down Remote File Sharing, bring it back up, and then readvertise your resources. This can be done automatically by going from **init 3** to **init 2** to **init 3**.

Monitoring

This section describes the commands used to monitor Remote File Sharing activity, the reports they produce, and possible action you can take to make sure that your system is operating at peak efficiency. In general, these reports can help you decide if you want to:

- Change parameter settings to match the way your system is used
- Move resources from machines with heavy RFS traffic to machines with lighter traffic
- Use sticky bit programs across the network

  (See the "Mounting Guidelines" section of this chapter for special rules relating to sharing sticky bit programs.)

A description of all Remote File Sharing tunable parameters and suggested initial settings appear at the end of this section.

The **-D** option of **sar** is used to produce RFS-specific information along with standard **sar** reports (**c**, **u**, and **b** options).

Note: The **sar** command, along with other performance analysis tools, are in the System Performance Analysis Utilities. You must install these utilities before you can use the performance analysis tools.

Remote System Calls (sar -Dc)

Your computer collects data each time a system call sends a message across a Remote File Sharing network to access a remote file. You can print this information using **sar -Dc** (see Figure 10-14).

The report produced by **sar -Dc** contains the average system calls per second; average read and write system calls per second, including average characters read and written per second; and average **execs** per second.

Information is divided into three categories: incoming requests (another computer's request for your resources), outgoing requests (your computer's request for a remote resource), or strictly local system calls.

```
$ sar -Dc
lucy lucy 3.0 2 3B2   02/14/86

00:00:04     scall/s  sread/s  swrit/s  fork/s  exec/s  rchar/s  wchar/s
01:00:04
   in              4        1        2            0.00      350      220
   out             3        2        1            0.00      240      300
   local         133       30       12    0.73    1.33    11202     3813
02:00:04
   in              4        1        2            0.00      350      220
   out             3        2        1            0.00      240      300
   local         133       30       12    0.73    1.33    11202     3813
03:00:02
   in              4        1        2            0.00      350      220
   out             3        2        1            0.00      240      300
   local         133       30       12    0.73    1.33    11202     3813
04:00:02
   in              4        1        2            0.00      350      220
   out             3        2        1            0.00      240      300
   local         133       30       12    0.73    1.33    11202     3813

Average
   in              4        1        2            0.00      350      220
   out             3        2        1            0.00      240      300
   local         133       30       12    0.73    1.33    11202     3813
$
```

Figure 10-14: Output From **sar -Dc**

Note: Some statistics will not reflect the actual number of messages sent across the network, since the client caching feature allows some remote read requests to be satisfied from data in local buffers. Outgoing scall/s, sread/s, and rchar/s fields include statistics for these read "hits" of remote data in the client cache. Though these reads do not result in actual messages to the remote machine, they are still categorized as outgoing, since they access remote data.

The following paragraphs describe how information from the **sar -Dc** report can be useful to you. If performance is poor, you can see how efficiently system read and write calls to and from your computer are using the Remote File Sharing network. For incoming (in) and outgoing (out) system calls, divide the characters read or written by the reads and writes, respectively.

If your computer is attempting more than about 30 remote system calls per second (in and out scall/s), you are probably nearing capacity. Performance problems will probably result from this much demand. Remote **exec**s also put a heavy demand on a computer. Selective use of sticky bit programs can help improve performance.

You may want to consider moving resources to machines where they are most in demand. (See the **fusage** command to determine what resources are being used most heavily.)

CPU Time (sar -Du)

You can list the percent of total Central Processing Unit (CPU) time spent on system calls from remote computers (%sys remote) with the **sar -Du** command (see Figure 10-15).

```
$ sar -Du
lucy lucy 3.0 2 3B2   02/14/86

00:00:04     %usr     %sys     %sys     %wio   %idle
                      local   remote
01:00:04        7        21       10       28      44
02:00:04       11         9       10        4      76
03:00:02        8        18       10       17      57
04:00:02        2         4       10        1      93
05:00:03        1         4       10        1      93
06:00:02        2         5       10        2      91
07:00:02        1         4       10        1      94
08:00:02        2         5       10        2      91
08:20:02       26        16       10       11      48
08:40:02       18        11       10        9      62
09:00:17       25        21       10       13      41
09:20:18       23        21       10       11      45
09:40:20       21        24       10       15      39
10:00:09       21        29       10       17      33
10:20:14       29        28       10       13      31
10:40:18       19        20       10        7      54

Average         9        12       10        8      71
$
```

Figure 10-15: Output From **sar -Du**

If the percent of CPU time spent servicing remote system calls is high, your local users may be suffering. (However, if the computer is a server machine, you would expect %sys remote to be high.)

To reduce the time spent servicing remote requests, you may want to place the resource(s) in demand on another computer (see the **fusage**(1M) command) or limit resource access by changing some of the tunable parameters. (See the section titled "Parameter Tuning.") You may also want to make sure clients are doing I/O in an efficient way (see **sar -Dc**).

Client Caching (sar –Db and sar –C)

The client caching feature of RFS improves RFS performance by reducing the number of times data are retrieved across the network. With client caching, the first read of data will bring the data into local buffers. Once data are in the local buffer, they will remain there so subsequent reads can get the data locally.

Client caching is assigned by default on a system-wide basis (RCACHETIME parameter) and when you mount a remote resource. You will almost always want to take advantage of the improved performance of client caching. There are only two very rare occasions when you may not want to use client caching.

- If buffer space is limited on your system, you may choose to turn off client caching for some resources or the entire system.
- If you are using programs that do their own private network buffering, you may not want to use client caching.

You can produce two **sar** reports to monitor caching activities.

Caching Buffer Usage

The **-b** option of **sar** reports the buffer pool usage for local (disk) reads and writes. The **sar -Db** option reports the same information, plus information on buffer pool usage of locally mounted remote resources (see Figure 10-16).

```
$ sar -Db

charlie charlie 3.1 2 3B2    09/03/86

14:37:15 bread/s lread/s %rcache bwrit/s lwrit/s %wcache pread/s pwrit/s
14:37:18
   local     2       40      93       1       3       64       0       0
   remote    1       11      92       1       1        0
14:37:21
   local     2       39      92       1       3       63       0       0
   remote    0       10      94       1       1        0
14:37:24
   local     2       40      93       1       3       64       0       0
   remote    1       12      93       1       1        0

Average
   local     2       40      93       1       3       64       0       0
   remote    1       11      93       1       1        0
```

Figure 10-16: Output From **sar -Db**

The fields on this report are as follows:

bread/s — The number of read buffer misses per second. (Each miss results in a read message to the server.)

lread/s — The number of read cache accesses per second.

%rcache — Read cache hit ratio [100 – ((breads)/(lreads) * 100)].

bwrit/s — Number of write buffer *misses* per second. All writes are sent to the server. Cache buffers affected by the writes are updated (write-through policy). This field indicates the numbers of lwrit/s that did not require a write-through. If data did not require a write-through it means that no data affected by the lwrit/s were present in the cache. (The information in this field has no performance implications when comparing using caching versus not using caching.)

lwrit/s — The total write cache accesses per second.

%wcache Write cache hit ratio [100 – ((bwrits)/(lwrits) * 100)].

pread/s Not reported for remote use.

pwrit/s Not reported for remote use.

Cache Consistency Overhead

Information on the overhead related to maintaining cache consistency is listed with **sar -C** (see Figure 10-17).

```
$ sar -C

charlie charlie 3.1 2 3B2    09/03/86

14:36:56   snd-inv/s   snd-msg/s   rcv-inv/s   rcv-msg/s dis-bread/s   blk-inv/s
14:36:59         0.0         1.1         0.0         1.5         0.0         0.2
14:37:02         0.0         0.6         0.0         0.5         0.0         0.4
14:37:05         0.3         0.6         0.0         0.5         0.0         0.1

Average          0.1         0.9         0.0         0.8         0.0         0.2
```

Figure 10-17: Output From **sar -C**

The fields on this report are as follows:

snd-inv/s The number of invalidation messages sent by the server per second to inform client machines about changes to server files.

snd-msg/s The total number of outgoing RFS messages sent per second.

rcv-inv/s The number of invalidation messages received by the client from the server. Each message informs the client that the contents of one or more of its cache buffers may have been modified by a write on the server. The client machine reacts by invalidating data in the affected buffers, so the buffers can be used for other purposes.

rcv-msg/s — The total number of incoming RFS messages received per second.

dis-bread/s — When an invalidation message is received, caching is turned off until the writing process closes or until a time interval has elapsed (set by the tunable parameter RCACHETIME). This counter tracks the number of buffer reads that normally would be eligible for caching in a resource with caching turned on, but that are not added to the buffer pool because caching for this resource is temporarily turned off. It indicates the penalty of running uncached and provides a basis for tuning the RCACHETIME parameter.

blk-inv/s — The number of buffers removed from the client cache as a result of receiving an invalidation message while a remote file is open or re-opening a remote file that has been modified since the last close on the client.

Server Processes (sar –S)

Every request from a remote computer to access your resources is handled by a server process. When there are too many requests for the servers to handle, they are delayed and placed on the request queue. Requests leave the request queue when servers are available. Information on server availability and requests awaiting service are listed with **sar –S** (see Figure 10-18).

```
$ sar -S

lucy lucy 3.0 2 3B2    02/14/86

00:00:04  serv/lo-hi   request      request   server    server
                3 - 6     %busy    avg lgth   %avail avg avail
01:00:04      3            0          0        100        3
02:00:04      3            0          0        100        3
03:00:04      4           80          8         20        2
04:00:04      6          100         25          0        0

Average      15           50         15         70        2
$
```

Figure 10-18: Output From **sar -S**

As an administrator you can set the number of server processes available to service remote system calls (see "Parameter Tuning"). There are two server variables you can set: MINSERVE and MAXSERVE. MINSERVE is the number of servers that are initially running to service remote requests.

MAXSERVE is the maximum number of servers that may ever exist. If demand goes beyond what the MINSERVE servers can handle, extra servers can be dynamically allocated so the total number of servers can be as high as the value of MAXSERVE. These processes disappear when they are no longer needed.

Information from **sar -S** can be used to tune your server parameters as shown below.

Too Few Servers

If the receive queue is almost always busy (request %busy), you may want to raise the number of servers. Here is how to decide the parameter to raise:

- Raise the MAXSERVE if the total average servers is high.
- Raise the MINSERVE if the total average servers is low.

Too Many Servers

If servers are available nearly 100% of the time (server %avail), you may have allocated too many servers. To decide the parameter to lower:

- Check the number of total servers. If this number is near the MINSERVE value, you can lower MINSERVE. Try reducing it by 50% or by the number of idle servers.
- Check the total servers that are idle. If this number is near the MAXSERVE value, you can lower MAXSERVE. Try reducing it by 50%.

Resource Usage (fusage)

You can find out how extensively remote computers are using your resources with the **fusage** command. It reports how many kilobytes were read and written from your resources, broken down by remote computers that have access to the resources. The form for **fusage** for reporting on a resource you have advertised is:

fusage *advertised-directory*

where *advertised-directory* is the full path name to one of the directories you have advertised. The **fusage** with no options produces a full report of data usage for all disks and advertised directories on your system, as shown in Figure 10-19.

```
# fusage

FILE USAGE REPORT FOR charlie

        /dev/dsk/c1d0s0       /

                              /
                                   charlie      649 KB
                                 Clients          0 KB
                                   TOTAL        649 KB

        /dev/dsk/c1d0s8       /usr2

                              /usr2
                                   charlie      563 KB
                                 Clients          0 KB
                                   TOTAL        563 KB
```

Figure 10-19: Output From **fusage**

If a remote computer's requests for your resources are high, it may be causing performance problems on your computer. With the output from **fusage**, you can see what resources are being particularly hard hit. You may then decide to move the resource. You may want to move or copy a resource to a computer that is constantly accessing it.

Remote Disk Space (df)

You can use the standard **df** command with a remote resource name to see the space left on the disk on which the remote resource resides. The form of the command to report on a remote resource is:

df *resource*

where *resource* is the name of a remote resource mounted on your machine. (The **df** command with no options will produce information for all mounted remote resources, plus all locally mounted devices.) Figure 10-20 contains an example of the **df** command using resource names as options.

```
# df USERsrc USERmail

/usr/src    (USERsrc     ):     5436 blocks     2202 i-nodes
/usr/mail   (USERmail    ):     5436 blocks*    2202 i-nodes
```

Figure 10-20: Output From **df**

Note: When multiple remote resources are reported that reside on the same disk, all listings of space on that disk, after the first, will be noted with an asterisk.

If you have write permission to a resource, you have as much access to file system space as a user on the system who owns a resource. This command will tell you the potential disk space available for you to write in. (Note that the space reported will only be for the top file system related to each resource.)

Parameter Tuning

There are several parameters you can tune to best suit the way you use Remote File Sharing. Remote File Sharing parameters control the amount of resources you devote to Remote File Sharing service. Each network transport provider may also have some tunable parameters that may affect performance characteristics of that particular network. See the network documentation for your network for more details.

All parameters have set default values that should work well for an average system (see the table at the end of this section). The following paragraphs describe these parameters and cases where you may want to change them.

RFS Parameters

RFS parameters define the extent the remote computers can use your resources, but they also control your own remote access to remote resources. If the values are too small, you may not be providing enough resources to properly handle your Remote File Sharing load. Requests for mounts, advertises, or even a file could fail if either of those values reach the maximum number allowed for your machine. If these parameters are too large, you could be allocating more system resources than you need to use.

The parameters described below are in **/etc/master.d/du**, except for NSRMOUNT, which is in **/etc/master.d/kernel**. Read these files for default values. If you change any of these values, see Chapter 6 for information on how to make the updated parameters take effect.

NRCVD (maximum number of receive descriptors)
: Your system creates one receive descriptor for each file or directory being referenced by remote users and one for each process on your machine awaiting response to a remote request. If you limit the number of receive descriptors, you limit the number of local files and directories that can be accessed at a time by remote users. The result of exceeding the limit would be error messages for remote user commands.

NSNDD (maximum number of send descriptors)
: For each remote resource (file or directory) your users reference, your system creates a send descriptor. A

send descriptor is also allocated for each server process and each message waiting on the receive queue. You can change this value to limit how many remote files and directories your machine can access at a time. This would, in effect, limit the amount of Remote File Sharing activities your users can perform. The result of exceeding the limit would be error messages for user commands.

NSRMOUNT (server mount table entries)

Each time a remote machine mounts one of your resources, an entry is added to your server mount table. This number limits the total number of your resources that can be mounted at a time by remote machines.

NADVERTISE (advertise table)

An entry is placed in your advertise table for each resource you advertise. This parameter sets the maximum resources you can advertise.

MAXGDP (virtual circuits)

There are up to two connections (virtual circuits) set up on the network between you and each machine with which you are currently sharing resources. There is one for each computer whose resources you mount and one for each computer that mounts your resources. A virtual circuit is created when a computer first mounts a resource from another, and it is taken down when the last resource is unmounted.

This parameter limits the number of Remote File Sharing virtual circuits your computer can have open on the network at a time. It limits how many remote computers you can share resources with at a time. Note that a given network may have a limited number of circuits on any one computer, so this parameter influences the maximum percentage of those that might be used for Remote File Sharing.

MINSERVE (minimum server processes)
Your system uses server processes to handle remote requests for your resources. This parameter sets how many server processes are always active on your computer. (See the **sar -S** command for information on monitoring server processes.)

MAXSERVE (maximum server processes)
When there are more remote requests for your resources than can be handled by the minimum servers, your computer can temporarily create more. This parameter sets the maximum total server processes your system can have (MINSERVE plus the number it can dynamically create).

NRDUSER This value specifies the number of receive descriptor **user** entries to allocate. Each entry represents a client machine's use of one of your files or directories. While there is one receive descriptor allocated for each file or directory being accessed remotely (NRCVD), there can be multiple receive descriptor **user** entries for each client using the file or directory (NRDUSER). These entries are used during recovery when the network or a client goes down. This value should be about one and one half times the value of NRCVD.

RFHEAP This value specifies the size in bytes of an area of memory set aside for RFS information. It contains the following information:

- The user and group ID mapping tables and the domain name of each machine currently sharing a resource(s) with your machine.
- A list of machine names supplied as a client list when you advertise resources.

The appropriate size for RFHEAP depends on:

- UID/GID tables (size and number).

  There will always be two global tables, one UID and one GID. Also, any machine with a **host** entry in **uid.rules** or **gid.rules** files will have

a table corresponding to each of these entries while it is connected to this machine. Machines that do not have separate entries in one of these files do not take any extra space.

To estimate the size on an individual table, type **idload -n**. There will be one 4-byte table entry per line of output from **idload**, plus up to 24 bytes of overhead per table.

- adv client lists (size and number)

Each advertise may have a list of authorized clients attached to it. This list is stored in this area, with its size unchanged, until the resource is unadvertised.

- currently connected resources

Each connection will use a maximum of 64 bytes to store the name of the connected resource. This memory is allocated dynamically, so some additional space is required to account for possible fragmentation as space is allocated and deallocated.

Since the total size is likely to be relatively small, 1 to 4 kilobytes, it is best to allow too much rather than not enough space.

NLOCAL (local access buffers)

This parameter sets the minimum number of local buffers, available from the common buffer pool, reserved for local access. RFS client caching shares the common buffer pool with the local accesses (usually disk or tape). This value, therefore, protects local data from adverse effects of competition with RFS buffer use.

When this threshold is turned off (set to 0), it defaults to the recommended value of one third of the entire buffer pool (NBUF). A non-zero value of NLOCAL overrides this default.

Note that if RFS is not running or has had no recent activity, the entire buffer pool will be available to local access.

NREMOTE (remote access buffers)

This parameter sets the minimum number of local buffers, available from the common buffer pool, reserved for remote resource read data. When this threshold is turned off (set to 0), it defaults to the recommended value of one third of the entire buffer pool (NBUF). A non-zero value of NREMOTE overrides this default.

Note that the sum of NREMOTE and NLOCAL must not be greater than NBUF. If this condition is detected, a console warning message is printed and the default value (one third of NBUF) is used for both NREMOTE and NLOCAL.

RCACHETIME (caching time off)

This parameter can be used in two ways: 1) to turn off caching for your entire machine; 2) to define the number of seconds that network caching is turned off when a file is modified.

To turn off caching for your entire machine, the parameter must be set to -1.

The second use of RCACHETIME requires some explanation. When a write to a server file occurs, the server machine sends invalidation messages to all client machines that have the file open. The client machines remove data affected by the write from their caches. Caching of that file's data is not resumed until the writing processes close the file or until the seconds in this parameter have elapsed.

The assumption is that write traffic is "bursty" and that the first write may be closely followed by other writes. Turning off caching avoids the overhead of sending invalidation messages for subsequent writes.

NHBUF (hash buckets)

While this parameter is not exclusively an RFS parameter, it has implication for RFS. The value of NHBUF will now be used to specify how many "hash buckets" to allocate for remote data in the buffer pool, as well as for local data. The hash buckets are used to search for a buffer given a remote server machine ID and file ID, rather than a linear search through the entire list of buffers. (See Chapter 6 for another discussion of NHBUF.)

Figure 10-21 lists the Remote File Sharing parameters and recommended values for different uses of Remote File Sharing. "Client Only" means that your machine will only be using remote resources, not sharing any from your own machine. "Server Only" means you will only offer your resources to other machines without mounting any remote resources. "Client+Server" means you will both offer local resources and use remote resources.

RFS Tunable Parameter Settings
(file /etc/master.d/du)

Parameter	Client Only			Server Only			Client+Server			Default Value	Size per Entry in Bytes
	2M	3M	4M	2M	3M	4M	2M	3M	4M		
NSRMOUNT*	0	0	0	50	50	50	50	50	50	50	24
MAXGDP	10	15	20	24	32	32	24	32	32	24	104
NADVERTISE	0	0	0	25	25	25	25	25	25	25	32
NRCVD	40	60	80	300	400	500	150	250	350	150	48
NRDUSER	0	0	0	450	600	700	225	375	525	225	24
NSNDD	150	250	350	30	30	30	150	250	300	150	44
MINSERVE	0	0	0	3	3	3	3	3	3	3	9K
MAXSERVE	0	0	0	6	6	6	6	6	6	6	-
RFHEAP	2048	2048	2048	3072	3072	3072	3072	3072	3072	3072	1
NREMOTE	0	0	0	0	0	0	0	0	0	0	-
NLOCAL	0	0	0	0	0	0	0	0	0	0	-
RCACHETIME	10	10	10	10	10	10	10	10	10	10	-

* This tunable is found in **/etc/master.d/kernel**.

Figure 10-21: RFS Tunable Parameter Settings

Appendices, Glossary, Index

Appendix A: Device Names and Designators

Introduction

Standard device names are used to identify the floppy disk, hard disk, and SCSI Cartridge Tape Drives. The term "integral" is used to define devices driven by a controller on the system board. The floppy disk drive is an integral device 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 floppy disk drive, the controller/drive designator is **c0d0** for both the raw and block device.
- For the SCSI hard disk drives with ESDI interfaces, the controller/drive designators are **c1t1d0** (single-disk system), or **c1t1d0** and **c1t1d1** (dual disk system) for both the raw and block device.
- For the hard disk drives with embedded SCSI controllers, the controller/drive designators are **c1t1d0** (single-disk system), **c1t1d0** and **c1t3d0** (dual disk system) or **c1t1d0**, **c1t3d0**, and **c1t4d0** (triple disk system).
- For the first SCSI Cartridge Tape Drive, the controller/drive designator is **c1t2d0** for the raw device. The SCSI Cartridge Tape Drive can only be a character (raw) device.

The partition or section designator is appended to the drive control/drive designator.

The number of possible hard disk partitions is 16 (0 through f hexadecimal). The other devices remain at a maximum of 8 partitions (0 through 7). The partitioning for the floppy disk is fixed by the processes used to format the media. The hard disk partitioning can be configured to best support the system application.

Certain of the device partitions are used for specific functions. These functions are required when a device is a bootable device. 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 by the **sysdump**(8) command to write the system image after a crash.
- Partition 4 is used by the **sysadm tapepkg**(1) command to move the data from the Operating System Utilities cartridge tape as it is installing the different utilities.
- Partition 6 specifies the entire device.
- Partition 7 specifies the boot area of a device.
- Partition 8 is used as the first hard disk area for user login directories (the default file system name is **usr2**).
- Partition 9 is used as the second hard disk area for user login directories (the default file system name is **usr3**).

Device partitions should fall on cylinder boundaries to obtain the best possible file system performance. For a root device, 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 of the root device. The otherwise wasted space in the boot block is used in the swap area without degrading the system performance.

SCSI Device Names and Designators

The SCSI information in this appendix should be used to supplement or update information in this guide. The device files for the SCSI devices are found under the **/dev** directory. The device file naming convention for SCSI devices is slightly different from other 3B2 computer devices. The following list gives the naming convention for SCSI devices.

SCSI Cartridge Tape Drives

The device file names for the SCSI Cartridge Tape are:

/dev/rmt/c?t*d#s0(n) for 60 MB
/dev/rmt/c?t*d#s1(n) for 120 MB

where:

? = Slot number for Host Adapter
***** = Target controller SCSI ID
# = SCSI Cartridge Tape Drive number
n = No rewind—optional.

SCSI Hard Disk Drives

The device file names for the SCSI hard disks are:

/dev/dsk/c?t*d#s& or **/dev/rdsk/c?t*d#s&**

where:

? = Slot number for Host Adapter
***** = Target controller SCSI ID
# = Hard disk Logical Unit ID
& = Partition number.

SCSI Hard Disk Default Partitions

Figure A-1 defines the use, size, and number of information nodes (I-nodes) for the defined partitions of a dual 155-megabyte SCSI (with ESDI interface) hard disk system. The rotational gap (Gap) and blocks per cylinder (Cyl) are also defined. The default Volume Table Of Content (VTOC) partitioning is used in this chart.

Configuration	Partition	Use	Sector Start	Size*	I-Nodes	FS Type
Wren III	c1t1d0s0	root	20790	25830	3216	2Kb
155-Megabyte	c1t1d0s1	swap	150	20640	—	—
(ESDI)	c1t1d0s3	sysdump	269632	32768	—	—
Hard Disk 1	c1t1d0s4	install	268065	1567	200	2Kb
(Gap=12)	c1t1d0s6	entire disk	0	302400	—	—
(Cyl=315)	c1t1d0s7	boot	0	150	—	—
	c1t1d0s8	usr2	46620	221445	27680	1Kb
Hard Disk 2	c1t1d1s2	usr	315	302085	37760	2Kb
	c1t1d1s6	entire disk	0	302400	—	—
	c1t1d1s7	boot	0	315	—	—

*Size is in 512-byte blocks

Figure A-1: 155-Megabyte (ESDI) Dual Hard Disk Default Partitioning

Figure A-2 defines the use, size, and number of information nodes (I-nodes) for the defined partitions of a single 155-megabyte SCSI (with ESDI interface) hard disk system.

Configuration	Partition	Use	Sector Start	Size*	I-Nodes
Wren III	c1t1d0s0	root	20790	25830	3216
155-Megabyte	c1t1d0s1	swap	150	20640	—
(ESDI)	c1t1d0s2	usr	46620	221445	27680
Hard Disk	c1t1d0s3	sysdump	269632	32768	—
(Gap=12)	c1t1d0s4	install	268065	1567	200
(Cyl=315)	c1t1d0s6	entire disk	0	302400	—
	c1t1d0s7	boot	0	150	—

*Size is in 512-byte blocks

Figure A-2: 155-Megabyte (ESDI) Single Hard Disk Default Partitioning

Figure A-3 defines the use, size, and number of information nodes (I-nodes) for the defined partitions of a dual 317-megabyte SCSI (with ESDI interface) hard disk system. The rotational gap (Gap) and blocks per cylinder (Cyl) are also defined. The default Volume Table Of Content (VTOC) partitioning is used in this chart.

Configuration	Partition	Use	Sector Start	Size*	I-Nodes	FS Type
317-Megabyte	c1t1d0s0	root	42330	52530	3264	2Kb
(ESDI) & SCSI	c1t1d0s1	swap	150	42180	—	—
Hard Disk 1	c1t1d0s3	sysdump	586882	32768	—	—
(Gap=12)	c1t1d0s4	install	584970	1912	96	2Kb
(Cyl=510)	c1t1d0s6	entire disk	0	619650	—	—
	c1t1d0s7	boot	0	150	—	—
	c1t1d0s8	usr2	94860	245310	30656	1Kb
	c1t1d0s9	usr2	340170	244800	30592	1Kb
Hard Disk 2	c1t1d1s2	usr	510	245310	15328	2Kb
	c1t1d1s6	entire disk	0	619650	—	—
	c1t1d1s7	boot	0	510	—	—
	c1t1d0s8	usr4	245820	245310	30656	1Kb
	c1t1d0s9	usr5	491130	128520	16064	1Kb

*Size is in 512-byte blocks

Figure A-3: 317-Megabyte (ESDI) Dual Hard Disk Default Partitioning

Figure A4 defines the use, size, and number of information nodes (I-nodes) for the defined partitions of an embedded SCSI using two 322-MB disk drives.

Configuration	Partition	Use	Sector Start	Size*	I-Nodes	FS Type
Hard Disk 1	c1t1d0s1	root	43008	53376	3328	2Kb
	c1t1d0s1	swap	150	42858	—	—
	c1t1d0s3	sysdump	596992	32768	—	—
	c1t1d0s4	install	595200	1792	96	2Kb
	c1t1d0s6	entire disk	0	629760	—	—
	c1t1d0s7	boot	0	150	—	—
	c1t1d0s8	usr2	96384	245760	30720	1Kb
	c1t1d0s9	usr3	342144	245760	30720	1Kb
	c1t1d0sa	usr4	587904	7296	912	1Kb
Hard Disk 2	c1t3d0s2	usr	384	245760	15260	2Kb
	c1t3d0s6	entire disk	0	629760	—	—
	c1t3d0s7	boot	0	384	—	—
	c1t3d0s8	usr5	246144	245760	30720	1Kb
	c1t3d0s9	usr6	491904	137856	17232	1Kb

*Size is in 512-byte blocks

Figure A-4: 322-Megabyte Dual Hard Disk Default Partitioning

Additional Hard Disk Partitions

Chapter 4, "Disk/Tape Management," shows how to partition a SCSI hard disk using System Administration menus. Although System Administration menus make partitioning easy to do, you can only allocate eight (user-defined) partitions (8 through 15) per disk. If necessary, you can define more partitions per disks by working outside the System Administration menus.

The 3B2 computer limit is 16 partitions per hard disk. Depending on the disk application, some of these partitions are already defined and should not be used. For example, Figure A-1 shows that partition 6 (s6) is defined as the entire disk and partition 7 (s7) is the boot block. However, the other 14 SCSI disk partitions are normally undefined.

The following list shows the operations that have to be done to repartition a disk outside System Administration menus.

- Using **prtvtoc**, get a copy of a VTOC for a SCSI device.
- Change the VTOC to include desired number of partitions.

  Note: You must be prepared to input the sector count information. Remember, partitions must start on cylinder boundaries.

- Use the **fmthard** command with **-s** and **-m** options to install the new VTOC and partition the disk.
- Use the **labelit** to change or add new file system names.
- Change **/etc/fstab** to change or add new file systems to be mounted.
- Use **mkdir** to add new mount points under the *root (/)* directory.

Floppy Disk Partitions

The following table defines the floppy disk partitions in terms of use, starting sector, and total number of sectors for the various controller (c), drive (d), and section (s) identifiers for the floppy disk. Note the following:

- The raw and block device partitions for the entire floppy disk (partition 6) are linked to **/dev/rSA/diskette1** and **/dev/SA/diskette1**, respectively. The use of these names when specifying the entire floppy disk is preferred over the use of the controller, drive, and section identifiers to avoid accidentally writing to a different device or partition.
- These identifiers are applicable to both the raw and block devices.
- The Volume Table Of Contents (VTOC) partitioning is not applicable to the floppy disk drive.

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	1(1*18)	1404
c0d0s6	entire disk	0	1422
c0d0s7	boot	0	18

*Sectors are equivalent to 512-byte blocks

Figure A-5: Floppy Disk Drive

Appendix B: Directories and Files

Introduction

Appendix B describes directories and files of interest to a System Administrator.

Directories

The directories of the **root** file system (/) are as follows:

bck	Directory used to mount a backup file system for restoring files.
bin	Directory containing public commands.
boot	Directory containing configurable object files created by the **/etc/mkboot**(1M) program.
dev	Directory containing special files that define all the devices on the system.
dgn	Directory containing diagnostic programs.
edt	Directory that contains equipped device table data.
etc	Directory containing administrative programs and tables.
install	Directory used by System Administration to mount utilities packages for installation and removal (**/install** file system).
lib	Directory containing public libraries.
lost+found	Directory used by **fsck**(1M) to save disconnected files.
mnt	Directory used to temporarily mount file systems during restoration of the operating system from floppy disks.
save	Directory used by Simple Administration for saving data on floppies.
shlib	Directory containing shared libraries.
tmp	Directory used for temporary files.
usr	Directory used to mount the **/usr** file system.

Files

The following files and directories are important in the administration of the 3B2 computer:

- /etc/checklist
- /etc/d_passwd
- /etc/dialups
- /etc/fstab
- /etc/gettydefs
- /etc/group
- /etc/init.d Directory
- /etc/inittab
- /etc/master.d Directory
- /etc/motd
- /etc/passwd
- /etc/profile
- /etc/rc0
- /etc/rc0.d Directory
- /etc/rc2
- /etc/rc2.d Directory
- /etc/rc.d Directory
- /etc/rc3
- /etc/rc3.d Directory
- /etc/save.d Directory
- /etc/shadow
- /etc/shutdown

- **/etc/shutdown.d Directory**
- **/etc/TIMEZONE**
- **/etc/utmp**
- **/etc/wtmp**
- **/usr/adm/conlog**
- **/usr/adm/errlog**
- **/usr/adm/loginlog**
- **/usr/adm/sulog**
- **/usr/lib/cron/log**
- **/usr/lib/help/HELPLOG**
- **/usr/lib/spell/spellhist**
- **/usr/news Directory**
- **/usr/options Directory**
- **/usr/spool/cron/crontabs Directory**

Each of these files is briefly described in this appendix.

/etc/checklist

The **/etc/checklist** file is used to define a default list of file system devices to be checked for consistency by **/etc/fsck** and **/etc/ncheck**. The character (raw) device partition for the file system should be identified. The devices listed normally correspond to those mounted when the system is in the multiuser mode (run level 2). The **root** file system (**/dev/rdsk/c1t1d0s0**) should not be listed in this file. Remember that a file system must be unmounted to be checked, except **root**. Therefore, the **checklist** file is a convenience for use when in the single-user mode of operation with only the **root** file system mounted. When the system is delivered, this file is empty. For a single-disk system, the list is understandably brief. A typical 3B2 computer **/etc/checklist** file is shown in Figure B-1. [See the **checklist**(4) manual page in the *User's and System Administrator's Reference Manual* for additional information.]

```
# cat /etc/checklist
/dev/rdsk/c1t1d0s8 /usr2
/dev/rdsk/c1t1d1s2 /usr
#
```

Figure B-1: Typical **/etc/checklist** File

/etc/d_passwd

The **/etc/d_passwd** file identifies the shells and their associated encrypted passwords. A sample 3B2 computer **/etc/d_passwd** file is shown in Figure B-2. This sample **/etc/d_passwd** file shows a password assigned to **/bin/sh** and no password assigned to **/usr/lib/uucp/uucico** login shell processes. Refer to Chapter 1, "System Identification and Security," for information on dial-up passwords.

```
# cat /etc/d_passwd
/bin/sh:yzPD4VPGeJk4U:
/usr/lib/uucp/uucico::
#
```

Figure B-2: Sample **/etc/d_passwd** File

/etc/dialups

The **/etc/dialups** files contains a list of the ports that require additional security. Comments can be added to entries in the file using a pound sign (#). All characters on a line following a pound sign are ignored. A sample 3B2 computer **/etc/dialups** file is shown in Figure B-3. This sample **/etc/dialups** file specifies the **contty** and **tty21** ports as dial-up ports. A dial-up password is NOT applied to port **/dev/tty31** because of the pound sign. Refer to Chapter 1, "System Identification and Security," for information on dial-up passwords.

```
# cat /etc/dialups
/dev/contty      #COMMENT
/dev/tty21       #COMMENT
#/dev/tty31
#
```

Figure B-3: Sample **/etc/dialups** File

/etc/fstab

The **/etc/fstab** file is used as an argument to the **/etc/mountall** command. The **fstab** file specifies the file system(s) to be mounted by **/etc/mountall**. A typical **/etc/fstab** file for a dual-disk 3B2 computer is shown in Figure B-4. The format of the file is the block device name followed by the mount point name. [See the **mountall**(1M) manual page in the *User's and System Administrator's Reference Manual* for additional information.]

```
# cat /etc/fstab
/dev/dsk/c1t1d0s8 /usr2
/dev/dsk/c1t1d1s2 /usr
#
```

Figure B-4: Typical **/etc/fstab** File

/etc/gettydefs

The **/etc/gettydefs** file contains information that is used by **/etc/getty** to set the speed and terminal settings for a line. The **getty** command accesses the **gettydefs** file with a label. The general format of the **gettydefs** file is as follows:

label# initial-flags # final-flags #login-prompt #next-label

Each line entry in the **gettydefs** file is followed by a blank line. [Refer to the **gettydefs**(4) manual page in the *User's and System Administrator's Reference Manual* for complete information.] Figure B-5 shows a typical 3B2 computer **/etc/gettydefs** file.

```
# cat /etc/gettydefs
38400# B38400 HUPCL # B38400 SANE IXANY TAB3 HUPCL #login: #19200
19200# B19200 HUPCL # B19200 SANE IXANY TAB3 HUPCL #login: #9600
9600# B9600 HUPCL # B9600 SANE IXANY TAB3 HUPCL #login: #4800
4800# B4800 HUPCL # B4800 SANE IXANY TAB3 HUPCL #login: #2400
2400# B2400 HUPCL # B2400 SANE IXANY TAB3 HUPCL #login: #1200
1200# B1200 HUPCL # B1200 SANE IXANY TAB3 HUPCL #login: #300
300# B300 HUPCL # B300 SANE IXANY TAB3 HUPCL #login: #38400
console# HUPCL OPOST ONLCR # SANE IXANY TAB3 #Console Login: #console1
console1# B9600 HUPCL OPOST ONLCR # B9600 SANE IXANY TAB3 #Console Login: #console2
console2# B1200 HUPCL OPOST ONLCR # B1200 SANE IXANY TAB3 #Console Login: #console3
console3# B300 HUPCL OPOST ONLCR # B300 SANE IXANY TAB3 #Console Login: #console4
console4# B2400 HUPCL OPOST ONLCR # B2400 SANE IXANY TAB3 #Console Login: #console5
console5# B4800 HUPCL OPOST ONLCR # B4800 SANE IXANY TAB3 #Console Login: #console6
console6# B19200 HUPCL OPOST ONLCR # B19200 SANE IXANY TAB3 #Console Login: #console
contty# B9600 HUPCL OPOST ONLCR # B9600 SANE IXANY TAB3 #login: #contty1
contty1# B1200 HUPCL OPOST ONLCR # B1200 SANE IXANY TAB3 #login: #contty2
contty2# B300 HUPCL OPOST ONLCR # B300 SANE IXANY TAB3 #login: #contty3
contty3# B2400 HUPCL OPOST ONLCR # B2400 SANE IXANY TAB3 #login: #contty4
contty4# B4800 HUPCL OPOST ONLCR # B4800 SANE IXANY TAB3 #login: #contty5
contty5# B19200 HUPCL OPOST ONLCR # B19200 SANE IXANY TAB3 #login: #contty
pty# B9600 HUPCL OPOST ONLCR # B9600 SANE IXANY TAB3 #PC login: #pty
4800H# B4800 # B4800 SANE IXANY TAB3 HUPCL #login: #9600H
9600H# B9600 # B9600 SANE IXANY TAB3 HUPCL #login: #19200H
19200H# B19200 # B19200 SANE IXANY TAB3 HUPCL #login: #38400H
38400H# B38400 # B38400 SANE IXANY TAB3 HUPCL #login: #2400H
2400H# B2400 # B2400 SANE IXANY TAB3 HUPCL #login: #1200H
1200H# B1200 # B1200 SANE IXANY TAB3 HUPCL #login: #300H
300H# B300 # B300 SANE IXANY TAB3 HUPCL #login: #4800H
conttyH# B9600 OPOST ONLCR # B9600 HUPCL SANE IXANY TAB3 #login: #contty1H
contty1H# B1200 OPOST ONLCR # B1200 HUPCL SANE IXANY TAB3 #login: #contty2H
contty2H# B300 OPOST ONLCR # B300 HUPCL SANE IXANY TAB3 #login: #contty3H
contty3H# B2400 OPOST ONLCR # B2400 HUPCL SANE IXANY TAB3 #login: #contty4H
contty4H# B4800 OPOST ONLCR # B4800 HUPCL SANE IXANY TAB3 #login: #contty5H
contty5H# B19200 OPOST ONLCR # B19200 HUPCL SANE IXANY TAB3 #login: #conttyH
#
```

Figure B-5: Typical **gettydefs** File

/etc/group

The **/etc/group** file describes each group to the system. An entry is added for each new group. Each entry in the file is one line that consists of four fields separated by a colon (:), such as the following:

group name:password:group id:login names

Explanations for these fields are as follows:

group name — The first field defines the group name. The group name is from three to six characters long. The first character is alphabetic. The rest of the characters are alphanumeric. No uppercase characters appear.

password — The second field contains the encrypted group password. The encrypted group password contains 13 bytes (characters). The password is limited to a maximum of 8 bytes. The encrypted password can be followed by a comma and up to 4 more bytes of password aging information. The use of group passwords is discouraged.

group id — The third field contains the group identification number, which must be between 0 and 60,000. Group identification numbers 0 through 99 are reserved; 0 identifies the super-user (root). Commas are not entered in this field.

login names — The fourth field contains a list of all login names in the group. Names in the list are separated by commas. The names listed may use the **/etc/newgrp** command to become a member of the group.

Figure B-6 shows a typical 3B2 computer **/etc/group** file.

```
# cat /etc/group
root::0:root
other::1:
bin::2:root,bin,daemon
sys::3:root,bin,sys,adm
adm::4:root,adm,daemon
mail::6:root
rje::8:rje,shqer
daemon::12:root,daemon
#
```

Figure B-6: Typical **/etc/group** File

/etc/init.d Directory

The **/etc/init.d** directory contains executable files used in upward and downward transitions to all system run levels. These files are linked to files beginning with **S** (start) or **K** (stop) in **/etc/rc*n*.d**, where *n* is the appropriate run level. Files are not executed from this directory. They are only executed from **/etc/rc*n*.d** directories.

/etc/inittab

The **/etc/inittab** file contains instructions for the **/etc/init** command. The instructions define the processes that are to be created or terminated for each initialization state. Initialization states are called run levels or run-states. By convention, run level 1 (or S or s) is single-user mode; run levels 2 and 3 are multiuser modes. Chapter 3, "Processor Operations," summarizes the various run levels and describes their uses. [See the **inittab**(4) manual page in the *User's and System Administrator's Reference Manual* for additional information.] Figure B-7 shows a typical 3B2 computer **/etc/inittab** file. The typical entry is a series of fields separated by a colon (:), such as the following:

identification:run-state:action:process

Explanations for these fields are as follows:

identification — The identification field is a one- or two-character identifier for the line entry. The identifier is unique for a line.

run-state — The run-state defines the run level in which the entry is to be processed.

action — The action field defines how **/etc/init** treats the process field. [Refer to the **inittab**(4) manual page in the *User's and System Administrator's Reference Manual* for complete information.]

process — The process field defines the shell command that is to be executed.

```
zu::sysinit:/etc/bzapunix </dev/console >/dev/console 2>&1
fs::sysinit:/etc/bcheckrc </dev/console >/dev/console 2>&1
sd::sysinit:sh -c /etc/rc2.d/S00scsi </dev/console >/dev/console 2>&1
xdc::sysinit:sh -c 'if [ -x /etc/rc.d/0xdc ] ; then /etc/rc.d/0xdc ; fi' >/dev/console 2>&1
mt:23:bootwait:/etc/brc </dev/console >/dev/console 2>&1
pt:23:bootwait:/etc/ports </dev/console >/dev/console 2>&1
is:2:initdefault:
pl:s1234:powerfail:/etc/led -f                  # start green LED flashing
p3:s1234:powerfail:uadmin 2 0
```

Figure B-7: Typical **/etc/inittab** File (Sheet 1 of 2)

```
fl:056:wait:/etc/led -f                          # start green LED flashing
s0:056:wait:/etc/rc0 >/dev/console 2>&1 </dev/console
s1:1:wait:/etc/shutdown -y -iS -g0 >/dev/console 2>&1 </dev/console
s2:23:wait:/etc/rc2 >/dev/console 2>&1 </dev/console
s3:3:wait:/etc/rc3 >/dev/console 2>&1 </dev/console
OF:0:wait:echo "\nPlease flip the power switch to the STANDBY position." >/dev/console 2>&1
of:0:wait:/etc/uadmin 2 0 >/dev/console 2>&1 </dev/console
un:56:wait:/etc/init.d/unlock > /dev/console 2>&1 < /dev/console
fw:5:wait:/etc/uadmin 2 2 >/dev/console 2>&1 </dev/console
RB:6:wait:echo "\nThe system is being restarted." >/dev/console 2>&1
rb:6:wait:/etc/uadmin 2 1 >/dev/console 2>&1 </dev/console
co:234:respawn:/etc/getty console console
ct:234:respawn:/etc/getty contty contty
he:234:respawn:sh -c 'sleep 20 ; exec /etc/hdelogger >/dev/console 2>&1'
mi::sysinit:/etc/init.d/mirdisk </dev/console >/dev/console 2>&1
lo::sysinit:/etc/init.d/lock < /dev/console > /dev/console 2>&1
21:234:respawn:/etc/getty tty21 9600 #rar
22:234:off:/etc/getty tty22 9600 #38400 baud rate is available
23:234:off:/etc/getty tty23 9600 #38400 baud rate is available
24:234:off:/etc/getty tty24 9600 #38400 baud rate is available
25:234:off:/etc/getty tty25 9600 #38400 baud rate is available
26:234:off:/etc/getty tty26 9600 #38400 baud rate is available
27:234:off:/etc/getty tty27 9600 #38400 baud rate is available
28:234:off:/etc/getty tty28 9600 #38400 baud rate is available
31:234:respawn:/usr/lib/uucp/uugetty -r -t 60 tty31 9600 #wr3b2a Direct Link
32:234:off:/etc/getty tty32 9600 #38400 baud rate is available
33:234:off:/etc/getty tty33 9600 #38400 baud rate is available
34:234:off:/etc/getty tty34 9600 #38400 baud rate is available
35:234:off:/etc/getty tty35 9600 #38400 baud rate is available
36:234:off:/etc/getty tty36 9600 #38400 baud rate is available
37:234:off:/etc/getty tty37 9600 #38400 baud rate is available
38:234:off:/etc/getty tty38 9600 #38400 baud rate is available
41:234:respawn:/etc/getty tty41 9600 #cms
42:234:off:/etc/getty tty42 9600 #38400 baud rate is available
43:234:off:/etc/getty tty43 9600 #38400 baud rate is available
44:234:off:/etc/getty tty44 9600 #38400 baud rate is available
45:234:off:/etc/getty tty45 9600 #38400 baud rate is available
46:234:off:/etc/getty tty46 9600 #38400 baud rate is available
47:234:off:/etc/getty tty47 9600 #38400 baud rate is available
48:234:off:/etc/getty tty48 9600 #38400 baud rate is available
```

Figure B-7: Typical **/etc/inittab** File (Sheet 2 of 2)

/etc/master.d Directory

The **/etc/master.d** directory contains files that define the configuration of hardware devices, software drivers, system parameters, and aliases. The files are used by **/etc/mkboot** to obtain device information for the generation of device driver and configurable module files. The **/etc/sysdef**(1M) 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. [Refer to the **/etc/master**(4) manual page in the *User's and System Administrator's Reference Manual* for additional information.]

/etc/motd

The **/etc/motd** file contains the message-of-the-day. The message-of-the-day is output by instructions in the **/etc/profile** file after a successful login. This message should be kept short and to the point. The **/usr/news** file(s) should be used for lengthy, more explicit messages.

/etc/passwd

The **/etc/passwd** file identifies each user to the system. An entry is added for each new user. Each entry in the file is one line and consists of seven fields. The fields are separated by a colon (:), such as the following:

login name:passwd:user:group:account:login directory:program

Explanations for these fields are as follows:

login name — The first field defines the login name. The login name is from three to six characters long. The first character is alphabetic. The rest of the characters are alphanumeric. No uppercase characters appear.

passwd — The second field contains the encrypted login password. The encrypted login password contains 13 bytes (characters). The password is limited to a maximum of 8 bytes. The encrypted password can be followed by a comma and up to 4 more bytes of password aging information. This field contains a lowercase letter 'x' when the shadow password feature is enabled. The **/etc/shadow** file contains the encrypted password and password aging information when shadow password is enabled.

user id	The third field contains the user identification number, which must be between 0 and 60,000. Group identification numbers 0 through 99 are reserved; 0 identifies the super-user (root). Commas are not entered in this field.
group id	The fourth field contains the group identification number, which must be between 0 and 60,000. Group identification numbers 0 through 99 are reserved; 0 identifies the super-user (root). Commas are not entered in this field.
account	The fifth field is used by accounting programs. This field typically contains the user name, department number, and bin number.
login directory	The sixth field defines the full path name of the login directory.
program	The seventh field defines the program to be executed after login. If it is null, the shell (**/bin/sh**) is invoked.

Figure B-8 shows a typical **/etc/passwd** file with shadow password enabled. [See the **passwd**(4) manual page in the *User's and System Administrator's Reference Manual* for additional information.]

```
# cat /etc/passwd
root:x:0:1:0000-Admin(0000):/:
daemon:x:1:1:0000-Admin(0000):/:
bin:x:2:2:0000-Admin(0000):/bin:
sys:x:3:3:0000-Admin(0000):/usr/src:
adm:x:4:4:0000-Admin(0000):/usr/adm:
uucp:x:5:5:0000-uucp(0000):/usr/lib/uucp:
nuucp:x:10:10:0000-uucp(0000):/usr/spool/uucppublic:/usr/lib/uucp/uucico
trouble:x:70:1:trouble(0000):/usr/lib/trouble:
lp:x:71:2:0000-lp(0000):/usr/spool/lp:
setup:x:0:0:general system administration:/usr/admin:/usr/bin/setup
powerdown:x:0:0:general system administration:/usr/admin:/usr/bin/powerdown
sysadm:x:0:0:general system administration:/usr/admin:/usr/bin/sysadm
checkfsys:x:0:0:check diskette file system:/usr/admin:/usr/bin/checkfsys
makefsys:x:0:0:make diskette file system:/usr/admin:/usr/bin/makefsys
mountfsys:x:0:0:mount diskette file system:/usr/admin:/usr/bin/mountfsys
umountfsys:x:0:0:unmount diskette file system:/usr/admin:/usr/bin/umountfsys
vmsys:x:100:100:FACE:/usr/vmsys:
oasys:x:101:1:Object Architecture Files:/usr/oasys:
rar:x:102:1:R. A. Riedel:/usr/rar:/bin/ksh
cms:x:103:1:C. M. Snyder:/usr/cms:
#
```

Figure B-8: Typical **/etc/passwd** File

/etc/profile

The default profile for all users is in the **/etc/profile** file. 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 following can be added to the **/etc/profile** for the **root** login to cause the erase character to back up and to set the TERM variable.

```
if [ ${LOGNAME} = root ]
        then
                stty echoe
                echo "Enter TERM: \c"
                read TERM
                export TERM
fi
```

Figure B-9 shows the 3B2 computer default profile.

```
# cat /etc/profile
# 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
        ;;
-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
        if [ ${LOGNAME} != root ]
        then
                news -n
        fi
        ;;
esac
umask 022
trap  2 3
#
```

Figure B-9: Standard **/etc/profile** File

/etc/rc0

The **/etc/rc0** file contains a shell script that is executed by **/etc/shutdown** for transitions to single-user state and by **/etc/init** on transitions to run levels 0, 5, and 6. Files in the **/etc/shutdown.d** and **/etc/rc0.d** directories are executed when **/etc/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 run levels 0, 5, or 6 can be done by adding a file to the **/etc/shutdown.d** directory. Figure B-10 shows a typical 3B2 computer **/etc/rc0** file.

```
# "Run Commands" for init state 0
# Leaves the system in a state where it is safe to turn off the power
# or go to firmware.
stty sane tab3 2>/dev/null
echo 'The system is coming down.  Please wait.'
if [ -d /etc/shutdown.d ]
then
        for f in /etc/shutdown.d/*
        { if [ -s $f ]
                then
                        /bin/sh ${f}
                fi
        }
fi
#       End of historical section
if [ -d /etc/rc0.d ]
then
        for f in /etc/rc0.d/K*
        {
                if [ -s ${f} ]
                then
                        /bin/sh ${f} stop
                fi
        }
```

Figure B-10: Typical **/etc/rc0** File (Sheet 1 of 2)

```
# system cleanup functions ONLY (things that end fast!)
for f in /etc/rc0.d/S*
        {
                if [ -s ${f} ]
                then
                        /bin/sh ${f} start
                fi
        }
fi
trap "" 15
kill -15 -1
sleep 10
/etc/killall  9
sleep 10
sync;sync;sync
/etc/umountall
stty sane 2>/dev/null
sync;  sync
echo '
The system is down.'
sync
```

Figure B-10: Typical **/etc/rc0** File (Sheet 2 of 2)

/etc/rc0.d Directory

The **/etc/rc0.d** directory contains files executed by **/etc/rc0** for transitions to system run levels 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**. The **K** identifies processes that are stopped, and **S** identifies processes that are started when entering run levels 0, 5, or 6.

/etc/rc2

The **/etc/rc2** file contains a shell script that is executed by **/etc/init** on transitions to run level 2 (multiuser state). Executable files in the **/etc/rc.d** and any executable files beginning with an **S** or a **K** in **/etc/rc2.d** directories are executed when **/etc/rc2** is run. All files in **rc2.d** are linked from files in the **/etc/init.d** directory. The following are descriptions of some of those files. These files are prefixed with an **S** or a **K** and a number in the **/etc/rc2.d** directory.

MOUNTFSYS	Sets up and mounts file systems. Builds the mount table and mounts the **root** (/) and user (**/usr**) file systems. Makes the **/usr/tmp** directory, cleaning up (deleting) any previous files in that directory.
autoconfig	Makes a **/unix** if self-configuration occurred during the boot sequence. The new in-memory operating system is copied to **/unix**.
cron	Starts the **cron** daemon by executing **/etc/cron**.
sysetup	Removes the **/etc/ps_data** to force the **/bin/ps** command to read the **/unix** file. Outputs the system configuration if the **/etc/prtconf** command exists. Outputs the system trademark information.
uucp	When basic networking is added to the system, the **uucp** file is added to the directory. The **uucp** file deletes **uucp** locks (LCK*), status files (STST*), and temporary files (TM*) under the **/usr/spool/uucp** directory structure.

Other files may also be added to **/etc/rc2.d** and **/etc/rc.d** directories as a function of adding hardware or software to the system. Figure B-11 shows a typical 3B2 computer **/etc/rc2** file.

```
# "Run Commands" executed when the system is changing to init state 2,
# traditionally called "multi-user."
. /etc/TIMEZONE
# Pickup start-up packages for mounts, daemons, services, etc.
set 'who -r'
if [ $9 = "S" ]
then
   echo 'The system is coming up.  Please wait.'
   BOOT=yes
   if [ -f /etc/rc.d/PRESERVE ] # historical segment for vi and ex
        then
                mv /etc/rc.d/PRESERVE /etc/init.d
                ln /etc/init.d/PRESERVE /etc/rc2.d/S02PRESERVE
        fi
elif [ $7 = "2" ]
then
        echo 'Changing to state 2.'
        if [ -d /etc/rc2.d ]
        then
                for f in /etc/rc2.d/K*
                {
                        if [ -s ${f} ]
                        then
                                /bin/sh ${f} stop
                        fi
                }
        fi
fi
```

Figure B-11: Typical **/etc/rc2** File (Sheet 1 of 2)

```
if [ -d /etc/rc2.d ]
then
        for f in /etc/rc2.d/S*
        {
                if [ -s ${f} ]
                then
                        /bin/sh ${f} start
                fi
        }
fi
if [ "${BOOT" = "yes" ]
then
        stty sane tab3 2>/dev/null
fi
if [ "${BOOT" = "yes" -a -d /etc/rc.d ]
then
        for f in `ls /etc/rc.d`
        {
                if [ ! -s /etc/init.d/${f} ]
                then
                        /bin/sh /etc/rc.d/${f}
                fi
        }
fi
if [ "${BOOT" = "yes" -a $7 = "2" ]
then
        echo 'The system is ready.'
elif [ $7 = "2" ]
then
        echo 'Change to state 2 has been completed.'
fi
```

Figure B-11: Typical **/etc/rc2** File (Sheet 2 of 2)

/etc/rc2.d Directory

The **/etc/rc2.d** directory contains files executed by **/etc/rc2** for transitions to system run level 3. Files in this directory are linked from the **/etc/init.d** directory and begin with either a **K** or an **S**. The **K** identifies processes that should be stopped, and an **S** identifies processes that should be started when entering run levels 2 or 3.

/etc/rc.d Directory

The **/etc/rc.d** directory contains executable files that do the various functions needed to initialize the system to run level 2. The files are executed when **/etc/rc2** is run. (Files contained in this directory before UNIX System V Release 3.0 were moved to **/etc/rc2.d**. This directory is only maintained for compatibility reasons.)

/etc/rc3

The **/etc/rc3** file is executed by **/etc/init**. It executes the shell scripts in **/etc/rc3.d** on transitions to system run level 3 (the Remote File Sharing state).

/etc/rc3.d Directory

The **/etc/rc3.d** directory contains files executed by **/etc/rc3** for transitions to system run level 3 (multiuser mode). Files in this directory are linked from the **/etc/init.d** directory and begin with either a **K** or an **S**. The **K** identifies processes that should be stopped, and an **S** identifies processes that should be started when entering run level 3.

/etc/save.d Directory

The **/etc/save.d** directory contains files that are used by the System Administration Menu commands associated with backing up data on floppy disks. The following files are included:

except	A list of the directories and files that should not be copied as part of a backup (**savefiles**) 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/timestamp** directory.

/etc/shadow

The **/etc/shadow** file contains the encrypted password and password aging information for the corresponding logins in the **/etc/passwd** file when the shadow password feature is enabled. The **/etc/passwd** file identifies each user to the system. An entry is added for each new user. The **/etc/shadow** tracks the **/etc/passwd** file. Each entry in the **/etc/shadow** file is one line and consists of five fields. The fields are separated by a colon (:), such as the following:

login_name:password:last_changed:minimum:maximum

Explanations for these fields are as follows:

login_name	The first field defines the login name. The login name is from three to six characters long. The first character is alphabetic. The rest of the characters are alphanumeric. No uppercase characters appear.
password	The second field contains the encrypted login password, no password, or a lock string. The encrypted login password contains 13 bytes (characters). The password is limited to a maximum of 8 bytes. No password is an empty field. Any character string not recognized as an encrypted password is a lock string.
last_changed	This field contains the date when the password was last modified. The date is the number of days counted from January 1, 1970. A date of zero means that the user has yet to log in and change the password for which password aging has been applied for the first time.
minimum	The number of days required between password changes.
maximum	The number of days the password is valid. This is also the number of days between the password expiration date and the *last_changed* field.

Figure B-12 shows a typical **/etc/shadow** file. [See the **passwd**(4) manual page in the *User's and System Administrator's Reference Manual* for additional information.]

```
# cat /etc/shadow
root:XXKioMT3NoQUQ:6661::
daemon:NP:6661::
bin:NP:6661::
sys:NP:6661::
adm:NP:6661::
uucp:NP:6661::
nuucp:v7.iVPYPQN4T.:6661::
trouble:NP:6661::
lp:NP:6661::
setup:63vhln04y/g8E:6661::
powerdown:NP:6661::
sysadm:2Gibb3ON0knzQ:6669::
checkfsys:NP:6661::
makefsys:NP:6661::
mountfsys:NP:6661::
umountfsys:NP:6661::
vmsys:8Y2iilr.87Q0Q:6664:7:63
oasys::0:7:63
rar:CfTW9v4I65vHY:6666:14:35
cms::6661::
#
```

Figure B-12: Typical **/etc/shadow** File

/etc/shutdown

The **/etc/shutdown** file contains a shell script to shut down the system gracefully in preparation for system backup or scheduled downtime. After stopping all nonessential processes, the **shutdown** script executes files in the **/etc/shutdown.d** directory by calling **/etc/rc0** for transitions to run level s or S. For transitions to other run levels, the **shutdown** script calls **/etc/init**. Figure B-13 shows a typical 3B2 computer **/etc/shutdown** file.

```
# Sequence performed to change the init state of a machine.
# This procedure checks to see if you are permitted and allows an
# interactive shutdown.  The actual change of state, killing of
# processes and such are performed by the new init state, say 0,
# and its /etc/rc0.
# Usage:  shutdown [ -y ] [ -g<grace-period> ] [ -i<init-state> ]
#!      chmod +x ${file}
if [ `pwd` != / ]
then
echo "$0:  You must be in the / directory to run /etc/shutdown."
exit 1
fi
#       Check the user id.
if [ -x /usr/bin/id ]
then
        eval `id  |  sed 's/[^a-z0-9=].*//'`
        if [ "${uid:=0}" -ne 0 ]
        then
                echo "$0:  Only root can run /etc/shutdown."
                exit 2
        fi
fi
grace=60
askconfirmation=yes
initstate=s
while [ $# -gt 0 ]
do
        case $1 in
        -g[0-9]* )
                grace=`expr "$1" : '-g)'`
                ;;
        -i[Ss0156] )
```

Figure B-13: Typical **/etc/shutdown** File (Sheet 1 of 4)

```
        initstate=`expr "$1" : '-i)'`
        ;;
    -i[234] )
        initstate=`expr "$1" : '-i)'`
        echo "$0:  Initstate $i is not for system shutdown"
        exit 1
        ;;
    -y )
        askconfirmation=
        ;;
    -* )
        echo "Illegal flag argument '$1'"
        exit 1
        ;;
    * )
        echo "Usage:  $0 [ -y ] [ -g<grace> ] [ -i<initstate> ]"
        exit 1
        esac
        shift
done
if [ -n "${askconfirmation}" -a -x /etc/ckbupscd ]
        then
#       Check to see if backups are scheduled at this time
        BUPS=`/etc/ckbupscd`
        if [ "$BUPS" != "" ]
        then
        echo "$BUPS"
        echo "Do you wish to abort this shutdown and return to
command level to do these backups? [y, n]
```

Figure B-13: Typical **/etc/shutdown** File (Sheet 2 of 4)

```
                read YORN
                if [ "$YORN" = "y" -o "$YORN" = "Y" ]
                        then
                        exit 1
                fi
        fi
fi
if [ -z "${TZ}"  -a  -r /etc/TIMEZONE ]
then
   . /etc/TIMEZONE
fi
echo '
Shutdown started.    date
echo
sync
cd /
trap "exit 1"  1 2 15
a="`who  |  wc -l`"
if [ ${a} -gt 1  -a  ${grace} -gt 0 ]
then
su adm -c /etc/wall<<-!
        The system will be shut down in ${grace} seconds.
        Please log off now.
!
        sleep ${grace}
fi
/etc/wall <<-!
        THE SYSTEM IS BEING SHUT DOWN NOW ! ! !
        Log off now or risk your files being damaged.
!
sleep ${grace}
```

Figure B-13: Typical **/etc/shutdown** File (Sheet 3 of 4)

```
if [ ${askconfirmation} ]
then
        echo "Do you want to continue? (y or n):   \c"
        read b
else
        b=y
fi
if [ "$b" != "y" ]
then
        /etc/wall <<-\!
        False Alarm:  The system will not be brought down.
        !
        echo 'Shut down aborted.'
        exit 1
fi
case "${initstate}" in
s | S )
    . /etc/rc0
esac
/etc/init ${initstate}
```

Figure B-13: Typical **/etc/shutdown** File (Sheet 4 of 4)

/etc/shutdown.d Directory

The executable files in **/etc/shutdown.d** do the various functions required during the transition to the single-user state (run levels 1, s, or S). (Files contained in this directory before UNIX System V Release 3.0 were moved to **/etc/rc0.d**. This directory is only maintained for compatibility reasons.)

/etc/TIMEZONE

The **/etc/TIMEZONE** file sets the time zone shell variable TZ. The TZ variable is initially established for the system via the System Administration **setup** function. The TZ variable in the **TIMEZONE** file is changed by the System Administration **timezone** command (**sysadm timezone**). 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 **/etc/rc2**. Figure B-14 shows a typical **/etc/TIMEZONE** file.

A typical format of the **TZ** is as follows [see the **TIMEZONE**(4) manual page in the *User's and System Administrator's Reference Manual* for information on other formats]:

TZ=*TTT#SSS*

Explanations for the fields of the **TZ** variable are as follows:

TTT	The three-character abbreviation for the local time zone.
#	The number of hours that the local time zone differs from Greenwich Mean Time (GMT). This field can be entered as a positive or negative number.
SSS	The three-character abbreviation for the local daylight savings time zone. This field is entered only if daylight savings time is observed.

```
# cat /etc/TIMEZONE
#ident  "@(#)/etc/TIMEZONE.sl 1.1 3.0 11/18/85 9786 "
#       Set timezone environment to default for the machine
TZ=EST5EDT
export TZ
#
```

Figure B-14: Typical **/etc/TIMEZONE** File

/etc/utmp

The **/etc/utmp** file contains information on the run-state of the system. This information is accessed with a **who -a** command.

/etc/wtmp

The **/etc/wtmp** 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 **>/etc/wtmp** when executed by **root** creates the file with nothing in it. The following command line limits the size of the **/etc/wtmp** file to the last 3600 characters in the file:

tail -3600c /etc/wtmp > /tmp/wtmp; mv /tmp/wtmp /etc/wtmp

Note: Make certain you use the character option for tail and that the number specified (3600) is a multiple of 36.

Note: The **/etc/cron**, **/etc/rc0**, or **/etc/rc2** can be used to clean up the **wtmp** file. To use one of these functions, add the appropriate command line to the **/usr/spool/cron/crontab/root**, **/etc/shutdown.d/...**, or **/etc/rc.d/, rc2.d, rc3.d ...** file.

/usr/adm/conlog

The **/usr/adm/conlog** file is a record of all console I/O (all stdin, stdout, and stderr are saved). You must be root to read the **conlog** file.

Caution: Use the *conslog -r* command to read the *conlog* file when you are logged in on the console with the console logger active. Deactivate the console logger before executing commands that put the console in raw mode (for example, *pg, vi*), or commands that are required to be attached to a tty device when the console logger is activated. Do not *cat* the conlog file from the console with the console logger active.

Every time the console logger is activated the old log file (**/usr/adm/conlog**) is moved to **/usr/adm/conlog***mmddhhmm*. The console log file will grow until the UNIX system file size limit is reached, at which time an error message is displayed. Because of this the log needs to be moved periodically. You can do this by simply deactivating and then activating the logger.

Figure B-15 shows the contents of a typical **/usr/adm/conlog** file.

```
WR3B2D 09:26:51
root> cat /usr/adm/errlog > stwtmp5
WR3B2D 09:27:13
root> uuto -p -m stwtmp5 wr3b2c!stw
WR3B2D 09:27:45
root> uustat
wr3b2cN02ba   03/20-09:27  S  wr3b2c  root 903 /etc/stwtmp5
WR3B2D 09:27:49
root> lpstat
WR3B2D 09:28:24
root> conslog-d
```

Figure B-15: Typical **/usr/adm/conlog** File

/usr/adm/errlog

The **/usr/adm/errlog** file contains all the driver error messages. This includes the messages that are reported to *hdelogger*. This logger is a complete history of all the driver error messages that have occurred on the system. There is a cron task that controls the growth of the error log file. This cron task copies the error log file into **/usr/adm/Oerrlog** once a week. If you find the **errlog** file growing too large in the one week interval, you may need to change the cron task.

A time stamp is placed in the buffer with each message. With this information you can determine when each error occurred. The time stamp is in the form *time*: where *time* is the number of seconds since January 1, 1970.

/usr/adm/loginlog

The occurrence of repetitive unsuccessful attempts to access the system (log in) can be tracked (logged). Fewer than five consecutive unsuccessful login attempts on a given port (CONSOLE, CONTTY, or tty) are not logged. As the system is delivered, this logging mechanism is turned off. To turn on the mechanism that logs unsuccessful attempts to access the system, the **/usr/adm/loginlog** file must be created. This file should have read and write permissions for only **root**. To turn off unsuccessful login logging, remove the **/usr/adm/loginlog** file.

The format of the entries in this file are in the following form:

login_name:port:time

Figure B-16 shows the contents of a typical **/usr/adm/loginlog** file resulting from one occurrence of five consecutive unsuccessful login attempts on the **/dev/contty** port.

```
# cat /usr/adm/loginlog
rar:/dev/contty:Thu Mar 31 05:19:50 1988
rar:/dev/contty:Thu Mar 31 05:19:57 1988
rar:/dev/contty:Thu Mar 31 05:20:05 1988
cms:/dev/contty:Thu Mar 31 05:20:16 1988
cms:/dev/contty:Thu Mar 31 05:20:24 1988
#
```

Figure B-16: Typical **/usr/adm/loginlog** File

/usr/adm/sulog

The **/usr/adm/sulog** file contains a history of substitute user (**su**) command usage. As a security measure, this file should not be readable by others. The **/usr/adm/sulog** file should be periodically truncated to keep the size of the file within a reasonable limit. Note that **/etc/cron**, **/etc/rc0**, or **/etc/rc2** can be used to clean up the **sulog** file. To use one of these functions, add the appropriate command line to the **/usr/spool/cron/crontab/root**, **/etc/shutdown.d/...**, or **/etc/rc.d/, rc2.d, rc3.d ...** file. The following command lines limit the size of the log file to the last 100 lines in the file:

```
tail -100 /usr/adm/sulog > /tmp/sulog
mv /tmp/sulog /usr/adm/sulog
```

Figure B-17 shows the contents of a typical **/usr/adm/sulog** file.

```
SU 08/18 12:35 + console root-sysadm
SU 08/18 16:11 + console root-sysadm
SU 08/18 16:16 + console root-sysadm
SU 08/18 23:45 + tty?? root-uucp
SU 08/19 11:53 + console root-sysadm
SU 08/19 15:25 + console root-sysadm
SU 08/19 23:45 + tty?? root-uucp
SU 08/20 10:16 + console root-adm
SU 08/20 10:33 + tty24 rar-root
SU 08/20 10:42 + console root-sysadm
SU 08/20 10:59 + console root-root
SU 08/20 11:01 + console root-sysadm
SU 08/20 12:36 + tty11 bin-bin
SU 08/20 12:37 + tty11 tws-bin
SU 08/20 14:42 - tty24 awa-sys
SU 08/20 14:47 - tty24 awa-sys
SU 08/20 14:48 + tty24 awa-root
SU 08/20 15:44 + console root-sysadm
```

Figure B-17: Typical **/usr/adm/sulog** File

/usr/lib/cron/log

A history of all actions taken by **/etc/cron** is recorded in the **/usr/lib/cron/log** file. The **/usr/lib/cron/log** file should be periodically truncated to keep the size of the file within a reasonable limit. Note that **/etc/cron**, **/etc/rc0**, or **/etc/rc2** can be used to clean up the **/usr/lib/cron/log** file. To use one of these functions to limit the size of a log file, add the appropriate command line to the **/usr/spool/cron/crontab/root**, **/etc/shutdown.d/...**, or **/etc/rc.d/ rc2.d, rc3.d ...** file, as applicable. The following command line is one example of how to limit the size of this file:

cp /usr/lib/cron/log /usr/lib/cron/oldlog; > /usr/lib/cron/log

Figure B-18 shows the information typically found in the **/usr/lib/cron/log** file.

```
! *** cron started ***   pid = 237 Thu May 19 14:06:45 1988
>  CMD: /usr/lib/uucp/uudemon.hour > /dev/null
>  root 251 c Thu May 19 14:11:00 1988
<  root 251 c Thu May 19 14:11:01 1988
>  CMD: /usr/lib/uucp/uudemon.poll > /dev/null
>  root 370 c Thu May 19 14:30:00 1988
<  root 370 c Thu May 19 14:30:03 1988
>  CMD: /usr/lib/uucp/uudemon.hour > /dev/null
>  root 417 c Thu May 19 14:41:01 1988
<  root 417 c Thu May 19 14:41:02 1988
>  CMD: /usr/lib/uucp/uudemon.poll > /dev/null
>  root 452 c Thu May 19 15:01:00 1988
<  root 452 c Thu May 19 15:01:04 1988
>  CMD: /usr/lib/uucp/uudemon.hour > /dev/null
>  root 460 c Thu May 19 15:11:00 1988
<  root 460 c Thu May 19 15:11:00 1988
>  CMD: /usr/lib/uucp/uudemon.poll > /dev/null
>  root 541 c Thu May 19 15:30:00 1988
<  root 541 c Thu May 19 15:30:07 1988
```

Figure B-18: Typical **/usr/lib/cron/log** File

/usr/lib/help/HELPLOG

Providing that HELP Utilities are installed and monitoring has been enabled, a history of all actions taken by the **/usr/bin/help** command is kept in the **/usr/lib/help/HELPLOG** file. The **HELPLOG** file is copied to **/usr/lib/help/oHELPLOG**, and a new **/usr/lib/help/HELPLOG** file is created by the **/usr/lib/help/helpclean** command. Executing the **helpclean** command twice in succession zeros out the **HELPLOG** and the **oHELPLOG** files. Note that **/etc/cron**, **/etc/rc0**, or **/etc/rc2** can be used to clean up the **HELPLOG** file. To use one of these functions, add the appropriate command line to the **/usr/spool/cron/crontab/root**, **/etc/shutdown.d/...**, or **/etc/rc.d/ rc2.d, rc3.d ...** file, as applicable. The following command lines limit the size of the log file to the last 100 lines in the file:

```
tail -100 /usr/lib/help/HELPLOG > /tmp/help
mv /tmp/help /usr/lib/help/HELPLOG
```

Figure B-19 shows the information typically found in the **/usr/lib/help/HELPLOG** file.

```
login=bin       uname=wr3b2a    date=Fri May 20 12:51:03 EDT 1988
name=locate     response='l'    status=OK
name=locate     response='d'    status=ERROR
name=getkey     response='k'    status=OK
name=keysrch    response='list' status=OK
name=quit       response='q'    status=OK
login=bin       uname=wr3b2a    date=Fri May 20 12:51:41 EDT 1988
```

Figure B-19: Typical **/usr/lib/help/HELPLOG** File

/usr/lib/spell/spellhist

If the Spell Utilities are installed, a history of all words that **spell**(1) fails to match is kept in the **/usr/lib/spell/spellhist** file. Periodically, this file should be reviewed for words that should be added to the dictionary. After the **spellhist** file is reviewed, it can be cleared.

/usr/news

The **/usr/news** 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).

/usr/options Directory

The **/usr/options** directory contains files that identify the utilities that are installed on the system. Figure B-20 shows a typical **/usr/options** directory. Because no system has all possible utilities installed on it at once, the list of files in the **/usr/options** directory will not include all utilities.

The example shown in Figure B-20 was taken from a dual-disk system. The full names of the utilities are contained in the **/usr/options** directories. The contents of the directories identified in Figure B-20 are shown in Figure B-21.

```
-rx-r--r--   1 root     other        32 Nov 19  1985 SuperCalc.name
-rx-r--r--   1 bin      bin          47 Nov 28 15:55 acct.name
-r-xr-xr-x   1 bin      bin          40 Jan 30  1987 acu.name
-r-xr-xr-x   1 bin      bin          22 Mar  3  1988 calc.name
-r-xr-xr-x   1 bin      bin          42 Jan 30  1987 cc.name
-r--r--r--   1 bin      bin          34 Feb 19  1986 crypt.name
-r-xr-xr-x   1 bin      bin          40 Feb  4  1987 dfm.name
-r-xr-xr-x   1 bin      bin          52 Dec  8  1985 dwb.name
-r--r--r--   1 bin      bin          18 Mar  3  1988 ed.name
-r-xr-xr-x   1 root     root         46 Dec 13 10:31 eports.name
-rw-r--r--   1 root     other        55 Dec 16 12:59 eps.name
-r-xr-xr-x   1 bin      bin          58 Jan 30  1987 esg.name
-r--r--r--   1 root     other        40 Jan  5 05:39 fmli.name
-r-xr-xr-x   1 bin      bin          56 Jun 23  1988 ipc.name
-r-xr-xr-x   1 bin      bin          22 Mar  3  1988 lp.name
-r--r--r--   1 bin      bin          51 Apr 28  1988 mirror.name
-r--r--r--   1 bin      bin          59 Nov 29 14:00 mpb.name
-r--r--r--   1 bin      bin          47 Nov 10 09:25 nsu.name
-r-xr-xr-x   1 bin      bin          59 Mar  3  1988 perf.name
-rw-r--r--   1 root     sys          53 Nov 17 12:23 rfs.name
-r-xr-xr-x   1 bin      bin          30 Jan 30  1987 sccs.name
-r--r--r--   1 bin      bin          49 Sep 15 10:40 scsi.name
-r--r--r--   1 bin      bin          52 Dec 12 15:38 sd01.name
-r-xr-xr-x   1 bin      bin          49 Jan 30  1987 sgs.name
-r-xr-xr-x   1 bin      bin          28 Mar  3  1988 shell.name
-r-xr-xr-x   1 bin      bin          16 Feb 28  1986 spell.name
-r--r--r--   1 bin      bin          51 Apr 28  1988 st01.name
-r-xr-xr-x   1 bin      bin          38 Jun  2  1988 sys.name
-r-xr-xr-x   1 bin      bin          50 Jun  2  1988 sysadm.name
-r-xr-xr-x   1 bin      bin          27 Feb  7  1986 term.name
-r--r--r--   1 bin      bin          31 Mar  3  1988 terminf.name
-r-xr-xr-x   1 bin      bin          27 Mar  3  1988 usrenv.name
-rwxr-xr-x   1 root     sys          27 Mar  3  1988 uucp.name
-r--r--r--   1 root     other        10 Jan  5 05:40 vm.name
-rw-r--r--   1 bin      bin          25 Feb  2  1987 windowing.name
-rw-r--r--   1 bin      bin          69 Aug 14  1985 wwb.name
```

Figure B-20: Typical **/usr/options** Directory

```
SuperCalc3     Release 2.0      11/04/85
AT&T 3B2 Job Accounting Utilities  Release 1.0
Advanced C Utilities: Issue 4 Version 1
Calculation Utilities
C Programming Language: Issue 4 Version 1
Security Administration Utilities
Directory and File Management Utilities
DOCUMENTER'S WORKBENCH System 2.0        November 21, 1985
Editing Utilities
AT&T 3B2 Enhanced Ports Utilities Release 1.2
Electronic Publishing System 2.0 Update - January 1989
Extended Software Generation Utilities: Issue 4 Version 1
AT&T Form and Menu Language Interpreter
Inter-Process Communication Utilities: Release 3.2.2 V3
LP Spooling Utilities
AT&T 3B2 SCSI Disk Mirroring Utilities Release 1.0
AT&T 3B2 Multiprocessor Enhancement Utilities Release 3.0
Networking Support Utilities: Release 3.2.2 V3
System Performance Analysis Utilities - Release 1.1 (SVR3)
Remote File Sharing Utilities: Issue 3.2.2 Version 3

Source Code Control Utilities
AT&T 3B2 SCSI Host Adapter Utilities Release 3.1
AT&T 3B2 SCSI Disk Controller Utilities Release 3.0
Software Generation Utilities: Issue 4 Version 1
Shell Programming Utilities
SPELL Utilities
AT&T 3B2 SCSI Cartridge Tape Utilities Release 3.0
System Header Files: Release 3.2.2 V3
System Administration Utilities: Release 3.2.2 V3
Terminal Filters Utilities
Terminal Information Utilities
User Environment Utilities
Basic Networking Utilities
AT&T FACE
AT&T Windowing Utilities
The UNIX (TM) WRITER'S WORKBENCH (TM) Software, Release 3.0, 8/14/85
WR3B2D 09:53:56 /usr/options
root>
```

Figure B-21: Typical **/usr/options** File Directory Contents

/usr/spool/cron/crontabs

The **/usr/spool/cron/crontabs** directory contains crontab files for **adm**, **root**, and **sys** logins. Providing their lognames are in the **/usr/lib/cron/cron.allow** file, users can establish their own **crontabs** file using the **crontab** command. If the **cron.allow** file does not exist, the **/usr/lib/cron/cron.deny** file is checked to determine if the user is denied the use of the **crontab** command.

As **root**, you can either use the **crontab**(1) command or edit the appropriate file under **/usr/spool/cron/crontabs** to make the desired entries. Revisions to the file take affect at the next reboot. The line entry format of a **/usr/spool/cron/crontabs/***logname* file is as follows:

minute hour day month day-of-week command

The various fields of a **crontabs/***logname* line entry are the following:

minute	The minutes field is a one- or two-digit number in the range 0 through 59.
hour	The hour field is a one- or two-digit number in the range 0 through 24.
day	The day field is the numerical day of the month in the range 1 through 31.
month	The month field is the numerical month of the year in the range 1 through 12.
day-of-week	The day-of-week field is the numerical day of the week where Sunday is 0, Monday is 1, . . . and Saturday is 6.
command	The command field is the program or command that is executed at the time specified by the first five fields.

The following syntax applies to the first five fields:

- Two numbers separated by a minus specifies an inclusive range of numbers between the two specified numbers.
- A list of numbers separated by commas specifies all the numbers listed.
- An asterisk specifies all legal values.

In the command field (sixth field), a percent sign (%) is translated to a new-line character. Only the first line of a command field (character string up to the percent sign) is executed by the shell. Any other lines are made available to the command as standard input.

Figure B-22 shows a typical **/usr/spool/cron/crontabs/***logname* file. The data shown are the **root** file. The file entries support the **calendar** reminder service and basic networking. Remember, you can use the **cron** function to decrease the number of data terminal driven system administration tasks: include recurring and habitual tasks in your crontab file.

```
0 1 * * * /usr/bin/calendar -
41,11 * * * * /usr/lib/uucp/uudemon.hour > /dev/null
45 23 * * * ulimit 5000; /bin/su uucp -c "/usr/lib/uucp/uudemon.cleanup"
  > /dev/null 2>&1
1,30 * * * * /usr/lib/uucp/uudemon.poll > /dev/null
```

Figure B-22: Typical **/usr/spool/cron/crontabs/root** File

[Refer to the **crontab**(1) manual page in the *User's and System Administrator's Reference Manual* for additional information.]

Appendix C: Error Messages

General

This appendix contains descriptions of some of the more common error messages that you will see as a System Administrator. It is not a complete listing. A complete listing of error messages is provided in the *AT&T 3B2 Computer Error Message Manual* which can be ordered by following the "ORDERING INFORMATION" given in the front of this document.

The following categories of error messages are included in this appendix:

- Firmware error messages
- EDT completion error messages
- Boot error messages
- Diagnostic Monitor error messages
- UNIX system error messages
- Expansion Disk Controller (XDC) error messages

Descriptions of SCSI error messages can be found in the *SCSI Operations Manual.*

The *AT&T 3B2 Computer Error Message Manual* contains all the error messages included in this appendix and the *SCSI Operations* manual plus many other error messages.

Error Message Tables

A tabular presentation is used in this appendix for listing the error messages. Two different table formats are used. One table format has two columns. The first column provides a "copy" of the error message as it is displayed by the computer. The second column of the table provides information about the error message in the form of a description, an action to be taken, and sometimes a reference (shown in *italic* type) to a helpful

document or to the source code that generates the error message. The following is an example of the two column format:

Error Message	Description/Action *(Reference)*

The other table format has three columns. This type of table is used for those error messages that have a number associated with them. An example of the three column table format is shown below. The first column provides the error message number; the second column provides the message that is displayed; and the third column provides a description of the message, the action to take and a reference when applicable.

Number	Message	Description/Action *(Reference)*

Error messages are grouped together according to the type of error. Within the tables the error messages are listed alphabetically or numerically.

Error Message Conventions

The following conventions are used throughout this appendix. If you cannot find the exact error message, look for possible variables within the message which may change the alphabetical placement of the message.

- An *n* is used to represent number variables, and *str* is used to represent string variables.

- The following identification (id) numbers may be printed out in the error messages:
  - **slot** *n* - slot number on the Input/Output (I/O) bus
  - **tc** *n* - target controller id number (SCSI only)
  - **Unit** *n* - disk driver id number
  - **lu** *n* - logical unit id number.
- The messages that require no action are "information only" messages. They may provide helpful hints for locating a problem.

There may be some variables used in this appendix that are not listed here. If other variables are used, they will be defined in the introductory section before the associated table.

Firmware Error Messages

The firmware mode is a run level of the 3B2 computer which allows you to run special programs; for example, diagnostic programs. If a problem occurs while the computer is in the firmware mode, a firmware error message is displayed on the console terminal. Firmware error messages are identified numerically and are prefaced by:

FW ERROR *n*

If the firmware Read Only Memories (ROMs) have an issue number greater than 0x20 (the firmware **version** program lists the firmware issue number), you can get additional information about DISK SANITY FAILURE, UNEXPECTED FAULT, or UNEXPECTED INTERRUPT errors. This capability is provided by the **errorinfo** command which is executed in the firmware mode. When prompted for the "name of program to execute," simply enter:

errorinfo<*CR*>

The following types of errors are reported when the **errorinfo** command is used:

- Interrupt Messages
- Exception Messages
- Abort Messages
- Thermal Shutdown.

Caution: Executing the firmware-level command `errorinfo` outputs and clears the expanded error information stored in NVRAM. Be sure to either copy the displayed output or have a printer enabled when the command is first executed.

Interrupt Messages

Interrupt message expansion provides values for the Program Counter (PC), Program Status Word (PSW), the Control Status Error Register (CSER), Fault Latches 1 and 2, and priority Level (LVL) at the time the system was interrupted. The format of the message is shown in the following display:

```
FIRMWARE MODE
password
Enter name of program to execute [   ]:errorinfo
INTERRUPT,    LVL=13
PC=0xnnnnnnnn
PSW=0xnnnnnnnn
CSER=0xnnnnnnnn
FL1=0xnnnnnnnn
FL2=0xnnnnnnnn
```

Exception Messages

Exception message expansion provides values for the Program Counter (PC), Program Status Word (PSW), the Control Status Error Register (CSER), and Fault Latches 1 and 2. The format of the message is as follows:

```
FIRMWARE MODE
password
Enter name of program to execute [   ]:errorinfo
EXCEPTION
PC=0xnnnnnnnn
PSW=0xnnnnnnnn
CSER=0xnnnnnnnn
FL1=0xnnnnnnnn
FL2=0xnnnnnnnn
```

Abort Messages

Abort message expansion provides values for the Program Counter (PC), Program Status Word (PSW), the Control Status Error Register (CSER), and Fault Latches 1 and 2. The format of the message is shown in the following display:

```
ABORT
PC=0xnnnnnnnn
PSW=0xnnnnnnnn
CSER=0xnnnnnnnn
FL1=0xnnnnnnnn
FL2=0xnnnnnnnn
```

Thermal Shutdown

The thermal shutdown message does not provide for message expansion. An example is shown in the following display:

```
FIRMWARE MODE

password

Enter name of program to execute [ ]:errorinfo

THERMAL SHUTDOWN
```

FIRMWARE ERROR MESSAGES

Number	Message	Description/Action *(Reference)*
2-01	**NVRAM SANITY FAILURE DEFAULT VALUES ASSUMED IF REPEATED, CHECK THE BATTERY**	Data stored in nonvolatile memory has been corrupted. Repeat system powerup. If the same message appears, check the voltage of the NVRAM battery. Default values are assumed.
2-02	**NO LOAD DEVICE IN SLOT** *n*	There is no Input/Output (I/O) board in the slot specified. The load device, as specified in nonvolatile RAM, must be an equipped peripheral (if other than the integral floppy). Ensure that the default load device refers to a peripheral slot that is equipped. The default device can be examined by trying a demand load of a program and examining the default device prompt. This can then be compared with the equipped device table.
2-03	**UNEXPECTED FAULT**	The processor checked any one of a number of fault types; most likely an external memory fault due to a parity error or an attempted access to an unequipped memory location. Additional information can be obtained using **errorinfo**. The resulting message will be of the form: **EXCEPTION, PC = 0x?, PSW = 0x?, CSER = 0x?, FL1 = 0x?, FL2 = 0x?** where **PC** is the value of the Program Counter, **PSW** is the Program Status Word for the system, **CSER** is the value of the Control Status Error Register, and **FL1** and **FL2** are Fault Latches that save the address where the failure occurred when the exception occurred. Check code if fault is localized. If not, check hardware.

Number	Message	Description/Action *(Reference)*
2-04	**UNEXPECTED INTERRUPT**	One of several sources of interrupts caused an interrupt at a time when firmware was not expecting one. Additional information can be obtained using **errorinfo**. The resulting message will be of the form: **INTERRUPT, PC = 0x?, PSW = 0x?, LVL = 0x?** where **PC** is the value of the Program Counter, **PSW** is the Program Status Word when the system was interrupted, and **LVL** is the priority Level of the interrupt. Check interrupt sources.
2-05	**SELF-CONFIGURATION FAILURE**	An exception occurred, other than the one expected (external memory fault) when trying to access the first unequipped slot. Check Input/Output (I/O) cards and buffered microbus devices for good connections or for a skipped slot.
2-06	**BOOT FAILURE**	Boot of a program failed. If floppy disk boots, ensure that correct floppy is in the drive. This message may also be a result of an incomplete **filledt**, diagnostic, or any program execution. If so, reboot the system.
2-07	**FLOPPY KEY CREATE FAILURE**	Unable to write floppy. Ensure that a formatted nonwrite-protected floppy disk is in the drive when "go" is entered.

Number	Message	Description/Action *(Reference)*
2-08	**MEMORY TEST FAILURE**	On powerup, the system tests the first 256 kilobytes of main store (main memory), the part of main store that contains the diagnostic code in the powerup sequence. Retry request. If it fails again, a test failure occurred in the first 256 kilobytes of system memory; reseat the memory card or substitute memory if possible and retry.
2-09	**UNEXPECTED SANITY TIME-OUT EXECUTION HALTED**	The user should never see this error message. The computer hardware has sensed that the machine is not operating correctly and issued a non-maskable interrupt. The firmware executes with the sanity timer disabled. Check diagnostic software for prolonged operation at an incorrectly high interrupt priority level that would prevent resetting of this timer or an omission of code to reset it.
2-11	**MEMORY CONFIGURATION OF *n* MEGABYTES UNSUPPORTED MAXIMUM IS 16 MEGABYTES**	The system hardware only supports 16 megabytes of memory; more than 16 megabytes of memory have been installed. The system simply fails to boot. The machine will not execute any program from a load device until the memory configuration is reduced. Firmware-level commands can still be used (that is, the **edt** command to show what memory cards are equipped.)

Number	Message	Description/Action *(Reference)*
2-12	**MEMORY GAP IN SLOT** *n*	This message appears if the memory slots are not filled in sequential order. The memory beyond the gap is not recognized, so you may incorrectly think that more memory is available than is actually available. Operation is unaffected, however, only the contiguous memory is accessible. Rearrange the memory boards so that they are in sequential order.

Error Message	Description/Action *(Reference)*
if CRC error at disk address *X*	The integral floppy disk 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).
PERIPHERAL I/O READ(WRITE) ERROR AT BLOCK *n*, **SUBDEVICE** *n*, **SLOT** *n*	Read/write failure detected on a peripheral boot device. If the system will boot (run UNIX operating system), add the identified defect to the defect map using the **hdeadd** and **hdefix** command. Capture the block number, subdevice number, and slot number from the error message.

Equipped Device Table Completion Error Messages

The Equipped Device Table (EDT) completion program (**filledt**) provides the ability to complete the EDT. If a problem occurs that affects the file system or system configuration while using the EDT completion program, an error message will be displayed on the terminal. All of the EDT errors, except 2-08 through 2-13, are suppressed during autoboot. If the system is manually booted, no errors are suppressed.

Each EDT completion error message is prefaced by:

EDT COMPLETION ERROR

The **editsa** command error messages are also in this section. The **editsa** command is meant to be used in the installation scripts of software packages. If an error occurs during the installation, either the software driver or the hardware device did not get initialized properly. The result is an incomplete installation of the package.

Number	Message	Description/Action *(Reference)*
2-00	**FILE SYSTEM IS INACCESSIBLE. CONTROL WILL RETURN TO MAINTENANCE CONTROL PROGRAM.**	The **filledt** code cannot locate the root file system offset. Since **filledt** itself is part of this file system, very recent corruption has occurred. Retry request. If it fails again, there is a problem with the root file system where diagnostics reside. It may be necessary to restore the file system.
2-01	**ERROR OCCURRED DURING SYSTEM CONFIGURATION. CONSOLE LOCATION PROCEEDING. CHECK EDT.**	A device may have failed the Determine Sub-Device (DSD) sequence of SYSGEN. An error occurred during system configuration. Check equipped device table; device entry garbled. Verify device ID code, and check the look-up table in */dgn/edt_data* using the **edittbl** routine.
2-02	**CANNOT FIND FILE: (file name)**	The **filledt** program cannot find the file identified in the message. Retry request. If it fails again, restore it from a backup.
2-03	**CANNOT LOAD FILE: (*name*)**	The **filledt** program cannot load the file from the root file system. Retry request. If it fails again, check the file. It may be zero length, have an invalid magic number, etc.

Number	Message	Description/Action *(Reference)*
2-04	**UNEXPECTED EXCEPTION**	The processor detected an unexpected exemption, probably due to an attempt to address an invalid memory location. Retry request. If it fails again, check hardware.
2-05	**UNEXPECTED INTERRUPT**	The processor detected an unexpected interrupt from any one of the components that can produce interrupts. Retry request. If it fails again, check interrupt sources such as peripheral cards, disk subsystems, etc.
2-06	**SYSGEN FAILED FOR (*name*) IN SLOT (slot *n*) EQUIPPED DEVICE TABLE COMPLETION WILL CONTINUE. CHECK EDT.**	The **filledt** program attempts to SYSGEN or "turn on" smart devices (those that support SYSGEN) before it can query them about possible hardware subdevices. This message appears if the SYSGEN attempt fails for a device. The peripheral device is not responding to configuration requests. Retry request. If it fails again, check device.
2-07	**DSD FAILED FOR (device name) IN SLOT (slot *n*) EQUIPPED DEVICE TABLE COMPLETION WILL CONTINUE. CHECK EDT.**	The **filledt** program asks each "smart" device what subdevice it has in the Determine Sub-Device (DSD) command once it has been SYSGENed. If it fails the DSD query, this message appears. The peripheral device is not responding to configuration requests. Retry request. If it fails again, check device.

Number	Message	Description/Action *(Reference)*
2-08	**UNKNOWN ID CODE (id code) IN SLOT (slot *n*) EQUIPPED DEVICE TABLE COMPLETION WILL CONTINUE. CHECK EDT.**	The **filledt** program prints this message when it cannot find a device ID code in the file edt_data to match the value that the device returned for the EDT. This may happen for newly-installed hardware during the installation process or when a device has malfunctioned. If device installation is in progress, proceed. If not, retry request. If a failure recurs, check the look-up table in */dgn/edt_data* using the **edittbl** routine, and check the device ID code using the "edt" firmware function.
2-09	**UNKNOWN SUBDEVICE ID CODE FOR DEVICE (device name) IN SLOT (slot *n*) EQUIPPED DEVICE TABLE COMPLETION WILL CONTINUE. CHECK EDT.**	The **filledt** program prints this message when a subdevice ID code collected with the SYSGEN-DSD queries is not part of the edt_data look-up table. This may happen for newly-installed hardware during the installation process or when a peripheral device or subdevice has malfunctioned. If device installation is in progress, proceed. If not, retry request. If a failure recurs, check the subdevice look-up table in */dgn/edt_data* using the **edittbl** routine, and check the subdevices ID code using the "edt" firmware function.

Number	Message	Description/Action *(Reference)*
2-10	**EDT EXCEEDS ALLOCATED SPACE AND CANNOT BE COMPLETED. REDUCE SYSTEM CONFIGURATION.**	The **filledt** program checks the upper address of the EDT as it completes the entries. If the maximum table size is about to be exceeded, this message will appear. It will not cause a problem for normal use. Remove unnecessary devices. Retry request.
2-11	**SOFTWARE APPLICATION FILE ERROR - ENTRY FOR SLOT *n* DOES NOT MATCH EDT DEVICE NAME, *name*. EQUIPPED DEVICE TABLE COMPLETION WILL CONTINUE. CHECK EDT.**	The software application file (*/dgn/.edt_swapp*) lists a device for slot n that is to be renamed. However, the Equipped Device Table (EDT) has a different device name than the one listed in the software application file (*/dgn/.edt_swapp).* Check EDT and */dgn/.edt_swapp* file and change either the system hardware configuration or the */dgn/.edt_swapp* file.
2-12	**SOFTWARE APPLICATION FILE ERROR - EDT HAS NO DEVICE IN SLOT *n*. EQUIPPED DEVICE TABLE COMPLETION WILL CONTINUE. CHECK EDT.**	The software application file (*/dgn/.edt_swapp*) lists a device for slot *n* that is to be renamed. However, the Equipped Device Table (EDT) has no entry for slot *n*. Check EDT and */dgn/.edt_swapp* file and change */dgn/.edt_swapp* file appropriately using **editsa** command.

Number	Message	Description/Action *(Reference)*
2-13	**SOFTWARE APPLICATION FILE ERROR - INCOMPLETE ENTRY FOR SLOT *n*. EQUIPPED DEVICE TABLE COMPLETION WILL CONTINUE. CHECK EDT**	The application file (*/dgn/.edt_swapp*) is missing one or both of the device name character strings for the device in slot *n*. Use **-l** option of **editsa** to inspect the contents of */dgn/.edt_swapp*. Make appropriate fixes to application file using the **editsa** command.

Message	Description/Action *(Reference)*
editsa: ERROR, driver *driver* not found in /boot	The software driver name (*driver*) specified in the **editsa** command is not valid. There is not a software driver under */boot* that matches the argument. Ensure that the software driver name (*driver*) specified is valid. Check for incorrect spelling, etc. Make sure that the software containing the specified driver has been properly installed on the hard disk media via the appropriate install procedure. *(/boot)* *(/dgn/.edt_swapp)* *(/etc/editsa)*
editsa: ERROR, HWNAME and SWNAME specified are identical	The new name given as an argument to the **editsa** command is the same as the existing name. Therefore, this invocation will have no affect on the system. This execution is assumed to be a mistake. Correct the command line and reexecute.
editsa: ERROR, *name* does not match EDT entry for slot *n*	This message indicates verification of the hardware name (*name*) and backplane slot (*n*) passed by the **editsa** command failed against the current entries in the Equipped Device Table (EDT). Ensure the command line is correct. If not, retry the command using proper arguments. If the command was correct, perform a **filledt** and reboot the system using */etc/system* as the boot file.

Message	Description/Action *(Reference)*
editsa: ERROR, missing software application file	An attempt to modify, add, or delete an entry in the */dgn/.edt_swapp* file failed because the file does not exist possibly due to file corruption. Restore the file from backup. If no previous entry existed, the file can be re-created by executing **touch /dgn/.edt_swapp**. Then reinstall the package that was being installed when the error occurred. *(/dgn/.edt_swapp)* *(/etc/editsa)*
editsa: ERROR, *name* not found in software application file	An attempt to delete an entry in the */dgn/.edt_swapp* file failed because the entry is not present. Check for obvious problems such as spelling errors. Use the **-l** option of the **editsa** command to display the contents of *.edt_swapp* file.
editsa: ERROR, slot number *n* is invalid	The backplane slot specified in the **editsa** command is not a valid number. Re-enter the command line using a valid backplane slot number. Valid numbers are: 1 through 12 for a 3B2/600 or 3B2/700 computer 1 through 7 for a 3B2/500 computer *(/dgn/.edt_swapp)* *(/etc/editsa)*

Boot Error Messages

Boot firmware provides the user the ability to execute a number of disk resident programs. These programs include the Diagnostic Monitor, the UNIX operating system, and the utilities. If a problem occurs while attempting to execute one of these programs, an error message is displayed at the console terminal.

A boot PANIC message results in a second message being printed and self-configuration entering an endless loop. The only escape from this loop is to reset the machine.

All boot error messages are listed alphabetically with a short description and corrective action. The variables used in this chapter and what they represent are as follows:

VARIABLE	MEANING
n	number
file	file name
name	file or device name
driver	driver name
string	an expression

Error Message (Prefaced by PANIC)	Description/Action *(Reference)*
name	Symbol *name* could not be resolved. Determine where symbol *name* should be defined, then recompile and reboot. *(boot/lboot/tables.c)*
cannot chdir(/)	Cannot change directory to root (/). Action depends on previously printed message. *(boot/lboot/main.c)*
cannot mount root	An I/O error occurred while the system was trying to mount the root file system. Make sure that the disk or tape you are trying to boot contains a copy of the root file system. Attempt to boot from a backup root. If unsuccessful, attempt to boot from a different root disk or tape. *(os/main.c)* *(boot/lboot/basicio.c)*
file table overflow	The maximum number of allowable open files as defined by NFILE in /etc/master.d/kernel was exceeded. Default value is 100. This indicates self-configuration has been corrupted. Reconfigure and reboot.
Illegal error action	This indicates that self-configuration has been corrupted. Try rebooting the system. *(boot/lboot/main.c)*

Error Message (Prefaced by PANIC)	Description/Action *(Reference)*
inode table overflow	The maximum number of i-node table entries was exceeded (system default is 100). Indicates that self-configuration has been corrupted. Reconfigure /etc/master.d/kernel and reboot. *(/etc/master.c/kernel)*
inode locked	The requested i-node is already in use. Indicates that self-configuration has been corrupted.
out of free blocks	All available buffers in use. Indicates self-configuration has been corrupted. Reboot system. *(boot/lboot/basicio.c)*
textSIZE	Actual text size of all object modules to be loaded plus size of interrupt routines not equal to calculated size. Indicates self-configuration has been corrupted. Try rebooting the system. *(boot/lboot/loadunix.c)*
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. Reboot the system. *(boot/lboot/loadunix.c)* *(boot/lboot/error.c)*

Self-Configuration Messages

The boot error messages that follow are warning and error messages printed by self-configuration.

Error Message	Description/Action *(Reference)*
***bootprgm* configured for more memory than available - use /etc/system**	This is a warning message indicating a fatal error and will only be seen during a manual boot of the system. The message indicates that the amount of physical memory has been decreased since the creation of the boot program (bootprgm). If this condition exists during autoboot (powerup), the message is suppressed and **/etc/system** is used. The message is associated with automatic tuning of the kernel NBUF parameter at "boot" time. Reboot the system specifying **/etc/system** as the boot program or increase the amount of physical memory available. *(boot/lboot/loadunix.c)* *(boot/lboot/error.c)*
***bootprgm* configured for less memory than available**	This is a warning message which will only be seen during a manual boot of the system. The message indicates that the NBUF kernel parameter is below its optimum value based on the amount of physical memory available and tuning values coded in the boot program (bootprgm). Reboot the system specifying **/etc/system** as the boot program. An optimized value of NBUF will be determined and utilized. *(boot/lboot/loadunix.c)* *(boot/lboot/error.c)*

Error Message	Description/Action *(Reference)*
driver: **dependent driver** *name* **is EXCLUDED**	*Driver* has dependencies on 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 the EXCLUDE line, remove it from the INCLUDE line if it exists there. *(/etc/system)* *(boot/lboot/loadunix.c)* *(boot/lboot/error.c)*
driver: **dependent driver** *name* **not available**	*Driver* has dependencies on driver *name*, but the object file for driver *name* is not found in boot directory. *Driver* will not be loaded. Place mkbooted 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. *(/etc/system)* *(boot/lboot/loadunix.c)* *(boot/lboot/error.c)*
driver: **device not equipped for dependent driver** *name*	*Driver* has dependencies on driver *name*, but hardware is 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 line if it exists there. *(/etc/system)* *(boot/lboot/loadunix.c)* *(boot/lboot/error.c)*

Error Message	Description/Action *(Reference)*
driver: **illegal character string initialization: zero assumed**	This is a warning message. Process was attempting to initialize a string variable for *driver* but found an illegal character string. Check master file for illegal character string initialization. *(/etc/master/filename)* *(boot/lboot/loadunix.c)* *(boot/lboot/error.c)*
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. It is illegal to EXCLUDE a required driver. Remove *name* from the EXCLUDE line of the system file and add it to the INCLUDE line, and then reboot. *(/etc/system)* *(boot/lboot/loadunix.c)* *(boot/lboot/error.c)*
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. *(boot/lboot/loadunix.c)* *(boot/lboot/error.c)*

Error Message	Description/Action *(Reference)*
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. Install device *name* in the correct slot, load the software, and reboot the system. *(boot/lboot/loadunix.c)* *(boot/lboot/error.c)*
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. *(boot/lboot/loadunix.c)* *(boot/lboot/error.c)*
I/O ERROR **id= block= count= jstat= erstat= xerstat=**	A disk read job failed. The message contains the buffer header pointer, disk block number, byte count, job status returned by disk subsystem, and failing status codes returned by the disk subsystem. Diagnose disk subsystem, make any necessary repairs, and then reboot. *(io/idfc.c)*

Error Message	Description/Action *(Reference)*
INCLUDE: *name*; **device not equipped**	Hardware not equipped for driver *name* to be included. This is a warning and 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. *(/etc/system)* *(boot/lboot/loadunix.c)* *(boot/lboot/error.c)*
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. *(/etc/system)* *(boot/lboot/loadunix.c)* *(boot/lboot/error.c)*
INCLUDE: *name*; **driver not found**	Driver *name* is marked to be included but is 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. *(/etc/system)* *(boot/lboot/loadunix.c)* *(boot/lboot/error.c)*

DGMON Error Messages

The Diagnostic Monitor (DGMON) program provides 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. Each error message is assigned a number that is prefaced by "DIAGNOSTIC MONITOR ERROR."

Number	Message	Description/Action *(Reference)*
2-00	**FILE SYSTEM IS INACCESSIBLE. CONTROL WILL RETURN TO MAINTENANCE CONTROL PROGRAM.**	The DGMON code cannot locate the file system offset of the root file system that contains the diagnostic files. Since the DGMON itself is part of the file system, very recent corruption of the system occurred. Retry request. If it fails again, a problem exists with the root file system where diagnostics reside. It may be necessary to restore the file system.
2-01	**UNKNOWN ID CODE (dev code) FOR DEVICE IN SLOT (slot *n*) NO DIAGNOSTIC TESTS RUN FOR THIS SLOT. CHECK EDT.**	An incomplete EDT may have devices with unknown ID codes. The DGMON will skip the device slot and proceed with any devices remaining to be tested. The device is not recognized because installation is incomplete or the device reports a bad ID code. If a message appears during the device installation, proceed with the installation. If not, retry request. If it fails again, check the look-up table in */dgn/edt_data* using **edittbl**, and check the device ID code using the "edt" firmware function. (***edittbl** manual page*)

Number	Message	Description/Action *(Reference)*
2-02	**CANNOT FIND FILE: (file name) DIAGNOSTIC REQUEST ABORTED.**	The DGMON cannot find the diagnostic file in the root file system. Retry request. If it fails again, the file is missing. Restore it from a backup.
2-03	**CANNOT LOAD FILE: (file name) DIAGNOSTIC REQUEST ABORTED**	The DGMON cannot load the diagnostic file from the root file system. Retry request. If it fails again, check the file. It may be zero length or have an invalid magic number.
2-04	**UNEXPECTED DIAGNOSTIC EXCEPTION. DIAGNOSTIC REQUEST ABORTED**	The processor detected an unexpected exception, probably due to attempts to address an invalid memory location or to parity errors. If the error flag has been enabled, this error message will contain the following additional information: **PC=0x***nnnnnnnn* **PSW=0x***nnnnnnnn* **FL1=0x***nnnnnnnn* **FL2=0x***nnnnnnnn* Retry request. If message reappears, check code and hardware. The Diagnostic Monitor command **errorinfo** enables/disables the error flag. See Chapter 3, "Processor Operations," for information on **errorinfo**.

Number	Message	Description/Action *(Reference)*
2-05	**UNEXPECTED DIAGNOSTIC INTERRUPT. DIAGNOSTIC REQUEST ABORTED.**	The processor detected an unexpected interrupt from any one of the components that can produce interrupts. If the error flag has been enabled, this error message will contain the following additional information: **PC=0x***nnnnnnnn* **PSW=0x***nnnnnnnn* **FL1=0x***nnnnnnnn* **FL2=0x***nnnnnnnn* **LEVEL=***nn* Retry request. If message reappears, check interrupt sources; for example, peripheral cards and disk subsystem. The Diagnostic Monitor command **errorinfo** enables/disables the error flag. See Chapter 3, "Processor Operations," for information on **errorinfo**.
2-06	**NON-EXISTENT UNIT: (device name) THE EQUIPPED UNIT TYPES ARE: (list of device names)**	The unit type requested is not in the EDT. A list of equipped units is provided. Retry request.
2-07	**INVALID UNIT NUMBER FOR (device name), THE EQUIPPED UNITS ARE: (list of device numbers) RETRY REQUEST**	The device number requested is not part of the EDT. The DGMON lists the equipped device numbers. Retry request.

Number	Message	Description/Action *(Reference)*
2-08	**(echo of input string) UNRECOGNIZABLE DIAGNOSTIC REQUEST. CHECK REQUEST SYNTAX AND RE-ENTER**	The string is echoed (shifted to uppercase). **H**(elp) command will list available diagnostic commands and syntax. Retry request. Check for possible nonprinting characters that some terminals may send to the system board (^s for example).
2-09	**INVALID REPEAT VALUE RE-ENTER REQUEST USING VALUE BETWEEN 1 AND 65536**	Repeat value is out of range. Retry request with an in-range value.
2-10	**INVALID PHASE(S) REQUESTED. CHECK REQUESTED PHASE TABLE AND RETRY.**	Use the **L**(ist) command to list the diagnostic phase table for the device to be tested. The **L**(ist) command appears in the diagnostic Help Menu. Retry request.
2-11	**REDUNDANT DIAGNOSTIC REQUEST OPTION. RE-ENTER REQUEST**	The **DGMON** checks for multiple definitions of options, such as repeat and phase range. At most, one of each is permitted. Retry request.
2-12	**SOAK AND UCL ARE INCOMPATIBLE DIAGNOSTIC OPTIONS. RE-ENTER REQUEST, OMITTING ONE.**	**SOAK** and **UCL** may not be combined for the same diagnostic request. Retry request.

Number	Message	Description/Action *(Reference)*
2-13	**UNIT OR UNIT TYPE NEEDED FOR PHASE OPTION REQUEST. RE-ENTER REQUEST.**	You must specify the device type if a special range of phases is desired. Retry request.
2-14	**USE UNIT TYPE ONLY FOR PHASE DISPLAY REQUEST. RE-ENTER REQUEST.**	The L(ist) command requires a device name and a device name only. Retry request.

UNIX System Error Messages

The UNIX system and kernel error messages are divided into the following three classes of severity: **NOTICE, WARNING,** and **PANIC**. The class of severity is displayed as the first part of each error message.

The error messages are listed alphabetically for each severity class. If you cannot find the exact message, look for a string variable (*str*) or a number variable (*n*) in the message. These variables may change the alphabetical placement of the message. A brief description of each severity class is given before the error message descriptions.

The referenced files found in the description/action column give additional information about the message. In some cases the complete path name of the file is not provided; for example, the complete path name for the reference io/if.c is /usr/src/uts/3b2/io/if.c. Referenced files that do not give the complete path name will be found under /usr/src/uts/3b2. These files, however, are only available if the Kernel Source option or Command Source option is installed. If Kernel Source or Command Source are not installed, the /usr/src directory is empty.

Some of the actions refer to the **sysdump** command. This command is covered in Procedure 3, "Processor Operations Procedures," and Chapter 3, "Processor Operations."

NOTICE Prefaced Messages

NOTICE error messages provide system status information that can, at times, help anticipate problems before they occur.

Error Message (Prefaced by NOTICE)	Description/Action *(Reference)*
bad block on floppy drive, slice *n*	An out-of-range block number was specified. Run **fsck** on the file system. *(io/if.c)* *(os/alloc.c)*
bad block on integral hard disk drive *n*, **partition** *n*	An out-of-range block number was specified. Take the system to the single-user mode, and run **fsck** on the file system. *(io/id.c)* *(os/alloc.c)*
bad count on floppy drive, slice *n*	A bad count in the super block. Run **fsck** on the file system. *(io/if.c)* *(os/alloc.c)*
bad count on integral hard disk drive *n*, **partition** *n*	A bad count in the super block. Take the system to single-user mode, and run **fsck** on the file system. *(io/id.c)* *(os/alloc.c)*

Error Message (Prefaced by NOTICE)	Description/Action *(Reference)*
Bad free count on floppy drive, slice *n*	The free list count is inconsistent. Run **fsck** on the file system. *(io/if.c)* *(os/alloc.c)*
Bad free count on integral hard disk drive *n*, partition *n*	The free list count is inconsistent. Take the system to the single-user mode and run **fsck** on the file system. *(io/id.c)* *(os/alloc.c)*
Changing console baud	Displayed when changing console baud via the **stty** command. When displayed, the software is updating the firmware baud rate saved in NVRAM. Therefore, future reboots of the system will retain the new baud rate. No action. *(stty man page)*
/dev/swap doesn't match swapdev; changing it on fs	The system was booted from a new device for the very first time. This is an advisory message and may be ignored. None. *(os/main.c)*

Error Message (Prefaced by NOTICE)	Description/Action *(Reference)*
File table overflow	The system file access table has overflowed. The corresponding code is not in the kernel, and sometimes the code is not displayed or seen. Increase the number of files and reboot the system. *(os/fio.c)* *(/etc/master.d/kernel)*
Floppy Access Error: Consult the Error Message Section of the System Administration Utilities Guide	This message could be caused by the following conditions: 1. A floppy disk is not in the floppy disk drive, the drive door is not closed, or the drive is not up to speed. 2. The floppy requires reformatting or its file system runs off the end of the floppy. In the latter case, the file system should be reconfigured so it does not extend beyond the end of the floppy disk. 3. The write-protect clip is in place on the floppy disk. This occurs when writes are done to a write-protected floppy unless the file system is mounted with the **-r** option (for read only). 4. The floppy disk being used is defective (bad). Make sure the floppy disk is in the drive and the door is closed. Remove write protect tab or mount file system with **r** (read only) option. If reformatting fails the verify pass, replace the floppy disk. *(io/if.c)*

Error Message (Prefaced by NOTICE)	Description/Action *(Reference)*
no space on floppy drive, slice *n*	The involved partition on the floppy disk is out of space. Copy fewer files to the partition or run **mkfs** to specify more i-nodes. Clean up the affected file system indicated by the partition number. If more free blocks are also needed, repartition the file system. Both **mkfs** and repartitioning destroy the data on the floppy disk. *(io/if.c)* *(os/alloc.c)*
Out of inodes on floppy drive, slice *n*	There are no free i-nodes in the involved partition. Copy fewer files to the partition or run **mkfs** to specify more i-nodes. Clean up the affected file system indicated by the partition number. If more free blocks are also needed, repartition the file system. Both **mkfs** and repartitioning destroy the data on the floppy disk. *(io/if.c)* *(os/alloc.c)*
page read error on floppy drive, slice *n*	An I/O error has occurred while trying to read in a page from a file. Retry command. If this fails, replace floppy diskette. *(io/if.c)* *(os/fault.c)*

Error Message (Prefaced by NOTICE)	Description/Action *(Reference)*
shmctl - couldn't lock *n* pages into memory	Could not lock a shared memory segment into memory because memory was over committed. Try again. *(io/shm.c)*
***str* - Insufficient memory to *str n* pages - system call failed**	A system call has failed due to insufficient memory. Try again. *(os/prf.c)*
swap space running out: needed *n* blocks	The system had to remove saved text sections of processes which were swapped out to provide enough swap space to swap out a new process. If this occurs frequently, run fewer simultaneous processes or expand system swap space. *(os/text.c)*
useracc - couldn't lock page	Insufficient space is available to lock a user data page into memory making the system unable to service a read or write system call to a raw device. Reduce the system load, reduce the size of raw I/O buffer in the user program, or add more memory to the system. *(os/probe.c)*

WARNING Prefaced Messages

WARNING error messages indicate that the UNIX system may stop functioning if corrective action is not taken.

Error Message (Prefaced by WARNING)	Description/Action *(Reference)*
floppy disk timeout: request flushed	This error message occurs when the floppy disk drive door is not shut, no floppy disk is in the drive, or the drive has gone off-line. Check cable connections and the insertion of the floppy disk. *(io/if.c)*
hard disk: Bad sanity word in VTOC on drive *n*.	The Volume Table Of Contents (VTOC) is either bad or the wrong version. Restore the hard disk from the restore floppy disks selecting the Full Restore option. *(io/id.c)*
hard disk: cannot access sector *n*, **head** *n*, **cylinder** *n*, **on drive** *n*	This message should only appear when a bad disk block is found. The hard disk error logger should report that this disk block is logged. To map this bad block, the user must be in single-user mode and execute the **hdefix** command. *(io/id.c)*

Error Message (Prefaced by WARNING)	Description/Action *(Reference)*
hard disk: Cannot read the VTOC on drive *n*	The hard disk must be restored. Restore the hard disk from the restore floppy disks. If trouble persists, replace drive. *(io/id.c)*
hard disk: partition *n* **on drive** *n* **is marked read only**	The disk partition being accessed is marked read only, and the disk request for that partition is write. If you wish to write it, the permissions in the VTOC must be changed. Use the **fmthard** command. *(io/id.c)*
HDE queue full, following report not logged	The hard disk error logger queue is full and can not receive any more entries. Log that the message occurred. Save the error message output and manually add the reports to the disk error log. See the "Bad Block Handling Feature" in Chapter 4, "Disk/Tape Management." *(io/hde.c)*
hdeeqd: major(ddev) = *n* **(>=cdevcnt)**	The hard disk error logger found a bad disk block and logged it. Log the error message and reboot the system. Make sure the major device number passed by the driver is valid. *(io/hde.c)*

Error Message (Prefaced by WARNING)	Description/Action *(Reference)*
iget - inode table overflow	The i-node table ran out of free slots. There were too many open or in-use files at one time. Run fewer applications at the same time, reduce the number of simultaneous users, or increase the number of i-node table entries. *(os/iget.c)* *(/etc/master.d/kernel)*
inode table overflow	The i-node table is full, and the machine has to wait for an entry to be freed. If persistent, reconfigure system with a larger i-node table (NINODE). *(os/iget.c)*
Lost date and time	Successive reads of the time-of-day clock hardware have failed. Set the clock using **sysadm datetime**. If the problem persists: 1. Run system diagnostics or 2. Reset the NVRAM using the floppy key (firmware password is defaulted to *mcp*). 3. Check the battery. 4. Run diagnostics on the time-of-day clock. There may be a problem with the clock hardware. 5. Call your AT&T Service Representative or authorized dealer. *(os/todc.c)*

Error Message (Prefaced by WARNING)	Description/Action *(Reference)*
mfree map overflow *n*. **Lost** *n* **items at** *n*	The free memory allocation map is full, and a request to free more memory has failed since an empty slot could not be located, or memory is fragmented so the piece to be freed does not connect with an existing map entry. If persistent, reconfigure system with a larger core map size (CMAPSIZ). *(os/malloc.c)*
No swap space for exec args	Swap space is fully utilized. If this problem occurs frequently, either reduce the number of simultaneous processes or increase the swap area. *(os/sys1.c)*
Null m_mount in iget mp: *n*	Search of mount table found null i-node pointer reference. If the error persists, reboot the UNIX System. *(os/iget.c)*
out of swap space: needed *n* **blocks**	A process was left in memory because there was no room to swap it out. If room becomes available, it will be swapped out. This problem can be avoided by running fewer processes or expanding the swap area. *(os/text.c)*

Error Message (Prefaced by WARNING)	Description/Action *(Reference)*
out of text	A request to execute a new process has failed due to a full process text table. If persistent, reconfigure system with increased text table limits (NTEXT). *(os/tex.c)*
PORTS: EXPRESS QUEUE OVERLOAD: One entry lost	A PORTS queue entry may have been lost or the PORTS board may be insane. Log that the error message occurred. Reboot the system. *(io/lla_ppc.c)*
PORTS: FAULT - opcode= *n*, **board** *n*, **subdev** = *n*, **bytecnt** = *n*, **buff address** = *n*	An invalid PORTS opcode was encountered or the PORTS board may be insane. "Pump" the associated PORTS board. If a problem still exists, reboot the system. *(io/ppc.c)*
PORTS: QFAULT - opcode= *n*, **board** *n*, **subdev** = *n*, **bytecnt** = *n*, **buff address** = *n*	The PORTS job queue is invalid. The PORTS board may be insane. "Pump" the associated PORTS board. If a problem still exists, reboot the system. *(io/ppc.c)*

Error Message (Prefaced by WARNING)	Description/Action *(Reference)*
PORTS: SYSGEN failure on board *n*	The ports board or the firmware has gone insane. Log the error message, and reboot the system. *(io/ppc.c)*
PORTS: timeout on drain board (*n*), port (*n*)	The ports board or the firmware has gone insane. Log the error message, and reboot the system. *(io/ppc.c)*
PORTS: unknown completion code: *n*	This is probably a hardware problem. Log the error message, and reboot the system. *(io/ppc.c)*
Region table overflow	Each text, data, stack, and shmem process segment requires one entry in the region table. Too many processes cause the table to overflow. The system call that tried for another region failed. Reduce the number of active processes or increase the number of region table entries (NREGION). *(os/region.c)* *(/etc/master.d/kernel)*

Error Message (Prefaced by WARNING)	Description/Action *(Reference)*
str **ECC hard disk error: maj/min** = *n*/*n*	This message is generated as a result of the disk hardware detecting a checksum error on a block of data accessed from the disk media. This message is typically followed by a message from the hard disk error logger indicating that the bad block has been logged. Empirical evidence has shown that this problem could be caused by one of the following conditions: ■ Unmapped defects on the disk media ■ A power failure during a write operation to a particular sector on the disk media ■ Hardware faults. In the case of media defects or a power failure, the bad block should be mapped using the **hdefix** command while in single-user mode. Hardware faults are usually characterized by persistent occurrences of this error message. Check for obvious problems such as loose or faulty cables. *(io/hde.c)*
str **on bad dev** *n***(8)**	This message appears if the file system runs out of space. Clean up the file system. Delete files no longer required or move them to a floppy. Reboot the system. *(os/prf.c)*
too few HDE equipped disk slots Bad block handling skipped for maj/min = *n*/*n*	If more disks are added than the system allows, change the tunable parameter set by the HDE logger. Log the error message, and reboot the system. *(io/hde.c)*

PANIC Prefaced Messages

PANIC error messages indicate a problem so severe that the UNIX system must stop. The cause is usually a hardware problem or problem in the kernel. Any programs running when the PANIC occurs are lost and some file systems may be corrupted. The UNIX system does check for file system damage when it is restarted.

Following a PANIC, a system crash dump should be taken, before proceeding. If a power cycle is required to regain control of the machine, a crash dump is not required. Instead, **/etc/errdump** should be executed to determine the cause of the PANIC.

As in most sophisticated computer systems, "crashes" (PANICs) will occasionally occur and should not cause much concern. If a particular PANIC occurs repeatedly (or predictably), however, you should seek help.

Error Message (Prefaced by PANIC)	Description/Action *(Reference)*
cannot mount root	An Input/Output (I/O) error occurred while the system was trying to mount the root file system. The error is either hardware related or the root file system is improperly specified, that is, a nonequipped device. After the panic completes, take the system to the firmware mode, and use the **sysdump** command. Reboot the system. If the reboot fails, do a partial restore. See Procedure 3, "Processor Operations Procedures." *(os/sys3.c)*

Error Message (Prefaced by PANIC)	Description/Action *(Reference)*
i/o error in swap	An access error occurred on the swap device. The device controller could cause the error requiring hardware service. Check the hard disk error log. After the panic completes, take the system to the firmware mode, and use the **sysdump** command. Reboot the system. *(io/bio.c)*
microbus timeout interrupt 0x*nnnnnnnn*	Microbus timed out. Check boards in microbus and reseat them. If the problem persists, the hardware may be bad.
Multiple-bit error interrupt at 0x*nnnnnnnn*	A multiple-bit memory error occurred. If this occurs repeatedly, the hardware requires servicing. This was possibly caused by dirty memory card connectors. After the panic completes, take the system down to firmware mode. Run the system board diagnostic phases for the Random Access Memory (RAM) cards. If diagnostics fail, the RAM cards need servicing.

Appendix D: Job Accounting

General

The UNIX system Job Accounting of the collects data on system usage by user and by process. In particular, Job Accounting records connect sessions, monitors disk usage, and charges fees to specific logins. To help you access this data, a number of C language programs and shell scripts are provided that reduce this accounting data into summary files and reports.

This chapter describes how Job Accounting operates. Specifically, the chapter describes the numerous files and programs that figure prominently in the Job Accounting system. The chapter also provides samples of the various reports that Job Accounting generates.

Overview of Job Accounting

UNIX system Job Accounting, once installed, runs mostly on its own.

Here is a general overview of how Job Accounting works:

- Between system start-up and shutdown, raw data about system use (such as logins, processes run, and data storage) is collected in accounting files.
- Once a day, **cron** initiates the **runacct** program, which processes the various accounting files and produces both cumulative summary files and daily accounting reports. The daily reports can be printed by executing the **prdaily** program.
- The cumulative summary files generated by **runacct** can be processed and printed 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.

The two main elements of Job Accounting are the files that hold the raw data on system usage and the C programs and shell scripts that initiate the accounting and process the data for reports and billing.

Setting Up the Accounting System

The automation of the operation of this accounting system has been done during the installation of the package. During installation, an entry of *S22acct* is made in the directory */etc/rc2.d* and an entry of *S22acct* is entered in the */etc/rc0.d* directory. This will cause the accounting package to be automatically initiated at boot time and automatically shut off during shutdown.

Also, entries have been added to the */usr/spool/cron/crontabs/adm* directory which will cause the routine generation of reports as well as maintenance to take place for the accounting package. These entries need only be uncommented and cron will execute the commands as indicated.

These entries added are the following:

```
#0    4   *   *   1-6   /usr/lib/acct/runacct 2>/usr/adm/acct/nite/fd2log
#0    *   *   *   *     /usr/lib/acct/ckpacct
#15   5   1   *   1-6   /usr/lib/acct/monacct
```

A *.profile* should be created in */usr/adm* (i.e., */usr/adm/.profile*) with the following *PATH* shell variable set:

PATH= /usr/lib/acct:/bin:/usr/bin; export PATH

Job Accounting Files

The */usr/adm* directory structure (see Figure D-1) contains the active data collection files and is owned by the *adm* login (currently user ID of 4).

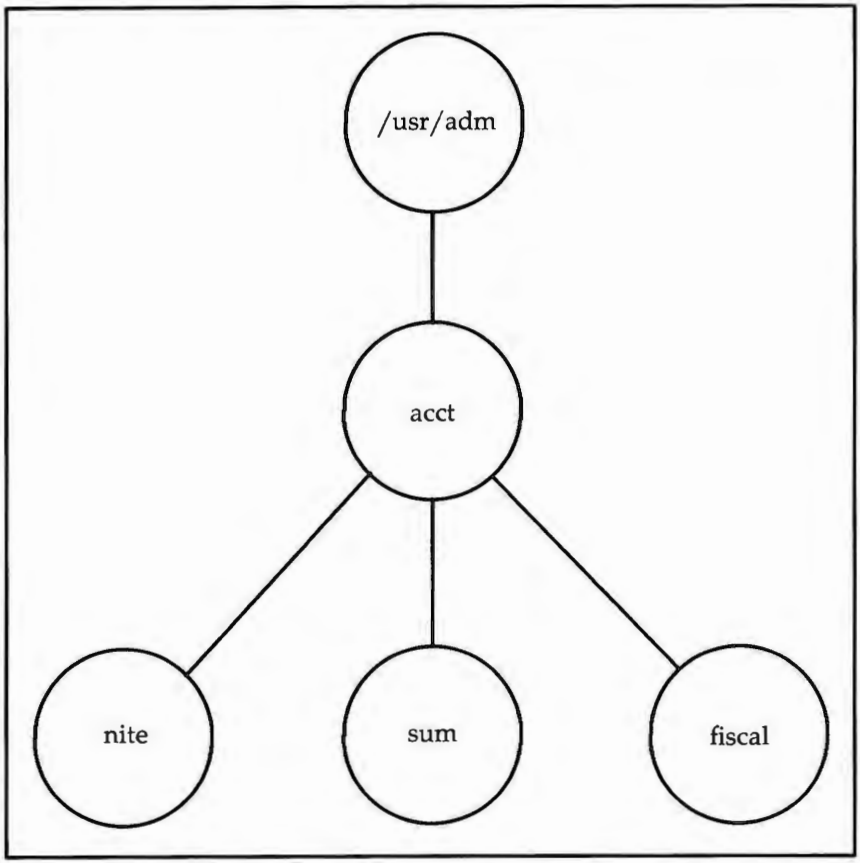


Figure D-1: Directory Structure of the adm Login

A brief description of the files found in the */usr/adm* directory is as follows:

- *diskdiag*
  diagnostic output during the execution of disk accounting programs
- *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 */usr/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. Following is brief summary of the files in the */usr/adm/acct/nite* directory:

- *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
- *daycms*
  ASCII daily command summary used by **prdaily**
- *daytacct*
  total accounting records for 1 day in *tacct.h* format

- *disktacct*
  disk accounting records in *tacct.h* format, created by **dodisk** procedure
- *fd2log*
  diagnostic output during execution of **runacct** (see **cron** entry)
- *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 **acctcon1**
- *logMMDD*
  same as **log** after **runacct** detects an error
- *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
- *wtmperrorMMDD*
  same as **wtmperror** after **runacct** detects an error
- *wtmp.MMDD*
  previous day's *wtmp* file.

The *sum* directory contains the cumulative summary files updated by **runacct** and used by **monacct**. Following is brief summary of the files in the */usr/adm/acct/sum* directory:

- *cms*
  total command summary file for current fiscal in internal summary format

- *cmsprev*
  command summary file without latest update
- *daycms*
  command summary file for yesterday in internal summary format
- *loginlog*
  created by **lastlogin**
- *pacct.MMDD*
  concatenated version of all pacct files for *MMDD*, removed after reboot by **remove** procedure
- *rprtMMDD*
  saved output of **prdaily** program
- *tacct*
  cumulative total accounting file for current fiscal
- *tacctprev*
  same as *tacct* without latest update
- *tacctMMDD*
  total accounting file for *MMDD*
- *wtmp.MMDD*
  saved copy of *wtmp* file for *MMDD*, removed after reboot by *remove* procedure.

The *fiscal* directory contains periodic summary files created by **monacct**. Following is brief description of the files in the */usr/adm/acct/fiscal* directory:

- *cms?*
  total command summary file for fiscal *?* in internal summary format
- *fiscrpt?*
  report similar to **prdaily** for fiscal *?*
- *tacct?*
  total accounting file for fiscal *?*.

Job Accounting Programs

All the C Language programs and shell scripts necessary to run the accounting system are in the */usr/lib/acct* directory. These programs, which are owned by *bin*, perform various functions. For example, */usr/lib/acct/startup* helps initiate the accounting process when the system enters the multiuser 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.

Daily Job Accounting

Here is a step-by-step outline of how UNIX system Job Accounting works:

1. When the UNIX system is switched into multiuser mode, the */usr/lib/acct/startup* program is executed. The **startup** program executes several other programs which initiate Job Accounting:

   - The **acctwtmp** program adds a "boot" record to */etc/wtmp*. In this record the system name is indicated as the login name in the *wtmp* record. Figure D-2 presents a summary of how the raw accounting data is gathered and where it is stored.
   - The **turnacct** program initiates process accounting. Process accounting begins when the **turnacct** program is executed with the **on** option. **Turnacct on** executes the **accton** program with the argument */usr/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 */etc/wtmp*. Any date changes (using **changedate**) are also written to */etc/wtmp*. Date changes, reboots, and shutdowns (via **acctwtmp**) are also recorded in the */etc/wtmp*.
3. When a process terminates, the kernel writes one record per process in the */usr/adm/pacct?* file in the form of *acct.h*.
4. The disk utilization programs **acctdusg** and **diskusg** break down disk usage by login.
5. Every hour, **cron** executes the **ckpacct** program to check the size of */usr/adm/pacct*. If the file grows past 500 blocks (default), **turnacct switch** is executed. The advantage of having several smaller *pacct* files becomes apparent when trying to restart **runacct** after a failure of processing these records.
6. If the system is shut down using **shutdown**, the **shutacct** program is executed. The **shutacct** program writes a reason record into */etc/wtmp* and turns off process accounting.
7. If a user requests a service such as a file restore, the **chargefee** program can be used to bill users (specific logins). It adds a record to */usr/adm/fee* which is picked up and processed by the next execution of **runacct** and merged into the total accounting records.

8. **Runacct** is executed via **cron** each night. It processes the active accounting files, */usr/adm/pacct*, */etc/wtmp*, */usr/adm/acct/nite/disktacct*, and */usr/adm/fee*. It also produces command summaries and usage summaries by login.

9. The **/usr/lib/acct/prdaily** program should be executed on a daily basis in order to print the daily accounting information collected by **runacct**.

10. The **monacct** program should be executed on a monthly basis (or whenever you believe appropriate, such as a 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.

File	Information	Written By	Format
/etc/wtmp	connect sessions	login, init	ctmp.h
	date changes	changedate	
	reboots	acctwtmp	
	shutdowns	shutacct shell	
/usr/adm/pacct?	active processes	kernel (when process terminates)	acct.h
		turnacct switch creates new file when old one reaches 500 blocks.	
/usr/adm/fee	special charges	chargefee	

Figure D-2: Raw Accounting Data

The runacct Program

The main daily accounting shell procedure, **runacct**, is normally initiated via **cron** during nonprime 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** or for billing purposes.

Files Produced by runacct

The following files produced by **runacct** are of particular interest:

- *nite/lineuse*
  The **acctcon** program gathers data on terminal line usage and writes the data to */etc/wtmp*. The **runacct** shell script processes the data in */etc/wtmp* and writes it to */usr/adm/acct/nite/lineuse*. 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 previous 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*
  The **acctcms** program generates the data pertaining to the commands used during a given day. The **runnacct** program processes this information and writes it to */usr/adm/acct/sum/daycms*. It contains the daily command summary. The ASCII version of this file is *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*
  The **lastlogin** program generates data concerning when a login was last used. The **runacct** program processes this data and writes it to */usr/adm/acct/sum/loginlog*.

- *sum/rprtMMDD*
  Each execution of **runacct** saves a copy of the daily report that can be printed by **prdaily**.

The **runacct** shell script takes care not to damage files in the event of errors. A series of protection mechanisms are used that attempt to recognize an error, provide intelligent diagnostics, and terminate 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 *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. 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 */dev/console*, mail is sent to *root* and *adm*, locks are removed, diagnostic files are saved, and execution is terminated.

Re-entrant States of the runacct Script

In order to allow **runacct** to be restartable, processing is broken down into separate re-entrant 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 terminates. States are executed as follows:

- SETUP
  The command **turnacct switch** is executed. The process accounting files, */usr/adm/pacct?*, are moved to */usr/adm/Spacct?.MMDD*. The */etc/wtmp* file is moved to */usr/adm/acct/nite/wtmp.MMDD* with the current time added on the end.

- WTMPFIX
  The **wtmpfix** program checks the *wtmp* file in the *nite* directory for correctness; because some date changes will cause **acctcon1** to fail, so

wtmpfix attempts to adjust the time stamps in the *wtmp* file if a date change record appears.

- CONNECT1
  Connect session records are written to *ctmp* in the form of **ctmp.h**. The *lineuse* file is created, and the *reboots* file is created showing all of the boot records found in the *wtmp* file.
- CONNECT2
  Ctmp is converted to *ctacct.MMDD* which are the connect accounting records. (Accounting records are in **tacct.h** format.)
- PROCESS
  The **acctprc1** and **acctprc2** programs are used to convert the process accounting files, */usr/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
  On the day after the **dodisk** procedure runs, merge *disktacct* with *daytacct*.
- MERGETACCT
  Merge *daytacct* with *sum/tacct*, the cumulative total accounting file. Each day, *daytacct* is saved in *sum/tacctMMDD*, so that *sum/tacct* can be re-created in the event it becomes corrupted or lost.
- CMS
  Merge in today's command summary with the cumulative command summary file *sum/cms*. Produce ASCII and internal format command summary files.

- USEREXIT
  Any installation-dependent (local) accounting programs can be included here.
- CLEANUP
  Clean up temporary files, run **prdaily** and save its output in *sum/rprtMMDD*, remove the locks, then exit.

runacct Error Messages

The **runacct** procedure can fail for a variety of reasons; usually due to a system crash, */usr* running out of space, or a corrupted *wtmp* file. If the *activeMMDD* file exists, check the file 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.

ERROR: acctg already run for *date* : check /usr/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.

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

> Self-explanatory.

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

Wtmpfix detected a corrupted *wtmp* file. Use **fwtmp** to correct the corrupted file.

ERROR: connect acctg failed: check /usr/adm/acct/nite/log

The **acctcon1** program encountered a bad *wtmp* file. Use **fwtmp** to correct the bad file.

ERROR: Invalid state, check /usr/adm/acct/nite/active

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

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 filesave backup. However, certain files must be fixed in order 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 UNIX system is in multiuser mode, a set of date change records is written into */etc/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 **acctcon1** to fail. The following steps show how to patch up a *wtmp* file.

```
cd /usr/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. **Acctprc1** will not be able to determine which login owned a particular process, but it will be charged to the login that is first in the password file for that 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 quite important. Occasionally, mysterious *tacct* records will appear with negative numbers, duplicate user IDs, or a user ID of 65,535. First, check *sum/tacctprev* with **prtacct**. If it looks all right, the latest *sum/tacct.MMDD* should be patched up, then *sum/tacct* re-created. A simple patchup procedure would be the following:

```
cd /usr/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
```

Remember that the **monacct** procedure removes all the *tacct.MMDD* files; therefore, *sum/tacct* can be re-created by merging these files together.

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. For example:

To start **runacct**:

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

To restart **runacct**:

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

To restart **runacct** at a specific state:

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

Billing Users

The monthly accounting stream **monacct** summarizes data on system usage by user. This information can be used to create bills. The **chargefee** program creates a related file, *fees*, which stores charges information for special services such as file restores.

To register special fees, the operator needs to enter the command

chargefee *login-id amount*

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

Setting Up Nonprime Time Discounts

UNIX system Job Accounting provides facilities to give users a discount for nonprime time system use. For this to work, you must inform the accounting system of the dates of holidays and the hours that are considered nonprime time. To do this, you must edit the */usr/lib/acct/holidays* file that contains the prime/nonprime 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 1988, prime time at 9:00 a.m., and nonprime time at 4:30 p.m., the following entry would be appropriate:

  1988 0900 1630

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

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

  day-of-year Month Day Description of Holiday

  The day-of-year field is a number in the range of 1 through 366 (January 1 is "1"; December 31 in a leap-year is "366") indicating the day for the corresponding holiday (leading white space is ignored). The other three fields are actually commentary and are not currently used by other programs. Therefore, you can use any format to indicate the month and day and any meaningful description of the holiday. See Figure D-3 for an example of a holiday list.

Day of Year	Month	Day	Holiday Name
01	01	01	New Year's Day
15	01	15	Martin Luther King
Varies	05	Varies	Memorial Day
185(186 Leap Year)	07	04	Independence Day
Varies	09	Varies	Labor Day
315(316 Leap Year)	11	11	Veteran's Day
Varies	11	Varies	Thanksgiving
359(360 Leap Year)	12	25	Christmas

Figure D-3: Holiday List

Daily Accounting Reports

The **runacct** shell script generates four basic reports upon each invocation. They cover the areas of connect accounting, usage by person 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 following:

- **Daily Report** shows line utilization by tty number.
- **Daily Usage Report** indicates usage of system resources by users (listed in UID order).
- **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**.
- **Last Login** shows the last time each user logged in (arranged in chronological order).

The following paragraphs describe the reports and the meanings of their tabulated data.

Daily Report

This report gives information about each terminal driver.

The **from/to** tells you the time period reflected in the report. The times are 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 */etc/wtmp* by the **acctwtmp** program [see **acct**(1M) in the *AT&T 3B2 Computer UNIX System V 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 multiuser state (able to be accessed through the terminal lines). The columns are as follows:

- LINE
  The terminal line or access port.
- MINUTES
  The total number of minutes the 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**(1) session.
- # ON
  This column does not have much meaning anymore. It used to give the number of times that the port was used to log a user on; but since **login**(1) 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 the **getty**(1M) is first invoked when the system is brought to multiuser 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, */etc/wtmp* should be monitored as this is the file that the connect accounting is geared from. If it grows rapidly, execute **acctcon1** to see which tty line is the noisiest. If the interrupting is occurring at a furious rate, general system performance will be affected. Refer to Figure D-4 for a sample daily report.

```
Jan 18 16:43 1988  DAILY REPORT FOR abcde Page 1

from Sat Jan 16 04:40:56 1988
to   Mon Jan 18 04:30:36 1988
1       acctg off
1       run-level 1
1       run-level 2
1       acctg on
1       runacct
1       acctcon1

TOTAL DURATION IS 2870 MINUTES
LINE      MINUTES    PERCENT    # SESS     # ON     # OFF
tty42     19         1          97         97       197
tty65     18         1          94         94       191
tty01     556        19         50         50       114
tty46     9          0          48         48       100
tty02     315        11         4          4        12
tty03     52         2          1          1        8
tty04     1          0          1          1        5
console   0          0          0          0        2
tty05     0          0          0          0        4
tty06     0          0          0          0        4
tty07     0          0          0          0        4
tty08     0          0          0          0        4
     .
     .
     .
TOTALS    971        --         295        295      769
```

Figure D-4: Sample Daily Report

Daily Usage Report

This report (see Figure D-5 for a sample) gives a breakdown of system resource utilization by user. Its data consists of the following:

- UID
  The user ID.
- LOGIN NAME
  The login name of the user; there can be more than one login name for a single-user ID. This identifies which one.

- CPU (MINS)
  CPU represents the amount of time the user's process used the central processing unit. This category is broken down into PRIME and NPRIME (nonprime) utilization. The accounting system's idea of this breakdown is located in the */usr/lib/acct/holidays* file. As delivered, prime time is defined to be 0900 through 1700 hours.

- 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 (MINS)
  This identifies "Real Time" used. What this column really identifies is the amount of time that a user was logged into the system. If this time is rather high and the column "# OF PROCS" is low, this user is what is called 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 (*tacct.h*) and shows up in this column. This disk accounting is accomplished by the program **acctdusg**.

- # 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 problems.

- # OF SESS
  This is how many times the user logged onto the system.

- # DISK SAMPLES
  This indicates how many times the disk accounting has been 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, etc.

```
Jan 16 16:43 1988  DAILY USAGE REPORT FOR abcde Page 1

UID   NAME     PRIME  NPRIME  PRIME  NPRIME  PRIME  NPRIME  BLOCKS  PROCS  SESS  SAMPLES
0     TOTAL    0      259     0      19223   0      971     523973  18434  295   138     0
0     root     0      78      0      4367    0      0       31767   2172   0     1       0
1     daemon   0      0       0      5       0      0       402     23     0     1       0
2     bin      0      0       0      0       0      0       159289  0      0     1       0
3     sys      0      1       0      15      0      0       334     199    0     1       0
4     adm      0      11      0      638     0      0       6582    931    0     1       0
5     uucp     0      8       0      602     0      0       5741    917    0     1       0
6     nuucp    0      72      0      5963    0      287     148     2530   283   1       0
30    dgn      0      0       0      0       0      0       6812    0      0     1       0
40    usors    0      0       0      9       0      0       1048    93     0     1       0
55    3bnet    0      0       0      0       0      0       1966    0      0     1       0
67    s        0      0       0      0       0      0       33618   0      0     1       0
68    rje      0      19      0      503     0      0       702     1263   0     1       0
71    lp       0      4       0      99      0      0       710     816    0     1       0
80    polaris  0      0       0      0       0      0       4662    0      0     1       0
90    emda     0      0       0      0       0      0       39      0      0     1       0
91    emna     0      0       0      0       0      0       319     0      0     1       0
98    notes    0      0       0      0       0      0       3594    0      0     1       0
99    news     0      20      0      2305    0      0       40750   961    0     1       0
101   jlc      0      1       0      97      0      20      11019   131    1     1       0
102   secl     0      0       0      0       0      0       45      0      0     1       0
103   ras      0      0       0      1       0      0       80488   95     0     1       0
        .
        .
        .
```

Figure D-5: Sample Daily Usage Report

Daily Command Summary

The Daily Command Summary report shows the system resource utilization by command. See Figure D-6 for a sample report. With this report, you can identify the heaviest used commands and, based on how those commands use system resources, gain insight on how to best tune the system. The Daily Command and Monthly reports are virtually the same except that the Daily Command Summary only reports 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.

- COMMAND NAME
  The name of the command. Unfortunately, all shell procedures are lumped together under the name **sh** since only object modules are reported by the process accounting system. The administrator should 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.
- NUMBER CMDS
  The total number of invocations of this particular command.
- TOTAL KCOREMIN
  The total cumulative measurement of the amount of kilobyte segments of memory used by a process per minute of run time.
- TOTAL CPU-MIN
  The total processing time this program has accumulated.
- TOTAL REAL-MIN
  Total real-time (wall-clock) minutes this program has accumulated. This total is the actual "waited for" time as opposed to kicking off a process in the background.
- 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, is a total count of the number of characters pushed around by the **read**(2) and **write**(2) system calls.
- BLOCKS READ
  A total count of the physical block reads and writes that a process performed.

```
Jan 18 04:40 1988  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       18434  18937.22     259.17    57560.98     73.07      0.01      0.00  449727168     819491

rnews          677   4234.03      43.91      133.97     96.43      0.06      0.33  251729856     336134
ad_d            11   2576.87      11.95      285.83    215.64      1.09      0.04    1896888       2947
expire           2   2161.66      16.61       35.25    130.12      8.31      0.47   12218368      22144
uucico         378   2025.56      29.37    12285.64     68.97      0.08      0.00   27336352      55954
find            15   1003.68      23.52       70.06     42.67      1.57      0.34    6964294      87961
fgrep          237   1001.80       8.35       11.02    120.03      0.04      0.76    2940159       5911
comp            43    830.26       3.87        5.98    214.59      0.09      0.65     776626       5367
du              79    717.50      17.89       29.64     40.11      0.23      0.60    2275238      31352
nibackup         1    512.17      14.13     2843.31     36.25     14.13      0.00    5459968         17
as              33    456.87       2.12        3.60    215.07      0.06      0.59    2962296       4055
rjelxmit       390    396.91      15.29     1190.88     25.95      0.04      0.01      34272       1748
acctcms          4    336.85       3.74        5.98     90.08      0.93      0.63    1038960        938
sh            3306    298.87      12.20    12407.39     24.51      0.00      0.00    2281868      27998
ar              18    242.14       3.78       19.11     64.06      0.21      0.20   45581312      40725
sed           1423    210.78       4.45       17.61     47.39      0.00      0.25    4267362       9022
acctprcl         3    159.10       3.06        6.11     52.06      1.02      0.50    1863744       1025
ld              14    119.26       0.64        1.46    185.62      0.05      0.44    2374592       2480
optim           33    101.79       1.20        1.78     85.10      0.04      0.67     655268       1706
 .
 .
 .
```

Figure D-6: Sample Daily Command Summary

Monthly Accounting Reports

The monthly accounting stream, **monacct**, produces monthly summary reports similar to those produced daily. See Figure D-7 for a sample report. The **monacct** program also summarizes the accounting information into the files in the */usr/adm/acct/fiscal* directory. This information can be used to generate monthly billing.

Note: To generate a monthly billing, many UNIX system installations customize the accounting process with their own shell scripts.

```
Jan 18 04:40 1988  MONTHLY TOTAL 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  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
il0          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
    .
    .
    .
```

Figure D-7: Sample Monthly Total Command Summary

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 which could be archived and deleted. See Figure D-8 for a sample report.

```
Feb 13 04:40 1988  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
          .
          .
          .
```

Figure D-8: Sample Last Login

Summary

UNIX system Job Accounting was designed from a UNIX system administrator's point of view. Every possible precaution has been taken to ensure that the system will run smoothly and without error. It is important to become familiar with the C programs and shell procedures. The manual pages should be studied, and it is advisable to keep a printed copy of the shell procedures handy. The accounting system should be easy to maintain, provide valuable information for the administrator, and provide accurate breakdowns of the usage of system resources for billing purposes. The accounting system can be used to tune your computer by checking which processes are used frequently and moving those processes to a separate disk from the users.

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. The **boot block** is 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.

bridge controller A target controller which is physically separate from the peripheral [Input/Output (I/O)] device. A bridge controller must be mounted separately from the peripheral device and can generally control from one to eight peripheral devices.

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 *mounted* an *advertised 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 cannot handle, it takes itself out of service, or crashes. Such conditions occur when the system cannot 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.

defect A defective area of storage media where data cannot reliably be stored.

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 *i-node 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, floppy disks, or tapes while they are in use.

dump A copy of the *core image* of the operating system.

embedded controller A target controller which is physically a part of the peripheral device. An embedded controller can generally control only one peripheral device. It exists within the industry standard form factor.

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 *i-node* 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*.

file

1. A *path name*
2. The last component name in a path name.

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 Nonvolatile Random Access Memory (NVRAM), which permanently holds a few, special programs.

floppy disk A magnetic storage medium which is smaller and more flexible than a hard *disk.*

floppy key A copy of the default *firmware* password for a 3B2 computer on a *floppy disk.* 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. The *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.

host adapter — Interface between the host computer system bus and the SCSI bus.

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 *i-nodes* of the file system; cf. *i-node number*.

indirect blocks — Data blocks that are not directly referenced by a *i-node* (because the file is larger than 10 1024-byte blocks); the i-node has 3 *addresses* that indirectly reference (by a cascade of pointers) some 2,114,114 data blocks (an extremely large

potential *file* size). The i-node 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*.

initiator — The target controller or Host Adapter which initiates a SCSI bus operation.

i-node — An element of a *file system* [1]; an i-node specifies all properties of a particular *file* [2] and locates the file's contents, if any.

i-node number, i-number — The position of an *i-node* 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 Inter–Process 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 *i-node*; either linked [1] or a *symbolic link*.

link count The number of *directory entries* that pertain to an *i-node*; 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.

logical unit A peripheral device connected to a target controller.

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,* and 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, and 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 a disk.

path, path name — A chain of names designating a *file*; a **relative path name** 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 path name** 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, if in a directory); permissions are granted separately to *owner*, *group*, and *others*. **Permission bit** is a permission, so called because each permission is encoded into one bit in an *i-node*.

peripheral device — Any I/O device which can be integrated by a target controller to the SCSI bus (for example, tape drive, disk drive, optical disk drive, line printer, or network interface).

physical block — A unit of data as it is actually stored and manipulated; the UNIX system handles data in 1024-byte physical blocks.

physical memory — 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 *mounted* 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 rewinding 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.

SCSI device A computer Host Adapter or target controller or an intelligent peripheral that can be attached to the SCSI bus.

search path In the *shell*, a list of *path names* of *directories* that determines the meaning of a *command;* the command name is prefixed with members of the search path in turn until a path name 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 (1) the **text segment**, occupied by executable code, (2) the **data segment**, occupied by *static* data that is specifically initialized, (3) the **bss segment**, occupied by static data that is initialed by default to zero values, and (4) the **stack segment**, occupied by *automatic* data, see *stack;* sometimes (2), (3), and (4) 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 *i-node* that designates a *device*, further categorized as either (1) a **block special file** describing a *block device*, or (2) 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 under *standard output*.

standard input — The second of three files described 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*.

super user — *Userid* 0, which can access any *file* regardless of *permissions* and can perform certain privileged *system calls*; for example, 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 *i-node* that contains the *path name* of another. References to the symbolic link become references to the named i-node.

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.

target controller — An intelligent (microprocessor based) circuit board which interfaces between the SCSI bus and the peripheral device. A target can take the form of a bridge controller or an embedded controller.

text file, ASCII file — A *file*, the bytes of which are understood to be in ASCII code.

track — An addressable ring of *sections* on a *hard disk* or *floppy disk*; each hard disk or floppy disk 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 codes 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

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W

Your comments and suggestions are appreciated and will help us to provide the best documentation for your use.

1. How would you rate this document for COMPLETENESS? (Please Circle)

   Excellent Adequate Poor
   4 ----------3 ----------2 ----------1 ----------0

2. Identify any information that you feel should be included or removed.

   _____

   _____

3. How would you rate this document for ACCURACY of information? (Please Circle)

   Excellent Adequate Poor
   4 ----------3 ----------2 ----------1 ----------0

4. Specify page and nature of any error(s) found in this document.

   _____

   _____

5. How would you rate this document for ORGANIZATION of information? (Please Circle)

   Excellent Adequate Poor
   4 ----------3 ----------2 ----------1 ----------0

6. Describe any format or packaging problems you have experienced with this document.

   _____

   _____

7. Do you have any general comments or suggestions regarding this document?

   _____

   _____

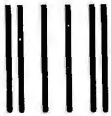
8. We would like to know a little about your background as a user of this document:

   A. Your job function _____ .

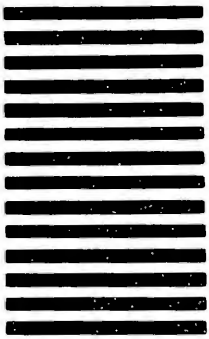
   B. Number of years experience with computer hardware: operation _____ , maintenance _____ .

   C. Number of years experience with computer software: user _____ , programmer _____ .

Your Name _____ Phone No. _____
Company _____
Address _____
City & State _____ Zip Code _____



NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES



BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO 1999 GREENSBORO, N C

POSTAGE WILL BE PAID BY ADDRESSEE

DOCUMENTATION SERVICES
2400 Reynolda Road
Winston-Salem, N.C. 27106-9989

Do Not Tear—Fold Here and Tape