System 6300
Administrator's Guide

Software Release FE07A

87601846C

Preface

This manual describes the tasks performed by the administrator of Systems 6300 and 6350. System configuration, adding and deleting users, and file system maintenance are described in detail.

Readers should be familiar with UNIX™ or UNIX-derived operating systems. See the Series 6000 Operating System Programmer's Guide and the Series 6000 Operating System Reference Manual for more information.

This issue covers release FE07A of the UNIX-derived operating system. New and changed material includes:

- o Information about streamer tape backup and the second hard disk (System 6350 only) is included in Sections 4 and 6.
- o Terminal configuration utility in Section 3. This routine provides menus and prompts to aid in terminal configuration.
- o Floppy diskette formatting utility in Section 4. This routine provides menus and prompts to aid in formatting floppy diskettes.
- o User configuration utility in Section 5. This routine provides menus and prompts to help you add users, remove users, and make directories.
- o Print configuration utility in Appendix D. This routine provides menus and prompts to automate the configuration of lp.
- o Appendix G, describing uucp, the routine for copying files between separate systems.
- o Appendix H, describing the utility for completely reconfiguring the system.

UNIX is a trademark of Bell Laboratories Inc.

Motorola, Inc.
10700 North De Anza Boulevard
Cupertino, California 95014

Contents

Tables

Illustrations

Section 1
Introduction

This manual describes the jobs performed by administrators of Systems 6300 and 6350. System configuration, adding and deleting users, and file system maintenance are described in detail. Each section details a separate topic, as follows:

Section 2 presents the four special ways in which the system administrator uses the system: as superuser, in single-user mode, using the standalone shell, and reading messages sent to the system console file.

Section 3 describes the procedures required for adding and removing peripheral devices from the system.

Section 4 provides details for using the fixed disk and removable diskettes, including formatting disks, and creating and checking file systems.

Section 5 covers the details of adding and removing users from the system, creating passwords, and moving users to different file systems.

Section 6 specifies the procedures for backing up files and restoring them; scheduling backups is also described.

Appendix A presents file system concepts.

Appendix B gives details on init and getty, which start processes and provide the correct internal environment for user log-in.

Appendix C describes "system accounting"--routines for keeping track of system processes.

Appendix D details the print system.

Appendix E explains the system-activity package.

Appendix F is the keyboard translation table for the TM30 keyboard.

Appendix G describes how to copy files between separate systems using the uucp utility.

Appendix H describes how to reconfigure the operating system.

RESPONSIBILITIES OF THE SYSTEM ADMINISTRATOR

The System 6300 administrator configures and allocates operating system resources. The administrator accesses the system in ways other users can't and controls the way other users use the system.

The administrator has the following specific responsibilities:

- Giving and denying other users access to the system.
- Specifying what diskettes and disk files users can access.
- Preparing new diskettes for use by the system.
- Telling the system how to use new terminals and printers.
- Performing routine backup of disk files to prevent accidental loss of data.
- Starting and stopping the system.

NOTATION CONVENTIONS

The System 6350 includes a second hard disk, and a streamer tape for doing backups. This manual does not make special mention of the System 6350. However, those portions of this manual that refer to the second hard disk and/or streamer tape apply only to the System 6350.

When examples are given to illustrate the use of a command, the administrator's input is shaded and the processor's response appears in normal type. Unechoed administrator input (for example, when the administrator is required to enter a password) is indicated by a shaded block. In the following example

```
login: Orr
$ passwd nuucp
new password:
retype new password:
```

the administrator has to enter the password twice; the shaded blocks indicate that the administrator's input does not appear on the screen.

Section 2
Administrative Interaction

The administrator has four special ways to interact with the operating system. Note that the first is not exclusive of the second.

- Superuser status. When using the operating system as superuser, you can ignore restrictions on file access and allowable commands.

- Single-user mode. When the operating system is in this mode, only one terminal is usable. Single-user mode is used for procedures that require an absence of normal disk activity. The single user in single-user mode has superuser status.

- Standalone shell. This is a program that provides some administrative commands when the operating system is not running. The single user running the standalone shell has superuser status.

- System console checking. The operating system logs its activities and problems for you to review in a file called /etc/log/confile.

This section describes these four interactions and also describes how to set the date.

CAUTION

> Do not halt, reset, or turn off a System 6300 unless the operating system is not running or is running in single-user mode.

SUPERUSER STATUS

Superuser status removes important operating system restrictions. The administrative commands in this manual require superuser status. The operating system gives the superuser three exemptions from normal restrictions:

- File read and write permissions do not apply to the superuser. The superuser can write to or read from any ordinary or special file. The superuser can create a file in or delete a file from any directory.

- Certain commands are executable only by the superuser.

- Some commands have built in safeguards or restrictions on the way they are used. Some safeguards and restrictions do not apply to the superuser.

When the operating system is running normally, there are two ways to obtain superuser status.

- o Log in as user "root."
- o Use the switch user program, su. Note that the default user is root.

Access to superuser status requires knowledge of root's user password. Consider root's password to be sensitive information. You should change the password from time to time, especially if there is any question of it becoming known. The default password is given in the UNIX-derived operating system *Software Release Guide* (SRG) shipped with your software.

When the operating system is running in single-user mode or the standalone shell, the sole user has superuser status.

The shell changes its prompt to remind you that you are superuser. Normally the default prompt is a dollar sign ($). When the superuser runs the shell, the default prompt is a pound sign (#).

The Root User

In the password file, /etc/passwd, the user called root has numeric user ID 0; this identifies root as the superuser. Under no circumstances change the name, numeric user ID, or numeric group ID of this user. Root should be the first user in the file.

Anyone who knows root's password can become superuser. When your system is first booted, root has a password of "Series6K". To change this password, run passwd:

```
    passwd
```

Passwd prompts for the old password once (if there is one) and the new password twice.

Root's home directory is / (slash), but this directory should not have any more files in it than necessary.

Example:

In this example, a user has changed his own password, then forgotten the new password. There is no way to reverse password encryption, so the valid password is lost forever. The only solution is for the user to get a new password, but only the user himself can change his own password and even then only if he knows his existing password. Fortunately, these restrictions don't apply to the superuser.

The administrator's input is shaded, the computer's responses are in normal type, and f indicates CTRL D. Note that passwd requires that the password be at least six characters long and contain at least two alphabetic characters and at least one numeric or special character when run by an ordinary user. If you run passwd as the superuser, then the restrictions on the password are minimal.

```
$ passwd walter
permission denied
f
$ login walter
password:
login incorrect
login: root
password:
# passwd walter
Changing password for walter
New password:
Retype new password:
# f
```

The su Command

To become superuser while logged in as an ordinary user, use the switch user command:

```
su
```

Su will prompt for a password; enter root's password. If the password is verified, su runs the shell with its numeric user ID set to 0, giving the shell the same status as a shell run by root.

To return to normal user status, terminate the su shell with CTRL D. You can also return to normal user status by using su with your own (or any other) user name, but this doesn't terminate execution of the superuser shell.

Example:

In this example, a system administrator changes root's password while logged in as a super user. The administrator's input is shaded, the computer's responses are in normal type, and f indicates CTRL D.

```
$ su
Password:
# passwd root
Changing password for root
Old password:
New password:
Retype new password:
# f
```

SINGLE-USER MODE

Single-user mode prevents ordinary users from communicating with the system. This prevents normal activity that might interfere with disk backup and maintenance.

There are two ways the operating system can go to its single-user mode:

- By commands from the system administrator.
- Automatically on start up if the operating system decides it is not safe to go to multiuser mode.

Both methods indirectly use the telinit command, a command that sends signals to the process initialization process, init. Do not change to single-user mode by using telinit directly: telinit does not give user programs a chance to terminate gracefully.

When the operating system is in single-user mode, only one terminal is usable: the terminal that was used to take the system to single-user mode. The user using this terminal has superuser status.

Taking the Operating System to Single-User Mode

To take the operating system to its single-user mode:

a. Make / (slash) your working directory:

```
cd /
```

b. Run shutdown:

```
/etc/shutdown grace
```

where

grace is the number of seconds the users get to log out by themselves. If grace is omitted, the users get 60 seconds. Shutdown runs wall to warn users, killall to terminate the users, and init to change the system's mode. This process takes about a minute.

Example:

Here's an example of going to single-user mode. A system administrator logged in as an ordinary user takes the system to single-user mode, giving the users two minutes to log out. The administrator's input is shaded, the computer's responses are in normal type, and f indicates CTRL D.

```
$ su
Password:
# cd /
# /etc/shutdown 120
```

The following display appears:

```
SHUTDOWN PROGRAM

<date> <time> <year>

Broadcast message from ROOT (tty020) <date>
SYSTEM BEING BROUGHT DOWN NOW ! ! !
All processes being killed


Do you want to continue? (c or n):  c

Error logging stopped.


INIT: New run level: S

INIT: SINGLE_USER MODE
Entered single user mode on <date>
OK to Stop Or Reset Processor.

Enter terminal type (default is tm31) ...
TERM=tm31
#
```

There is a two-minute delay after the broadcast message.

Automatically Going to Single-User Mode

The operating system has a start-up sequence that is executed whenever the system is turned on or reset. The start-up sequence includes a check of the file systems. The file-system check can have three outcomes.

o Nothing is wrong with any file system. The operating system goes to multiuser mode.

o One or more file systems is corrupt, but it is possible to fix them without destroying any data. The operating system fixes the corrupt file systems, then goes to multiuser mode. (However, if it had to fix the root file system, the operating system reboots itself, starting all over.)

o One or more file systems is so corrupt that no automatic fix is evident. The operating system goes to administrator mode.

When the operating system is in administrator mode, designated terminals prompt for the administrator to log in. If an ordinary user logs in, the system promptly logs that user off. If root logs in, the system switches to single-user mode so root can fix the file-system problem. The single working terminal in single-user mode is the terminal on which root logged in.

The system administrator specifies which terminals are to be active in administrator mode when the administrator configures the terminals. See Section 3, "Adding New Peripheral Devices," for more information.

Taking the Operating System to Multiuser Mode

To return to normal multiuser mode, terminate the shell. Press CTRL D in response to a shell prompt. The system prompts "Run level?" Enter 2, then press the RETURN key.

STANDALONE SHELL

The standalone shell provides a limited operating system environment when you cannot or should not boot the system. All commands are implemented on a diskette, so access to the fixed disk is avoidable.

There are three restrictions on standalone shell commands:

o No files can be written to. Existing files can be read.

o No input/output redirection is allowed. The ¦, <, and > constructs are illegal.

- Only the following commands are implemented:

  cat
  dd
  fsck
  fsdb
  ls
  mkfs
  iv
  volcopy

To run the standalone shell:

a. Take the operating system to single-user mode, if it is running.

b. Insert the standalone diskette in the drive.

c. Press the RESET button on the back of the processor.

CHECKING THE SYSTEM CONSOLE FILE

The operating system sends important messages about the system's activities, problems, and potential problems to the file /etc/log/confile. Examine this file periodically for information such as the following:

- Out of resource messages; for example, "disk out of space".

- System tables.

- Warning that a diskette was removed without a dismount command. This is the "pulled, may be inconsistent" message. See "Routine Checks of Fixed- and Removable-Disk File Systems" in Section 4, "Using Disks."

Use the "more" command, or some other utility, to look at the file:

```
more /etc/log/confile
```

SETTING THE DATE

It is important that the date is set correctly every day, especially if you are doing backups and restores.

If you change the date by more than one hour, first bring the system to single-user mode, as described under "Single-User Mode" earlier in this section.

To set the date, type

```
    date MMddhhmmyy
```

where

MM is the month of the yearm 01 to 12.

dd is the day of the month, 01 to 31.

hh is the hour in a 24-hour system, 00 to 23.

mm is the minute of the hour, 00 to 59.

yy is the year, expressed as the last two digits. If omitted, the year defaults to the current date setting.

Example:

The following command string sets the date to September 14, 1985, 12:25 AM.

```
    date 0914002585
```

Section 3
Adding New Peripheral Devices

This section describes the operating system changes required by new peripheral devices. This consists of changing certain configuration files that the operating system uses and of making sure that the operating system responds to the change.

CONFIGURING A NEW TERMINAL WITH TERM.CNFG

You can configure automatically with the term.cnfg command, as described below, or you can configure it manually, as described under "Configuring a New Terminal Manually" later in this section.

Term.cnfg(1M) manages terminal download files and terminal character-set translation source-files. Through a series of menus and prompts, term.cnfg lets you

- select terminal download files.
- select additional download files for international language support for TM31 Terminals.

If you want to include international character sets, make sure that the Utility Set 9 kernel is installed before you run term.cnfg. If it is not, you will have to exit term.cnfg, install the kernel, and begin again. To check for utility set 9, enter the following command:

```
ls /install
```

This command lists the contents of the install directory. Look for "rm.use9" in this list. If you don't find it, you don't have utility set 9 on your system.

To run term.cnfg, enter the following command:

```
/etc/term.cnfg
```

You must be logged on as root and the system must be in single-user mode. If you aren't logged on as root, this error message appears and the program aborts:

```
YOU MUST LOG ON AS ROOT TO CONFIGURE THE TERMINAL DOWNLOAD AREA

Aborting terminal configuration
```

If the system is not in single-user mode, this error message appears and the program aborts:

```
YOU MUST BE IN SINGLE-USER MODE!
EXECUTE 'shutdown' THEN RUN 'term.cnfg'

Aborting terminal configuration
```

To exit the program at any time without making any changes, type Q and press RETURN. To correct a mistake, you must exit and restart the program. When you exit the program, this message appears:

```
Aborting terminal configuration
```

When you start term.cnfg, a terminal-selection menu appears. The menu is similar to the one below:

```
There are nnn 1024 byte blocks left in the terminal area

# Terminal Protocol Language

1    gt     RS-232   ascii
2    gt     RS-422   ascii
3    tm30   RS-232   ascii
4    tm30   RS-422   ascii
5    tm31   RS-232   ascii
6    tm31   RS-422   ascii
7    tm40   RS-422   ascii

Please choose a download number:
```

where

Nnn is the space, in blocks of 1024 bytes, remaining in the terminal download area.

The menu that appears includes terminals added in the previous configuration, and thus can change each time you run term.cnfg. The language abbreviations are as follows:

Name	Language
ascii	ASCII
cdn	French-Canadian
deut	German
engli	English (U.S.A.)
esp	Spanish
fra	French
hol	Dutch
sve	Swedish
uk	English (U.K.)

Enter the number of a terminal download file and press RETURN. You need to enter a terminal download number only once. If you enter it again, this message appears:

```
    Download has already been selected
```

If you entered a number that is not listed, the following error message appears:

```
    Input out of range

    Please choose a download number:
```

Enter another number and press RETURN.

If you selected a TM30 or TM31 Terminal attached to an RS-422 port, this prompt appears:

```
    Type 'D' to make this terminal type the default tm30/tm31 download:
```

Type D and press RETURN to have the terminal automatically downloaded with that tm30 or tm31 download file when it is turned on. If you don't want this terminal download to be the default, press RETURN. If you have TM31 Terminals you may want to wait until after you install international character sets to select automatic download files. If you do not choose a default download file, you will have to enter the 3-digit number of the terminal download file when you boot the terminal. Term.cnfg lists the download files and their numbers at the end of the program.

If you selected a terminal type for which the system has no established download file number, this prompt appears:

```
    Enter the download number for this terminal:
```

Enter the the number of the download file you want for this terminal and press RETURN. Valid download numbers are in the range 100-300. If you enter a number outside this range, the following error message appears:

```
    You must enter a number in the range of 100 to 300.
    Please reenter number.
```

Enter a number between 100 and 300 and press RETURN.

If you enter a number that is already being used, this error message appears:

```
    The download number nnn is already in use.
    Please enter another number.
```

where

nnn is the number you just entered. Enter another number between 100 and 300 and press RETURN.

Continue selecting terminals with the terminal-selection menu. term.cnfg subtracts the size of each selected download file from the remaining space in the download area and updates the selection menu. If you choose a terminal download file that is too large for the remaining download area, the following message appears:

```
    You have exceeded the space allowed for terminal download files by 'nnn'
    This terminal type will not be included on your system
```

where

nnn is the amount by which the file would have overflowed the terminal download area. The terminal you selected will not be included in the terminal download area. The terminal-selection menu reappears.

When you have finished selecting terminal types, press RETURN. The following prompt appears:

```
    Do you want to include International character sets
    (from floppy disk) for your tm31 terminal?

          —Type 'I' to include International character sets for
           your tm31 terminal followed by the key marked 'Return'

          --or just press the key marked 'Return' to omit
           International character sets:
    ?:
```

If you don't want to install international character sets, press RETURN. The program creates the download area from the selections you made, displays messages similar to the ones below, then exits.

```
    Building the description file

    your download area consists of:
    dl # dl file
    ---- -------
    100  /usr/lib/iv/dl.term/tm31.4.ascii
    101  /usr/lib/iv/dl.term/tm31.2.ascii


    Writing the description file
```

To install international character sets, press I and RETURN. If the Utility Set 9 kernel is not installed, the error message below appears and the terminal-selection menu returns.

```
    Utility Set number 9 'Kernal' must be installed
    before you can add International Character Sets
    Install Utility Set 9 then rerun /etc/term.cnfg
```

If you want to include international character sets, press Q and RETURN to stop term.cnfg, install Utility Set 9, and rerun term.cnfg.

If the Utility Set 9 kernel has been installed, the following prompt appears:

```
Please mount the International Character Set floppy
and press the key marked 'Return' :
```

Mount the appropriate international character-set floppy diskette and press RETURN. Use either the RS-422 or the RS-232 diskette, depending on the protocol of the terminal types you selected.

If you are including international character sets, the following terminal language-selection menu appears.

```
There are nnn 1024 byte blocks left in the terminal area

#     Terminal   Protocol   Language

1     tm31       RS-ttt     cdn
2     tm31       RS-ttt     deut
3     tm31       RS-ttt     engli
4     tm31       RS-ttt     esp
5     tm31       RS-ttt     fra
6     tm31       RS-ttt     hol
7     tm31       RS-ttt     sve
8     tm31       RS-ttt     uk

Please choose a download number:
```

In this display, nnn is the number of 1024 byte blocks remaining in the terminal download area and ttt is either 422 or 232, depending on which international character-set diskette you are using. The language abbreviations are as follows:

Name	Language
cdn	French-Canadian
deut	German
engli	English (U.S.A.)
esp	Spanish
fra	French
hol	Dutch
sve	Swedish
uk	English (U.K.)

Enter the number of a language download file and press RETURN. Term.cnfg copies that language's terminal download file and translation table to the system disk, which takes about 30 seconds. This message appears:

```
Please Wait Files Being Copied
```

Once you have selected a language's terminal download file, it becomes a permanent part of the system; you will not need the international character-set diskette to reconfigure the download area with the terminal type. For example, if you run term.cnfg and select the hol download file, then the next time you run term.cnfg, the hol download file will already be on your system.

As you select different languages, the space remaining in the terminal download area is recalculated and displayed on the language-selection menu.

If you have not chosen a default download file for a tm30- or a tm31-type terminal, the following prompt appears each time you select a tm31 download file:

```
Type 'D' to make this terminal type the default tm30/tm31 download:
```

Type D and press RETURN if you want the terminal type to be automatically downloaded when the terminal is turned on. If you don't want this file to be the default, press RETURN. If you don't choose a default download file, you will have to specify the 3-digit download file number when you boot the terminal.

To end language selection, press RETURN. The following prompt appears:

```
It is safe to remove exchangeable disk /dev/rfp020

Building the description file

Remove the diskette from the drive.
```

After you finish adding international character-set download files, messages similar to the ones below appear. The list shows what files are contained in the download area of the system disk.

```
Building the description file

your download area consists of:
dl # dl file
---- -------
100  /usr/lib/iv/dl.term/tm31.4.ascii
131  /usr/lib/iv/dl.term/tm30.2
102  /usr/lib/iv/dl.term/tm31.4.esp

Writing the description file
```

If you selected international character sets, a new operating system kernel is made and the system is rebooted automatically. This process takes several minutes.

If there is no TRANSLATE entry in root's /.profile and if you selected nonascii terminal download files, this error message appears:

```
    Warning: There is an error in /.profile -- no TRANSLATE entry
```

Without this line in the .profile, no language translation can occur. To restore the .profile, copy /etc/user.profile to /.profile.

CONFIGURING A NEW TERMINAL MANUALLY

The following procedures detail the manual steps you would need to go through to do what the term.cnfg command does automatically. Each new terminal requires the following actions:

a. Determine the new terminal's number.

b. Create an entry in the configuration file for the init command.

c. Create an entry in the file that lists terminal types.

d. Make sure that init rereads its configuration file.

It is important to distinguish between RS-232 terminals and RS-422 terminals. The terms RS-232 and RS-422 actually describe the kind of communication link between the terminal and the System 6300. Each RS-232 line supports a single terminal. The RS-422 line supports up to eight terminals. Most terminals can only be used on an RS-232 line. Motorola TM30 Terminals can be used on either kind of line.

The Terminal Number and the Console

Each terminal has a three-digit decimal number. Terminal numbers start from 000. Note that a terminal number is always expressed in three digits. Numbers are expressed this way so that system programs can be the same on System 6600 and System 6300 systems.

Terminal numbers 000 through 017 designate RS-232 terminals. The terminal number corresponds to the channel number on the back panel label.

Terminal numbers 020 through 035 designate RS-422 terminals. An RS-422 terminal does not automatically get a certain number: the number is assigned when you turn the terminal on. Turning the terminal on gives it the lowest RS-422 terminal number not already in use. Turning the terminal off frees its number; this number may then be appropriated by some other terminal. Thus the only way to make sure that an RS-422 terminal gets a specific number is to control the order in which the RS-422 terminals are turned on. Normally it is not necessary to make sure that a terminal has a specific terminal number. But note how many RS-422 terminals are in use so you will know what terminal numbers will be allocated.

Certain important system messages are sent to the system console file, /dev/console. This file is simply a link to terminal 020, the default, but the link can be to any terminal. See "Checking the System Console File" in Section 2.

Configuring getty

The text file /etc/gettydefs is used by getty, the operating system's terminal initializer. Each entry specifies a set of communication options and a log-in message. Each set of similar terminals connected to the system requires two entries: one for multiuser mode and one for administrator mode.

As distributed, /etc/gettydefs contains eleven entries:

o The RS422 entry, which defines communication options suitable for an RS-422 terminal and a multiuser mode log-in message.

o The CRS422 entry, which defines communication options suitable for an RS-422 terminal and an administrator mode log-in message.

o The tm30.9600 entry, which defines communication options suitable for an RS-232 9600-baud terminal (including the tm31) and a multiuser mode log-in message.

o The Ctm30.9600 entry, which defines communication options suitable for an RS-232 9600-baud terminal (including the tm31) and an administrator mode log-in message.

o The 9600 entry, which defines communication options suitable for an RS-232 9600-baud terminal and a multiuser mode log-in message or 9600-baud communication line.

o The C9600 entry, which defines communication options suitable for an RS-232 9600-baud terminal and an administrator mode log-in message.

o The E9600 entry, which defines communication options suitable for an RS-232 9600-baud, even parity terminal and a multiuser mode log-in message.

o The 4800 entry, which defines communication options suitable for an RS-232 4800-baud terminal and a multiuser mode log-in message.

o The 2400 entry, which defines communication options suitable for an RS-232 2400-baud terminal and a multiuser mode log-in message.

o The 1200 entry, which defines communication options suitable for an RS-232 1200-baud terminal and a multiuser mode log-in message.

o The 300 entry, which defines communication options suitable for an RS-232 300-baud terminal and a multiuser mode log-in message.

Each entry in /etc/gettydefs is a line of the form

label#ioptions#foptions#message#next

where

label identifies the entry. The only strict rule is that *label* be unique in the file. A common convention labels a multiuser-mode entry with its baud rate and an administrator-mode entry with C followed by the baud rate; use this convention only if it's convenient.

ioptions is a list of communications options for getty to apply when it first opens the terminal. Specify options by using the symbolic constants described under termio(7) in the *Series 6000 Operating System Reference Manual*. Symbolic constants are separated from each other by spaces or tabs.

foptions is a list of communications options for getty to apply before calling login (that is, after a user first enters a log-in name). *Foptions* contains the same kind of information as *ioptions*.

message is text to print when the terminal is first opened. The text should end with "login: ".

next indicates another entry to use if getty receives a break while it's using this entry. If you don't know one or more of a terminal's communication options in advance (most often the speed), use the *next* fields to form a circular linked list of entries. A user can then select the right entry by pressing the RETURN key until a log-in message appears.

To include nongraphic characters in the entry, use one of the following sequences:

\n	new line
\t	tab
\v	vertical tab (CTRL K)
\b	backspace
\r	carriage return
\f	form feed
\xxx	

where

xxx is a 1- to 3-digit octal number.

A backslash (\) followed by any character not mentioned above just stands for the second character. Thus you enter \t to get a t.

There should be only one difference between a multiuser-mode entry and the corresponding administrator mode entry. The administrator-mode entry should have a message that reminds the user that the system is in administrator mode.

Check the correctness of the new /etc/gettydefs after modifying it. The following command finds errors:

```
/etc/getty -c /etc/gettydefs
```

Configuring init

The text file /etc/inittab is used by init, the master process spawner. Each new terminal requires that the init table be modified so that the operating system monitors the terminal for attempted log-ins. Use a text editor to modify /etc/inittab.

Each terminal requires a line in /etc/inittab of the form

```
ttt:23:respawn:/etc/getty ttyttt def
```

where

ttt is the terminal number.

def indicates a multiuser-mode definition in /etc/gettydefs.

Each terminal that is to be active in administrator mode requires a line in /etc/inittab of the form

```
Cttt:6:respawn:/etc/getty ttyttt def
```

where

ttt is the terminal number.

def indicates an administrator-mode definition in /etc/gettydefs.

Normally, only terminals 000 and 020 are active in administrator mode, although any terminal can be the console. If you need any RS-422 terminals to be active in administrator mode, make all RS-422 terminals active: you cannot know in advance which specific RS-422 terminal gets which specific terminal number.

Configuring Terminal Type

The terminal type file, /etc/ttytype, lists the kind of terminal represented by each terminal number. Use an editor to modify this file.

Each entry in /etc/ttytype is a line of the form

```
    type ttyttt
```

where

type is a terminal type. A Motorola TM30 Terminal is pt. For other codes, search the file /etc/termcap. To add new terminals to this file, see the discussion of *termcap*(4) in the *Series 6000 Operating System Reference Manual*.

ttt is the three-digit terminal number.

Rereading Terminal Configuration Files

When the operating system is running normally, init rereads /etc/inittab every time a user logs off and every five minutes. If this is not soon enough and it is inconvenient to reboot, the following command tells init to reread /etc/inittab.

```
    telinit q
```

CAUTION

> Use the telinit command carefully and precisely. The wrong parameter will stop the operating system suddenly and corrupt open files.

Getty uses new entries in /etc/gettydefs without any prompting, but getty acting on an obsolete /etc/gettydefs entry can tie up a terminal. If a terminal remains unusable after a change to /etc/gettydefs, try the following two steps:

a. Find the process number of the getty monitoring the terminal:

```
    ps -tttt
```

where

ttt is the three-digit terminal number.

b. Terminate getty:

```
    kill n
```

where

n is the process number displayed by ps.

If the above steps are unsuccessful since the terminal number is not displayed, then kill all getty's that have a "?" for a terminal number.

Example:

In this example, a System 6300 currently has a single terminal: a Motorola TM30 Terminal connected to an RS-232 line. The system administrator adds four new terminals: a Datamedia 2500, connected to an RS-232 line; and three Motorola TM30 Terminals, connected to the RS-422 line.

The system administrator has just connected the new terminals and has logged in as root. All new terminals are on the RS-422 line or run at 9600 baud, so there is no need to modify /etc/gettydefs. None of the new terminals are to be active in administrator mode. The administrator does not know the terminal type that corresponds to the new RS-232 terminal, so he checks /etc/termcap.

```
# fgrep datamedia /etc/termcap
D0 dm1520 dm1521 1521 1520 datamedia 1520:              Determine if
D1 dm1521 1521 datamedia 1521:                          "datamedia" is
D2 dm2500 datamedia2500 2500 datamedia 2500:            supported by
D3 dm3025 datamedia 3025a:is= EQ EU EV:                 termcap
D4 3045 dm3045 datamedia 3045a:is= EU EV:
D5 dt80 dmdt80 dm80 datamedia dt80/1:
D6 dt80132 dmdt80132 datamedia dt80/1 in 132 char mode:
# ed /etc/inittab
32
1,$p
000:23:respawn:/etc/getty tty000 9600                   Spawn "getty"
C000:6:respawn:/etc/getty tty000 C9600                  on new ports
$a                                                      for login
001:23:respawn:/etc/getty tty001 9600
020:23:respawn:/etc/getty tty020 RS422
021:23:respawn:/etc/getty tty021 RS422
022:23:respawn:/etc/getty tty022 RS422
.
w
610
e /etc/ttytype
6
$p
pt 000
$a
dm2500 001
pt 020
pt 021
pt 022
.
w
34
q
# /etc/telinit q
#
```

Removing Terminals

Be absolutely sure that /etc/inittab does not refer to any terminal numbers not in use. Init may experience difficulties bringing the system to multiuser state if it tries to open nonexistent terminals.

Note that removing an RS-422 terminal always abandons the highest terminal number. It does not matter which RS-422 terminal you remove, since RS-422 terminals are assigned terminal numbers dynamically when they are turned on or rebooted.

CONFIGURING MODEMS

Modems require special treatment because they allow two kinds of communication: dial-in connections and dial-out connections. Dial-in connections are established by remote systems calling yours. Dial-out connections to remote systems are established by the operating system using uucp, cu, or ct. To configure your system to use a modem, you will have to add special entries to various files used by uucp, cu, and ct.

Dial-In and Dial-Out Connections

For dial-in connections, the line your modem uses must be monitored by getty, as directly connected terminals are. When your modem responds to a dial-in call from another modem, getty wakes up and starts the log-in process.

For dial-out connections, a user program such as cu, ct, or uucp opens the terminal line and uses the modem to establish a communication link. In this case, getty must not be monitoring the line, or it will interfere with the user program.

To establish this kind of dual use, give the modem a special multiuser mode entry in /etc/inittab, as shown below, so that getty will monitor the modem's line in run level 3 (see Appendix B for details on /etc/inittab).

```
    ttt:3:respawn:/etc/getty ttyttt def
```

where

ttt is the terminal number of the modem's line.

3 is the run level at which init causes getty to monitor the modem's line.

def indicates a multiuser mode definition in /etc/gettydefs.

Note that all regular terminal entries in /etc/inittab should specify run levels 2 and 3, as shown below, so that terminals will work in both run level 2 and run level 3.

```
     ttt: 23:respawn;/etc/getty ttyttt def
```

No reboot or message to init is necessary initially. At normal run level (2), dial-out connections are possible because getty does not monitor the line at run level 2; dial-in connections are possible in run level 3 only. To allow dial-in connection, do

```
    telinit 3
```

To banish dial-in connections again, use the following procedure.

a. Look at your list of users to make sure no one is logged in over the modem lines:

```
        who
```

b. Tell the system to banish getty from the modem lines:

```
        telinit q
```

CAUTION

> Use the telinit command carefully and precisely.
> The wrong parameter will stop the operating system
> suddenly and corrupt open files.

Note that when the operating system is booted it comes up in run level 2, and dial-in connections are impossible until you allow them.

If you lose track of which run level you're in, do

```
    who -r
```

The number printed after "run level" is the current run level.

Special File Entries for Modems

For your system to handle modem connections to other processors, you must establish the appropriate entries in the files listed below. You must be superuser to modify these files. See "UUCP" in the Series 6000 Operating System Programmer's Guide for complete details on the formats of entries in these files.

- /usr/lib/uucp/modemcap
- /usr/lib/uucp/L.sys

- /usr/lib/uucp/L-devices
- /usr/lib/uucp/USERFILE
- /ect/rc

UPDATING /usr/lib/uucp/modemcap

This file describes the call-placing protocol of smart modems. The system supports any modem. Some of the more common modems already have entries in the modemcap file, so check to see if there is one for yours. This file contains documentation on how to construct an entry if you have to make one. Also, see "MODEMCAP (5)" in the *Series 6000 Operating System Reference Manual* (Volume 2) for details on /usr/lib/uucp/modemcap.

UPDATING /usr/lib/uucp/L.sys

This file tells uucp (on your system) what other systems it can talk to and how they are connected. You must make sure there is an entry for each remote system your system will be calling. Entries have the form:

system name *time* *device* *speed* *phone number* *login*

where

system name is the unique symbolic name of the remote system.

time is a string that indicates the days of the week and times of day when the system should be called. You can use Su Mo Tu We Th Fr Sa for the days of the week, or Wk for any week day, or Any for any day. For example, "MoWeFr0800--1200" indicates Monday, Wednesday, and Friday between 8:00 AM and 12:00 noon.

device is either ACU or the hardwired device to be used for the call. For the hardwired device, the port number is used, such as tty001.

speed is the line speed for the call, unless the C library routine "dialout" is available, in which case this field is the dialout class.

phone number is the phone number of the remote system. If you are dialing an outside line, the number must include the correct digit (usually 9) followed by an equal sign (=), which tells uucp to wait for a dial tone.

login is a series of fields and subfields in the format

```
[expect send] ...
```

where

expect is the string expected to be read.

send is the string to be sent when the "expect" string is received. There can be subfields for passwords.

Standard usage on the System 6300 is as follows:

```
login:--login: nuucp password: password
```

where

password is the password for nuucp on the system you are calling. To get this password, call the appropriate system administrator.

Example:

This example describes a hypothetical system named "Scott" for use with a dial-up line at 1200 baud. The system can be called at any time.

```
Scott Any ACU 1200 9=8642252 login:--login: nuucp password: password
```

UPDATING /usr/lib/uucp/L-devices

This file describes all direct and dial-up lines that can be referenced by uucp and cu.

Entries in this file have the form:

```
type line call-unit speed
```

where

type is a device type, such as ACU for an automatic call-unit (smart modem) or DIR for a direct hardwired line.

line is the device for the line (the call-unit line/port number) such as cu05.

call-unit is the automatic call-unit associated with *line*. Hardwired lines have a number "0" in this field.

speed is the line speed.

Although the modem is technically a dial-up line, it is convenient to be able to give it commands directly for debugging purposes. Since a direct line and a dial-up line cannot have the same device name, link /dev/tty001 onto another device:

```
    ln /dev/tty001 /dev/modem
```

where

modem is a unique device name of your choice.

Now you can treat /dev/modem as a dial-up line and still maintain /dev/tty001 as a direct connection. Put the following lines into /usr/lib/uucp/L-devices

```
    ACU modem call-unit speed
    DIR tty001 0 speed
```

where

ACU indicates "automatic call unit" (this is the device type).

modem is the device name previously mapped onto tty001 using the ln command.

call-unit is part of the entry for your modem in /usr/lib/uucp/modemcap. Use the name between the pipe symbols (¦name¦).

speed is the line speed at which your modem operates. This must be the same for both lines.

DIR indicates "direct" line.

UPDATING /usr/lib/uucp/USERFILE

This file contains information needed by uucp when determining generalized access permissions to directories and functions. Remote systems are restricted in their access to your system based on the contents of this file. There must be an entry in this file on your system for each remote system that will be calling you.

Example:

This example shows what you would enter to give full access to the remote system named "Scott," that was described under "Updating /usr/lib/uucp/L.sys."

```
    nuucp, Scott /
```

UPDATING /etc/rc

Each system in a communications network has a unique symbolic name by which it is known to the other systems. The first item of any entry in the L.sys file on your system is the unique name used by uucp when communicating with the remote system that the entry describes. And likewise, on the other systems, there is an entry and unique name in L.sys for your system.

Booting the operating system changes the unique system name to a default that is set in the file /etc/rc, so this file should be edited to reflect your unique system name. Use vi to change the system name on the line that reads

```
    setuname -n system name
```

where

system name is the currently set system name.

To reestablish the use of the unique system name without rebooting the system, use the following command string:

```
    setuname -n system name
```

where

system name is the name remote systems use in their L.sys files to denote your system.

Sample Session for Configuring Modems

This session describes the steps you need to take to configure two Hayes 1200 smartmodems between two System 6300s named Bert and Ernie. Assume that each system will use tty001 for its modem line. For details on the file entries given below, refer to the preceding paragraphs.

ON SYSTEM BERT

- For each terminal entry in /etc/inittab, change the "level" descriptor so terminals will work in run levels 2 and 3.

```
      ttt:23:respawn:/etc/getty ttyttt def
```

- Make a special multiuser mode entry in /etc/inittab so getty will monitor the modem's line only in run level 3.

```
001:3:respawn:/etc/getty tty001 def
```

- Make the following entry in /usr/lib/uucp/L.sys:

```
Ernie Any ACU 1200 <phone number> login:--login: nuucp password: *ducky
```

- Establish an nuucp password (*pigeon) in the password file /etc/passwd. This is used in L.sys on system Ernie.

```
passwd nuucp
new password:
retype new password:
```

- Link /dev/tty001 onto another device named "modem",

```
ln /dev/tty001 /dev/modem
```

  and make the following entries in /usr/lib/uucp/L-devices:

```
ACU modem smartmodem 1200
DIR tty001 0 1200
```

- Make the following entry in /usr/lib/uucp/USERFILE:

```
nuucp, Ernie /
```

- Run setuname:

```
setuname -n Bert
```

- Change the system name in the /etc/rc file to Bert.

ON SYSTEM ERNIE

- For each terminal entry in /etc/inittab, change the "level" descriptor so terminals will work in run levels 2 and 3.

```
ttt:23:respawn:/etc/getty ttyttt def
```

- make a special multiuser mode entry in /etc/inittab so getty will monitor the modem's line only in run level 3.

```
001:3:respawn:/etc/getty tty001 def
```

- Make the following entry in /usr/lib/uucp/L.sys:

```
Bert Any ACU 1200 <phone number> login:--login: nuucp password: *pigeon
```

- Establish an nuucp password (*ducky) in the password file. This is used in L.sys on system Bert.

```
passwd nuucp
new password:
retype new password:
```

- Link /dev/tty001 onto another device named "modem",

```
ln /dev/tty001 /dev/modem
```

  and make the following entries in /usr/lib/uucp/L-devices:

```
ACU modem smartmodem 1200
DIR tty001 0 1200
```

- Make the following entry in /usr/lib/uucp/USERFILE:

```
nuucp, Bert /
```

- Run setuname.

```
setuname -n Ernie
```

- Change the system name in the /etc/rc file to Ernie.

PRINTERS

The operating system provides two incompatible print spooling programs: lp and lpr. Lp supports multiple printers and has advanced features. Lpr is simple but adequate for a system with a single printer. Appendix D tells how to configure lp; the remainder of this subsection tells how to configure lpr.

Lpr can use two kinds of printers:

o Centronics-compatible parallel printer. This is connected to the special parallel printer line.

o Serial printer. This is connected to any RS-232 terminal line.

The parallel printer line is associated with the special file /dev/plp.

An RS-232 line used by a receive-only printer must not be monitored for log-ins. Examine /etc/inittab and verify the absence of any reference to the line you intend to use. Terminal line usage is discussed earlier in this section.

To make lpr use the correct line, associate the special file /dev/lp with the correct line. If the printer is connected to the parallel printer line, do

```
ln /dev/plp /dev/lp
```

If the printer is connected to an RS-232 terminal line, do

```
ln /dev/ttyttt /dev/lp
```

where

ttt is the three-digit terminal number.

As shipped, the System 6300 operating system has /dev/lp associated with the parallel printer line.

Use ls to verify that a /dev/lp is associated with the correct line. Two special files are associated with the same line if they have the same i-number. To see if /dev/lp has the same i-number as /dev/plp, do

```
ls -li /dev/plp /dev/lp
```

This command produces a listing like this:

```
3438 crwxrwxrwx 2 root root        6, 0 /dev/plp
3438 crwxrwxrwx 2 root root        6, 0 /dev/lp
```

Only the i-number (first item on each line) and the special file name (last item on each line) are of interest. Verify that the two i-numbers are identical; if they aren't, /dev/lp is not associated with /dev/plp.

To see if /dev/lp has the same i-number as an RS-232 terminal line, do

```
    ls -li /dev/ttyttt /dev/lp
```

where

ttt is the three-digit terminal number.

This command produces a listing much like that of the other version of the ls command. Again, compare i-numbers: they must be the same or /dev/lp is not associated with the indicated line.

If a serial printer is used and the printer's communication requirements do not match the operating system's defaults, you must arrange for the operating system to set and hold terminal options. To do that, add the following two lines at the end of /etc/rc:

```
    nohup sleep 2000000 > /dev/lp &
    stty options < /dev/lp
```

where

options is a list of valid stty options: see stty(1) in the *Series 6000 Operating System Reference Manual*.

CONFIGURING CALL-UP DEVICES

When your system calls a remote system, nuucp opens the terminal line and makes a communications link between the two systems. To enable nuucp to establish this link, you must first alter the /etc/inittab file for tty001 on each system. For the initiating system, create or alter the line for tty001 so that it looks like this:

```
    001:4:respawn:/etc/getty tty001 9600
```

For the responding system, create or alter the line for tty001 so that it appears this way:

```
    001:2:respawn:/etc/getty -t 30 tty001 9600
```

If you wish to attach a password to the nuucp utility, type

```
    passwd nuucp
```

and enter the password you want to use for this utility. When the password prompt appears the second time, reenter the password to make certain you spelled it correctly the first time.

For both the initiating and responding systems you must set the permission modes to read and write permission for owner, group, and others. For each system then, write

```
chmod 0666 /dev/ttyxxx
```

where

xxx is the number of the terminal that you are using for communication.

Section 4
Using Disks

This section describes the use of the System 6300 disks. It describes how disks are organized and how the system administrator makes the disk's storage capacity available to users for various purposes.

There are two basic types of disks: fixed and removable (floppy diskettes). Fixed and removable disks are organized in a similar way; however, to avoid confusion, we will discuss their use separately. A System 6300 can have either one or two fixed disks. Some systems use quarter-inch tape as a second removable medium; using tape is described in Section 6.

Note that for the System 6300, disk divisions can properly be called either slices or partitions. "Partition" is the preferred term, although "slice" is more appropriate when referring to physical sector layout on the disk. Both terms are used in this manual.

This section discusses the use of the iv, mkdir, labelit, mount, umount, mklost+found, chown, chmod, dismount, and fsck commands. For complete information on these commands, see the Series 6000 Operating System Reference Manuals, Volumes 1 and 2.

FIXED-DISK ORGANIZATION

This subsection explains how the fixed disk is organized. Concepts are introduced here that express how the disk is divided into manageable units and what purpose the units serve.

A disk has 1 to 16 partitions, numbered from 0 to 15. A partition is a section of the disk that is used as a unit. Partition 0, also called the reserved area, is preassigned to hold the data required to manage the disk; there is normally at least one additional partition on the disk.

If a disk is used to boot the operating system (the first fixed disk always can), partitions 1 and 2 also have preassigned purposes. Partition 1 contains the root file system. Partition 2 provides swap space; the size of this partition places a practical limit on program size.

Fixed-Disk File Systems

The basic use of a partition is to contain a file system. A file system consists of the files and the data structures the operating system kernel requires to support the file system. Normally, all of the partitions on the first fixed disk except for partition 0 (the reserved area) and partition 2 (the swap area) contain file systems. For a second fixed disk, normally all the partitions except for 0 (the reserved area) contain file systems.

To be used, a file system must be mounted to give it a place in the directory hierarchy. The root file system in partition 1 of the fixed disk is permanently mounted; other file systems are mounted manually by use of the mount command or by an automatic procedure (using mount) that is executed when the system is booted. The file /etc/rc contains control information about various processes to be started when the system is booted--including which file systems to mount automatically. The mount commands themselves are contained in the file /etc/mountable, which is referenced from /etc/rc.

The file system appears to users as a single hierarchy of directories and files. For the most part, the division of the file hierarchy into partitions and separate mountable file systems is not apparent to users and disk organization is not an ordinary user's concern.

Fixed-Disk Swap Partition

The size of the swap partition determines the total limit on program memory usage. Partition two on the fixed disk is always the swap partition; thus the swap partition on the fixed disk is /dev/fp002. A sign that the swap partition is too small is the failure of a large number of commands with messages like "not enough space." Experience will tell you how much swap space you need, but a good rule of thumb is that each terminal uses up 1 megabyte. The swap partition should be at least a little larger than the system's memory so that the memory is fully used.

Never write a file system or any other data into the swap partition.

Initializing and Configuring the Fixed Disks

The System 6300 first fixed disk is configured and initialized during the installation of the UNIX-derived operating system as outlined in the UNIX-derived operating system Software Release Guide (SRG) shipped with the operating system. No discussion is presented in this manual concerning modification of the partitions of the first fixed disk, as doing so might cause the operating system to crash and valuable files and data to be lost. To format and configure the second fixed disk, refer to the Software Release Guide.

REMOVABLE-DISKETTE ORGANIZATION

Removable floppy diskettes can be divided into as many as 16 partitions, numbered 0 to 15. As with the fixed disk, partition 0 is always the area reserved for the data required to manage the diskette's use; there is normally at least one additional partition on the diskette.

Since the System 6300 boots the UNIX-derived Operating System from the first fixed disk, this manual will not attempt to describe how to make a removable diskette bootable. See iv(1) in the Series 6000 Operating System Reference Manual, Volume 1, for more information on bootable disks.

A removable diskette does not have to contain a file system. Some data base systems manage partitions directly, without using a file system. Removable diskettes are often used for routine file backups and for the transfer of data between systems; such diskettes contain a single partition with no file system.

Unlike a file system that is managed by the operating system kernel, a plain partition is managed by the programs that use it.

Initializing and Configuring Removable Diskettes

This subsection explains the procedures that initialize a removable diskette and divide it into partitions. If you use these procedures on a diskette that is already in use (for example, adjusting partitions to accommodate another partition on the diskette), assume that all data on the diskette will be lost.

To initialize and configure a diskette, you can use the floppy disk format utility described below, or you can use the procedure described under "Formatting a Diskette Manually" later in this section.

FLOPPY DISK FORMAT UTILITY

The initf utility formats a floppy disk and places a file system on it. The floppy then is ready to mount and copy files to. To run this utility you must be logged on as root. To run initf, use the following procedures:

a. Start the utility by typing:

```
initf
```

The system displays the following screen as a warning that formatting the floppy disk will erase anything that is on it now.

```
******** Floppy Disk Format Utility ********

This utility will format a floppy disk erasing its contents.
Is this what you want? (Y/N):
```

If you do not want to erase the contents of the floppy disk, press 'N' and then RETURN, and the utility will stop.

b. Otherwise press 'Y', and the system will display the next prompt:

```
Insert the floppy disk into the drive and close the door.
Then hit <return>.
```

Insert the floppy into the drive with the write notch on the bottom edge, close the door, and press RETURN.

c. The next prompt appears.

```
Starting format of disk
```

Formatting takes about two minutes. Then the systems prompts:

```
Floppy is now formatted.

Do you want a file system on this floppy? (Y/N)
```

Press 'N' if you will be using the floppy as a raw device--for example, if you plan to use the cpio(1) utility with the -o option. However, if you plan to copy files to the floppy and want to access them randomly, then press 'Y' and then RETURN. The system proceeds, and displays the following message when formatting is done:

```
File system made. Floppy is now ready to mount.
```

d. To mount a file system on the floppy disk you have just formatted, enter:

```
mnt
```

Now you are ready to copy files to or from the floppy through the /flp directory. For example, to copy the utility initf to the floppy, type:

```
cp /usr/local/bin/initf /flp
```

e. After you are through using the floppy disk you must unmount it. To do this, type:

```
mnt -d
```

f. To mount a floppy disk for reading only, or one which has a write protect tab on it, type:

```
mnt -r
```

FORMATTING A DISKETTE MANUALLY

To format a diskette by hand, use the following procedure. This achieves the same results as using the initf command.

a. Determine the dimensions of the diskette.

b. Plan the diskette's partitions, and how the partitions are to be used.

c. If necessary, create a diskette description file or use one of the description files supplied on the System 6300 under /usr/lib/iv.

d. Run the iv utility to initialize the diskette.

After initializing a removable diskette, use the following command just before revoming the disk from the drive:

```
dismount -f
```

Use these procedures, discussed in the following subsections, to initialize a new diskette and to specify new partition boundaries on an old diskette.

NOTE

> Before applying these procedures to an old diskette, copy or back up the diskette. You may not be able to rearrange partition boundaries without damaging the existing partitions.

Determining Removable-diskette Dimensions

The prototype disk description files in /usr/lib/iv describe the various disks commonly used with the System 6300.

The disk description file that is used to initialize a System 6300 removable diskette is named desc.flpy. This is a text file that gives the diskette's physical characteristics. You can use a single diskette description file desc.flpy to initialize diskettes for the purposes of backup and offline file storage. The diskettes are used as follows:

- Backup. Each diskette serves as a "tape reel" for backups. The diskette needs only to be initialized; the backup programs will manage the diskette.

- Offline file storage. Users use these diskettes for their file systems. Partition 1 will contain a file system that users can mount and create files in.

Use cat or some other text file utility to examine the file, as in the example below:

```
    cat /usr/lib/iv/desc.flpy
```

The output of /usr/lib/iv/desc.flpy looks like this:

```
    #    iv description file for 96 TPI Floppy file system disk.
    type           FD
    name           Floppy
    cylinders      80
    heads          2
    sectors        8
    steprate       0
    exchangeable
    $
    $
    $
    0
    1
    $
    $
```

This prototype description file gives physical characteristics of the diskette. Of interest are heads, cylinders, and sectors. If sectors is an odd number, then the diskette has a sector on each track reserved as a bad block alternate, and you should subtract one from sectors before using it in calculations. You can calculate the total number of 1024-byte logical blocks the diskette can contain. Note that one logical block equals two physical sectors.

heads x cylinders x sectors / 2 = diskette size in logical blocks

Using the information from desc.flpy:

2 x 80 x 8 / 2 = 640 logical blocks of 1024 bytes each

Also calculate the number of logical blocks per track (call this value bpt):

sectors / 2 = logical blocks per track (bpt)

Using the information from desc.flpy:

8 / 2 = 4 logical blocks per track

Finally, calculate the total number of tracks:

heads x cylinders = number of tracks on the diskette

Using the information from desc.flpy:

2 x 80 = 160 tracks on the diskette

Planning the Diskette Partitions

In planning the size of the diskette's partitions, consider the applications and users who are likely to use the diskette. How many 1024-byte logical blocks is each application or user likely to require? If you have many users who will create many small files, they can share a file system, provided none of them is careless about using up extra space. What's a rough estimate of their total requirements?

Try to put user files in a file system separate from the root file system and other file systems that hold system utilities. On the System 6300 this is easy, as file systems can be mounted under /mnt/exch1..f (where 1..f is a partition number in hexadecimal for the file system being mounted). Users can then create and access directories under /exch1..f on the removable diskette.

Partitions always contain a whole number of tracks. Round off your estimates to the nearest multiple of bpt (blocks per track).

The first and most important partition is the reserved area (partition 0). Preparing a reserved area that supports a system boot is beyond the scope of this manual, but if you reconfigure a diskette that contains an operating system, do not decrease the size of the reserved area. A reserved area that does not support a system boot only needs one track.

Creating a Disk Description File to Use with iv

If you plan to configure the disk with different partition boundaries from those in the standard description file, you must create a disk description file that contains the proper partition information.

If you are working with a prototype description file, create a new disk description file in the following way:

a. Copy the appropriate prototype description file from /usr/lib/iv.

b. Use a text editor to change the disk name and partition information in the copy.

It is not absolutely necessary that the disk description file have any particular name or be in any particular directory. However, you will find it useful to keep all the disk description files you have created in a special directory, separate from /usr/lib/iv, and to name them in a way that indicates their purposes.

If you are working with a description file created by iv, use a text editor to change the disk name and partition information in the actual description file.

The disk name goes on an existing line. The line is of the form

```
    name tab diskname
```

where

name identifies the name line.

tab is a tab character, generated by pressing the TAB key or CTRL I.

diskname is the name of the disk. Only the first six characters are used. If the specified name is less than six characters long, the actual name is padded out to six with blanks.

Running iv

The iv program formats a diskette and divides it into partitions. This program is used to initialize new diskettes and to reinitialize old ones.

CAUTION

> Reinitializing an old diskette destroys any data on it, so be sure that the data is not needed or has been copied to another diskette.

The iv command has the following form:

```
    /etc/iv -i /dev/rfp020 file
```

where

file is the name of the diskette description file to use for formatting. A typical file would be /usr/lib/iv/desc.flpy.

Making and Using a Removable-Diskette File System

You can use the initf command, described under "Floppy Disk Format Utility," to make a file system on a diskette. You can also use the procedures described below. Each file system requires the following steps:

a. Create the file system with the mkfs command.

b. Mount the file system.

c. Create a lost+found directory for the file system.

CREATING THE REMOVABLE-DISKETTE FILE SYSTEM

mkfs creates a file system by writing file system structures into a partition. labelit puts a label on the partition.

```
    /etc/mkfs /dev/fp02p
    /etc/labelit /dev/rfp02p ldir vname
```

where

p is the partition number, in hexadecimal ("a" through "f" stand for "10" through "15"). This value is usually 1; do not specify partition 0, the reserved area. Normally, a floppy diskette has only two partitions, 0 and 1, due to its limited size.

ldir is the local name of the directory on which the file system is normally mounted. This will be flp.

vname is your name for the disk that holds the file system. Suggested name: d2 for the diskette.

Example:

In this example, an administrator wants a file system on a diskette. The administrator inserts a blank diskette into the drive and enters the following commands. The administrator's input is shaded.

```
# /etc/iv -i /dev/rfp020 /usr/lib/iv/desc.flpy
# /etc/mkfs /dev/fp021
# /etc/labelit /dev/rfp021 flp d2
```

MOUNTING AND UNMOUNTING THE REMOVABLE-DISKETTE FILE SYSTEM

To use a diskette file system, it must be mounted. Insert an initialized and formatted diskette containing an empty file system into the drive. Issue the following command for each file system on the diskette:

```
/etc/mount /dev/fp0tp /dirn roption
```

where

t identifies the type of disk; 0 means the fixed disk, 2 means the diskette.

p is the partition number in hexadecimal (a through f stand for "10" through "15"). P generally has a value of 1; do not specify 0, the reserved area.

dirn is a directory name such as /flp created by the user with mkdir.

roption (-r) controls access to the file system. If -r is specified, the file system is mounted read only: files in the file system can be read, subject to normal permission rules, but cannot be modified, created, or deleted by any user. If -r is not specified, the file system is mounted read/write: all files in the file system can be read, modified, created, and deleted, subject to normal permission rules.

Example:

To create the directory used to mount a diskette and then mount it, enter

```
cd /
mkdir flp

/etc/mount /dev/fp021 /flp
```

The last command mounts partition 1 of the floppy diskette as /flp. Here, /flp is a directory created in the root partition of the system.

Users can access the file systems on the removable diskette by referring to the subdirectories of /flp.

To unmount a file system, use the umount command:

```
    /etc/umount /dev/fp0tp
```

where

t is the disk containing the partition: 0 means the fixed disk, 2 means the diskette.

p is the partition number, in hexadecimal ("a" through "f" stand for "10" through "15"). This value is usually 1; do not specify partition 0, the reserved area. Normally, a floppy diskette has only two partitions, 0 and 1, due to its limited size.

Note that umount will fail with a "busy" message if a file on the file system is open or is a working directory. The fuser command identifies processes using files in a particular file system and is useful in tracing open files that cause umount to fail. For example, to find out who is using the floppy disk drive partition 1, become superuser and type:

```
    fuser -u /dev/fp021
```

Unmounting is automatic in some circumstances:

- Unmounting is implied by the dismount command, which cleanly disconnects a removable diskette from the operating system.

- The shutdown command, which takes the system to single-user mode, unmounts all file systems except the root file system. The shutdown command is discussed in Section 2.

NOTE

> The root file system (the file system in partition 1 of the disk from which the operating system was booted) is always accessible without a mount. Do not apply umount or mount to the root file system.

CREATING THE LOST+FOUND DIRECTORY

Each file system must have a special directory for use by the file system maintenance program fsck. To create this directory:

a. Make sure the file system is mounted.

b. Make the file system's root directory your working directory.

c. Run the program: /etc/mklost+found

ACCESS TO THE FILE SYSTEM

The system administrator must specify who can access the root directory of a removable diskette file system. Since removable diskettes are relatively small, two possibilities are likely:

- One particular user controls the root directory.
- All users are allowed to use the root directory.

To give a particular user control of the root directory, use the chown program:

```
    chown name /flp
```

where

name is a user name.

The user can then specify access permissions for the directory. If the user gives only himself write permission, then he has the exclusive ability to create files and subdirectories in /flp.

To give all users free access to the root directory, and thus to the file system, use the chmod program:

```
    chmod a+rwx /flp
```

All users can then create files and subdirectories in /flp.

Before removing the diskette from the drive, use dismount:

```
    dismount -f
```

Dismount dismounts the diskette's file systems, completes input/output, and tells the operating system kernel that removal of the diskette is safe. The following message appears after the dismount operation is completed:

```
    It is safe to remove the exchangeable diskette /dev/rfp020.
```

If a diskette is removed without a complete dismount, it is marked as being potentially inconsistent. If such a diskette is inserted in the drive, a warning message appears on the terminal. Information pertaining to the warning is written in the file /dev/console (see "Checking the Console File" in Section 2).

ROUTINE CHECKS OF FILE SYSTEMS

Use the fsck command (file system check) to test the integrity of the file systems on the fixed and removable disks. fsck runs automatically on fixed-disk file systems whenever the operating system is booted. However, you should run fsck for either of the following reasons:

- On each file system of a removable diskette if the system warns that it has been removed without being dismounted. This also happens if the diskette was in use when the system was stopped suddenly.
- On the file systems of all disks after not more than one week's worth of work.

Running fsck

The operating system must be in single-user state to check the root file system /dev/fp001 (fixed disk) and should be for any other file system. Any of the following procedures will bring the system to single-user state:

- If no other terminals are active, issue the halt command.
- If other terminals are active, issue the /etc/shutdown xxx command

  where

  xxx is a decimal number, representing the number of seconds "grace" period before the system actually goes down).
- If no other terminals are active, issue the telinit s or S command.

CHECKING THE ROOT FILE SYSTEM

Once the operating system is in single-user state, use the following command to run fsck on the root file system.

```
/etc/fsck -p /dev/fp001
```

NOTE

> You should never use the umount command on the root file system.

CHECKING OTHER FILE SYSTEMS

To run fsck on a file system other than the root file system, execute the following two commands.

```
    /etc/umount /dev/fp0tp
    /etc/fsck -p /dev/fp0tp
```

where

_t_ indicates the drive number: 0 indicates the first fixed disk, 1 indicates the second fixed disk, 2 indicates the removable diskette.

_p_ is the partition number in which the file system is located, in hexadecimal (a through f stand for "10" through "15").

The first command assures that the file system is unmounted; this is very important, but must not be done on the root file system. The second starts the file system checking program in automatic "-p" mode. fsck should handle most problems automatically. If it cannot, run it again in manual mode (/etc/fsck /dev/fp0_tp_).

Running fsck manually

To check a file system manually, run fsck without the -p option:

```
    /etc/fsck /dev/fp0tp
```

where

_t_ indicates the drive number: 0 indicates the first fixed disk, 1 indicates the second fixed disk, and 2 indicates the diskette.

_p_ is the partition number, in hexadecimal (a through f stand for "10" through "15"). Do not specify partition 0 (the reserved area).

This form of the fsck command requires your permission before each action.

Repair the normal root file system (/dev/fp001) first. This simplifies work on the other file systems. If fsck actually modified the root file system, it will reboot the operating system.

If you can't understand fsck's actions, do the following:

a. Familiarize yourself with the following fsck description and with Appendix A.

b. Continue running fsck, but assume you'll have to run it again. Grant permission only for minor repairs. Assess the condition of the file systems. Examples of minor repairs are:

- removing a small or unimportant file
- linking a file to lost+found
- removing an empty unreferenced file
- fixing the i-node count

c. For each damaged file system, consider how much work would be lost if the file system were thrown away and restored from back up. If this loss is not significant, abandon further work on the file system and restore from back up.

fsck Description

These are the messages and suggested actions fsck displays. When run with the -p option, fsck takes most of its own suggestions, stopping only when action would mean loss of data.

For a discussion of the terminology employed in this section, see Appendix A. These messages each report a disk problem and suggest a standard fix. These are the standard suggestions:

(CONTINUE)	Results may be invalid. Continue anyway?
(CLEAR)	Clear this i-node?
(REMOVE)	Remove this directory entry?
(FIX)	This value is wrong. Replace it with the right value?
(RECONNECT)	We have a good i-node but no directory entry for it. Make a directory entry for it in lost+found?

The various phases of the fsck routine are described below.

INITIALIZATION

CAN NOT SEEK: BLK *B* (CONTINUE)
CAN NOT READ: BLK *B* (CONTINUE)
CAN NOT WRITE: BLK *B* (CONTINUE)

I/O failed on the file system. If you decide to continue, do a second run to confirm the results of the first. Make sure the disk isn't write-protected.

PHASE 1: CHECK BLOCKS AND SIZES

UNKNOWN FILE TYPE I=*I* (CLEAR)

I-node *I* has an invalid type. If you decide to clear the i-node, its directory entries will be UNALLOCATED in Phase 2.

LINK COUNT TABLE OVERFLOW (CONTINUE)

An fsck internal table is full. A second fsck run will be necessary to confirm the results of the first. This message will repeat each time fsck encounters an allocated i-node whose link count is 0.

B BAD I=*I*

Block *B* on i-node *I* is bad. You'll get a BAD/DUP message in Phase 2 and Phase 4.

EXCESSIVE BAD BLKS I=*I* (CONTINUE)

I-node *I* has a large number of bad blocks. If you choose to continue, fsck will skip to the next i-node. Run fsck again to verify your results.

B DUP I=*I*

I-node *I* claims block *B*, but this block is already on fsck's list of allocated blocks. This will cause a BAD/DUP message in Phase 2 and Phase 4. This invokes Phase 1b. Be careful if you see this error. A good procedure is to note the i-numbers with this error and finish running fsck without changing the file system. Before running fsck again, run ncheck to discover the names of the affected files:

```
/etc/ncheck -i numbers /dev/name
```

where

numbers is a list of i-numbers. The numbers are separated from each other by spaces.

name is the same special file name used with fsck.

EXCESSIVE DUP BLKS I=*I* (CONTINUE)

I-node *I* has a large number of duplicate blocks. If you choose to continue, fsck will skip to the next i-node. Run fsck again to verify your results.

DUP TABLE OVERFLOW (CONTINUE)

An fsck internal table is full. A second fsck run will be necessary to confirm the results of the first. This message will repeat each time fsck encounters a duplicate block.

POSSIBLE FILE SIZE ERROR I=*I*

The indicated file has the wrong number of blocks for a file its size. A file size error does not necessarily indicate a real error: it can indicate a file that was written to nonsequentially.

DIRECTORY MISALIGNED I=*I*

Size of the indicated directory is not a multiple of 16.

PHASE 1B: RESCAN FOR MORE DUPS

B DUP I=*I*

I-node *I* claims block *B*, but this block is already on fsck's list of allocated blocks.

PHASE 2: CHECK PATHNAMES

ROOT I-NODE UNALLOCATED. TERMINATING

The root directory of a file system is always linked to i-node 2. If i-node 2 is not allocated, the file system is damaged beyond repair.

ROOT I-NODE NOT DIRECTORY (FIX)

I-node 2 must be a directory.

DUPS/BAD IN ROOT I-NODE (CONTINUE)

Some of the duplicate blocks belong to i-node 2. This will make it very difficult to repair the file system.

I OUT OF RANGE I=I NAME=F (REMOVE)

F, directory entry, refers to an i-node that doesn't exist.

UNALLOCATED I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F (REMOVE)

The specified directory entry is a link to an unallocated i-node.

DUP/BAC I=I OWNER=O MODE=M SIZE=S MTIME=T

You specified the -q option and fsck spotted inconsistent data in the specified directory.

PHASE 3: CHECK CONNECTIVITY

UNREF DIR I=I OWNER=O MODE=M SIZE=S MTIME=T (RECONNECT)

The indicated directory is nonempty and uncorrupted but lacks a directory entry (its former parent has no link to it).

SORRY, NO lost+found DIRECTORY

No files can be reconnected until you replace the missing lost+found directory. If you really need to reconnect your unreferenced i-nodes: first, finish this fsck run (being careful not to clear any i-nodes!); then recreate the missing lost+found directory (see "Creating the lost+found Directory," above); and finally run fsck on the file system again.

DIR I=I1 CONNECTED. PARENT WAS I=I2

The directory whose i-number is I1 now has a link in lost+found. fsck has made the directory's ".." entry refer to lost+found; formerly, ".." referred to i-node I2.

PHASE 4: CHECK REFERENCE COUNTS

UNREF FILE I=I OWNER=O MODE=M SIZE=S MTIME=T (RECONNECT)

The indicated ordinary file is nonempty and uncorrupted but lacks a directory entry (the directories that had links to it lost them).

SORRY, NO lost+found DIRECTORY
SORRY, NO SPACE IN lost+found DIRECTORY

No files can be reconnected until you replace the missing lost+found directory. If you really need to reconnect your unreferenced i-nodes: first, finish this fsck run (being careful not to clear any i-nodes!); then recreate the missing lost+found directory (see "Creating the lost+found Directory," above); and finally run fsck on the file system again.

(CLEAR)

A chance to abandon the last UNREF file without rerunning fsck. Be absolutely sure you want to clear this i-node.

LINK COUNT FILE I=I OWNER=O MODE=M SIZE=S MTIME=T COUNT=X SHOULD BE Y (ADJUST)

The link count for the directory is X but Y files actually have links to it.

UNREF DIR I=I OWNER=O MODE=M SIZE=S MTIME=T (CLEAR)
UNREF FILE I=I OWNER=O MODE=M SIZE=S MTIME=T (CLEAR)

Ordinarily, unreferenced and empty files and directories silently disappear. If the -n option is specified, this prompt appears for empty files and directories.

BAD/DUP DIR I=I OWNER=O MODE=M SIZE=S MTIME=T (CLEAR)
BAD/DUP FILE I=I OWNER=O MODE=M SIZE=S MTIME=T (CLEAR)

The specified directory or file is unreferenced and has bad or duplicate blocks.

FREE I-NODE COUNT WRONG IN SUPERBLK (FIX)

fsck's count of free i-nodes doesn't match the count in the superblock.

PHASE 5: CHECK FREE LIST

EXCESSIVE BAD BLKS IN FREE LIST (CONTINUE)
EXCESSIVE DUP BLKS IN FREE LIST (CONTINUE)

This is your last chance to avoid reconstructing the free list.

BAD FREEBLK COUNT
X BAD BLKS IN FREE LIST
X DUP BLKS IN FREE LIST
X BLKS MISSING

Final notes on the dire state of the free list. Any of these will invoke the BAD FREE LIST message.

FREE LBK COUNT WRONG IN SUPERBLOCK (FIX)

fsck's count of free blocks does not match the value in the superblock.

BAD FREE LIST (SALVAGE)

If the file system is otherwise all right, it's always a good idea to salvage the free list.

Rebooting the System

The kernel can undo all your painful repair work by writing over its copies of file system tables. Give the kernel no chance to do I/O to the file systems until you return to multiuser state or halt the processor:

- Unmount all file systems (except the root file system, of course). Keep them unmounted.
- Under no circumstances run sync from the time you start repairing the file systems to the time you switch to multiuser state or halt the processor.

Section 5
User Support

ADDING USERS WITH USER.CNFG

You can add and remove users to and from the system automatically using the utility named /etc/user.cnfg (as described below). User.cnfg modifies /etc and makes directories. You can do these things manually by following the procedures described under "Adding Users Manually" later in this section.

To run the user.cnfg program, you must be logged in as root. Then follow these steps:

a. enter the command

```
/etc/user.cnfg
```

The main menu prints on the screen.

```
Mon, Aug 5                16:46:13

1 - Add a user to the system
2 - Delete a user from the system
3 - Display passwd entry for a specific name
4 - Display all of passwd & group files
<cr> - Exit this program

cmd(1-4,<cr>):
```

The day, date and time are printed at the top of the screen for your convenience.

At this point you can enter a number in the range 1 to 4, or press RETURN. Any other input is invalid.

b. To add a user, coose option 1. You get this prompt:

```
Enter user's name to add: <new user>
```

If you enter a name that is already in the password file, then you get the following message, and return to the main menu after pressing RETURN.

```
<new user> is already in the password file

hit <cr>
```

For a valid new user name, the system prompts:

```
Enter comment:
```

This is for the comment section of the password file. Enter a comment for the new user (such as Ima New user). Next the user id for the new user is printed for your information:

```
UID for <new user>: 110
```

The group ID is the next field to fill in. A group is a set of users that are working on the same project and need to be able to share files. A table of the group names and numerical values is printed:

```
The group names and id's are:
     root      0
     other     1
     bin       2
     sys       3
     adm       4
     mail      6
     sccs      7
     rje       8
     daemon    12
     user      101
```

The default group ID is the same as the previous last entry group ID entered. Verification of the group ID is then prompted for:

```
Is group id of "user" (101) ok (y/n):
```

If you answer no at this point, you are asked for the correct group:

```
Enter the correct group id name:
```

You must enter the character string for the ID value you want. For example, if the new user should have the group ID of 7, enter sccs.

c. The next item is the home directory for the new user. You will see the message:

```
The default directory for users is /user
Is this ok (y/n):
```

If /user is acceptable, answer yes. Otherwise, answer no, and answer the next question.

```
Enter the full path name for the user's directory:
```

The input for this should be of the form /dir, not /dir/username. The user name is appended to the string. The directory you enter should already exist on the system, or errors will occur when the mkdir command is executed.

If a system directory is entered, such as /bin, /usr, /etc or /lib, an error message is printed:

```
     Invalid path, /bin is a system directory

     Enter a different full path name:
```

Just reenter a valid directory name.

d. After all the data has been entered, user.cnfg generates the password file entry and prints it out:

```
     now adding <new user> to /etc/passwd file:
          <new user>::110:Ima New User:/d2/<new user>:/bin/nsh
```

Because there is no password yet assigned for this entry, passwd(1) is run and will require a password for the user:

```
     Please enter a password for <new user>
     Changing password for <new user>
     Please enter password:

     Re-enter password:
```

e. At this point all the administrative input is through. User.cnfg continues to install the new user:

```
     now making <new user> home directory

     <new user> is now entered into the system

     hit <cr>
```

The main menu reappears when you press RETURN.

f. To remove a user from the system, enter number 2 from the main menu. Note that you must backup all files of the user to be deleted. Once their home directory is gone, so are all their files.

When you select option 2, you get this prompt:

```
     Enter name to remove: <user name>
```

Enter the user name (found as the first entry in the /etc/passwd file). If you enter a valid name, the following messages are printed, and you are returned to the main menu.

```
     removing <user name> from /etc/passwd file
     removing <user name> home directory

     <user name> is no longer on the system

     hit <cr>
```

g. If you just want to see the passwd entry for a specific user, choose option 3. You'll get the following prompt:

```
Enter name to display: <user name>
```

This name is used as a search pattern and when found the complete entry is printed:

```
<user name>:/71iI5KZkJ8qg:110:7:Ima New User:/d2/<user name>:/bin/nsh

hit <cr>
```

When you press RETURN, the main menu reappears.

h. If you would like to see all of the /etc/passwd and /etc/group files, choose option 4. The are printed to the screen by the more(1) command.

```
::::::::::::::
/etc/passwd
::::::::::::::
root:faCpyMSk1QQvE:0:0:Root:/:/bin/nsh
daemon:NONE:1:1:Admin:/:
bin:NONE:2:2:Admin:/bin:
sys:NONE:3:3:Admin:/usr/src:
adm:NONE:4:4:Admin:/usr/adm:
uucp:NONE:5:1:uucp:/usr/lib/uucp:
nuucp::6:1:uucp:/usr/spool/uucppublic:/usr/lib/uucp/uucico
sync::20:1:S6300 sync command:/:/bin/sync
lp:NONE:71:2:lp Administrator:/bin:
isam:NONE:96:3:ISAM Administrator:/user/isam:
sccs:NONE:97:7:SCCS Pools:/source:
gtdl::98:101:Gt RS-232 auto-download:/:/usr/local/bin/gtdl
GTDL::98:101:Gt RS-232 auto-download:/:/usr/local/bin/gtdl
newuser:/71iI5KZkJ8qg:110:7:Ima New User:/d2/<user name>:/bin/nsh
::::::::::::::
/etc/group
::::::::::::::
root:NONE:0:root
other:NONE:1:
bin:NONE:2:root,bin,daemon
sys:NONE:3:root,bin,sys,adm
adm:NONE:4:root,adm,daemon
mail:NONE:6:root
sccs:NONE:7:root,sccs
rje:NONE:8:rje,shqer
daemon:NONE:12,root,daemon
user:NONE:101:

hit <cr>
```

When you press RETURN, controll returns to the main menu. Press RETURN again to exit the program.

ADDING USERS MANUALLY

The following actions give a new user basic access to the system; note that you can do these things automatically by running user.cnfg, as described under "Adding Users With User.cnfg" earlier in this section.

a. Assign the user a unique log-in name.

b. Choose a file system and home directory to hold the user's files.

c. Create an entry for the user in the password file.

d. Create the user's home directory.

Log-In Names

The log-in name uniquely identifies the user. The log-in name must be one to eight characters long and consist of letters or digits. If the name has letters, at lease one of them must be lower case. Two users must not have the same log-in name.

Choosing a File System and Home Directory

Decide which file system will hold the user's permanent files. If possible, reserve one or more file systems for non-administrative users. Divide the file systems between users based on your estimate of the users' storage needs.

Name user home directories in a way that is convenient for you. One good system follows two conventions:

- The user's home directory is in the root directory of the user's file system.

- The simple name of the home directory is the same as the user's log-in name.

Decide on the name of the user's home directory, but don't create the actual directory yet. This will be easier to do once you have created the user's password file entry.

Example:

In this example, four new users choose log-in names "john," "dick," "gwen," and "jack." The system administrator decides that there is room for them on the file system whose special file name is /dev/fp003. This file system is normally mounted on /user. The new users' home directories will be /user/john, /user/dick, /user/gwen, and /user/jack.

The Password File

Each user requires an entry in the password file, /etc/password. This file is read by login every time someone tries to log in. Use the following procedure to add a new user to the password file:

a. Use a text editor to add the user's entry to the file.

b. Use the passwd program to assign a password to the new user.

Each entry in the password file is a line of the form

```
    name:pass:uid:gid:unused:home:shell
```

where

name is the user's log-in name.

pass is user's encrypted password. Leave this field blank; it will be filled in when you run the passwd program.

uid is the user's unique numeric user ID. This must be a decimal number and is generally greater than or equal to 100.

gid is the user's initial numeric group ID. To implement groups, see *group*(4) in the *Series 6000 Operating System Reference Manual*. If the user is not associated with any group, set this field to 100.

unused is a field without any standard use. It often holds the user's name and office location.

home is the name of the user's home directory.

shell is the full path name (not the command name) of the user's shell, the program that is executed when the user logs in. If this field is empty, login uses the Bourne shell /bin/sh.

To specify a password for the new user, run passwd. The form of the command is

```
    passwd name
```

where

name is the user's log-in name.

passwd prompts for a password. The password does not appear on the screen as you type it in. To prevent error, passwd makes you type the password twice. If you run passwd as an ordinary user, then the password must be at least six characters long and contain at least two alphabetic characters and at least one numeric or special character. If you run this command as the superuser, then the restrictions on the password are minimal.

Example:

In this continuing example, the system administrator creates password file entries for "john," "dick," "gwen," and "jack." Each of these users uses the Bourne shell. The system administrator also invents a user that exists solely to allow people to run sync without logging in. The administrator's input is shaded, and the computer's response is in normal type.

```
# ed /etc/passwd
5328
$p
frank:UCOW7.pjZUBcw:115:100::/a/frank:
a
john::116:100::/user/john:
dick::117:100::/user/dick:
gwen::118:100::/user/gwen:
jack::119:100::/user/jack:
sync::120:100::/:/bin/sync
.
w
5460
q
# passwd john
new password:
retype new password:
# passwd dick
new password:
retype new password:
# passwd gwen
new password:
retype new password:
# passwd jack
new password:
retype new password:
#
```

The administrator has kept the password file ordered by numerical user ID, so it's only necessary to examine the last entry to determine the next numerical user ID.

The "sync" user requires no password and no home directory of its own.

User Home Directory

The new user's home directory requires the following steps.

a. Create the directory.

b. Give the user ownership of the directory.

c. Give the user's group group ownership of the directory.

d. Set the protection mode of the directory.

Use mkdir to create the home directory. The command has the form

```
    mkdir dir
```

where

dir is the home directory name. Actually, any number of directory names is permitted.

Use chown to give the user ownership of the home directory. The command has the form

```
    chown name dir
```

where

name is the user's log-in name.

dir is the user's home directory.

Use chgrp to give the user's group group ownership of the home directory. The command has the form

```
    chgrp group dir
```

where

group is the name of the group or the numerical group ID.

dir is the user's home directory.

Use chmod to set the protection mode of the user's home directory. The command has the form

```
    chmod mode dir
```

where

mode is a protection mode. The value 755 gives the owner all access to the directory and gives other users read-only access. The value 700 gives the owner all access to the directory and gives other users no access at all. For other modes, see *chmod*(1) in the *Series 6000 Operating System Reference Manual*.

dir is the user's home directory.

Each of these commands can do multiple files or directories.

Example:

In this continuing example, the system administrator now provides home directories for "john," "dick," "gwen," and "jack." The administrator makes each directory with all access for the owner and read-only access for other users. The administrator's input is shaded, the computer's response in normal type.

```
    # mkdir /user/john /user/dick /user/gwen /user/jack
    # chown john /user/john
    # chown dick /user/dick
    # chown gwen /user/gwen
    # chown jack /user/jack
    # chgrp 100 /user/john /user/dick /user/gwen /user/jack
    # chmod 755 /user/john /user/dick /user/gwen /user/jack
    #
```

BARRING, RESTORING, AND DELETING USERS

This section describes procedures for denying users access to the system. The subsection "Barring and Restoring a User" tells how to deny access when the user may not be removed permanently and how to restore access. The subsection "Permanently Removing a User" tells how to completely undo the steps that gave the user access and remove the user's files.

Barring and Restoring a User

To bar a user without permanently removing him or her, invalidate the user's password file entry. One way to do this is to insert a % at the beginning of the entry's encrypted password; use a text editor to do this.

When a user's password file entry is invalid, any attempt by that user to log in is rejected as "incorrect."

To restore the user, remove the %.

Example:

At this point, the system administrator bars "jack." The administrator's input is shaded, and the computer's responses in normal type.

```
    # ed /etc/passwd
    7454
    /jack/
    jack:wcBUZjp.7WOCU:119:100::/user/jack:
    s/:/:%/p
    jack:%wcBUZjp.7WOCU:119:100::/user/jack:
    w
    7455
    q
    #
```

Permanently Removing a User

It is a good idea to postpone permanent removal of a user until after regular file backups. If this is inconvenient, consider using the temporary procedure, above, until the next regular backup.

To remove a user from the system permanently:

a. Remove the user's password file entry.

b. Remove the user's files

Use the text editor to remove the password file entry.

Rm will remove the user's home directory and all the files it contains. The command takes the form

```
rm -fr dir
```

where

dir is the user's home directory.

CAUTION

> The above form of the rm command can remove a large number of directories quickly. Note that the rm command does not announce the files it is removing. Use this command carefully.

If a user's file outlasts the user's password file entry, an ls -l on the file produces the former user's numeric user ID.

The former user may leave files in places other than his home directory. The following command does a comprehensive search for the former user's files.

```
    find / -user x -print
```

where

x is the user's log-in name or numeric user ID. If the user's password file entry is gone, the log-in name will not work but the numeric user ID will.

Example:

Next, the system administrator of this continuing example removes "jack" and his files.

```
    # ed /etc/passwd
    7530
    /jack/
    jack:%wcBUZjp.7WOCU:119:100::/user/jack:
    d
    w
    7490
    q
    # rm -rf /user/jack
    #
```

MOVING USERS

If space runs short on a file system, you may need to move a user to a new file system. Moving a user requires the following steps.

a. Inform the user of his or her new home directory name.

b. Copy the user's files to their new location.

c. Update the user's password file entry.

d. Delete the user's old files.

Use the same system for assigning moved home directory names that you use for assigning new home directory names.

The following command will create the file copies. The copies will have the same modification dates as the originals; this is desirable if the user uses make.

```
    cd old ; find . -depth -print ¦ cpio -pdm new
```

where

old is the full name of the old directory.

new is the full name of the new directory.

To remove the old user files, use rm:

```
    rm -rf old
```

where

old is the full name of the old directory.

Example:

Lastly, the system administrator from the previous examples wants to move "frank" from the root file system to file system mounted on /user. The system administrator's input is shaded; the computer's responses are in normal type.

```
    # ed /etc/passwd
    7000
    /frank/
    frank:UCOW7.pjZUBcw:115:100::/frank:
    s/\//\/user\//p
    frank:UCOW7.pjZUBcw:115:100::/user/frank:
    w
    7000
    q
    # cd /frank; find . -depth -print ¦ cpio -pdl /user/frank
    # rm -r /frank
    #
```

Section 6
Backups and Restores

Regular backup provides copies of files and file systems for protection against accident, carelessness, and technical mishap.

SCHEDULING BACKUPS

The following sample schedule has the basic features of a good backup schedule.

- Permanent total. Every fourth Friday, each file system is completely copied. The copies are saved permanently.

- Temporary total. Every Friday, except on days when a permanent volume backup is done, each file system is completely copied. The copies are saved for four weeks.

- Incremental. Every working day, except on days when total backups are done, all files created or modified since the last total backup are copied. The copies are saved for a week.

Backups occur at the end of the working day.

Some of the features of this schedule are arbitrary, some are not. Every four weeks may be too often for you to make permanent backups; but if you increase the time between permanent total backups, make the same increase in the time you keep temporary total backups. Total backups need not occur on Fridays, but should occur at the same time each week; backups need not occur at the end of the working day, but the time they do occur should not change from day to day.

The most important feature of this schedule is that it does not permit the loss of more than a day's work due to the complete loss of the fixed disk's files. It also protects files against accidental removal: the longer a file is left on the fixed disk, the harder it is to lose it permanently.

DOING BACKUPS

You can back up a file-system using diskettes or quarter-inch tape. The following steps are required to back up a file system. When doing a backup, you should be logged in as root and the file system should be unmounted. Having the file system unmounted before the backup ensures that no file changes occur during the backup, but remember that the root file system can't be unmounted. The file system should be unmounted during step "a" of the following procedure, taking the system to single-user mode.

a. Take the system to single-user mode, as described in Section 2.

b. If you are using diskettes, prepare enough data diskettes using the procedures in Section 4. Each diskette should have a minimal partition 0 and a partition 1 that takes up the remainder of the diskette. Do not create a file system in partition 1.

If you are using tapes, retension each tape the first time you use it. To retension a tape, insert it in the drive and type:

```
tsioctl -c retension /dev/rmt0
```

c. Use volcopy or cpio (recommended) to perform the backup.

d. Print out a log of the files backed up, as described later in this section.

e. If this is a total backup, register the time for the benefit of this week's incremental backups.

There are two distinct backup procedures:

- The total backup. Each fixed disk file system is separately copied onto a tape or a set of diskettes.

- An incremental backup. A list of files modified since the last total backup is prepared and each file on the list is copied to backup floppy diskettes.

Total Backups

The labelit and volcopy commands accomplish a total backup of a single file system. The general form of each command is the same whether the backup medium is tape or diskette; however, since the options and device names differ, the commands are given once for each backup medium. The definitions of *ldir*, *t*, *p*, and *vname* are the same in each case.

The system should be in single-user mode before starting volcopy, and all partitions, except the root partition, must be unmounted before the volcopy can be performed.

Note that the root file system is already labeled. When you backup the root file system, use its current *ldir* name and *vname* (these are parameters in the labelit and volcopy commands). To find out what the current names of a labeled file system are, type

```
/etc/labelit /dev/rfp0tp
```

where

t indicates the disk that holds the file system: 0 for the first fixed disk, 1 for the second fixed disk, and 2 for the removable disk.

p is the hexadecimal number of the partition that holds the file system.

Using diskettes:

To do a total backup, insert the first backup diskette into the drive and execute the labelit command. Label all diskettes you will need.

```
     /etc/labelit -t /dev/rfp021 ldir backup
```

where

ldir is the local name of the directory on which the file system is normally mounted, or root for the root file system.

Example:

A file system normally mounted on /user is user.

Insert the first labeled diskette into the drive and run volcopy:

```
    /etc/volcopy -to ldir /dev/rfp0tp vname /dev/rfp021 backup
```

where

ldir is the local name of the directory on which the file system is normally mounted or root for the root file system.

Example:

A file system normally mounted on /user is user.

t indicates the drive number of the disk with the partition to be backed up.

p is the number of the partition on the disk to be backed up. The number is hexadecimal (a through f stand for "10" through "15").

vname is your name for the disk that holds the file system.

When a diskette is used up, the system prompts:

```
     Safe to remove the exchangeable disk /dev/rfp020
     Mount disk 2
     Type volume-ID when ready:
```

Insert the next diskette and type the volume name specified in the labelit command, backup.

Using quarter-inch tape:

To do a total backup onto quarter-inch tape, insert the tape cassett into the tape drive and execute the following commands:

```
     /etc/labelit /dev/rmt0 ldir backup -n
     /etc/volcopy -Q ldir /dev/rfp0tp vname /dev/rmt0 backup
```

GENERATING A LOG OF BACKED UP FILES

To generate a log of files backed up, remount the file system and use ff (See the UNIX-derived operating system *Software Release Guide* (SRG) shipped with your system) redirecting output to a file:

```
/etc/mount /dev/fp0tp ldir
/etc/ff -p ldir -s -u /dev/fp0tp > /etc/log/backup
```

where

ldir is the name of the directory on which the file system is normally mounted.

t is the number of the hard disk; 0 for the first hard disk, and 1 for the second hard disk.

p is the hexadecimal partition number.

The log will appear shortly on the system printer. This log contains important information for file restoration, so keep it in a safe place.

Use the modification time of a file to register the time of a total backup. Enter the following command after each total backup:

```
> /etc/log/TOTAL
```

This creates a file who's date of creation is the same as the date of the total backup you just did.

Incremental Backups

An incremental backup copies some files from all fixed disk file systems. This is different from a total backup, which copies all files from specified file systems. The following steps accomplish an incremental backup.

a. Remount the mountable file systems in their normal place.

b. Generate a list of files modified since the last total backup.

c. Copy every file in the list to backup diskettes or tape.

d. Print the list.

e. Unmount the file systems.

REMOUNTING THE MOUNTABLE FILE SYSTEMS

The mountable file systems were unmounted when you took the operating system to single-user mode. To remount them, execute the startup mounting procedure:

```
sh /etc/mountable
```

GENERATING A LIST OF FILES TO COPY

Use the find command to generate a list of recently modified files:

```
    find dir -newer /etc/log/TOTAL -print | sort > /etc/log/INCd
```

where

dir is the directory to be searched for modified files.

d is the number of days since the total backup.

COPYING RECENTLY MODIFIED FILES

Use the cpio command to copy the recently modified files to the archive.

Using diskettes:

Insert the first diskette in the drive and enter:

```
    cpio -ovB < /etc/log/INCd > /dev/rfp021
```

where

d is the number of days since the total backup.

When cpio uses up the diskette, it tells you it is safe to remove the device and insert the next one in the series.

Remove the diskette and insert the next one in the set. You are prompted as each of the diskettes is filled, until all the files are copied. Be careful in using this procedure, since terminating cpio by accident means that you must start over, beginning with the first diskette. For example, be sure that you put the diskette in the drive with the write-protect notch down; otherwise, you will terminate cpio and need to run it again.

Using quarter-inch tape:

Insert the tape cartridge in the drive and do:

```
     cpio -ovQ < /etc/log/INCd > /dev/rmt0
```

where

d is the number of days since the total backup.

NOTE

The Q option with cpio does not work with operating system releases previous to FE07.

GENERATING A LOG

To print out the log of files backed up, enter:

```
    xargs ls -ld < /etc/log/INCd ¦ lpr
```

where

_d_ is the number of days since the last total backup.

UNMOUNTING THE FILE SYSTEMS

Unless you plan to go directoy back to multiuser mode, be sure to unmount the file systems again. The surest way to do this is to run halt:

```
     /etc/halt
```

The Backup Log

The log printout is different for a total backup and an incremental backup. The total backup log lists four data on each file backed up.

- The full file name. The first part of this name is the name of the directory on which the file's file system is mounted.
- The file's i-number. The i-number is unique for each file on a file system, but not for each file name. If two file names list the same i-number, they are two links to one file.
- The file's size, in bytes.
- The login name of the file's owner.

The incremental log uses the format of the ls command. See ls(1) in the _Series 6000 Operating System Reference Manual_.

RESTORES

A mishap can destroy an entire file system or just a few files. Restoring an entire file system requires copying the file system from the last total backup, then copying each of the subsequent incremental backups. Restoring specific files simply means copying those files from the latest backups that have them.

Restoring an Entire File System

The following steps completely restore a file system:

a. Take the operating system to single-user mode, as described in Section 2.

b. Copy the file system from the total backup.

c. Unmount the file system.

d. Retrieve the file system's files from the incremental backups.

NOTE

> Be sure that the file systems are unmounted with the umount command before you execute volcopy. If the file systems are mounted, then running volcopy causes all data from the backup to be lost. This problem can be corrected by unmounting the file system and running volcopy again to restore the data.

RESTORING FROM THE LAST TOTAL BACKUP

You can restore from diskette or tape.

Using diskette:

Put the diskette in the drive and type:

```
/etc/volcopy -ti ldir /dev/rfp021 backup /dev/rfp0tp vname
```

where

ldir is the local name of the directory on which the file system is normally mounted, or root for the root file system.

Example:

A file system normally mounted on /user is user.

t is the number of the hard disk to which you are restoring (0 is the first hard disk, 1 is the second hard disk).

p is the number of the partition on the disk to receive the "restore." The number is hexadecimal (a through f stand for "10" through "15").

vname is your name for the disk that holds the file system.

Note the reversal in parameters from the command that backed up the file system. Restoring a file system this way completely rewrites file system data structures, using the version on the backup diskettes. If the file system was malfunctioning before the restore, using volcopy to restore it eliminates current problems, but restores any problems it had at the time of the total backup.

Using quarter-inch tape:

Insert the tape cartridge in the drive, and type the following command. Note that the options are the same as described for use with diskettes.

```
/etc/volcopy -Q ldir /dev/rmt0 backup /dev/rfp0tp vname
```

RESTORING FROM INCREMENTAL BACKUPS

To completely restore the incrementally backed-up files, restore each incremental backup for the file system. Restore the oldest backup first, but do not restore any backup made before the last total backup.

This procedure restores an incremental backup. First, mount the file system:

```
sh /etc/mountable
```

Using diskettes:

Insert the first diskette of the incremental set in the drive and type

```
cpio -iBduvm 'ldir/*' < /dev/rpf021
```

where

ldir is the name of the directory on which the file system is normally mounted.

When cpio reads through the diskette, it tells you it is safe to remove the disk and insert the next one in the series.

Remove the diskette and insert the next one in the set. You are prompted as each of the diskettes is finished, until all the files are copied. Be careful in using this procedure, since terminating cpio by accident means that you must start over, beginning with the first diskette. For example, be sure that you put the diskette in the drive with the write-protect notch down; otherwise, you will terminate cpio and need to run it again.

Using quarter-inch tape:

Insert the cartridge into the drive and type:

```
    cpio -iQduvm 'ldir/*' < /dev/rmt0
```

Restoring Specific Files

Backup diskettes made with volcopy require a different restoration procedure than those made with cpio. It is best to have the system in single-user mode to prevent use of files that are being restored, although it isn't required. Note that if the files are used during the restoration process, file damage could occur. The file systems must be mounted; in single user mode the simplest way to make sure that all file systems are mounted is to type:

```
    sh /etc/mountable
```

RESTORING SPECIFIC FILES FROM TOTAL BACKUP

The frec command restores individual files from backup diskettes made by volcopy:

```
    /etc/frec devicefile i:name ...
```

where

devicefile is the type of removable medium: /dev/rmt0 for tape, /dev/rfp021 for diskette.

i is the i-number for the file. File i-numbers are contained in the log of backed-up files created in the subsection on "Total Backups."

name is the full pathname the file will have when it is restored. This is normally the same name it had when it was backed up, but need not be.

... indicates additional files for recovery. Each argument takes the i:name form and is separated from other parameters by spaces.

Frec will prompt you to insert the backup tape or diskettes.

Do all recoveries from a single backup tape or set of backup diskettes with one frec run. If a large number of files are to be recovered, use this procedure:

a. Use a text editor to create a file that lists all the files to be recovered. Each line in the file must be of the form

```
i:name
```

where

_i_ is the i-number for the file. File i-numbers are contained in the log of backed-up files created in the subsection on "Total Backups."

name is the name the file will have when it is restored. This is normally the same name it had when it was backed up, but need not be.

b. Run frec:

```
/etc/frec -f file devicefile
```

where

file is the name of the file created in the first step.

devicefile is /dev/rmt0 for tape, or /dev/rfp021 for diskettes.

If a missing file or directory to be restored was in a directory that is also missing, restore the parent directory also. If you fail to do this, frec will recreate the missing directory, but it will be a new directory and may not have the same ownership and modes as the original.

RESTORING SPECIFIC FILES FROM INCREMENTAL BACKUP

Restoring specific files from incremental backup is very similar to restoring an entire file system from incremental backup.

Using diskettes:

Insert the first diskette in the set in the drive and type:

```
cpio -iBduvm list < /dev/rfp021
```

where

list is a list of files to be recovered.

Using quarter-inch tape:

Insert the cartridge into the drive and type:

```
     cpio -iQduvm list < /dev/rmt0
```

If you need to recover a large number of specific files, create a text file with all the file names in it, and use the following forms of the cpio command instead of the ones above:

Using diskettes:

```
    cpio -iBduvm `cat file` < /dev/rfp021
```

where

file is the file that contains a list of files to be recovered.

Using quarter-inch tape:

```
     cpio -iQduvm `cat file` < /dev/rmt0
```

Note that the second form of the cpio command uses grave accents (`). Do not confuse them with single quotes (').

As with recovering the entire incremental backup, recovering specific files requires intervention each time cpio reads through a diskette. See the previous subsection, "Restoring an Entire File System," for what to do.

Appendix A
File System Concepts

The operating system provides a standard way for programs to use peripheral devices. Each special file represents a particular way to access a particular peripheral. A special file appears on the file system (by convention, in /dev) and ordinary input/output operations on special files have standard meanings standard for the peripheral.

Special files are either block or character. Block special files identify kernel routines that are most efficient with input/output (I/O) operations precisely 1024 bytes long. Character special files identify kernel routines that don't prefer any particular size operation. Some kinds of peripherals are represented by both block special files and character special files.

The operating system is normally set up with 32 special files for each disk drive, providing a block special file and a character special file for each possible partition. The name of a disk's special files takes the form

```
    /dev/rfp0tp
```

where

r is missing on block special files and is r (for "raw") on character special files.

t indicates the particular drive: 0 for the first fixed disk, 1 for the second fixed disk, and 2 for diskette.

p is the partition number in hexadecimal.

If a disk has fewer than 16 partitions, it is an error to use the special files for the nonexistent partitions.

SECTORS AND BLOCKS

A block is the basic unit of disk (I/O). There are two kinds of blocks:

- Physical sector, which is a physical entity 512 bytes long. A disk drive's basic access to the disk reads or writes a physical sector.
- Logical block, which consists of two physical sectors (1024 bytes). An I/O operation involves one or more logical blocks (never a fraction). Using two physical sectors per logical block improves system I/O performance.

The utilities that initialize the disk, create the file systems, and report on disk sizes consider blocks to be synonymous with physical sectors. Most programs, however, including fsck, consider the basic unit to be the logical block. In the remainder of this appendix, a "block" is a logical block.

DIRECTORIES

A directory is simply a file that only the operating system kernel can write to. Each directory entry consists of the file name and the file's i-number. A file can have more than one directory entry (*link*). The number of directory entries that refer to a file is that file's *link count*.

FILE SYSTEM FORMAT

File system is a storage area (normally a disk partition) with the following structures.

- A block reserved for use in booting the operating system.
- The *super block*, containing data structures that describe the file system.
- The *i-list*. This is a sequence of records, called *i-nodes*, that describe the operating system files. The size of the i-list is fixed when the file system is created. Each i-node has an *i-number* that gives the i-node's place in the i-list. All file status information is in the i-node, as are the direct and indirect pointers to the file's data blocks.
- The *free list*. This is a linked list of blocks not used by any file. Each element of the free list is a block that contains pointers to 50 additional blocks.

The program that creates these structures also creates a directory that is the first file on the file system. This directory is the *root* of the file system.

Two structures in an i-node are important to the administrator: the *link count* and the *disk address*.

The link count is an integer value. It is 0 when the i-node is not in use. Creating a file sets the link count to 1. Each additional directory entry (link) for the file increments the link count; each removal of a directory entry decrements the link count. If the link count returns to 0, the file's blocks are returned to the free list—the file is removed.

There are 13 disk addresses in the i-node. The first 10 point to the first 10 blocks of the file (the *direct blocks*). If the file is more than 10 blocks long, the 11th address points to a block that has pointers to the next 256 blocks of the file (the *indirect blocks*). If the file is more than 266 blocks long, the 12th address points to a block that points to up to 256 blocks containing pointers to the next 65,536 blocks of the file (the *double-indirect blocks*). If 65,802 blocks aren't enough, the 13th address provides access to *triple-indirect blocks*.

These data structures can become inconsistent through incomplete I/O operations, usually those caused by a power failure or through halting the system while the operating system is running. One of the administrator's jobs is to repair file system data structures using the maintenance programs described in Section 4.

A mount places a file system on the file system hierarchy. A mount specifies an empty directory and the special file that holds the mountable file system. A mount tells the operating system that any reference to the specified directory is really a reference to the directory of the file system. The directory on which a file system is mounted can itself be on a mounted file system, but naturally the parent file system must be mounted first.

The root file system (the file system whose root is /) is, in effect, always mounted. It is the only file system that has no parent file system.

The term "file system" actually has two meanings. A file system can refer either to the organization imposed on a single partition ("the file system mounted on /a") or the whole disk hierarchy of files ("the operating system file system").

CAUSES OF FILE SYSTEM CORRUPTION

File system corruption is caused by incomplete or garbled I/O instructions. That can be the result of any of the following:

o Improper shutdown. In particular, all I/O must be complete before the processor is halted. To assure completeness of I/O, kill all user processes and perform two syncs. All these procedures are contained in the shell script /etc/shutdown.

o Use of a corrupt file system. This causes further errors because of the incorrect file system structures.

o Hardware failure.

fsck AND THE FILE SYSTEM

fsck detects errors in three areas:

o The superblock.

o The i-nodes.

o Directory data.

fsck checks the following in the superblock:

- File system size and i-list size. The file system must be bigger than the superblock plus the i-list. There must not be more than 65,534 i-nodes.

  fsck relies heavily on these two data. Except to check that they are reasonable values, there is no way to confirm their correctness. All other checks depend on the correctness of the file system and i-list sizes.

- Free block list. The first block in the list is in the superblock. Each block in this list contains pointers to additional free blocks. Each block's count of pointed-to-blocks must not be less than 0 or greater than 50. Each block pointer must not point past the end of the file system or before the first data block. No block in the free list can be in fsck's list of blocks claimed by the i-nodes.

  If fsck finds errors in the free list, or if it can't account for every block in the file system, it will ask for permission to reconstruct the free list. The new free list will include all blocks not claimed by any i-node. In the absence of any other serious errors, rebuilding the free list is always safe.

- Free block count. If this does not agree with the actual number of free blocks, fsck asks permission to reset the count.

- Free i-node count. If this count is not the same as the size of the i-list minus the number of i-nodes in use, fsck asks for permission to reset the count.

fsck checks the following fields in each i-node:

- Format and type. These fields specify the kind of file (ordinary, directory, block special, character special) and the i-node status (allocated or unallocated). Invalid values indicate that bad data have been written into the i-list. fsck will prompt for permission to clearthe i-node; this is always unavoidable.

- Link count. This value must equal the number of directories that actually list the i-node. An inconsistency here indicates a failure to update a directory or the i-node; this is always a minor error.

  If the i-node's link count and the number of links are unequal and both are nonzero, fsck asks permission to correct the i-node link count.

  If the i-node link count is nonzero and the actually link count is zero, fsck asks permission to provide a link in the file system's lost+found directory.

- Duplicate blocks. These are blocks claimed by more than one i-node. fsck spots duplicate blocks as it builds its list of allocated blocks; this condition requires a second pass of the i-list to find the first i-node. Then fsck tries to suggest which i-node should be cleared; usually this is the one with the earlier modification time.

  A large number of duplicate blocks probably indicates that the operating system failed to physically write out a block of pointers to indirect blocks. fsck asks for permission to clear both i-nodes.

- Bad blocks. These are blocks that cannot be found because their addresses are invalid.

  If an i-node has a large number of bad blocks, the operating system probably failed to write out a block of pointers to indirect blocks. fsck asks for permission to clear the i-node.

- File size. Two kinds of errors can appear here: block allocation consistency and proper directory size.

  fsck computes the number of blocks required to accomodate a file of the indicated size. If this value doesn't match the number of blocks the file actually has allocated, fsck prints a warning. Note that this condition may be the result of a program seeking past the end of a file before writing to the file, a perfectly valid action.

  If the file is a directory, the file size should be a multiple of 16. If it is not, fsck prints a warning but takes no action.

fsck looks for the following errors in directory data:

- Reference to unallocated i-nodes. This probably is the result of the operating system's failure to write out a modified i-node. fsck requests permission to remove the directory entry.

- Invalid i-number. This probably is the result of bad data output to the directory. fsck requests permission to remove the directory entry.

- Incorrect "." and ".." entries: "." must be the first entry in the directory and have an i-number equal to the i-number for the directory itself; ".." must be the second entry in the directory and be a link to the directory's parent directory. If these entries are incorrect, fsck asks for permission to correct them.

Appendix B
init and getty

In the UNIX-derived operating system environment, spawning the initial process is controlled and overseen by the first process forked by the operating system as it comes up at boot time. This process is known as init. One of the major jobs of init is to fork processes that will become the getty-login-sh sequence. This sequence of processes allows users to log in and takes care of setting up the initial conditions on the outgoing terminal lines so that the speed and the other terminal-related states are correct. Init and these other processes also keep an accounting file, /etc/wtmp, that is available to processes on the system. With these files it is possible to determine the state of each process that init has spawned, and if it is a terminal line, who the current user is. One program in particular, who(1), provides a means of examining these files.

This appendix describes the capabilities of each program used in this new implementation, the databases involved, and how to create and maintain these databases. In addition, the debugging features designed in both init and getty are described in the event remedial action is required or modifications are attempted.

THE init PROCESSES

Init is driven by a database, its previous internal level, its current internal level, and events that cause it to wake up.

The Database: /etc/inittab

Init's database, kept in the file /etc/inittab, consists of any number of separate entries, each with the form:

```
    ID:level:type:process
```

where

ID is a one-to four-letter identifier that is used by init internally to label entries in its process table. It is also placed in the dynamic record file, /etc/utmp, and the history file, /etc/wtmp. The ID should be unique.

level specifies at which levels init should be concerned with this entry. Level is a string of characters consisting of [0-6a-c]. Any time that init's internal level matches a level specified by level, this entry is active. If init's internal level does not match any of the levels specified, then init makes certain that the process is not running. If the level field is empty it is equivalent to the string "0123456".

<u>type</u> specifies some further condition required for or by the execution of an entry.

off

The entry is not to run even if the levels match.

once

The entry is to be run only if init is entering a level. This means if init has been awakened by powerfail or because a child died, this entry will not be activated. Only when a user signal requests a change of init's internal state to a state that is different from its current state, and this new state is one in which this entry should be active, will this entry be activated.

wait

Wait has all the characteristics of once, plus it causes init to wait until the process spawned dies before reading any more entries from its database. This allows for initialization actions to be performed and completed before allowing other processes which might be affected to start running. It is common for shared memory segments to be initialized this way and semaphores to be continued.

respawn

Respawn requests that this entry continue to run as long init is running in a level that is in this entry's level field. Most processes spawned by init fall into this category. All getty processes are marked as respawn. Whenever init detects the death of a process that was marked respawn, it spawns a new process to take its place.

boot

Boot entries have the execution behavior of once entries. They are started only when init is switching to a numeric run state for the first time. Most commonly, boot entries have an empty level string, meaning that no matter which level init switches to the first time, the boot entry will be run. Should there be a more specific level string, for example "01", then the boot entry would only be run if init switched to either the 0 or 1 run state as its first numeric level.

bootwait

Bootwait entries have the execution behavior of wait entries and they, like boot entries, are only run as init switches to a numeric level for the first time.

power

Power entries act like once entries and are activated if init receives a SIGPWR signal (19) and is in a state that matches the active states of the entry.

powerwait

Powerwait entries act like wait entries and are activated if init receives a SIGPWR signal and is in a state which matches the active states for the entry.

initdefault	Initdefault is a non-standard entry in that it does not specify some process to be spawned. Instead, it only specifies which level init is to go to initially when it is coming up at boot time. This allows the system to be rebooted without an operator having to make entries at the system console, if so desired. If there is no initdefault entry, then init will ask at the system console, /dev/syscon, for the initial run state. In addition to specifying the numbered states, the single-user state(s) may also be specified.

process is a field indicating the action that init will ask a sh to perform whenever the entry is activated. The string in the process field process is given a prefix of "exec" so that each entry will generate only one process initially. Init then forks and execs

```
sh -c "exec process"
```

This means that the process string can take full advantage of all sh syntax. The only peculiarities arise from the string "exec," which was prefixed to the string, and because initially there is no standard input, output, or error output. The addition of "exec" to the string means that if the user wants to have a single entry generate more than one process, for example making a list of the people on the system at the time of a powerfail and mailing it to root by the command "who ¦ mail root", it would have to be put in as

```
pf::powerwait:sh -c "who ¦ mail root"
```

to work. If it was put is simply as "who ¦ mail root", it would be executed as "exec who ¦ mail root", and only the who process would be created before the sh disappeared. The lack of standard input and output channels must be addressed by explicitly specifying them.

Levels

A level is one of seven numeric levels, denoted 0, 1, 2, 3, 4, 5, or 6, three temporary levels, denoted a, b, or c, or the single-user level, s. Normally, init runs in a numeric level. Precisely how a particular level is used depends entirely on the database and the system administrator. The temporary levels allow certain entries to be started on demand without affecting any processes that were started at a particular level. The temporary levels immediately revert to the previous numeric level once all entries in the database have been scanned to see if they should be started at the temporary level. When an entry is started by a switch to a temporary level, it becomes independent of future level changes by init, except a change to the single-user level. The only way to kill a process that was started as a respawnable demand process, without going to the single-user level, is to modify the database, declaring the entry to be off.

The single-user level is the one level independent of the database. For this reason it is not a level in the normal sense. In the single-user level init spawns off a su process on the system console, and that is the only process that it maintains while at the single-user level. The single-user level can be entered at two different places in init. If it is entered at boot time it allows the operator to look over the file systems without having init attempt to do any file I/O, which might cause further problems. Init will not attempt to recreate /etc/utmp or access /etc/wtmp until after it has left this initial single-user level. If the single-user level is entered at any other time, init does do the bookkeeping in the record files.

The system administrator requests init to change levels by running a secondary copy of init itself. /etc/init is linked to /bin/telinit, and it is usually through the telinit name that this is accomplished. Init can only be run by root or a privileged group. Whenever init starts running and finds that its process ID is not 1, it assumes that it is a user-initiated copy, which is supposed to send a signal to the real init. The usage is

```
telinit [0123456sSqQabc]
```

where the single character argument specifies the signal to be sent to init. If the request is to switch to the single-user level, "S" or "s," then init also relinks /dev/syscon to the terminal originating the request so that it becomes the virtual system console, thus insuring that future messages from init will be directed to the terminal where the operator is located. When it does this relinking, it also sends a message to /dev/systty, saying that the console is being relinked to some other terminal so that there is a record of the fact at the physical system console.

Waking Events

There are four events that will wake init: boot, a powerfail, death of a child process, or a user signal.

boot — Init operates in the boot state until it has entered a numeric state for the first time. It is not possible for init to reenter the boot state a second time. Commands labeled boot and bootwait are executed when changing to a numeric state for the first time, if the levels match.

powerfail — Any time power fails, the operating system sends a SIGPWR signal to all processes. Init will execute commands with types of power and powerfail.

child death — Any time a child process of init dies, init receives a SIGCLD signal (18). The dead child process may be one of two types, a direct descendant of init, or a process whose own parent process died before it did. The parent of a process automatically becomes init, if its real parent should die before it does. Init determines immediately if the defunct process was one of its own children or an orphan. If it was one of its own, it performs the necessary bookkeeping on its internal process table to note that the process died. If init was busy at the time it received the SIGCLD signal, it then returns to complete whatever action it was performing. If init was asleep, it then scans its database to determine if any other actions should be taken, such as respawning the process.

user signal — Init catches all signals that it is possible for a process to catch. Most signals have specific meaning to init, usually requesting it to change its current state in some way. There is one signal, the "Q" signal, that is used just to waken init and cause it to scan its database. This is often issued after a change has been made to the database so that init will put the new change into effect immediately. If this was not done, the change would not become effective until init had wakened for some other reason. Other than during the initialization phase, it is solely with signals that the system administrator controls the internal level at which init is running.

Normal Operational Behavior

Init scans /etc/inittab once or twice for each event that wakes it up. If it is in the boot or powerfail state, it scans the table once, looking for entries of these types, and then switches itself back to a normal state and scans again.

Its first action in the normal state is to scan /etc/inittab and remove all processes that are currently active and should not be at the current level. Init employs one of two methods when killing its child processes, depending on whether it is changing levels or not. If init is not changing levels, it forks a child process for each child that needs to be killed, and has that child process send the signals to the process targeted for extinction.

Killing a process involves sending it two signals. First the SIGTERM signal (15) is sent so that it can clean up after itself and die gracefully. After waiting the amount of time defined as TWARN (the default value is 20 seconds), a SIGKILL signal (9) is sent, which guarantees that the child will die, if it hasn't done so already.

Forking a child to do the killing has the advantage that the main init process need not wait for all the processes it is killing to die before beginning the spawning of new processes. The disadvantage is that if many processes were being killed this way, there would be a very real chance of the operating system process table filling up, which causes the fork system call to fail. This in turn would upset init at the very least and cause it to have to wait anyway. For this reason, when init is changing levels, it assumes that it may have many processes to terminate and so it sends the signals itself, waits for the required 20 seconds, and sends the final termination signals, before continuing.

Once the old processes have been removed, init makes an entry in its accounting files if it is changing levels. At this point it either enters the single-user level or rescans its database looking for processes that need to be spawned at the current level and in the current state. In the normal state of operation, init is looking for entries whose types are off, once, wait, or respawn.

With the completion of the scan of the database in the normal state, init is ready to wait for another event. To ensure that a user who just logged off has had his or her files updated to the disk and to insure that the bookkeeping is also updated to the disk, init performs a sync system call and then pauses until it is awakened again for some new reason.

If init finds that it is being requested to switch to the single-user level when it wakens from the pause, it saves all the ioctl information about the system console in the file /etc/ioctl.syscon before proceeding to remove all its other children. It does this so that if the system is being taken down, the new init process will know how to set up the system console to talk to it. It is a convenient feature to not have to change the baud rate and terminal specifications if you are rebooting a system remotely. Because init preserves the ioctl state of the system console across system reboots, messages coming out during reboots are legible to the operator, no matter where the system console happens to be linked.

All written messages from init are sent to /dev/syscon. In reality, init itself does not send the message, but forks a child to send the message. This is because init must never open a terminal line or it will be assigned a controlling terminal. Since init has no controlling terminal, it can spawn getty processes that initially have no controlling terminal. When such a getty opens its assigned terminal, the terminal becomes the controlling terminal for it and its children. In the one instance, init needs input from the system administrator during the initialization phase. In this case, the child process that is asking for the run level opens /dev/systty, which is always the physical system console, before opening /dev/syscon, the virtual system console. This causes /dev/systty to be the child's controlling terminal. Thus, should the computer be coming up, /dev/syscon not be linked to /dev/systty, and /dev/syscon be down (perhaps because the datalink went down during the reboot), it is possible for a person at /dev/systty to regain control by typing a <DEL> character. This causes a SIGINT signal (2) to be sent to the child process, which will relink /dev/systty to /dev/syscon and ask again for a run level, this time at the physical system console.

THE getty PROGRAM

Getty is responsible for making appropriate settings of terminal characteristics and baud rate so that a user can communicate with the operating system. The most important of those features is the choice of a baud rate so that input and output make sense. In the old version of getty, there was a hardwired table in getty that controlled the search for the correct speed. The starting point in the search is specified by the arguments passed to getty. If there was some reason to change the baud rate search, getty itself had to be modified and recompiled. In the new getty, the search is controlled by an ASCII file, /etc/gettydefs, and changing or augmenting the behavior only requires that the file be edited.

Usage

Getty is normally started from /etc/inittab by init. Getty takes from one to six arguments:

```
    getty [-h] [-t time] line [speed_label][term_type][line_disc]
```

where

-h is a switch telling getty that it should not drop the Data Terminal Ready signal before resetting the line. This switch currently only works in the CB-UNIX-derived system environment. Normally, getty ensures that DTR goes down so that connections to the Develcon dataswitch will be disconnected every time. The EIA protocol requires that a dataset see DTR drop and be reasserted before answering another call. It is possible for getty to come back on a line before all the processes spun off by the previous user have died and closed their connections to the line. In this case, DTR would not drop if getty didn't ensure it. This switch is required for programs like ct, which initiate a call from the computer to a user (instead of the user calling the computer), putting a getty on the resulting connected line. Without the -h switch, the getty would immediately disconnect the user again.

-t is a switch specifying that the getty should die after the specified number of seconds if nothing is typed. This prevents datasets from being tied up if someone isn't actually logging in after they've gotten connected.

line is the name of the terminal line, which getty is to open and set up. It is minus /dev since getty does a chdir to the /dev directory and expects to find it in that directory.

speed label is usually something like "1200" or "9600," which appears to directly specify a baud rate, but in reality can be anything since it is really a label of an entry in /etc/gettydefs for which getty looks. It specifies the entry getty will start with when trying to find an appropriate speed for the terminal. It defaults to "300" if there is none given.

term type specifies which terminal discipline is to be used. If it is specified, the virtual terminal protocol becomes immediately effective on the line. Typical types might be "vt100," "hp45," or "tek". Whatever type is specified, it must be a terminal handler that has been compiled into the operating system to be effective. This argument is given for lines that are hardwired to the computer.

line disc, the line discipline, is the last argument that can be specified. The most common is "half" or "half_duplex," when there is a half duplex terminal coming into the computer. This causes the appropriate line discipline to be associated with the line.

The Database: /etc/gettydefs

Whenever getty is invoked, it references its database to determine certain information about how to set up the line. Each entry in the database has a fixed format:

```
    label# initial flags # final flags # login msg #nextlabel
```

Getty matches its speed_label argument against the "label" field. It stops searching when it finds an entry with a label that matches. The entry specifies how the terminal is supposed to be set up during the initial phase, the phase when getty prints out the "login msg" and reads in the user's login name, and the final phase, when getty exec's the login program to continue the login process. The baud rate is specified as an ioctl flag in both the initial and final flags fields.

The flags themselves are strings matching the define variables found in /usr/include/termio.h. It should be noted that these flags can be partially or totally overridden if there is a terminal type specified. When a terminal type is enabled, it resets various flags to suitable conditions automatically.

During the initial phase, getty always puts the terminal into a non-echoing raw mode. This allows it to take each character as it comes in and infer certain things about the terminal. For instance, if it sees uppercase alphabetic characters but no lowercase, it then assumes that the terminal is uppercase only and sets it up in the final configuration so that the upper-to lowercase conversions are made. Also, if the speed is wrong, it will get a <NULL> character (or <ESC> <NULL> character if a terminal type is set) if there is a framing or parity error. This means that the speed is wrong and another speed should be tried.

The typical "initial flags" would only include the speed, for example "B1200 CS7 PARENB HUPCL". "CS7 PARENB" sets the line for 7 bits, even parity characters. "HUPCL sets the line to hang up on close. Typical final flags would be "B120 SANE IXANY TAB3". "SANE" is not a real flag found in the header file, but a collection of ioctl flags used for normal terminal behavior. "IXANY" permits the use of any character to restart output. "TAB3" says to expand tabs output.

The "login msg" field is the message that getty will print before waiting for the user to enter his or her login name. It can contain anything desired and getty understands normal special character conventions so that "\n" means <lf> (line feed), as does "\012". On systems that are not using the terminal handlers and where lines are hardwired, people have been known to make up special entries for different terminal types.

Example:

```
    vt100-2400# B2400 # B2400 SANE TAB3 79530GIN: #vt100-1200
           # 33 [H 32[2JAMACCS System B
```

In this example, the "login msg" contains the special vt100 characters required to clear the screen. Notice also that the entry can take more than one line. Entries are delimited by a blank line. Lines that begin with a pound sign (#) are ignored so that comments can be added to the file.

The "next_label" field tells getty which entry to try next if it gets an indication that the speed is wrong. In the above example, it would look for an entry with the name "vt100-1200" if this one wasn't at the proper speed. Normally, the entries don't contain terminal-specific information, and the various speed choices are linked together in a closed circle of some sort. For example, it is common to have 9600 -> 4800 -> 1200 -> 300 -> 9600. In this way, no matter where you enter the circle, sooner or later you should be able to get to the speed that is correct for your terminal.

To enable the system administrator to check the database for readability by getty, there is a checking mode in which getty can be run:

```
    getty -c gettydefs_like_file
```

When getty is run in this mode, it scans the entire input file specified and deciphers each entry, printing out the resulting modes that it will set. If it finds a line that it cannot read, it prints an appropriate message that allows the administrator to correct the entry. By this mechanism it is possible to avoid installing a misformatted gettydefs file and have it tie up the system.

Also as a safety measure, should getty be unable to find /etc/gettydefs, it does have one fallback entry built in. Should gettydefs disappear for some reason, a user could still log in at 300 baud, since this is the default setting in the built-in entry.

Operational Behavior

As has been shown earlier, getty sets up a line as specified by an entry from /etc/gettydefs and from any additional arguments, outputs the "login msg" field, and then tries to read the user's login name from the line. During the input of the login name, getty checks for speed mismatches that the operating system will report as a <NULL> character. If such a mismatch occurs, getty tries the next speed specified by the current entry, and repeats the whole sequence. Also while reading in the login name, getty makes a guess whether the terminal is uppercase only. If it sees some uppercase characters, but no lowercase characters, it assumes that the terminal is uppercase only and sets the ioctl state of the line to translate uppercase letters to lowercase on input, and lower- to uppercase on output.

An addition has been made to getty and login that allows for environmental variables to be set up at the time a user enters his or her login name. This allows users to control the behavior of their .profile at the time they specify their login names. Getty executes the login program by passing all the separate words given to it in response to the login message as arguments to login. If, for example, the user responded with "jls f", then getty would execute "login jls f" as its final action. See the login subsection to see how this modifies the command's behavior.

THE login PROGRAM

Unlike init and getty, login did not require a great deal of modification. The only required change was that it should write to /etc/utmp and /etc/wtmp in the new format. This change was minor. At the time this change was made, a change visible to the user was also made: the ability to add to the environment. This change was added as a convenience. It allows the user to modify the behavior of his or her .profile by having environmental variables set which the .profile script knows about.

The basic change was that any additional words provided in response to the basic "login:" query are placed in the environment of the sh executed by login as its last act in the following way. If the word does not contain an "-", then a shell variable of the type "Ln=word" is created. Here, n is a number starting at 0 and for each new environment variable it is incremented by one. If the word does contain an "-", then the whole string is passed in the environment unchanged. For example, "TERM-2621" would be placed in the environment unchanged and the shell variable $TERM would be defined as "2621."

To preserve security, there are a couple of exceptions. It is not possible to change the shell variables $PATH or $SHELL by this mechanism. That means that a restricted shell will remain restricted and that the user cannot gain access to commands that might allow him to avoid the usual restrictions of rsh.

THE who PROGRAM

Who(1) is the program that reads the history files maintained by init, getty, and login. Since the format of these files was changed substantially, it was necessary to change who. In the process, some additional features were added to who so that it would convey more useful information to users. The standard usage for who is:

```
    who [-uTlpdbrtas] [[am i] or [utmp_like_file]]
```

where

u means return a listing of useful information for all the users. This information includes login time, activity, pid, and comment from inittab file.

T means report the writability state of the terminal for that entry.

l means report all entries that are living getty processes.

p means report all entries for living children of init, excluding getty and descendants of getty.

d means report all the entries for processes that have died.

b means report the boot time entries that init has made. In /etc/utmp there is only one such entry.

r means report the run level entries that init has made. In /etc/utmp there is only one such entry, the current run level entry. The current state, the number of times in that state, and the previous state are also reported.

t means report the change-of-date entries that have been made by the date(1) command when the clock was reset. These are required in the history file, /etc/wtmp, if accounting is to be done.

a means report all the entries.

s means report information for all users in short form; this is the default. If no file is specified, then /etc/utmp is assumed. The who am i sequence returns the entry for the user typing the command.

There are various output formats for the different kinds of entry. In particular, entries for users and getty processes list the amount of time since output to the terminal occurred. This is often of interest since it shows other users whether someone is actually working at a terminal or not. The comment field at the end of the entry from /etc/inittab is also included, which can conveniently be set up to be the location of the terminal. Dead entries report the exit status for the process that died. This can be of use, since it shows whether the process terminated abnormally or not.

OTHER AFFECTED PROGRAMS

All programs accessing the accounting files were affected by the new utmp structure. In particular, date (1) makes two entries indicating the old time and new time, whenever it changes the system clock. Also affected are the commands in /usr/lib/acct, which produces reports based on the information in /etc/wtmp.

utmp FORMAT

A major change in going to the new init was that it uses a different format in writing out its records in /etc/utmp and /etc/wtmp. The new format is shown below in Figure B-1.

```
/*  <sys/types.h> must be included                                 */

#define UTMP_FILE       "/etc/utmp"
#define WTMP_FILE       "/etc/wtmp"
#define ut_name         ut_user

struct utmp
  {

         char ut_user[8] ;         /* User login name */
         char ut_id[4] ;           /* /etc/lines id (usually line #) */
         char ut_line[12] ;            /* device name (console, lnxx) */
         short ut_pid ;                /* process id */
         short ut_type ;           /* type of entry */
         struct exit_status
           {
            short e_termination ; /* Process termination status */
            short e_exit ;        /* Process exit status */
           }
         ut_exit ;                 /* The exit status of a process
                                            * marked as DEAD_PROCESS.
                                            */
         time_t ut_time ;            /* time entry was made */
  };
```

Figure B-1. utmp Format (page 1 of 2)

```
/*       Definitions for ut_type                                  */

#define EMPTY          0
#define RUN_LVL        1
#define BOOT_TIME      2
#define OLD_TIME       3
#define NEW_TIME       4
#define INIT_PROCESS 5      /* Process spawned by "init" */
#define LOGIN_PROCESS       6    /* A "getty" process waiting for login */
#define USER_PROCESS 7      /* A user process */
#define DEAD_PROCESS        8
#define ACCOUNTING   9

#define UTMAXTYPE  ACCOUNTING /* Largest legal value of ut_type */

/*    Special strings or formats used in the "ut_line" field when    */
/*    accounting for something other than a process.                 */
/*    No string for the ut_line field can be more than 11 chars +    */
/*    a NULL in length.                                              */

#define RUNLVL_MSG "run-level %c"
#define BOOT_MSG   "system boot"
#define OTIME_MSG  "old time"
#define NTIME_MSG  "newtime"
```

Figure B-1. utmp Format (page 2 of 2)

The ut_type field completely identifies the type of entry, whereas the ut_id field only contains the "id" as found in the "id" field of /etc/inittab. The ut_line field was expanded and freed so that it can contain things like console or other things that are not of the form /dev/lnxx. Finally, ut_exit contains the exit status of processes that init has spawned and that have subsequently died.

Appendix C
System Accounting

The accounting system for this operating system provides methods to collect per-process resource utilization data, record connect sessions, monitor disk utilization, and charge fees to specific logins. A set of C language programs and shell procedures is provided to reduce this accounting data into summary files and reports. This section describes the structure, implementation, and management of this accounting system, as well as a discussion of the reports generated and the meaning of the columnar data.

GENERAL

The following list is a synopsis of the actions of the accounting system:

- At process termination, the operating system kernel writes one record per process in /usr/adm/pacct in the form of acc.h. (See the subsection "System Accounting Data Files" for a description of data files.)
- The login and init programs record connect sessions by writing records into /etc/wtmp. Date changes, reboots, and shutdowns are also recorded in this file.
- The disk utilization program acctdusg breaks down disk usage by login.
- Fees for file restores and other services can be charged to specific logins with the chargefee shell procedure.
- Each day the runacct shell procedure is executed through cron to reduce accounting data and produce summary files and reports. (See the subsection "Daily Reports" for a sample report output).
- The monacct procedure can be executed on a monthly or fiscal period basis. It saves and restarts summary files, generates a report, and cleans up the sum directory. These saved summary files could be used to charge users for operating system usage.

FILES AND DIRECTORIES

The /usr/lib/acct directory contains all of the C language programs and shell procedures necessary to run the accounting system. The adm login (currently user ID of four) is used by the accounting system and has the directory structure shown in Figure C-1.

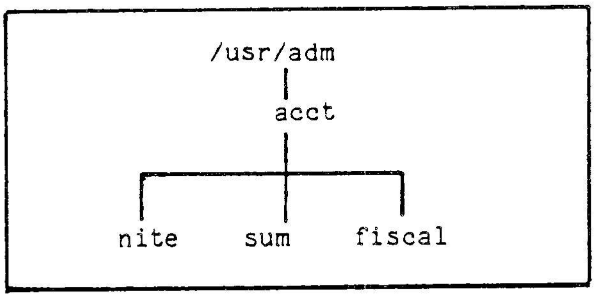


Figure C-1. Directory Structure of the adm Login

The /usr/adm directory contains the active data collection files. (For an explanation of the files used by the accounting system, see the subsection on "Accounting System Files.") The nite directory contains files that are reused daily by the runacct procedure. The sum directory contains the cumulative summary files updated by runacct. The fiscal directory contains periodic summary files created by monacct.

DAILY OPERATION

When the operating system is switched into multiuser mode, /usr/lib/acct/startup is executed, which does the following:

1. The acctwtmp program adds a "boot" record to /usr/adm/wtmp. This record is signified by using the system name as the login name in the wtmp record.

2. Process accounting is started through turnacct. Turnacct on executes the accton program with the argument /usr/adm/pacct.

3. The remove shell procedure is executed to clean up the saved pacct and wtmp files left in the sum directory by runacct.

The ckpacct procedure is run through cron every hour of the day to check the size of /usr/adm/pacct. If the file grows past 1000 blocks (default), turnacct switch is executed. Although ckpacct is not absolutely necessary, the advantage of having several smaller pacct files becomes apparent when trying to restart runacct after a failure in processing these records.

The chargefee program can be used to bill users for file restores and so on. It adds records to /usr/adm/fee that are picked up and processed by the next execution of runacct and merged into the total accounting records.

runacct is executed through cron each night. It processes the active accounting files /usr/adm/pacct, /usr/adm/wtmp, /usr/adm/acct/nite/disktacct, and /usr/adm/fee. It produces command summaries and usage summaries by login.

When the system is shut down using shutdown, the shutacct shell procedure is executed. It writes a shutdown reason record into /usr/adm/wtmp and turns process accounting off.

After the first reboot each morning, the computer operator should execute /sur/lib/acct/prdaily to print the previous day's accounting report.

SETTING UP THE ACCOUNTING SYSTEM

In order to automate the operation of this accounting system, perform the following steps:

a. If not already present, add this line to the /etc/rc file in the state 2 section:

```
/bin/su-adm-c/usr/lib/acct/startup
```

b. If not already present, add this line to /etc/shutdown to turn off the accounting before the system is brought down:

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

c. For most installations, the following three entries should be made in /usr/lib/crontab so that cron will automatically run the daily accounting.

```
"0 4 * * 1-6 /bin/su-adm -c " /usr/lib/acct/runacct
                2>   /usr/adm/acct/nite/fd2log"
 0 2 * * 4 /usr/lib/acct/dodisk
 5 * * * * /bin/sy-adm-c" /usr/lib/acct/ckpacct"
```

Note that dodisk is invoked with superuser privileges of root so that directory searching is not road-blocked.

d. To facilitate monthly merging of accounting data, the following entry in crontab will allow monacct to clean up all daily reports and daily total accounting files and deposit one monthly total report and one monthly total accounting file in the fiscal directory.

```
15 5 1 * * /bin/su-adm -c/usr/lib/acct/monacct
```

The above entry takes advantage of the default action of monacct that the current month's date as the suffix for the file names. Notice that the entry is executed at such a time as to allow runacct sufficient time to complete. This will, on the first day of each month, create monthly accounting files with the entire month's data.

e. The PATH shell variable should be set in /usr/adm/profile to:

```
PATH=/usr/lib/acct:/bin:/usr/bin
```

THE runacct PROCEDURE

runacct is the main daily accounting shell procedure. It is normally initiated through cron during nonprime time hours. Runacct processes connect, fee, disk and process accounting files. It also prepares daily and cumulative summary files for use by prdaily or for billing purposes. The following files produced by runacct are of particular interest.

nite/lineuse — Produced by acctcon, which reads the wtmp file and produces usage statistics for each terminal line on the system. 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 possiblity that the line is failing.

nite/dayacct — 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, which can be used for billing purposes. It is restarted each month or fiscal period by the monacct procedure.

sum/daycms — Produced by the acctcms program, it contains the daily command summary. The ASCII version of this file is nite/daycms.

sum/cms — The accumulation of each day's command summaries. It is restarted by the execution of monacct. The ASCII version isnite/cms.

sum/loginlog — Produced by the lastlogin shell procedure, it maintains a record of the last time each login was used.

sum/rprt.MMDD — Each execution of runacct saves a copy of the output of prdaily.

runacct 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. To prevent multiple invocations in the event of two crons or other problems, runacct will complain if the files lock and lock1 exist when invoked. 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, the locks are removed, diagnostic files are saved and execution is terminated.

In order to allow runacct to be restartable, processing is broken down into separate reentrant states. This is accomplished by using a case statement inside an endless while loop. Each state is one case of the case statement. A file is used to remember the last state completed. When each state completes, statefile is updated to reflect the next state. In the next loop through the while, statefile is read and the case falls through to the next state. 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 /usr/adm/wtmp file is moved to /usr/adm/acct/nite/wtmp.MMDD with the current time added on the end.

WTMPFIX — The wtmp file in the nite directory is checked for correctness the wtmpfix program. 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 connect accounting records. (Accounting records are in tacc.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 ptact 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 sdisk 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 recreated 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.

Recovering From Failure

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

ERROR: locks found, run aborted

> The files lock and lock1 were found. These files must be removed before runacct can restart.

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.

Restarting runacct

Runacct called without arguments 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.

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&
```

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 file save 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 operating system is in multiuser mode, a set of date change records is written into /usr/adm/wtmp. The wtmpfix program is designed to adjust the time stamps in the wtmp records when a date change is encountered. Some combinations of date changes and reboots, however, 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
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 ID's, 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 recreated. A simple patchup procedure would be:

```
cd /usr/adm/acct/sum
acctmerg -v < tacct.MMDD> xtacct
ed xtacct
  remove the bad records
  write duplicate uid records to another file
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 recreated by merging these files together.

UPDATING pnpsplit

The pnpsplit subroutine is used by acctcon1 and acctprc1 to determine the difference between prime and nonprime time. Prime time is defaulted from 9:00 a.m. to 5:00 p.m., Monday through Friday. Nonprime time is considered to be all other hours and the entire day for those days listed in the holidays structure in pnpsplit.c. The holidays listed are accurate for the year the operating system was released. Every year on the day after Christmas (the last holiday of the calendar year), the following message will be printed on the system console terminal and appear in log:

```
*** RECOMPILE pnpsplit WITH NEW HOLIDAYS ***
```

This message will continue to be sent each time the accounting is run until pnpsplit, acctcon1, and acctprc1 are recompiled. The following steps should be taken to recompile these programs successfully.

a. Edit pnpsplit.c to change the thisyear variable to the new year. Update the holidays structure to reflect the new holidays. The numeric entry in the structure is the day of the year, less one. For example, New Year's Day (January 1) is entered as 0. Pnpsplit.c is in /usr/src/cmd/acct/lib.

b. Update the accounting library a.a and recompile acctprc1 and acctcon1 by

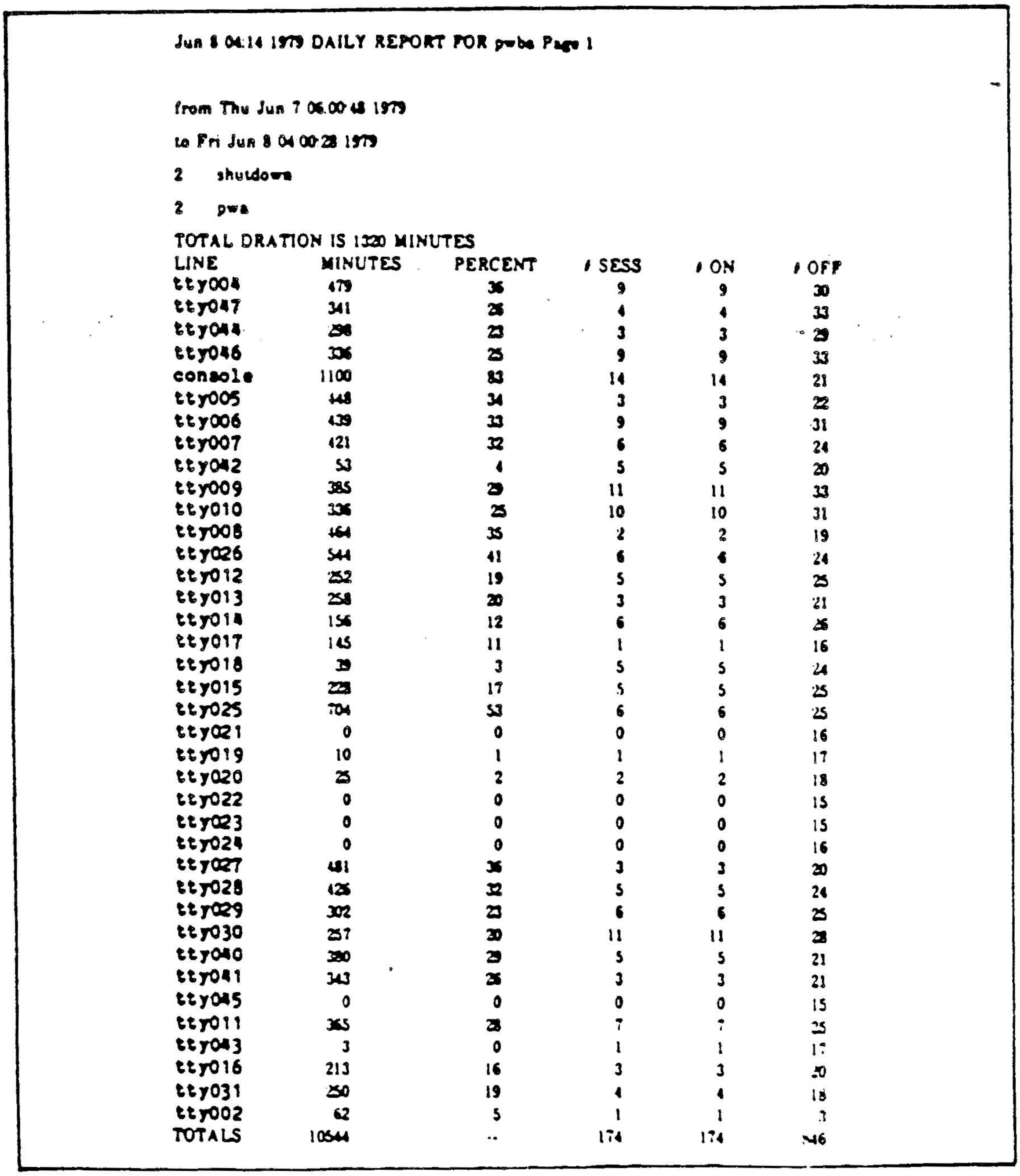
```
superuser to root

ARGS="acctcon1 acctprc1"  /usr/src/:mkcmd acct
```

DAILY REPORTS

Runacct generates five basic reports upon each invocation. These reports 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. Samples of these reports are shown in Figures C-2 through C-6.

Daily Report

As shown in Figure C-2, the first part of a daily report contains a from/to banner indicating the report period. 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 /usr/adm/wtmp by the acctwtmp program (see acct(1M) in the Series 6000 Operating System Reference Manual, Volume 1).



```
Jun 8 04:14 1979 DAILY REPORT FOR pwba Page 1


from Thu Jun 7 06:00:48 1979

to Fri Jun 8 04:00:28 1979

2     shutdown

2     pwa

TOTAL DRATION IS 1320 MINUTES
LINE          MINUTES     PERCENT     # SESS     # ON     # OFF
tty004          479          36          9         9        30
tty047          341          26          4         4        33
tty044          298          23          3         3        29
tty046          336          25          9         9        33
console        1100          83         14        14        21
tty005          448          34          3         3        22
tty006          439          33          9         9        31
tty007          421          32          6         6        24
tty042           53           4          5         5        20
tty009          385          29         11        11        33
tty010          336          25         10        10        31
tty008          464          35          2         2        19
tty026          544          41          6         4        24
tty012          252          19          5         5        25
tty013          258          20          3         3        21
tty014          156          12          6         6        26
tty017          145          11          1         1        16
tty018           39           3          5         5        24
tty015          228          17          5         5        25
tty025          704          53          6         6        25
tty021            0           0          0         0        16
tty019           10           1          1         1        17
tty020           25           2          2         2        18
tty022            0           0          0         0        15
tty023            0           0          0         0        15
tty024            0           0          0         0        16
tty027          481          36          3         3        20
tty028          426          32          5         5        24
tty029          302          23          6         6        25
tty030          257          20         11        11        28
tty040          380          29          5         5        21
tty041          343          26          3         3        21
tty045            0           0          0         0        15
tty011          365          28          7         7        25
tty043            3           0          1         1        17
tty016          213          16          3         3        20
tty031          250          19          4         4        18
tty002           62           5          1         1        13
TOTALS        10544          --        174       174       946
```

Figure C-2. Sample Daily Report

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:

LINE The terminal line or access port.

MINUTES The total number of minutes that line was in use during the accounting period.

PERCENT The total number of MINUTES the line was in use divided into the TOTAL DURATION.

SESS The number of times this port was accessed for a login(1) session.

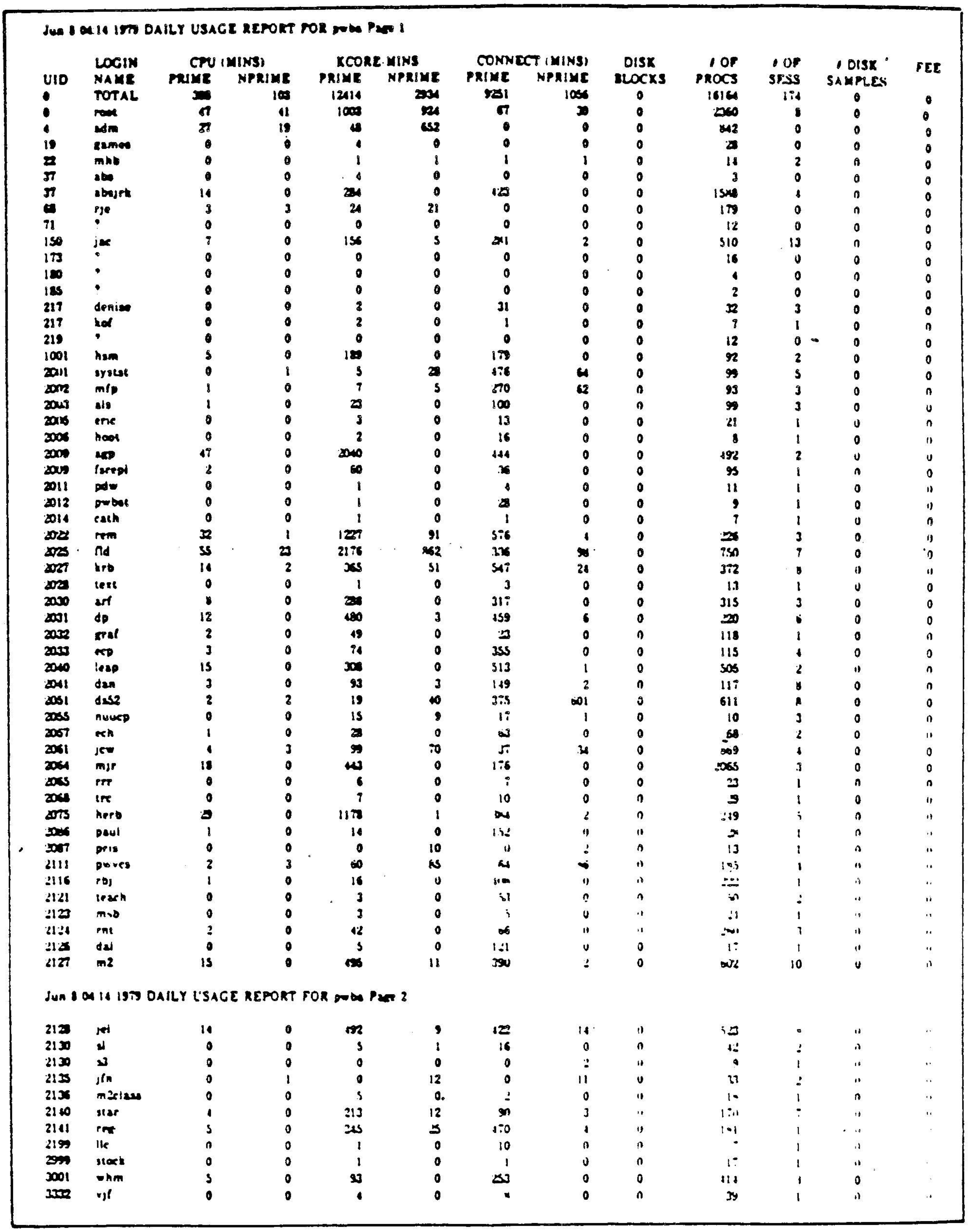
ON This column does not have much meaning anymore. It used to give the number of times that the port was used to log on a user; but since login(1) can no longer be executed explicitly to log on a new user, this column should be identical with SESS.

OFF This column reflects not only the number of times a user logged off but also any interrupts that occur on that line. Generally, interrupts occur on a port when the getty is first invoked when the system is brought to multiuser state. These interrupts occur at a rate of about two per event; therefore, it is not uncommon to see in excess of twice the amount of OFF than ON or SESS. 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, /usr/adm/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 great rate, general system performance will be effected.

Daily Usage Report

This report gives a by-user breakdown of system resource utilization. Figure C-3 is a sample of this report.



```
Jun 8 04 14 1979 DAILY USAGE REPORT FOR pwba Page 1

        LOGIN      CPU (MINS)        KCORE-MINS       CONNECT (MINS)    DISK     # OF    # OF   # DISK    FEE
UID     NAME     PRIME   NPRIME    PRIME   NPRIME    PRIME   NPRIME    BLOCKS   PROCS   SESS   SAMPLES
0       TOTAL      388      108    12414     2934     9251     1056       0     16164    174      0        0
0       root        47       41     1008      924       67       30       0      2360      8      0        0
4       adm         27       19       48      652        0        0       0       842      0      0        0
19      games        0        0        4        0        0        0       0        28      0      0        0
22      mhb          0        0        1        1        1        1       0        14      2      0        0
37      abe          0        0        4        0        0        0       0         3      0      0        0
37      abejrk      14        0      284        0      423        0       0      1548      4      0        0
68      rje          3        3       24       21        0        0       0       179      0      0        0
71      *            0        0        0        0        0        0       0        12      0      0        0
150     jac          7        0      156        5      241        2       0       510     13      0        0
173     *            0        0        0        0        0        0       0        16      0      0        0
180     *            0        0        0        0        0        0       0         4      0      0        0
185     *            0        0        0        0        0        0       0         2      0      0        0
217     denise       0        0        2        0       31        0       0        32      3      0        0
217     kof          0        0        2        0        1        0       0         7      1      0        0
219     *            0        0        0        0        0        0       0        12      0      0        0
1001    ham          5        0      189        0      179        0       0        92      2      0        0
2001    systat       0        1        5       28      476       64       0        99      5      0        0
2002    mfp          1        0        7        5      270       62       0        93      3      0        0
2003    als          1        0       23        0      100        0       0        99      3      0        0
2005    eric         0        0        3        0       13        0       0        21      1      0        0
2006    hoot         0        0        2        0       16        0       0         8      1      0        0
2008    agp         47        0     2040        0      444        0       0       492      2      0        0
2009    farepl       2        0       60        0       36        0       0        95      1      0        0
2011    pdw          0        0        1        0        4        0       0        11      1      0        0
2012    pwbst        0        0        1        0       28        0       0         9      1      0        0
2014    cath         0        0        1        0        1        0       0         7      1      0        0
2022    rem         32        1     1227       91      576        4       0       226      3      0        0
2025    fld         55       23     2176      862      336       98       0       750      7      0        0
2027    krb         14        2      365       51      547       24       0       372      8      0        0
2028    text         0        0        1        0        3        0       0        13      1      0        0
2030    arf          8        0      288        0      317        0       0       315      3      0        0
2031    dp          12        0      480        3      459        6       0       220      6      0        0
2032    graf         2        0       49        0       23        0       0       118      1      0        0
2033    ecp          3        0       74        0      355        0       0       115      4      0        0
2040    leap        15        0      308        0      513        1       0       506      2      0        0
2041    dan          3        0       93        3      149        2       0       117      8      0        0
2051    da52         2        2       19       40      375      601       0       611      8      0        0
2055    nuucp        0        0       15        9       17        1       0        10      3      0        0
2067    ech          1        0       28        0       63        0       0        68      2      0        0
2061    jcw          4        3       99       70       37       34       0       569      4      0        0
2064    mjr         18        0      443        0      176        0       0      1065      3      0        0
2065    rrr          0        0        6        0        7        0       0        23      1      0        0
2068    trc          0        0        7        0       10        0       0        39      1      0        0
2075    herb        29        0     1178        1      844        2       0       219      5      0        0
2086    paul         1        0       14        0      152        0       0        54      1      0        0
2087    pris         0        0        0       10        0        2       0        13      1      0        0
2111    pwves        2        3       60       85       64  [illegible]  0       195      1      0        0
2116    rbj          1        0       16        0  [illegible]     0       0  [illegible]  1      0        0
2121    teach        0        0        3        0       51        0       0        50      2      0        0
2123    msb          0        0        3        0        5        0       0        21      1      0        0
2124    rnt          2        0       42        0       66        0       0  [illegible]  3      0        0
2126    dal          0        0        5        0      121        0       0        17      1      0        0
2127    m2          15        0      496       11      390        2       0       602     10      0        0

Jun 8 04 14 1979 DAILY USAGE REPORT FOR pwba Page 2

2128    jel         14        0      492        9      422       14       0       523  [illegible]  0     0
2130    sl           0        0        5        1       16        0       0        42      2      0  [illegible]
2130    s3           0        0        0        0        0        2       0         9      1      0        0
2135    jfn          0        1        0       12        0       11       0        33      2      0        0
2136    m2class      0        0        5       0.        2        0       0        14      1      0        0
2140    star         4        0      213       12       90        3       0       170      7      0        0
2141    rmg          5        0      245       25      470        4       0       141      1      0  [illegible]
2199    lle          0        0        1        0       10        0       0  [illegible]  1      0  [illegible]
2999    stock        0        0        1        0        1        0       0        17      1      0  [illegible]
3001    whm          5        0       93        0      253        0       0       114      1      0  [illegible]
3332    vjf          0        0        4        0        4        0       0        39      1      0        0
```

Figure C-3. Sample Daily Usage Report

The data in Figure C-3 consist of:

UID
The user ID.

LOGIN NAME
The login name of the user. There can be more than one login or a single user ID; this identifies which one.

CPU (MINS)
This represents the amount of time the user's process used the processor. This category is broken down into PRIME and NPRIME (nonprime) utilization. The accounting system's idea of this breakdown is located in the accounting library function pnpsplit where the holidays array, which also determines nonprime time, is also defined. As delivered, prime time is defined to be 0900-1700 hours. The holidays array is correct for the year of the release.

KCORE-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 later column called # OF PROCS is low, this user is what is called a "line hog". That is, this person logs in first thing in the morning but does not use the terminal much the rest of the day. Watch out for this kind of user. This column is also subdivided into PRIME and NPRIME utilization.

CONNECT (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.

DISK BLOCKS
When the disk accounting programs have been run, their 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 runs amiss. The most common example of this is for a crontab entry to try to execute a user's profile through su- that unfortunately prompts for a terminal type and sits in an endless loop trying to read from the terminal (there is not one when cron is executing a process). Preventive coding is encouraged in the .profile.

# OF SESS This is how many times the user logged onto the system.

# DISK SAMPLES This indicates how many times the disk accounting was run to obtain the average number of DISK BLOCKS listed earlier.

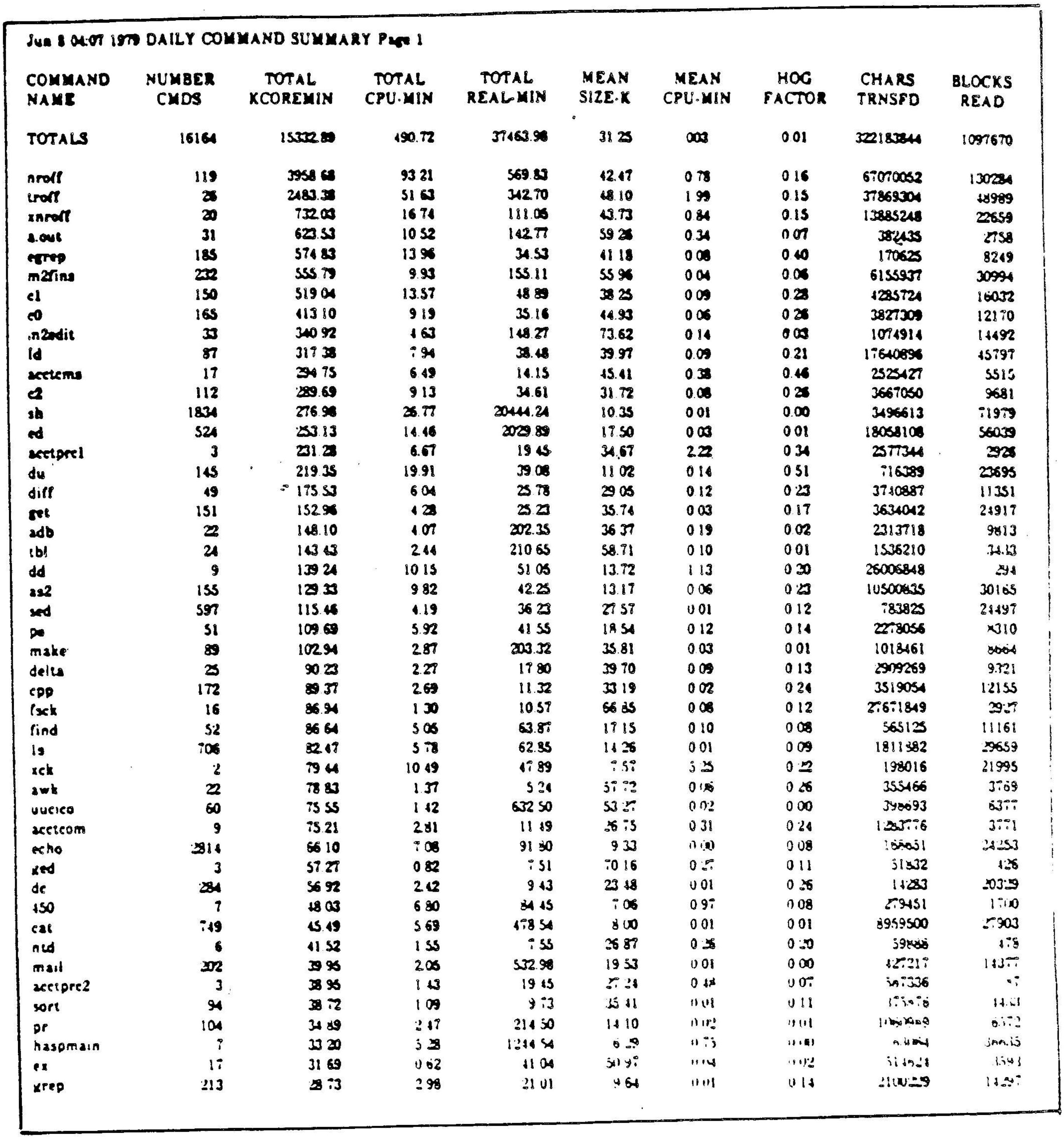
FEE An often unused field in the total accounting record, the FEE represents the total accumulation of widgets charged against the user by the chargefee shell procedure (see acctsh(1M)). The chargefee procedure is used to levy charges against a user for special services performed, such as file restores, tape manipulation by operators, and so on.

Daily Command and Monthly Total Command Summaries

These two reports are virtually the same, except that the daily command summary reports only on the current accounting period, whereas the monthly total comand 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.

The data included in these reports give an idea as to the most heavily-used commands. They also hint at what to weigh more heavily when tuning the system, based on those commands' characteristics of system resource utilization. Figures C-4 and C-5 are samples of these summaries.

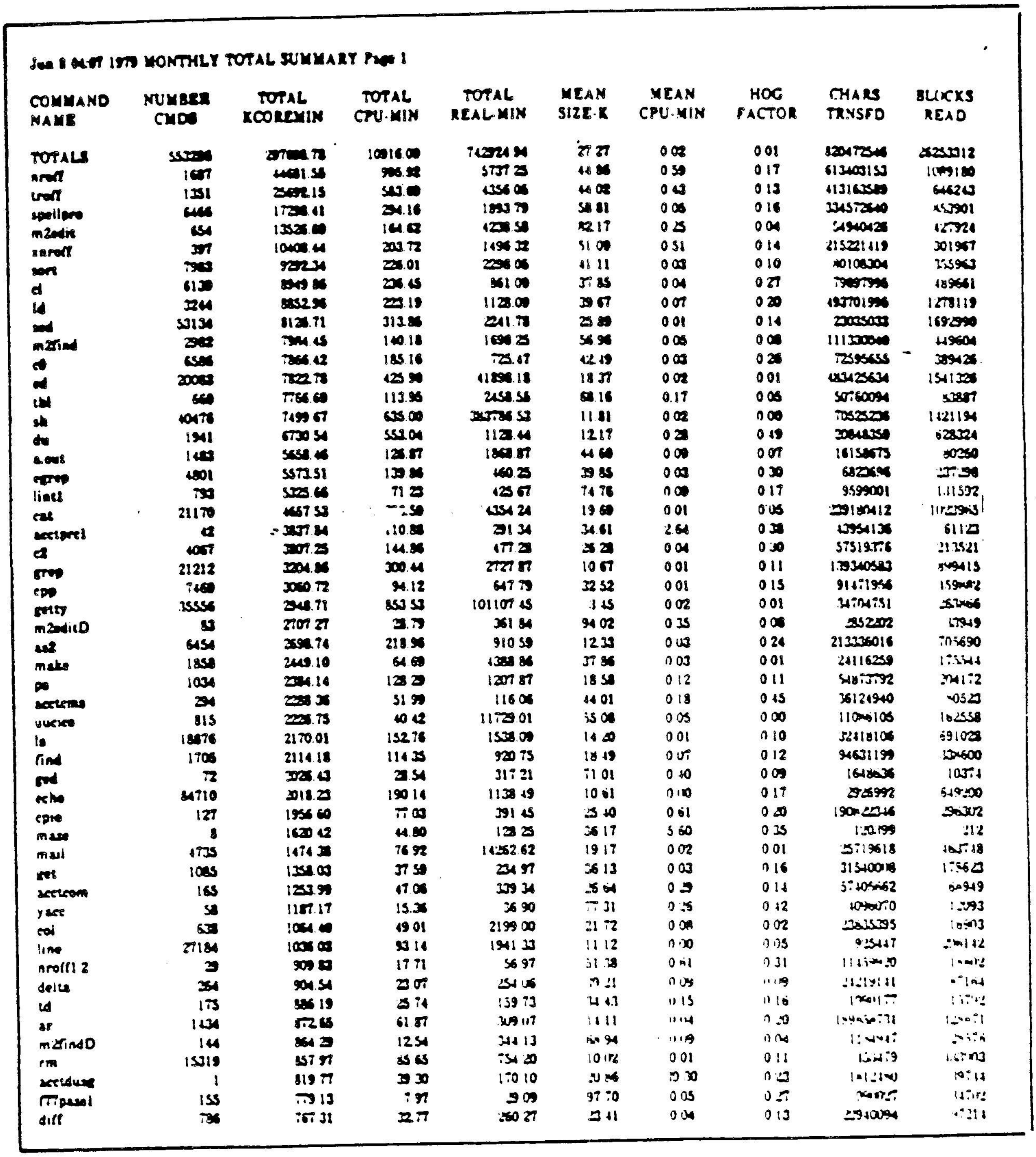
These reports are sorted by TOTAL KCOREMIN, which is an arbitrary yardstick, but often a good one for calculating "drain" on a system.



Jun 8 04:07 1979 DAILY COMMAND SUMMARY Page 1

COMMAND NAME	NUMBER CMDS	TOTAL KCOREMIN	TOTAL CPU-MIN	TOTAL REAL-MIN	MEAN SIZE-K	MEAN CPU-MIN	HOG FACTOR	CHARS TRNSFD	BLOCKS READ
TOTALS	16164	15332.89	490.72	37463.98	31.25	003	0.01	322183844	1097670
nroff	119	3958.68	93.21	569.83	42.47	0.78	0.16	67070052	130284
troff	26	2483.38	51.63	342.70	48.10	1.99	0.15	37869304	48989
xnroff	20	732.03	16.74	111.06	43.73	0.84	0.15	13885248	22659
a.out	31	623.53	10.52	142.77	59.26	0.34	0.07	382435	2758
egrep	185	574.83	13.96	34.53	41.18	0.08	0.40	170625	8249
m2fins	232	555.79	9.93	155.11	55.96	0.04	0.06	6155937	30994
c1	150	519.04	13.57	48.89	38.25	0.09	0.28	4285724	16032
c0	165	413.10	9.19	35.16	44.93	0.06	0.26	3827309	12170
m2edit	33	340.92	4.63	148.27	73.62	0.14	0.03	1074914	14492
ld	87	317.38	7.94	38.48	39.97	0.09	0.21	17640896	45797
acctcms	17	294.75	6.49	14.15	45.41	0.38	0.46	2525427	5515
c2	112	289.69	9.13	34.61	31.72	0.08	0.26	3667050	9681
sh	1834	276.98	26.77	20444.24	10.35	0.01	0.00	3496613	71979
ed	524	253.13	14.46	2029.89	17.50	0.03	0.01	18058108	56039
acctprc1	3	231.28	6.67	19.45	34.67	2.22	0.34	2577344	2926
du	145	219.35	19.91	39.08	11.02	0.14	0.51	716389	23695
diff	49	175.53	6.04	25.78	29.05	0.12	0.23	3740887	11351
get	151	152.96	4.28	25.23	35.74	0.03	0.17	3634042	24917
adb	22	148.10	4.07	202.35	36.37	0.19	0.02	2313718	9813
tbl	24	143.43	2.44	210.65	58.71	0.10	0.01	1536210	3413
dd	9	139.24	10.15	51.05	13.72	1.13	0.20	26006848	294
as2	155	129.33	9.82	42.25	13.17	0.06	0.23	10500835	30165
sed	597	115.46	4.19	36.23	27.57	0.01	0.12	783825	24497
ps	51	109.69	5.92	41.55	18.54	0.12	0.14	2278056	8310
make	89	102.94	2.87	203.32	35.81	0.03	0.01	1018461	8664
delta	25	90.23	2.27	17.80	39.70	0.09	0.13	2909269	9321
cpp	172	89.37	2.69	11.32	33.19	0.02	0.24	3519054	12155
fsck	16	86.94	1.30	10.57	66.85	0.08	0.12	27671849	2927
find	52	86.64	5.06	63.87	17.15	0.10	0.08	565125	11161
ls	706	82.47	5.78	62.85	14.26	0.01	0.09	1811382	29659
xck	2	79.44	10.49	47.89	7.57	5.25	0.22	198016	21995
awk	22	78.83	1.37	5.24	57.72	0.06	0.26	355466	3769
uucico	60	75.55	1.42	632.50	53.27	0.02	0.00	398693	6377
acctcom	9	75.21	2.81	11.49	26.75	0.31	0.24	1263776	3771
echo	2814	66.10	7.08	91.80	9.33	0.00	0.08	166651	24253
ged	3	57.27	0.82	7.51	70.16	0.27	0.11	51832	426
dc	284	56.92	2.42	9.43	23.48	0.01	0.26	14283	20329
450	7	48.03	6.80	84.45	7.06	0.97	0.08	279451	1700
cat	749	45.49	5.69	478.54	8.00	0.01	0.01	8959500	27903
ntd	6	41.52	1.55	7.55	26.87	0.26	0.20	59888	475
mail	202	39.96	2.06	532.98	19.53	0.01	0.00	427217	14377
acctprc2	3	38.96	1.43	19.45	27.24	0.48	0.07	567336	[illegible]
sort	94	38.72	1.09	9.73	35.41	0.01	0.11	175878	[illegible]
pr	104	34.89	2.47	214.50	14.10	0.02	0.01	[illegible]	6572
haspmain	7	33.20	5.28	1244.54	6.29	0.75	0.00	[illegible]	36615
ex	17	31.69	0.62	41.04	50.97	0.04	0.02	514624	3593
grep	213	28.73	2.98	21.01	9.64	0.01	0.14	2100229	14297

Figure C-4. Sample Daily Command Summary



```
Jun 8 04:07 1979 MONTHLY TOTAL SUMMARY Page 1

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     553298  297088.78   10916.00   742924.94    27.27     0.02     0.01   820472546    26253312
nroff        1687   44681.58     986.92     5737.25    44.86     0.59     0.17   613403153     1099180
troff        1351   25692.15     583.00     4356.06    44.02     0.43     0.13   413163589      646243
spellpro     6466   17298.41     294.16     1893.79    58.81     0.06     0.16   334572640      453901
m2edit        654   13526.00     164.62     4238.58    82.17     0.25     0.04    54940428      427924
xnroff        397   10408.44     203.72     1496.32    51.09     0.51     0.14   215221419      301967
sort         7983    9292.34     226.01     2298.06    41.11     0.03     0.10    80108304      355963
cl           6139    8949.86     236.45      961.00    37.85     0.04     0.27    79897996      489661
ld           3244    8852.96     223.19     1128.09    39.67     0.07     0.20   493701996     1278119
sed         53134    8126.71     313.86     2241.78    25.89     0.01     0.14    23035033     1692990
m2find       2982    7984.45     140.18     1698.25    56.96     0.06     0.08   111330040      449604
c0           6586    7866.42     185.16      725.47    42.49     0.03     0.26    72595655      389426
ed          20083    7822.78     425.90    41898.18    18.37     0.02     0.01   483425634     1541326
tbl           660    7766.60     113.96     2458.56    68.16     0.17     0.06    50760094       53887
sh          40476    7499.67     635.00   383786.53    11.81     0.02     0.00    70525236     1421194
du           1941    6730.54     553.04     1128.44    12.17     0.28     0.49    20848359      628324
a.out        1483    5658.46     126.87     1868.87    44.60     0.09     0.07    16158675       80250
egrep        4801    5573.51     139.86      460.25    39.85     0.03     0.30     6823696      237398
lintl         793    5325.66      71.23      425.67    74.76     0.09     0.17     9599001      111592
cat         21170    4667.53      72.50     4354.24    19.60     0.01     0.05   239180412     1122963
acctprcl       42    3837.84     110.88      291.34    34.61     2.64     0.38    43954136       61123
c2           4067    3807.25     144.86      477.28    26.28     0.04     0.30    57519376      213521
grep        21212    3204.86     300.44     2727.87    10.67     0.01     0.11   139340583      549415
cpp          7460    3060.72      94.12      647.79    32.52     0.01     0.15    91471956      159482
getty       35556    2948.71     853.53   101107.45     3.45     0.02     0.01    34704751      253466
m2editD        83    2707.27      28.79      361.84    94.02     0.35     0.08     5852202       17949
as2          6454    2698.74     218.96      910.59    12.33     0.03     0.24   213336016      705690
make         1858    2449.10      64.60     4388.86    37.86     0.03     0.01    24116259      173344
ps           1034    2384.14     128.29     1207.87    18.58     0.12     0.11    54873792      204172
acctcms       294    2288.36      51.99      116.06    44.01     0.18     0.45    36124940       40523
uucico        815    2226.75      40.42    11729.01    55.08     0.05     0.00   110446105      162558
ls          18876    2170.01     152.76     1538.09    14.20     0.01     0.10    32418106      691028
find         1706    2114.18     114.35      920.75    18.49     0.07     0.12    94631199      134600
ged            72    2026.43      28.54      317.21    71.01     0.40     0.09     1648636       10374
echo        84710    2018.23     190.14     1138.49    10.61     0.10     0.17     2926992      649200
cpio          127    1956.60      77.03      391.45    25.40     0.61     0.20   190422346      296302
maze            8    1620.42      44.80      128.25    36.17     5.60     0.35     1201199         212
mail         4735    1474.38      76.92    14262.62    19.17     0.02     0.01    25719618      463748
get          1085    1358.03      37.59      234.97    36.13     0.03     0.16    31540018      175623
acctcom       165    1253.99      47.08      339.34    26.64     0.29     0.14    57405662       64949
yacc           58    1187.17      15.36       36.90    77.31     0.26     0.42     4096070       12093
col           638    1064.40      49.01     2199.00    21.72     0.08     0.02    23635295       16903
line        27184    1036.03      93.14     1941.33    11.12     0.00     0.05      925447      266142
nroff12        29     909.83      17.71       56.97    31.38     0.61     0.31  [illegible]  [illegible]
delta         264     904.54      23.07      254.06    39.21     0.09     0.09    21219141       47164
td            173     886.19      25.74      139.73    14.43     0.15     0.16  [illegible]  [illegible]
ar           1434     872.65      61.87      309.07    11.11     0.04     0.20  [illegible]  [illegible]
m2findD       144     864.29      12.54      344.13    66.94     0.09     0.04  [illegible]  [illegible]
rm          15319     857.97      85.65      754.20    10.02     0.01     0.11  [illegible]      167803
acctdusg        1     819.77      39.30      170.10    20.86    39.30     0.23     1412190       19714
f77pass1      155     779.13       7.97       39.09    97.70     0.05     0.27  [illegible]  [illegible]
diff          796     767.31      32.77      260.27    23.41     0.04     0.13    25940094       47214
```

Figure C-5. Sample Monthly Total Command Summary

The data in Figures C-4 and C-5 are defined as follows:

COMMAND NAME This is 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, only 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 superuser privileges were used.

NUMBER CMDS

This is 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

The 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

This is a relative measurement of the ratio of system availability to system utilization. It is computed by the formula

(total CPU time) / (elapsed time)

This gives a relative measure of the total available processor time consumed by the process during its execution.

CHARS TRNSFD

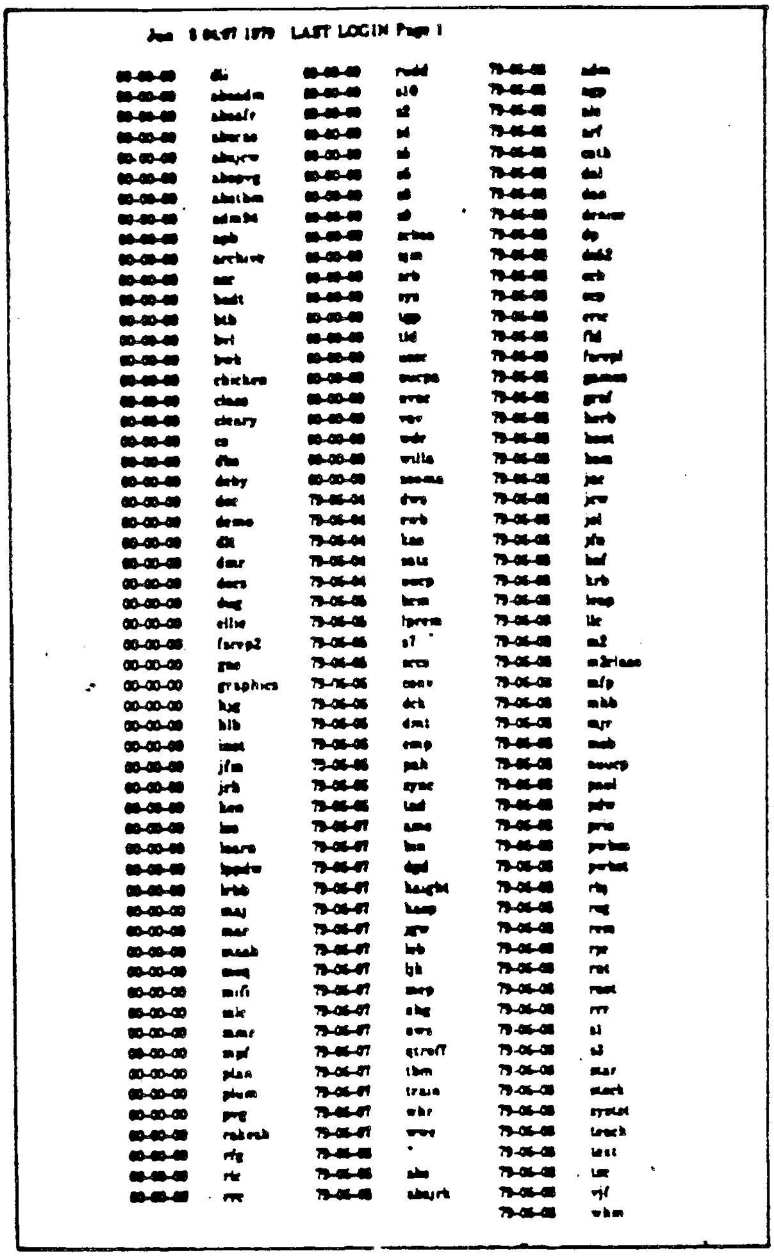
This column, which may go negative, is a total count of the number of characters pushed around by the read(2) and write(2) system calls.

BLOCKS READ

A total count of the physical block reads and writes that a process performed.

Last Login

This report simply gives the date when a particular login was last used. This could be a good source for finding likely candidates for the tape archives or getting rid of unused logins and login directories. Figure C-6 is a sample login report.



```
        Jun  8 06:07 1979  LAST LOGIN Page 1

00-00-00   di         00-00-00   rudd       79-06-08   adm
00-00-00   abnadm     00-00-00   sl0        79-06-08   agp
00-00-00   abnafr     00-00-00   s2         79-06-08   alo
00-00-00   aburae     00-00-00   s4         79-06-08   arf
00-00-00   abuyrw     00-00-00   s5         79-06-08   cnth
00-00-00   abeprg     00-00-00   s6         79-06-08   dad
00-00-00   abutbm     00-00-00   s8         79-06-08   dno
00-00-00   adm36      00-00-00   s9         79-06-08   drawr
00-00-00   apb        00-00-00   arbon      79-06-08   dp
00-00-00   archive    00-00-00   sgm        79-06-08   dsk2
00-00-00   asr        00-00-00   srb        79-06-08   erb
00-00-00   badt       00-00-00   sys        79-06-08   ecp
00-00-00   bth        00-00-00   tgp        79-06-08   erne
00-00-00   brl        00-00-00   tld        79-06-08   fld
00-00-00   bwk        00-00-00   unc        79-06-08   farvpl
00-00-00   chicken    00-00-00   uucpa      79-06-08   games
00-00-00   class      00-00-00   vvac       79-06-08   graf
00-00-00   cleary     00-00-00   vev        79-06-08   herb
00-00-00   co         00-00-00   wdr        79-06-08   hunt
00-00-00   dfm        00-00-00   wills      79-06-08   ham
00-00-00   derby      00-00-00   zooma      79-06-08   jar
00-00-00   doc        79-06-04   dws        79-06-08   jcw
00-00-00   demo       79-06-04   rwb        79-06-08   jml
00-00-00   dlt        79-06-04   kas        79-06-08   jfe
00-00-00   dmr        79-06-04   sats       79-06-08   laf
00-00-00   docs       79-06-04   uucp       79-06-08   lrb
00-00-00   dug        79-06-05   brm        79-06-08   lmp
00-00-00   ellie      79-06-05   lprem      79-06-08   lle
00-00-00   farvp2     79-06-05   s7         79-06-08   m2
00-00-00   gne        79-06-05   arcs       79-06-08   m2rlaac
00-00-00   graphics   79-76-05   conv       79-06-08   mfp
00-00-00   hug        79-06-05   dch        79-06-08   mhb
00-00-00   hlb        79-06-05   dmt        79-06-08   mjr
00-00-00   inst       79-06-05   emp        79-06-08   mab
00-00-00   jfm        79-06-05   psh        79-06-08   nuucp
00-00-00   jrh        79-06-05   sync       79-06-08   paul
00-00-00   ken        79-06-05   tad        79-06-08   pdw
00-00-00   kss        79-06-07   amo        79-06-08   prio
00-00-00   learn      79-06-07   bsn        79-06-08   purbm
00-00-00   lpadw      79-06-07   dgd        79-06-08   purbat
00-00-00   lrbb       79-06-07   haught     79-06-08   rlq
00-00-00   maj        79-06-07   kamp       79-06-08   rug
00-00-00   mar        79-06-07   jgw        79-06-08   rwm
00-00-00   mash       79-06-07   lrb        79-06-08   rjr
00-00-00   meq        79-06-07   tjh        79-06-08   rnt
00-00-00   mifi       79-06-07   merp       79-06-08   root
00-00-00   mlc        79-06-07   shg        79-06-08   rrr
00-00-00   mmr        79-06-07   ows        79-06-08   s1
00-00-00   mpf        79-06-07   qtroff     79-06-08   s3
00-00-00   plan       79-06-07   tbm        79-06-08   star
00-00-00   plum       79-06-07   train      79-06-08   stark
00-00-00   prg        79-06-07   whr        79-06-08   sysist
00-00-00   rakesh     79-06-07   wwv        79-06-08   teach
00-00-00   rfg        79-06-08   *          79-06-08   test
00-00-00   rlc        79-06-08   abn        79-06-08   tuc
00-00-00   rrc        79-06-08   abuyrh     79-06-08   vjf
                                           79-06-08   whm
```

Figure C-6. Sample Login Report

SYSTEM ACCOUNTING DATA FILES

Figures C-7 through C-11 provide a description of system accounting data files.

```
/*      %W%      */
/*      <sys/types.h> must be included.                         */
#define UTMP_FILE       "/etc/utmp"
#define WTMP_FILE       "/etc/wtmp"
#define ut_name ut_user

struct  utmp
  {
        char ut_user[8] ;              /* User login name */
        char ut_id[4] ;                /* /etc/lines id (usually line #) */
        char ut_line[12] ;             /* device name (console, lnxx) */
        short ut_pid ;                 /* process id */
        short ut_type ;                /* type of entry */
        struct exit_status
          {
            short e_termination ;      /* Process termination status */
            short e_exit ;             /* Process exit status */

          }
        ut_exit ;                      /* The exit status of a process
                                        * marked as DEAD_PROCESS.
                                        */

        time_t ut_times                /* time entry was made */

  };
/*       Definitions for ut_type

#define EMPTY           0
#define RUN_LVL         1
#define BOOT_TIME       2
#define OLD_TIME        3
#define NEW_TIME        4
#define INIT_PROCESS    5    /* Process spawned by "init" */
#define LOGIN_PROCESS   6    /* A "getty" process waiting for login */
#define USER_PROCESS    7    /* A user process */
#define DEAD_PROCESS    8
#define ACCOUNTING      9

#define UTMAXTYPE       ACCOUNTING       /* Largest legal value of ut_type */

/*    Special strings or formats used in the "ut_line" field when        */
/*    accounting for something other than a process.                     */
/*    No string for the ut_line field can be more than 11 chars +        */
/*    a NULL in length.                                                  */

#define RUNLVL_MSG      "run-level %c"
#define BOOT_MSG        "system boot"
#define OTIME_MSG       "old time"
#define NTIME_MSG       "new time"
```

Figure C-7. Format of utmp Files (utmp.h)

```
/*      %W% of %G%          */
/*
 *      defines, typedefs, and so on used by acct programs
 */


/*
 *      acct only typedefs
 */
typedef unsigned short  uid_t;

#ifdef u3b
#define HZ          100
#else
#define HZ          60
#endif

#define LSZ         12      /* sizeof line name */
#define NSZ          8      /* sizeof login name */
#define P            0      /* prime time */
#define NP           1      /* nonprime time */

/*
 *      limits that may have to be increased if systems get larger
 */
#define SSIZE       1000    /* max number of sessions in 1 acct run */
#define TSIZE       100     /* max number of line names in 1 acct run */
#define USIZE       500     /* max number of distinct login names in 1 acct run */

#define EQN(s1, s2)         (strncmp(s1, s2, sizeof(s1)) == 0)
#define CPYN(s1, s2)        strncpy(s1, s2, sizeof(s1))

#define SECSINDAY           86400L
#define SECS(tics)          ((double) tics)/HZ
#define MINS(secs)          ((double) secs)/60
#define MINT(tics)          ((double)tics)/(60*HZ)

#ifdef pdp11
#define KCORE(clicks)       ((double) clicks/16)
#endif
#ifdef vax
#define KCORE(clicks)       ((double) clicks/2)
#endif
#ifdef u3b
#define KCORE(clicks)       ((double) clicks*2)
#endif
```

Figure C-8. Accounting Program Definitions (acctdef.h)

```
/*
 * Accounting structures
 */
typedef ushort comp_t;          /* "floating point" */
               /* 13-bit fraction, 3-bit exponent */

struct  acct

        char    ac_flag         /* Accounting flag */
        char    ac_stat;        /* Exit status */
        ushort  ac_uid;         /* Accounting user ID */
        ushort  ac_gid;         /* Accounting group ID */
        dev_t   ac_tty;         /* control typewriter */
        time_t  ac_btime;       /* Beginning time */
        comp_t  ac_utime;       /* acctng user time in clock ticks */
        comp_t  ac_stime;       /* acctng system time in clock ticks */
        comp_t  ac_etime;       /* acctng elapsed time in clock ticks */
        comp_t  ac_mem;         /* memory usage */
        comp_t  ac_io;          /* chars transferred */
        comp_t  ac_rw;          /* blocks read or written */
        char    ac_comm[8];     /* command name */

;

extern  struct  acct   acctbuf;
extern  struct  inode  *acctp;   /* inode of accounting file */

#defineAFORK           01        /* has executed fork, but no exec */
#defineASU             02        /* used superuser privileges */
#defineACCTF           0300      /* record type: 00 = acct */
```

Figure C-9. Format.of pacct Files (acct.h)

```
/*
 *      total accounting (for acct period) also for day
 */

struct  tacct
        uid_t           ta_uid;         /* userid */
        char            ta_name[8];     /* login name */
        float           ta_cpu[2];      /* cum. cpu time, p/np (mins) */
        float           ta_kcore[2];    /* cum. kcore-minutes, p/np */
        float           ta_con[2];      /* cum. connect time, p/np, mins */
        float           ta_du;          /* cum. disk usage */
        long            ta_pc;          /* count of processes */
        unsigned short  ta_sc;          /* count of login sessions */
        unsigned short  ta_dc;          /* count of disk samples */
        unsigned short  ta_fee;         /* fee for special services */
```

Figure C-10. Format of tacct Files (tacct.h)

```
/*
 *      connect time record (various intermediate files)
 */
struct  ctmp
        dev_t    ct_tty;              /* major minor */
        uid_t    ct_uid;              /* userid */
        char     ct_name[8];          /* login name */
        long     ct_con[2];           /* connect time (p/np) secs */
        time_t   ct_start;            /* session start time */
```

Figure C-11. Format of ctmp File (ctmp.h)

ACCOUNTING SYSTEM FILES

The files described below are the files used by the accounting system.

Files in the /usr/adm directory:

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 through turnacct

Spacct?.MMDD — process accounting files for MMDD during execution of runacct

wtmp — active wtmp file for recording connect sessions

Files in the /usr/adm/acct/nite directory:

active — used by runacct to record progress and print warning 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-sesion records in ctmp.h format

daycms — ASCII daily command summary used by prdaily

dayacct	total accounting records for one 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

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
rprt.MMDD	saved output of prdaily program
tacct	cumulative total accounting file for current fiscal
tacctprev	same as tacct without latest update

tacct.MMDD	total accounting file for MMDD
wtmp.MMDD	saved copy of wtmp file for MMDD, removed after reboot by remove procedure

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?

SUMMARY

The system accounting for this operating system was designed from a 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 charging purposes.

Appendix D
lp Spooling System

The lp system of commands performs diverse spooling functions under the operating system. lp allows administrators to customize the system to spool to a collection of printers of any type and to group printers into logical classes in order to maximize the throughput of the devices. Users can queue and cancel print requests, prevent and allow queuing to and print on specific devices, start and stop lp processing requests, change configuration of printers, and find the status of the lp system. This section describes how the administrator performs restricted functions and oversees lp operation.

DEFINITIONS

Several terms must be defined before presenting a brief summary of lp commands. The lp was designed with the flexibility to meet the needs of users on different UNIX-derived operating systems. Changes to the lp configuration are performed by the lpadmin(1M) command.

lp makes a distinction between printers and printing devices. A device is a physical peripheral device or a file and is represented by a full UNIX-derived operating system pathname. A printer is a logical name that represents a device. At different points in time, a printer may be associated with different devices. A class is a name given to an ordered list of printers. Every class must contain at least one printer. Each printer may be a member of zero or more classes. A destination is a printer or a class. One destination may be designated as the system default destination. The lp(1) command will direct all output to this destination unless the user specifies otherwise. Output that is routed to a printer will be printed only by that printer, whereas output directed to a class will be printed by the first available class member.

Each invocation of lp creates an output request that consists of the files to be printed and options from the lp command line. An interface program that formats requests must be supplied for each printer. The lp scheduler, lpsched(1M), services requests for all destinations by routing requests to interface programs to do the printing on devices. An lp configuration for a system consists of devices, destinations, and interface programs.

COMMANDS FOR GENERAL USE

The lp(1) command is used to request the printing of files. It creates an output request and returns to the user a request ID of the form

```
    dest-seqno
```

where

seqno is a unique sequence number across the entire lp system.

dest is the destination where the request was routed.

Cancel is used to cancel output requests. The user supplies request ID's as returned by lp or printer names, in which case the currently printing requests on those printers are cancelled.

Disable prevents lpsched from routing output requests to printers.

Enable(1) allows lpsched to route output requests to printers.

COMMANDS FOR lp ADMINISTRATORS

Each lp system must designate a person or persons as LP administrator to perform the restricted functions listed below. Either the superuser or any user who is logged into the operating system as lp qualifies as an lp administrator. All lp files and commands are owned by lp, except for lpadmin and lpsched, which are owned by root. The following commands are described inmore detail later in this subsection.

lpadmin(1M) Modifies lp configuration. Many features of this command cannot be used when lpsched is running.

lpsched(1M) Routes output requests to interface programs that do the printing on devices.

lpshut Stops lpsched from running. All printing activity is halted, but other lp comands can still be used.

accept(1M) Allows lp to accept output requests for destinations.

reject Prevents lp from accepting requests for destinations.

lpmove Moves output requests from one destination to another. Whole destinations can be moved at once. This command cannot be used when lpsched is running.

CONFIGURING lp WITH LP.CNFG

Lp.cnfg(1M) configures and maintains the print system (lp). Through a series of menus and prompts, lp.cnfg lets you:

- Delete, add, and change printer configurations
- Enable and disable printers
- Display printers and printer statuses
- Install national language character-set translation tables

To run lp.cnfg, enter the following command:

```
/etc/lp.cnfg
```

You must be logged on as root or lp. Only one person at a time can run this utility. If you are not logged on as root or lp, this error message appears:

```
lp.cnfg: You must either be "root", or "lp" to use this utility.
```

If someone else is already using lp.cnfg, this message is displayed:

```
lp.cnfg: Only one person may use this utility at a time.
```

When you start lp.cnfg, the following message appears:

```
--- Now initializing the Lp configuration utility ---
```

The program then displays the Lp System Configuration Utility menu.

```
Lp System Configuration Utility

1:   Delete all previous Lp configurations.
2:   Add printer configurations into Lp.
3:   Change a printer's configuration.
4:   Delete one printer from the Lp configuration.
5:   Enable a printer.
6:   Disable a printer.
7:   Display a list of all known printers.
8:   Display the status of all printers.
9:   Install language character set translation tables.

Enter the number of the function you want to perform.
Hit the <RETURN> key when you have finished:
```

To choose a function, type the number indicated and press RETURN. After you complete a function, you will return to this main menu. Generally, you can exit a function without using it by pressing RETURN when the first prompt appears. Once you have passed the initial prompt, you must complete the function. Expressions in angle brackets (< >) in the messages below indicate terms you entered, such as printer names.

If you enter something other than a valid number or RETURN, the following error message appears:

```
    *** You must enter a valid number or <RETURN> ***
```

Enter a number and press RETURN.

Deleting All Previous Lp Configurations

This function deletes all lp configurations except those of active printers; you can't delete printers that are active. Deleting an entire configuration is usually done when you want to reconfigure all your printers. To avoid accidental deletion, this warning appears:

```
    You are about to delete the systems entire printer configuration

         --Press "C" and <Return> to continue

         --or press just <Return> to abort this function:
```

To delete the entire configuration, press C and RETURN. Lp.cnfg deletes all the printers defined on your system.

If you do not want to delete the printer configuration, press RETURN. The following message appears:

```
    *** function aborted
```

After you delete the configuration or abort the function, the main menu returns.

Adding Printer Configurations to Lp

Use this function to add printer configurations to the print system. The prompt is as follows:

```
    Add printer configurations into Lp

    Please enter the printer device name
    (Such as tty001 for /dev/tty001
    or tp1021 for /dev/tp1021):
```

If you don't want to add printer configurations, press RETURN to go back to the main menu.

Otherwise, enter the name of the physical device to which the printer being configured is attached, and press RETURN. Devices are named as follows:

For a serial printer, the device name is the tty number to which the printer is connected. For example, the second serial port on a System 6300 (labeled either CHANNEL 1 or RS232B) corresponds to device tty001.

If the printer is attached to an RS-422 terminal, the device number is the same as the terminal's tty number, but is preceded by "tp1". Thus, for example, a printer connected to an RS-422 terminal whose tty device number is tty021 has a device name of tp1021.

The device name for a parallel printer attached to a parallel port on a System 6300 is lp.

After you enter a device name, lp.cnfg checks if the device exists and if there is a program is running on it. If the device you named doesn't exist, this error message appears:

```
*** /dev/<device name >isn't a valid device name. ***
```

Reenter a valid device name when the device-name prompt reappears.

If the device you named has a program running on it, the following error message is displayed:

```
*** You must not have anything running on /dev/<device name>.***
```

Reenter the device name when the device-name prompt reappears. If you need to stop programs running on the device, press RETURN to go back to the main menu, then press RETURN to exit lp.cnfg

Name the printer with the following prompt:

```
What name do you want to give this printer?
```

The name must:

- be no longer than 14 characters
- consist of alphanumeric and underscore characters only
- be a unique printer name
- not be lpr or cp

Enter the name and press RETURN. Lp.cnfg checks if the name is a valid printer name. If it is not, then one of the following error messages appears:

```
*** You must enter a printer name. ***

*** Printer "<printer name>" is already defined in lp. ***

*** The size of the name is too large.  Please pick a new printer name. ***

*** Printer "<printer name>" is not a legal printer name. ***
*** You can only use characters "A-Z", "a-z", "0-9", and "_". ***

*** You may not use either "cpr", or "lpr" as printer names. ***
```

If any of these errors occur, the printer device-name prompt reappears and you must enter another name.

If there were no errors, the printer-type prompt appears:

```
What printer type do you have on "<device name>"?
(Such as pt30, pt31, pt32, pt34, pt35, pt36):
```

Enter the printer type and press RETURN. Valid printer types are pt30, pt31, pt32, pt34, pt35, and pt36. If you don't enter a printer type, the following error message appears:

```
*** You must enter a printer type. ***
```

If you enter an invalid printer type, this error message is diplayed:

```
*** Printer type "<type>" is not a supported printer. ***
```

If either of these errors occurs, the printer device-name prompt reappears and you must enter the information again.

If there were no errors, the program asks for the language the printer will use.

```
What language is this printer going to be printing?
(Default is engli):
```

Enter the abbreviation for the language you want the printer to use and press RETURN. Press RETURN to specify the default, English (U.S.A.). The PT31 Character Printer supports only English (U.S.A.) and the PT34 Character Printer supports all listed languages except Dutch. The languages are:

Name	Language
cdn	French-Canadian
deut	German
engli	English (U.S.A.)
esp	Spanish
fra	French
hol	Dutch
sve	Swedish
uk	English (U.K.)

If you entered a language that is not supported for the printer type, the error message below appears:

```
*** Printer type "<printer type>" doesn't support the "<language>"
language. ***
```

The printer device-name prompt reappears and you must reenter all the data for this function.

When you have finished entering information, the following message is displayed:

```
--- Now defining the "lp" printer configuration. ---
destination "<printer name>" now accepting requests
printer "<printer name>" now enabled
destination "cpr" now accepting requests
```

The printer you just configured can now accept print requests and those requests will be printed. The printer is a member of the class or group of printers called cpr. Any request sent to cpr instead of a specific printer name will be printed by the first available printer on the system.

The main menu returns.

If you configured a printer that is attached to a terminal, you should add this line to the user additions section of the .profile file of that terminal's user:

```
. /usr/spool/lp/setlocalptr
```

Changing a Printer's Configuration

Use this function to change the configuration of a printer. The prompt below appears:

```
Change A Printer Configuration

Enter the printer's name:
```

If you want to leave this function without changing any configurations, press RETURN. The main menu appears.

Enter the name of the printer you want to reconfigure and press RETURN. The following message is displayed:

```
    Working Please Wait
```

If the name you entered is for an undefined printer, this error message appears:

```
    *** Printer "<printer name>" doesn't exist. ***
```

Reenter the printer name when the printer-name prompt reappears.

After you enter the printer name, the printer-configuration-change menu is displayed:

```
    Printer "<printer name>" is currently attached to device "<device name>"

         Printer type = <printer type>
         Language = <language>


    Select the item you wish to change

         1:  Change device "<printer name>" is attached to
         2:  Change printer type of "<printer name>"
         3:  Change language of "<printer name>"

    --Enter the number of the function you want to perform and <Return>
    --or just press <Return> to end changes:
```

where,

<printer name> is the name of the printer being reconfigured.

<device name> is the name of the device to which the printer is currently attached.

<printer type> is the type of printer, such as pt30 for a PT30 Character Printer.

<language> is the language the printer uses.

Type the number of a function and press RETURN. If you enter an invalid number, this message appears:

```
    *** You have entered an invalid function. ***
```

Enter a valid number and press RETURN, or press RETURN to go back to the main menu.

CHANGING THE PRINTER'S PHYSICAL DEVICE

Use this function to change the physical device to which the printer is connected. The prompt below appears:

```
Current device for printer "<printer name>" is <device name>

Please enter the new device name for printer "<printer name>"
(Such as tty001 for /dev/tty001
or tp1021 for /dev/tp1021):
```

Enter the device name and press RETURN. The naming conventions are as follows:

For a serial printer, the device name is the tty number to which the printer is connected. For example, the second serial port on a System 6300 (labeled either CHANNEL 1 or RS232B) corresponds to device tty001.

If the printer is attached to an RS-422 terminal, the device number is the same as the terminal's tty number, but is preceded by "tp1". Thus, for example, a printer connected to an RS-422 terminal whose tty device number is tty021 has a device name of tp1021.

The device name for a parallel printer attached to a parallel port on a System 6300 is lp.

After you enter the device name, lp.cnfg checks if the device exists and if a program is running on it. If you don't enter a printer device-name, this error message appears:

```
*** You must enter a printer device name. ***
```

If the device you named doesn't exist, this error message appears:

```
*** "/dev/<device name>" isn't a valid device. ***
```

For both of these errors, enter the device name when the prompt appears.If the device you named has a program running on it, the following error message appears:

```
*** You must not have anything running on /dev/<device name>. ***
```

The printer-configuration-change menu reappears. If you need to stop programs running on the device, press RETURN to go back to the main menu, then press RETURN to exit lp.cnfg

If the device exists and no program is running on it, this message appears:

```
Now changing printer "<printer name>" device /dev/<old device name> to <new
device name>
```

This message indicates that the printer's device is being changed in the printer's configuration. The printer-configuration-change menu is then displayed.

CHANGING PRINTER TYPES

Select this option to change the printer type. The following prompt appears:

```
Printer "<printer name>" is currently type "<current printer type>"

What type of printer do you want to change printer "<printer name>" to?
(Such as pt30, pt31, pt32, pt34, pt35, pt36):
```

You must enter a printer type; valid types are pt30, pt31, pt32, pt34, pt35, and pt36. If you don't enter a type, this error message appears:

```
*** You must enter a printer type. ***
```

If you enter an invalid printer type, the following error message is displayed:

```
*** Printer type "<printer type>" is not a supported printer. ***
```

If either of these errors occurs, the printer-configuration-change menu reappears. If you change a printer type to one that doesn't support the previously-configured language, this error message appears:

```
*** Printer type "<new printer type>" doesn't support the "<language>"
language. ***
```

The change-printer-type prompt reappears.

After you enter the printer type, this message appears:

```
Now changing printer "<printer name>" from type "<old type>" to type "<new
type>"
```

This message indicates that the printer type is being changed in the printer's configuration. The printer-configuration-change menu reappears.

CHANGING PRINTER LANGUAGES

Choose this function to change the language the printer uses. The prompt below is displayed:

```
Printer "<printer name>" currently prints in "<current language>"

What language is this printer going to be printing?
(Default is engli):
```

Enter the abbreviation for the language you want the printer to use and press RETURN. Press RETURN to specify the default, which is English (U.S.A.). The PT31 Character Printer supports only English (U.S.A.) and the PT34 Character Printer supports all listed languages except Dutch. The languages are:

Name	Language
cdn	French-Canadian
deut	German
engli	English (U.S.A.)
esp	Spanish
fra	French
hol	Dutch
sve	Swedish
uk	English (U.K.)

If you enter a language that is not supported for the printer type, this error message appears:

```
*** Printer type "<printer type>" doesn't support the "<language>"
language. ***
```

The printer-configuration-change menu reappears and you must reenter the data.

After you enter the language abbreviation, this message is displayed:

```
Now changing printer "<printer name>" from language "<old language>" to
"<new language>"
```

The language in the printer's configuration is being changed. The printer-configuration-change menu reappears.

If you configured a printer that is attached to a terminal, you should add this line to the user additions section of the .profile file of that terminal's user:

```
. /usr/spool/lp/setlocalptr
```

Deleting a Printer from the Lp Configuration

Choose this function to delete the configuration of one printer from lp. You can't delete a printer that is active. The prompt below appears:

```
Delete one printer from the Lp configuration

What printer do you want to delete?
```

If you don't want to delete a printer, press RETURN to go back to the main menu.

Enter the name of the printer you want to delete. If you enter the name of a nonexistent printer, this error message appears:

```
    *** Printer "<printer name>" doesn't exist. ***
```

Enter the printer name when the printer-name prompt returns.

After you enter the printer name, this warning appears:

```
    Do you really want to delete printer "<printer name>"?
```

To delete the printer, press Y and RETURN; to abort this function, press RETURN. If you entered Y, the message below appears to confirm that the printer's configuration has been deleted.

```
    --- Printer "<printer name>" has been deleted. ---
```

The main menu reappears.

Enabling a Printer

Use this function to restart a printer that has been disabled. If a printer has been disabled, such as for a paper change or for repairs, the system needs to know that the printer is ready for service again. You don't need to use this function for newly configured printers, since they are automatically enabled. The following prompt appears:

```
    Enable a printer

    What printer do you want to enable?
```

To exit the function without enabling a printer, press RETURN. The main menu appears.

Enter the name of the printer you want to enable and press RETURN. If the system is not configured for the printer you named, this error message appears:

```
    *** Printer "<printer name>" doesn't exist. ***
```

Enter another name when the printer-name prompt returns.

After you enter the printer name, this message appears:

```
    printer "<printer name>" now enabled
         destination "<printer name>" now accepting requests
```

The printer has been enabled and is accepting print requests. The main menu returns.

Disabling a Printer

Select this function to disable a printer. You may need to disable a printer to change the paper, ribbon, or print wheel, or to take it off line for repair. The following prompt appears:

```
Disable a printer

What printer do you want to disable?
```

To exit this function without disabling a printer, press RETURN. The main menu appears.

Enter the name of the printer you want to disable and press RETURN. If the system is not configured for the printer you named, this error message appears:

```
*** Printer "<printer name>" doesn't exist. ***
```

Enter another name when the printer-name prompt returns.

After you enter a printer name, this message is displayed:

```
printer "<printer name>" now disabled
destination "<printer name>" is no longer accepting requests
```

The printer has been stopped and will no longer accept print requests. The main menu returns.

Displaying a List of All Known Printers

This function lists the printers and the devices to which they are attached. Each printer that has a configuration is listed. The display is in this format:

```
Display a list of all known printers

Printer "<printer name>" is currently connected to device /dev/<device name>

Hit the <RETURN> key to return to the main menu.
```

If no printers are assigned, this message appears:

```
Display a list of all known printers

You don't have any printers currently assigned on your system.

Hit the <RETURN> key to return to the main menu.
```

To return to the main menu, press RETURN.

Displaying the Status of All Printers

This function shows the status of all printers on the system. The display is in this format:

```
         Display the status of all printers


    printer <printer name> disabled since Feb 16 08:24 -
         <reason>
    <printer name> not accepting requests since Feb 17 12:32 -
         <reason>
    <printer name> accepting requests since Feb 10 15:00


    Hit the <RETURN> key to return to the main menu.
```

If no printers are assigned, this message appears:

```
    Display the status of all printers

    You don't have any printers currently assigned on your system.

    Hit the <RETURN> key to return to the main menu.
```

Press RETURN to return to the main menu.

Installing Printer Language-Translation Tables

Use this function to install additional printer language support for systems with the national character-set upgrade package. This extra support lets you configure printers to print in different languages. See the operator's manuals for these printers to determine what, if any, hardware changes to the printers are needed before they can print in a new language.

The following prompt is displayed:

```
         Install Language Character Translate Tables

    Please mount the Printer Character Set
    translation table floppy
    and press the key marked <Return> :
```

To exit this function, press Q and RETURN. The message below is displayed, the function is aborted, and the main menu returns.

```
    Aborting terminal configuration
```

If you want to include printer languages, mount the national language support diskette containing printer translation-tables and press RETURN. The following menu appears:

```
Select A Printer Type That You Have On Your System:

Available identifiers are:-

Type Description

pt30 Diablo D630 Printer
pt32 150CPS 132 Column Matrix Printer
pt34 200/50 CPS matrix printer
pt35 Fujitsu 55 CPS printer
pt36 600 lpm band printer

--Respond by entering the printer type
followed by the key marked "Return"

--or just press the key marked "Return" when you have
completed your selections.
```

Enter the type of printer and press RETURN. The valid types are pt30, pt32, pt34, pt35, and pt36. If you enter an invalid type, this error message appears:

```
Sorry I do not recognize <printer type> as a printer,
please retry or contact your Customer Service Representative
```

Enter the printer type again when the printer-type menu is redisplayed. If this message appears again, call your Customer Support Representative for assistance.

After you select a printer type, the language-selection menu appears:

```
Select a Language Character Set Translation Table

     Available languages are:-

     deut      German language
     sve       Swedish language
     fra       French language
     esp       Spanish language
     uk        United Kingdom variant
     hol       Dutch language
     engli     USA variant
     cdn       French Canadian variant

--Respond by entering the language character set translation
table you require to be installed for printer type <printer type> followed
by the key marked "Return"

--or just press the key marked "Return" when you have
completed your selections.
```

Enter the language abbreviation and press RETURN. The language translation-table is installed and the language-selection menu reappears. Note that the PT34 Character Printer does not support Dutch.

If you enter an invalid or misspelled language abbreviation, this error message appears:

```
Sorry language variant lll is not supported in this release, please try
again or contact your nearest Customer Support Representative
```

where

lll is the language abbreviation you specified. Enter the language abbreviation again when the language character-set translation-table menu is redisplayed. If this message appears again, call your Customer Support Representative for assistance.

After you have selected all the language translation-tables you need for this printer type, press RETURN. The message below indicates that the language translation-tables for the printer have been installed:

```
Installation for printer pppp complete
```

where

pppp is the printer type. The printer-type selection menu is redisplayed so you can select another printer type. When you have added all the languages you want to the printers on your system, press RETURN. The printers that don't have other languages specified default to English (U.S.A.). The message below indicates that the print system is completely configured.

```
Print system file installation complete.
```

The main menu returns.

CONFIGURING lp: THE lpadmin COMMAND

Changes to the lp configuration should be made by using the lpadmin command and not manually. Note that you can also use the lp.cnfg command, described under "Configuring lp with lp.cnfg" earlier in this section. lpadmin will not attempt to alter the lp configuration when lpsched is running, except where explicitly noted below.

Introducing New Destinations

The following information must be supplied to lpadmin when introducing a new printer.

o The printer name (-p printer) which is an arbitrary name that must conform to the following rules:

  1. It must be no longer than 14 characters.

  2. It must consist solely of alphanumeric characters and underscores.

  3. It must not be the name of an existing lp destination (printer or class).

o The printer interface program. This may be specified in one of three ways:

  1. It can be selected from a list of model interfaces supplied with lp (-m model).

  2. It can be the same interface that an existing printer uses (-e printer).

  3. It can be a program supplied by the lp administrator (-i interface).

o The device associated with the printer (-v device). This is the pathname of a hardwired printer, a login terminal, or other file that is writable by lp.

Information that need not always be supplied when creating a new printer includes:

o The user can specify -h to indicate that the device for the printer is hardwired or the device is the name of a file (this is assumed by default). If, on the other hand, the device is the pathname of a login terminal, then -l must be included on the command line. This indicates to lpsched that it must automatically disable this printer each time lpsched starts running. This fact is reported by lpstat when it indicates printer status:

```
$ lpstat -pa
printer a (login terminal) disabled Oct 31 11:15 --
     disabled by scheduler: login terminal
```

This is done because device names for login terminals can be (and usually are) associated with different physical devices from day to day. If the scheduler did not take this action, somebody might log in and be surprised that lp is spooling to his or her terminal!

- The new printer may be added to an existing class or added to a new class (-cclass). New class names must conform to the same rules for new printer names.

Examples:

The following examples will be referenced by further examples in later subsections.

1. Create a printer called pr1 whose device is /dev/printer and whose interface program is the model hp interface:

```
$ /usr/lib/lpadmin -ppr1 -v/dev/printer -mhp
```

2. Add a printer called pr2 whose device is /dev/tty022 and whose interface is a variation of the model prx interface. It is also a login terminal:

```
$ cp /usr/spool/lp/model/prx xxx
  < edit xxx >
$ /usr/lib/lpadmin -ppr2 -v/dev/tty022 -ixxx -1
```

3. Create a printer called pr3 whose device is /dev/tty023. The pr3 will be added to a new class called cl1 and will use the same interface as printer pr2:

```
$ /usr/lib/lpadmin -ppr3 -v/dev/tty023 -epr2 -ccl1
```

Modifying Existing Destinations

Modifications to existing destinations must always be made with respect to a printer name (-p printer). The modifications can be one or more of the following:

- The device for the printer can be changed (-v device). If this is the only modification, then this can be done even while lpsched is running. This facilitates changing devices for login terminals.

- The printer interface program can be changed (-m model, -e printer, -i interface).

- The printer can be specified as hardwired (-h) or as a login terminal (-1).

- The printer can be added to a new or existing class (-cclass).

o The printer can be removed from an existing class (-r class). Removing the last remaining member of a class causes the class to be deleted. No destination can be removed if it has pending requests. In that case, lpmove or cancel should be used to move or delete the pending requests.

Examples:

These examples are based on the lp configuration created by those on the previous page.

1. Add printer pr2 to class cl1:

```
$ /usr/lib/lpadmin -ppr2 -ccl1
```

2. Change pr2's interface program to the model prx interface, change its device to /dev/tty024, and add it to a new class called cl2:

```
$ /usr/lib/lpadmin -ppr2 -mprx -v/dev/tty024 -ccl2
```

Note that printers pr2 and pr3 now use different interface programs even though pr3 was originally created with the same interface as pr2. Printer pr2 is now a member of two classes.

3. Specify printer pr2 as a hardwired printer:

```
$ /usr/lib/lpadmin -ppr2 -h
```

4. Add printer pr1 to class cl2:

```
$ /usr/lib/lpadmin -pprl -ccl2
```

The members of class cl2 are now pr2 and pr1, in that order. Requests routed to class cl2 will be serviced by pr2 if both pr2 and pr1 are ready to print; otherwise, they will be printed by the one that is next ready to print.

5. Remove printers pr2 and pr3 from class cl1:

```
$ /usr/lib/lpadmin -ppr2 -rcl1
$ /usr/lib/lpadmin -ppr3 -rcl1
```

Since pr3 was the last remaining member of class cl1, the class is removed.

6. Add pr3 to a new class called cl3.

```
$ /usr/lib/lpadmin -ppr3 -ccl3
```

Specifying the System Default Destination

The system default destination can be changed even when lpsched is running.

Examples:

1. Establish class cl1 as the system default destination:

```
$ /usr/lib/lpadmin -dcl1
```

2. Establish no default destination:

```
$ /usr/lib/lpadmin -d
```

Removing Destinations

Classes and printers can be removed only if there are no pending requests that were routed to them. Pending requests must either be cancelled using cancel or moved to other destinations using lpmove before destinations can be removed. If the removed destination is the system default destination, then the system will have no default destination until the default destination is respecified. When the last remaining member of a class is removed, then the class is also removed. The removal of a class never implies the removal of printers.

Examples:

1. Make printer pr1 the system default destination:

```
$ /usr/lib/lpadmin -dpr1
```

   Remove printer pr1:

```
$ /usr/lib/lpadmin -xpr1
```

   Now there is no system default destination.

2. Remove printer pr2:

```
$ /usr/lib/lpadmin -xpr2
```

   Class cl2 is also removed since pr2 was its only member.

3. Remove class cl3:

```
$ /usr/lib/lpadmin -xcl3
```

   Class cl3 is removed, but printer pr3 remains.

MAKING AN OUTPUT REQUEST: THE lp COMMAND

Once lp destinations have been created, users can request output by using the lp command. The request ID that is returned can be used to see if the request has been printed or to cancel the request.

The lp program determines the destination of a request by checking the following list in order:

1. If the user specifies -d dest on the command line, then the request is routed to dest.

2. If the environment variable LPDEST is set, the request is routed to the value of LPDEST.

3. If there is a system default destination, then the request is routed there.

4. Otherwise, the request is rejected.

Examples:

1. There are at least four ways to print the password file on the system default destination:

```
lp /etc/passwd
lp < /etc/passwd
cat /etc/passwd¦lp
lp -c/etc/passwd
```

The last three ways cause copies of the file to be printed, whereas the first way prints the file directly. Thus, if the file is modified between the time the request is made and the time it is actually printed, then the changes will be reflected in the output.

2. Print two copies of file abc on printer xyz and title the output "my file":

```
pr abc ¦ lp -dxyz -n2 -t " my file "
```

3. Print file xxx on a printer called zoo in 12-pitch and write to the user's terminal when printing has completed:

```
lp -dzoo -o12 -w xxx
```

In this example, "12" is an option that is meaningful to this printer's interface program that prints output in 12-pitch mode (see lpadmin(1M)).

FINDING 1p STATUS: lpstat

The lpstat command is used to find status information about 1p requests, destinations, and the scheduler.

Examples:

1. List the status of all pending output requests made by this user:

```
lpstat
```

The status information for a request includes the request ID, the logname of the user, the total number of characters to be printed, and the date and time the request was made.

2. List the status of printers p1 and p2:

```
lpstat -pp1,p2
```

CANCELLING REQUESTS: cancel

The 1p requests can be cancelled using the cancel command. Two kinds of arguments can be given to the command: request ID's and printer names. The requests named by the request ID's are cancelled and requests that are currently printing on the named printers are cancelled. Both types of arguments can be intermixed.

Example:

Cancel the request that is now printing on printer xyz:

```
cancel xyz
```

If the user that is cancelling a request is not the same one that made the request, then mail is sent to the owner of the request. 1p allows any user to cancel requests in order to eliminate the need for users to find 1p administrators when unusual output should be purged from printers.

ALLOWING AND REFUSING REQUESTS: accept AND reject

When a new destination is created, lp will reject requests that are routed to it. When the lp admnistrator is sure that it is set up correctly, he or she should allow lp to accept requests for that destination. The accept command performs this function.

Sometimes it is necessary to prevent lp from routing requests to destinations. If printers have been removed or are waiting to be repaired or if too many requests are building for printers, then it may be desirable to cause lp to reject requests for those destinations. The reject command performs this function. After the condition that led to the rejection of requests has been remedied, the accept command should be used to allow requests to be taken again.

The acceptance status of destinations is reported by the -a option of lpstat.

Examples:

1. Cause lp to reject requests for destination xyz:

```
/usr/lib/reject -r" printer xyz needs repair " xyz
```

   Any users that try to route requests to xyz will encounter the following:

```
$ lp -dxyz file
lp: cannot accept requests for destination "xyz"
        --printer xyz needs repair
```

2. Allow lp to accept requests routed to destination xyz:

```
/usr/lib/accept xyz
```

ALLOWING AND INHIBITING PRINTING: enable AND disable

The enable command allows the lp scheduler to print requests on printers. That is, the scheduler routes requests only to the interface programs of enabled printers. Note that it is possible to enable a printer but to prevent further requests from being routed to it.

The disable command cancels the effects of the enable command. It prevents the scheduler from routing requests to printers, independently of whether or not lp is allowing them to accept requests. Printers can be disabled for several reasons, including malfunctioning hardware, paper jams, and end-of-day shutdowns. If a printer is busy at the time it is disabled, then the request that it was printing will be reprinted in its entirety either on another printer (if the request was originally routed to a class of printers) or on the same one when the printer is reenabled. The -c option causes the currently printing requests on busy printers to be cancelled in addition to disabling the printers. This is useful if strange output is causing a printer to behave abnormally.

Examples:

1. Disable printer xyz because of a paper jam:

```
$ disable -r" paper jam" xyz
printer "xyz" now disabled
```

2. Find the status of printer xyz:

```
$ lpstat -pxyz
printer "xyz" disabled since Jan 5 10:15
          --paper jam
```

3. Now, reenable xyz:

```
$ enable xyz
printer "xyz" now enabled
```

MOVING REQUESTS BETWEEN DESTINATIONS: lpmove

Occasionally, it is useful for lp administrators to move output requests between destinations. For instance, when a printer is down for repairs, it may be desirable to move all of its pending requests to a working printer. This is one way to use the lpmove command. The other use of this command is to move specific requests to a different destination. lpmove will refuse to move requests while the lp scheduler is running.

Examples:

1. Move all requests for printer abc to printer xyz:

```
$ /usr/lib/lpmove abc xyz
```

   All of the moved requests are renamed from abc-nnn to xyz-nnn. As a side effect, destination abc is no longer accepting further requests.

2. Move requests zoo-543 and abc-1200 to printer xyz:

```
$ /usr/lib/lpmove zoo-543 abc-1200 xyz
```

   The two requests are now renamed xyz-543 and xzy-1200

STOPPING AND STARTING THE SCHEDULER: lpshut AND lpsched

Lpsched is the program that routes the output requests that were made with lp through the appropriate printer interface programs to be printed on line printers. Each time the scheduler routes a request to an interface program, it records an entry in the log file, /usr/spool/lp/log. This entry contains the logname of the user that made the request, the request ID, the name of the printer that the request is being printed on, and the date and time that printing first started. In the case that a request has been restarted, more than one entry in the log file can refer to the request. The scheduler also records error messages in the log file. When lpsched is started, it renames /usr/spool/lp/log to /usr/spool/lp/oldlog and starts a new log file.

No printing will be performed by the lp system unless lpsched is running. Use the command

```
lpstat -r
```

to find the status of the lp scheduler.

Lpsched is normally started by the /etc/rc program as described above and continues to run until the operating system is shut down. The scheduler operates in the /usr/spool/lp directory. When it starts running, it will exit immediately if a file called SCHEDLOCK exists. Otherwise, it creates this file in order to prevent more than one scheduler from running at the same time.

Occasionally, it is necessary to shut down the scheduler in order to reconfigure lp or to rebuild the lp software. The command

```
/usr/lib/lpshut
```

causes lpsched to stop running and terminates all printing activity. All requests that were in the middle of printing will be reprinted in their entirety when the scheduler is restarted.

To restart the lp scheduler, use the command

```
/usr/lib/lpsched
```

Shortly after this command is entered, lpstat should report that the scheduler is running. If not, it is possible that a previous invocation of lpsched exited without removing SCHEDLOCK, so try the following:

```
rm -f /usr/spool/lp/SCHEDLOCK
/usr/lib/lpsched
```

The scheduler should be running now.

PRINTER INTERFACE PROGRAMS

Every lp printer must have an interface program that does the actual printing on the device that is currently associated with the printer. Interface programs can be shell procedures, C programs, or any other executable programs. The lp model interfaces are all written as shell procedures and can be found in the /usr/spool/lp/model directory. At the time lpsched routes an output request to a printer P, the interface program for P is invoked in the directory /usr/spool/lp as follows:

```
        interface/P id user title copies options file ...
```

where

_id_ is the request ID returned by lp

user is logname of the user who made the request

title is the optional title specified by the user

copies is the number of copies requested by the user

options is a blank-separated list of class or printer-dependent options specified by the user

file is the full pathname of a file to be printed

Examples:

The following examples are requests made by user "smith" with a system default destination of printer "xyz". Each example lists an lp command line followed by the corresponding command line generated for printer xyz's interface program:

```
1.  lp /etc/passwd /etc/group
    interface/xyz xyz-52 smith " " 1 " " /etc/passwd/etc/group

2.  pr /etc/passwd | lp -t" users " -n5
    interface/xyz xyz-53 smith users 5  " "
    /usr/spool/lp/request/xyz/d0-53

3.  lp /etc/passwd -oa -ob
    interface/xyz xyz-54 smith " " 1 " a b " /etc/passwd
```

When the interface program is invoked, its standard input comes from /dev/null and both the standard output and standard error output are directed to the printer's device. Devices are opened for reading as well as writing when file modes permit. In the case where a device is a regular file, all output is appended to the end of the file.

Given the command line arguments and the output directed to a device, interface programs can format their output in any way they choose. Interface programs must ensure that the proper stty modes (terminal characteristics such as baud rate, output options, and so on) are in effect on the output device. This can be done as follows in a shell interface only if the device is opened for reading:

```
    stty mode ... <&1
```

That is, take the standard input for the stty command from the device.

When printing has completed, it is the responsibility of the interface program to exit with a code indicative of the success of the print job. Exit codes are interpreted by lpsched as shown in Table D-1.

Table D-1. Exit Codes Interpreted by lpsched

Code	Meaning to lpsched
zero	The print job has completed successfully.
1 to 127	A problem was encountered in printing this particular request (for example, too many nonprintable characters). This problem will not affect future print jobs. Lpsched notifies users by mail that there was an error in printing the request.
greater than 127	These codes are reserved for internal use by lpsched. Interface programs must not exit with codes in this range.

When problems occur that are likely to affect future print jobs (such as a device filter program is missing), the interface programs would be wise to disable printers so that print requests are not lost. When a busy printer is disabled, the interface program will be terminated with signal 15.

SETTING UP HARDWIRED DEVICES AND LOGIN TERMINALS AS lp PRINTERS

Both hardwired devices and login terminals can be used as lp printers. The following two subsections describe how to set them up by use of examples.

Hardwired Devices

As an example of how to set up a hardwired device for use as an lp printer, consider using tty line 15 as printer xyz. As superuser, perform the following:

a. Avoid unwanted output from non-lp processes and ensure that lp can write to the device:

```
$ chown lp /dev/tty015
$ chmod 600 /dev/tty015
```

b. Change /etc/inittab so that tty015 is not a login terminal. In other words, ensure that /etc/getty is not trying to log users in at this terminal. Change the entries for line 15 to:

```
1:15:o:
2:15:o:
```

Enter the command:

```
$ init 2
```

If there is currently an invocation of /etc/getty running on tty015, kill it. Now, and when the operating system is rebooted, tty015 will be initialized with default stty modes. Thus, it is up to lp interface programs to establish the proper baud rate and other stty modes for correct printing to occur.

c. Introduce printer xyz to lp using the model prx interface program:

```
$ /usr/lib/lpadmin -pxyz -v/dev/tty015 -mprx
```

d. When xyz is created, it will initially be disabled and lp will be lp to accept requests for xyz:

```
/usr/lib/accept xyz
```

This will allow requests to build up for xyz and to be printed when it is enabled at a later time.

e. When it is desired for printing to occur, be sure that the printer is ready to receive output. For several printers, this means that the top of form has been adjusted and that the printer is on line. Enable printing to occur on xyz:

```
enable xyz
```

When requests have been routed to xyz, they will begin printing.

Login Terminals

Login terminals can also be used as lp printers. To do this for a terminal called abc, perform the following:

a. Introduce printer abc to lp using the model 1640 interface program:

```
$ /usr/lib/lpadmin -pabc -v/dev/null -m1640 -1
```

Note that /dev/null is used as abc's device because we will specify the actual device each time that abc is enabled. This device may be different from day to day. When abc is created, it will initially be disabled and lp will be rejecting requests routed to it. If it is desired, allow lp toaccept requests for abc:

```
/usr/lib/accept abc
```

This will allow requests to build up for abc and to be printed when it is enabled at a later time. It is not advisable to enable abc for printing, however, until the following steps have been taken.

b. Log terminal in if this has not already been done.

c. Assuming the tty(1) command reports that this terminal is /dev/tty002, associate this device with printer abc:

```
$ /usr/lib/lpadmin -pabc -v/dev/tty002
```

Note that lpadmin may be used only by an LPA. If it is desired for other users to routinely perform this step, then an LPA can establish a program owned by lp or by root with set-user-ID permission that performs this function.

d. When it is desired for printing to occur, be sure that the printer is ready to receive output. For several printers, this means that the top of form has been adjusted. Enable printing to occur on abc:

```
enable abc
```

When requests have been routed to abc, they will begin printing.

e. When all printing has stopped on abc or when you want it back as a regular login terminal, you can prevent it from printing more output:

```
$ disable abc
printer " abc " now disabled
```

If abc is enabled when the operating system is rebooted or when lpsched is restarted, it will be disabled automatically.

SUMMARY

The administrative functions of the lp administrator have been described in detail. These functions include configuring and reconfiguring lp; maintaining printer interface programs; accepting, rejecting, and moving print requests; stopping and starting the lp scheduler; and enabling and disabling printers. lp offers administrators the following advantages over other centrally supported printer packages:

- Printers can be grouped into classes.
- lp can be configured to meet the needs of each site.
- Administrators can supply interface programs to format output in any way desirable.
- lp functions are performed by simple commands and not by hand.

Appendix E
System Activity Package

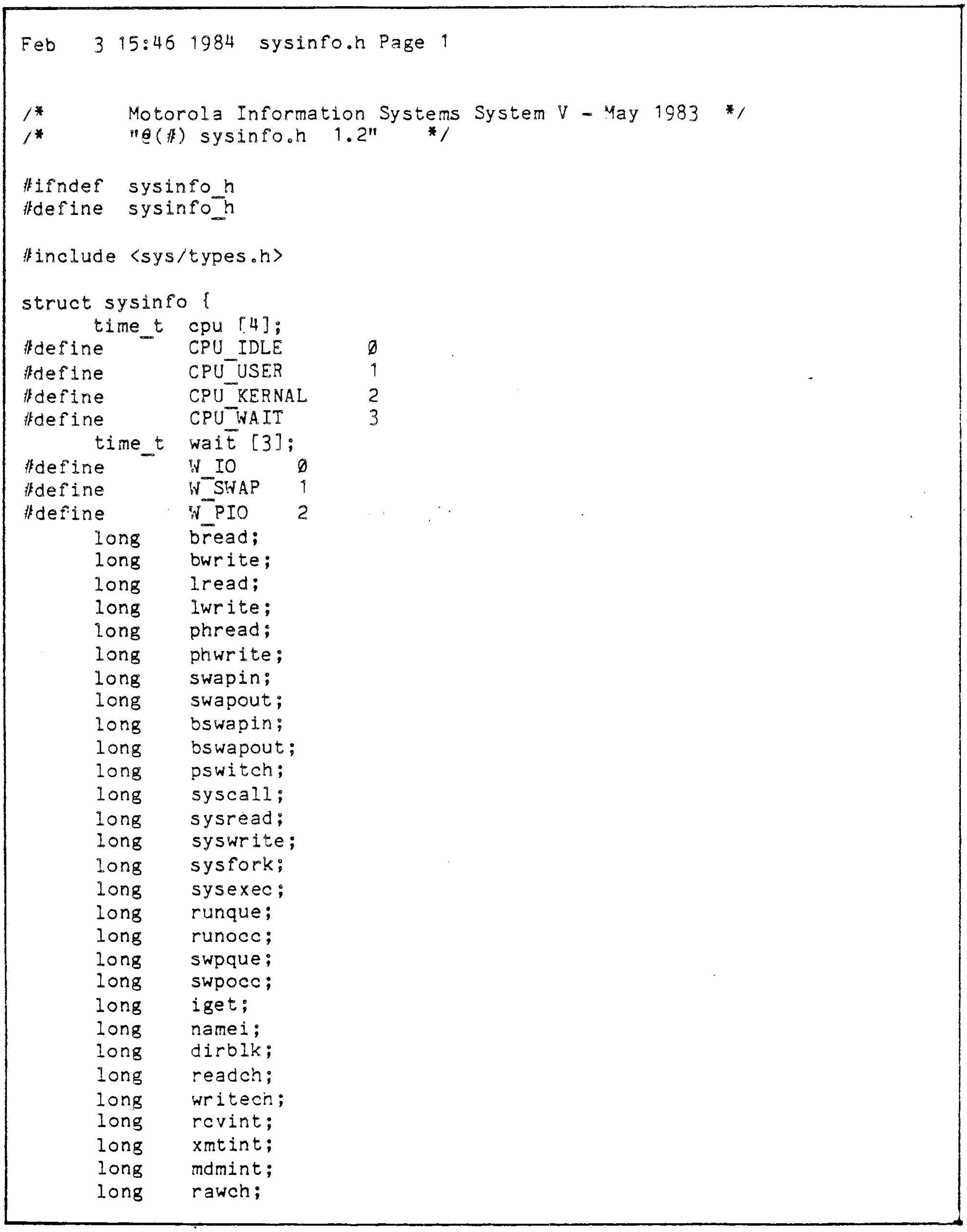
This section describes the design and implementation of the system activity package for this operating system. This operating system contains a number of counters that are incremented as various system actions occur. The system activity package reports operating-system-wide measurements, including processor utilization, disk and tape input/output (I/O) activities, terminal device activity, buffer usage, system calls, system switching and swapping, file-access activity, queue activity, and message and semaphore activities. The package provides four commands that generate various types of reports. Procedures that automatially generate daily reports are also included. The five functions of the activity package are:

- sar(1) command--allows a user to generate system activity reports in real-time and to save system activities in a file for later usage.
- sag(1G) command--displays system activity in a graphical form.
- sadp(1) command--samples disk activity once every second during a specified time interval and reports disk usage and seek distance in either tabular or histogram form.
- timex(1)--a modified time(1) command that times a command and also reports concurrent system activity.
- system activity daily reports--procedures are provided for sampling and saving system activities in a data file periodically and for generating the daily report from the data file.

The following subsections describe the system activity counters located in the operating system kernel: the commands in the system activity package, the procedure for generating daily reports, source file descriptions, and an explanation of some statistics.

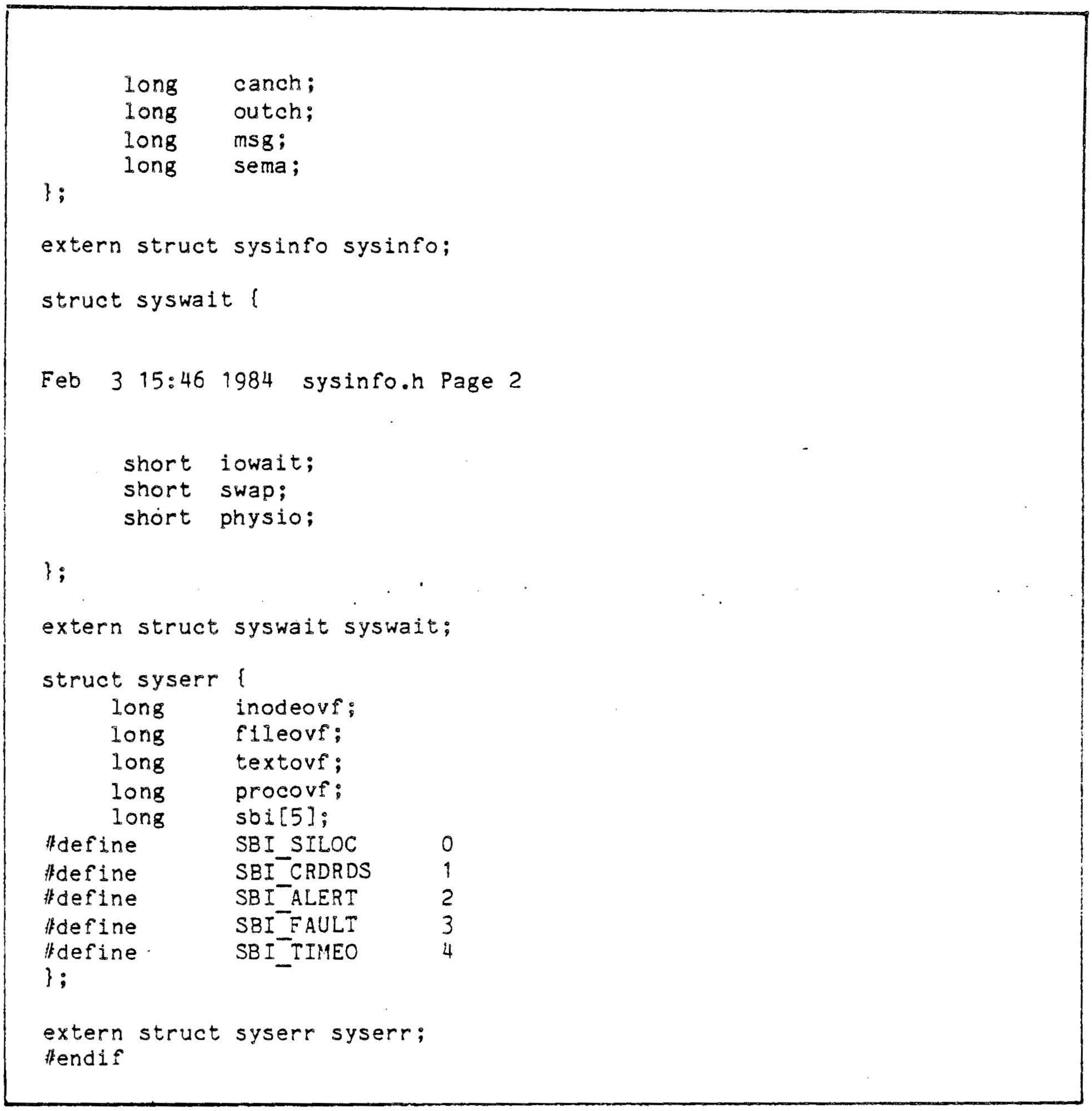
SYSTEM ACTIVITY COUNTERS

The system activity counters provide the basis for the system activity reporting system. Most of these counters are described by the sysinfo data structure in /usr/include/sys/sysinfo.h, as shown in Figure E-1. The system table overflow counters are in the _syserr structure. The device activity counters are extracted from the device status tables. The I/O activity of all disk devices is recorded.



```
Feb   3 15:46 1984  sysinfo.h Page 1


/*       Motorola Information Systems System V - May 1983  */
/*       "@(#) sysinfo.h  1.2"    */

#ifndef  sysinfo_h
#define  sysinfo_h

#include <sys/types.h>

struct sysinfo {
        time_t  cpu [4];
#define         CPU_IDLE        0
#define         CPU_USER        1
#define         CPU_KERNAL      2
#define         CPU_WAIT        3
        time_t  wait [3];
#define         W_IO      0
#define         W_SWAP    1
#define         W_PIO     2
        long    bread;
        long    bwrite;
        long    lread;
        long    lwrite;
        long    phread;
        long    phwrite;
        long    swapin;
        long    swapout;
        long    bswapin;
        long    bswapout;
        long    pswitch;
        long    syscall;
        long    sysread;
        long    syswrite;
        long    sysfork;
        long    sysexec;
        long    runque;
        long    runocc;
        long    swpque;
        long    swpocc;
        long    iget;
        long    namei;
        long    dirblk;
        long    readch;
        long    writech;
        long    rcvint;
        long    xmtint;
        long    mdmint;
        long    rawch;
```

Figure E-1. The Sysinfo Structure (Page 1 of 2)



```
        long    canch;
        long    outch;
        long    msg;
        long    sema;
};

extern struct sysinfo sysinfo;

struct syswait {


Feb  3 15:46 1984  sysinfo.h Page 2


        short  iowait;
        short  swap;
        short  physio;

};

extern struct syswait syswait;

struct syserr {
       long     inodeovf;
       long     fileovf;
       long     textovf;
       long     procovf;
       long     sbi[5];
#define         SBI_SILOC       0
#define         SBI_CRDRDS      1
#define         SBI_ALERT       2
#define         SBI_FAULT       3
#define         SBI_TIMEO       4
};

extern struct syserr syserr;
#endif
```

Figure E-1. The Sysinfo Structure (Page 2 of 2)

In the following paragraphs, the system activity counters that are sampled by the system activity package are described.

CPU time counters There are four time counters that can be incremented at each clock interrupt 60 times per second. Exactly one of the cpu[] counters is incremented on each interrupt, according to the mode the processor is in at the interrupt: idle, user, kernel, and wait for I/O completion.

Lread and lwrite — The lread and lwrite counters are used to count logical read and write requests issued by the system to block devices.

Bread and bwrite — The bread and bwrite counters are used to count the number of times data is transferred between the system buffers and the block devices. These actual I/O's are triggered by logical I/O's that cannot be satisfied by the current contents of the buffers. The ratio of block I/O to logical I/O is a common measurement of the effectiveness of the system buffering.

Phread and phwrite — The phread and phwrite counters count read and write requests issued by the system to raw devices.

Swapin and Swapout — The swapin and swapout counters are incremented by each system request initiating transfer from or to the swap device, including virtual memory page transfers. Frequently used programs are kept on the swap device and swapped in rather then loaded from the file system. The swapin counter reflects these initial loading operations as well as resumptions of activity, while the swapout counter reveals the level of actual "swapping." The amount of data transferred between the swap device and memory is measured in blocks and counted by bswapin and bswapout.

Pswitch and Syscall — These counters are related to the management of multiprogramming. Syscall is incremented every time a system call is invoked. The numbers of invocations of read(2), write(2), fork(2), and exec(2) system calls are kept in counters sysread, syswrite, sysfork, and sysexec, respectively. Pswitch counts the times the switcher was invoked, which occurs when:

- a system call resulted in a road block
- an interrupt occurred resulting in awakening a higher priority process
- 1-second clock interrupt

Iget, namei,
and dirblk

These counters apply to file-access operations. Iget and namei, in particular, are the names of operating system routines. The counters record the number of times that the respective routines are called. Namei is the routine that performs file system path searches. It searches the various directory files to get the associated i-number of a file corresponding to a special path. Iget is a routine called to locate the inode entry of a file (i-number). It first searches the in-core inode table. If the inode entry is not in the table, routine iget will get the inode from the file system where the file resides and make an in the in-core inode table for the file. Iget returns a pointer to this entry. Namei calls iget, but other file access routines also call iget. Therefore, counter iget is always greater than counter namei.

Counter dirblk records the number of directory block reads issued by the system. It is noted that the directory blocks read divided by the number of namei calls estimates the average path length of files.

Runque, runocc,
swpque and swpocc

These counters are used to record queue activities. They are implemented in the clock.c routine. At every one-second interval, the clock routine examines the process table to see whether any processes are in core and in ready state. If so, the counter runocc is incremented and the number of such processes are added to counter runque. While examining the process table, the clock routine also checks whether any processes in the swap device are in ready state. The counter swpocc is incremented if the swap queue is occupied, and the number of processes in swap queue is added to counter swpque.

Readch and writech

The readch and writech counters record the total number of bytes (characters) transferred by the read and write system calls, respectively.

Monitoring
terminal device
activities

There are six counters monitoring terminal device activities. Rcvint, xmtint, and mdmint are counters measuring hardware interrupt occurrences for receiver, transmitter, and modem individually. Rawch, canch, and outch count number of characters in the raw queue, canonical queue, and output queue. Characters generated by devices operating in the cooked mode, such as terminals, are counted in both rawch and (as edited) in canch, but characters from raw devices, such as communication processors, are counted only in rawch.

Msg and sema
counters

These counters record message sending and receiving activities and semaphore operations, respectively.

Monitoring I/O activities

As to the I/O activity for a disk or tape device, four counters are kept for each disk or tape drive in the device status table. Counter io_ops is incremented when an I/O operation has occurred on the device. It includes block I/O, swap I/O, and physical I/O. Io_bcnt counts the amount of data transferred between the device and memory in 512 byte units. Io_act and io_resp measure the active time and response time of a device in time ticks summed over all I/O requests that have completed for each device. The device active time includes the device seeking, rotating, and data transferring times, while the response time of an I/O operation is from the time the I/O request is queued to the device to the time when the I/O completes.

Inodeovf, fileovf, textovf, procovf

These counters are extracted from_syserr structure. When an overflow occurs in any of the inode, file, text, and process tables, the corresponding overflow counter is incremented.

SYSTEM ACTIVITY COMMANDS

The system activity package provides three commands for generating various system activity reports and one command for profiling disk activities. These tools facilitate observation of system activity during:

- a controlled stand-alone test of a large system
- an uncontrolled run of a program to observe the operating environment
- normal production operation

Commands sar and sag permit the user to specify a sampling interval and number of intervals for examining system activity and then to display the observed level of activity in tabular or graphical form. The timex command reports the amount of system activity that occurred during the precise period of execution of a timed command. The sadp command allows the user to establish a sampling to establish a sampling period during which access location and seek distance on specified disks are recorded and later displayed as a tabular summary or as a histogram.

The sar Command

The sar command can be used in the following ways:

- When the frequency arguments t and n are specified, it invokes the data collection program sadc to sample the system activity counters in the operating system every t seconds for n intervals and generates system activity reports in real-time. Generally, it is desirable to include the option to save the sampled data in a file for later examination. In addition to the system counters, a time stamp is also included, which gives the time the sample was taken.

- If no frequency arguments are supplied, it generates system activity reports for a specified time interval from an existing data file that was created by sar at an earlier time.

A convenient usage is to run sar as a background process, saving its samples in a temporary file but sending its standard output to /dev/null. Then an experiment is conducted after which the system activity is extracted from the temporary file. The sar(1) manual entry describes the usage and lists various types of reports. The subsection on "Derivation of Basic Statistics" gives a formula for deriving each reported item.

The sag Command

sag displays system activity data graphically. It relies on the data file produced by a prior run of sar after which any column of data or the combination of columns of data of the sar report can be plotted. A fairly simple but powerful command syntax allows the specification of cross plots or time plots. Data items are selected using the sar column header names. The sar(1G) manual entry describes its options and usage. The system activity graphical program invokes graphics(1G) and tplot(1G) commands to have the graphical output displayed on any of the terminal types supported by tplot.

The timex Command

The timex command is an extension of the time(1) command. Without options, timex behaves exactly like time. In addition to giving the time information, it also prints a system activity report derived from the system counters. The manual entry timex(1) explains its usage. It should be emphasized that the user and sys times reported in the second and third lines are for the measured process itself, including all its children, while the remaining data (including the cpu user % and cpu sys %) are for the entire system.

While the normal use of timex will probably be to measure a single command, multiple commands can also be timed. To measure multiple commands, combine the commands in an executable file and time the file, or more concisely, type:

```
    timex sh -c "cmd1; cmd2; ...;"
```

This establishes the necessary parent-child relationships to correctly extract the user and system times consumed by cmd1, cmd2, and so on, (and the shell).

The sadp Command

sadp is a user level program that can be invoked independently by any user. It requires no storage or extra code in the operating system and allows the user to specify the disks to be monitored. The program is reawakened every second, reads system tables from /dev/kmem, and extracts the required information. Because of the one second sampling, only a small fraction of disk requests are observed; however, comparative studies have shown that the statistical determination of disk locality is adequate when a sufficient number of samples is collected.

In the operating system, there is an iobuf for each disk drive. It contains two pointers which are head and tail of the I/O active queue for the device. The actual requests in the queue can be found in three buffer header pools--system buffer headers for block I/O requests, physical buffer headers for physical I/O requests, and swap buffer headers for swap I/O. Each buffer header has a forward pointer that points to the next request in the I/O active queue and a backward pointer that points to the previous request.

Sadp snapshots the iobuf of the monitored device and the three buffer header pools once every second during the monitoring period. It then traces the requests in the I/O queue, records the disk access location, and seeks distance in buckets of eight cylinder increments. At the end of monitoring period, it prints out the sampled data. The output of sadp can be used to balance load among disk drives and to rearrange the layout of a particular disk pack. The usage of this command is described in manual entry sadp(1).

DAILY REPORT GENERATION

The previous part described the commands available to users to initiate activity observations. It is probably desirable for each installation to routinely monitor and record system activity in a standard way for historical analysis. This part describes the steps that a system administrator may follow to automatically produce a standard daily report of system activity.

Facilities

- sadc--The executable module of sadc.c (see the subsection on "Source Files") that reads system counters from /dev/kmem and records them to a file. In addition to the file argument, two frequency arguments are usually specified to indicate the sampling interval and number of samples to be taken. In case no frequency arguments are given, it writes a dummy record in the file to indicate a system restart.

- sa1--The shell procedure that invokes sadc to write system counters in the daily data file /usr/adm/sa dd where dd represents the day of the month. It can be invoked with sampling interval and iterations as arguments.

- sa2--The shell procedure that invokes the sar command to generate thedaily report /usr/adm/sa/sar dd from the daily data file /usr/adm/sa/sa dd. It also removes daily data files and report files after seven days. The starting and ending times and all report options of sar are applicable to sa2.

Suggested Operational Setup

It is suggested that the cron(1M) control the normal data collection and report generation operations. For example, the sample entries in /usr/lib/crontab

```
0 * * * 0,6 su sys -c " /usr/lib/sa/sa1 "
0 18- * * 1-5 su sys -c " /usr/lib/sa/sa1 "
0 8-17 * * 1-5 su sys -c " /usr/lib/sa/sa1 1200 3 "
```

would cause the data collection program sadc to be invoked every hour on the hour. Moreover, depending on the arguments presented, it writes data to the data file one to three times every 20 minutes. Therefore, under the control of cron(1M), the data file is written every 20 minutes between 8:00 and 18:00 on weekdays and hourly at other times.

Note that data samples are taken more frequently during prime time on weekdays to make them available for a finer and more detailed graphical display. It is suggested that sa1 be invoked hourly rather than invoking it once every day; this ensures that if the system crashes, data collection will be resumed within an hour of restarting the system.

Because system activity counters restart from zero when the system is restarted, a special record is written on the data file to reflect this situation. This process is accomplished by invoking sadc with no frequency arguments within /etc/rc when going to multiuser state:

```
su adm -c " /usr/lib/sa/sadc /usr/adm/sa/sa'date + %d'"
```

Cron(1M) also controls the invocation of sar to generate the daily report via shell procedure sa2. You can choose the time period the daily report is to cover and the groups of system activity to be reported. For instance, if

```
    0 20 * * 1-5 su sys -c " /usr/lib/sa/sa2 -s 8:00 -e 18:00 -i 3600 -uybd"
```

is an entry in /usr/lib/crontab, cron will execute the sar command to gemerate daily reports from the daily data file at 20:00 on weekdays. The daily report reports the processor utilization, terminal device activity, buffer usage, and device activity every hour from 8:00 to 18:00.

In case of a shortage of the disk space or for any other reason, these data files and report files can be removed by the superuser. The manual entry on sar describes the daily report generation procedure.

SOURCE FILES

When source code is provided, the following source file and shell programs are in the directory /usr/src/cmd/sa.

sa.h — The system activity header file defines the structure of data and device information for measured devices. It is included in sadc.c, sar.c, and timex.c.

sadc.c — The data collection program that accesses /dev/kmem to read the read the system activity counters and writes data either on standard output or on a binary data file. It is invoked by the sar command generating a real-time report. It is also invoked indirectly by entries in /usr/lib/crontab to collect system activity data.

sar.c — The report generation program invokes sadc to examine system activity data, generates reports in real-time, and saves the data to a file for later usage. It may also generate system activity reports from an existing data file. It is invoked indirectly by cron to generate daily reports.

saghdr.h — The header file for saga.c and sagb.c. It contains data structures and variables used by saga.c and sagb.c.

saga.c
sagb.c — The graph generation program that first invokes sar to format the data of a data file in a tabular form and then displays the sar data in graphical form.

sa1.sh — The shell procedure that invokes sadc to write data file records. It is activated by entries in /usr/lib/crontab.

sa2.sh — The shell procedure that invokes sar to generate the report. It also removes the daily data files and daily report files after a week. It is activated by an entry in /usr/lib/crontab on weekdays.

timex.c — The program that times a command and generates a system activity report.

sadp.c — The program that samples and reports disk activities.

DERIVATION OF BASIC STATISTICS

Here is how the basic system activity statistics are derived. Each item discussed below is the data difference sampled at two distinct times, t2 and t1.

Processor Utilization

```
    %-of-cpu-x=cpu-x / (cpu-idle + cpu-user + cpu-kernel + cpu-wait) * 100
```

where

cpu-x is cpu-idle, cpu-user, cpu-kernel (cpu-sys), or cpu-wait.

Cached Hit Ratio

```
    %-of-cached-I/O = (logical-I/O -- block-I/O) / logical-I/O * 100
```

where

cached I/O is cached read or cached write.

Disk or Tape I/O Activity

```
    %-of-busy = I/O-active / (t2 - t1) * 100;
    avg-queue-length = I/O-resp / I/O-active;
    avg-wait = (I/O-resp -- I/O-active) / I/O-ops;
    avg-service-time = I/O-active / I/O-ops.
```

Queue Activity

```
    avg-x-queue-length = x-queue / x-queue-occupied-time;
    %-of-x-queue-occupied-time = x-queue-occupied-time / (t2 - t1);
```

where

x-queue is run queue or swap queue.

The Rest of System Activity

avg-rate-of-x = x / (t2 - t1)

where

x is swap in/out, blocks swapped in/out, terminal device activities, read/write characters, block read/write, logical read/write, process switch, system calls, read/write, fork/exec, iget, namei, directory blocks read, disk/tape I/O activities, message or semaphore activities.

Appendix F
TM30 Keyboard Translation Table

Table F-1 shows the ASCII character set for the Motorola TM30 Terminal. The characters in the left column are hexadecimal digits from 0 (zero) to F. The middle column shows the keystrokes that generate the ASCII code. The column on the right gives the characters, if any, that appear on screen when you press the key(s) shown in the middle column.

Table F-1. TM30 Keyboard Translation Table

Hexadecimal Code	TM30 Keyboard Keystrokes	Displayable Characters
00	Ctrl =	No
01	Ctrl A	No
02	Ctrl B	No
03	Ctrl C	No
04	Ctrl D	No
05	Ctrl E	No
06	Ctrl F	No
07	Ctrl G	No
08	<-- (left arrow, bcksp)	No
	or Ctrl H	No
09	Tab or Ctrl I	No
0A	Ctrl Accept or Ctrl J	No
0B	Ctrl K	No
0C	Ctrl L	No
0D	Return or Ctrl M	No
0E	Ctrl N	No
0F	Ctrl O	No
10	Ctrl P	No
11	Ctrl Q	No
12	Ctrl R	No
13	Ctrl S	No
14	Ctrl T	No
15	Ctrl U	No
16	Ctrl V	No
17	Ctrl W	No
18	Ctrl X	No
19	Ctrl Y	No
1A	Ctrl Z	No
1B	(See "Hex Code/Escape Sequences," below)	
1C	Ctrl , (comma)	No
1D	Ctrl {	No
1E	Ctrl . (period)	No
1F	Ctrl [	No
20	Space bar	No

Table F-1. TM30 Keyboard Translation Table (Cont.)

Hexadecimal Code	TM30 Keyboard Keystrokes	Displayable Characters
21	!	!
22	"	"
23	#	#
24	$	$
25	%	%
26	&	&
27	'	'
28	(	(
29	)	)
2A	*	*
2B	+	+
2C	,	,
2D	- (dash)	-
2E	. (period)	.
2F	/	/
30	0	0
31	1	1
32	2	2
33	3	3
34	4	4
35	5	5
36	6	6
37	7	7
38	8	8
39	9	9
3A	:	:
3B	;	;
3C	<	<
3D	=	=
3E	>	>
3F	?	?
40	@	@
41	Shift A	A
42	Shift B	B
43	Shift C	C
44	Shift D	D
45	Shift E	E
46	Shift F	F
47	Shift G	G
48	Shift H	H
49	Shift I	I
4A	Shift J	J
4B	Shift K	K
4C	Shift L	L
4D	Shift M	M
4E	Shift N	N

Table F-1. TM30 Keyboard Translation Table (Cont.)

Hexadecimal Code	TM30 Keyboard Keystrokes	Displayable Characters
4F	Shift O	O
50	Shift P	P
51	Shift Q	Q
52	Shift R	R
53	Shift S	S
54	Shift T	T
55	Shift U	U
56	Shift V	V
57	Shift W	W
58	Shift X	X
59	Shift Y	Y
5A	Shift Z	Z
5B	[	[
5C	Ctrl /	\
5D	]	]
5E		
5F	_ (underscore)	
60	Ctrl '	⊤
61	A	a
62	B	b
63	C	c
64	D	d
65	E	e
66	F	f
67	G	g
68	H	h
69	I	i
6A	J	j
6B	K	k
6C	L	l
6D	M	m
6E	N	n
6F	O	o
70	P	p
71	Q	q
72	R	r
73	S	s
74	T	t
75	U	u
76	V	v
77	W	w
78	X	x
79	Y	y
7A	Z	z
7B	{	{
7C	¦	¦

Table F-1. TM30 Keyboard Translation Table (Cont.)

Hex Code/ Escape Sequence	TM30 Keyboard Keystrokes	Displayable Characters
7D	}	}
7E		
7F	Ctrl -	No
ESC	Alt	No
ESC O A		
ESC O B	Ctrl 1	No
ESC O C	Ctrl 2	No
ESC O D	Ctrl 3	No
ESC O E	Ctrl 4	No
ESC O F	Ctrl 5	No
ESC O G	Ctrl 6	No
ESC O H	Ctrl 7	No
ESC O I	Ctrl 8	No
ESC O J	Ctrl 9	No
ESC O K	Ctrl 0	No
ESC O L	Ctrl]	No
ESC O M	Ctrl ;	No
ESC O N	Shift Return	No
ESC O O	Ctrl Return	No
ESC [b	1/2	No
ESC [g	Exit	No
ESC [h	Erase	No
ESC [i	1 (Numeric Island)	No
ESC [j	2 (Numeric Island)	No
ESC [k	3 (Numeric Island)	No
ESC [l	Accept	No
ESC [n	Copy	No
ESC [o	0 (Numeric Island)	No
ESC [r	Char Attr	No
ESC [s	Undo	No
ESC [t	Home	No
ESC [u	Up Arrow	No
ESC [v	Help	No
ESC [w	Windo	No
ESC [x	Print	No
ESC [y	F1	No
ESC [z	F3	No
ESC [I	F2	No
ESC [O	F8	No
ESC [Q	Reset	No
ESC [U	Slct	No
ESC [V	Move	No
ESC [W	7 (Numeric Island)	No
ESC [X	8 (Numeric Island)	No
ESC [Y	9 (Numeric Island)	No
ESC [	Down Arrow	No

Table F-1. TM30 Keyboard Translation Table (Cont.)

Hex Code/ Escape Sequence	TM30 Keyboard Keystrokes	Displayable Characters
ESC [~	F7	No
ESC [#	5 (Numeric Island)	No
ESC [$	6 (Numeric Island)	No
ESC [%	- (Numeric Island)	No
ESC [¦	F5	No
ESC [&	Ins	No
ESC [-	--> (Right Arrow)	No
ESC [{	F4	No
ESC [}	F6	No
ESC [[	F9	No
ESC [ ]	Delete	No
ESC ['	. (Numeric Island)	No
ESC ["	4 (Numeric Island)	No
ESC [\	Cmd	No
ESC [2 b	Shift 1/2	No
ESC [2 g	Shift Exit	No
ESC [2 h	Shift Erase	No
ESC [2 i	Shift 1 (Numeric Island)	No
ESC [2 j	Shift 2 (Numeric Island)	No
ESC [2 k	Shift 3 (Numeric Island)	No
ESC [2 l	Shift Accept	No
ESC [2 n	Shift Copy	No
ESC [2 o	Shift 0 (Numeric Island)	No
ESC [2 r	Shift Char Attr	No
ESC [2 s	Shift Undo	No
ESC [2 t	Shift Home	No
ESC [2 u	Shift Up Arrow	No
ESC [2 v	Shift Help	No
ESC [2 w	Shift Windo	No
ESC [2 x	Shift Print	No
ESC [2 y	Shift F1	No
ESC [2 z	Shift F3	No
ESC [2 I	Shift F2	No
ESC [2 O	Shift F8	No
ESC [2 Q	Shift Reset	No
ESC [2 R	Shift <-- (Lft Arrw, Bsp)	No
ESC [2 U	Shift Slct	No
ESC [2 V	Shift Move	No
ESC [2 W	Shift 7 (Numeric Island)	No
ESC [2 X	Shift 8 (Numeric Island)	No
ESC [2 Y	Shift 9 (Numeric Island)	No
ESC [2 Z	Shift Alt	No
ESC [2	Shift Down Arrow	No
ESC [2 ~	Shift F7	No
ESC [2 #	Shift 5 (Numeric Island)	No
ESC [2 $	Shift 6 (Numeric Island)	No

Table F-1. TM30 Keyboard Translation Table (Cont.)

Hex Code/ Escape Sequence	TM30 Keyboard Keystrokes	Displayable Characters
ESC [2 %	Shift - (Numeric Island)	No
ESC [2 ¦	Shift F5	No
ESC [2 &	Shift Ins	No
ESC [2 -	Shift --> (Right Arrow)	No
ESC [2 {	Shift F4	No
ESC [2 }	Shift F6	No
ESC [2 [	Shift F9	No
ESC [2]	Shift Delete	No
ESC [2 '	Shift . (Numeric Island)	No
ESC [2 "	Shift 4 (Numeric Island)	No
ESC [2 \	Shift Cmd	No
ESC [4 b	Ctrl 1/2	No
ESC [4 g	Ctrl Exit	No
ESC [4 h	Ctrl Erase	No
ESC [4 i	Ctrl 1 (Numeric Island)	No
ESC [4 j	Ctrl 2 (Numeric Island)	No
ESC [4 k	Ctrl 3 (Numeric Island)	No
ESC [4 n	Ctrl Copy	No
ESC [4 o	Ctrl 0 (Numeric Island)	No
ESC [4 r	Ctrl Char Attr	No
ESC [4 s	Ctrl Undo	No
ESC [4 t	Ctrl Home	No
ESC [4 u	Ctrl Up Arrow	No
ESC [4 v	Ctrl Help	No
ESC [4 w	Ctrl Windo	No
ESC [4 x	Ctrl Print	No
ESC [4 y	Ctrl F1	No
ESC [4 z	Ctrl F3	No
ESC [4 I	Ctrl F2	No
ESC [4 O	Ctrl F8	No
ESC [4 R	Ctrl <-- (Lft Arrw, Bsp)	No
ESC [4 U	Ctrl Slct	No
ESC [4 V	Ctrl Move	No
ESC [4 W	Ctrl 7 (Numeric Island)	No
ESC [4 X	Ctrl 8 (Numeric Island)	No
ESC [4 Y	Ctrl 9 (Numeric Island)	No
ESC [4 Z	Ctrl Alt	No
ESC [4	Ctrl Down Arrow	No
ESC [4 ~	Ctrl F7	No
ESC [4 #	Ctrl 5 (Numeric Island)	No
ESC [4 $	Ctrl 6 (Numeric Island)	No
ESC [4 %	Ctrl - (Numeric Island)	No
ESC [4 ¦	Ctrl F5	No
ESC [4 &	Ctrl Ins	No
ESC [4 -	Ctrl --> (Right Arrow)	No
ESC [4 {	Ctrl F4	No

Table F-1. TM30 Keyboard Translation Table (Cont.)

Hex Code/ Escape Sequence	TM30 Keyboard Keystrokes	Displayable Characters
ESC [4 }	Ctrl F6	No
ESC [4 [	Ctrl F9	No
ESC [4]	Ctrl Delete	No
ESC [4 '	Ctrl . (Numeric Island)	No
ESC [4 "	Ctrl 4 (Numeric Island)	No
ESC [4 \	Ctrl Cmd	No

Appendix G
UUCP, Copying Files Between Systems

WHAT IS UUCP?

uucp is a network system that runs under the operating system. uucp provides for copying files between computer systems and for execution of shell commands on remote computer systems.

uucp is a batch network. When a uucp job is entered, it is spooled for later transmission. Actual execution may happen right away, at some standard time of day, or when a remote system happens to call the local system. Users must wait for notification (usually through mail) that the job is actually executed.

Hardware requirements for uucp communication are minimal. Two systems can communicate via directly wired RS-232 ports or via telephone modems. Other communication media are not supported but are not necessarily incompatible with the uucp system; see the following section on uucp configuration.

A uucp network requires no overall administraton. A new system is added to the network by establishing communication between the new system and a system already on the network. Two uucp networks are merged by providing communication between them. Each individual computer system provides for its own security by restricting remote use of the system. See the following subsection on uucp use.

USING UUCP

The uucp commands in this section allow users to copy files between computer systems and to execute commands on remote systems. The section discusses six areas.

- The uucp public directory,
- The uucp convention for file, command, and user names,
- Copying files between systems,
- Remote execution of commands,
- Limitations of uucp commands,
- Querying and controlling uucp job annd network statuses.

The Uucp Public Directory

The uucp public directory, /usr/spool/uucppublic, is a standard destination for uucp copies and output files. In some cases, it is the only legal destination directory. uucp commands accept the tilde-slash (~/) sequence as an abbreviation for /usr/spool/uucppublic/.

Some uucp messages refer to the public directory as PUBDIR.

Move files from the public directory to a private directory as soon as possible. uucp purges the public directory of old files every day.

Conventions for File, Command, and User Names

A simple convention designates the computer system on which a file, command, or user is located. Using the convention requires that you know the node names of systems used to route your uucp jobs. Each system has a node name. To find the node name of an operating system, login to the system and type

```
    uname -n
```

To specify a specific system, use a route of systems used to reach that system from your system. Elements of the route are separated by exclamation points (!).

For example, consider a simple circular network with four operating systems (Figure G-1). System alpha can communicate directly with system beta and system gamma; system beta with system alpha and system homebase; system homebase with system beta and system gamma; and system gamma with system homebase and system alpha. A user on homebase specifies alpha as beta!alpha or gamma!alpha, specifies beta as beta, and specifies gamma as gamma. A user on alpha specifies gamma as gamma or beta!homebase!gamma.

Normally, the shortest route is the best way to specify a remote system, but this may depend on the hardware configuration of your network.

To specify a file, command, or user on a particular system, precede the file, command, or user name with the system node name and an exclamation point. Thus a user on homebase who wants /etc/passwd on alpha specifies beta!alpha!/etc/passwd.

A null system specification (for example, !/etc/passwd) or a missing system specification (/etc/passwd) specifies the local system.

File names given to uucp are interpreted using file name conventions that extend those of the shell. The metacharacters *, ?, [, and] expand on the appropriate system as they do for the shell. Individual file names are interpreted by one of the following conventions:

- o If the file name begins with a slash (/), uucp assumes that the name is a full path name and uses it as is.

- If the file name begins with a tilde followed by a slash (~/), the first two characters are a reference to the uucp public directory /usr/spool/uucppublic. Thus beta!~/jon/docs means /usr/spool/uucppulbic/jon/docs on the system beta. The public directory is used when it is not possible to copy to a user directory; see the subsection below on uucp limitations.

- If the file name begins with a tilde (~) followed by a user name, the initial sequence references the specified user's home directory. For example, if jon's home directory is /a/jon, then ~jon/src/s.c is the same as /a/jon/src/s.c.

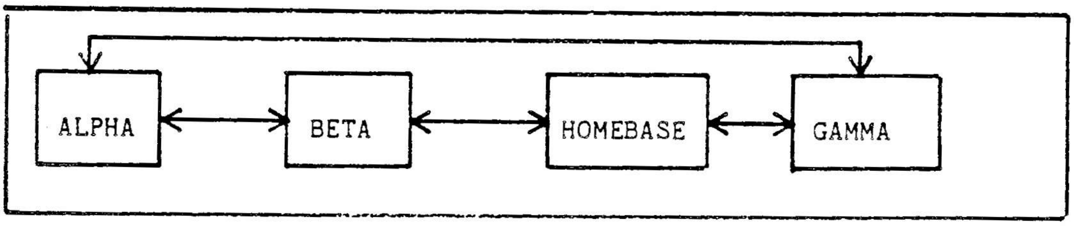


Figure G-1. Uucp Configuration Example

- If the file name does not begin with one of the three sequences given above, uucp assumes that the name is a partial path name and adds the working directory name (as printed by pwd), in front. Note that this convention includes files on remote systems, even though the working directory is a directory on the local system.

Copying Files Between Systems

There are two ways to copy files between systems.

- The uucp command is similar to the cp command (see cp(1) in the Series 6000 Operating System Reference Manual). A copy requires a single uucp command issued anywhere in the uucp network. The uucp command can only access files accessible to the user uucp.

- The uuto command sends files to a particular user. A user on the source system, who has access to the files, originates the copy, specifying a remote system and user. The specified user completes the copy.

Actual data transmission does not necessarily occur right away. Each copy must wait for the next establishment of a communication link (normally within 24 hours) or for other uucp jobs to finish with the link. There are two ways to prevent a file from being modified or deleted before it is transmitted:

- Specify that uucp immediately make its own copy of the file and transmit the copy (-C option of uucp; -p option of uuto). This method is simplest, but avoid it when copying large amounts of data at once.

- Use the uucp job status features to find out when the transmission actually takes place. See the subsection below on job and network statuses.

THE UUCP COMMAND

The uucp command takes the following form:

uucp *options* *filelist* *destination*

where

options can be omitted. The following options are used by ordinary users (other options are primarily used by administrators):

-C — Copy specified files to spool directory and use copies for transmission.

-m — Send mail to sender when copy is complete.

-m*file* — Write message to *file* when transfer is complete.

-n*user* — Notify *user* on destination system, via mail, when copy is complete.

-j — Print job number. This allows tracking and control of the uucp job; see the subsection below on uucp job and network statuses.

filelist is a list of files to be copied, with list elements separated by spaces. Each file name can begin with a system specification. File names with *, ?, and [...] metacharacters are expanded into multiple names on the specified system. Don't specify a directory without trailing file names or metacharacters; copying /x/dir/* is meaningful, but /x/dir is not.

destination is the destination file for the copy. If *destination* is not a directory, *filelist* must specify exactly one name; *destination* is the name of the copy. If *destination* is a directory, *filelist* can specify any number of names; the files are copied into *destination* under the same local names.

The destination directory must be writeable by the user uucp. Here are two ways to deal with this limitation:

- Copy files to the directory /usr/spool/uucppublic. uucp comands accept a tilde-slash sequence (~/) as an abbreviation for that name.

o Provide a directory useable by everyone as the destination directory. The following command modifies a directory in this way:

```
chmod a+wrx directory
```

where directory is the destination directory.

If you attempt to copy to a directory not writeable by the user uucp, the uucp system attempts to salvage the copy by copying the files to the public directory. In this case, the copied files end up in a directory called /usr/spool/uucppublic/user, where user is the originating user's name.

The following examples are from the network described in Figure G-1. The first example is run on homebase; it copies phonelist in the user's working directory to /usr/spool/uucppublic/phonelist.hb on alpha.

```
uucp phonelist beta!alpha!~/phonelist.hb
```

The next example, run on homebase, copies files from homebase (/a/jon/q.h and /a/jon/q.c) and from gamma (the entire contents of the directory /a/bill/dd) to the directory on beta (/b/sal/copies):

```
uucp /a/jon/q.[hc] gamma!/a/bill/dd* beta!/b/sal/copies
```

THE UUTO AND UUPICK COMMANDS

The uuto command originates the transmission of files. It takes the following form:

```
uuto options filelist recipient
```

where

options can be omitted. The following options are understood:

-p Copy the files to the uucp spooling directory and use the copies for transmission.

-m Send mail to the sender when the copy is complete.

filelist is a list of files to be sent; elements of the list are separated by spaces. The files must be on the local system; *, ?, and [...] metacharacters are expanded on the local system. Only ordinary shell conventions are understood; filelist must not contain a system designation or a tilde (~) directory designation.

Uuto interprets a directory name in filelist in a way more usefule than that of uucp. If a directory is specified, the entire hierarchy under the directory is copied.

recipient specifies the user to whom the files are sent. The user name normally begins with a system specification.

Note that uuto provides no option that prints uucp system job numbers. To get job numbers for uuto requests, see the subsection below on uucp job and network statuses.

The uuto command uses the uucp command to actually send the files. If directories are specified, uuto may call uucp more than once, resulting in multiple uucp system jobs.

The following example is run on the system homebase in the network example shown in Figure G-1. The example sends the files red and white in the working directory to user sal on system beta.

```
    uuto red white beta!sal
```

The uupick command picks up files sent by uuto. It must run on the receiving system. The command takes the following form:

```
    uupick -ssystem
```

where

-ssystem can be omitted. If -s is specified, only files from *system* can be picked up; otherwise, all files can be picked up.

Uupick queries for an action on each file waiting to be picked up. Uupick prints the file's origin, whether the file is an ordinary file or a directory, and the name of the file. Uupick then prompts for an action. An empty input line means no action just now; other input lines are the following:

d — Delete the file without copying it.

m — Move the file into the current directory.

m *dir* — Move the file into the directory *dir*.

a — Move the file and all other files from the same system into the current directory.

a *dir* — Move the file and all other files from the same system into the directory *dir*.

p — Print the contents of the file.

q — Terminate uupick.

An end-of-file condition also terminates uupick. An unrecognized command prints a command summary.

Remote Command Execution

The uux command provides remote execution of shell commands. It also permits local execution of commands with remote file arguments. Uux accepts simple commands and pipelines; all commands in a pipeline must execute on a single system.

Uux is limited by security requirements: it can only execute commands specifically allowed by the executing system. The system administrator for the system on which you want to do the execution can tell you what commands are allowed.

Uux uses mail to notify the user when the command has executed.

A uux command follows the following form:

uux _options_ _command_

where

options can be omitted. The following options are understood:

-	Read the standard input of uux and provide it as the standard input of the command executed.
-n	Don't notify the user when the command has executed.
-m_file_	When command has executed, write message to _file_. Don't send mail.
-j	Print uucp job number.

command is one or more parameters that make up the command to be executed. Different parts of the command can be separated by spaces within the parameters or by being put in separate parameters. Uux understands the same metacharacters as does the shell, in addition to uucp conventions for command and file names. However, only the first program in a uux pipeline can specify a system, the other programs automatically being executed on the same system. The shell metacharacters that are to be interpreted by uux must be quoted to prevent interpretation by the shell.

Uucp Command Limitations

uucp has limitations that prevent unauthorized use of a remote system. There are four basic limitations: limitations of the user uucp; parent directory names; forwarding from private directories; and administrative restrictions on access.

THE USER UUCP

All uucp actions are taken by the user uucp, a user without any special or privileged status. Thus if you prevent other users on your local system from using a file in a certain way (reading, writing) you are placing the same restriction on users on remote systems.

If the user uucp cannot read a file, you cannot use uucp to copy it, even if you own it. If a uucp command refuses to read a file, make sure that the file is readable by everyone. If the file is readable, check the read and search (execute) permissions of all the directories it is under. For example, suppose the uucp system cannot copy /a/jon/doc/bills. The following ls commands check all pertinent permissions:

```
    ls -l /a/jon/doc/bills
    ls -ld /a/jon/doc
    ls -ld /a/jon
    ls -ld /a
    ls -ld /
```

For information on file location and permission, see chmod(1), ls(1), and pwd(1) in the Series 6000 Operating System Reference Manual.

PARENT DIRECTORY NAMES NOT PERMITTED

Names that include parent directory references (for example, /usr/../xyzzy) are not permitted, because it is too difficult for uucp to determine whether such a name violates other restrictions.

FORWARDING OF PRIVATE DIRECTORIES

An extra restriction is placed on file copies that involve forwarding (communicating with systems not directly connected to the local system). When forwarding, the remote file must not be in a private directory: the remote name must begin with ~/ or /usr/spool/uucppublic/. This restriction applies to both uucp and uux. Uuto is not affected because it uses the public directory anyway.

ADMINISTRATIVE RESTRICTIONS ON ACCESS

The system administrator controls how other systems on the uucp network use the local system. There are three kinds of controls:

- Controls on who can access which files in the local system. The administrator imposes such controls by adding restriction information to the uucp user file.

- o Controls on execution of local commands. uucp will not execute a command unless the administrator adds the command to the uucp commands file.

- o Forwarding restrictions. If the administrator creates an origin file or forwarding file, there are restrictions on who can use the local system for forwarding uucp jobs, and there are restrictions on which systems can receive forwarded jobs.

Job And Network Statuses

The uustat command allows you to follow and control the progress of a uucp job until your system transmits the job to the first system in its route. It also monitors the status of your system's direct communication with other systems.

You need a uucp job number to follow and control a specific job. The -j option of the uucp and uux commands prints the uucp job number. For other commands, and to make job number printing automatic, set the environment variable JOBNO to ON. To restore manual control over the job number, set JOBNO to OFF.

The uustat command takes the following form:

```
    uustat options
```

where

options is a list of options and parameters. The following options are meant for ordinary users.

-jjob Report the status of job, which can be a job number to indicate a specific job; "all", to indicate all recent jobs; or missing, to indicate all recent jobs by the current user. Job status reports give the job number, the name of the user who originated the job, the next system on the job's route, the time the job was queued, the last time the job's status changed, and a description of the job's current status.

-uuser Report the status of all jobs sent by user.

-ssys Report the status of all jobs that communicate with the system sys.

-kjobno Kill the job whose uucp job number is jobno.

-r*jobno* Rejuvenate the job whose job number is *jobno*. A new or rejuvenated job is safe from uucp's maintenance program for a period determined by your administrator (normally a week). A job that waits through the standard period without being transmitted or rejuvenated is automatically killed.

-m*sys* Give the status of the communication link with the system *sys*. If *sys* is "all," give status for all systems known to the local system. A status report consists of the machine name, the time the status last changed, and the current status.

-M*sys* Like -m, but also gives the time of the last successful transmission.

CONFIGURING UUCP

This section tells how to configure individual uucp communication links and how to configure an operating system that runs uucp. The discussion is of three parts:

- Basic uucp concepts
- Configuring a uucp communication link
- Maintaining the system that runs uucp.

This section describes uucp configuration files and demons. However, it is not a complete reference to uucp user and administrative commands. For this purpose, refer to the following entries in the *Series 6000 Operating System Reference Manual*: uuclean(1M), uucp(1C), uustat(1C), uusub(1C), uuto(1C), and uux(1C).

Basic Uucp Concepts

Uucp uses the same communication methods used by users on interactive terminals. A communication link is established between two computer systems when one computer system calls another. The call is required even when the two systems are directly connected.

A communication link is controlled from one end: uucp only permits one of the two systems on a link to exercise the link. The other system must wait to be called. The only way to permit two systems to call each other at any time is to provide two links. An alternative to a double link is to have the controlling system call the waiting system at regular intervals (polling).

uucp supports two kinds of communication links:

- *Direct link*. The two systems are directly connected, normally with two RS-232 ports connected by a null modem cable.

- o *Telephone link*. Each system has a modem. The system that exercises the link must have a smart modem (a modem that can dial up another modem) or an automatic call unit. (The operating system kernel currently supports smart modems but not automatic call units.)

If a system uses multiple telephone links, it does not need a modem for each, although multiple modems increase the number of links that can be active at one time.

The examples in this section configure the network shown in Figure G-2. This is the same network discussed in the previous section, but shown in more detail here. System alpha and system beta are at the same location; system homebase is at a different location in the same city; system gamma is in a different part of the country. There is one direct link: alpha to beta. There are four telephone links: alpha to gamma; beta to homebase; homebase to beta; and homebase to gamma. The direct link runs at 9600 baud, the telephone links at 1200 baud. Note that gamma controls no links; all uucp jobs that originate at gamma must wait for a call from alpha or homebase.

Actual data sending and receiving is done by the uucp copy-in/copy-out demon, uucico. (A demon is a program that normally runs in background). Uucico runs with its effective user ID set to uucp, implementing the uucp feature that all uucp commands are executed by that user. Each communication link requires a uucico running on each system, in dialog with the uucico on the other system. The uucico on the system that initiated communication is the *master*; the other uucico is the *slave*.

uucico run as a master is the program that actually establishes a communication link. It runs in one of three modes, controlled by its -s parameter:

- o Clean up: If the -s parameter is missing, uucico scans the uucp spooling directory. uucico calls each neighboring remote system that has jobs waiting on the local system. Uucp, uux, and uudemon.hr (a maintenance demon that normally runs once an hour) run a master uucico without a -s parameter; uucp and uux do not wait for uucico to finish.

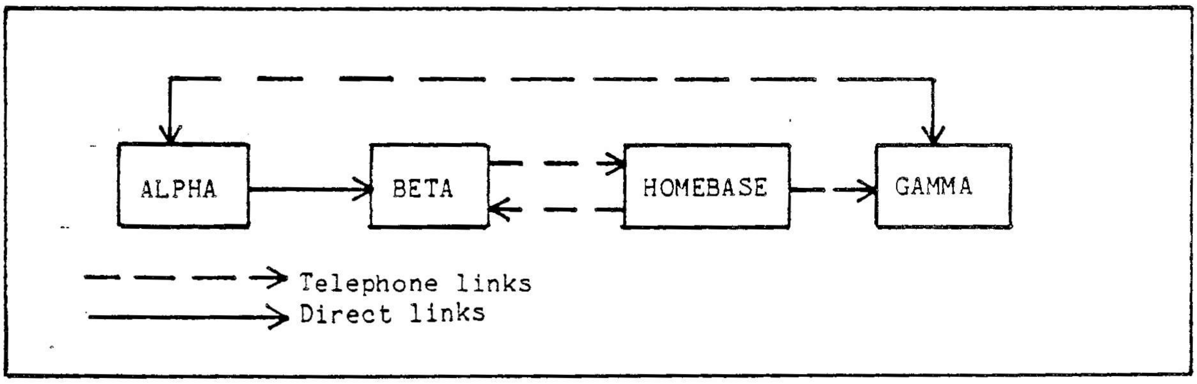


Figure G-2. Uucp Configuration Example

o Exercise all connections: If the -sall option is specified, the master uucico tries to call all neighboring systems. uudemon.day (a maintenance demon that normally runs once a day) runs a master uucico with -sall, guaranteeing that each communication link is exercised at least once a day.

o Exercise a specific connection: If the -sname option is specified, where name is a neighboring system's name, the specified system is called. This mode is used for call-back links, to poll a system, or to test a communication link.

If a master uucico calls more than one system, the order in which the systems are called is random.

Any number of uucico processes can be running at once. uucp and uux spawn uucico, to make sure a new job is transmitted right away, if possible. Thus, the more users who use uucp, the more uucicos are run. Lock files prrevent two uucicos from communicating with the same remote system at the same time and from trying to use the same modem or line. The uucp system file prevents uucicos from calling a system at a time forbidden by the administrator.

Each time a master uucico selects a system to call (chosen at random from its list or specifically ordered by the -sname option), master and slave follow the following procedure. To simplify things, SysM designates the master's system, SysS the slave's.

a. The master checks the SysM uucp system file to see when calls to SysS are permitted and to find out how to call all communication with SysM, the master gives up on calling SysS (end of procedure).

b. The master checks the lock files in the SysM uucp spooling directory. If all possible communication lines are locked (in use by other programs, including other uucicos) or if SysS is locked (another SysM uucico already talking to SysS (end of procedure).

c. The master creates lock files to claim the communication line and to prevent other SysM uucicos from talking to SysS.

d. If the line requires a telephone connection, the master opens the line and has the smart modem or automatic call unit dial up SysS. If the line is direct, the master just opens it.

e. Emulating a user on a terminal, the master logs in to SysS, using a special uucp user name. The special user's shell is uucico; since the SysS uucico is run without the option making it a master, it runs as a slave.

f. The master tells the slave the SysM is calling.

g. The slave checks to see whether communication with SysM requires call-back. If call-back is required, the slave refuses to talk to the master, executes a master uucico of its own, and terminates. The slave's termination logs the master off of SysS. The SysM master removes its lock files and gives up on calling SysS (end of procedure).

h. The slave creates a lock file for SysM in the SysS uucp spooling directory.

i. Master and slave execute SysS's uucp jobs queued on SysM.

j. The master asks the slave if it wants to terminate the conversation. If SysS doesn't have jobs queued for SysM, the slave says Yes -- go to step m. If SysS has jobs queued for SysM, the slave says No -- go to the next step.

k. Master and slave execute SysM's uucp jobs queued on SysS.

l. The slave asks the master if it wants to terminate the conversation. If SysM doesn't have jobs for SysS, the master says Yes -- go to the next step. If SysM has jobs for SysS (queued while SysS's jobs were being executed), go to step i.

m. The slave removes SysM's lock file and terminates; the slave's termination logs the master off of SysS. The master closes the line and removes its lock files. The master is done with SysS (end of procedure).

Configuring Communication Links

Each communication link requires the following configuration steps:

a. Install the necessary communication hardware.

b. Assign each system a node name.

c. If the system that makes the calls uses a smart modem, get the modemcap name for that modem.

d. Make sure that getty does not monitor the caller's terminal interfaces.

e. Create an appropriate special file for the caller's lines.

f. Configure the caller's terminal interfaces for use by uucp.

g. Make sure that getty does monitor the called system's terminal interfaces.

h. Provide a user name on the called system for use by the caller.

i. Change the uucp system file on the caller to make calls to the other system; change the uucp system file on the called system to accept calls from the other system.

j. Control each system to place necessary security restrictions on the other system.

Some of the steps above need not be repeated for each new communication link:

- A node name is usually assigned only once. Only conflicts (such as when two networks are merged) require new node names.
- A single terminal interface and smart modem can call more than one system.
- A single terminal interface and modem can receive calls from more than one system.
- A single user name can handle logins from more than one system.
- If the system file allows calls to be made to another system, it also allows calls from that system.

Thus at some steps in the procedure above it is only necessary to make sure that the existing configuration supports the new link.

HARDWARE INSTALLATION

RS-232 ports on System 6300's can handle communication up to 9600 baud.

A direct link requires a null modem (cross connected) cable. Limited distance modems (line drivers) may be necessary for cables longer than fifty feet.

A telephone link requires a smart modem on the calling system and a compatible autoanswer modem on the other system. See the installation instruction for your brand of modem.

In following examples, we will assume the following hardware configuration. B & F is an imaginary manufacturer of autoanswer, autodial modems that run at 1200 baud.

alpha	cable to beta on tty001 B & F modem on tty002
beta	B & F modems on tty001 and tty002 cable to alpha on tty019

homebase	B & F modems on tty001 and tty002
gamma	B & F modem on tty001

Note that operating system terminal numbers must be expressed in three digits.

ASSIGNING A NODE NAME

The node name is the system name that appears in all uucp system designations. The setuname command sets a system's node name:

```
setuname -n name
```

where

name is the node name.

setuname must be executed each time the operating system is rebooted. Arrange for this to happen automatically by adding the command to one of the system start-up scripts, /etc/rc.

Either of the following commands verifies that the node name has been correctly set:

```
uname -n
uuname -1
```

In the following example, the administrator of homebase sets the node name and provides for future automatic setting. The admnistrator's input is shaded, the system's responses plain.

```
# setuname -n homebase
# ex /etc/rc
"/etc/rc", 33 lines, 608 characters
:/node name
:       Set node name here
:a
setuname -n homebase
.
:xit
"/etc/rc", 34 lines, 629 characters
#
```

Appendix G
Uucp

SMART MODEM NAMES

The smart modems that uucp can use are described in the text file /usr/lib/uucp/modemcap. Each modem has several short names, listed at the beginning of its description. This file is documented under modemcap(5) in the *Series 6000 Operating System Reference Manual*. New modems are defined simply by editing the file.

In following examples, the name for the imaginary B & F smart modem is assumed to be bf.

GETTY ON THE CALLING SYSTEM

Getty is the system program that monitors terminal lines for attempted logins; a separate process monitors each line. Getty must not monitor lines used for calling other systems.

The who command tells you which lines getty monitors:

```
    who -l
```

or

```
    who -l ¦ fgrep ttyxxx
```

where

xxx is the three-digit terminal number. The second form of the command restricts output to the line you are interested in. If getty monitors the line, who prints a status for the line:

```
    LOGIN     tty001    Apr 12 11:19   old      5309
```

You are interested in the status's first field (who or what is using the line) the second field (the terminal number) and the last field (the numeric ID of the process controlling the line).

To get getty off of the line, do the following steps:

a. If no status is shown, getty is not monitoring the line: skip the remaining steps. If the first field of the user status is "LOGIN," no one is using the line: go to the next step. If the first field in the who status is a user name, that user is using the line: have the user log off and get who status.

b. Edit /etc/inittab and comment out the terminal configuration entries for the line (there are normally two entries). To comment out a line, insert a pound sign (#) at the beginning of the line.

c. Have init reread /etc/inittab:

```
            telinit q
```

CAUTION

> Use the telinit command carefully and precisely. The wrong parameter will stop the operating system suddenly and corrupt open files.

d. Make sure that the last getty died:

```
kill x
```

where

x is the process number given in the last who status. The kill command will fail if init has already killed getty for you.

e. Repeat the who command to verify that you did the last three steps correctly.

In following examples, the systems from the previous examples use the following lines for calling other systems:

alpha	tty001 (direct line to beta) tty002 (smart modem)
beta	tty001 (smart modem)
homebase	tty001 (smart modem)

In the next example, homebase's administrator removes the getty from tty001.

```
# who -l ¦ fgrep tty001
LOGIN      tty001      Apr 12 11:20    old     5309
# ex /etc/inittab
"/etc/inittab" 54 lines, 2144 characters
:/001
001:2:respawn:/etc/getty tty001 9600
:substitute/ /#/
#001:2:respawn:/etc/getty tty001 9600
:+
C001:6:respawn:/etc/getty tty001 C9600
:substitute/ /#/
#C001:6:respawn:/etc/getty tty001 C9600
:xit
"/etc/inittab" 54 lines, 2146 characters
# telinit q
# kill 5309
kill: 5309: no such process
# who -1 ¦ fgrep tty001
#
```

CREATING SPECIAL FILES

New special files for calling lines are not strictly necessary, but do avoid mistakes when using the lines: accessing the wrong modem or direct line is apt to be less drastic a mistake than accessing the wrong terminal line.

Use ln and chown to create a special file used to call over a direct line:

```
    ln /dev/ttyxxx /dev/d.name
    chown uucp /dev/d.name
```

where

xxx is the terminal interface's three-digit line number.

name is the name of the system that the direct line calls.

Also use ln and chown to create a special file used to call over telephone lines:

```
    ln /dev/ttyxxx /dev/cuay
    chown uucp /dev/cuay
```

where

xxx is the terminal interface's line number.

y is a sequence number that identifies the particular outgoing phone line.

The last part of the special file name (d.*name* or cua*y*) is the *device name*. The device name is used in uucp configuration files and by interactive communications programs such as cu.

The following command verifies that a terminal line special file is equivalent to special file:

```
    ls -li /dev/ttyxxx /dev/devname
```

where

xxx is the terminal interface's line number.

devname is the corresponding device name.

The ls command prints out two file status lines, each of which begins with an i number. The two i numbers should be the same.

In the following examples, the administrators on alpha, beta, and homebase create calling special files. The first example is for alpha.

```
    # ln /dev/tty001 /dev/d.beta
    # ln /dev/tty002 /dev/cua0
    # chown uucp /dev/d.beta /dev/cua0
    #
```

The next example is for beta and for homebase.

```
    # ln /dev/tty001 /dev/cua0
    # chown uucp /dev/cua0
```

Note that this configuration is not necessary on the system being called because to that system the lines used by uucp are just login terminals.

CONFIGURING THE CALLER'S TERMINAL INTERFACE

The uucp lines file, /usr/lib/uucp/L-devices, configures lines used to call out. This test file has two kinds of lines: comments, which are ignored, and line configurations. A comment begins with a pound sign (#). A line configuration describes a single line and is a text line of the form

type *name* *cd* *speed* *protocol*

where

type is the line type: DIR for direct lines, ACU for phone lines (both smart modem and automatic call unit).

name is the device name (d.name or cua*y*).

cd is the calling device. For automatic call units, give the call unit's device name. For smart modems, give the smart modem name, as listed in the modemcap file. For direct lines, put any nonblank text to hold the place.

speed is the normal baud rate of the line.

protocol is the communication protocol. This field is x for X.25, missing for any other protocol.

The following examples configure the calling lines on alpha, beta, and homebase, using "bf" modems. First alpha:

```
# ex /usr/lib/uucp/L-devices
"/usr/lib/uucp/L-devices" [Read only] 6 lines, 78 characters
:append
DIR d.beta 0 9600
ACU cua0 bf 1200
.
:xit
"/usr/lib/uucp/L-devices" File is read only
:xit!
"/usr/lib/uucp/L-devices" 8 lines, 116 characters
#
```

The "Read only" message indicates that the file lacks write permission. Thus, although uucp owns the file, only the superuser can modify it. Now for beta:

```
# ex /usr/lib/uucp/L-devices
:/usr/lib/uucp/L-devices" [Read only] 6 lines, 78 characters
:append
ACU cua0 bf 1200
.
:xit!
"/usr/lib/uucp/L-devices" 7 lines, 100 characters
#
```

Homebase is the same as beta.

GETTY ON THE CALLED SYSTEM

Lines for receiving uucp calls are configured precisely like lines for receiving users. A telephone dial-up line can, in fact, serve both uucp and ordinary user logins. (Direct lines work both ways too, but the cu program is needed to get access to them.)

uucp does not require the called system to answer the call with a particular terminal line.

UUCP USER NAMES

A system that wants to accept calls from other systems must provide a user whose home directory is /usr/spool/uucppublic and whose shell is /usr/lib/uucp/uucico (the copy in/copy out demon). The called system must also mention the user name in the uucp user file.

Your operating system is distributed with such a usr, named nuucp.
Use the passwd command to set nuucp's password (passwd(1) in the Series 6000 Operating System Reference Manual). Disseminate nuucp's password carefully, as it gives access to the network.

The system will work with nuucp as the only special user, but additional uucp user names provide security features:

- o You can change the password for one uucp user, locking out one group of systems without affecting communications with other (presumably more trustworthy) systems.

- o uucp allows you to impose file copying restrictions in addition to those imposed by operating system's file permissions. These restrictions divide neighboring systems into classes according to which name they use to log in to the local system.

If the nuucp user is not already in the user file, /usr/lib/uucp/USERFILE, the following line in the file properly mentions it without imposing any special restrictions:

```
    nuucp, /
```

The above entry in the user file allows any remote system to log in and access any file not protected from the user uucp. To impose restrictions, see the subsection below on maintenance.

In the network example in Figure G-2, the four systems modify their password and user files as follows:

alpha: This system lacks any facilities for remote logins. On the presumption that logins by other systems must be unauthorized, alpha's administrator edits nuucp's entry in /etc/passwd:

```
nuucp:x::6:1:/usr/spool/uucppublic:/usr/lib/uucp/uucico
```

The invalid second field renders logins by other systems impossible. Since alpha doesn't permit other systems to log in, no change is necessary to /usr/lib/uucp/USERFILE.

beta: The administrator decides that other systems at the same site will login as nuucp, but offsite (and potentially untrustworthy) systems will login as ouucp. Therefore, he edits the following lines into /etc/passwd:

```
nuucp:::6:1:/usr/spool/uucppublic:/usr/lib/uucp/uucico
ouucp:::6:1:/usr/spool/uucppublic:/usr/lib/uucp/uucico
```

He then uses passwd to assign the password "ourgang" to nuucp and the password "xyzzy" to ouucp. He tells alpha's administrator the nuucp password and homebase's administrator the ouucp passwd.

The administrator edits /usr/lib/uucp/USERFILE to contain the following lines:

```
nuucp, /
ouucp, /
```

homebase: This system currently accepts remote system logins by beta. Alpha or gamma may want access later, but if they do they'll have the same status as beta. Homebase already has the user nuucp; the administrator gives nuucp the passwork "greekaccess".

gamma: Gamma treats all the other systems the same. The administrator gives nuucp the password "compo" and adds the standard line to /usr/lib/uucp/USERFILE:

```
nuucp, /
```

THE UUCP SYSTEM FILE

The Uucp system file, /usr/lib/uucp/L.sys, tells which systems are neighbors to the local system. If the local system can call the other system, the system file provides one or more procedures for making the call.

Each line in the system file is either a comment or a system line. A comment begins with a pound sign (#) and is ignored. Each neighboring system line has one of two forms. The first form permits the other system to call the local system:

name

where

name is the node name of the other system.

The second form allows the local system to call the other system and the other system to call the local system. Don't provide both forms for one system. If there is more than one procedure to call the remote system (as with a system that has more than one phone number), provide one system line for each such procedure; the different procedures will be tried in the order listed. Here is the second form:

name *when* *device* *speed* *number* *login*

where

name is the node name of the other system.

when specifies permitted calling times and minimum retry period. The simplest value is Any, which places no restrictions. *When* takes the form shown below. Note that hours and retry are optional; omit the comma if you omit retry.

dayshours,*retry*

where

days is a concatenation of abbreviations indicating the days on which the procedure line can be used. Abbreviations are Su, Mo, Tu, We, Th, Fr, and Sa for specific days; Wk for any weekday; and Any, for any day of the week.

hours specifies the period of the day when the proceure can be used. If *hours* is omitted, the procedure can be used any time of day. The period is specified by two 24-hour clock times separated by a dash (for example: 0900-1700 for 9 am to 5 pm). The period can extend through midnight.

retry specifies the minimum waiting period, in minutes, after an unsuccessful call. If *retry* is omitted, the minimum waiting period is 30 minutes. The retry period applies if the last call to *name* was unsuccessful: the procedure cannot be used unless the minimum waiting period has expired. This feature prevents uucp from continuously trying to access a system that may be unavailable.

Here are some examples for *when*: Any0800-0700 means any time except 7 am to 8 am; SaSu means any time from 1201 am Saturday to midnight Sunday; Any,10 means any time but not if the last try was unsuccessful and less than 10 minutes ago.

device is ACU for telephone links. uucp will search the devices file for a dial-out device with the same speed. For direct links, this field must be the device name for the line.

speed is the speed used. This field must match the speed in the devices file.

number is the phone number. The characters have the following meaning:

0-9	dial 0-9
* or :	dial *
# or ;	dial #
-	delay four seconds
w or =	wait for next dial tone
f	flash off hook for one second

On direct lines, *number* is the same as *device*.

login defines the login procedure. *Login* consists of a list of *expect/send* sequences:

expect *send* *expect* *send* ...

The last item can be either an *expect* or *send*. The local system waits for the remmote system to print a line containing the first *expect*. When it arrives, the local system sends a line consisting of the next *send* and waits for the next *expect*; and so on. When the local system sends or reads the last item, the remote system should have let it log in and be running a slave uucico.

An *expect* can include a subprocedure used if the remote system doesn't print the desired string before a certain period of time. In this case the *expect* consists of *subexpect*/*subsend* sequences:

```
    subexpect-subsend-subexpect-subsend ...
```

If the remote system prints a line containing the first *subexpect*, the rest of the *expect* is skipped. If *subexpect* doesn't appear within a certain period of time, the local system sends the next *subsend*, waits for the next *subexpect*, and so on.

Here is an example:

```
    ogin--ogin--ogin nuucp assword xyzzy
```

This *login* logs in to a remote system as nuucp, passwd xyzzy. If the master uucico can't get a login prompt at first, it tries twice to provoke one by sending an empty line. Note that the first letter of most words is missing. By matching the end of each word, allowances are made for capitalization.

Here are some system file examples. The first example is alpha's system file, which allows it to call (and be called by) gamma and beta:

```
    gamma Any ACU 1200 408-5558328 ogin--ogin--ogin nuucp assword compo
    beta Any d.beta 9600 d.beta ogin--ogin--ogin nuucp assword ourgang
```

The next example is beta's system file, which allows it to call homebase and accept calls from alpha and homebase:

```
    alpha
    homebase Any ACU 1200 5550101 ogin--ogin--ogin nuucp assword ourgang
```

The following example is homebase's system file. It provides procedures for homebase to call beta and gamma. Calls to gamma are restricted to night hours and weekends to save long-distance costs.

```
    beta Any ACU 1200 9w5559393 ogin--ogin--ogin ouucp assword xyzzy
    gamma Wk2200 ACU 1200 9w408-5558382 ogin--ogin--ogin nuucp assword compo
    gamma SaSu ACU 1200 9w408-5558382 ogin--ogin--ogin nuucp assword compo
```

Finally, we have gamma's system file, which permits calls from alpha and homebase:

```
    alpha
    homebase
```

TESTING THE LINK

You can test a link using the cu program or using uucp itself.

Testing with Cu

To test a direct link with cu, run the following command on the calling system:

```
    cu -l device
```

"ell"

where

device is the device name for the link. Attempt to log in and out of the remote system. Enter a line beginning with tilde-period (~.) when you are done.

The following command tests a telephone link:

```
    cu -l device number
```

"ell"

where

device is the device name for the link.

number is the telephone number of the remote system.

If this version of cu doesn't work, try it without the phone number; you will have to enter modem commands directly. As with the direct link, attempt to log in and out of the remote system. Enter a line beginning with tilde-period (~.) when you are done.

Testing with Uucp

The following two steps use uucp to test a link:

a. Execute a uucp transfer between the two systems, using the -r option to suppress the uucico demon. For example:

```
        uucp -r shortfile gamma!~/
```

b. Run the uucico demon, as a master, in debug mode, and in foreground. If one system must call the other, be sure to use that system.

```
        /usr/lib/uucp/uucico -r1 -x4 -sname
```

"eins"

-x n [illegible]
-r 1 [illegible]

where

name is the name of the system to be called.

The demon's debug output contains many messages that are meant to debug the demon itself and are not documented. But messages relating to your configuration are self-explanatory.

To find out how normal demons have done, check the log files: /usr/spool/uucp/LOGFILE (today's activity) and /usr/spool/uucp/Log-WEEK.

Maintenance

This section discusses day-to-day maintenance. There are three kinds:

- Automatic Maintenance
- Security configuration
- Emergency

AUTOMATIC MAINTENANCE

Automatic maintenance is performed by uucp background programs (demons). Certain chores are standard and are performed by standard uucp demons, which must be configured. Nonstandard demons poll slave systems.

Standard Demons

Each system running uucp should regularly execute the following standard demons: uudemon.hr, at four minutes before each hour; uudemon.day, at 4 am each day; and uudemon.wk, at 5:3Ø am every Sunday. To arrange this, add the following lines to /usr/lib/crontab:

```
    Ø 4 * * * /usr/lib/uucp/uudemon.day
    56 * * * * /usr/lib/uucp/uudemon.hr
    3Ø 5 * * Ø /usr/lib/uucp/uudemon.wk
```

This is what the standard demons do:

uudemon.hr   Run a master uucico without the -s option to attempt completion of uucp jobs waiting on the local system. Use uulog to merge recent status reports into the log file.

uudemon.day	Use uuclean to kill all uucp jobs waiting on the local system that are at least 168 hours old. Use uuclean to kill all uux commands trying to execute on the local system that are 72 hours old. Move the contents of today's log to the end of the weekly log. Use uusub to call all systems this system knows how to call and to gather traffic statistics for the last 24 hours. Use find to remove all ordinary files from the public directory, /usr/spool/uucppublic, that are more than 3Ø days old.
uudemon.wk	Handle the weekly logs, getting rid of those that are two weeks old.

Since the demons are shell scripts, you can edit them to add, change, or delete features. Systems that can't afford the amount of space used by /usr/spool/uucp and /usr/spool/uucppublic usually shorten the time period parameters for the uuclean and find commands in uudemon.day.

Polling Demons

If a system must wait for all calls, it may be desirable to have other systems poll it. Use cron and uusub to do this: see cron(1M) and uusub(1M) in the Series 6ØØØ Operating System Reference Manual.

The following example is a line for /usr/lib/crontab; it attempts to poll system gamma every four hours. It is done one-half hour after the previous uudemon.hr to minimize the chance that these two demons will interfere with each other's use of the outgoing lines.

```
26, Ø,4,8,12,16,2Ø * * * /usr/lib/uucp/uusub -cgamma
```

SECURITY MEASURES

You can place the following restrictions on uucp use:

o How specific remote users and systems can use your system.

o To what extent your system will forward files and commands from one remote system to another.

o Which commands will be executed by uucp.

Remote Access to the Local System

The uucp system has three kinds of restrictions on local file access:

o Files not accessible to the user uucp (a user without special status or privileges) are not accessible to the uucp system.

- Systems that access the local system through forwarding (that is, systems not directly connected to the local system) can only access files under the uucp public directory, /usr/spool/uucppublic.

- Additional restrictions imposed by the local administrator.

The first two kinds of restrictions cannot be modified without modifying uucp. The remainder of this section discusses the third kind.

The user file, /usr/lib/uucp/USERFILE, is a text file that controls the way users and systems use uucp to access the local system. It has four kinds of controls that apply to general classes of files and to specific users and systems:

- Which files local users can copy using uucp.

- Which files remote systems can access.

- Which login name each remote system must use to talk to the local system.

- Whether a neighboring system must be called back to confirm its identity.

Controls are written into the user file with lines of the following form:

user,*system* *callback* *prefixlist*

where

user is a user name on the local system (either a real user or a user used for uucp logins) or a null string.

system is the node name of a remote system or a null string.

callback is the call-back flag, c, or a null string.

prefixlist is a list of initial parts of full path names of files. Elements of the list are separated by spaces.

uucp uses four rules to interpret the user file:

- When a remote uucico logs in, the local uucico searches the user file for an entry that allows the remote system to call. The user name in the entry must match the user name used by the remote uucico; the system name in the entry must be null or match the remote system's name. The first entry with such a match allows communication.

- If the entry matched by the previous rule has a call-back flag, that system must be called back.

- When a uucp job is stored on the local machine, uucp compares the name of the user responsible with the user names in the usr file. The first name that matches yields a list of file name prefixes. If no name matches, the first line with a null user name yields a list of file name prefixes. In any case, the full name of each file to be accessed by the uucp job is compared with the list of prefixes. If the full path name doesn't begin with one of the prefixes, access to that file is denied.

- When a uucp job from a remote system is executed, uucp compares the name of the remote system with the system names in the user file. The first name that matches yields a list of file name prefixes. If no name matches, the first line with a null system name yields a list of file name prefixes. In any case, the full name of each file to be accessed by the uucp job is compared with the list of prefixes. If the full path name doesn't begin with one of the prefixes, access to that file is denied.

Here are some examples. If a user name is of the form _x_uucp, assume that it is a special uucp user (it has uucico as its shell). The first example allows any system to log in as nuucp and access any file:

```
nuucp, /
```

The next example allows system homebase to log in as ouucp and access any file whose full path name starts with /a/scott:

```
ouucp,homebase /a/scott
```

This example allows the local user bill to access files with uucp commands, but only if their full path names begin with /a/bill:

```
bill, /a/bill
```

The next example allows any remote machine to log in as nuucp and access files whose names begin with /usr/spool. In addition, alpha can access files whose names begin with /a/bill.

```
nuucp,alpha /usr/spool /a/bill
nuucp, /usr/spool
```

The last example allows any local user to access any file whose name begins with /usr/spool; further, it allows the local user root to access any file at all.

```
root, /
, /usr/spool
```

Note that even root cannot access a file that is not accessible by the user uucp!

Forwarding

You can place two restrictions on uucp jobs forwarded through your system:

- Specify the systems your system will call in order to forward other systems' jobs.
- Specify the systems and users on whose behalf your system will forward jobs.

The system forward file, /usr/lib/uucp/FWDFILE, is a list of neighboring systems, one per line. If the system forward file exists, the next destination (not necessarily the final destination) of each job travelling through your system is checked; if the next system is not in your system forward file, the job is killed and the originator notified. This restriction does not aply to your own users. If the system forward file is absent, your system does *not* check the next destination of jobs forwarded through your system.

The system forward file is typically created to deny other systems access to an expensive communication link.

The origin file, /usr/lib/uucp/ORIGFILE, grants users on other systems permission to forward through your system. Each entry is a single line. if the entry is a node name, all users on that system can forward through your system. If the entry is a list of the form:

system!*list*

then users on the system whose node name is *system* can forward through your system only if their names appear in *list*. The elements of *list* are separated by exclamation points (!).

A job allowed by the system forward file (or the absence of the system forward file) can be killed by the origin file, and vice versa.

Suppose that the administrator of system alpha decides that homebase users are specifying beta!alpha!gamma! instead of gamma! to avoid incurring costs to their own system. He creates /usr/lib/uucp/ORIGFILE with the following contents:

```
beta
gamma
```

The administrator of homebase makes a similar decision about alpha, beta, and gamma. To cut off all forwarding through homebase, he creates two empty files: /usr/lib/uucp/ORIGFILE, and /usr/lib/uucp/FWDFILE (although either would have been sufficient).

Permitted Commands

For uucp to execute a program other than its own demons, the command name for the program must be entered in the commands file, /usr/lib/uucp/L.cmds, one command name per line. Without such an entry, a command cannot be used within uux. If rmail (a restricted version of mail) is not mentioned in the commands file, local users cannot get mail notification of job outcomes or receive mail from users on other systems.

Add commands to the command file with care, since a sufficiently general command (such as cat) permits remote users to overcome your uucp security restrictions. A security-conscious system typically permits only rmail.

EMERGENCIES

uucp creates most of its temporary files in /usr/spool/uucp. uucp users are prone to create many files in /usr/spool/uucppublic. Therefore, keep a close eye on the free space of the file system that holds /usr/spool: running out will paralyze uucp and possibly other parts of your system as well.

It is worth noting that if you restart uucp on your system after a hiatus, jobs queued on neighboring systems to execute on or pass through your system will arrive all at once, possibly creating a new logjam.

uucp and some other communications programs create lock files called /usr/spool/uucp/LCK..name, where name is a device or remote system name. If the system or uucp crashes while a communication program is working, these files may stick around, preventing your restarting communication. Remove the lock files to get things going again, but be quite sure they don't belong to an active uucico, cu, or other such program. The safest time to remove a uucp lock file is when the operating system is in single-user mode. You may find it useful to have the rm command in operating system start-up script, /etc/rc.

A Direct Link Example

This subsection describes the configuration of a simple direct link. The network consists of two machines: a System 63ØØ and a System 66ØØ, Each system has a minimal communication hardware: the System 63ØØ lacks any input/output expansion and thus only has the input/output channels provided by the main processor board; the System 66ØØ has a single Cluster Processor and no terminal processors. They are linked by a null modem cable.

The node names chosen for the systems are "63ØØ1" and "66ØØ1". The link is controlled by 63ØØ1 and runs at 96ØØ baud.

This example does not provide any security restrictions.

CONFIGURING 66ØØ1

The following communication configuration steps apply to 66ØØ1:

- Assign the node name.
- Configure the terminal interface that 63ØØ1 will use to call 66ØØ1.
- Provide a user name that 63ØØ1 will use to log in.
- Create an entry for 63ØØ1 in the uucp system file.
- Specify 63ØØ1's permissions for accessing 66ØØ1.

The administrator executes the following command and also adds it to /etc/allrc:

```
    setuname -n 66ØØ1
```

The cable is connected to the Cluster Processor's Channel 2. (There is nothing special about a Cluster Processor Channel 2, but it is the one usually used for uucp links because Channel 1 usually has an RS-232 user terminal and Channel 3 is less reliable at receiving large volumes of data at speeds above 96ØØ baud.) Since there is only one communication board, Channel 2 is the operating system's terminal number ØØ1.

To enable logins on ØØ1, the administrator adds the following line to /etc/inittabØØ:

```
    ØØ1:2:respawn:/etc/getty ttyØØ1 96ØØ
```

Terminal ØØ1 must not be active in Administrator Login Mode, enabled by the following line:

```
    CØØ1:6:respawn:/etc/getty ttyØØ1 C96ØØ
```

The administrator comments this line out:

```
    :CØØ1:6:respawn:/etc/getty ttyØØ1 C96ØØ
```

If the administrator had to modify /etc/initabØØ, he must make the modification effective by executing the following command on Application Processor ØØ:

```
    telinit q
```

The user name "nuucp", in the distributed version of the operating system, will serve for logins by 63ØØ1. The administrator confirms that the password file, /etc/passwd, already defines nuucp with the following line:

```
    nuucp:x:6:1::/usr/spool/uucppublic:/usr/lib/uucp/uucico
```

The second (password) field does not define a valid password, so the administrator used the passwd command to specify one:

```
passwd nuucp
```

The new password is "directtalk".

The administrator adds the following line to the uucp system file, /usr/lib/uucp/L.sys:

```
63ØØ1
```

This network only contains two systems and security is not a problem. Therefore, the administrator merely verifies that the following (very permissive) line is in the uucp user file, /usr/lib/uucp/USERFILE:

```
, /
```

CONFIGURI:NG 63ØØ1

The following communication configuration steps apply to 63ØØ1:

o Assign the node name.

o Make sure that getty does not monitor the line that 63ØØ1 uses to call 66ØØ1.

o Create a special file that uucp will use to access the line.

o Configure the line.

o Create an entry for 66ØØ1 in the uucp system file.

o Specify 66ØØ1's permissions for accessing 63ØØ1.

o Have the link exercised at regular intervals.

The administrator executes the following command and also adds it to /etc/rc:

```
setuname -n 63ØØ1
```

The line is connected to Channel B, chosen because the only other RS-232 channel is used by a user terminal. Channel B operating system terminal ØØ1.

Getty must not monitor ØØ1, so ØØ1 must not have any entries in /etc/inittab. The administrator checks the file, and discovers the following:

```
ØØ1:2:respawn:/etc/getty ttyØØ1 96ØØ
CØØ1:6:respawn:/etc/getty ttyØØ1 C96ØØ
```

He comments them out:

```
    :001:2:respawn:/etc/getty tty001 9600
    :C001:6:respawn:/etc/getty tty001 C9600
```

He then makes his change effective with telinit:

```
    telinit q
```

The following commands create and check the special file used by uucp:

```
    ln /dev/tty001 /dev/d.66001
    chown uucp /dev/d.66001
    ls -li /dev/tty001 /dev/d.66001
```

To specify communication configuration, the administrator adds the following line to the uucp devices file /usr/lib/uucp/L-devices:

```
    DIR d.66001 x 9600
```

The administrator makes the following entry in the uucp system file, /usr/lib/uucp/L.sys. Since the line is direct and costs nothing to use, there is no restriction on calling time, and a short retry period.

```
    66001 Any,5 d.66001 9600 d.66001 ogin--ogin--ogin nuucp assword directtalk
```

The administrator verifies that the following line is in the uucp user file, /usr/lib/uucp/USERFILE:

```
    , /
```

The link is now established and can be used. To make sure that no job originating at 66001 will wait more than thirty minutes, the administrator adds the following line to 63001's cron table, /usr/lib/crontab:

```
    0,30 * * * * /usr/lib/uucp/uusub -c66001
```

EXERCISING THE LINK

The link will be exercised every 30 minutes plus every time a 63001 user refers to 66001 in a uucp copy or remote execution command. If 66001 is down, then 63001 will try to exercise the link no more often than every 5 minutes. If an urgent uucp job is waiting on 66001, any 63001 user can get it going by entering the following command:

```
    /usr/lib/uucp/uusub -c66001
```

Appendix H
Reconfiguring the Operating System

You can use the SYSGEN utility to reconfigure the operating system. You might want to do this in order to:

- Add an RS-232 Expansion Board. The RS-232 Expansion Board is a hardware device that is added to the main processor board of the System 6300. It increases the number of RS-232 ports on the System 6300 from two to ten.

- Configure the system to support international character sets you may want to install.

- Configure the system for the maximum size of a single file in a database.

- Use ORACLE. You can configure the maximum number of users that can use ORACLE at the same time.

- Configure the system to support "BSC", "LAN", and "SNA".

SYSGEN is used to reconfigure the operating system kernel. The program SYSGEN and the files needed to run it are in "Utility Set 9," so it must be installed before you can run SYSGEN. See the UNIX-derived operating system Software Release Guide (SRG) for details on installing utility set 9 and other software products.

Before you reconfigure the operating system with SYSGEN you must install each of the software products you plan to use. For example, to run ORACLE on your system, you should first use the upgrade program (described in the UNIX-derived operating system Software Release Guide) to install ORACLE on your system. Then you can run the SYSGEN and use ORACLE.

To run SYSGEN, you must be in single-user mode. Then follow these directions:

a. Then enter the following command:

```
/etc/SYSGEN
```

The following messages appear on the screen:

```
***** UNIX-derived Operating System -- Kernel Sysgen *****

Default values are displayed in brackets [].
Type 'Return' to enter the default value.
Acceptable answers are displayed in parentheses ().
[Note: y, yes, Y, YES, Yes and n, no, N, NO, No are all acceptable.]

Type 'q' or 'Q' to quit sysgen.
```

Or you may see one of the following error messages:

```
Error: SYSGEN not available...Install Utility Set 9 of the Release
FE07 floppies

Error: You must be logged in as root!
Error: You must be in single-user mode!
       Execute shutdown then run SYSGEN
```

b. If all is well, you see the next question:

```
Are you changing your configuration since last running SYSGEN? [no]
(y,n) :
```

In either case, SYSGEN creates a new operating system and reboots the system automatically.

If you enter N and press RETURN, SYSGEN proceeds directly with building the operating system kernel. This gives you a fresh copy of the current operating system.

c. If you enter Y and press RETURN, SYSGEN asks this series of questions concerning the use of various Motorola Information Systems software products:

```
Initializing...

Are you using the RS-232 Expansion Board? [no] (y,n) :
```

With a yes answer, SYSGEN adds support for the Expansion Board to the operating system, and asks the next question. A no answer displays the next question directly.

d. Database Management Questions:

```
Are you using a data base management system? [no] (y,n) :
```

If yes, type Y and RETURN, and answer the next question.

```
Maximum size (in megabytes) for a data base? [1] (1-24) :
```

Your answer determines the maximum size that a file can have on the system. The default size is one megabyte. The maximum size is 24 megabytes.

e. Next you are asked:

```
If using ORACLE, what is the maximum number of users who may be
running ORACLE simultaneously? (0-8) :
```

This is not necessarily the total number of terminals on the system, just the maximum number that are likely to use ORACLE at the same time.

f. If your system has international character set tables to be installed, you will see the following:

```
     Loading international translate tables...
```

g. If you installed the "BSC" software product, you will be asked "BSC" questions:

```
     Do you want BSC support [no] (y,n) :
```

With a yes answer, SYSGEN continues to configure BSC. Refer to the "BSC" Software Release Notice for more information on answering "BSC" questions.

If you answer no, SYSGEN continues to the next question.

i. If you installed the "SNA" product, you get "SNA" questions:

```
     Do you want SNA support [no] (y,n) :
```

With a yes answer, SYSGEN continues to configure SNA. Refer to the "SNA" Software Release Notice for more information on answering "SNA" questions.

Answering no brings up the next question.

j. SYSGEN now rebuilds the operating system with the changes you have made. In doing so, SYSGEN first checks to see if the products you have configured will increase the size of the operation system beyond its maximum limit of 913408 bytes. If it does, you see the following message:

```
     Error:  Maximum kernel size exceeded!
              Check /usr/sys/cf/sysgen.log, adjust your requirements, and
          try again.
```

This message tells you that you need to cut down on something inorder to reconfigure the operating system. The "sysgen.log" file tells you where you are using space and helps yu see where you might save. An example of this file follows:

```
UNIX-derived Operating System, Release FE07

SYSGEN Logfile -- Thu Jun 27 11:01:56 PST 1985


Maximum kernel size is 913408 bytes.

Current size:
 text  = 131248
 data  =  11620
 bss   = 138508
 total = 281376

RS-232 Expansion Board:  adding 3432 bytes.

Installing BSC drivers.
 BSC:  adding 11016 bytes text for program code.
 BSC:  adding  1428 bytes data for program code.
 BSC:  adding   532 bytes bss for trace data.
3270:  adding  3048 bytes text for program code.
3270:  adding    48 bytes data for program
3270:  adding     8 bytes bss for 2 buffer pointer(s).
3270:  adding  4256 bytes bss for 8 device buffer(s).
x780:  adding  2716 bytes text for program code.
x780:  adding    28 bytes data for program code.
x780:  adding  4496 bytes bss for 2 serial port buffer(s).

BSC contribution to kernel size:
 text =  16780
 data =   1504
  bss =   9292
Total =  27576

Installing LAN drivers
 adding 26144 bytes text for program code
 adding  2912 bytes data for porgram code
 adding 24576 bytes bss for heap size
 adding  2436 bytes bss for socket ports
 adding   244 bytes bss for network tty ports
 adding   264 bytes bss for transmit buffers
 adding   136 bytes bss for receive buffers

LAN contribution to kernel size:
 text =  26144
 data =   2912
  bss =  28984
Total =  58040

Oracle:  adding 192 bytes bss for 4 message queues.
```

```
Adding 80 bytes bss for semaphore map.
Adding 8192 bytes bss for message buffer.
Adding 800 bytes bss for message map.

New size:
 text = 174828
 data =  16676
  bss = 191132
Total = 382636

Created .5Mb kernel + 3.5Mb user process virtual space
```

k. If the extimated size does not exceed the maximum size, you will see the following message:

```
Making UNIX-Derived Operating System, Release FE07A...
```

If the actual size of the new kernel exceeds 389120 bytes, KSYSGEN automatically builds a 1Mb kernel. In this event, you wil see the following message:

```
Created 1Mb kernel + 3.0Mb user process virtual space
```

l. On completion of SYSGEN, you get the following messages:

```
/usr/sys/FE07A installed as /unix

***** Sysgen Complete *****
Rebooting the system...
```

m. The system reboots automatically and you will be running with the operating system you have just configured.

Index