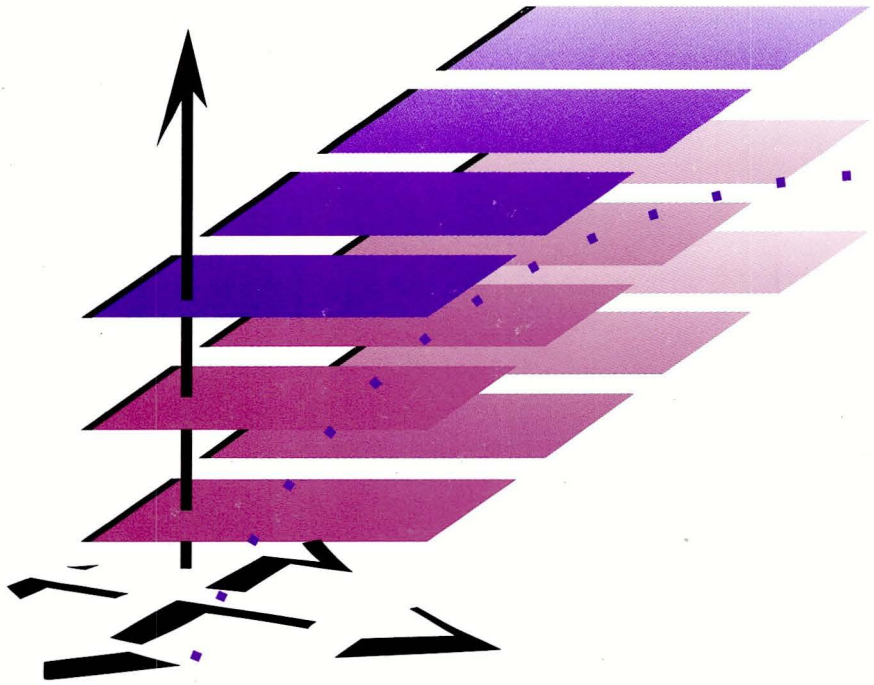

CONVEX

SPP-UX System Administration Guide

Third Edition

Hewlett-Packard Company
Convex Technology Center
3000 Waterview Parkway
P.O. Box 833851
Richardson, TX 75083-3851
United States of America

SPP-UX System Administration Guide

Order No. DSW-853

Third Edition

June 1996

Hewlett-Packard Company
Convex Technology Center
Richardson, Texas
United States of America

SPP-UX System Administration Guide

Order No. DSW-853

Notice

The information contained in this document is subject to change without notice.

Hewlett-Packard makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance or use of this material.

This entire book is recyclable.

Printed in the United States of America

Revision Information for

SPP-UX System Administration Guide

Edition	Document No.	Description
Third	710-029330-002	Released June, 1996 with SPP-UX V4.0. Added chapters for licensing and OBP and ROBP. Expanded file system chapter.
Second	710-029330-001	Released July, 1995 with Convex SPP-UX V3.1.
First	710-029330-000	Initial release October, 1994.

Contents

Figures

Tables

How to use this guide

The *SPP-UX System Administration Guide* describes fundamental concepts and tasks associated with setting up and maintaining an SPP-UX system.

Audience

This manual is intended for system administrators of all skill levels.

You should be able to:

- Log in and out of a remote computer system (using a command such as `rlogin` or `telnet`).
- Understand how the SPP-UX file system works, how to navigate through the file system using the `cd` command, and how to use both relative and absolute path names.
- Understand the SPP-UX file permission system, and be able to change permissions using the `chmod`, `chgrp`, and `chown` commands.
- Edit files using one of the SPP-UX editors (such as `vi` or `emacs`).
- Move, copy, and delete files using `mv`, `cp`, and `rm`, respectively.
- Search for text in files using `grep`.
- Display the contents of files using the `head`, `cat`, and `more` commands.
- Use one of the SPP-UX shells (`csh`, `ksh`, or `sh`).
- Understand how SPP-UX processes work and know how to start, stop, and display processes.
- Run an X Window System server on your local host.

Notational conventions

This section discusses notational conventions used in this book.

`Bold monospace`

In command examples, text shown in **`bold monospace`** identifies user input that must be typed exactly as shown.

`Monospace`

In paragraph text, `monospace` identifies command names, system calls, and data structures and types.

In command examples, `monospace` identifies command output, including error messages.

In command syntax diagrams, text shown in `monospace` must be typed exactly as shown.

Italic

In paragraph text, *italic* identifies new and important terms and titles of documents.

In command syntax diagrams, *italic* identifies variables that must be supplied by the user.

{ }

In command syntax diagrams, text surrounded by curly brackets indicate a choice. The choices available are shown inside the curly brackets and separated by the pipe (|) sign.

The following command example indicates that you can enter either `a` or `b`:

`command {a | b}`

[ ]

In command syntax diagrams, square brackets indicate optional data.

The following command example indicates that the variable *output_file* is optional:

`command` *input_file* [*output_file*]

...

In command syntax, horizontal ellipsis shows repetition of the preceding item(s).

The following command example indicates you can optionally specify more than one *input_file* on the command line.

`command` *input_file* [*input_file ...*]

KEYCAP

In paragraph text, text shown in **KEYCAP** indicates keyboard keys you must press to execute the command. For example, **RETURN** refers to the carriage return key.

Two **KEYCAP** terms separated by a hyphen indicate two keys that you must press simultaneously. For example, **CTRL-d** indicates that you must press the **d** key while holding down the **CTRL** key.

Notes and cautions

This document presents notes and cautions in the following formats.

Note

A Note highlights supplemental information.

Caution

A Caution highlights information necessary to avoid damage to the system.

Associated documentation

For more information, refer to:

Guide to Exemplar Documentation (DSW-851)

Describes the documentation available for Exemplar computer systems.

Exemplar SPP1000/1200 Architecture (DHW-014)

Describes the architecture of Exemplar computer systems.

Exemplar Networking Guide (DSW-865)

Describes how to configure and manage a network on an Exemplar system.

Exemplar Open Boot Quick Reference (DSW-854)

Summarizes the commands associated with the Exemplar Open Boot firmware.

HP-UX Reference (Hewlett-Packard order number B2355-90004)

Describes the end-user interface to HP-UX, which is compatible with the SPP-UX end-user interface.

Ordering documents

To order the current edition of this or any other Convex document, send requests to:

Hewlett-Packard
Convex Technology Center
Customer Service
P.O. Box 833851
Richardson TX 75083-3851 USA

Please include the order number (DSW or DHW number) or the exact title of the document.

Technical assistance

If you have questions that are not answered in this book, contact the Hewlett-Packard Convex Technical Assistance Center (TAC) at the following locations:

- Within the continental U.S., call 1 (800) 952-0379.
- From Canada, call 1 (800) 345-2384.
- All other locations, contact the local Convex office.

You can also use the `contact` utility to report any problems you may have with SPP-UX or its associated documentation. For more information, refer to the contact(1) man page on any Convex computer system.

Introduction 1

Setting up an SPP-UX system

Each Exemplar computer system is shipped with a bootable copy of the most recent release of SPP-UX. You should first read and follow the instructions in the release notice that you receive with your release of SPP-UX. When you complete the instructions presented in the release notice, continue by configuring your system as described in "Configuring an SPP-UX system" on page 9.

When you receive a new version of SPP-UX, read and follow the instructions in the release notice you receive. You need to follow the installation procedures described in the *Exemplar Software Distribution Notice* to install the new release. When you complete the installation procedures, continue by configuring your system as described in "Configuring an SPP-UX system" on page 9.

The system console

The system console starts automatically on the Exemplar test station when SPP-UX is booted on the Exemplar system. The `sn.cnsld` daemon must be running on the test station in order for the SPP-UX system console to run; if `sn.cnsld` is not running, you can start it by entering the following command on the test station (you must be logged in to the test station as root):

```
/spp/etc/sn.cnsld -t 1
```

The SPP-UX system console appears in an xterm window with a title bar that reads "System Console."

Accessing the system console remotely

You can access the SPP-UX system console remotely (that is, from a system other than the test station) by using the `sn_cnsl` command on the Exemplar test station. You must be logged in to

the test station (either locally or remotely) as root in order to use this command.

The `sn_cnsl` command allows you to either remotely view or assume control of the SPP-UX console window. The `sn_cnsl` command has the syntax:

`/spp/bin/sn_cnsl` [**`-fFsS`**] *console_id*

Where:

`-f`

Forces control of the SPP-UX system console to the host from which the command was entered.

`-F`

Forces control of the SPP-UX system console to the host from which the command was entered; also print the console messages stored in the message buffer.

`-s`

Spys on the main SPP-UX system console; all system messages written to the SPP-UX system console are copied to the host from which the command was entered.

`-S`

Spys on the main SPP-UX system console; all system messages written to the SPP-UX system console are copied to the host from which the command was entered; also print the console messages stored in the message buffer.

console_id

Identifies the console you wish to view; the default value is 1. If your system's root node is not node 0, *console_id* is the root node number + 1.

To exit `sn_cnsl`, enter:

CTRL ec. (the period is part of the command)

Disabling remote console access

If, for security reasons, your site does not allow remote access to the SPP-UX system console, you can

- Disable remote access by changing permissions for the `sn_cnsl` command.

  The following command removes execute permission from the `sn_cnsl` command:

  ```
  chmod 600 /spp/bin/sn_cnsl
  ```

- Remove the `sn_cnsl` command from the test station.

System administration utilities

You can access SPP-UX system administration utilities from any host that is connected to your Exemplar system. Most system administration utilities require that you be logged in to the Exemplar system as `root`; some require you to be logged in as `adm`.

The Graphical User Interfaces (GUI) for system administration utilities require that your local host be running an X Windows server. The SPP-UX X client is compatible with X11R4 and later versions of X Windows.

Using SAM

SAM is the System Administration Manager utility for SPP-UX. This tool allows you to perform many system administration tasks without using the underlying SPP-UX commands associated with the task. SAM can save you time and keystrokes.

SAM can help you with tasks in the following areas:

- Working with users' accounts
- Working with groups of users
- Maintaining system security
- Working with file systems
- Configuring your swap space
- Adding or removing peripherals
- Working with the line printer spooler
- Backing up and recovering files
  - Automated system backups
  - Manual system backups
- Configuring the following network connections:
  - Ethernet
  - FDDI
  - ATM
  - HIPPI
- Configuring UUCP communication
- Administering systems remotely from one location

SAM interfaces

SAM has two user interfaces:

- An X Window System graphical user interface (GUI)
- A text terminal interface

The two interfaces differ in screen appearance and in keyboard and mouse interactions. The examples in this manual generally show how to perform tasks using the GUI.

Starting SAM

To start up SAM, make sure that you are logged in to the Exemplar system with superuser (root) privileges. Since you run the SAM GUI on your local host, make sure that your `DISPLAY` environment variable is set to your local host; if you are running `csh` or `tcsh`, enter:

```
setenv DISPLAY localhost:0.0
```

Start SAM by entering:

```
/usr/bin/sam
```

The SAM main menu screen appears in an xterm window as shown in Figure 1.

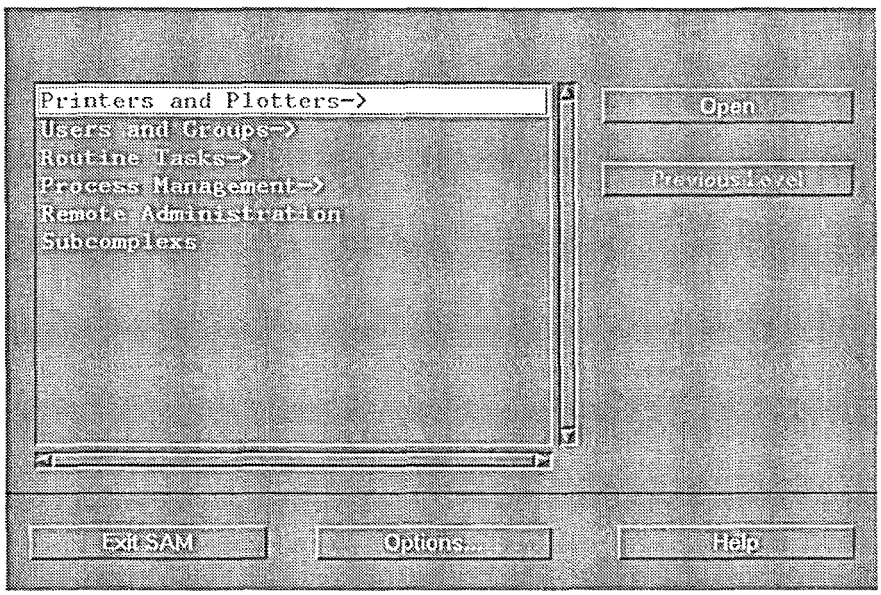

Figure 1 SAM main window

Exiting SAM

To exit SAM, press the Exit SAM button in the lower left corner of the SAM main window. If you are in a different SAM window, first select the Exit option from the List menu in that window to get to the SAM main window.

If a popup window or error message box is displayed, you must first close the window or box before you can perform actions in the SAM windows.

A sample SAM session: Creating a new user account

To add a new user to your system complete the following steps.

Step 1 Log in to your Exemplar system, set your DISPLAY environment variable, and start a SAM session:

```
rlogin exemplar -l root
setenv DISPLAY myhost:0.0
/usr/bin/sam
```

Step 2 Select Users and Groups from the SAM main window.

Do this either by left-clicking on the Users and Groups option and then left-clicking on the Open button, or by double-left-clicking on the Users and Groups option.

Selecting the Users and Groups option displays the Users and Groups window, as shown in Figure 2.

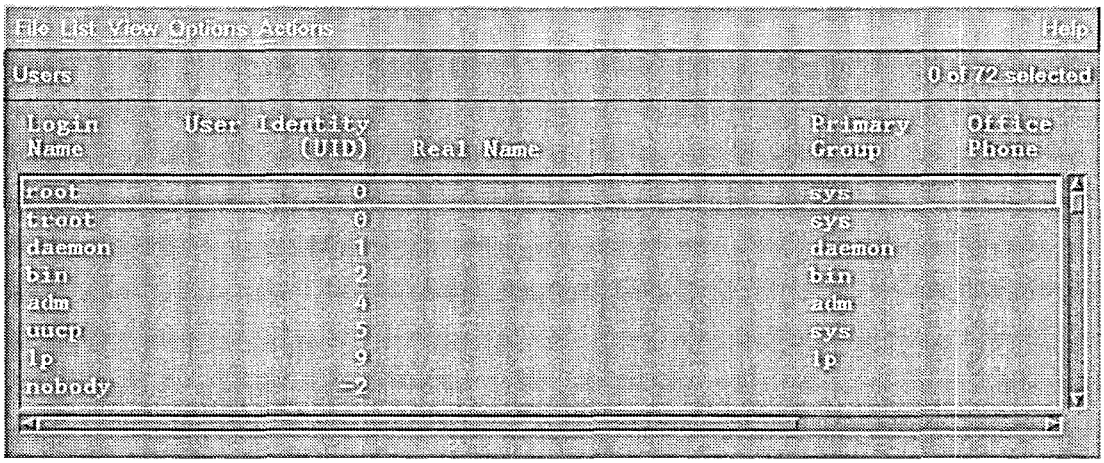

Figure 2 SAM Users and Groups window

Step 3 Select the Add option from the Actions menu.

The Add a User Account window appears, as shown in Figure 3.

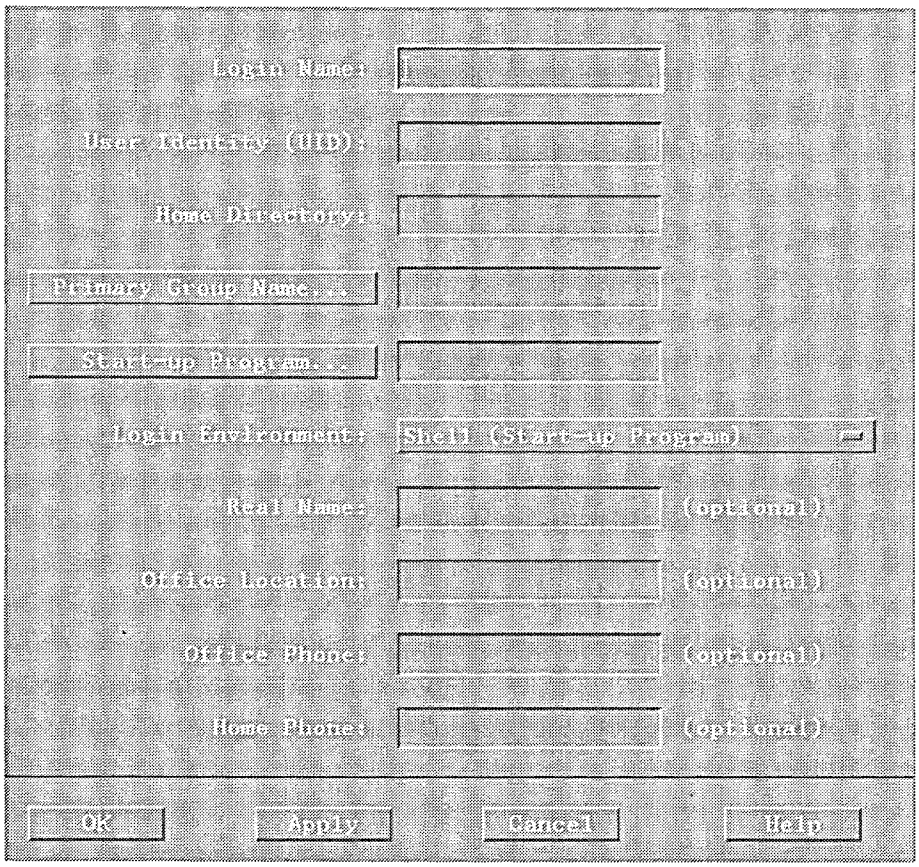

Figure 3 SAM Add a User Account window

Some data entry fields have buttons either to the left of the field or in the field itself. Pressing the button allows you to choose from a list of options for the data entry field.

If the button is in the data entry field, as is the case with the Login Environment field in the Add a User Account window, you may choose only one of the items in the list shown when you press the button.

If the button is to the left of the data entry field, as is the case with the Primary Group Name and Start-up Program fields, you can either select one of the items from the button's list or enter a different value.

Step 4 Enter a login name, change any of the default values you wish to change, enter information into the optional fields if you wish, then press the Apply button to create a new account.

All required fields except the Login Name field have default information already supplied.

Step 5 Enter a password for the new account.

A popup window asks you to enter a password for the new user or press the OK button to create the account with no initial password. You are prompted to reenter the password to verify it, then press OK to create the account.

A popup message window verifies that the new account has been created; press the OK button to close this window.

The new user account appears immediately in the SAM Users and Groups window.

Step 6 Quit your SAM session by selecting Exit from the List menu of the Users and Groups window; then press the Exit SAM button in the SAM main window.

Getting help in SAM

The SAM GUI contains extensive online help. All windows, with the exception of message boxes and some dialog boxes, have online help available. If the window has a menu bar along the top, select the Help menu from the menu bar. If the window has action buttons but no menu bar, press the Help button.

Configuring an SPP-UX system

2

Installing and upgrading SPP-UX

Your Exemplar system is shipped with a bootable copy of SPP-UX. There are two circumstances that require you to install SPP-UX:

- If your working copy is damaged or destroyed—for example, if the disk containing SPP-UX crashes—you need to perform a scratch installation of the operating system.
- When you receive a new version, you need to upgrade your operating system.

The *Exemplar Software Distribution Notice* provides instructions for performing both an SPP-UX scratch installation and an SPP-UX upgrade.

Installing and upgrading optional software

Use the installation procedures described in the *Exemplar Software Distribution Notice* to install optional Convex software (such as compilers) to your system or to upgrade optional software products to a new release. These procedures can also be used to install software supplied by other vendors.

When adding and updating software, be sure to

- Follow the supplier's directions.
- Use the appropriate installation utility (`sd`, `update`, or `tar`).

Post-installation tasks

Perform the following set of system administration tasks after installing or upgrading SPP-UX.

Common post-installation tasks

Perform the following tasks each time you install a new version of SPP-UX. These steps are detailed in the sections indicated.

Step 1 Supply the information the `set_parms` program requests. The `/etc/set_parms` program runs automatically after a new version of SPP-UX is installed and booted. See "Setting system parameters" on page 12.

Step 2 Update the log file. Log in as root and read the /tmp/update.log file to locate any problems. Follow any instructions you find in this file.

Step 3 Set up networking. Read the networking documentation supplied with SPP-UX, and follow the instructions appropriate for the type of network at your installation. See "Setting up networking" on page 14.

Step 4 Set up man pages. You can preformat man pages and remove the man page source from your system to save disk space. See "Setting up man pages" on page 14.

Step 5 Configure optional software products. Read the release notices for any new or updated optional software products you have installed with the new version of SPP-UX and follow any instructions in the *Installation notes* sections of those release notices.

First-time post-installation tasks

Perform the following tasks after installing SPP-UX on your system for the first time. These steps are detailed in the sections indicated.

Step 1 Set the root password. Change the root (superuser) password from the value that is shipped with the system to the value you use at your installation. See "Setting the root password" on page 14.

Step 2 Add users. Create login accounts for system users. See"Adding users and groups" on page 14.

Step 3 Add user groups. Create user groups for system users. See "Adding users and groups" on page 14.

Step 4 Create and mount file systems onto the root file system. See "Managing your SPP-UX file system" on page 91.

Step 5 Set up and enable system users to send and receive electronic mail. See "Setting up electronic mail" on page 17.

Step 6 Set up a printer spooler. Manage printers using the printer spooling system. See "Setting up the line-printer spooler" on page 18.

Step 7 Set up an electronic news system. See "Setting up news" on page 18.

Step 8 Create and configure an accounting system appropriate for the needs of your installation. See "Setting up system accounting" on page 19.

Step 9 Customize the system. There are a number of system features you can customize, including .

- System startup.
- System login prompt.
- Support for nonstandard terminal types.

See "Customizing the system" on page 19.

Step 10 Create a crashdump partition. Create a raw disk partition to hold crashdump data in the event of a system crash. See "Creating a crashdump partition" on page 246.

Setting system parameters

When you first log on to your SPP-UX system after installing or upgrading SPP-UX, the `/etc/set_parms` program automatically executes. This program

- Adds your Exemplar system to a network.
- Sets the host name.
- Sets the system clock, including
  - Time zone
  - Date
  - Time
- Sets the Internet (IP) address for the system.

If you exit the `set_parms` interactive session without providing all the information requested, your system automatically shuts down and requires a power-on reboot. If you provide the requested information, the system files that use this information are updated immediately.

Adding the system to a network

The `/etc/set_parms` program first asks if you are ready to link your system to a network. To complete this procedure, you must provide the following information:

- The host name for your system.
- The Internet (IP) address for your system.
- The time zone your system uses.
- The system clock time.

Have this information ready when `/etc/set_parms` executes. If you do not, you will need to exit the program and run it again when you have the information.

Setting a host name

You can select any host name containing up to eight characters for your system.

As a convention, you may want to select the same host name for your Exemplar system and your Exemplar test station. If you do this, you must select a base name containing no more than six characters; the characters `_d` are automatically appended to the test station's host name on the network connecting the test station to the Exemplar diagnostic bus, and the test station's host name on the external Ethernet port has the characters `_t` appended to the base host name.

Creating an internet address

You need a unique Internet address for your Exemplar system. This address consists of four integer fields separated by periods. The first three fields are the address of your network's subnet, and the last field is the host address of your system on that subnet.

Setting your system's time zone

The `/etc/set_parms` program sets the time zone for your system based on the information you provide about your geographic location. You are prompted to specify your country, then your state or province, and so on until `/etc/set_parms` has enough information to determine your time zone setting.

The value you select for your system's time zone is copied to the following system files:

- /etc/rc
- /etc/profile
- /etc/csh.login

Setting your system clock

The `/etc/set_parms` program displays the current system clock value—date and time—and allows you to change the setting. If you choose to change the clock setting, you are prompted separately to enter the following information:

- Month
- Day of the month
- Hour—in 24-hour format
- Minutes
- Year

If you change the system clock setting notify your users in advance. Changing the system clock setting can cause problems for some utilities.

Known problems when setting the system clock

Setting the clock backward by even a small amount may cause `make` to behave unexpectedly. For example, `make` may fail to locate files that it should locate when it checks the timestamps for files.

If you set the clock back, `cron` does not run any jobs until the clock catches up to the point from which it was set back. For example, if

you set the clock back from 8:00 to 7:00, `cron` will not run any jobs until the clock again reaches 8:00.

If you set the clock ahead, `cron` attempts to catch up by immediately starting all jobs scheduled to run between the old time and the new time. For example, if you set the clock ahead from 9:00 to 10:00, `cron` immediately starts all jobs scheduled to run between 9:00 and 10:00.

Adding users and groups

If you are setting up your system for the first time, add *users* and *groups* to the system. These are the names of the data structures by which your SPP-UX system recognizes a given person or class of people who use it.

There are SAM screens for setting up users and groups, or you can do it by means of SPP-UX commands. Complete directions are in "Controlling access" on page 51.

Setting the root password

SPP-UX is initially installed with a default root password, which is shipped in a separate document. For security reasons, you should immediately change the root password after installing SPP-UX. Use the `/bin/passwd` command to change the root password.

Caution

Do not forget the root password to your system! You will need to perform a system recovery if the root password is lost.

Setting up networking

All Exemplar systems are shipped with an FDDI network adapter and associated driver software. Some Exemplar systems are equipped with additional network adapters.

Network set up depends on the type of network you're using and whether you're connecting the computer to an existing network or setting up a new one.

For further information about setting up a network, see the *Exemplar Networking Guide.*

Setting up man pages

Every SPP-UX command, system call, library function, and system file is documented in SPP-UX man pages. Perform man page setup by three different methods, you can

- Format all man pages.

- Format only selected man pages.
- Format without using `catman`.

The method you choose depends on your disk space requirements and response time preferences. The following sections explain the benefits of each method and how to perform the steps to setup your man pages.

Formatting all man pages

Formatting all man pages ensures that users get quick response when they call up a man page online; however, the formatted versions take up a considerable amount of disk space—about as much again as the originals from which they are created. Once the pages have been formatted, you can recover disk space by getting rid of the originals.

This is a good method if you have enough disk space to hold both versions of the man pages for as long as it takes to finish formatting them.

If you decide to use this method, enter:

```
/etc/catman
```

Formatting all man pages can take as long as five or six hours, so you might want to run it at a lower priority, in the background, or at a nonpeak time.

Formatting selected man pages

Formatting selected man pages gives you the advantage of quicker access to heavily used sections without incurring the cost in disk space of formatting all sections.

If you decide to use this option, enter:

`/etc/catman` *sections*

where *sections* is one or more of the man page sections. Man page sections are listed in Table 1.

Table 1 Man page sections and topics

Section	Topic
1	User commands
1m	System administration commands
2	System Calls
3	Functions and function libraries
4	File formats
5	Miscellaneous
7	Device special files
8	Local man pages

For example, to preformat only the man pages for user commands and file formats, enter:

```
/etc/catman 1 4
```

Formatting without using catman

Use this method if you can spare some disk space but do not want to use any more than is necessary.

If you do not run `/etc/catman`, SPP-UX formats each man page the first time a user calls it up via the `man` command. The formatted version is added to the appropriate `cat` directory and used in subsequent accesses.

If you use this method, you must make directories to hold the formatted man pages. The following script creates these directories:

```
cd /usr/man
for num in 1 1m 2 3 4 5 7 8 9
do
    mkdir cat$num
done
```

When all man pages have been formatted, you can remove the source files.

Setting up electronic mail

Electronic mail (*email*) can be run by any of these three utilities: `elm`, `mailx`, or `mail`. Each user can use any one of these utilities, but cannot switch between utilities without losing previously queued and received mail.

If your users only exchange messages with each other and not send mail to users on other systems in a network, then you need not do any setup.

The mailer does the initialization needed for each user when the user first invokes the mailer. However, you may want to supply each `mailx` and `elm` user with a customization file, setting up useful defaults. Depending on the mailer, the customization file should be:

`mail`

(none)

`mailx`

`$HOME/.mailrc` (that is, a file named .mailrc in the user's home directory)

In addition, `mailx` uses a system-wide /usr/lib/mailx/mailx.rc defaults file.

`elm`

`$HOME/.elm/elmrc`

If users send and receive mail over a network, set up routing through Internet utilities. To configure Internet utilities, follow the directions in the *Exemplar Networking Guide*. You also need to install the Internet utilities `sendmail` utility.

Setting up the line-printer spooler

You share printers among users via the line-printer spooler, which intercepts print requests, organizes them into a queue, and feeds them to the printer one by one. A printer that has been configured into the line-printer spooler is referred to as a *spooled* printer. Any printer SPP-UX supports can be spooled.

If your system is used by more than one user at any one time, you should spool your printers; if you don't, any listing sent to the printer while another listing is printing will be interleaved with it, garbling both listings.

If your system is part of a network, the line-printer spooler also lets you send print requests to, or receive them from, other computers in the network, allowing you to make the most efficient use of your printers.

Setting up the line-printer spooler is one of the more complicated tasks you need to do at this stage. "Managing printers" on page 127 contains full explanations and directions. If you have not administered an SPP-UX system before, or have not set up a spooled printer before, read the chapter carefully before you begin the task.

Setting up news

The `news` utility allows you to post messages for users to read.

Step 1 Create a `news` item, create a file with your text editor and place it in the directory /usr/news.

Step 2 To make sure users know about `news` items they have not read yet, do the following:

- For Korn and Bourne shell users, edit /etc/profile to include the following statements:

```
if [ -f /usr/bin/news ]
    then news -n #notify if news.
fi
```

- For C shell users, edit /etc/csh.login to include the following:

```
if ( -f /usr/bin/news ) then
    news -n #notify if news.
endif
```

Upon log in, if there are unread news items, users see a message like this:

`news:` *news_filename*

where *news_filename* is the name you gave the file in /usr/news.

Users can enter `news` and the item or items will print on the screen. For more information, see the news(1) man page.

Setting up system accounting

System accounting allows you to:

- Monitor individual users' disk space usage.
- Record logins and logouts.
- Collect data about individual processes, such as memory usage and execution time.
- Charge fees for usage.
- Generate summaries and reports to analyze system performance and bill users for resource consumption.

If you need to set up system accounting see "System accounting" on page 159.

Customizing the system

Customizing the system usually means editing a system file, either to change the way the system behaves in general or to modify the way a particular user interacts with it.

The most important files you can customize are listed in Table 2.

Table 2 System customization files

File	Description
/etc/inittab	Contains information about system run levels with one entry for each terminal.
/etc/rc	Defines actions taken during system startup.
/etc/passwd	Determines who can log into your system. You can add, delete, and modify entries, either by editing the file, or by means of SAM screens under the Users menu. Refer to "Starting and stopping SPP-UX" on page 41 for more information.
/etc/group	Identifies the users that form a group, associates group IDs (GIDs) with group names. For more information see "Starting and stopping SPP-UX" on page 41.

/etc/ttytype	Lists terminal types on your system. Edit this file when you add a new type of terminal or modem to your SPP-UX system. For example:
	`300h console`
	`2397 tty00`
	`2397 tty01`

.exrc	Maps terminal characteristics and sets up key definitions for the `ex` family of SPP-UX editors.
/etc/issue	Determines what a user sees before the login prompt.

Table 2 System customization files —*(continued)*

File	Description
/etc/motd	Contains the message of the day.
/etc/profile, /etc/csh.login	Execute automatically during the login process. The /etc/profile file executes for Bourne, Korn, and restricted shell users. The /etc/csh.login file executes for C shell users.
$HOME files	Files in a user's home directory
~/.profile	Executes each time a user successfully logs in using the Bourne shell, Korn shell, or restricted shell.

~/.kshrc	Korn shell script that supplements actions taken by the .profile file.
	Executes whenever a new Korn shell is spawned, if specified by the following statements in a user's .profile:
	`ENV=$HOME/.kshrc`
	`export ENV`
	The name .kshrc is a convention; whichever file you specify executes.

~/.cshrc	Executes when a new C shell starts.
~/.login	Executes when a C shell user logs in, after .cshrc.

Login prompt

The /etc/issue file contains text that users see immediately before the login prompt. Normally this prompt contains the system (by the host name from /etc/hosts) and the release of SPP-UX. You can edit this file to add any other information you want users to see. For example:

```
Parsec [SPP-UX V3.1 SPP-1000XA] 32 processors
```

Message of the day (/etc/motd)

The message of the day appears each time a user logs in if the system-wide user customization file (/etc/profile for Bourne and Korn shell users or /etc/csh.login file for C shell users) has the following line:

```
cat /etc/motd # message of the day
```

Use /etc/motd to display topical messages. For example:

```
Scheduled power outage in Bldg. 801 8AM-noon
Saturday; power off all equipment in Bldg. 801 when
you leave on Friday.
```

Users' login environments

The following files execute when a Bourne or Korn shell user logs in:

- /etc/profile
- $HOME/.profile

The following files execute when a C shell user logs in:

- /etc/csh.login
- $HOME/.cshrc
- $HOME/.login

/etc/profile and /etc/csh.login contain defaults for variables such as time zone setting, terminal type, search path, and mail and news notification. These can be overridden if necessary in individual users' .profile or .login files.

The .cshrc file in a user's home directory performs additional set-up such as setting aliases (user-defined commands). The .kshrc file performs similar tasks for Korn shell users.

When you add a new user, you may want to place default versions of these files in the user's home directory. Use the following sample files as templates:

- /etc/d.cshrc
- /etc/d.exrc
- /etc/d.login
- /etc/d.profile

If you use SAM to add a user, SAM puts the appropriate files in the home directory for you.

Users' editing environments

Editing the .exrc file in a user's home group enables keyboard features such as the cursor arrow keys and sets other options in the `ex` family of editors, including `vi`.

The .exrc file functions only if the `EXINIT` environment variable is not defined in the /etc/profile or $HOME/.profile files. /etc/d.exrc is a sample .exrc file. You may want to customize the file and provide it to new users as a default.

Setting up nonstandard terminal types

Files in directories under /usr/lib/terminfo enable your users to use a wide variety of terminals.

To set up a user with a nonstandard terminal, complete the following steps:

Step 1 Locate the file that corresponds to the terminal you want to set up in the /usr/lib/terminfo directory. If there is no terminfo file for the terminal you want to add see "Creating a new terminfo file" on page 23 for instructions.

For example, if you want to set up a Wyse 100 terminal, all supported terminals whose names begin with "w" are listed under /usr/lib/terminfo/w.

Enter:

```
ls /usr/lib/terminfo/w
```

Find the entry called `wy100`. This is the terminfo file for the Wyse 100.

Step 2 Find the terminal name in the file.

For example, enter:

```
more /usr/lib/terminfo/w/wy100
```

This produces a screen full of special characters, but near the beginning you will see `wy100|100|wyse 100`. This means you can refer to the Wyse 100 by any of the names `wy100`, `100`, or `wyse 100`.

Step 3 Set the user's `TERM` environment variable in the appropriate login script in their home directory: .profile for a Korn or Bourne shell user, .login for a C shell user.

For example, using Bourne or Korn shell, enter:

```
TERM=wy100
export TERM
```

Using C shell, enter:

```
set TERM wy100
```

The default versions of these scripts prompt the user for the terminal type when he or she logs in, so rather than editing the script, you could simply tell the user to respond with the terminal name, for example:

```
TERM = (hp) wy100
```

Creating a new terminfo file

If there is no terminfo file for the terminal you want to set up, you can create one. The terminfo(4) man page explains the rules for constructing a terminfo file.

You may want to copy an existing terminfo file. In this case, get into the directory containing the file you want to copy and create an ASCII version of the file.

For example, to copy the file /usr/lib/terminfo/w/wy100, do the following:

Step 1 Log in as superuser.

Step 2 Change directories:

```
cd /usr/lib/terminfo/w
```

Step 3 Make an ASCII version of the file:

```
untic wy100 > filename
```

where *filename* is whatever you want to call the new file. Make it similar to the terminal's product name and model if you can.

Step 4 Edit the file to reflect the capabilities of the new terminal.

Change the names of the terminal in the first line. See the terminfo(4) man page for rules for entries.

Step 5 Compile the new terminfo file:

```
tic filename
```

For more information on using the `terminfo` compiler, see the tic(1M) man page.

Using the Subcomplex Manager (SCM) 3

Overview

The Subcomplex Manager (SCM) allows you to

- Configure processor and memory resources.
- View the current allocation of system resources.

The primary unit for allocating computing resources to Exemplar users is the *subcomplex*. A subcomplex is a collection of processors and global memory from one or more nodes of the system. Every system user is authorized to use one or more subcomplexes. Each processor and block of global memory can belong to only one subcomplex. Every process runs within a subcomplex; a process can use only the processors and global memory allocated to that subcomplex.

The *system subcomplex* is a special subcomplex that is created automatically at boot time to run system processes, including `init` and processes spawned by `init`. You can modify the system subcomplex, but leave at least one processor on the root node (node 0) allocated to the system subcomplex. Allocating more processors to the system subcomplex increases the percentage of the system's computing resources devoted to operating system functions; allocating fewer processors to the system subcomplex increases the percentage of the system's computing resources devoted to user programs.

SCM interfaces

The Subcomplex Manager provides both a graphical user interface (GUI) and a command interface. The command interface is useful for loading a system configuration when the system is booted or when the system is reconfigured automatically, such as at scheduled times of the day. The GUI provides a convenient way to view system resources and to create complex configuration files.

Starting the SCM GUI

The Subcomplex Manager GUI runs on any X Windows server that is running X Windows Version X11R5. To start the Subcomplex Manager GUI, follow these steps:

Step 1 Set your `DISPLAY` environment variable to the X Windows server you are using. For example:

```
setenv DISPLAY mydisplay:0.0
```

Step 2 Enter the `scm` command with no command-line options:

```
scm
```

SCM GUI

The Subcomplex Manager GUI allows you to:

- View information about system processor and memory resources and the current organization of subcomplexes.
- Create and save a complex configuration file.
- Load or overlay a complex configuration file into the Subcomplex Manager.
- Apply a complex configuration to the system (system administrators only).

The Subcomplex Manager GUI displays and saves information in a manner similar to a word processor. You can load, edit, and save a complex configuration file in much the same way you would load, edit, and save a text file.

Viewing system resources

The Subcomplex Manager main window displays general information about system subcomplexes and processors. It lists subcomplexes running on the system and shows which processors are allocated to each subcomplex. Figure 4 shows the Subcomplex Manager main window.

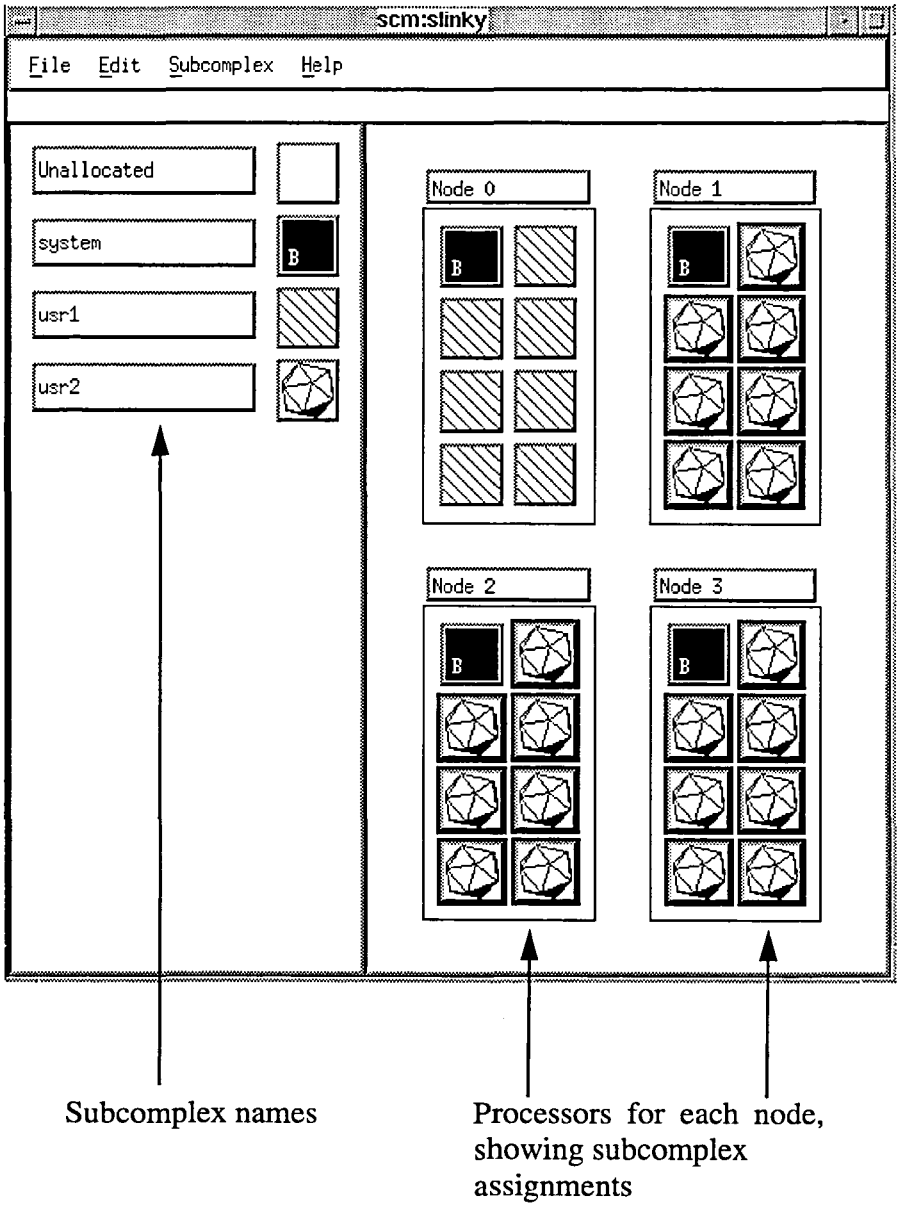

Figure 4 SCM main window

The main window initially displays the current system configuration, including the subcomplexes loaded and the processors allocated to each subcomplex. If you modify any of the information in the main window, you can revert to the current

system configuration display by selecting the Revert option from the File menu.

To view detailed information about a particular subcomplex, left-click on the name of that subcomplex in the subcomplex list on the left side of the main window. This causes the Subcomplex Attributes dialog box to appear. Figure 5 shows the Subcomplex Attributes dialog box.

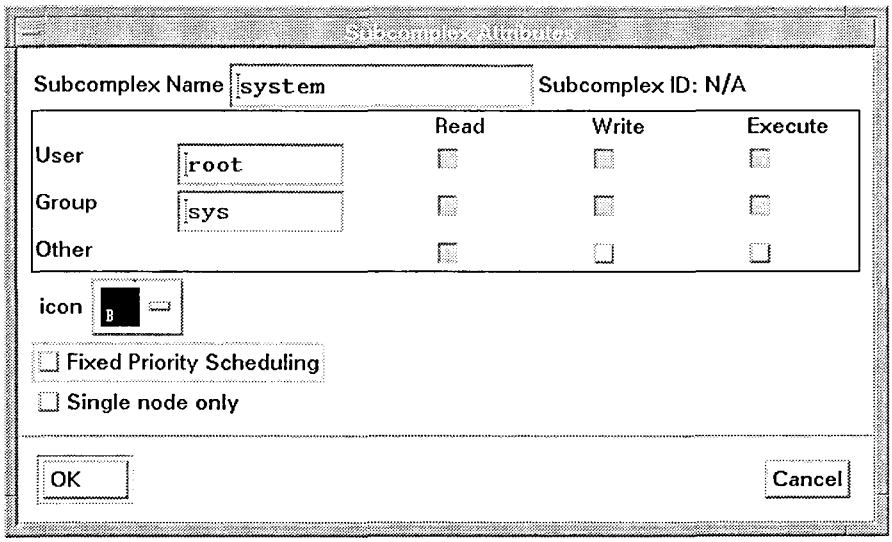

Figure 5 Subcomplex Attributes dialog box

To view detailed information about a particular node, left-click on the name of that node in the node display on the right side of the main window. This causes the Node Attributes dialog box to appear. Figure 6 shows the Node Attributes dialog box.

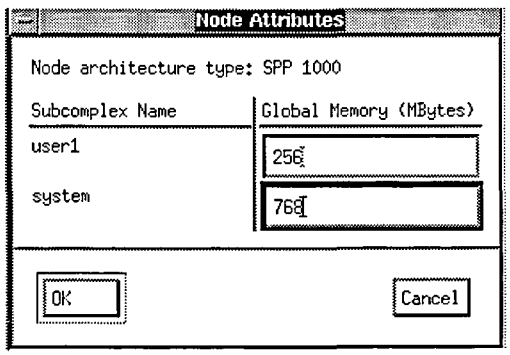

Figure 6 Node Attributes dialog box

Creating and saving a configuration file

A complex configuration file contains a set of specifications for allocating processors and memory to subcomplexes. To create and save a complex configuration file, change the settings in the Subcomplex Manager and save these settings to a file by selecting the Save option or the Save As option from the File menu.

Loading a configuration file

You can load a previously saved complex configuration file into the Subcomplex Manager and then modify that configuration or apply it to the system. To load a complex configuration file, select the Load option from the File menu; then specify the name of the complex configuration file you wish to load.

Overlaying a configuration file

You can overlay a previously saved complex configuration file onto the contents of the Subcomplex Manager main window and then modify that configuration or apply it to the system. To overlay a complex configuration file, select the Overlay option from the File menu and specify the name of the complex configuration file. The contents of the configuration file are merged with the contents of the Subcomplex Manager main window. This operation succeeds only if no processor is specified as allocated in both the configuration file and the window.

System reconfiguration

To apply the configuration currently displayed in the Subcomplex Manager to your system, select the Perform Reconfiguration option from the File menu.

The Subcomplex Manager returns the following information when you select the Perform Reconfiguration option:

- A summary of all changes made to the system.
- A list of subcomplexes idled before the new configuration can be applied. This list appears only if you attempt to remove all processors for a subcomplex from a node; in this case, you must kill all processes running in that subcomplex.
- If the configuration cannot be applied to the system, the Subcomplex Manager lists the reasons. For example, if you attempt to remove all processors from the system

subcomplex, the Subcomplex Manager informs you that the configuration cannot be applied to the system.

If the configuration can be applied to the system, you can kill the necessary processes, then perform the reconfiguration operation.

SCM windows

The Subcomplex Manager GUI consists of the Subcomplex Manager main window, the Node Attributes dialog box, and the Subcomplex Attributes dialog box.

SCM main window

The main window displays a list of subcomplexes on the left side and a representation of all nodes and processors in the system on the right side. This window displays the current complex configuration when you start the interface unless you specified the `-n` option in the `scm` command; you can then use this window as a work area to create a modified complex configuration.

The Subcomplex Manager main window consists of the subcomplex list, the node display, and the menu bar.

Subcomplex list

The subcomplex list appears on the left side of the window by default and lists subcomplexes used in the current configuration, along with an icon representing each subcomplex. You can add and delete subcomplexes from this list. You can perform the following actions:

- View and edit detailed information about a subcomplex.

  Select a subcomplex name to activate the Subcomplex Attributes dialog box.

- Assign all chosen processors to a subcomplex.

  Select a subcomplex icon to assign all processors currently chosen in the Node display to the that subcomplex.

Node display

The node display on the right side of the window shows the processors on each node of the system and the subcomplex assignment of each processor. Processors are displayed in rows of two; the processors in each row share a single CPU agent board. You can perform the following actions on this display:

- View and edit detailed information about the node.

Select a node name to activate the Node Attributes dialog box.

- Assign a set of processor to a subcomplex.

  Highlight a set of processors and select the subcomplex's icon from the Subcomplex list.
- Assign a set of processors to a nodes' server set—if the Server set list is currently displayed in the left side of the window.
- Remove a set of processors from a node's server set—if the Server set list is currently displayed in the left side of the window.

Select a single processor by left-clicking on it. Select a contiguous group of processors by rubber-banding—outlining a rectangle that includes all of the processors while holding down your left mouse button. Normally, selecting a processor or group of processors deselects all other processors; if you hold down your Shift key while selecting a processor or group of processors, any processors already selected remain selected.

Menu bar

The menu bar along the top of the window allows you to perform a variety of functions on the contents of the Subcomplex Manager main window. The menu bar contains the following pull-down menus:

File menu

New

Clears all subcomplexes from the window, allowing you to create a new machine configuration from scratch.

Load

Prompts you for the name of a configuration file and replaces the contents of the window with the contents of the configuration file.

Overlay

Prompts you for the name of a configuration file and merges the contents of the window with the contents of the configuration file. This operation only succeeds if no processor is specified as allocated in both the configuration file and the window.

Save

Saves the configuration currently displayed to the current file name.

Save As

Prompts you for a file name, and saves the configuration currently displayed to that file.

Perform reconfiguration

Reconfigures the machine using the configuration displayed in the window.

Revert

Replaces the contents of the window with the current machine configuration.

Quit

Exits the Subcomplex Manager.

Edit menu

Select All

Selects all processors on all nodes in the Node display portion of the window.

Subcomplex menu

Create

Activates the Subcomplex Attributes dialog box containing information for a new subcomplex; you can edit this information and then select Ok to create a new subcomplex.

Remove

Activates a popup dialog box that allows you to select a subcomplex to remove from the Subcomplex list and the Node display.

Help menu

General

Provides an overview of Subcomplex Manager windows and enables you to search for text strings.

Functions

Provides a brief description of actions you can perform with the Subcomplex Manager.

Windows

Provides an overview of Subcomplex Manager windows and dialog boxes.

Node Attributes dialog box

Selecting a node from the Node display portion of the Subcomplex Manager window pops up the Node Attributes dialog box. The dialog shows each node's

- Architecture type.
- Designated global memory, in MBytes, for all associated subcomplexs.

To allocate global memory enter the amount of the node's global memory, in MBytes, you want to allocate to each listed subcomplex. When you have finished changing values, select the Ok button to make the changes or select the Cancel button to cancel the changes and exit the dialog.

Subcomplex Attributes dialog box

Selecting a subcomplex from the subcomplex list portion of the Subcomplex Manager window pops up the Subcomplex Attributes dialog box. The dialog displays information about a single subcomplex, including

- Subcomplex name.
- Subcomplex ID.
- Ownership.
- Read, write, and execute permissions.
- Scheduling policy.
- Pixmap icon.

You can change the following subcomplex attributes:

Subcomplex Name

A unique name identifying the subcomplex. Subcomplex names are limited to 32 characters.

User

Owner of the subcomplex. This value must be a valid user login ID. This value is used in conjunction with permission information to control access to the subcomplex. Use the user ID root in this field if you wish to limit ownership to system administrators.

Group

User group that can use the subcomplex. This value must be a valid group ID to which the subcomplex owner belongs. This value is used in conjunction with permission information to control access to the subcomplex; generally, this is the group

of users allowed to run processes on the subcomplex—indicated by execute permission.

Permissions

Read, write, and execute permissions for the user, group, and world, or other. Read permission indicates that the user or group may view the subcomplex definition. Write permission indicates that the user or group may modify the set of enabled scheduling policies for the subcomplex. Execute permission indicates that the user or group may run processes on the subcomplex.

Icon

Click on the icon shown to view a list from which to choose an icon to represent this subcomplex. The default icon is a black square with a numeral n in the lower left corner indicating that this is the *n*th subcomplex defined in the current machine configuration.

Fixed Priority Scheduling

Select this field to enable fixed priority scheduling.

Single node only

Restricts the subcomplex to a single node. This allows the operating system to use local memory when an application asks for global memory. A subcomplex with this attribute can not have `GMEM` configured.

Reconfiguration constraints

Some constraints apply when you reconfigure the complex while in use or when making changes to a subcomplex currently loaded on the complex—even if it is not yet in use. A subcomplex with one or more user processes currently assigned to it is "busy."

The constraints are as follows:

- You cannot remove a busy subcomplex from the complex without first killing all user processes on that subcomplex. The Subcomplex Manager will not kill processes; you must do this explicitly using the `kill` utility.
- You cannot reconfigure the global memory of a subcomplex once any global memory has been allocated.
- If a subcomplex already has global memory allocated, you cannot add new processors *unless* they belong to one of the nodes that is already being used by the subcomplex. This constraint applies even if the subcomplex is not currently busy.

- The system subcomplex
  - Can never be removed from the complex.
  - Must have at least one processor on the root node (node 0) assigned at all times.

  The system subcomplex is created automatically on the root node when the complex is booted.
- A subcomplex with the Single node only attribute enabled can not have global memory configured.

The following reconfiguration actions *are* allowed at any time on a busy subcomplex:

- Adding processors to a subcomplex—except if global memory has been allocated.
- Removing processors from a subcomplex—except if global memory has been allocated.

  Removing all processors for a subcomplex from a node suspends all threads running in that subcomplex.
- Changing the following subcomplex attributes:
  - Name.
  - UID.
  - GID.
  - Permissions.
- Enabling or disabling scheduling policies.

Global memory regions are contiguous in physical memory. It is not possible to create global memory for a subcomplex on a node if a sufficiently large contiguous region of free memory does not exist on the node. Creation of a large global memory region is most likely to succeed immediately after booting the system.

SCM command-line interface

The `scm` command-line interface allows you to manage subcomplexes in a noninteractive mode. If no options are specified, or if only the `-n` option is specified, the GUI is activated. If you specify any other option, the command is run without the GUI. The `scm` command has the syntax:

`scm [-n` *nnodes* `| -c | -l` *filename* `| -o` *filename* `| -r` *scname* `| -sc]`

where

`-c`

Returns a full description of the current system configuration. If you are not running with superuser privileges, only information about subcomplexes for which you have read access is returned.

`-l` *filename*

Loads the complex configuration from *filename* to the system Requires superuser privileges.

`-n` *nnodes*

Performs all operations on a hypothetical machine with *nnodes* nodes. This option disables the `-l`, `-o`, and `-r` options and their equivalent functions in the GUI.

`-o` *filename*

Overlays the complex configuration from *filename* to the system. Requires superuser privileges.

`-r` *scname*

Removes subcomplex *scname* from the system. Requires superuser privileges.

`-s`

Returns a list of the names of all subcomplexes currently loaded on the system. Only those complexes for which you have read access are displayed.

`scm` command examples

To see a description of the current machine configuration, log in as the superuser and enter:

```
scm -s
```

To load the complex configuration file weekendconfig onto the system, log in as the superuser and enter:

```
scm -l weekendconfig
```

To create a machine configuration file for a hypothetical system containing 16 nodes (128 processors) using the `scm` GUI, enter:

```
scm -n 16
```

SCM configuration file format

The SCM configuration file is created automatically by the SCM GUI. This file can be stored anywhere in the file system. You can also be create or edit the file manually.

The configuration file consists of a list of subcomplex definitions. All text following a subcomplex SC keyword, up to the next SC keyword, describes that subcomplex.

All statements in the configuration file have the form:

`KEYWORD=`*value*

Except for the SC and PIXMAP keywords, *value* must be an integer value.

The SCM configuration file can contain the following keywords:

SC

The name of the subcomplex. All statements following this statement, up to the next SC statement, are part of this subcomplex's definition. The value must be a character string from 1 to 32 characters long.

UID

The UID of the owner of the subcomplex. This value, along with the subcomplex permissions, determines the actions the owner can perform on the subcomplex. Generally, the owner is the only user with write permission to the subcomplex. The default is the UID of the user who invokes the scm command.

GID

The GID of the group that uses the subcomplex. This value, along with the subcomplex permissions, determines the actions the group can perform on the subcomplex. Generally, this group has execute permission (permission to execute processes on the subcomplex). The default is the effective GID of the user who invokes the scm command.

PERM

The read, write, and execute permissions for the subcomplex owner, user group, and the world. The value must be a four-digit octal number; the first digit should be 0. The remaining digits represent the permissions for the owner, group, and world, respectively; the values are the same as those for the chmod(1) command. The default value is 0754 (read, write, and execute permission for the owner, read and execute permission for the group, and read permission for the world).

POLICY

The scheduling policy for this subcomplex. The values for this keyword are defined in /usr/include/sys/cnx_tattr.h

PIXMAP

The name of the file containing the pixmap for the icon to be used in the SCM GUI for this subcomplex. The default is to let SCM generate a unique icon for the subcomplex.

NODEID

The physical node ID of one of the nodes that has processors allocated to this subcomplex. The value can range from 0 through one less than the number of nodes in the system.

PROCID

The physical processor ID for one of the processors on the node specified in the previous NODEID entry. The value can range from 0 through 7.

SERVERSET

A value of 0 indicates that the processor specified in the previous PROCID entry is not in the server set; a value of 1 indicates that the processor is in the server set. The default value is 1.

GMEM

The amount, in megabytes, of global memory allocated to this subcomplex on the node specified in the previous NODEID entry. The default value is 0.

SINGLE_NODE_ONLY

Restricts the subcomplex to a single node. This allows the os to use local memory when an application asks for global memory. A subcomplex with this attribute enabled can not have GMEM configured. A value of 1 restricts subcomplex memory access to a single node, 0 utilizes global memory. The default value is 1.

NETCACHE

The amount, in megabytes, of Coherent Toroidal Interconnect (CTI) cache to request for this subcomplex on the node specified in the previous NODEID entry. The value must not exceed the amount of global memory present on the node. The default value is 0.

DEFAULT_GM

The default amount, in megabytes, of global memory allocated to processors on the node specified in the previous NODEID entry if they are added in the future. The default value is 0.

DEFAULT_NC

The default amount, in megabytes, of CTI cache to request for this subcomplex on nodes added to the subcomplex in the future. The default value is 0.

Example subcomplex configuration file

The following example is a definition of a subcomplex named zymurgy. This subcomplex contains processors 6 and 7 from nodes 0 and 1. 128 MB of global memory is allocated to this subcomplex; both nodes have 16 MB of CTI cache. The two

processors on node 0 remain in their node's server set; the two processors on node 1 are not in their node's server set. The subcomplex is owned by the user with UID 26645, and can be used by the user group with GID 524.

```
NODEID=0: CTICACHE=16:
NODEID=1: CTICACHE=16:
SC=zymurgy:
  UID=26645: GID=524:
  NODEID=0:
          GMEM=128:
          NETCACHE=16:
          PROCID=6:
          PROCID=7:
  NODEID=1:
          GMEM=128:
          NETCACHE=16:
          PROCID=6: SERVERSET=0:
          PROCID=7: SERVERSET=0:
```

Starting and stopping SPP-UX

4

Introduction

Starting and stopping SPP-UX are routine tasks. When the system is turned on, you can allow the default operating system to boot or select other boot options. When you stop a system, you must use the appropriate shutdown process. Turning the system off with the power switch can corrupt the file system. When you change the system to an administrative or single-user state, during shutdown, you can reboot or restart the system without turning it off, or you can shut the system down completely.

The following sections describe how to start SPP-UX if you are starting with a system that has everything connected but not turned on.

Turning on the system

Before turning on the Exemplar system you must,

- Turn on the Exemplar test station.
- Set up a working environment on the test station.
- Perform all the test station installation instructions provided in the *Exemplar Software Distribution Notice* shipped with your Exemplar software.

When you turn on the Exemplar system, system diagnostics are run before SPP-UX is booted. When the SPP-UX microkernel boots, a System Console window appears on the Exemplar test station. You can begin to control SPP-UX at this point.

During the boot sequence, use the **ESC** key to terminate or interrupt the current process. The Open Boot software controls the boot process until the entire operating system is finished loading.

To run the Exemplar system in single-user mode, press the **ESC** key, then enter the following command in the System Console window:

```
boot -s
```

If you do not interrupt the boot process, the SPP-UX operating system automatically boots to multiuser mode. See the *Exemplar Open Boot Quick Reference* for other operations you can perform while the system is booting and for information on how to configure the boot process on your system.

Watch the startup messages. Compare what starts up with what you expect, and note possible problems. The exact messages depend on your configuration. The startup process ends when you see the login prompt in the System Console window. During the startup process, the system performs a file system consistency check of the root disk if the system was shut down improperly.

Starting SPP-UX

You must start up, or boot, SPP-UX when the operating system has been completely shut down or after you have partially shut down the operating system to perform system administration tasks.

To ensure your system starts properly

- Reset the system. Reset the system by turning on the main power. The operating system begins to boot, and-if this is the first time that the system has been booted-prompts you for initial system parameters. See "Setting system parameters" section on page 12 for details.

  Some SAM tasks and SPP-UX commands may restart (reboot) the system for you (for example, /etc/reboot).
- Properly configure and install the hardware and the software.
- Ensure that you have thhhe files needed for proper start up, including
  - /etc/init
  - /etc/inittab
  - /dev/console
  - /etc/bcheckrc
  - /etc/brc
  - /etc/rc

Without these files, the startup process fails. Open Boot hands off control to the /etc/init process, which sequentially executes the contents of /etc/inittab. The /etc/bcheckrc, /etc/brc, and /etc/rc files are scripts specified in the /etc/inittab that must be

executed to check the file system and initialize your system. See the "Reviewing the state of the file system" section on page 43 for details. Errors found in the file system might require you to perform additional tasks.

The operating system continues to boot up and displays additional information about your system. After the bootup process completes, a login prompt appears in the System Console window.

You are now ready to use your SPP-UX system.

Reviewing the state of the file system

During the startup process, the /etc/bcheckrc script executes `/etc/fsclean`. This command determines the shutdown status of the system and returns three possibilities:

1. Proper file system shutdown—The startup process continues and you see the following message:

```
/etc/fsclean:/dev/dsk/0s0(root device) ok file
system is OK, not running fsck
```

2. Improper file system shutdown—The startup process is interrupted and you see:

```
/etc/fsclean:/dev/dsk/0s0 not ok run fsck FILE
SYSTEM(S) NOT PROPERLY SHUTDOWN, BEGINNING FILE
SYSTEM REPAIR.
```

   At this point, the system runs `/etc/fsck` in a mode that corrects certain inconsistencies in the file systems without your intervention and without removing data. The `fsck` command does one of the following:

   - Repairs and reboots the system, incorporating the changes.
   - Prompts you to run the `fsck` command manually. If you need to run `fsck` manually, see "Managing your SPP-UX file system," on page 91.

3. Other errors detected—For example, not being able to open a specified device file, you get an error message. The startup process ends, and you need to solve the problem.

Stopping SPP-UX

Typically, you shut down the system to:

- Put it in single-user state so you can
  - Update the system.
  - Check the file systems.
- Turn it off to perform a task such as installing a new disk drive.

Caution

Never stop the system by turning off the power. Stopping the system improperly can corrupt your file systems.

Shutdown considerations

The following list outlines some of the considerations for shutting down the system:

- Only the system administrator or a designated superuser can shut down the system.
- You can use SAM to shut down the system.
- The `/etc/shutdown` command
  - Warns all users to log off the system, using a grace period you specify.
  - Halts daemons.
  - Kills unauthorized processes.
  - Unmounts file systems.
  - Puts the system in single user mode.
  - Writes the contents of the I/O buffers to a disk.
- Do not run `shutdown` from a remote system via `rlogin` if you use a network service. The shutdown process logs you out prematurely and returns control to the system console. Only run `shutdown` from the System Console window on the Exemplar test station.

See the shutdown(1M) man page for information about options and features.

Using shutdown(1M)

The `shutdown` command warns all users to log off the system, using a grace period you specify.

If you do not specify a grace period, users get 60 seconds to log off. You should notify active users when the system will be shut down. Give them enough time to finish their work and log off. You

can do this physically or use the `/etc/wall` or `/etc/cwall` commands.

The following examples show alternatives for shutting down to the single-user state:

`shutdown`

Shuts down to single-user state, allowing the default 60-second grace period.

`shutdown 0`

Shuts down the system with no grace period.

`shutdown 30`

Begins the shutdown to the single-user state after a 30-second grace period.

As always, watch the messages to see that everything is happening correctly.

System shutdown authorization

You can designate which users are authorized to run `shutdown` by listing these users in the file /etc/shutdown.allow. If this file is empty, only the superuser has shutdown authority. If this file is not empty, and the superuser login—usually `root`—is not listed in the file, the superuser can not shut down the system. When users are listed in the shutdown.allow file, only the listed users have shutdown authority.

Customizing the shutdown process

You can customize the shutdown process by placing Bourne shell scripts in the file /etc/shutdown.d. These scripts execute in an ASCII (machine-collated) order. The scripts are optional, and are not required to run `shutdown`.

Power failure considerations

A *local power failure* means a power failure that halts the computer by affecting its central bus.

A *remote power failure* (affecting a remote bus) or device power failures (affecting a device) do not affect the system as a whole, unless the remote devices provide a vital system resource.

Power failure-related tasks

If you know power is going out soon, shut down the computer and turn off the power.

If a local power fail occurs, turn off all computer equipment affected by a power failure until power is completely restored. An electrical surge, such as when power is coming back on, could seriously damage hardware that has been left turned on.

Going to single-user mode

You must be a system administrator with superuser capabilities or a designated user with superuser capabilities to shutdown the system.

Step 1 Ensure you have superuser capabilities.

Step 2 Change to the root directory.

```
cd /
```

Step 3 Shut down the system. Enter

```
shutdown
```

Shuts down to single-user state. If you do not specify a grace period, in seconds, users get 60 seconds to log off. You should notify active users when the system will be shut down. Give them enough time to finish their work and log off. You can do this physically or use the `/etc/wall` or `/etc/cwall` commands.

To learn more about the shutdown(1M) command see "Using shutdown(1M)" on page 44.

Step 4 Perform the necessary system administration tasks while the system is in single-user state.

Step 5 Start up the system without turning off anything by executing:

```
/etc/reboot
```

Shutting down the system completely

You must be a system administrator with superuser capabilities or a designated user with superuser capabilities to shut down the system.

Stopping the system improperly can corrupt or damage the file systems. Never stop the system by turning off the power.

Step 1 Ensure you have superuser capabilities.

Step 2 Change to the root directory.

```
cd /
```

Step 3 Shut down the system. Execute

```
shutdown 20
```

This puts the system into single-user state, allowing a 20-second grace period.

The shutdown process asks if you want to send a message. If you elect to broadcast a message, respond with `y` and then type the message. When you finish, press **RETURN** (or **ENTER**), and then **CTRL-d**.

Step 4 Bring the system to a complete halt. Execute

```
reboot -h
```

You see several messages during the process. Watch them to note actions and possible problems. The system is shut down completely when the system displays `halted` and pressing a key in the System Console window causes no response.

Step 5 When the system is halted, turn the system power off.

Note

From the multiuser state, you can also shut down the system completely by executing

```
shutdown -h
```

This process is more harsh and sudden than the multistep procedure described above.

Using SAM to halt or reboot the system

You can use SAM to reboot or shut down your Exemplar system. From the SAM main window, perform the following steps:

Step 1 Start SAM.

Enter

```
/usr/bin/sam
```

Step 2 From the SAM main window, select Routine Tasks and then select the Open button.

Step 3 Select System Shutdown from the Routine Tasks display and then select the Open button.

The window shown in Figure 7 appears.

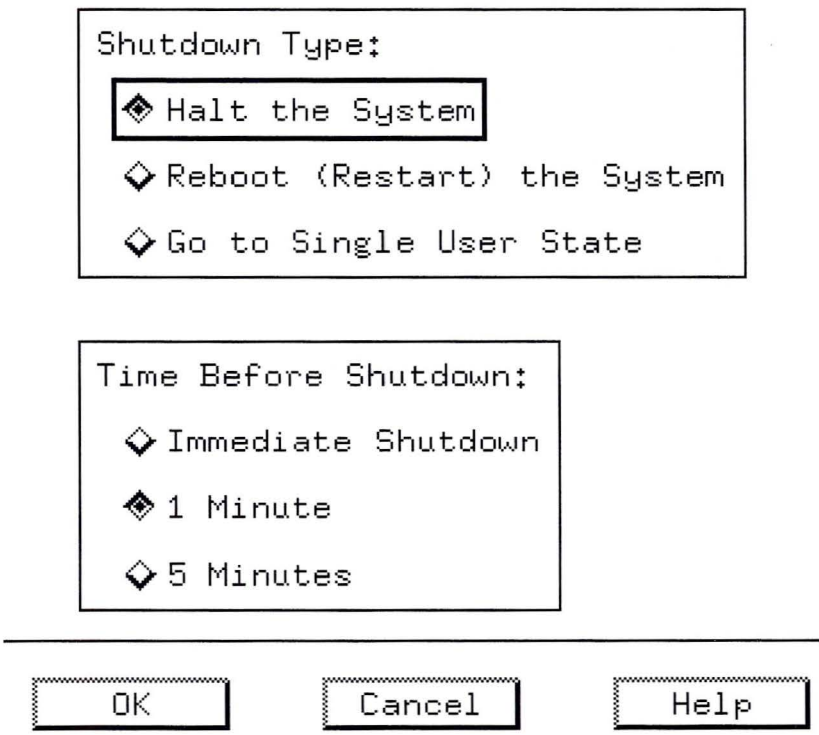

Figure 7 SAM System Shutdown window

Step 4 Select the Shutdown Type and designate a Time Before Shutdown.

Select a button to halt the system, reboot the system, or bring the system to single-user state. The window allows you to specify a time delay before the operation is performed.

Press the OK button after making a selection. SAM presents you with a confirmation dialog box.

Step 5 SAM prompts you for a message to broadcast to all system users. Enter a message and confirm the operation. The shutdown procedure begins after the message is sent to all system users and the designated waiting period has elapsed.

Controlling access 5

Types of control

Securing your data against deliberate, unauthorized access is only one reason for controlling access to your system. Three levels of access control to your system are

- Controlling user account
- Controlling file access
- Controlling run-levels

Controlling user accounts and groups

Each user is defined by an entry in the file /etc/passwd. The `/usr/bin/vipw` command is the recommended editor for modifying the /etc/passwd file. The `vipw` command guarantees exclusive access to the /etc/passwd file. The `vipw`, `chsh`, `chfn`, and `passwd` commands create the /etc/ptmp file when granting access to the /etc/passwd file. You must set the `EDITOR` environment variable to `vi` to use `vipw`.

If you are in single-user mode and the `vipw` command denies access to the /etc/passwd file with an error message, "password file busy," delete the /etc/ptmp file and try the `vipw` command again. It is possible that the process that created the ptmp file terminated without removing it.

See the vipw(1M), chsh(1), chfn(1), passwd(1), and passwd(4) man pages for additional information.

Users on your system can be divided into various working groups, so that files owned by members of a given group can be shared and yet protected from access by users who are not members of the group. A user can be a member of more than one group. A group can have a maximum of 200 members.

If you prefer not to divide the users of your system into separate working groups, set up one group—usually called `users`—and assign all of your system users to that group.

Users may change their current group by using the newgrp command. The new group is referred to as the *effective group* for the user. Changing to an effective group does not alter the user's primary group entry in the /etc/passwd file. The user can return to his or her primary group by specifying no parameters or options to the `newgrp` command.

Group information is defined in /etc/group and /etc/logingroup, ASCII files that you edit with a text editor.

Each entry in the /etc/group file defines a group's

- Group name
- Encrypted password (optional)
- Numerical group identifier (group_ID)
- Group members—In a comma-separated list of by user login name

The following sample shows the contents of a /etc/group file.

```
root:*:0:
other:*:1:
bin:*:2:
sys:*:3:
adm:*:4:
daemon:*:5:
mail:*:6:
lp:*:7: users:*:20:john,naomil,patrickd,kerschen
pub:*:24:patrickd,naomil,dennism
```

If a blank line appears in the /etc/group file, all entries after the blank line are ignored.

The /etc/logingroup file contains identical information, except for the group name and encrypted password fields. It is common practice to link the /etc/group and /etc/logingroup files together using the link(1M)command.

The newgrp command uses the /etc/group file to check access privileges. If the user's login name appears in the access list of the group for which access is being requested, access is granted, changing the user's current group to the requested group. /etc/logingroup, in contrast to /etc/group, allows users listed in more than one group access to files belonging to other groups without changing their primary or effective groups.

For additional information about group related tasks, see the "Managing user accounts and group tasks" section on page 56. For more details on the /etc/group and /etc/logingroup files, see

the "Adding a group" section on page 62, and the group(4) man page.

Primary groups

A user can be a member of multiple groups, but only one of those groups is considered to be the user's primary or default group. In addition to being listed as a member of the group in the file /etc/group, an entry exists in the file /etc/passwd, indicating which group is the user's primary group. When users first log in to the system, they are affiliated with their primary group.

To change a user's primary, or default, group membership, change the user's entry in the /etc/passwd file to reflect a new *group_ID* value. The group_ID uniquely identifies an entry in /etc/group and /etc/logingroup. For instructions, see "Displaying and modifying a user's account" section on page 61, and "Changing a user's primary group" section on page 82.

Group passwords

When a user first logs in to your system, the default, or primary, group affiliation is designated by their group_ID entry—the fourth field in /etc/passwd. A user may be a member of more than one group. To change the group a user is affiliated with, a user can use the `newgrp` command. `newgrp` requires a password if the group has a password and the user does not, or if the group has a password and the user is not listed as being a member of that group in the file /etc/group. If the user only needs to access files in another group, an entry in the /etc/logingroup permits access to other group's files without changing the user's effective group. See the group(4) man page for more information.

Caution

Do not use group passwords. They encourage poor security practices.

Controlling file access

All files on an SPP-UX system have access permissions and group ownership associated with them. Together the permissions and ownerships determine who can access the file.

File and ownership access permissions

There are three types (modes) of file access:

read

Determines who can view the file's contents. For directories, read access allows access to the directory with the cd command.

write

Determines who can alter the file's contents. For directories, write access allows modify and remove privileges.

execute

If the file is an executable program, execute permissions determine who can run the program. For directories, execute access allows listing the directory contents.

There are three sources of file access:

Owner

The owner is usually the person who created the file (unless ownership has since been changed using the `chown` command).

Group

Members of the group to which the that the file belongs.

Other

All other users on your system.

Changing access privileges

Three commands change file access privileges:

See the chmod(1), chown(1), and chgrp(1) man pages for additional information.

`chmod`

The `chmod` command changes the type of access (read, write, and execute privileges) for every access source (owner, group, or other). For example, you can give the owner of the file read, write, and execute privileges, restrict group members to read and execute, and give only execute privileges to all other users on the system. Only the owner of a file (or the superuser) can change its read, write, and execute privileges.

`chown`

The `chown` command changes file ownership. In order to change the owner, you must own the file or have superuser privileges.

`chgrp`

The `chgrp` command changes file group ownership. In order to change the group, you must own the file or have superuser privileges.

The system assigns default file permissions—governed by your `umask`—whenever you create a new file or directory. Unless set up otherwise by you or your system administrator, your default `umask` setting is 0, which means that new files you create have read/write permission for everyone (666 or -rw-rw-rw-) and new directories you create have read/write/search permission for everyone (777 or drwxrwxrwx).

For more information see the ll(1), setprivgrp(1M), and umask(1) man pages.

Controlling run-levels

A *run-level* is an SPP-UX state of operation in which a specific set of processes—and their offspring—is permitted to run. This set of processes is defined for each run-level in a file called /etc/inittab.

Run-level `2` is the normal operating mode (often called *multiuser mode*). Special processes called `gettys` run in this mode and post the login prompt on your system's terminals. When users log in to your system, the `gettys` initiate other processes (usually SPP-UX shells), which in turn allow still other processes to be executed as users enter commands.

A special run-level called run-level `s` is also defined. Run-level `s` is a special system administration mode, called *single-user mode.* Use it to perform special tasks when you have no one else on the system.

Run levels `s` and `2` are predefined. You can create new run-levels or change which processes can run at these predefined run-levels, if your needs require. You can define up to six run-levels (1-6). Most systems do not need to define additional run-levels, and modifications to the predefined run-level `2` usually allow `getty` processes to run on terminals you add to the system.

To create a new run-level, make (or change) entries in the /etc/inittab file that define how you want the system to operate when the system is in that run-level. For information on the /etc/inittab file, see the inittab(4) man page.

When you use SAM to add terminals to your system, SAM makes the entries in the file /etc/inittab for you.

Only the superuser can use the `init` command, which changes the system from one run level to another, but anyone with write

permission to the file /etc/inittab can create new run-levels or redefine existing run-levels.

To protect your system from tampering, ensure that the permissions and group ownership of the /etc/init and /etc/inittab files are:

```
-r-xr-xr-x root other /etc/init
-r--r--r-- root root /etc/inittab
```

For more information see

- "Creating a new run-level" on page 87.
- "Changing system run-levels" on page 88.
- "Entering the system administration run-level" on page 89.
- "Returning from the system administration run-level" on page 90.

Managing user accounts and group tasks

There are two ways to perform user account and user group tasks on your system:

- Using the System Administration Manager (SAM)
- Using SPP-UX Commands (editing a series of files and creating user directories)

Generally, you should use SAM because it is simpler and faster than performing the task with commands. For information about running SAM and navigating within SAM, see Chapter 1.

SAM allows you to control access to your system through its menu-selection and data-entry screens. By combining multiple "manual commands" into single tasks, SAM can save you time and keystrokes. SAM also eliminates the need to know command names and options.

Although SPP-UX commands require you to learn more details than SAM does, you might need or prefer to use SPP-UX commands, for the following reasons:

- SPP-UX commands give you greater control.
- SAM might not be configured into your system. You have to use SPP-UX commands.
- You might be more comfortable using SPP-UX commands.

The following tasks are covered in this section:

- Adding a user
- Removing a user
- Customizing SAM 'adding and removing a user' capabilities
- Deactivating a user's account

- Reactivating a user's account
- Displaying/modifying a user's account information
- Adding a group
- Removing a group
- Changing a user's primary group
- Adding users to groups
- Removing users from groups

Each task has an ordered list of instructions, an area for additional information if necessary, and specific examples.

Using SAM to control access

If you use SAM to add, remove, or modify users accounts, SAM edits /etc/group for you. You can also use SAM to work directly with groups—add a new group, remove a group, change the list of users in a group. SAM does not edit /etc/logingroup explicitly; but, if you link /etc/group and /etc/logingroup together, changes are reflected in the /etc/logingroup file.

SAM provides an online help system for assistance you when you need additional information.

Adding a user

To add a user to your system using SAM:

Step 1 Ensure that you have superuser capabilities.

Step 2 Run SAM; enter:

```
/usr/bin/sam
```

Step 3 Select Users and Groups, then press the Open control button.

Step 4 Select Users, and press the Open control button.

Step 5 Select Add from the Actions menu.

Step 6 Fill in the Add a User Account window fields and press the Apply control button.

Step 7 After reading the messages, press the OK control button.

To return to the functional area list or functional subarea, choose Exit from the List menu.

Multiple root users with one UID

Even though the user should be unique, there are a few circumstances where it is useful to have several /etc/passwd entries with the same user_ID number. For example, consider the following three /etc/passwd entries:

```
root:9WsbljlTvWbbw:0:3:Root User Account:/:/bin/sh
croot:NPt3HW.jBpVz2:0:3:Root User Account (C-Shell):/:/bin/csh
kroot:dGJbw/DBeDLdo:0:3:Root User Account (K-Shell):/:/bin/ksh
```

On the system with these entries in the passwd file, three separate accounts—`root`, `croot`, `kroot`—have superuser capability. Depending on which one is used, the superuser starts up in the Bourne Shell, the C Shell, or the Korn shell.

Because all three accounts have the user_ID `0`, the system views all three as the same user. When the system checks to see which user owns a file, it compares the user_ID associated with the file against the user_ID entries in the /etc/passwd file. When it does so, it scans the /etc/passwd file from the beginning until it finds a user_ID match. Because `root` is the first of the three entries, files created by the users `croot` and `kroot` are listed as owned by the user `root`.

Removing a user

To remove a user from your system using SAM:

Step 1 Ensure that you have superuser capabilities.

Step 2 Run SAM; enter:

```
/usr/bin/sam
```

Step 3 Select Users and Groups, and press the Open control button.

Step 4 Select Users and press the Open control button.

Step 5 Select Remove from the Actions menu.

Step 6 Turn on the check box associated with the action regarding user's files and press the OK control button.

Step 7 After reading the messages, press the OK control button.

To return to the functional area list or functional subarea, choose Exit from the List menu.

SAM views a user as a specific user_ID—as opposed to a specific login name—and does not remove any user with the same user_ID as the superuser-user_ID 0.

Caution

Never remove the user called "root" from your system. If you have other superusers (users with the user_ID of zero) on your system, you can remove them by removing their entry from the /etc/passwd file.

If SAM detects that another user has the same user_ID, SAM does not remove the user's files.

SAM does not remove system directories, even if they are owned by a given user.

SAM does not remove files across NFS mounts.

SAM does not update the /etc/logingroup file. If you use /etc/logingroup, edit the file to remove the user from all group entries.

Customizing SAM

To customize the procedure for adding and/or removing a user using SAM:

Step 1 Ensure that you have superuser capabilities.

Step 2 Run SAM; enter:

```
/usr/bin/sam
```

Step 3 Select Users and Groups and press the Open control button.

Step 4 Select Users and press the Open control button.

Step 5 Choose Task Customization from the Actions menu.

Step 6 Fill in the script file name to be executed in one or a combination of the before/after adding/removing a user fields.

Step 7 Press the OK control button.

Step 8 After reading the messages, press the OK control button.

Using shell scripts

There often steps specific to your operations that you may want to perform when you add or remove a user on your system. SAM allows you to set up shell scripts or executable programs which it runs for you

- Before adding or removing the user.
- After adding or removing the user.
- Before and after adding or removing the user.

You must follow strict permission and ownership requirements for your custom script:

- The file must be owned by root.
- The file must be writable and executable only by root.
- The file must reside in a directory path where all directories are writable only by root.

Suppose the custom command lies in directory /usr/local/bin. To successfully pass the validation, the permissions on /usr, /usr/local, and /usr/local/bin must all be writable only by the owner "drwxr-xr-x." The permissions cannot be "drwxrwxr-x" or "drwxrwxrwx". This is typically a problem because /usr/local and /usr/local/bin are installed with permissions "drwxrwxrwx."

Take care to locate a directory that meets the requirements—/usr, /usr/bin, /usr/sam, /usr/sam/bin, /usr/sam/config are a few examples of directories that meet the criteria when installed, or create a directory that meets the requirements.

These restrictions apply only at the time of SAM field validation of the command. Once SAM has registered a custom command, the restrictions are no longer checked by SAM for that command unless the user alters the custom script with SAM.

Deactivating a user's account

It is sometimes useful to temporarily suspend a user's ability to log in to your system—such as when the user will be away for an extended period of time. The user's files remain intact on the system. You must reactivate the account when the user returns.

To remove the ability to log into an account, complete the following procedure:

Step 1 Ensure that you have superuser capabilities.

Step 2 Run SAM; enter:

```
/usr/bin/sam
```

Step 3 Select Users and Groups and press the Open control button.

Step 4 Select Users and press the Open control button.

Step 5 Select the user entry in the object list you wish to deactivate.

Step 6 Choose Deactivate from the Actions menu.

Step 7 Turn on the check box for the action regarding the user's files and press the OK control button.

Step 8 After reading the messages, press the OK control button.

To return to the functional area list or functional subarea, choose Exit from the List menu.

Reactivating a user's account

To reactivate a user's account using SAM:

Step 1 Ensure that you have superuser capabilities.

Step 2 Run SAM; enter:

```
/usr/bin/sam
```

Step 3 Select Users and Groups, and press the Open control button.

Step 4 Select Users and press the Open control button.

Step 5 Select the user entry in the object list you wish to reactivate.

Step 6 Choose Reactivate from the Actions menu.

Step 7 Optionally fill in a password for the user and press the OK control button.

Step 8 After reading the messages, press the OK control button.

To return to the functional area list or functional subarea, choose Exit from the List menu.

Displaying and modifying a user's account

To display or modify a user's account information using SAM:

Step 1 Ensure that you have superuser capabilities.

Step 2 Run SAM; enter:

```
/usr/bin/sam
```

Step 3 Select Users and Groups, and press the Open control button.

Step 4 Select Users and press the Open control button.

Step 5 Select the user in the object list.

Step 6 Choose Modify from the Actions menu.

Step 7 View or modify the user's information.

To view the user's information, press the Cancel control button after gathering the information you need.

To modify the user's information, fill in the new information in the Modify a User window and press OK. After reading the messages, press OK. The following user information can be modified:

- Login name (user_name)
- Password
- User identification number (user_ID)
- Primary group identification number (group_ID)
- Comment
- Login directory
- Startup program

To return to the functional area list or functional subarea, choose Exit from the List menu.

Adding a group

To add a new group using SAM:

Step 1 Ensure that you have superuser capabilities.

Step 2 Run SAM; enter:

```
/usr/bin/sam
```

Step 3 Select Users and Groups and press the Open control button.

Step 4 Select Groups and press the Open control button.

Step 5 Choose Add from the Actions menu.

Step 6 Fill in the new group name.

Step 7 (Optional) Select the users of the newly created group.

Step 8 Press the OK control button if this is the only group you are adding. Otherwise, press the Apply and subsequent OK control buttons to return to the Add a Group window.

To return to the functional area list or functional subarea, choose Exit from the List menu.

Removing a group

To remove a group using SAM:

Step 1 Ensure that you have superuser capabilities.

Step 2 Run SAM; enter:

```
/usr/bin/sam
```

Step 3 Select Users and Groups, and press the Open control button.

Step 4 Select Groups and press the Open control button.

Step 5 Choose Remove from the Actions menu.

You can assign the group's files to another group if desired. Otherwise, SAM reassigns the group's files to the primary group of each of the file's owner.

Step 6 After reading the messages, press the OK control button.

To return to the functional area list or functional subarea, choose Exit from the List menu.

Adding and removing users from groups

To modify a group's membership list using SAM:

Step 1 Ensure that you have superuser capabilities.

Step 2 Run SAM; enter:

```
/usr/bin/sam
```

Step 3 Select Users and Groups, and press the Open control button.

Step 4 Select Groups and press the Open control button.

Step 5 Choose Modify from the Actions menu.

Step 6 Follow the instructions displayed in the Modify a Group window.

Step 7 Press the OK control button.

To return to the functional area list or functional subarea, choose Exit from the List menu.

Using SPP-UX commands to control access

Generally, you should use the SAM method because it is simpler and faster than performing the task with SPP-UX commands. For information about running SAM and navigating within SAM, see "Introduction" on page 1.

Each task has an ordered list of instructions, an area for additional information if necessary, and specific examples. In some of the SPP-UX commands method examples the `xargs` command is used with the `find` command. Output from `find` is piped to `xargs` instead of using the `-exec` *primary* option to the `find` command. The xargs(1) command collects filenames or directory names into multiple arguments to a single `chgrp` or `chown` command, resulting in fewer processes and greater system efficiency. Specify the full pathname to the command following `xargs` to guarantee expected command behavior. See the find(1) and xargs(1) man pages for additional information.

Adding a User

The following list summarizes the steps you take to add a user to your system using SPP-UX commands:

1. Log in as root.
2. Make a backup copy of the /etc/passwd file.
3. Use the `/etc/vipw` command to update the /etc/passwd file to include the users' entries. The `/etc/vipw` command requires the `EDITOR` environment variable to be set to `vi`.
4. Create a login directory for each of the new users.
5. Check the /etc/passwd file format with `pwck`.
6. Copy local initialization files to each user's login directory.
7. Create or customize initialization files for the users.
8. Change the ownership and permissions of all of the files and subdirectories created in the login directory using the `chown` and `chgrp` commands.
9. Update the /etc/group file, and optionally, the /etc/logingroup file to add the user's login name to their primary group member list.
10. Check the /etc/group file format with `grpck`.
11. Instruct the new user to log in.

The following procedure explains the steps you must follow to add a user to your system in more detail:

Step 1 Ensure that you have superuser capabilities.

Step 2 Make a copy of /etc/passwd so that it is easy to undo any mistakes that you might make

```
cp /etc/passwd /etc/passwd.old
```

If you need to undo a mistake later, you can restore the old contents of the file using the command

```
cp /etc/passwd.old /etc/passwd
```

Step 3 Create an entry in the file /etc/passwd for the new user.

Use the `vipw` command to ensure that you have exclusive access to the /etc/passwd file. You must set the EDITOR environment variable set to `vi` to use `vipw`.

Each user attribute must be colon-separated with no leading or trailing spaces. The one-line entry must be in the form:

user_name:password:UID:GID:comment:login_directory:start_up_program

where:

user_name

Is the name the user enters at the login: prompt. The login name must have the following characteristics:

- It must begin with an alphabetic character.
- It can include up to eight alphanumeric characters.
- It cannot contain blank spaces.
- It cannot already exist on the system.

password

Ensures the user sets a password when they log in for the first time, place the characters ",.." in this field. For example:

```
smith:,..:567:40:Paul Smith:/users/smith:/bin/csh
```

Putting an unencrypted password in the password field does not work. For example, if you want to assign a user the password "secret," the following entry will not allow the user to log in using the password "secret:"

```
smith:secret:567:40:Paul Smith:/users/smith:/bin/csh
```

To set or change a user's password, use the `/bin/passwd` command.

To leave the new user's account without a password, do not put any characters between the colon (:) separators. For example:

```
smith::567:40:Paul Smith:/users/smith:/bin/csh
```

Caution

Do not leave an account unprotected (without a password), even for a short period of time. If you enter ",.." in the password field, have the user log in as soon as possible to set a password for the account.

UID

Is a unique integer value the system uses to identify the user. If the *user_ID* is 0 (zero), that user has superuser capabilities. The *user_ID* "0" is associated with the user root.

By convention, the values 1 through 99 are reserved for system use. When you add a new user to your system, pick any unused number greater than 99 but less than 60000 for this field. *user_ID*s greater than 59999 are invalid.

GID

Is the user's primary group GID, or *group_ID*, as defined in the third field of the user's primary group entry in the /etc/group file. The *group_ID* is an integer value shared by all members of the same group. See"Adding a group" section on page 78 for a description of the /etc/group and /etc/logingroup file formats.

comment

Logs information about the identity of the user or to identify the entry. Although this field is free-format, use the following comma-separated subfield format:

```
User's Full Name, Office Location, Office Phone, Home Phone
```

login_directory

Is the absolute pathname of the directory the user is placed in upon initial log in to the system. The directory need not exist when the entry to /etc/passwd is made. However, the directory must exist before the user can log in. This field can be no longer than 63 characters.

start_up_program

Contains the name of a single command to be executed for the user when logging in; it should be an absolute pathname for the command. This field can be no longer than 44 characters. Typically this is the name of a shell—`/bin/sh`, `/bin/csh`, `/bin/ksh`, etc. However, it can be any executable program or command. The command can be either a compiled program or a shell script, but no arguments to the command or script should be supplied. If the command field is left blank, `/bin/sh` executes by default.

When the user logs in, the command listed in this field executes and passes control to that program. Once the program terminates, the user is logged out.

Step 4 Create a login or home directory with the `mkdir` command:

`/bin/mkdir [-p] [-m` *mode*`]` *directory*

where:

`-p`

Specifies intermediate directories are created as necessary. Otherwise, the full path prefix of directory must already exist. The `mkdir` command requires write permission in the parent directory.

`-m` *mode*

Specifies creating the directory as specified by *directory* with the file permissions are set to *mode*, which is a symbolic mode string as defined for chmod (see chmod(1)). The umask(1) has precedence over -m.

directory

Specifies the user login directory.

The login directory is the directory the user is placed in upon initial log in to the system.

The login directory defined for a user in the /etc/passwd file must exist when the user logs into the system or the login attempt fails.

It is not necessary for each user to have a separate login directory. It is, however, easier to keep the files of the various users separated if each has their own login directory. This, in turn, makes it easier to work with a given user's files—for example when doing backups, determining how much disk space a given user is using, etc.

Step 5 Use the `pwck` command to verify that the /etc/passwd file has valid entries:

`/etc/pwck`

The `pwck` command validates the number of fields, login name, user ID, group ID, and whether the login directory and optional program name exist. See the pwck(1M) man page for additional information.

Step 6 Create login shell initialization files.

In most cases, the *start_up_program* is one of the SPP-UX shells—`/bin/sh`, `/bin/csh`, `/bin/ksh`, `/bin/rsh`, etc. Each of these shells has a set of initialization files it reads when a user logs

in to the system. Table 3 lists the initialization files executed for each shell in the order executed.

Table 3 SPP-UX shell initialization files

Shell	Initialization files executed at login
/bin/ksh /bin/rksh /bin/rsh /bin/sh	/etc/profile $HOME/.profile
/bin/csh /bin/tcsh	/etc/csh.login $HOME/.login $HOME/.cshrc

The /etc/profile and /etc/csh.login files contain global instructions that you want executed for every user who logs into the system. These files are in the /etc directory so that they are accessible to all users. Do *not* copy them to each user's directory.

Local initialization files are located in the user's login directory ($HOME). These local files typically contain shell commands and environment variable definitions that customize the user's environment or automatically run one or more programs for the user.

If local initialization files exist in a user's directory, the shell executes the commands in the local files after completing the global files, but before the user receives the first shell prompt.

Examples of the local initialization files are located in the /etc directory. Their names begin with the character d—for example, d.profile. You may copy these files to a user's login directory and customize them. When you copy these files to the user's login directory, rename the files without the d prefix.

Step 7 Create or customize other initialization files for the user.

Other programs such as text editor (vi, emacs) and the various mail programs (mail, elm, mailx) have initialization files which you may also want to set up.

You may need to change the access permissions of files within the user's login directory. See the ll(1), chmod(1), chown(1), chgrp(1), and umask(1) man pages for additional information.

Step 8 Change the file ownership of the new user's home directory and the files to the user's login name with the chown command.

You must change the ownership of the new files to that of your new user so that the new user can access them.

The `chown` command has the following syntax:

`/bin/chown` [**`-R`**] *new_owner login_dir*

where:

`-R`

Specifies to recursively change the file ownership to *new_owner*. For each *login_dir*, the owner of the directory and all files and subdirectories in the file hierarchy below it are changed to *new_owner*.

new_owner

Specifies the login name of the new user.

login_dir

Specifies the login directory of the new user.

Step 9 Change the group ownership of the new user's home directory and files to the user's primary group with the `chgrp` command.

You must change the group ownership to that of your new user so that the new user can access them.

The `chgrp` command has the following syntax:

`/bin/chgrp` [**`-R`**] *new_group login_dir*

where:

`-R`

Specifies to recursively change the group ownership to *new_group*. For each *login_dir*, the group of the directory and all files and subdirectories in the file hierarchy below it are changed to *new_group*.

new_group

Specifies the primary group of the new user.

login_dir

Specifies the login directory of the new user.

Step 10 Edit the /etc/group file, and optionally, the /etc/logingroup file to add the user's *user_name* to the names in the comma-separated list of members for the group. If you want the user to be able to access files belonging to another group other than their primary group without using the `chgrp` command, edit the /etc/logingroup file to add the user to all necessary groups.

If a blank line appears in the files, all entries after the blank line are ignored.

A group can have a maximum limit of 200 users.

Step 11 Use the `grpck` command to check for inconsistencies and verify all entries in the /etc/group and /etc/logingroup files. This verification includes a check of the number of fields, group name, group ID, and whether all login names appear in the password file. The `grpck` command has the following format:

`/etc/grpck [`*group_file*`]`

where *group_file* is the name of the group file to be checked. The default group file is /etc/group.

Step 12 Have the new user log in to the system so that you can verify that you have set up his (or her) environment correctly.

Multiple users with the same UID

Even though the user should be unique, there are a few circumstances where it is useful to have several /etc/passwd entries with the same user_ID number. For example, consider the following three /etc/passwd entries:

```
root:9WsbljlTvWbbw:0:3:Root User Account:/:/bin/sh
croot:NPt3HW.jBpVz2:0:3:Root User Account (C-Shell):/:/bin/csh
kroot:dGJbw/DBeDLdo:0:3:Root User Account (K-Shell):/:/bin/ksh
```

On the system with these entries in the passwd file, three separate accounts—`root`, `croot`, `kroot`—have superuser capability. Depending on which one is used, the superuser starts up in the Bourne Shell, the C Shell, or the Korn shell.

Because all three accounts have the user_ID `0`, the system views all three as the same user. When the system checks to see which user owns a file, it compares the user_ID associated with the file against the user_ID entries in the /etc/passwd file. When it does so, it scans the /etc/passwd file from the beginning until it finds a user_ID match. Because `root` is the first of the three entries, files created by the users `croot` and `kroot` are listed as owned by the user `root`.

Getting user information

If the comment field information in the /etc/passwd file is entered in a comma-separated subfield format, you can use the `/usr/bin/finger` command to display this information. If you need to modify the user's comment field information, you can modify the /etc/passwd file directly with the `vipw` command or you can use the `/usr/bin/chfn` command. The information in the comment field is referred to as "gecos" information. Refer to the finger(1) and chfn(1) man pages for additional information.

If you have several users sharing a login directory, the ownerships and permissions of the shell local initialization files may need to be adjusted so that all users sharing that login directory have read access to the local initialization files. There are several ways to do this. One way is to assign one of the users sharing the login directory to be the owner of the files, have all of the users sharing the directory be members of the same group, and give the group members read access to the files. For additional information on file protection, see the ll(1), chmod(1), chown(1), chgrp(1), setprivgrp(1M), and umask(1) man pages.

Removing a user

To remove a user from your system using SPP-UX commands:

Step 1 Ensure that you have superuser capabilities.

Step 2 Decide the future of the user's files and directories. Your choices are:

- Remove the user's files and directories from the system.
- Reassign the user's files and directories to another user.
- A combination of removal and reassignment of the users files.

Step 3 Determine which files on your system are owned by the user you are removing with the `find` command. Enter:

```
/bin/find / -fsonly hfs -user user_name -print
```

where *user_name* is the user's login name as defined in the user's /etc/passwd file entry.

Files, especially executable programs, can be shared among users in a group or among all users of the system. If you decide to remove the user's files, be sure that they are not be needed by other users of your system. See the find(1) man page for additional information.

Step 4 Search for and remove the user's login name from all entries in the /etc/group file.

Use `grep` or the search command in your text editor to find the entries in /etc/group that contain the user's login name.

Step 5 (Optional) Remove the user's login name from all entries in the /etc/logingroup file if it exists.

Use `grep` or the search command in your text editor to find the entries in /etc/group that contain the user's login name.

Step 6 Check for inconsistencies and verify all entries in the /etc/group and /etc/logingroup files with the `grpck` command.

This verification includes a check of the number of fields, group name, group ID, and whether all login names appear in the password file. The `grpck` command has the following format:

```
/etc/grpck [group_file]
```

where *group_file* is the name of the group file to be checked. The default group file is /etc/group.

Step 7 Make a copy of the /etc/passwd file so that it is easy to undo any mistakes that you might make:

```
cp /etc/passwd /etc/passwd.old
```

If you need to undo a mistake later, you can restore the old contents of the file using the command:

```
cp /etc/passwd.old /etc/passwd
```

Step 8 Remove the user's entry in the /etc/passwd file using the `vipw` command.

Step 9 Verify that the /etc/passwd file has valid entries with `pwck`:

```
/etc/pwck
```

The `pwck` command validates the number of fields, login name, user ID, group ID, and whether the login directory and optional program name exist. See the pwck(1M) man page for additional information.

Caution

Never remove the user called "root" from your system.

Example

To remove user michelem from the system:

Step 1 Log in as root.

Step 2 Change file ownership of all files in the login directory for michelem to rykl and remove all of the files on the rest of the system owned by user michelem.

Change file ownership of all files and directories in the /users/michelem login directory to rykl:

```
xargs chown -R rykl
```

Remove files owned by michelem from the system:

```
xargs rm
```

Remove all of the empty directories owned by user michelem:

```
xargs rmdir
```

Step 3 Check the /etc/group file format:

```
/etc/grpck
```

Step 4 Make a backup copy of the /etc/passwd file.

```
cp /etc/passwd /etc/passwd.old
```

Step 5 Update the /etc/passwd file to delete the line containing the information for user michelem with the `/etc/vipw` command. If your `EDITOR` environment variable is not set to `vi` use one of the following commands:

For the Korn and Bourne shell `EDITOR` environment variable, enter:

```
export EDITOR=vi
```

For the C shell `EDITOR` environment variable, enter:

```
setenv EDITOR vi
```

Step 6 Check the /etc/passwd file format:

```
pwck
```

Deactivating a user's account

Sometimes it is useful to temporarily suspend a user's ability to log in to your system (such as when the user will be away for an extended period of time). The user's files remain on the system and intact, ready for the user when they return and you reactivate their account.

Deactivating a user's account means to make it so that no login password is valid.

To deactivate a user's account using SPP-UX commands:

Step 1 Ensure that you have superuser capabilities.

Step 2 Make a copy of the /etc/passwd file so that it is easy to undo any mistakes that you might make:

```
cp /etc/passwd /etc/passwd.old
```

If you need to undo a mistake later, you can restore the "old" contents of the file using the command:

```
cp /etc/passwd.old /etc/passwd
```

Step 3 Edit the /etc/passwd file using the `vipw` command and complete the following steps:

1. Locate the entry in the /etc/passwd file that corresponds to the user's account that you are planning to deactivate.
2. Replace the encrypted password in the second field with an asterisk (*).
3. Save the /etc/passwd file and exit the editor.

Step 4 Use the `pwck` command to verify that the /etc/passwd file has valid entries:

```
/etc/pwck
```

The `pwck` command validates the number of fields, login name, user ID, group ID, and whether the login directory and optional program name exist. See the pwck(1M) man page for additional information.

Caution

Never deactivate the user called root. This destroys the ability to perform necessary system administration tasks on SPP-UX and usually requires a scratch installation of the operating system.

Example

To deactivate the user account for paul, edit the /etc/passwd file. Use the `/etc/vipw` command. The `/etc/vipw` command requires the `EDITOR` environment variable set to `vi`.

To set the Korn and Bourne shell `EDITOR` environment variable, enter:

```
export EDITOR=vi
```

To set the C shell `EDITOR` environment variable, enter:

```
setenv EDITOR vi
```

The /etc/passwd file entry before deactivating user paul:

```
paul:sIgNXHLuFptVE:209:20:Paul Avonette, Mailstop F13, 555-7086,:/users/paul:/bin/ksh
```

The /etc/passwd file entry after deactivating user paul:

```
paul:*:209:20:Paul Avonette,Mailstop F13,555-7086,:/users/paul:/bin/ksh
```

Check the /etc/passwd file format:

```
pwck
```

Reactivating a user's account

To reactivate a user's account using SPP-UX commands:

Step 1 Ensure that you have superuser capabilities.

Step 2 Make a copy of the /etc/passwd file so that it is easy to undo any mistakes that you might make:

```
cp /etc/passwd /etc/passwd.old
```

If you need to undo a mistake later, you can restore the old contents of the file using the command:

```
cp /etc/passwd.old /etc/passwd
```

Step 3 Edit the /etc/passwd file using the `vipw` command:

1. Locate the entry in the /etc/passwd file that corresponds to the user's account that you are planning to reactivate.
2. Force the user to set a new password for the account upon log in by replacing the asterisk (*) in the second field of the file with the string ",.." (comma-dot-dot).
3. Save the /etc/passwd file and exit the editor.

Step 4 Use the `pwck` command to verify that the /etc/passwd file has valid entries:

```
/etc/pwck
```

The `pwck` command validates the number of fields, login name, user ID, group ID, and whether the login directory and optional program name exist. See the pwck(1M) man page; manual for additional information.

Displaying a user's account information

To display a user's account information use either of the following methods:

- Show the user's /etc/passwd file information with the `finger` command.

  /usr/bin/finger *user_name*

  where *user_name* is the user's login name as defined in the user's /etc/passwd file entry.

- Show the user's /etc/passwd file information with the `grep` command.

  /bin/grep *user_name* **/etc/passwd**

  where *user_name* is the user's login name as defined in the user's /etc/passwd file entry.

Modifying a user's account information

To modify a user's account information:

Step 1 Ensure that you have superuser capabilities.

Step 2 Make a copy of the /etc/passwd file so that it is easy to undo any mistakes that you might make:

```
cp /etc/passwd /etc/passwd.old
```

If you need to undo a mistake later, you can restore the old contents of the file using the command:

```
cp /etc/passwd.old /etc/passwd
```

Step 3 Edit the /etc/passwd file using the `vipw` command to update the following user information:

- Login name (user_name)
- Password
- User identification number (user_ID)
- Primary group identification number (group_ID)
- Comment
- Login directory
- Start up program

Step 4 Use the `pwck` command to verify that the /etc/passwd file has valid entries:

```
/etc/pwck
```

The `pwck` command validates the number of fields, login name, user ID, group ID, and whether the login directory and optional program name exist. See the pwck(1M) man page for additional information.

Examples

To display account information for user jdoe:

```
finger jdoe
```

```
Login name: jdoe In real life: John Doe
Bldg: Building 5
Directory: /users/jdoe Shell: /bin/ksh
On since Feb 10 11:17:04 on pty/ttys5 from
mountian.net.ca
2 minutes 25 seconds Idle Time
No Plan.
```

Instead of `finger`, you can use the `grep` command:

```
grep jdoe /etc/passwd
```

```
jdoe:QAJZL4Xjg/BMM:1667:20:John Doe,Building 5,555-1234: \
/users/jdoe:/bin/ksh
```

To update the telephone number for user jdoe with the `chfn` command:

```
chfn jdoe
```

```
Default values are printed inside of of '[]'.
To accept the default, type .
To have a blank entry, type the word 'none'.

Name [John Doe]:
Location (Ex: 42U-J4) [Building 5]
Office Phone (Ex: 1632) [1234]: 2233
Home Phone (Ex: 5555678) []:
```

Run `pwck` to check your /etc/passwd file format.

Adding a group

The following list summarizes the steps you take to add a group to your system using SPP-UX commands

1. Log in as root.
2. Make a backup copy of /etc/group and /etc/logingroup:
3. Create an entry in the /etc/group file for the new group.
4. Check the /etc/group file format with `grpck`.
5. (Optional) Edit the /etc/logingroup file to enable users access to files belonging to other groups without changing effective groups.
6. (Optional) Check the /etc/logingroup file format with `grpck`.

The following procedure explains the steps you must follow to add a group to your system in more detail:

Step 1 Ensure that you have superuser capabilities.

Step 2 Make a copy of the /etc/group file and the /etc/logingroup file, so that it is easy to undo any mistakes that you might make

```
cp /etc/group /etc/group.old
```

```
cp /etc/logingroup /etc/logingroup.old
```

If you need to undo a mistake later, you can restore the old contents of the files using the commands:

```
cp /etc/group.old /etc/group
```

```
cp /etc/logingroup.old /etc/logingroup
```

Step 3 Create an entry for the new group in the /etc/group file.

The /etc/group entries must have the following format:

group_name:group_password:group_ID:user1[,user2][,user3]...

where:

group_name

Is the name of your new group. It must begin with an alphabetic character and can include up to 16 alphanumeric characters.

group_password

Is the encrypted password for the group. This field is not used. It is recommended that you put an asterisk (*) in this field, which indicates that you will not be using group passwords.

group_ID

Is a unique integer, which is used by SPP-UX to identify the group.

user1,...

Is a list of comma-separated user names (from the first field of the entries in the /etc/passwd file).

Step 4 (Optional) Create an entry in the /etc/logingroup file to allow access to files belonging to other groups without changing the users' effective group.

The /etc/logingroup entries must have the following format:

group_name:group_password:group_ID:user1[,user2][,user3]...

where:

group_name

Is the name of your new group. It must begin with an alphabetic character and can include up to 16 alphanumeric characters.

group_password

Is the encrypted password for the group. This field is not used. It is recommended that you put an asterisk (*) in this field, which indicates that you will not be using group passwords.

group_ID

Is a unique integer, which is used by SPP-UX to identify the group.

user1,...

Is a list of comma-separated user names (from the first field of the entries in the /etc/passwd file).

Step 5 Check for inconsistencies and verify all entries in the /etc/group file, and optionally the /etc/logingroup file, with the `grpck` command.

This verification includes a check of the number of fields, group name, group ID, and whether all login names appear in the password file. The `grpck` command has the following format:

`/etc/grpck` [*group_file*]

where *group_file* is the name of the group file to be checked. The default group file is /etc/group.

Group file considerations

The /etc/group file is used by the `newgrp` command to check access privileges when a user attempts to change their effective group. If the login name appears in the access list of the group for which access is being requested, access is granted, changing the user's current group to the requested group.

The /etc/logingroup file—in contrast to /etc/group—allows users listed in more than one group to access files belonging to other groups that they are members of without changing their primary group.

A blank line in the /etc/group or /etc/logingroup file is not allowed. If a blank line appears in the files, all entries after the blank line are ignored.

A group can have a maximum limit of 200 users.

Removing a group

To remove a group from your system using SPP-UX commands use the following instructions:

Step 1 Ensure that you have superuser capabilities.

Step 2 Make a copy of the /etc/group file, and optionally the /etc/logingroup file, so that it is easy to undo any mistakes that you might make:

```
cp /etc/group /etc/group.old
```

```
cp /etc/logingroup /etc/logingroup.old
```

If you need to undo a mistake later, you can restore the old contents of the files using the commands:

```
cp /etc/group.old /etc/group
```

```
cp /etc/logingroup.old /etc/logingroup
```

Step 3 Make a copy of the /etc/passwd file so that it is easy to undo any mistakes that you might make:

```
cp /etc/passwd /etc/passwd.old
```

If you need to undo a mistake later, you can restore the old contents of the file using the command:

```
cp /etc/passwd.old /etc/passwd
```

Step 4 Use the `find` command to globally search the system for files with particular file and group ownership:

`/bin/find / -fsonly hfs -user` *user_name* **`-group`** \
group_name **`-depth -print`**

If you are globally reassigning file or group ownership of all files and directories, use the `find` and `xargs` commands, and the command to be executed globally (`rm`, `rmdir`, `chown`, or `chgrp`). For example:

`/bin/find / -fsonly hfs -user` *user_name* **`-group`** \
group_name **`-depth -print | xargs chgrp`** *new_group*

If you are not globally removing or changing file access permissions, use the `find` separately from the `rm`, `rmdir`, `chown`, or `chgrp` command.

Step 5 (Optional)Edit the /etc/passwd file to remove user entries if you are removing users from the system.

Step 6 Verify that the /etc/passwd file has valid entries with `pwck`.

```
/etc/pwck
```

The `pwck` command validates the number of fields, login name, user ID, group ID, and whether the login directory and optional program name exist. See the pwck(1M) man page for additional information.

Step 7 Edit the /etc/group file, and optionally the /etc/logingroup file, to remove the group entry for the group you are removing. If users are being removed from the system, remove the user from any other groups.

Step 8 Check for inconsistencies and verify all entries in the /etc/group and /etc/logingroup files with `grpck`

This verification includes a check of the number of fields, group name, group ID, and whether all login names appear in the /etc/passwd file. The `grpck` command has the following format:

`/etc/grpck` [*group_file*]

where *group_file* is the name of the group file to be checked. The default group file is /etc/group.

If a blank line appears in the /etc/group or /etc/logingroup files, all entries after the blank line are ignored. Refer to "Using SPP-UX commands to control access" on page 64 for a description of the /etc/group and /etc/logingroup files. See also the group(4) man page.

Changing a user's primary group

To change a user's primary group using SPP-UX commands:

Step 1 Ensure that you have superuser capabilities.

Step 2 Make a copy of the /etc/passwd file so that it is easy to undo any mistakes that you might make:

```
cp /etc/passwd /etc/passwd.old
```

If you need to undo a mistake later, you can restore the old contents of the file using the command:

```
cp /etc/passwd.old /etc/passwd
```

Step 3 Determine the *group_ID* number of your user's new primary group by looking at the new primary group's entry in the /etc/group file. The third colon (:) separated field of the group entry in the /etc/group file contains the *group_ID*.

Step 4 Edit the /etc/passwd file using the `vipw` command to replace the primary group ID, *pri_group_ID* (fourth field), of your user's entry with the new primary group ID.

Step 5 Verify that the /etc/passwd file has valid entries with `pwck`.

```
/etc/pwck
```

The pwck command validates the number of fields, login name, user ID, group ID, and whether the login directory and optional program name exist. See the pwck(1M) man page for additional information.

Step 6 Make a copy of the /etc/group file, and optionally the /etc/logingroup file, so that it is easy to undo any mistakes that you might make:

```
cp /etc/group /etc/group.old

cp /etc/logingroup /etc/logingroup.old
```

If you need to undo a mistake later, you can restore the old contents of the files using the commands:

```
cp /etc/group.old /etc/group

cp /etc/logingroup.old /etc/logingroup
```

Step 7 Edit the /etc/group file to add the user's login name to the entry which corresponds to the new primary group. If you do not want the user to continue to be a member of their previous primary group, remove the user's login name from the list of user members of their previous primary group.

If a blank line appears in the files, all entries after the blank line are ignored.

A group can have a maximum limit of 200 users.

Step 8 Use the grpck command to check for inconsistencies and verify all entries in the /etc/group and /etc/logingroup files. This verification includes a check of the number of fields, group name, group ID, and whether all login names appear in the /etc/passwd file.

The grpck command has the following format:

/etc/grpck [*group_file*]

where *group_file* is the name of the group file to be checked. The default group file is /etc/group.

Adding users to groups

The following list summarizes the steps you take to add uses to groups using SPP-UX commands:

1. Log in as root.
2. Make a copy of the /etc/group file. Optionally, make a copy of the /etc/logingroup file.

3. Edit the /etc/group file to add the user names to the group member lists. Optionally, edit the /etc/logingroup file.
4. Check the /etc/group file format with `grpck`. Optionally, run `grpck` on the /etc/logingroup file.

The following procedure explains in more detail the steps you must follow to add users to groups.

Step 1 Ensure that you have superuser capabilities.

Step 2 Make a copy of the /etc/group file, and optionally the /etc/logingroup file, so that it is easy to undo any mistakes that you might make:

```
cp /etc/group /etc/group.old

cp /etc/logingroup /etc/logingroup.old
```

If you need to undo a mistake later, you can restore the old contents of the files using the commands:

```
cp /etc/group.old /etc/group

cp /etc/logingroup.old /etc/logingroup
```

Step 3 Edit the /etc/group file, and optionally the /etc/logingroup file, to add the user's *user_name* to the names in the comma-separated list of members for the group(s) for which they are to be members. If you want the user to be able to access files belonging to group other than their primary group without using the `chgrp` command, edit the /etc/logingroup file to add the user to all necessary groups.

Step 4 Use the `grpck` command to check for inconsistencies and verify all entries in the /etc/group and /etc/logingroup files. This verification includes a check of the number of fields, group name, group ID, and whether all login names appear in the /etc/passwd file. The `grpck` command has the following format:

/etc/grpck [*group_file*]

where *group_file* is the name of the group file to be checked. The default group file is /etc/group.

A blank line in the /etc/group or /etc/logingroup file is not allowed. If a blank line appears in the files, all entries after the blank line are ignored.

A group can have a maximum limit of 200 users.

Removing users from groups

If you are removing users from the system, see "Removing a user" section on page 58. If you are changing the user's primary group, see "Changing a user's primary group" section on page 82. Otherwise, this procedure assumes that you are not removing users from their primary groups.

To remove users from groups using SPP-UX commands:

Step 1 Ensure that you have superuser capabilities

Step 2 Make a copy of the /etc/group file, and optionally the /etc/logingroup file, so that it is easy to undo any mistakes that you might make:

```
cp /etc/group /etc/group.old

cp /etc/logingroup /etc/logingroup.old
```

If you need to undo a mistake later, you can restore the old contents of the files using the commands:

```
cp /etc/group.old /etc/group

cp /etc/logingroup.old /etc/logingroup
```

Step 3 List the files and directories owned by the user with the `find` command. The `find` command has the following syntax:

```
/bin/find / -fsonly hfs -user user_name -group group_name -depth -print
```

where

user_name

Is the login name of the user as defined in the /etc/passwd file.

group_name

Is the group from which the user is being removed.

If the list of files is long you can redirect the output to a file or redirect the output to the `more` command. For example:

```
/bin/find / -fsonly hfs -user user_name -group group_name -depth -print | more
```

Step 4 Edit the /etc/group file, and optionally the /etc/logingroup file, to remove the user's *user_name* from the list of group members.

Do not remove the user's name from the user's primary group defined in the /etc/passwd file unless you are removing the user from the system. If you want to change the user's primary group, see "Changing a user's primary group" section on page 82.

A blank line in the /etc/group or /etc/logingroup file is not allowed. If a blank line appears in the files, all entries after the blank line are ignored.

Step 5 Use the `grpck` command to check for inconsistencies and verify all entries in the /etc/group and /etc/logingroup files. This verification includes a check of the number of fields, group name, group ID, and whether all login names appear in the password file. The `grpck` command has the following format:

`grpck` [*group_file*]

where *group_file* is the name of the group file to be checked. The default group file is /etc/group.

Managing run-levels with SPP-UX commands

A run-level is an SPP-UX state of operation which permits a specific set of processes and their offspring to run. This set of processes is defined for each run-level in a file called /etc/inittab.

This section covers

- Creating a new run-level using SPP-UX commands
- Changing system run-levels using SPP-UX commands
- Entering the system administration run-level
- Returning from the system administration run-level

Creating a new run-level

To create new run-levels using SPP-UX commands use the following instructions:

Step 1 Ensure that you have superuser capabilities.

Step 2 Make a copy of the /etc/inittab file so that it is easy to undo any mistakes that you might make:

```
cp /etc/inittab /etc/inittab.old
```

If you need to undo a mistake later, you can restore the old contents of the file using the command:

```
cp /etc/inittab.old /etc/inittab
```

See the init(1M) and inittab(4) man pages for a detailed description of entries in the /etc/inittab file.

Step 3 Edit the /etc/inittab file to change the `initdefault` entry in your test version to "s."

By changing the initdefault entry to "s," your system boots up in run-level "s". To change to run-level 2 after booting, execute `init 2`. If your new run-level 2 does not work, you can still reboot.

"s" is not a normal run-level. If you create a test version, replace the "s" with "2" after your complete testing. Run-level "s" is for system maintenance only.

Step 4 Test the changes you have made.

Step 5 After thoroughly testing your changes, restore the original initdefault value.

If you do not have a working state for the `initdefault` state, you may not be able to boot your system.

Sample /etc/inititab file

The following is an example /etc/inittab for a system with a system console and six terminals. The `initdefault` run-level is run-level 2. Run-level 2 is a multiuser run-level, with a `getty` on every terminal.

```
init:2:initdefault:

stty::sysinit:stty 9600 clocal icanon echo opost onlcr ienqak ixon \
icrnl ignpar </dev/systty

brc1::bootwait:/etc/bcheckrc &1 # fsck, etc.

slib::bootwait:/etc/recoversl &1 #shared libs

brc2::bootwait:/etc/brc >/dev/console 2>&1 # boottime commands
```

```
link::wait:/bin/sh -c "rm -f /dev/syscon; ln /dev/systty /dev/syscon" \
>/dev/console 2>&1
cwrt::bootwait:cat /etc/copyright >/dev/syscon # legal requirements
rc ::wait:/etc/rc &1 # system initialization
powf::powerwait:/etc/powerfail >/dev/console 2>& #power fail routines
lp ::off:nohup sleep 999999999 </dev/lp & stty 9600 </dev/lp
cons:012456:respawn:/etc/getty -h console console # system console
t1:2:respawn:/etc/getty tty01 H
t2:2:respawn:/etc/getty tty02 H
t3:2:respawn:/etc/getty tty03 H
t4:2:respawn:/etc/getty tty04 H
t5:2:respawn:/etc/getty tty05 H
t6:2:respawn:/etc/getty tty06 H
```

Changing system run-levels

This section contains a general procedure for changing the system from one run-level to another. You must be logged in at the system console as the superuser to change the system's run-level.

Step 1 Warn all users who are currently logged in before you change run-levels.

Caution

Changing to another run-level while users are logged on kills their processes if the run-level you are moving to does not contain rstate entries in /etc/inittab for their terminal.

Use the `write` or `wall` commands to communicate with the users. The `wall` (write all) command immediately sends your message to the terminal of each user on the system.

If each `getty` terminal entry has the new run-level in its rstate field, or if the rstate field is empty—implies all numbered run-levels—you don't need to ask them to log off; their processes will not be killed unless your new run-level is run-level "s".

Step 2 Change to a run-level other than run-level "s". Enter

`/etc/init` *new_run-level*

where *new_run-level* is the number of the run-level you want to enter.

To change to run-level "s," enter:

`/etc/shutdown`

Caution

Do not change to run-level "s" without using the `shutdown` command (that is, do not execute `init s`). The `shutdown` command provides safeguards to cleanly bring your system to single-user mode (run-level "s").

run-level 0 is a special run-level reserved for system installation. Do not use run-level 0.

Entering the system administration run-level

Many of the system maintenance tasks you perform as system administrator require the system to be in single-user mode, run-level "s", to ensure no one else is on the system while you're performing those tasks. In this run-level,

- The only access to the system is through the system console by the user root.
- The only processes running on the system are the
  - Shell on the system console
  - Background daemon processes started by `/etc/rc`
  - Processes that you invoke

Commands requiring an inactive system (such as `fsck`) should be run in run-level "s."

Use the `shutdown` command instead of `init s` when you change your system's run-level from any numbered run-level to run-level "s". The `shutdown` command kills all nonessential processes and brings the system safely to run-level "s"—without leaving system resources in an unusable state.

The `shutdown` command also allows you to specify a grace period to allow your users to terminate their work before the system goes down. The grace period is given, in number of seconds, immediately following the command name.

To enter run-level "s" with a grace period of 30 seconds, enter:

```
/etc/shutdown 30
```

This automatically warns all users that they have 30 seconds to log off, kill all processes, and safely bring the system to run-level "s."

For details on how to use the `shutdown` command, see "Starting and stopping SPP-UX" section on page 41; also see the shutdown(1M) man page.

Returning from the system administration run-level

To change your system's run-level from run-level "s," reboot your system with the `reboot` command. Use the `init` command as described earlier in this chapter. See "Changing system run-levels" on page 88 to change to the new run-level.

Managing your SPP-UX file system

6

File systems

This chapter provides a breif overview of file system basics and describes the tasks you perform to manage your SPP-UX file system.

- Creating file systems
- Adding and removing file systems
- Monitoring and controlling disk space
- Moving file systems

SPP-UX does not support management of your file system using HP-UX's system administration manager utility, SAM. You must perform these tasks manually.

Overview of SPP-UX file systems

This section briefly describes the types of file systems you work with on your computer system.

What is a file system?

In one sense, the term "file system" refers to the entire SPP-UX file system tree, the organization of all files on the system.

The SPP-UX file system is a hierarchical, tree-like structure in which the files are at the bottom of a branching structure that leads upward through subdirectories to a single root directory. The root directory of the file system is designated in path names by the character "/."

Mountable file systems

The term "file system" also refers to the specific collection of files on a storage device such as a disk.

You can create the structure for a new file system on a disk by using the `newfs` command. Once created, the file system, even though it is empty, encompasses the area on the disk in which it is created.

To use or access the file system, mount that file system to the existing file system tree. Except for the root (/) file system on the system disk, you can mount and unmount all file systems on disks to and from the existing SPP-UX file system tree.

You refer to an auxiliary file system by the name of the device file associated with the disk that contains the file system. You can mount the auxiliary file system by attaching it to a directory (the mount directory) in the root file system with the `mount` command.

You can unmount a file system, too, and then mount it again at a different mount point.

When users traverse the file system, they can move from the / directory to the files in /users/Bob as easily as they can move from / to the files in /usr even though /usr and /users/Bob are on different disks. As a user moves from one part of the SPP-UX file system to another, it isn't apparent that the file system actually consists of separate file systems on different devices.

Listing mounted file systems

To list the file systems mounted on your system, use the `bdf` command as shown in the following example:

```
# bdf
Filesystem            kbytes    used       avail  capacity   Mounted on
/dev/dsk/sd0a         484960  239992      196472     55%      /
/dev/dsk/sd1a         237810  47943       166086     22%      /users
```

In the above example listing, the file system on disk with the device file /dev/dsk/sd1a is mounted at the mount directory /users on the root (/) file system in /dev/dsk/sd0a.

Types of file systems

The principal types of file systems used by SPP-UX are the

- High-performance File System (HFS)

HFS file systems physically reside on mass storage devices, usually hard disk drives. HFS file systems are limited to 2 GB in size.

- Network File Services (NFS)

  NFS file systems are remote HFS file systems accessible over a network that can be used in a local file system.

- Convex large file systems

  Convex large file systems allow for file systems larger than 2GB, as well as individual files larger than 2 GB.

SPP-UX system files

Many important system files are located in the directories and subdirectories of the root (/) file system, described in Table 4.

Table 4 Root file system subdirectories

Directory	Contents
/bin	Compiled, often-used commands.
/dev	Block and character special device files used to communicate with devices. See the mknod(1M) man page.
/etc	Most system administrator commands and configuration (customization) files.
/etc/newconfig	Customized configuration files and shell scripts during an update so you can use them for reference. You typically copy many of these files back into /etc. The /etc/newconfig/README file contains useful information about files in /etc/newconfig.
/etc/conf	Kernel configuration description files.
/etc/filesets	A list of loaded filesets.
/lib	Object code libraries and related utilities.
/mnt	Users' home directories (usually).
/os	Contains the following files by default: mach—the low level kernel server—Provides systems services tunables—Tunes the server and mach without recompiling mach_init—Low level kernel initiator

Table 4 Root file system subdirectories —*(continued)*

Directory	Contents
/system	Revision lists and customize scripts from installation.
/tmp	Temporary files.
/usr	Commands and log files.
/usr/adm	System administration data files.
/usr/bin	Commands not required to boot, restore, or repair the file system.
/usr/contrib	Files and commands contributed by user groups.
/usr/contrib/bin	Contributed commands.
/usr/contrib/lib	Contributed object libraries.
/usr/contrib/man	Online documentation for contributed software.
/usr/convex	Files and commands for optional Convex software products.
/usr/convex/bin	Commands for optional Convex software products.
/usr/convex/lib	Object libraries for optional Convex software products.
/usr/convex/man	Online documentation for optional Convex software products.
/usr/diag	Diagnostic tools.
/usr/include	High-level C language header files; the shared definitions.
/usr/include/local	Site-specific C language header files.
/usr/include/sys	Low-level, kernel-related C language header files.
/usr/lib	Less-used object-code libraries, utilities, `lp` commands, and miscellaneous data files.
/usr/lib/uucp	Configuration files for UUCP at install.
/usr/local	Localized, site-specific files.
/usr/local/bin	Localized, site-specific commands.
/usr/local/lib	Object code libraries for the site-specific commands.
/usr/local/man	Online documentation for the site-specific software.
/lost+found	Orphaned files and directories created by `newfs` and used by `fsck`.

Table 4 Root file system subdirectories —*(continued)*

Directory	Contents
/usr/mail	Used by the mail facilities for your mail box.
/usr/news	System-wide news files.
/usr/spool	Spooled (queued) files for various programs.
/usr/spool/cron	Spooled jobs for `cron` and `at`.
/usr/spool/lp	Control and working files for the `lp` spooler.
/usr/spool/uucp	Queued work files, lock files, log files, status files, and other files for UUCP.
/usr/spool/uucppublic	Files freely accessible to remote systems via LAN and `uucp`.
/usr/tmp	Temporary large files.
/usr/man	All shipped online documentation for SPP-UX.
/usr/man/cat1...cat9	Online documentation that has already processed to speed up access.
/usr/man/cat1.Z...cat9.Z	Compressed versions of `cat` directories.
/usr/man/man1...man9	The unformatted online documentation pages.
/usr/man/man1.Z...man9.Z	Compressed versions of the online documentation pages.

The diskutil disk utility

The `diskutil` command performs many of the tasks associated with installing and using disks under SPP-UX. It can be used interactively or strictly through the command line. If no command is supplied on the command line, `diskutil` operates as an interactive shell. Otherwise, the single supplied command is executed and `diskutil` exits.

There are several operational modes you should be aware of. `diskutil` can be run under SPP-UX or HP-UX (to prepare disks for use under SPP-UX). Different techniques must be used internally by `diskutil` to operate on a disk that may be active when running under SPP-UX. `diskutil` attempts to detect this operating mode automatically.

You can use `diskutil` to prepare disks for Exemplar hardware or HP 700 Series workstations. There are some differences in how disks are configured for these two systems. There is not currently a way to determine what system the disk is targeted for, so Exemplar hardware is assumed.

diskutil command syntax

The `diskutil` command has the following format:

`diskutil` [`-d` *disk*] [`-h` | `-s`] [`-H` | `-S`] [`-v`] [*command*]

`diskutil` recognizes the following command-line options:

`-d` *disk*

Equivalent to a 'Select Disk *disk*' command. See "SELect Disk command" on page 99.

`-v`

Verbose mode.

`-V`

Displays `diskutil` version.

command

Any of the following commands:

- `Create`
- `Destroy`
- `Exit`
- `Force`
- `Help`
- `MAKe`
- `SELect`
- `SET`
- `SHow`
- `Quit`
- `SHow`
- `UNMap`
- `UNSet`

Interactive diskutil commands

The following sections explain each `diskutil` command.

`Create Stripe` command

Creates a disk stripe. The `Create Stripe` command has the following syntax:

`Create Stripe` *stripe_name* [`Blocksize` *blocksize*] *partition*

where:

stripe_name

Is the name of the new disk stripe.The format of the stripe name is `stripe`*n* where *n* is a decimal number between 0 and 255.

blocksize

Is the block size on each stripe partition.

partition

Partition to stripe.

`Destroy Stripe` command

Destroys a specified stripe and has the following syntax:

`Destroy Stripe` *stripe_name*

where *stripe_name* is the name of the disk stripe to be destroyed.

The stripe to be destroyed must not be selected at the time the `Destroy Stripe` command is entered. Before entering the command, first select a disk or a different stripe.

`Exit` and `Quit` commands

Terminate `diskutil`. There are no parameters for these commands.

`Help` command

Prints a usage statement and a brief command description. The sequence of upper and lowercase letters indicate the shortest acceptable version of the command. The following example shows output for the `Help` command:

```
DiskUtil: Help Help
          The available commands are:
          Create   Destroy   Exit   Force   MAKe   MAP   Quit
          SELect   SET       SHow   UNMap   UNSet
          To obtain more information about a command, type:
          Help <command-name>
          Each command keyword is always shown as a sequence of upper
          and lowercase letters.  The uppercase letters at the beginning of
          each keyword show the shortest abbreviation acceptable for the
          command keyword.
```

`Force Remap` command

Changes the logical unit name for a selected disk at the next system reboot. Use this command with caution as misuse can make your system unbootable. Syntax is

`Force Remap` *disk_name logical-unitX* `to` *logical-unitY*

where

disk_name

Is the disk to rename.

logical-unitX

Is the original logical unit number

logical-unitY

Is the new logical unit number

`MAKe Partition` command

Creates or changes a partition with the following syntax:

`MAKe Partition` *partition* `Size` *size* { `Offset` *offset* | `After` *partition* | `Before` *partition* } [`Description` *"string"*]

where

partition

Is the existing or new partition .

`Size`

Is the size of the new partition. *size* is designated in bytes and must be a multiple of 1024 bytes. A 'K' or 'M' suffix indicates kilobytes or megabytes. When modifying an existing partition, the *size* can be omitted.

`Offset`, `After`, or `Before`

Required.

offset is designated in bytes and must be a multiple of 1024 bytes. A 'K' or 'M' suffix indicates kilobytes or megabytes. When modifying an existing partition, *offset* can be omitted.

The following example creates an "a" partition 10 megabytes in size with and offset of 0 and named `number 1`. To verify the new partition is what you want, use the Show Partition command.

```
DiskUtil: make partition a size 10m offset 0 description "number 1"
DiskUtil: sh p
Logical disk name: sd0
partition table: (space available for file systems = 2098744)
part    offset      size    | partition description          | flags
```

```
a:          0K    10240K  |number 1                          |
b:    1024000K  1024000K  |                                  | *
```

`MAP` command

Changes the mapping of device files to actual hardware with the following syntax:

`MAP Disk` *type node:ctlr:tgt:lun* `To` *logical-unit*

The mapping of an active drive cannot be changed. A newly mapped drive is selected automatically. Any previous disk selection becomes invalid after this command.

`SELect Disk` command

Selects a target disk for further `diskutil` operations. The `SELect Disk` command has the following syntax:

`SELect Disk` *disk_name*

`SELect Stripe` command

Selects a disk stripe for further `diskutil` operations. The `SELect Stripe` command has the following syntax:

`SELect Stripe` *stripe_name*

`SET Partition` command

Sets or updates a partition's description or flags. The `SET Partition` command has the following syntax:

`SET Partition` *partition* [`Description` *"string"*] [`Flag` { `Crashdump` | `Default_pager` | `Raw` | `Vnode_pager` }]

`SHow Disks` command

Displays a list of available disks and associated nodes. The `SHow Disks` command has the following syntax:

```
DiskUtil: SHow Disks

SD 0:0:2:0 mapped to sd30
SD 0:0:3:0 mapped to sd1
```

The four colon separated fields denote

1. Node number
2. Controller
3. SCSI address
4. Logical unit number

SHow Lif_directory command

Displays the Lif directory. The SHow Lif_directory command has the following syntax:

```
SHow Lif_directory
```

SHow Partitions command

Displays all partition information on the selected disk.This command also accepts the following flags:

SHow Partitions [*partition* Stripeinfo]

where

partition

Is the selected partition.

Stripeinfo

Requests stripe related information.

Information from this command includes: logical disk name, available space for file systems, and a description of existing partitions. Table 5 explains partition flags.

```
DiskUtil: show partitions
Logical disk name: sd30
partition table: (space available for file systems = 4194157)
part    offset      size    | partition description          | flags
-------------------------------------------------------------------
 a:          0K   512000K  |/root filesystem                | *
 b:     512000K   512000K  |default pager                   | *D
 c:    1024000K  2048000K  |/usr filesystem                 | *
 d:    3072000K   256000K  |/tmp filesystem                 | *
 e:    3328000K    40960K  |crashdump filesystem            | C
 f:    3368960K   727040K  |vnode filesystem                | *V
```

Table 5 Partition flag descriptions

Flag	Description
None	Standard filesystem.
*	Busy filesystem. It is either mounted or in use by the default pager.
D	Default_Pager partition. This is swap space.
V	Vnode_pager. This is swap space for the kernel.

Table 5 Partition flag descriptions

Flag	Description
S	Stripe. The partition is part of a stripe.
C	Crashdump partition. This is the partition that will be used by crashdump.
R	Raw Partition.

`SHow STDisk` command

Displays information for a particular disk stripe. The `SHow Stripe Disk` command has the following syntax:

```
SHow STDisk
```

You must select a stripe using the `SELect Stripe` command before using the `SHow STDisk` command.

`SHow STRipe` command

Displays all valid stripes within the entire subcomplex, including nodes connected to and partitions.

This command shows a display of all disk stripes. The `SHow Stripe` command has the following syntax:

```
SHow STRipe
```

`UNMap Disk` command

Removes the logical unit name mapping for a disk with the following syntax:

`UNMap Disk` [*type:node:ctlr:tgt:lun*]

The mapping of an active drive cannot be removed.

`UNSet Partition` command

Removes a partition's description or clears a flag with the following syntax:

`UNSet Partition` *partition* [`Description`][`Flag` { `Crashdump` | `Default_pager` | `Filesystem` | `Raw` | `Stripe` | `Vnode_pager` }]

The disk stripe driver

In a normal file system configuration, a file system resides on a single disk and all operations on that file system are handled by I/O operations on that disk. File system performance is limited to the speed at which a single disk can process I/O requests. Also, the size of the file system is limited by the disk's capacity.

Disk striping improves I/O performance by overcoming the limitations of using a single disk for a file system. Disk striping also allows stripes that are larger than a single disk; this allows file systems to be larger than a single disk's capacity. You can create disk stripes spanning from 1 to 64 disks. All of the disks in a stripe must be physically located on the same hypernode.

The disk partitions in a stripe are arranged so that data blocks are allocated sequentially to the disk partitions. For example, if four partitions are in a stripe, the first block is on partition 1, the second block is on partition 2, the third block is on partition 3, and the fourth block is on partition 4. Subsequent blocks repeat the same sequence, so the fifth block is on partition 1, the sixth block is on partition 2, and so on.

When an I/O request is made to a disk stripe, the request is broken into several requests and redirected to each physical disk in the stripe. Each disk completes its portion of the I/O request. The total time to process the request can be as fast as 1 / n times the total time of a comparable request on a normal file system, where n is the number of disks in the stripe.

The SPP-UX disk stripe driver is part of the diskutil(1M) utility. The following commands perform basic operations on disk stripes:

- `Create Stripe`
- `Destroy Stripe`
- `SELect Stripe`
- `SHow STDisk`
- `SHow STRipe`

Creating a disk stripe

To create a new disk stripe, perform the following steps:

Step 1 Ensure that you have superuser capabilities.

Step 2 View or print the /etc/checklist file to identify the disk partitions that you can use to create a stripe.

Step 3 Start `diskutil`.

```
diskutil
```

Step 4 Enter the `Create Stripe` command, specifying a name for the stripe (*stripe_name*), a block size (*size*), and the disk partitions (*partitions*) that will be assigned to the stripe:

`Create Stripe` *stripe_name* **`[Blocksize`** *size* **`]`** *partitions*

If you do not specify a block size when you create the stripe, a default block size of 512 KB is assigned to the stripe.

```
DiskUtil: create stripe stripe0 (sd17a sd15c sd19b)
DiskUtil: show stripe
stripe Num   Valid/invalid    node  partitions
stripe0              valid      0   17a 15c 19b
DiskUtil: select stripe stripe0
DiskUtil: show stdisk
stripe0: unit=1, node=0, flags=0x1, serial num=354
        size =2096640, stripe blocksize =65536, disk blocksize =1024
        partitions are:
        17a 15c 19b
```

Destroying a disk stripe

To destroy a disk stripe, perform the following steps:

Step 1 Ensure that you have superuser capabilities.

Step 2 Start `diskutil`.

`diskutil`

Step 3 Locate the stripe you wish to destroy by displaying all the disk stripes in the system:

`SHow STRipe`

Step 4 Enter the `Destroy Stripe` command, specifying the name of the stripe (*stripe_name*):

`Destroy Stripe` *stripe_name*

Examining a disk stripe

To display all information about a disk stripe, perform the following steps:

Step 1 Ensure that you have superuser capabilities.

Step 2 Start `diskutil`.

`diskutil`

Step 3 Locate the stripe you wish to examine by displaying all the disk stripes in the system:

```
show stripes
```

Step 4 Select the stripe you wish to examine:

```
select stripe stripe_name
```

Step 5 Enter the `SHow STDisk` command:

```
show stdisk
```

Creating file systems

You can expand your file system using one of the following methods:

- Add a new disk drive and create a file system with SPP-UX commands.
- Add a disk with an existing file system and mount the file system.
- Mount an existing auxiliary file system that is now unmounted with SPP-UX commands.

Adding a disk and creating a file system

Complete the following steps to expand your file system when adding a disk. The procedures employ the direct use of SPP-UX commands.

- Install your disk.
- Map the disk with `diskutil`.
- Run newfs to create the file system.
- Create the new file system with `newfs` or `cnx_newfs`.

Step 1 Ensure that you have superuser capabilities.

Step 2 Install your disk.

To install your disk drive, refer to the documentation that came with your disk. Make the physical connections, determine the physical address of your disk, and add the disk to your system's I/O configuration.

Step 3 Map the disk with `diskutil`. Enter

`MAP Disk` *type node:ctlr:tgt:lun* `To` *logical-unit*

`diskutil` creates the required device files.

Step 4 Create a new file system using `newfs` or `cnx_newfs`.

See "Using newfs and cnx_newfs" on page 106 for more information about these commands.

The `newfs` and `cnx_newfs` utilities have the following syntax:

`newfs [-L | -S] [-n] [-v] [`*mkfs-options*`] special dev_file ype`

`cnx_newfs -L [ -F] [-N] [-n] [-v] [`*mkfs-options*`] special dev_file type`

where:

`-L`

Creates a file system with long file names (up to 255 characters).

`-S`

Creates a file system with short file names, maximum length is 14 characters. By default, `newfs` creates the same type as the root file system.

`-F`

Forces `newfs` to continue processing on a mounted file system. If you do not specify `-F`, `newfs` tells you the file system is mounted and waits for you to ok further actions. You cannot force `newfs` to work on a swap device, even if you use `-F`.

`-N`

Disallows large files (2 gigabytes - 1). This option is only valid with `cnx_newfs`. By default, `cnx_newfs` creates a filesystem with a magic number/feature bit pair that allows creation of files greater than 2 gigabytes. The maximum file size supported by the operating system is 1 terabyte - 512 bytes.

`-n`

Prevents installation of bootstrap programs.

`-v`

Specifies verbose mode.

mkfs-options

If you choose not to accept the default file system configuration you can use any of the following command-line options:

`-b` *blksize*

Designates block size in bytes.

`-f` *fragsize*

Designates fragment size in bytes.

`-i` *bytes_per_inode*

Specifies the density of inodes in the file system. Default is an inode for each 2048 bytes of data space. If you want fewer inodes use a larger number; to create more inodes a use a smaller number.

`-m` *%free_space_percent*

Desinates the percentage of space reserved from regular— or non root—users. Default is 10%.

special

Specifies the character—or raw—special file that references the disk.

dev_file

Specifies the character—or raw—device file that references the disk.

type

Specifies the type of disk.The disk type `scalios` sets the appropriate filesystem parameters for optimal layout on Exemplar systems. This disk type is set by default when invoking `cnx_mkfs`.

The `mkfs` and `cnx_mkfs` commands have more options than are listed here. Refer to the man page for more information.

Using newfs and cnx_newfs

The `newfs` and `cnx_newfs` commands are friendly front-ends to the mkfs(1) program. To construct a new file system `newfs`

- Calculates the appropriate options to use when calling `mkfs`.
- Installs the necessary bootstrap programs in the initial 8192 byes of the device—if the new file system is a root section.

`cnx_newfs` is a hard link to `newfs`. In order to create a filesystem that allows the creation of files greater than 2 gigabytes in size, you must use `cnx_newfs` to create the filesystem. To get only HP arguments use `newfs`.

When you make a root filesystem you *must* use one of the following commands:

- `newfs -L`
- `cnx_newfs -N -L`

in order to obtain the FS_MAGIC_LFN magic number. Open Boot Prom (OBP) only supports booting from a filesystem with FS_MAGIC_LFN as the magic number.

By using `newfs` or `cnx_newfs` options you can use these commands to

- Choose a maximum file name length.
  - Long.
  - Short.
- Modify the `mkfs` options that specify the way space is used in a file system.
- Support the creation of files greater than 2 gigabytes or large files.

  Large files capable file systems can also be either short or long file name filesystems.

Table 6 lists command line syntax and describes file system characteristics associated with optional flags.

Table 6 Command line syntax for newfs and cnx_newfs

Command line syntax	File system characteristics
`/etc/newfs [-F][-n][-v]` [*mkfs-options*] `special` *disk_type*	File names are the same length as the root file system.
`/etc/newfs -S[-F][-n][-v]` [*mkfs-options*] `special` *disk_type*	Creates a short file name file system—maximum of 14 bytes—set by DIRSIZ.
`/etc/newfs -L[-F][-n][-v]` [*mkfs-options*] `special` *disk_type*	Creates a long file name file system—up to 255 bytes—set by MAXNAMLEN.
`/etc/cnx_newfs -L [-F][-N][-n][-v]` [*mkfs-options*] `special scalios`	Creates a large file—greater than 2 gigiabytes— file system.

Options include the following:

`-L`

Creates a file system with long file names (up to 255 characters).

`-S`

Creates a file system with short file names, maximum length is 14 characters. By default, `newfs` creates the same type as the root file system.

`-F`

Forces `newfs` to continue processing on a mounted file system. If you do not specify `-F`, `newfs` tells you the file system is mounted and waits for you to ok further actions. You cannot force `newfs` to work on a swap device, even if you use `-F`.

`-N`

Disallows large files (2 gigabytes - 1). This option is only valid with `cnx_newfs`. By default, `cnx_newfs` creates a filesystem with a magic number/feature bit pair that allows creation of files greater than 2 gigabytes. The maximum file size supported by the operating system is 1 terabyte - 512 bytes.

`-n`

Prevents installation of bootstrap programs.

`-v`

Specifies verbose mode.

mkfs-options

You can use any of the following options on the command-line:

`-b` *blksize*

Designates block size in bytes.

`-f` *fragsize*

Designates fragment size in bytes.

`-i` *bytes_per_inode*

Specifies the density of inodes in the file system. Default is an inode for each 2048 bytes of data space. If you want fewer inodes use a larger number; to create more inodes a use a smaller number.

`-m` *%free_space_percent*

Desinates the percentage of space reserved from regular— or non root—users. Default is 10%.

device_file

Specifies the character device file name for the disk on which you are creating the file system.

disk_type

Specifies the type of disk.

Overriding default mkfs file system parameters

Table 7 provides a file system parameter reference and the following sections discuss reasons for overriding `mkfs` parameters.

Table 7 File system parameters

Parameter	Range	Default
File system size	Maximum size is 1 terabyte minus 512 bytes	None
Block size	4096 bytes (4KB), 8192 bytes (8KB), 16,384 bytes (16KB), 32,768 bytes (32KB), or 65,536 bytes (64KB)	8192 bytes (8KB)
Fragment size	1024 (1KB) to 8192 (8KB); fragment size must be at least one-eighth block-size	1024 bytes (1KB)
% space reserved	0 to 100	10 %
Bytes per inode	1 to (function of file system size and other parameters)	2048 bytes (2KB)

Overriding default block and fragment values

Configuring blocks and fragments represent a time/space trade-off. The larger the block size, the greater the access speed. However, more disk space is wasted. You can use one of the following suggested block and fragment combinations for the listed file systems (block size/fragment size):

- /tmp is usually 8K/8K to allow quick access. Most files in this directory are temporary. Therefore, wasting space is not a problem here.
- /usr is 8K/1K, which is the median trade-off between speed and space utilization.
- /mnt is usually 4K/1K because files that reside here are typically small and remain for a long time.

Overriding the default reserved disk space

The value of *free_space_percent* is the percentage of disk space reserved for the superuser when the file system fills up. It allows the superuser to reserve space for system use. The file system throughput degrades as the number of choices for free blocks is reduced. By setting *free_space_percent* at 10%, the default, you are

ensuring that the file system throughput does not degrade significantly.

Decreasing the value of the *free_space_percent* parameter lets you write to an additional percentage of file system space. The lower the percentage, the greater the possibility that your file's blocks will be scattered on the disk. Performance decreases as the disk fills up.

Overriding the default bytes per inode

The *bytes_per_inode* parameter dictates the relationship between the number of data bytes on the disk and the number of inodes allocated on the disk. Each file requires an inode. If you increase the number of bytes per inode, you are asking for fewer inodes. The default is to create one inode for every 2048 bytes of data space.

If your system has many small files, you can decrease the average number of bytes per inode. This gives you more inodes, and lets you create more, but smaller, files. Having many inodes takes more space on your file system.

If your system has a few large files, you can increase the space available for data by increasing the average number of bytes per inode.

Choosing file name length

By default, `newfs` creates a file system of the same type as the root (/) file system. However, you can explicitly specify the type of file system with either -L, which indicates a file system that allows long file names, or -S, which indicates a file system that permits only short file names.

Long vs. shortfile names

When configured with the short file names, SPP-UX file systems are compatible with earlier system releases that are not configured to accept long file names. Typically, file systems are configured with short file names.

Generally, long file names—the 255 character limit—provide flexibility in naming files. Also, files created on other systems that allow long file names can be moved to your system without being renamed.

Avoid long file names if:

- You plan to use applications that read directory file information and do not use portable directory routines. If

these applications assume that directories are in an array of fixed-size entries, they will not work with long file names. To correct this, rewrite the application to correct the assumptions about directories using the directory file information required by long file names.

- You have programs, with no source code available, developed for or compiled on releases of SPP-UX that do not support long file names.
- Other systems in your organization run versions of SPP-UX that impose a 14 character limit on file name length. In this environment, uniformity across the systems enables file movement between systems.

Modifying file name length

Table 8 lists potential problems and details troubleshooting suggestions you should consider prior to changing from short to long file names.

Table 8 File name length modification troubleshooting

Problem	Action
A program opens directories and reads directory entries directly.	Change the program to use directory library routines or use `get directories` system calls.
A program assumes that the maximum length of a file name in a buffer is 14 characters. For example, char filename[14] or char filename[DIRSIZ] MAXNAMLEN should be used for the buffer size, if only a few buffers are involved.	Enabe the DIRSIZ_MACRO compilation flag. This makes DIRSIZ a macro instead of a constant of 14. The macro accepts an argument that is a pointer to a struct direct and returns the size of the file name rounded to a 4-byte boundary. You can then allocate more memory for the file names.
A program includes and uses <dir.h> and uses `struct direct`. The `struct direct` for systems that support long file names is a variable length structure and the struct direct for systems that support only short file names is a fixed-size structure.	Include ndir.h and use directory libraries.
A program assumes there is only one file system magic number. The magic number for a system that supports long file names is different from the magic number for a system that supports only short file names.	Change the program to allow the new magic number for long file names.

Table 8 File name length modification troubleshooting —**(continued)**

Problem	Action
A program uses MAXNAMLEN and assumes it has a value of 14; when you convert to long file names, you need a MAXNAMLEN of 255.	Recompile the program.
A program uses DIRSIZ and assumes it is a constant of 14—meaning the maximum file name length is 14 characters.	Use MAXNAMLEN, Instead of DIRSIZ , to dictate the maximum file name length on a system that supports long file names. For systems that support only short file names, use DIRSIZ_CONSTANT to dictate the maximum file name length.

Enabling long file names

If you have an SPP-UX file system that allows only short file names, use the convertfs(1M) utility to convert the file system to allow long file names. See "Using convertfs(1M)" on page 113 for more information.

The process used to convert a file system to long file names is not reversible.

Step 1 Back up your entire file system.

The process used to convert file systems to long file names is not reversible. You should do a backup before you perform any operation that alters the file system.

Step 2 Shut your system down to single-user state by typing:

```
shutdown
```

Step 3 Run `fsck`.

When convertfs runs on file systems containing inconsistencies, the file systems might become corrupted, you correct the inconsistencies by running fsck on your file systems before converting them.

Step 4 Unmount all of your file systems:

```
umount -a
```

Step 5 Start `convertfs`.

```
convertfs
```

You receive the following messages:

```
Warning: Conversion to long file names is
irreversible and certain programs may not work with
long file names.
```

```
Converting the file system will cause a system
reboot. The system should be shut down into single
user state and all non-root file systems should be
unmounted before this utility is run.

Do you wish to continue? [y/n]
```

If you have your system in single-user state and have all non-root file systems unmounted, answer `y`. You then receive the following message:

```
This utility can convert all file systems in
/etc/fstab or can allow you to pick and choose file
systems.

Do you wish to convert ALL file systems to allow
long file names [y/n]?
```

If you answer `no`, the utility lists each /etc/fstab entry, its mount point, and asks you if you wish to convert the file system. Answer all the questions

As part of the conversion process, convertfs performs an fsck on each file system. See the fsck(1M) man page.

After you reboot the system or remount the converted file systems, you can use long file names on the converted file systems. The `newfs` and `mkfs` utilities create new files of the same type as the root file system. If you converted the root file system, all new file systems you create allow long file names. If you need a file system with short file names, use the -S option to either `newfs` or `mkfs`.

Using convertfs(1M)

The convertfs(1M) utility converts an existing file system which supports the default maximum file name length of 14 characters into one that supports file names up to 255 characters long. Once you convert a file system to long file names, it cannot be restored to its original state, since longer file names require a directory representation incompatible with the default directory format. Because this is an irreversible operation, `convertfs` prompts for verification before it performs a conversion.

Without arguments, the program interactively prompts you with a list of the file systems from /etc/fstab. You can choose any of the listed file systems for conversion. Typically, you should convert all of the file systems in /etc/fstab to avoid inconsistencies between two file systems mounted on the same system.

The convertfs utility performs the following actions:

- Modifies the superblocks.

- Reformats the directories
- Executes fsck
- Mounts the file system

Convert all or none of your normally mounted file systems listed in /etc/fstab. This prevents inconsistencies and undesired events, such as the overwriting of files, that can occur if you mix long and short file names on the same system.

If you have converted the root file system, convertfs reboots the system so that the changes made to the file system superblock will not be overwritten by an update of the superblock in the system memory.

Invoking `convertfs` with a file system's block special file name or character special file name converts that file system. If you want to convert a mounted root file system use the block special file.

If you execute `convertfs` with the name of the specific file system you want to convert:

```
convertfs /dev/rdsk/sd0c
```

The convertfs utility converts the named file system without prompting for input.

Disabling long file names

When you use convertfs(1M) to enable long file names, you cannot use the utility to convert back to short file names. If you must convert back to short file names, the file system should not have any file names longer than 14 characters. Recreate the file system with short names using the -S option to `newfs` or `mkfs` and then recover the original files from your backup media.

If your root file system needs to be converted back to short file names, it must be reinstalled. Be sure to save any files customized for your system so these files can be recovered after the reinstallation.

The following steps describe the process of converting your file system back to short file names:

Step 1 Examine all file names to make sure they have 14 or fewer characters. Use the mv command to rename files with long names so that they have names of 14 or fewer characters.

Step 2 Backup your entire file system after you have shortened the file names.

Step 3 Recreate the file system with short file names by executing `newfs -S`

Step 4 If you must change the root file system, reinstall it from the installation tape.

After creating a file system

After you have created a file system, you can add it, or mount it, to your existing file system by using the SPP-UX mount command. See the section "Mounting file systems" on page 115 for more information.

Mounting file systems

Mounting a file system links the file system contained on a specific device to a directory, or mount directory, in the existing file system tree. Once mounted, a file system becomes accessible to users.

Unmounted file systems are inaccessible to users. Unmounting a file system removes its files from the existing file system's hierarchy. The files themselves remain on the disk and can be accessed by mounting the file system again. Mounted file systems are automatically unmounted at shutdown time. The root (/) file system cannot be unmounted.

If you create a file system using newfs(1M), you can mount it using the mount(1M) command.

The mount directory

The mount directory becomes the root directory for the file system you add. The mount directory should be an empty subdirectory on the existing file system. Create the directory using mkdir(1M) if it does not exist. To specify the mount directory, indicate the full absolute path name.

If you specify a non-empty directory for the mount point, any files in the mount point directory will be inaccessible when the new file system is mounted. The files still exist, but remain inaccessible until you unmount the overlaying file system. Therefore, do not mount a file system over data that you will need later.

Using mount(1M)

The mount command attaches a remote file system existing on a specific device to a mount directory in the existing local file system. The remote file system can be on a disk connected to your system or it can be a file system that is part of an NFS type file system.

The mount command updates the file /etc/mnttab, which lists existing mounted file systems. The contents of /etc/mnttab are displayed when you enter the command `mount` without arguments.The mount(1M) command has the following syntax:

`mount` *fsname directory*

where:

fsname

Specifies the name of the block device file associated with the device containing the file system to be mounted. For example, you might specify the disk with the block device file name, `/dev/dsk/sd0c`.

If you specify a remote file system, use the form: host:path. For example,

`hpfcd:/build`

directory

Specifies the directory in the existing file system where the file system is to be mounted.

The mount command has many options that are fully described in the mount(1M) man page.

Mounting local file systems

To use the mount command to add a file system, follow the steps outlined below.

If necessary, connect the disk containing the file system to your computer. To install your disk, refer to the documentation that came with your disk.

Step 1 Determine the device file for the disk containing the file system. If device files for the disk do not exist, create them using `/dev/MAKEDEV`.

Step 2 Determine the mount point directory for the file system. Use `mkdir` to create the directory if it does not yet exist.

Step 3 Determine which mount command options—if any—to use when you mount the file system.

Step 4 Use the mount command to add the file system.

For example, if you want to add the file system on a SCSI disk with the device file /dev/dsk/sd0c. The disk contains the files of most of the users of the system, so you specify the empty directory /users as the mount point directory for the file system. Type:

```
mount /dev/dsk/sd0c /users
```

This command mounts or attaches the file system on the disk using the device file /dev/dsk/sd0c to the mount point /users.

Step 5 Edit /etc/checklist to have the file system automatically mounted when you boot your system.

Refer to the section "Making entries in the checklist file" on page 119 for details concerning adding entries to the /etc/checklist file.

If an HFS file system has been unmounted improperly and not checked for inconsistencies with the fsck(1M) utility, the mount command can not mount the file system. Run fsck on that file system before attempting to remount it. See "Using fsck" on page 124 for more information.

Mounting remote file systems

You can use the mount(1M) command to mount file systems located on a remote system.

Preparations for mounting remote file systems

Before you can mount file systems located on a remote system, Network File Systems (NFS) utilities software must be installed and configured on both local and remote systems. See the *Exemplar Networking Guide* (DSW-865) for more information on NFS utilities.

Note the following about using NFS file systems.

- Entries must exist in the /etc/hosts file for both
  - The NFS Server—the system where the file systems reside, which is the machine exporting the file systems.
  - The NFS Client—the machine where remote file systems are mounted.

  A machine can be an NFS server (export file systems), an NFS client (mount remote file systems), or both.
- Entries must exist in the /etc/exports file on the NFS Server for both
  - The file systems you want to mount.
  - The NFS Clients that can mount the file systems.
- Run /etc/exportfs after you add an entry to /etc/exports.

The /etc/exports file lists file system names, left justified, followed by client machine names separated by colons (:).

Client machine names are searched for in /etc/netgroup then in /etc/hosts. An entry in /etc/exports that lists only a file system

name, and no machine name, indicates that all machines can access the file system. A pound sign (#) anywhere in the file indicates a comment extending to the end of that line. The following example shows sample lines from an /etc/exports file.

```
/usr/games fudge        # export only to machine fudge
/usr/local              # export to the world
```

Mounting an NFS file system

To mount a remote file system to a mount directory on your local file system, complete the following steps:

Step 1 Create the mount directory if it does not already exist:

`mkdir` *directroy*

where *directroy* is the absolute path name of the mount directory on your local machine.

Step 2 Mount the remote file system using the mount(1M) command.

`mount` *host:path directory*

This makes the files in the remote directory named in *host:path* accessible from your local machine as *directory.*

where

host:path

Is the remote hostname and absolute path name of the filesystem you want to attach to your system.

directory

Is the name of the directory you created in Step 1.

NFS mount options

When mounting NFS file systems, you can use options that affect mounting conditions, user permissions, and so on. See the mount(1M) man page for more information.

NFS mount problems

When the mount command succeeds, it is silent. Otherwise, you receive an error message. If the attempt to mount a remote file system fails, be sure to verify that:

- The client machine has an entry in NFS server's /etc/exports file that allows it to mount the remote file system.
- The mount directory exists, is not currently being used as a mount point, or that no files in the directory are in use.

Mounting file systems automatically at bootup

To automatically mount your file systems at bootup, complete the following steps:

Step 1 Login as superuser.

Step 2 Edit the /etc/checklist file and add the file systems you want automatically mounted.

Step 3 Edit the /etc/rc script to add a mount -a command, if it is not already present.

To mount all file systems described in /etc/checklist, as well as hfs type file systems, add the line

```
/etc/mount -a -t hfs
```

to the hfsmount() section of the /etc/rc file.

Making entries in the checklist file

To display information about all of the currently mounted file systems enter the following command:

```
/etc/mount -p
```

You can also see the same information by typing `mount` without options; although the display is not formatted.

An entry for a file system in the /etc/checklist file has seven fields separated by blank spaces. In all but the last of the fields, you need to put either an entry or , in some fields, a placeholder. The following is a sample entry from an /etc/checklist file:

```
/dev/dsk/sd0c /users hfs defaults 0 2 # users
```

where

`/dev/dsk/sd0c`

Is the block device file corresponding to the disk used for the file system.

`/users`

Is the mount point directory; this directory is located in the existing file system and will serve as the root directory for the mounted file system.

`hfs`

Is the type of the entry, the type must be one of the following depending on the type of file system:

- `hfs` (local high-performance file system)
- `nfs` (file system available through NFS Services)

It is also possible to enter ignore in this field when you want to retain the entry for the file system in the checklist file, yet do not want the file system mounted at the time the system boots. You can mount the system later.

`defaults`

Uses all default options—rw,suid. Other options include

`rw`

Read-write (default).

`ro`

Read-only.

`suid`

Set user ID execution allowed (default).

`nosuid`

Set user ID execution not allowed.

These options correspond to the options used with the mount command. More options are available for use with the NFS type file systems.

`0`

This field is reserved for future use by backup utilities. 0 is a placeholder.

`2`

Is a pass number. of 1 for the root file system and larger numbers to other file systems. The pass number is used by the fsck command issued with the -p command. The fsck utility ignores file systems with pass numbers of 0, which is typically used for NFS file systems.

`# users`

Is an optional comment preceded by the # character.

The following sample checklist file describes a system using two disks;

```
more /etc/checklist

/dev/dsk/sd0a     /      hfs defaults 0 1 # root
/dev/dsk/sd2a     /users hfs defaults 0 2 # users
```

How SPP-UX uses /etc/checklist

At boot up, the /etc/bcheckrc and /etc/rc file system scripts are executed from /etc/inittab. The /etc/bcheckrc script checks each file system listed in /etc/checklist and determines whether the

file system was properly shutdown. If a file system appears to have been previously shutdown improperly, bcheckrc runs fsck during the startup process. The /etc/rc script mounts all file systems listed in /etc/checklist.

Unmounting file systems

To unmount a file system, use the umount(1M) utility. All files on the file system to be unmounted must be closed. Attempting to unmount a file system that has open files, including your working directory, causes the umount(1M) utility to fail without unmounting the file system.

Syntax of the umount command

The umount command has the following syntax:

umount [*sfname*] [*directory*]

where:

sfname

Is the pathname of the block device file for the device containing the file system to be unmounted or the name of a remote file system in the form: host:path.

directory

Is the directory where the file system is mounted.

Always unmount file systems contained on a mass storage device before removing the device from the system. Removing a device containing mounted file systems—for example, disconnecting or turning off the power to a disk, or removing a disk pack from a mass storage device—can corrupt your file systems.

Unmounting a file system

To unmount a local HFS file system on the disk with the device file /dev/dsk/sd0c mounted at the mount point directory users, issue the command:

```
umount /dev/dsk/sd0c
```

or

```
umount /users
```

To unmount a remote NFS file system, you can issue a command such as:

```
umount hpfcf8:/users
```

Unmounting currently mounted file systems

The umount command removes the specified file system from the /etc/mnttab file. If you wish to unmount all your currently mounted file systems contained in the /etc/mnttab file execute:

```
umount -a
```

Unmounting an exported file system

To unmount a file system that is currently exported

- Unmount the file system from every client.

  Unmounting an exported file system on which users of remote client systems have open files could result in lost data.
- Unmount the file system from the server.

Unmounting file systems by type

To unmount currently mounted file systems of a particular type, you can use the -t option. For example, to unmount all NFS file systems, use the command:

```
umount -a -t nfs
```

The root (/) file system cannot be unmounted.

Remove the file system entry in the file /etc/checklist for the file system you have just unmounted.

If you have unmounted a file system and you do not wish to remove the entry from the /etc/checklist file, you can enter ignore in the type field. The file system is not mounted when the system boots but can be mounted later.

Automatically unmounting file systems

When you execute the shutdown(1M) command, the system unmounts all of the file systems listed in the /etc/mnttab file. `shutdown` uses `umount -a` to unmount file systems. File systems are also unmounted when you use the reboot(1M) command.

Moving file systems

This section outlines and gives examples on how to copy data from one disk to another disk of a different type.

- Back up files from the current disk onto tape.
- Add your new disk to your system.

- Create new file systems using the `newfs` command on your new disk.
- Edit the files to create entries for new file systems.
- Mount your new file systems.
- Restore the files backed up on tape to your new file systems.

Copy this file system to the new disk as follows:

Step 1 Back up your current file system onto tape with the fbackup(1M) command.

`/etc/fbackup -i` *filesystem*

where

filesystem

Is the directory path to include in the backup. This option may be specified multiple times.

Step 2 Add the new disk to your system and make your new file system.

Refer to the documentation that came with your disk for installation information. Use `newfs` to make the new file systems on your new disk. Refer to "Creating file systems" on page 104 for more information.

Step 3 Edit /etc/checklist.

After create the file systems on your new disk, edit the /etc/checklist file. Change the device file to reflect your new disk. Editing this file tells the system to automatically mount the new file system each time you boot your system.

Step 4 Mount the new file system with the mount command. Enter

`mount` *device_file filesystem*

device_file

Specifies the name of the block device file associated with the device containing the file system to be mounted.

filesystem

Specifies the directory in the existing file system where the new file system is to be mounted.

Step 5 Restore your backed up files to the file system on the new disk with the frecover(1M) command.

Load the tape with your files on the tape drive and restore the files to their new file system. Use the following commands:

`/etc/frecover -x -i` *filesystem*

where

`-x`

Extracts the files identified by the `-i` option from the backup tape.

`-i` *filesystem*

Is the directory path of the recovered file system.

Checking file systems

The SPP-UX file system can develop inconsistencies over a period of time. For example, turning off the computer without previously shutting down or unmounting the file system can cause some corruption of the system. Except for obvious events, such as a power failure, the causes of file system corruption are difficult to determine.

Check the file system for consistency periodically and anytime you suspect a problem. Some commands, such as `/etc/update` or `convertfs`, will not function properly unless the file system is free of inconsistencies.

Using fsck

Use the `fsck` utility to check file systems for any inconsistencies and to make any necessary repairs. When you run `fsck`, make sure the file systems are inactive.

Step 1 Make sure all users are logged off the system.

Step 2 Bring the system to single-user mode. Enter

```
/etc/shutdown
```

This terminates running processes and places the system in single-user mode.

Step 3 Run fsck in single-user state.

Large file systems

SPP-UX large file systems allow the creation of files and file systems larger than 2 GB. A large file system allows you to create and manipulate files up to one terabyte minus 512 bytes in length.

Caution

Do not create a large file system on the root partition. The Exemplar Open Boot utility can not boot SPP-UX on a root partition that contains a large file system. The root partition must be a standard HFS long-filename file system that is smaller than 2 GB.

The Open Boot file reader cannot follow symbolic links, so filenames must be hard links, like the standard /os/mach, /os/tunables, and /os/server.

Large file system utilities

The following file system utilities allow you to create and maintain large file systems:

`cnx_dumpfs`

Prints the super block and cylinder group information for a file system. When used with the `-l` option, `cnx_dumpfs` prints only the file system magic number and feature bits. If a file system's magic number is FD_MAGIC_2 and the feature bits are FSF_LARGEFILES, the file system allows the creation of files larger than 2 GB. For more information, see the cnx_dumpfs(1M) man page.

`cnx_mkfs`

Constructs a file system. This command uses a prototype specification for the file system; in order to create a large file system using `cnx_mkfs`, you must specify a prototype for a large file system. The `cnx_newfs` command provides an easy-to-use front end to `cnx_mkfs`. For more information, see the cnx_mkfs(1M) man page.

`cnx_newfs`

This command is a front end to the `cnx_mkfs` command. By default, a file system created using `cnx_newfs` is a large file system. For more information, see the cnx_newfs(1M) man page.

Creating a large file system

To create a large file system, Log in as `root`, make sure the partition is mounted, and enter the following command:

`/etc/cnx_newfs` *special_file* `scalios`

For example, to create a /dev/rdsk/sd1a partition capable of containing large files, enter:

```
/etc/cnx_newfs /dev/rdsk/sd1a scalios
```

Checking a file system for large file capability

To check a file system in order to determine whether it is capable of having large files, enter the following command:

```
/etc/cnx_dumpfs -l filesystem
```

If the `cnx_dumpfs` command returns the following information, the file system does not have large file capability:

```
/etc/cnx_dumpfs -l /

magic    FS_MAGIC_LFN

featurebits     FSF_LFN
```

Your system's root partition (/) must not have large file capability.

If the `cnx_dumpfs` command returns the following information, the file system does have large file capability:

```
/etc/cnx_dumpfs -l /scratch

magic    FD_MAGIC_2
featurebits     FSF_LARGEFILES
```

Managing printers 7

Line Printer Spooling System (lpss)

The Line Printer Spooling System (`lpss`) is a set of programs, shell scripts, and directories that control your printers and the flow of data going to them. The `lpss` command

- Prevents intermixed listings.
- Provides control of printout routing.
- Allows users to control print requests.

Once a printer has been added to your system, it can be added to `lpss`. We recommend adding all printers to `lpss`. If you do not add your printer or plotter to `lpss`, there is no coordination between multiple users, and intermixed listings can occur. The purpose of `lpss` is to automatically coordinate between multiple users and prevent intermixed listings.

Note

Local printers are not supported on Exemplar systems.

The term *printer* can be interchanged with the term *plotter* for the tasks described in this chapter.

This chapter covers the following topics:

- Components of `lpss`
- Remote printing
- Controlling data flow through `lpss`
- Priorities of printers and print requests
- Using plotters with `lpss`
- Collecting and reporting statistics

This chapter describes how to accomplish the tasks associated with these topics using SAM and SPP-UX commands. The SPP-UX commands method is provided for those who do not have access to SAM or choose not to use it.

If you are already familiar with the basic concepts of `lpss` you may want to proceed directly to the tasks described in this chapter. "Using SAM to manage printers" begins on page 137 and "Using SPP-UX to manage printers" begins on page 144.

If you are working with `lpss` for the first time or if you want to review key concepts before performing a particular task, read "`lpss` overview."

lpss overview

You can think of `lpss` as if it were a plumbing system. The data to be printed represents the water in this system. Various request directories, sometimes referred to as printer queues, serve as temporary holding tanks for print requests until they are sent to a printer to be printed. The flow of print requests is controlled at the request directory and printer level. The terms *accept* and *reject* refer to controlling the flow of print requests to the request directories while the terms *enable* and *disable* refer to controlling the flow of print requests to the printers. Accepting, rejecting, enabling, and disabling print requests control the data through `lpss` as valves would control the flow of water in a plumbing system. Shell scripts (called interface scripts) near the end of the data flow serve as pumps that pump an orderly flow of data to the printers.

The line printer scheduler controls the routing of print requests from the request directories to the printers. It functions as an automated flow controller to prevent intermixed listings and to provide efficient use of the printers on your system. Intermixed listings are multiple print requests printing on a printer simultaneously that result in printed pages with characters from different print requests mixed together.

If the "drain gets clogged" for one printer, you can reroute the print requests for that printer to another printer and you can "flush" unwanted print requests from the spooling system. You can also send a print request to another computer to be printed. Sending print requests to another computer to be printed is called *remote spooling*, and the other computer is referred to as a remote system. When you use remote spooling, a special shell script called `pump` is used to send the data to a remote system. A program on the remote system receives the data and directs it into the remote system's LP spooler.

lpss components

The components of `lpss` are:

- Printer names
- Printer classes
- Print destinations
- System default printer
- Printer interfaces
- Printer models
- Line printer scheduler
- Local printer
- Remote printer
- Print request identification number

Printer names

When you configure a printer into `lpss`, you assign a printer name that you refer to when you later submit print requests. Printer names can contain up to 14 characters, which can be alpha numeric or underscores. The name may or may not be the same as the device file name; but some correspondence between the printer name and device file name is suggested. The printer name is the name of the printer that shows up when you request the status of the printer queue with the `lpstat` command.

A hypothetical system "hypo1" has the following printers defined in its `lpss`. The printers have the following names:

- laser1
- laser2
- phred
- letterhead
- invoices
- check_printer

Printer classes

You can treat a group of printers as if they were one printer. A printer class is a name that you can use to refer to the group of printers. When submitting a print request you can specify a particular printer name or a printer class name. When submitting a print request to a printer class, the print requests print on the first available printer in the group rather than on a specific printer. Printers that are members of a printer class can still be referenced individually. Creating a printer class is optional.

The following considerations and restrictions apply to printer classes:

- Printer class names can contain up to 14 characters, which can be alpha numeric or underscores.
- Printer class names and printer names on the same system cannot be the same name. Printer names and class names must all be unique.
- Printer classes cannot include remote printers.
- A printer class must contain at least one printer.
- A printer can only belong to one printer class at a time.
- To remove a printer from a printer class, you must remove the printer from `lpss`. Then add it back to the lp system without specifying a printer class.

On the hypothetical system "hypo1," three of the printers are grouped into a printer class called "laser".

- printer class: laser
  - laser1
  - laser2
  - phred

Print destinations

Several of the commands for `lpss` require you to specify a print destination. A destination is the name of a printer or printer class.

For our example system "hypo1", possible destinations are:

- printer class: laser
- laser1
- laser2
- phred
- invoices
- check_printer
- letterhead

System default printer (destination)

You can appoint one of the print destinations in your `lpss` to be the system default printer. It is not necessary to have a system default printer, but it is recommended. A system default printer receives any print requests not sent to a specific print destination. You can have only one system default printer.

In addition to, or instead of, a system default printer, you can assign each user a default printer to use. To do this, set the user's

LPDEST shell environment variable to the name of the system default printer.

- If LPDEST is set and a user does not specify a different printer to use, the printer referenced by LPDEST is used.
- If LPDEST is not set for a user, and the user does not specify a printer, the system default printer—if one is set—is used.
- If neither LPDEST or the system default printer is set, a user must specify a printer or printer class.

Printer interfaces

A printer interface, also known as an interface script, is the final stage of lpss. It is the part of lpss responsible for sending data to a printer. Each printer that you have defined for use by lpss has its own interface script that resides in the /usr/spool/lp/interface directory. When printers are added to lpss, an interface script is copied from /usr/spool/lp/model to /usr/spool/lp/interface and given the printer name.

If we were to list the directory /usr/spool/lp/interface on our hypothetical system "hypo1," it would contain the printer interface files laser1, laser2, phred, letterhead, invoices, and check_printer.

The entry for the class name laser would be located in the directory /usr/spool/lp/class; it would not be found in the interface directory.

The network-based printer script copies the system default interface script for the printer to /usr/spool/lp/interface/model.orig and renames the interface script the model or printer name. The script then replaces the system default script with a custom script in /usr/spool/lp/interface.

Printer models

When you configure your printer into lpss, you must specify which printer model interface script you want to use. The model is automatically copied from the /usr/spool/lp/model directory into the /usr/spool/lp/interface directory and given the name that you specified as your printer name.

If you have an HP printer, you will probably find a model script that matches its model number or name. Those interface model scripts that match your printers typically do not need to be changed. If you know how to do shell programming, you can customize printer interface model scripts to meet your specific printing needs.

If you do not have an HP printer, try using the dumb interface model. You might have to modify it to be able to use all of the features of your non-HP printer, but "dumb" should work for basic ASCII text printing. If the dumb printer interface model script does not work, contact your printer supplier for a SPP-UX line printer spooler interface script or try the script that most closely matches your non-HP printer type.

Line printer scheduler

The line printer scheduler is the heart of `lpss`. It is the part of `lpss` that prevents intermixed listings—output from more than one print request mixed together on a printed page—and controls flow of print requests to the printers. Its duties also include

- Monitoring printer and print request priorities
- Monitoring/adjusting printer status
- Logging `lpss` activities.

The `lpsched` command starts the LP spooler. Because of the central role it plays, starting `lpsched` is referred to as "starting the LP spooler", and stopping `lpsched` is often referred to as "stopping the LP spooler." You can use the `lpsched` command directly or through SAM.

Local printer

A local printer is a printer physically connected to your system. Local printers are not supported on Exemplar systems.

Remote printer

A remote printer is a printer not physically connected to your system, but accessed by your system through a local area network (LAN). To configure a remote printer into your local `lpss`, you must be able to access the remote system via a LAN.

Network-based printer/plotter

A network-based printer or plotter is connected directly to the local area network (LAN). A network-based printer or plotter is not physically connected to any system.

Print request identification number

When you submit a print request by means of the `lp` command, `lp` responds with a print request identification number consisting of the name of the printer (or printer class) followed by a number.

Remote spooling

If you have several systems connected to a Local Area Network (LAN) and would like the systems to share a printer, you can set up `lpss` on the systems not physically connected to automatically send their print requests—via the LAN— to `lpss` of the system that does have the printer. The systems without printers act as a user on the system with the printer, submitting print requests to that system's `lpss`. This is accomplished by a special program known as the Remote Spooling Daemon (`rlpdaemon`).

The `rlpdaemon` program runs in the background (on the system with the printer) monitoring the incoming LAN traffic for any remote print requests from other systems. When these requests arrive, the `rlpdaemon` program submits them to its local `lpss` on behalf of the remote user. In addition to remote print requests, the remote spooling daemon must also handle "cancel" and "status" requests from remote systems.

Special interface scripts on the remote systems issue cancel and status requests. These special interface scripts are similar to printer interface scripts. They have a model directory that holds sample versions of the scripts, and they have an interface directory where scripts currently in use reside. The cancel and status models are copied into their respective interface directories automatically when adding a remote printer.

The directory /usr/spool/lp/cmodel contains a sample interface script, `rcmodel`, that sends a remote cancel command to the system with the printer. Configuring a remote printer into your `lpss`, copies the cancel model script into the /usr/spool/lp/cinterface directory and gives it the same name as the printer.

The directory /usr/spool/lp/smodel contains a sample interface script, called `rsmodel`, that sends a remote status command to the system with the printer. Configuring a remote printer into `lpss`, copies the status model script into the /usr/spool/lp/sinterface directory and gives it the same name as the printer.

It is unlikely that you will need to customize the remote cancel and status model scripts. If you do customize these "remote control" scripts, you must copy them to a different file name to avoid destroying your changes when updating the system with the update utility.

Configuring a remote printer into your `lpss` requires the following information:

- The name of the system with the printer.

- The interface script to use when issuing a remote cancel request.
- The interface script to use when issuing a remote status request.
- The name of the destination printer—as it is defined in `lpss` of the remote system.

Priorities of printers and print requests

To control the order of printed requests, assign priority values to printers and to specific print requests.

Note

Assigning priorities is not required.

- Priority values must be in the range of 0 to 7.
- Priority 7 is the highest priority.
- A printer's fence priority determines which print requests get printed. A request's fence priority must be equal to or greater than the printer's fence priority to print. SPP-UX assigns a printer fence priority value of zero (0) when you add a printer to `lpss`. You can change printer fence priorities dynamically with SAM or the `lpfence` command. See "Changing a printer fence priority" on page 143 for more information.
- Each print request has a request priority, which is automatically assigned the destination printer's default request priority. The line printer spooling system determines the print request priority for each printer. If the print request priority changes after a print request goes into the print queue, the print request's priority does not change.
- Print request priorities lower than the printer priority do not print. If a print request's priority is lower than its printer's priority, it remains in the request directory —printer queue— for that printer. The request remains there until
  - Its priority is raised
  - Its printer's priority is lowered
  - The request is canceled
- You cannot directly set a printer class priority. The class priority is the same as the highest priority of any printer in the class. See "Creating a printer class" on page 146 for an example of a printer class.
- If multiple print requests wait to print on a specific printer and all have priorities high enough to print:

- `lpss` prints next the print request with the highest priority.
- If more than one print request has the highest priority, all print requests with that priority print in the order they were received by `lpss`.

Controlling data flow through the spooler

There are three points in `lpss` where you can control the flow of data:

1. You can start or stop the LP spooler. This has a global effect. If you stop the LP spooler, all printing stops.
2. You can tell `lpss` to accept or reject any new print requests for a printer. If you instruct `lpss` to reject print requests for a printer class, users are given a message telling them that the printer class that they requested is not accepting requests. Rejecting print requests should be used when a printer or a class of printers is taken off the system for an extended period of time. Do not reject print requests when making the printer unavailable for a short time. For example, rejecting print requests for adding paper or changing the toner cartridge. A minor delay due to these short term services is usually acceptable.
3. You can tell `lpss` to enable or disable a printer for printing. Print requests continue to be accepted for a disabled printer unless you have explicitly rejected print requests. Disabling a printer should be done to make the printer unavailable for a short time, for example, to add paper or change toner. Do not disable a printer for a long time without also rejecting requests for that printer; otherwise users' print requests keep accumulating in the print queue, and they do not get any notice that their requests will not print. Once you reject print requests for a printer, a user submitting a print request to that printer gets a message stating that the printer is not accepting requests.

To print, a printer must be accepting and enabled.

Note

When you use SAM to enable or disable a printer, SAM performs both the accept/reject operation and the enable/disable. If you wish to disable a printer but still accept requests for that printer, letting them accumulate in the request directory for the printer, you must use the SPP-UX commands method, see "Disabling a printer" on page 142.

Logging and analyzing printer activity

Analyzing printer activity helps you determine if there are bottlenecks in `lpss`. It also helps you determine/justify the need to add additional printers to your `lpss`. There are facilities to help you analyze the flow of data through your `lpss`.

There are two phases to analyzing `lpss` activity:

- Data collection phase
- Data reporting phase.

The data collection phase begins when `lpss` starts. The `-a` option to the `lpsched` command turns on data collection processes when you start the LP spooler. The data reporting phase occurs any time after `lpss` starts. The following statistics are calculated:

- Average waiting time from when a print request is submitted to the start of printing.
- Standard deviation for waiting time.
- Average printing time from start to end of print request.
- Standard deviation of printing time.
- Average number of bytes (characters) printed per request.
- Standard deviation for number of bytes.
- Sum of bytes printed for all requests in Kbytes.
- Total number of requests since logging started.

Initial spooler setup

Initial LP spooler setup consists of the following tasks:

1. Add at least one printer to `lpss`.
2. Tell `lpss` to accept print requests for this printer.
3. Tell `lpss` to enable the printer for printing.
4. Turn on the LP spooler.

When you use SAM to add a printer, SAM:

- Tells `lpss` to accept print requests for the printer.
- Enables the printer.
- Starts `lpss`.

Spooler tasks

The two methods of controlling `lpss` are:

- The System Administration Manager (SAM)
- SPP-UX commands

SAM allows you to control `lpss` through its menu-selection and data-entry screens. By combining multiple manual commands into single tasks, SAM saves you time and keystrokes. SAM also eliminates the need to know command names and options for `lpss`.

Although SPP-UX commands require you to learn more details, you might want to use SPP-UX commands to give you a greater degree of control over `lpss`. If SAM is not configured into your system, you have to use SPP-UX commands to control `lpss`.

If you want to use the data collection facility, you must enter the `lpsched -a` command to start data collection.

To start SAM, ensure that you have superuser capabilities, then enter:

```
/usr/bin/sam
```

SAM presents you with the main window. For additional information about using SAM, activate the Help control button or see "Using SAM" on page 3.

Using SAM to manage printers

To perform system administration tasks on a remote system using SAM, run SAM, highlight Remote Administration, and activate the Open control button. For additional information refer to the SAM help system and "Using SAM" on page 3.

SAM provides an online help system to assist you when you need additional information.

Activating the Help button from the SAM main window, a dialog box, or message box gives you information about the attributes and tasks you perform from the currently displayed window.

From within a functional area, choosing an item from the Help menu gives you information about:

- Using the current functional area
- Navigating within SAM using the keyboard
- Using the SAM help system
- Displaying the version of SAM you are currently running

From a dialog box—a window displaying fields to be filled in, pressing the F1 key gives you context-sensitive information for the object at the location of the cursor.

Viewing printers

To view printers using SAM:

Step 1 Run SAM; enter:

```
/usr/bin/sam
```

Step 2 Highlight Printers and Plotters, and activate the Open control button.

Step 3 Highlight Printers/Plotters and activate the Open control button.

Viewing print request status

To view print requests:

Step 1 Run SAM; enter:

```
/usr/bin/sam
```

Step 2 Highlight Printers and Plotters and activate the Open control button.

Step 3 Highlight Print Requests and activate the Open control button.

You can also view print requests by choosing Print Requests from the List menu within the Printer/Plotter Manager Window.

The print request queue can change rapidly. To ensure that you are viewing the current data, choose Refresh List from the Options menu to view the current state of the print request queue.

The "Printer/Plotter Manager" object list displays the following information:

- System default printer
- Status of LP spooler (RUNNING or STOPPED)
- Printer name
- Printer status—Enabled, disabled, idle, busy
- Priority for each printer and printer class
- Printer type—Remote or network
- Printer location with
  - Printer name
  - Remote printer's system
  - No entry shows for network-based printers

Adding a remote printer

To add a remote printer to your line printer spooler system complete the following steps:

Step 1 Ensure that the remote system has the printer installed and configured into the remote system's line printer spooler system.

Step 2 Gather the following information:

- The name you are giving to this printer.
- Whether or not you wish to make this device your system's default printer.
- The name of the remote system to which the printer is attached.
- The name of the remote printer.
- (Optional) The "cancel" model on the remote system.
- (Optional) The "status" model on the remote system.
- Whether or not you wish to allow any user to cancel any print request.
- Whether or not the remote printer is on a system using BSD (Berkeley Software Distribution) UNIX. Using BSD disables any `lp -o`*parm* options.

Step 3 Run SAM; enter:

```
/usr/bin/sam
```

During the process of adding and removing printers, SAM stops `lpss`. Print requests printing at the time `lpss` stopped might not complete successfully. Add a printer when there are no requests currently printing.

Step 4 Highlight Peripheral Devices and activate the Open control button.

Step 5 Highlight Printers and Plotters and activate the Open control button.

Step 6 Highlight Printers/Plotters and activate the Open control button.

Step 7 Choose Add a remote printer/plotter and the menu item associated with the printer interface type from the Actions menu.

Step 8 Fill in the printer interface dialog box fields and turn on/off check box values.

Activating the Help button from a dialog or message box gives you information about the attributes and tasks you perform from the currently displayed window.

Pressing the F1 key gives you context-sensitive information for the object field at the location of the cursor.

Step 9 Activate the OK control button.

To configure a remote printer into your `lpss`, you must be able to access the system with the printer via a local area network.

Remote printers cannot be members of a printer class.

Adding a network-based printer

To add a network-based printer using SAM:

Step 1 Ensure that the printer is

- Connected to the network according to the installation instructions shipped with the network-based printer.
- Connected to the network according to the installation instructions shipped with the network interface card for the printer.

Step 2 Gather the following information:

- The name you are giving to this printer.
- The printer node name.
- The model or interface the printer will use.
- The link-level address of the network card installed in the printer.
- The Internet Protocol (IP) address.
- The priority for this printer.
- (Optional) The class to which the printer is added.

In addition, decide whether or not you wish to make this device your system's default printer.

Step 3 Run SAM; enter:

```
/usr/bin/sam
```

Step 4 Highlight Peripheral Devices and activate the Open control button.

Step 5 Highlight Printers/Plotters and activate the Open control button.

Step 6 Choose Add a network-based printer then Add TCP-IP protocol printer from the Actions menu.

Step 7 Fill in the printer interface dialog box fields and turn on/off check box values.

Activating the Help button from a dialog or message box gives you information about the attributes and tasks you can perform from the currently displayed window.

Pressing the **F1** key gives you context-sensitive information for the object field at the location of the cursor.

Step 8 Activate the OK control button.

The software SAM needs to configure your network-based printer is shipped separately. Follow the instructions shipped with your printer to load the software.

Removing a printer

Perform the following steps to remove a printer from the Line Printer Spooler using SAM.

Step 1 Run SAM; enter:

```
/usr/bin/sam
```

Step 2 Highlight Peripheral Devices and activate the Open control button.

Step 3 Highlight Printers and Plotters and activate the Open control button.

Step 4 Highlight Printers/Plotters and activate the Open control button.

Step 5 Highlight the printer you want to remove in the object list.

Step 6 Choose Remove a printer/plotter from the Actions menu.

During the process of adding and removing printers, SAM stops `lpss`. Print requests printing at the time `lpss` stops might not complete successfully. Stop `lpss` when there are no requests printing.

SAM cancels all print requests in the request directory for the printer you are removing.

Starting the spooler

To start the Line Printer Spooler, perform the following steps:

Step 1 Run SAM; enter:

```
/usr/bin/sam
```

Step 2 Highlight Peripheral Devices and activate the Open control button.

Step 3 Highlight Printers/Plotters and activate the Open control button.

Step 4 Choose Start up printer spooler from the Actions menu.

Stopping the spooler

To stop the LP spooler, perform the following steps.

Step 1 Run SAM; enter.

```
/usr/bin/sam
```

Step 2 Highlight Peripheral Devices and activate the Open control button.

Step 3 Highlight Printers and Plotters and activate the Open control button.

Step 4 Choose Shut Down Printer Spooler from the Actions menu.

Printing on all printers stops. When SAM stops `lpss` there is no guarantee that print requests printing at the time complete successfully. Stop `lpss` when there are no requests currently printing.

Determining the status of the spooler

To determine the status of lpss using SAM, perform the following steps:

Step 1 Run SAM; enter:

```
/usr/bin/sam
```

Step 2 Highlight Peripheral Devices and activate the Open control button.

Step 3 Highlight Printers/Plotters and activate the Open control button.

The status area of the object list displays

```
Scheduler: RUNNING
```

or

```
Scheduler: STOPPED
```

Disabling a printer

Only disable a printer if there are no requests currently printing.

Perform the following steps to disable a printer using SAM:

Step 1 Run SAM; enter:

```
/usr/bin/sam
```

Step 2 Highlight Peripheral Devices and activate the Open control button.

Step 3 Highlight Printers/Plotters and activate the Open control button.

Step 4 Highlight the printer you want to disable in the object list.

Step 5 Choose Disable printer from the Actions menu.

Enabling a printer

To enable a printer using SAM, perform the following steps.

Step 1 Run SAM; enter:

```
/usr/bin/sam
```

Step 2 Highlight Peripheral Devices and activate the Open control button.

Step 3 Highlight Printers/Plotters and activate the Open control button.

Step 4 Highlight the printer you want to enable in the object list.

Step 5 Choose Enable printer from the Actions menu.

Changing a printer fence priority

To change a printer priority using SAM:

Step 1 Run SAM; enter:

```
/usr/bin/sam
```

Step 2 Highlight Peripheral Devices and activate the Open control button.

Step 3 Highlight Printers/Plotters and activate the Open control button.

Step 4 Highlight the printer for which you want to change the priority.

Step 5 Choose Modify fence priority from the Actions menu.

Step 6 Choose the new priority value from the Printer priority menu button.

Step 7 Activate the OK control button.

Using SPP-UX to manage printers

The SPP-UX commands method is provided for those who do not have access to SAM or choose not to use it.

Although SPP-UX commands require you to learn more details than SAM does, you might want to use SPP-UX commands to give you a greater degree of control over `lpss`. If SAM is not configured into your system, you have to use SPP-UX commands to control `lpss`.

If you want to use the data collection facility, you must enter the `lpsched -a` command to start data collection.

Adding a remote printer

To add a remote printer using SPP-UX commands, perform the following steps.

Step 1 Ensure that you have superuser capabilities.

Step 2 Stop the LP spooler with the `lpshut` command:

```
/usr/lib/lpshut
```

Stop the LP spooler when there are no requests currently printing.

Step 3 Add the printer to `lpss` using the `lpadmin` command.

When you use the `lpadmin` command, do not put any spaces between the options and their respective values.

The `lpadmin` command has the format

`/usr/lib/lpadmin -p`*pname* `-v`*devfile* `-m`*model* `[-d] [-g`*priority*`] [-ocm`*cmodel*`] [-osm`*smodel*`] [-orm`*remsys*`] [-orp`*rpname*`] [-ob3] [-orc]`

where:

pname

Is the name that you use to send print requests to this printer. Printer names can be up to 14 characters in length, and the characters must either be alphanumeric (A-Z, a-z, 0-9) or an underscore (_).

devfile

Is the device file associated with the printer. Since the printer is not physically connected to your local system, use the `/dev/null` device file.

model

Is the `/usr/spool/lp/model/rmodel`. `lpss` copies this into the /usr/spool/interface directory with the name you specified in *pname*.

`-d`

Specifies that you want this printer to be the system default printer.

priority

(Optional) Sets print priorities for multiple printers. You only need to use this option if you want your printer to have a priority other than zero.

cmodel

Is the `/usr/spool/lp/cmodel/rcmodel` to use when forwarding a `cancel` request to the remote system's `lpss`. `lpss` copies /usr/spool/lp/cmodel/rcmodel to the /usr/spool/lp/cinterface directory with the name you specified in *pname*.

smodel

Forwards a `status` request to the remote system's `lpss`. `lpss` copies /usr/spool/lp/smodel/rsmodel to the /usr/spool/lp/sinterface directory with the name you specified in *pname*.

remsys

Is the name of the remote system to which the printer is physically connected. Get the remote system name by entering the command `hostname`—with no options—on the system with the printer. The name of the remote system must be available to the local system, either from a name server or in the /etc/hosts file on the local system.

rpname

Is the printer name defined on the remote system.

Step 4 Allow print requests to enter the request directory for the newly added remote printer with the `accept` command:

`/usr/lib/accept` *pname*

pname is the local name given to this remote printer.

Step 5 Enable the newly added remote printer to process print requests with the `enable` command:

`/usr/bin/enable` *pname*

pname is the local name given to this remote printer.

Step 6 Start the line printer scheduler with the `lpsched` command:

```
/usr/lib/lpsched
```

A remote printer is a printer that is not physically connected to your system, but accessed by your system through a local area network (LAN). To configure a remote printer into your local `lpss`, you must have access to the remote system via a LAN.

Remote printers cannot be members of a printer class.

Adding a printer to `lpss` is not the same as adding a printer to your system. The first involves connecting the printer to your computer and configuring SPP-UX to communicate with the printer. The second involves configuring the software subsystem—known as the Line Printer Spooling System—that manages printer output.

Because the `lpadmin` constructs and modifies files that are used by the line printer scheduler, stop the scheduler when you use the `lpadmin` command to add a new printer.

You need to use only the `-ob3` option if your print request is printed on or passes through a system that uses the Berkeley Software Distribution (BSD) style `lpss`. BSD systems use three-digit (rather than four-digit) print request-ID numbers—these are the numbers returned when you send something to print. The `-ob3` option disables any `lp -o`*parm* options.

Use the `-orc` option if you want users to cancel only their own print requests.

Adding a network-based printer

To add a network-based printer or plotter using SPP-UX commands, follow the instructions shipped with the network-based printer or the network interface card for the printer.

The software needed to configure your network-based printer is shipped separately. Follow the instructions shipped with your printer to load the software and configure the printer.

Creating a printer class

To create a class of printers, use the `-c` option to the `lpadmin` command when you add a printer to `lpss` or after you add several printers.

Printer class restrictions and considerations

The following considerations and restrictions apply to printer classes:

- Printer classes cannot include remote printers.
- It is not necessary to specify the model and device file options because the printers have already been defined for `lpss`.
- Printer class names can be up to 14 characters in length, and the characters must either be alphanumeric (A-Z, a-z, 0-9) or an underscore (_).
- Class and printer names must be unique.
- A printer can only belong to one printer class at a time.
- A printer class must contain at least one printer.
- To remove a printer from a printer class
  - Remove the printer from `lpss`.
  - Re-add the printer without the printer class.

To create a printer class after you add several printers to `lpss`:

Step 1 Ensure that you have superuser capabilities.

Step 2 Stop `lpss`.

Stop the LP spooler when there are no requests currently printing.

Step 3 Create the printer class. Enter the following command for every printer you wish to add to a class of printers:

`/usr/lib/lpadmin -p`*`pname`* **`-c`***`classname`*

where

`-p` pname

Specifies name of the printer you want to add.

`-c` classname

Specifies name of the new printer class.

Step 4 Start `lpss`.

Step 5 Allow print requests to enter the request directory for the newly added printer class with the `accept` command:

`/usr/lib/accept` *classname*

classname is the name given to this printer class.

Removing a printer or printer class

To remove a printer or printer class using SPP-UX commands, perform the following steps.

Step 1 Ensure that you have superuser capabilities.

Step 2 Deny any further print requests with the `reject` command. The `reject` command has the format:

`/usr/lib/reject [-r` *"message"*`]` [*name*]

where:

message

Is the displayed message when users obtain status information about the printer and or printer class.

name

Is the name of the printer or printer class.

Step 3 If you are removing a printer class, skip this step and continue with step 4.

Disable the printer with the `disable` command. The `disable` command has the format:

`/usr/bin/disable [-r` *"message"*`] [-c]` *pname* [*pname*]

where:

message

Is the displayed message when users obtain status information about the printer.

`-c`

Cancels all print requests waiting to print on *pname*.

pname

Is the name of the printer to disable.

Step 4 Stop `lpss`:

`/usr/lib/lpshut`

Before you stop the line printer scheduler (spooling system), beware of the following:

- All printing stops until you restart the scheduler.
- Any print requests currently printing completely reprint when you restart the scheduler. This includes the print requests that were printing page 9,999 of a 10,000 page printout.

Step 5 Move all print requests in the request directory for the printer or printer class to another printer or printer class request directory to preserve the print requests in the request directory.

```
/usr/lib/lpadmin -xname
```

name is the name of the removed printer or printer class.

Step 6 (If you have just removed your only printer, omit this step.) Start `lpss`:

```
/usr/lib/lpsched
```

Because `lpadmin` deletes and modifies files examined by the line printer scheduler, stop the scheduler when you use the `lpadmin` command to remove the printer from `lpss`.

When you remove a printer class, the printers in it are not removed. You can still use them as individual printers. If the only printer in a printer class is removed, the printer class is removed also.

Accepting and rejecting print requests

To accept print requests for a printer or printer class, use the `accept` command. The accept(1M) command has the format:

`/usr/lib/accept` [*name*]

where *name* is the name of the printer or printer class whose request directory is enabled to receive print requests.

You can issue individual commands for each printer class or you can combine the printer classes in one command.

To reject print requests for a printer or printer class, use the `reject` command. The `reject` command has the format:

`/usr/lib/reject [-r"`*message*`"]` *name*

where:

message

Is the message displayed when users obtain status information about the printer or printer class.

name

Is the name of the printer or printer class whose request directory is prohibited from receiving print requests.

You can issue individual commands for each printer class or you can combine the printer classes in one command separated by spaces. If you combine them, you can also specify different

reasons for rejecting printer requests for different printers and printer classes.

Even if all printers that are members of a class are accepting requests, the class can still reject requests. Users need to specify a specific printer, not the class, in later print requests.

If all printers in a class are rejecting requests, but the class itself is accepting requests, the print requests remain in the request directory until at least one of the printers in the class begins to process print requests.

If you do not specify a reason, the status requests get the following response:

```
Printer is NOT ACCEPTING requests: Reason is
unknown.
```

Enabling or disabling a printer

When you disable a printer, any print requests waiting to print for that printer remain in the printer's request directory. When the printer is enabled again, the print begins. Any print requests printing at the time the disable command is issued completely reprint when the printer is enabled. If you wish to cancel all print requests for a printer at the time you disable it, use the `-c` option with the `disable` command.

To enable a printer to process print requests, use the `enable` command. The `enable` command has the following format:

`/usr/bin/enable` [*pname*]

pname is the name of the printer to be enabled.

You can issue individual commands for each printer or you can combine the printers in one command separated by spaces.

To disable a printer from processing print requests, use the `disable` command. The `disable` command has the format:

`/usr/bin/disable [-c [-r`*"message"*] [*pname*]

where:

`-c`

Cancels requests printing on the designated printer.

message

Is the message to display when users obtain status information about the printer.

pname

Is the name of the printer to be disabled to process print requests.

You can issue individual commands for each printer class or you can combine the printer classes in one command separated by spaces. If you combine them, you can also specify different reasons for disabling printer requests for different printers and printer classes.

Setting a printer fence priority

When you add a printer to `lpss`, the default priority is 0. To set or change a printer priority, perform the following steps.

Step 1 Ensure that you have superuser capabilities.

Step 2 Stop `lpss`:

```
/usr/lib/lpshut
```

Step 3 Set the priority for a particular printer with the `lpfence` command.

The `lpfence` command has the format:

`/usr/lib/lpfence` *pname priority*

where:

pname

Is the printer name.

priority

Is the minimum required priority a print request must have in order to be printed on printer *pname*. The value range is 0 (lowest) to 7 (highest).

Step 4 Restart `lpss`:

```
/usr/lib/lpsched
```

Starting and stopping `lpss`

Before you stop the line printer scheduler (`lpss`), beware of the following:

- All printing stops until you restart the scheduler.
- Any print requests currently printing are completely reprinted when you restart the scheduler. This includes the print requests that were printing page 9,999 of a 10,000 page printout.

- To report statistics about data flow through your `lpss`, you must tell `lpss` that you want it to keep track of these statistics by specifying the `-a` option when starting `lpss` with the lpsched command. The `-a` option tells `lpss` to log statistical information about its activities to the file /usr/spool/lp/lpana.log. The `lpana` command uses this file to report statistics.

To start `lpss`, use the `lpsched` command:

```
/usr/lib/lpsched
```

To stop `lpss`, use the `lpshut` command:

```
/usr/lib/lpshut
```

Canceling print requests

You can issue individual `cancel` commands for each print request or you can combine the print requests in one command separated by spaces.

You do not need superuser capabilities to use the `cancel` command.

To list print request identification numbers, use the `lpstat` command.

The `cancel` command has several options that allow you to

- Cancel all submitted print requests.
- Cancel all requests associated with a particular printer or printer class.

Here are a few helpful `cancel` options and their descriptions:

To cancel print requests, use the `cancel` command. The `cancel` command has the format:

`/usr/bin/cancel` *req-ID* [*printer*] `[-a]-e] [-u`*user*`]`

where:

req-ID

Is the print request identification number.

printer

Is the printer name.

`-a`

Removes all requests a user owns on the specified printer. The owner is determined by the user's login name and host name on the machine where the `lp` command was invoked.

`-e`

Empties the spool queue of all requests for the specified printer. Only users with superuser capabilities can use this option.

`-u` *user*

Removes any requests queued belonging to user. Multiple `-u` options are allowed. Only users with superuser capabilities can use this option.

Moving all print requests

To move all print requests to another request directory using SPP-UX commands, perform the following steps:

Step 1 Ensure that you have superuser capabilities.

Step 2 Prohibit further requests from entering the request directory with the `reject` command. The `reject` command has the format:

`/usr/lib/reject` *pname*

where *pname* is the name of the rejected printer or printer class.

You can issue individual commands for each printer class or you can combine the printer classes in one command separated by spaces.

Step 3 Disable the printer with the `disable` command. The `disable` command has the following format:

`/usr/bin/disable` [`-r`*"message"*] *pname*

where:

message

Is the message displayed when users obtain status information about the printer(s).

pname

Is the name of the disabled printer.

You can issue individual commands for each printer class or you can combine the printer classes in one command separated by spaces.

Step 4 Stop `lpss`:

`/usr/lib/lpshut`

Step 5 Relocate all of the print requests in the request directory to another request directory with the `lpmove` command:

`/usr/lib/lpmove` *source dest*

where:

source

Is the disabled printer or printer class print directory.

dest

Is the printer or printer class request directory to receive the print requests.

Step 6 Restart the line printer scheduler with the `lpsched` command:

```
/usr/lib/lpsched
```

Step 7 If the *source* printer or printer class is made available to receive print requests:

1. Reenable the printer(s) to process print requests with the `enable` command. The `enable` command has the format:

   `/usr/bin/enable` *pname*

   where *pname* is the name of the enabled printer.

   You can issue individual commands for each printer or you can combine the printers in one command separated by spaces.

2. Reenable the printer or printer class request directory to accept print requests with the `accept` command. The `accept` command has the format:

   `/usr/lib/accept` *name*

   where *name* is the name of the enabled printer or printer class request directory.

You can issue individual commands for each printer class or you can combine the printer classes in one command.

Moving selected print requests

To move selected print requests to another request directory using SPP-UX commands, perform the following steps.

Step 1 Ensure that `lpss` is running.

Step 2 Move selected print requests using the `lpalt` command. The `lpalt` command has the format:

`/usr/bin/lpalt` *source* `-d`*dest*

where:

source

Is the identification number of the print request to be moved.

dest

Is the printer or printer class request directory to receive the print request specified by *source*.

The `lpalt` command cannot be used to alter a print request currently printing.

The `lpalt` command alters a print request from a remote printer

- Only if the print request is owned by the user who issues the `lpalt` command.
- Only if the print request is not currently printing.

Viewing printer and print requests status

To view the status of printers and print requests, use the `lpstat` command. The `lpstat` command has the format:

```
/usr/bin/lpstat [-t]
```

With no options, lpstat displays the status of all requests made by the user. The `-t` option lists the following additional information:

- Status of `lpss`.
- System default printer.
- List of class names and members.
- List of printers and associated device files.
- Status of each print request directory—accepting or rejecting. If a reason was specified when the requests were rejected the reason is displayed.
- Status of each printer—enabled or disabled. If a reason was specified when the printer was disabled, the reason is displayed.
- Priority for each printer.
- List of print requests for each printer that includes the following attributes for each print request:
  - Print request identification number
  - Name of user that submitted the print request
  - Priority
  - Date and time submitted
  - File name
  - Size

The `-t` option of the `lpstat` command is very detailed. For information on other options of this command, refer to the lpstat(1) man page.

Sample lpstat output

The following output shows the summary information available from the `lpstat` command:

```
lpstat -t
scheduler is running system default destination: laser: laser1 laser2 phred
device for letterhead: /dev/null
device for check_printer: /dev/null
device for laser1: /dev/lj1
      remote to: shasta on mountian
device for laser2: /dev/lj2
      remote to: hood on mountian
device for phred: /dev/lj3
device for invoices: /dev/invoices
laser1 accepting requests since Apr 18 14:46
laser2 accepting requests since May 13 14:08
phred accepting requests since Apr 18 14:46
laser accepting requests since Apr 18 14:46
letterhead accepting requests since Apr 18 14:46
invoices accepting requests since Apr 18 14:56
check_printer accepting requests since May 3 14:57
printer laser1 now printing
laser1-1807. enabled since Apr 23 13:47
      fence priority: 0
printer laser2 now printing
laser2-1809. enabled since Apr 23 13:47
      fence priority: 0
printer phred is idle. enabled since Apr 18 14:46
      fence priority: 3
printer letterhead now printing
letterhead-1810. enabled since Apr 23 13:47
      fence priority: 4
printer invoices is idle. enabled since Apr 19 10:24
      fence priority: 0
printer check_printer is idle. enabled since Apr 18 14:56
      fence priority: 0
```

Changing print request priority

The two primary reasons for changing a print request priority include

1. To move the print request ahead of other requests within the request directory.

   For example, you can elevate the priority of your print request to be higher than that of the large print request ahead of yours. When the line printer scheduler selects the next print request to send to the printer, it takes the one with the highest priority.

   Once a print request begins to print, it does not yield to a print request of higher priority. In this case, you can move your print request to another printer.
2. To match or exceed the printer's priority, enabling the print request to print.

Unless you tell it otherwise, the `lp` command—used to print files—assigns your print request a priority equal to the printer's priority setting. If your print request is assigned to a printer class, `lp` uses the highest printer priority setting among all the printers in the class.

To change the priority of a print request, use the `lpalt` command. The `lpalt` command has the format:

`/usr/bin/lpalt` *preq-ID* `-p`*new_priority*

where:

preq-ID

Is the print request identification number for the targeted print request.

new_priority

Is the new priority. Valid values are 0 to 7.

Displaying printer statistics

To display statistics about printer activity, use the `lpana` command. The `lpana` command has the format:

`/usr/lib/lpana` [`-d`*dest*]

where *dest* defines the printer or printer class for which statistics are displayed.

By default, `lpana` reports statistics for all printers and printer classes (optional).

You must start `lpss` with the `/usr/lib/lpsched -a` command to create a log of activity in the /usr/spool/lp/lpana.log file.

System accounting

8

Overview

SPP-UX allows concurrent sharing of computer resources among users; several users can be logged in, all sharing processors, disk space, and memory. SPP-UX system accounting provides the means to:

- Monitor disk space usage for individual users.
- Record connect session data (logins/logouts).
- Collect resource utilization data (such as memory usage and execution times) for individual processes.
- Charge fees to specific users.
- Generate summary files and reports that can be used to analyze system performance and bill users for resource consumption.

SPP-UX system accounting allows you to accomplish accounting tasks through a number of versatile commands. This chapter illustrates the use of these commands.

Installation and usage

The purpose of this section is to show you:

- What you must do to get system accounting running on your system
- How system accounting automatically creates daily and monthly accounting data and reports

After reading this section, you should be able to install system accounting on your system. Once properly installed, system accounting automatically generates daily and monthly accounting data and reports.

How to install system accounting

Not all users require accounting services. For this reason, system accounting is provided as an option; if you want to use system accounting, you must install it yourself. The installation procedure is covered here.

There are three steps in the installation process:

Step 1 Update /etc/rc.

Step 2 Create crontab file entries.

Step 3 Set the `PATH` environment variable for accounting commands.

Each step must be carried out to ensure that system accounting automatically creates daily and monthly accounting information. Detailed descriptions of each step follow.

Update /etc/rc

The system initialization shell script `rc` must be updated to automatically start system accounting when the system is started in multiuser mode. This requires adding the following entry in the localrc section of /etc/rc:

```
/bin/su - adm -c /usr/lib/acct/startup
```

Create crontab file entries

To automate the daily and monthly creation of accounting data, you should create a crontab file that `cron` can use to automatically run certain accounting commands. `runacct`, the main accounting shell script, should be executed daily (during nonprime hours) to generate daily accounting reports.

The dates and times shown in the following crontab entries are only suggestions; tailor your crontab entries to suit your needs.

This process consists of the following steps:

Step 1 Log in to SPP-UX as user adm.

Step 2 Use an editor to add the following accounting commands to the crontab file:

```
0 4 * * 1-6 /usr/lib/acct/runacct 2> /usr/adm/acct/nite/fd2log
0 2 * * 4 /usr/lib/acct/dodisk
5 * * * * /usr/lib/acct/ckpacct
15 5 1 * * /usr/lib/acct/monacct
```

The first line executes `runacct` at 4:00 a.m. every Monday through Saturday. Error messages are redirected to the file

/usr/adm/acct/nite/fd2log, if any errors occur while `runacct` executes.

In the next line, `dodisk` creates total accounting records that summarize disk space usage for individual users at 2:00 a.m. every Thursday morning.

The third line invokes `ckpacct` at five minutes into every hour. `ckpacct` ensures that the process accounting file, pacct, does not get too large. `ckpacct` should be executed hourly.

The forth line executes `monacct` at 5:15 a.m. on the first day of every month. `monacct` generates a monthly total report and total accounting file.

Step 3 Execute the `crontab` command, specifying the file created in Step 2 as input. This step insures that the crontab file created in Step 2 is scanned by `cron` every minute. After invoking this command, the file created in Step 2 file is stored in the file /usr/spool/cron/crontabs/adm.

At this point, you are finished creating crontab file entries. If you ever want to change the entries, reedit the file created in Step 2 and use the `crontab` command again. See the crontab(1) man page for more information.

Set PATH for accounting commands

Finally, you should set the `PATH` environment variable in /usr/adm/.profile so that system accounting knows where to look for commands. `PATH` should be set as follows:

```
PATH=/usr/lib/acct:/bin:/usr/bin:/etc:/usr/adm
```

How daily accounting works

When SPP-UX boots in multiuser mode, the system initialization shell script `rc` executes the `startup` command. `startup` performs the following functions:

- Calls `acctwtmp` to add a boot record to the wtmp file. This record is marked by storing `acctg on` in the device name field of the wtmp record.
- Turns process accounting on via the `turnacct on` command.
- Next `accton` executes with the filename argument `/usr/adm/pacct`.
- Removes work files left in the sum directory by `runacct`.

A report of the previous day's accounting information can be created by running `prdaily`. This step is omitted the first day that system accounting is installed, because the previous day's

accounting information does not exist yet. However, after `runacct` has been executed, `prdaily` generates valid reports.

The `ckpacct` command is executed every hour via `cron` to insure that the process accounting file pacct does not become too large. If pacct grows past a set maximum number of blocks, `turnacct switch` is invoked, creating a new pacct file. Other conditions may also limit the size of the process accounting file or turn process accounting off; for more details, refer to the discussion of `ckpacct` in the "Process accounting" on page 180. The advantage of having several smaller pacct files is that `runacct` can be restarted faster if a failure occurs while processing these records.

The `chargefee` program can be used to charge fees to users. It adds records to the file fee. These records are processed during the next execution of `runacct` and merged in with total accounting records.

`runacct` is executed via `cron` each night. It processes the active fee file and the process, connect session, and disk total accounting files. It produces command and resource-usage summaries by login name.

When the system is turned off using `shutdown`, the `shutacct` command is executed. The purpose of `shutacct` is to stop system accounting. `shutacct` performs the following functions:

- Writes a termination record to wtmp via the command `acctwtmp`. This record is marked by having `acctg off` in the device name field.
- Turns process accounting off by calling `turnacct off`.

Overview of system accounting

In this section, the intrinsics of system accounting are examined. Key terms are defined, commands are introduced, system data flow is described, and finally, you are shown the login and directory structure of system accounting.

Definitions

The following terms are specific to system accounting.

Prime/nonprime connect time

Prime time is the time during the day when the computer system is most heavily used; for example, from 9:00 a.m. to 5:00 p.m. Nonprime time is the remaining time during the day when the system is less heavily used; from 5:00 p.m. to 9:00 a.m. in this example.

When reporting computer time usage, system accounting distinguishes between prime and nonprime time usage. You can specify prime and nonprime time on your system by editing the file /usr/lib/acct/holidays. For details on the holidays file, refer to the section "Updating the holidays file" on page 208.

Prime time is in effect only on weekdays (Monday through Friday); nonprime time is in effect during the weekends (Saturdays and Sundays) and on any holidays specified in the holidays file.

Process accounting records

Once system accounting is installed and turned on, the following occurs: whenever a process terminates, the kernel writes a process accounting record for the terminating process into the current process accounting file, /usr/adm/pacct by default. (You can specify that a file other than pacct be used as the process accounting file, if desired.)

A process accounting record contains resource-usage data for a single process; it summarizes how much of the various resources the process used during its lifetime. Examples of information contained in process accounting records are:

- The user ID of the process's owner
- The name of the command that spawned the process
- The amount of time it took the process to execute

For greater detail on the contents and format of process accounting records, see the acct(4) man page.

Total accounting records

These records, created by various accounting commands, contain summary accounting information for individual users. They provide the basic information for many reports generated by system accounting. Some examples of information contained in these records are:

- The ID and user name of the user for whom the total accounting record was created
- The total number of processes that the user has spawned during the accounting period for which the total accounting record was created
- Fees for special services rendered to this user

The exact contents and format of total accounting records can be found in the acct(4) man page. In addition, commands covered in later sections of this chapter show how these records are created and used by system accounting.

Introduction to commands

System accounting provides many versatile commands to accomplish numerous, varied tasks. There are commands that create data, display data, remove data, merge data, and summarize and report data. In addition, the output of one command may become the input to other commands.

System accounting commands can be logically categorized into six basic command groups:

- Installation
- Disk usage accounting
- Connect session accounting
- Process accounting
- Charging fees
- Summarizing and reporting accounting information

Descriptions of these command groups, along with a brief synopsis of each command, follow.

Installation commands

These two commands insure that system accounting is properly installed. They are used to turn accounting on when SPP-UX is powered up and turn accounting off when the system is shut down. They may also do some file cleanups.

`startup`

Starts accounting when SPP-UX is switched to multiuser mode. `startup` is invoked from /etc/rc.

`shutacct`

Turns off accounting when SPP-UX is turned off via the `/etc/shutdown` shell.

Disk space usage accounting commands

These four commands produce disk usage accounting information; they show disk space usage (in blocks) for individual users. They also produce total accounting records.

`acctdusg`

Shows how many blocks of disk space users are consuming. `acctdusg` takes its input from a list of path names created by `find`.

`diskusg`

Shows how many blocks of disk space users are consuming. `diskusg` looks at the inodes of the file system to create its output.

`acctdisk`

Produces total accounting records. Its input is supplied (either directly or indirectly) from `acctdusg` or `diskusg`.

`dodisk`

Produces total accounting records by using the `diskusg` and `acctdisk` commands. `dodisk` is normally invoked by `cron`.

Connect session accounting commands

Independent of system accounting, the programs `login` and `init` record connect sessions by writing records into /etc/wtmp. System accounting commands can display or fix this file, and can produce total accounting records for this file.

There are six commands:

`acctwtmp`

Writes records to wtmp.

`fwtmp`

Displays the information contained in wtmp.

`wtmpfix`

Normalizes connect session records that span date changes (see the date(1) man page). Also validates login names in connect session records.

`acctcon1`

Summarizes wtmp in ASCII readable format, producing one line per connect session.

`acctcon2`

Takes input of the format produced by `acctcon1` and produces total accounting records as output.

`prctmp`

Displays the session record file, normally called /usr/adm/acct/nite/ctmp.

Process accounting commands

When process accounting is turned on, the kernel writes a process accounting record to pacct whenever a process terminates. A number of accounting commands summarize and report this accounting information. In addition, certain commands turn

process accounting on or off and insure that pacct does not become too large. The process accounting commands are:

`accton`

Turns process accounting on or off, depending on whether or not a filename argument is supplied with the command. If no filename is given, then process accounting is turned off; the kernel stops writing process accounting records to pacct. If a filename is specified, then the kernel starts writing process accounting records to the specified filename.

`accton` uses the system call `acct` to turn process accounting on or off. Only the superuser can execute `accton`.

`ckpacct`

Checks the size of the process accounting file pacct. If pacct becomes too large, then a new pacct file is created via `turnacct switch`. If disk space becomes critically short, then process accounting is turned off until sufficient space is available. This command is normally invoked by `cron`.

`turnacct [on || off || switch]`

Performs one of three functions, depending on which argument (`on`, `off`, or `switch`) is specified. `turnacct on` turns process accounting on by calling `accton` with the default filename argument /usr/adm/pacct; `turnacct off` turns process accounting off by calling `accton` with no filename argument; `turnacct switch` renames the current pacct file (so that it is no longer the current process accounting file) and creates a new, empty pacct file.

`acctcom`

Displays process accounting records contained in pacct (or any specified file).

`acctcms`

Takes pacct as input, and produces summary accounting information by command, as opposed to by process.

`acctprc1`

Produces readable process accounting information, mainly for input into `acctprc2`.

`acctprc2`

Takes input of the form produced by `acctprc1` and produces total accounting records.

Charging fees command

Occasionally, you may want to charge a user for something. For example, you might charge fees to users for fixing damaged files.

The `chargefee` command allows you to charge fees to specific users.

Accounting summary commands

This group of commands summarize the data created through the command groups. The following five commands are probably used most frequently; they represent the highest level of accounting commands:

`prtacct`

Takes as input total accounting records and displays the records in ASCII readable format.

`acctmerg`

Combines the contents of separate total accounting files into a single total accounting file. This command allows the merging of disk, process, and connect session total accounting records.

`runacct`

Is the main accounting shell script. Normally invoked daily by `cron`, this command processes disk, connect session, process, and fee accounting information and produces summary files and reports. It accomplishes its task by proceeding through various states. In each successive state it invokes accounting commands to perform a specific task. For example, in one state, total accounting records for connect sessions are created; in another, disk, connect session, process, and fee total accounting records are merged to create one total accounting file.

Accounting reporting commands

The following two commands format and report the data created through command groups:

`prdaily`

Invoked by `runacct` to format a report of the previous day's accounting data; the report is in the file /usr/adm/acct/sum/rpt *mmdd* where *mmdd* is the month and day of the report. `runacct` may also be used to display a report of the current day's accounting information.

`monacct`

Invoked once a month or accounting period, this command summarizes daily accounting files and produces a summary file for the accounting period.

Login and directory structure

Accounting uses a unique login name and directory structure. This section describes the login characteristics and directory structure.

Logging in

The login name for system accounting is adm; the user ID for adm is 4. The adm login is a member of the user group adm; the group adm also has a group ID of 4.

The home directory for the adm login is /usr/adm. You log in to system accounting the same way you do for any account; supply the login name to the SPP-UX login prompt:

```
login: adm
```

Note

Use a password with the adm login. The integrity of accounting data files must be maintained if system accounting is to generate accurate reports.

Directory structure

System accounting uses a multilevel directory structure to organize its many accounting files. Each directory in this structure stores related groups of files, commands, or other directories. (Refer to the section "System accounting files" on page 215 for definitions of the accounting data files.)

Table 9 describes directories used by the accounting system.

Table 9 Accounting system directory structure

Directory	Contents
/usr/adm	Contains all active data-collection files, such as pacct and fee.
/usr/adm/acct	Contains the nite, sum, and fiscal directories described below.
/usr/adm/acct/nite	Stores data files that are processed daily by `runacct`.
/usr/adm/acct/sum	Contains cumulative summary files updated by `runacct`.
/usr/adm/acct/fiscal	Contains periodic (monthly) summary files created by `monacct`.
/usr/lib/acct	Contains system accounting commands.
/etc	Contains wtmp, and shell scripts `rc` and `shutdown`.

Disk space usage accounting

System accounting provides the means to monitor disk space utilization for individual users. In this section, disk space usage accounting commands are explained.

Disk usage commands provide two main functions: they report disk usage (in blocks) for individual users and create disk total accounting records (supplied as inputs to commands such as `prtacct` or `runacct`).

Reporting disk space usage

Two commands, `acctdusg` and `diskusg`, report disk usage for individual users; both commands show the number of disk blocks allocated to specific users. However, each command has slightly different options. In addition, each differs in the manner in which it produces accounting information.

`acctdusg`

`acctdusg` takes a list of path names, usually created by the `find` command, from standard input. For each file in the list, `acctdusg` identifies the owner of the file, computes the number of blocks allocated to the file, and adds this amount to a running total for the file's owner. When finished looking through the list, `acctdusg` displays the information accumulated for each user: user ID, user name, and number of blocks used.

This command is useful for reporting disk usage information for specific users or files. For example, suppose you want to know how many blocks of disk space you are using: your user ID is 351, user name is bill, and your home directory is /users/pseudo/bill. The following example illustrates how to use the `find` and `acctdusg` commands to show this information:

```
find /users/pseudo/bill -hidden -print > bills.files
acctdusg < bills.files
00351     bill                30
00351 bill 30 rm bills.files
```

In this example, `bill` is using 30 blocks of disk space. The series of commands shown could easily have been combined into the following line:

```
find $HOME -hidden -print | acctdusg
00351   bill                30
```

To use `acctdusg` to generate disk usage information for all files in the system enter the following command:

```
find / -hidden -print | acctdusg
00350 fred 11
00351 bill 30
00352 mike 17
00353 sarah 13
00354 molly 18
00000 root 3
00004 adm 36
00001 bin 2434
```

Two options are included with `acctdusg`:

`-u` *no_owners*

If `-u` is given, then path names of the files for which no owner is found are written into the file *no_owners*. This option could potentially find users who are trying to avoid disk charges.

`-p` *p_file*

The password file /etc/passwd is the default file used by `acctdusg` to determine ownership of files. If the `-p` option is used, then `acctdusg` uses *p_file* instead. This option is not needed if your password file is /etc/passwd.

The shell script `grpdusg`, provided in the section "Sample accounting shell scripts" later in this chapter, displays disk accounting information for users in a given group. It illustrates the use of the `-u` option with `acctdusg`.

`diskusg`

This command reports disk usage information in the same format as `acctdusg`; user ID, user name, and total disk blocks used. However, `diskusg` generates disk accounting information by looking through the inodes of a specified device file. (See the inode(4) man page for more information on inodes and special files.) Therefore, `diskusg` is faster and more accurate than `acctdusg`.

The syntax of the `diskusg` command is:

`diskusg` [*options*] [*files*]

It generates a disk usage report from data in *files*, if specified; otherwise standard input is used. `diskusg` is normally invoked with the *files* argument. When specified, *files* are the special file names of the devices containing the inode information used by diskusg to generate its report. *files* is normally a device file from the /dev directory.

The following options may be used with `diskusg`:

`-s`

Tells `diskusg` that input is in `diskusg` output format, and that all lines for a single user should be combined into a single line. This option is used to merge data from separate files, each containing the output from using `diskusg` on different devices.

`-v`

Is useful for finding users who are trying to avoid disk space accounting charges. When this option is specified, `diskusg` writes records to stderr (standard error output) showing the special file name, inode number, and user ID of files that apparently have no owner.

`-i` *fnmlist*

Causes `diskusg` to ignore the data on those file systems whose file system name is in *fnmlist*. *fnmlist* is a list of file systems separated by commas or enclosed within quotes.

`-p` *p_file*

Is the same as the `-p` option of `acctdusg`.

`-u` *u_file*

Produces exactly the same output as the `-v` option. The difference between the two options is that `-v` writes its output to stderr; this option writes its output to the file *u_file*.

The output of `diskusg` is normally used by `acctdisk` to create disk total accounting records. In addition, `diskusg` is normally called by `dodisk`.

The following example illustrates using `diskusg` to create disk usage information for all users whose files reside on the disk whose device file is /dev/rdsk/1s0. This is the same filesystem used in the previous `acctdusg` example

```
diskusg /dev/rdsk/1s0
0 root 10616
1 bin 778
4 adm 96
350 fred 14
351 bill 32
352 mike 20
353 sarah 16
354 molly 22
355 julie 2
```

```
501 guest 2
```

The differences between `diskusg` and `acctdusg` are best illustrated by comparing their outputs:

- `acctdusg` places leading zeros on user IDs; `diskusg` does not.
- `acctdusg` counts files only under each user's `$HOME` directory. Files that users own in directories other than their home directory (for example, files in the /tmp directory) are counted as files with no owner.
- Two extra users, julie and guest, show up in the output of `diskusg` when compared with the output from `acctdusg`. This occurred because the directories of these two users were empty; therefore, no disk usage totals were generated by `acctdusg`. However, `diskusg` looked at inodes and saw that julie and guest were actually using two blocks for the directories themselves.
- If two or more users have links to a particular file, then `acctdusg` prorates disk space usage for the file between each user. For example, if three users had a link to a 300-block file called skurbnich.dat, each user would be charged for 100 blocks of this file.

Creating disk usage totals accounting records

Two commands are used to create total accounting records for disk usage: `acctdisk` and `dodisk`.

`acctdisk` command

`acctdisk` uses standard input records of the format produced by `acctdusg` and `diskusg`. From these records, `acctdisk` produces disk total accounting records that may be inputs to `prtacct` or `runacct`.

The following command writes disk total accounting records to the file disktacct for all users in the group pseudo:

```
find / -group pseudo -print | acctdusg | acctdisk > disktacct
```

The next example generates disk total accounting records for all users who have files on the disk /dev/rdsk/1s0. The total accounting records are written to the file disktacct.

```
diskusg /dev/rdsk/1s0 | acctdisk > disktacct
```

`acctdisk` has no options and is normally invoked by `dodisk`.

dodisk command

`dodisk` is normally invoked by `cron` to create disk total accounting records for daily usage by system accounting. The syntax for `dodisk` is:

`dodisk [-o] [`*files...*`]`

In the default case, `dodisk` creates disk total accounting records on the special files whose names are stored in /etc/checklist; the special file names are supplied as input to `diskusg`, which pipes its output to `acctdisk`, which in turn creates total accounting records.

If the `-o` option is used, `dodisk` creates total accounting records more slowly by using `acctdusg` instead of `diskusg`.

If *files* are used, disk accounting is done on these file systems only. When the `-o` option is used, *files* should be mount points of mounted file systems; if omitted, *files* should be the special file names of mountable file systems.

Refer to the "Installation and usage" on page 159 for more information on how `dodisk` should be invoked by `cron`.

It is possible for malicious users to defeat disk space accounting by giving their files to other users with the `chown` command or system call (by default, all users can execute them). To avoid this, take away the ability to use these commands from some or all users with the `setprivgrp` command. To let only the superuser use the change-ownership abilities, add the following line to /etc/rc:

```
setprivgrp -n CHOWN
```

To let one or more groups of users use the change-ownership abilities, add a line for each group to /etc/rc, similar to the following:

```
setprivgrp CHOWN
```

Taking away the change-ownership ability may cause problems when running some commands or applications.

Connect session accounting

Whenever a user logs in to or out of SPP-UX, the program login records the connect session in the file /etc/wtmp. Records in wtmp contain the following information:

- The terminal name on which the connect session occurred
- The login name of the user
- The current time/date at login or logout

- Other status information (see the utmp(4) man page for details)

System accounting provides commands that allow you to write records to wtmp, to display and manipulate wtmp, and to create total accounting records from wtmp. These commands are covered in this section.

Writing records to wtmp (acctwtmp)

The command `acctwtmp` allows you to write records to wtmp. `acctwtmp` is invoked by `startup` and `shutacct` to record when system accounting was turned on and off, respectively. The format of the command is:

`acctwtmp` *"reason"*

where *reason* is a string describing the reason for writing the record to wtmp. `acctwtmp` does not directly write records to wtmp: it writes a record containing the terminal name, current time, and reason string to standard output. To actually write the record to wtmp you must append the output from `acctwtmp` to the wtmp file as follows:

```
acctwtmp "reason" >> /etc/wtmp
```

The *reason* string may be any combination of letters, numbers, spaces, and the dollar sign ($), but may not exceed 11 characters in length. (*reason* must be enclosed in double quotes as shown.)

Displaying connect session records (fwtmp)

To display the contents of wtmp, you can use the command `fwtmp`. When no options are used, `fwtmp` uses standard input records of the format contained in wtmp; it writes to standard output the ASCII readable equivalent of the input records. Two alternatives exist for the output from this command:

- The output of this command can be edited, via an editor such as `vi`, and then rewritten to wtmp using special `fwtmp` options described below.
- The output can be supplied as input to commands which convert the information to total accounting records.

The syntax of `fwtmp` is:

```
fwtmp [-ic]
```

If no option is specified for the `fwtmp` command, then input is in binary format and is to be converted to ASCII readable format. The various combinations of the options `i` and `c` provide other

combinations of input and output formats. The possible options are described below:

`-ic`

Input is in ASCII readable form and is to be converted to binary form. This is essentially the opposite of using `fwtmp` without any options.

`-i`

Both input and output are in ASCII readable format. This is the same as performing an ASCII to ASCII copy.

`-c`

Both input and output are in binary format; a binary-to-binary copy.

Figure 8 shows the output produced by `fwtmp`.

```
fwtmp < /etc/wtmp
           system boot 0     2 0000 0000 479472540 Mar 12 03:49:00 1995
root  co   console     0     7 0000 0000 479475173 Mar 12 04:32:53 1995
           acctg on    0     9 0000 0000 479493135 Mar 12 09:32:15 1995
mike  a1   ttya1      352   7 0000 0000 479493590 Mar 12 09:40:00 1995
mike  a1   ttya1      352   8 0011 0000 479496000 Mar 12 10:20:00 1995
sarah 07   tty07      353   7 0000 0000 479518335 Mar 12 16:32:15 1995
bill  10   tty10      351   7 0000 0000 479521475 Mar 12 17:24:35 1995
sarah 07   tty07      353   8 0011 0000 479522478 Mar 12 17:41:18 1995
bill  10   tty10      351   8 0011 0000 479526487 Mar 12 18:48:07 1995
      co   console     0     8 0011 0000 479526488 Mar 12 18:48:08 1995
           acctg off   0     9 0000 0000 479526493 Mar 12 18:48:13 1995
           system boot 0     2 0000 0000 479389800 Mar 12 05:00:00 1995
```

Figure 8 `fwtmp` output

Table 10 describes each column in the `fwtmp` report.

Table 10 `fwtmp` report description

Column	Description
1	The login name of the user who logged in or out.
2	/etc/inittab ID (this is usually the number of the line on which the connect session took place).
3	The name of the device on which the connect session occurred.
4	Process ID of the user who logged in or out.

Table 10 `fwtmp` report description —(continued)

Column	Description
5	Entry type. This field contains information on the type of record — for example, it shows whether the record is a login record (entry type=7), logout record (entry type=8), or if the record was written by `acctwtmp` (entry type=9). See the utmp(4) man page for more details on this field.
6-7	Exit status for connect session; see the login(1) and utmp(4) man pages for details.
8	Time the entry was made (in elapsed seconds since January 1, 1970).
9-12	The equivalent of column 8 in date/time format showing month, day, time of day (in 24-hour format), and year.

Fixing wtmp errors with wtmpfix

When a user logs into SPP-UX, the `login` program stores the value seven (7) in the entry type field of the connect session record. When the same user logs out, an entry type of eight (8) is recorded. You can see this by examining the sample output created by `fwtmp` in the previous section. Note that in the example, login records precede their corresponding logout records in chronological order.

Occasionally, this time-stamped ordering becomes inconsistent: logout records might precede login records. This occurs when the date and time are reset while users are still logged in. When this happens, the commands that create connect session total accounting records do not work properly.

The `wtmpfix` command fixes corrupted wtmp files. `wtmpfix` takes wtmp binary records as input and corrects the time/date stamps to be consistent; its standard output is also binary wtmp records. The syntax for `wtmpfix` is:

`wtmpfix` [*files*]

If *files* is given, then input is taken from *files*. A dash (-) can be used in place of *files* to indicate standard input. If you specify wtmp as both input to and output from this command, wtmp is destroyed. Therefore, take care not to destroy wtmp. The following example shows how to properly fix wtmp using `wtmpfix`.

```
wtmpfix /etc/wtmp > wtmp.temp
fwtmp -c < wtmp.temp > /etc/wtmp
rm wtmp.temp
```

Creating total accounting records

This final set of connect session accounting commands is used to create connect session total accounting records.

acctcon1 command

`acctcon1` converts a sequence of login/logoff records (of the format contained in wtmp) read from its standard input to a sequence of records, one per login session. Its input is normally redirected from wtmp; its output is columnar ASCII and can be supplied as input to prctmp or acctcon2.

Figure 9 illustrates using `acctcon1` by first displaying the contents of wtmp with `fwtmp`, and then using `acctcon1` to create a connect session summary file.

```
fwtmp < /etc/wtmp
             system boot   0     2 0000 0000 479472540 Mar 12 03:49:00 1995
root     co  console       0     7 0000 0000 479475173 Mar 12 04:32:53 1995
             acctg on      0     9 0000 0000 479493135 Mar 12 09:32:15 1995
mike     a1  ttya1        352   7 0000 0000 479493590 Mar 12 09:40:00 1995
mike     a1  ttya1        352   8 0011 0000 479496000 Mar 12 10:20:00 1995
sarah    07  tty07        353   7 0000 0000 479518335 Mar 12 16:32:15 1995
bill     10  tty10        351   7 0000 0000 479521475 Mar 12 17:24:35 1995
sarah    07  tty07        353   8 0011 0000 479522478 Mar 12 17:41:18 1995
bill     10  tty10        351   8 0011 0000 479526487 Mar 12 18:48:07 1995
         co  console       0     8 0011 0000 479526488 Mar 12 18:48:08 1995
             acctg off     0     9 0000 0000 479526493 Mar 12 18:48:13 1995
acctcon1 < /etc/wtmp
20095488     353   sarah 1665   2478 479518335 Tue Mar 12 16:32:15 1995
521012224    352   mike                479493590 Tue Mar 12 09:40:00 1995
520095488    351   bill  0      5012 479521475 Tue Mar 12 17:24:35 1995
521011712    0     root  41047  6488 479475173 Tue Mar 12 04:32:53 1995
```

Figure 9 Creating `acctcon1` output

Table 11 describes each column in the `acctcon1` report.

Table 11 `acctcon1` report

Column	Description
1	The device address (in decimal equivalent of major/minor device address) at which the connect session occurred
2	The user ID for the connect session record
3	The login name for the user
4	The number of prime connect time seconds used during the connect session
5	The number of nonprime connect seconds used during the connect session
6	The connect session starting time (in seconds elapsed since January 1, 1970)
7-11	The conversion of column six to date/time format showing month, day, time of day (in 24-hour format), and year

In addition to its normal usage, `acctcon1` has four options:

`-p`

This option tells `acctcon1` not to produce one record per connect session. Instead, `acctcon1` simply echoes its input; one line per wtmp record, showing line name, login name, and time (in both seconds and day/time format). Using this option is similar to using `fwtmp`, except that this option does not show status information, whereas `fwtmp` does.

`-t`

`acctcon1` maintains a list of lines on which users are logged in. When it reaches the end of its input, it emits a session record for each line that still appears to be active. It normally assumes that its input is a current file, so that it uses the current time as the ending time for each session in progress. The `-t` flag causes it to use, instead, the last time found in its input, thus assuring reasonable and repeatable numbers for noncurrent files.

`-l` *file*

This option places a line usage summary report in *file*. This report shows each line's name, number of minutes used, percentage of total elapsed time used, number of sessions charged, number of logins, and number of logins and logoffs. This report can be used to keep track of line usage, identify bad lines, and find software/hardware oddities. Note that hang-up, termination of `login`, and termination of the login shell each generate logoff records; therefore, the number of logoffs is often three to four times the number of connect sessions.

-o file

Using the `-o` option (for example, `acctcon1 -o f_overall`) causes *file* to be filled with an overall record for the accounting period, giving starting time, ending time, number of reboots, and number of date changes.

Figure 10 shows the line use file (line_use) created from the same wtmp file used in the previous `acctcon1` example; the standard output of `acctcon1` has been redirected into the file ctmp

```
TOTAL DURATION IS 899 MINUTESS

LINE     MINUTES  PERCENT  # SESS  # ON    # OFF
console  856      95       1       1       1
tty07    69       8        1       1       1
ttya1    40       4        1       1       1
tty10    84       9        1       1       1
TOTALS   1049     --       4       4       4
```

Figure 10 Line use file

prctmp command

The `prctmp` command's function is to put headings on the output created by `acctcon1`. `prctmp` makes a readable report from the output of `acctcon1`.

`prctmp` takes its input from standard input; therefore, to create a `prctmp` report from `acctcon1` information, you can simply pipe the output from `acctcon1` into `prctmp` as follows:

```
acctcon1 < /etc/wtmp | prtcmp
```

`prctmp` responds by generating a report with appropriate headings over the columns of output from `acctcon1`.

acctcon2 command

`acctcon2` creates connect session total accounting records from standard input of the format created by `acctcon1`. In other words, to create connect session total accounting records, simply send the output from `acctcon1` into the input of `acctcon2`.

The total accounting records created by `acctcon2` are sent to standard output. If you want to store these records, you must redirect standard output. The following command line shows how to write total accounting records from the connect session record file (wtmp) into the file ctacct:

```
acctcon1 < /etc/wtmp | acctcon2 > ctacct
```

Process accounting

Process accounting commands provide the means to accumulate execution statistics including; memory usage, CPU time, number of input/output transfers for individual processes. This section describes how to:

- Turn process accounting on
- Turn process accounting off
- Make sure that the process accounting file (pacct) does not become too large
- Display process accounting records
- Generate a command summary report
- Create total accounting records from the process accounting file

Turning process accounting on

Before system accounting can generate process accounting data, process accounting must be turned on. Two commands accomplish this task: `turnacct on` and `accton`. After process accounting has been turned on, the kernel writes a process accounting record for every terminating process. The record is written into the current process accounting file (pacct by default).

The `startup` command, placed in the system initialization shell script /etc/rc, automatically turns process accounting on. Therefore, if you have updated /etc/rc for system accounting (as described in the section "How to install system accounting" in this chapter), process accounting automatically activates, and you should seldom need to use the commands described here.

These commands are described in case you ever need to manually turn process accounting on or off.

turnacct on command

The command used most often to activate accounting is `turnacct on`; only the superuser and the adm login can execute this command. `turnacct on` assumes that the process accounting file is the default file pacct. The action of `turnacct on` can be summarized as follows:

- Check to see if the process accounting file pacct exists.
- If pacct does not exist, create a new pacct file.
- Turn process accounting on by invoking `accton` with the filename argument `pacct`.

To execute this command, simply enter `turnacct on` at the SPP-UX prompt.

accton command

Only the superuser and the adm login can execute `accton`. When invoked with a filename argument, `accton` turns on process accounting and makes the specified filename the current process accounting file. For example,

```
accton /usr/adm/pacct
```

tells the kernel to start writing process accounting records to the file called /usr/adm/pacct. The next example activates process accounting and makes the current process accounting file /usr/adm/XX107:

```
accton /usr/adm/XX107
```

The filename you specify must be an existing file; otherwise, `accton` fails.

`accton` calls the system routine `acct`, which instructs the kernel to start writing process accounting records.

Turning process accounting off

Two commands are used to turn process accounting off: `turnacct off` and `accton` (with no filename argument). These commands tell the kernel to stop writing records to the current process accounting file.

If you have updated the /etc/shutdown shell script as described in "How to install system accounting" on page 160, you will seldom, if ever, use these commands. The reason is that the `shutacct` command, added to /etc/shutdown, automatically turns process accounting off.

turnacct off command

`turnacct off` can be executed only by the superuser and the adm login. `turnacct off` turns process accounting off by invoking the `accton` command without the optional *filename* argument. You execute this command by typing:

```
turnacct off
```

accton command

When `accton` is invoked without the optional filename argument, process accounting is turned off. You would enter this command as:

```
accton
```

`accton` calls the system routing acct, which instructs the kernel to stop writing process accounting records.

Checking the size of pacct

System accounting has built-in mechanisms that insure that the default process accounting file pacct does not become too large. The two commands used for this purpose are

- `ckpacct`
- `turnacct switch`

The commands described here work only on the default process accounting file, pacct.

`ckpacct`

The command `ckpacct` is normally invoked by `cron` every hour to ensure that the current process accounting file pacct hasn't become too large. The format of `ckpacct` is:

`ckpacct` [*blocks*]

If the size of pacct exceeds the *blocks* argument, 1 000 by default if *blocks* is not specified, then `turnacct switch` is executed. `turnacct switch` renames the current pacct file and creates a new pacct file.

If the amount of free space falls below a certain threshold, `ckpacct` automatically turns off process accounting via `turnacct off`. The threshold is determined by the settings of the `acctresume` system tunable parameter (see "SPP-UX system tunable parameters," on page 239 for more information). When free space goes over the specified percentage, process accounting is reactivated.

`turnacct switch`

`turnacct switch` is used to create a new pacct file when the current pacct file is too large. The action of `turnacct switch` can be summarized as follows:

- Process accounting is temporarily turned off.
- The current pacct file is renamed to `pacct` *incr*, where *incr* is a number starting at 1 and incrementing by one for each additional pacct file that is created via `turnacct switch`.
- After the old pacct file is renamed to `pacct` *incr*, a new, current pacct file is created.
- Process accounting is restarted; the kernel starts writing records to the newly created pacct file.

Displaying process accounting records using acctcom

The `acctcom` command allows you to display records from any file containing process accounting records. Normally you would use this command to display records from the pacct files (pacct, pacct1, pacct2 ...).

`acctcom` has the following syntax:

`acctcom` [[*options*] [*file*]]...

If no *file* is specified, `acctcom` uses the current pacct file as input. Input can also be taken from standard input. Some of `acctcom`'s options allow you to select only the records that you want to see; other options control the format of the report.

The information contained in this section is organized as follows:

- Definitions are given for the columnar data produced by `acctcom`.
- Command options that control the format of the report are discussed.
- Options that allow you to select particular records are described.
- Sample `acctcom` reports are shown to help you understand how to use `acctcom`'s options.

Definitions of information produced by acctcom

`acctcom` generates a columnar report with descriptive headings on each column. Each line of the report represents the execution statistics that a particular process accumulated during its lifetime. The standard columns in the report, that is, the columns that are displayed when none of `acctcom`'s options are specified, are shown in Table 12.

Table 12 `acctcom` report with no options

Column	Description
`COMMAND NAME`	The name of the command or program that spawned the process is shown here. Whenever you enter a command, you are spawning a process. For example, if you enter the command `ll /usr/lib/acct` you are creating a process with the command name `ll`. If a command requiring superuser privileges is executed, a # appears before the command name.
`USER`	The login name of the user who created the process
`TTYNAME`	The name of the terminal from which the process was executed. If the process was not executed from a known terminal (for example, if it was executed via `cron`), then a question mark(?) appears in this column.
`START TIME`	The time the process began executing, in *hh:mm:ss* format
`END TIME`	The time the process finished executing, in *hh:mm:ss* format
`REAL (SECS)`	The number of seconds that elapsed from `START TIME` to `END TIME`
`CPU (SECS)`	The CPU time, in seconds, the process used during its execution
`MEAN SIZE(K)`	An estimate, in kilobytes, of the amount of memory that the process used during execution. This estimate is determined from the current process's memory usage at each system clock interrupt. It is, therefore, subject to statistical sampling errors. Only the memory resident pages of a process are counted; no pages in the swap space are counted. Shared code and data is divided among the processes using it. The size is divided by the number of processes sharing the code or data.

Table 13 describes columns that are not displayed on the standard report, but which can be displayed by using `acctcom` command options.

Table 13 `acctcom` report optional columns

Column	Description
`F`	For a process created by `fork` that does not do an `exec`, this column has a value of 1; commands that require superuser privileges show a 2; superuser commands that do a `fork` without an `exec` show a 3; otherwise, this column contains a 0.
STAT	The system exit status. (This is not the status returned by exit to a parent process during wait.) When a process terminates normally, this field shows a 0. If a command terminates abnormally, then a value other than zero is shown. For example, if you interrupt a command with the **DEL** key, this column contains a 2.
HOG FACTOR	The hog factor. This is computed as the CPU time divided by real time; it provides a relative measure of the available CPU time used by the process during its execution. For example, a hog factor of less than 0.50 indicates that the process spent less than half of its time using the CPUs. A hog factor of 0.75 indicates that a process spent 75% of its time using the CPUs.
KCORE MIN	This calculation provides a combined measurement of the amount of memory used (in kilobytes) and the length of time it was used (in minutes). It is computed as follows: `KCORE MIN = CPU (SECS) * MEAN SIZE(K) / 60`
CPU SYS	The portion of total CPU time that was spent executing operating system code, such as system calls (for example, writing to disk)
USER (SECS)	The remaining portion of CPU time. User CPU time is the amount of time actually spent executing a process's code (rather than system code)

`CPU FACTOR`	Whenever you execute a command, the CPU spends part of its time actually executing the command's code (user CPU time) and spends the rest of its time performing system functions, such as writing to the disk or terminal (system CPU time). That is, total CPU time is comprised of both `CPU SYS` and `USER CPU` time:
	`CPU (SECS) = CPU SYS + USER (SECS)`
	The CPU factor shows the ratio of user CPU time to total CPU time:
	`CPU FACTOR = USER (SECS) / (CPU SYS + USER (SECS))`
	For example, if a command has a CPU factor of 0.35, that means that the CPU spent 35% of its time executing user code and 65% performing system functions.

Table 13 `acctcom` report optional columns —(continued)

Column	Description
`CHARS TRNSFD`	The number of characters (bytes) read and/or written by the command
`BLOCKS R/W`	The number of file system blocks read and/or written as a result of executing this command. This number is not directly related to `CHARS TRNSFD` and may vary each time the command is executed, because `BLOCKS R/W` is affected by directory searches made before opening files, other processes accessing the same files, and general file system activity.

Report format options

When no report format options are specified, `acctcom` produces a report containing only the default information. Optional information can be displayed only by using the report format options. Definitions of the report format options follow:

`-a`

Cause average statistics to be displayed at the end of the report. The following information is shown:

- Total number of commands processed (cmds=xxx)
- Average real time per process (Real=x.xx)
- Average CPU time per process (CPU=x.xx)
- Average USER CPU time per process (USER=x.xx)
- Average SYS CPU time per process (SYS=x.xx)
- Average characters transferred (CHARS=x.xx)
- Average blocks transferred (BLK=x.xx)
- Average CPU factor (USR/TOT=x.xx)
- Average HOG factor (HOG=x.xx)

`-b`

Display the process records in reverse order; most recently executed commands are shown first.

`-f`

Print the fork/exec flag (F column) and process exit status (STAT column) on the report.

`-h`

Cause the optional HOG FACTOR column to be displayed, instead of the standard mean memory size column MEAN SIZE(K).

`-i`

Replace the standard MEAN SIZE(K) column in the report with the optional I/O counts, CHARS TRNSFD and BLOCKS R/W.

`-k`

Replace the standard MEAN SIZE(K) column with KCORE MIN.

`-m`

Show the default column MEAN SIZE(K) on the report. This option is used to include MEAN SIZE(K) when it has been bumped off by another option. The following example produces a report showing both KCORE MIN and MEAN SIZE(K):

```
acctcom -km
```

`-r`

Include the optional CPU FACTOR column in the report.

`-t`

Show separate system and user CPU times (CPU SYS and USER (SECS), respectively).

`-v`

Suppress the printing of column headings at the top of the report.

`-q`

This option is the same as the `-a` option, except that individual process accounting records are not displayed; only the averages are displayed.

`-o` *ofile*

Copy the input process accounting records to *ofile.*

Record selection options

The options described here allow you to select the records that are included in the report produced by `acctcom`. For each option, descriptions and examples are provided:

`-l` *line*

Display only the processes that were executed from the user terminal /dev/*line*. For example:

```
acctcom -l console
```

would display records only for the processes that were created from the terminal console.

-u *user*

Show only the processes belonging to *user*. *user* can be any of the following:

- A user ID (for example, `acctcom -u 355`)
- A user name (`acctcom -u julie`)
- A cross-hatch (#) (`acctcom -u#`)
- A question mark (?) (`acctcom -u?`)

If # is specified as the user name, only the commands that require superuser privileges are displayed by `acctcom`. If ? is specified as the user name, only the processes with unknown process IDs are displayed. For example, the following two commands are equivalent:

```
acctcom -u 0
acctcom -u root
```

-g *group*

Show only the processes belonging to *group*. *group* may be specified as either a group name or group ID. For example, suppose the group pseudo with group ID 300 is defined in /etc/group; then the following two commands are equivalent:

```
acctcom -g 300
acctcom -g pseudo
```

-s *time*

Select processes existing at or after *time*. Time is given in 24-hour format, *hr*[:*min*[:*sec*]]. The following example would display all the processes that existed at or after 3:30 p.m.:

```
acctcom -s 15:30
```

-e *time*

Select processes that existed at or before *time*. *time* is supplied in 24-hour format, *hr*[:*min*[:*sec*]]. The next example would display all the processes that existed between midnight and 12:15 am:

```
acctcom -e 0:15
```

-S *time*

Select processes starting at or after *time*. *time* is in 24-hour format, *hr[:min[:sec]]*. The following example would display all the processes that started at 1:30:42 p.m. or after:

```
acctcom -S 13:30:42
```

-E *time*

Display only the processes that terminated at or before *time*. *time* is in 24-hour format, *hr[:min[:sec]]*. Both the `-S` and `-E`

options with the same *time* argument cause `acctcom` to display only the processes that existed at the specified time. For example, to see all the processes that existed at exactly 30 minutes past noon, you would enter:

```
acctcom -S 12:30 -E 12:30
```

`-n` *pattern*

Show only the commands matching *pattern*. *pattern* can be a regular expression as described in ed(1), except that + means one or more occurrences. For example, to display all processes that were created by executing the `ls` command, you would enter:

```
acctcom -n ls
```

To display all the commands that start with `acct`, enter:

```
acctcom -n acct
```

To see all the commands that contain the letter m in their spelling you can use the wild card character *. Enter:

```
acctcom -n .*m.*
```

`-H` *factor*

Display only those processes whose hog factor exceeds *factor*. For example,

```
acctcom -H 0.85
```

would display all the processes that spent over 85% of their execution time in the CPUs. You can use this option to find processes that are hogging the CPUs.

`-O` *time*

Show only those processes whose system CPU time exceeds *time*. *time* is specified in seconds. The following example would be used to determine which processes took more than 8.25 seconds of operating system CPU time to execute:

```
acctcom -O 8.25
```

This option could be used to determine which processes are making heavy use of the operating system calls.

`-C` *sec*

Show only the processes whose total CPU time (SYS + USER) exceeds *sec* seconds. The next example would display all the processes that used over 5.28 seconds of CPU time to execute:

```
acctcom -C 5.28
```

`-I` *chars*

Display only the processes transferring more characters than the limit given by *chars*. For example,

```
acctcom -I 10240
```

displays all the processes that transferred over ten kilobytes of characters (10240 = 10 * 1024 bytes).

Command summary report (acctcms)

The `acctcms` command takes process accounting records as input; but instead of reporting on the individual processes, `acctcms` generates a report on the commands that generated the process accounting records. The action of `acctcms` can be summarized as follows:

1. `acctcms` looks through the input process accounting records and accumulates execution statistics for each unique command name. This information is stored in internal summary format, one record per command name.
2. Depending on the `acctcms` options used, the command summary records created in Step 1 are sorted.
3. The command summary records are written to standard output in the internal summary format mentioned in Step 1. This format is not readable.

The syntax of the `acctcms` command is:

`acctcms` [*options*] *files*

files is a list of the input process accounting files for which the command summary report is to be generated. *options* are discussed in the following sections.

Producing a readable report (-a option)

By default, the output of `acctcms` is in internal summary record format; it is not human-readable. To get an ASCII readable report, use the `-a` option.

The `-a` option causes `acctcms` to produce a report with descriptive column headings. Total and average (mean) execution statistics for each command are displayed, one line per command, along with total and average statistics over all commands in the report. Descriptions of the columnar data produced by `acctcms` are shown in Table 14.

Table 14 `acctcms -a` report

Column	Description
COMMAND NAME	The name of the command for which execution statistics are summarized. All shell procedures are lumped together under the name `sh`, because only object modules are reported by the process accounting system
NUMBER CMDS	The total number of times the command was invoked
TOTAL KCOREMIN	The total amount of kcore minutes accumulated for the command. (See "Definitions of information produced by acctcom" on page 184 for a more complete description of kcore minutes.)
TOTAL CPU-MIN	The total CPU time that the command has accumulated
TOTAL REAL-MIN	Total accumulated real time in minutes that the command has accumulated
MEAN SIZE-K	The average amount of memory (in kilobytes) used by the command
MEAN CPU-MIN	The average CPU time consumed per command invocation is shown here; the following equation shows how it is computed: MEAN CPU-MIN = TOTAL CPU-MIN / NUMBER CMDS
HOG FACTOR	The average hog factor over all invocations of the command. It is computed as: HOG FACTOR = TOTAL CPU-MIN / TOTAL REAL-MIN
CHARS TRNSFD	The total number of characters transferred by the command. This number may be negative.
BLOCKS READ	A total count of the physical blocks read and written by the command

When only the `-a` option is specified, the report is sorted in descending order on the `TOTAL KCOREMIN` column; commands using more `TOTAL KCOREMIN` are shown before those using fewer `TOTAL KCOREMIN`. This report gives a relative measure of the amount of memory used over time by the various commands; commands toward the start of the report are making more use of memory resources than are commands toward the end of the report.

Other options

In addition to the `-a` option, several other options can be used to control the format of the report generated by `acctcms`. Some options specify which field to sort the report on; other options control the printing of prime/nonprime time usage. The options and a description of their use follow:

`-c`

Sort the commands in descending order on `TOTAL CPU-MIN`, rather than the default `TOTAL KCOREMIN`. This report can be used to determine which commands are using most of the computer's CPU time.

`-n`

Cause the report to be sorted in descending order on the column named `NUMBER CMDS`. Commands toward the start of this report are the ones used most frequently; commands toward the end are used least often.

`-j`

Combine all commands invoked only once on one line of the report. This line is denoted by having "***other" in the `COMMAND NAME` column. This option is useful for shortening a report that has many one-invocation commands.

`-o`

Used only with the `-a` option, `-o` causes the report to be generated only for commands that were executed during nonprime time (as specified in the holidays file). You can use this option to get a nonprime time command summary report.

`-p`

Also used only with the `-a` option, `-p` elicits a report generated only for commands that were executed during prime time (as specified in holidays). This option is used to get a prime time command summary report.

`-apo`

When the options `-o` and `-p` are used together with `-a`, a combination prime and nonprime time report is produced. The output of this report is the same as that produced by `-a` alone, except that the `NUMBER CMDS`, `TOTAL CPU-MIN`, and `TOTAL REAL-MIN` columns are divided into two columns—one for prime time totals, the other for nonprime time. (Prime time columns have a (P) header, while nonprime time columns are headed by (NP).)

`-s` *files*

Specifies that any named input *files* following the `-s` on the command line are already in internal summary format. This option is useful for merging previous `acctcms` reports with current reports. The following example uses `-s` to create a command summary report from previous process accounting files (pacct?) and the current process accounting file (pacct). The final ASCII report is stored in the file ascii_summary.

```
acctcms pacct? > old_summary
```

```
acctcms pacct > new_summary
acctcms -as old_summary new_summary > ascii_summary
```

Sample report

The ASCII reports produced by `acctcms` contain more than 80 characters per line. When these reports are displayed at an 80-column terminal, the lines wrap around on the screen. In addition, if the report is printed on an 80-column printer, some of the rightmost columns is lost. Therefore, be sure to do one of the following:

- Use a printer with compressed print capabilities, so that all of the report fits on standard computer paper.
- Use a printer with enough columns to display all the information for example, a 132-column printer.

Figure 11 shows a summary report for the current process accounting file (no file is specified, so the current pacct file is assumed). By giving the `-j` option, all the commands that were executed only once are grouped under the command name ***other. Note also that total execution statistics for all commands are grouped under the command name TOTALS.

```
acctcms -aj
TOTAL COMMAND SUMMARY
COMMAND    NUMBER TOTAL     TOTAL    TOTAL     MEAN    MEAN     HOG    CHARS     BLOCKS
NAME       CMDS   KCOREMIN CPU-MIN REAL-MIN SIZE-K CPU-MIN FACTOR TRNSFD   READ
TOTALS     61   17.63     0.38    164.49    46.25   0.01    0.00   104553   1027
acctcms    17   12.13     0.16    0.35      76.72   0.01    0.45   49192    306
sh         8    2.43      0.09    152.86    26.79   0.01    0.00   9043     163
more       3    0.73      0.02    10.50     31.00   0.01    0.00   21618    83
ll         6    0.61      0.04    0.11      16.50   0.01    0.33   5715     95
acctcom    4    0.58      0.02    0.07      28.50   0.01    0.30   15319    42
***other   9    0.54      0.02    0.14      25.26   0.00    0.16   459      161
cat        4    0.19      0.01    0.35      22.97   0.00    0.02   3112     52
rm         2    0.11      0.00    0.02      22.22   0.00    0.29   0        29
chmod      2    0.10      0.00    0.01      2.00    0.00    0.35   0        15
accton     2    0.08      0.00    0.02      19.00   0.00    0.29   0        22
sed        2    0.08      0.01    0.04      14.50   0.00    0.13   73       38
echo       2    0.05      0.00    0.02      20.00   0.00    0.16   22       21
```

Figure 11 `acctcms -aj` summary report

Creating total accounting records

Two commands — `acctprc1` and `acctprc2` — are used to create total accounting records from the process accounting files. The output from `acctprc1` is supplied as input to `acctprc2`, which produces the total accounting records. These commands are normally invoked by `runacct` to produce daily accounting information.

acctprc1 command

This command reads process accounting records from standard input, adds login names corresponding to the user ID of each record, and then writes for each process an ASCII line showing:

- The ID of the user that created the process
- The user's login name
- Prime CPU time in ticks (a tick is one fiftieth of a second)
- Nonprime CPU time, also in ticks
- Mean memory size (in pages, 4 Kbytes per page)

The format of `acctprc1` is:

`acctprc1` [*ctmp*]

ctmp contains a list of login sessions of the form created by `acctcon1`, sorted by user ID and login name.

The number of sessions should be 1000 or less. If there are more than 1000 sessions, the accounting system hangs (suspends indefinitely) and must be killed manually via the `kill` command and restarted.

To use `acctprc1`, input must be redirected from a process accounting file. The following example creates a file, ascii_ptacct, containing ASCII process accounting information that can be used to create process total accounting records. This file is created from the current process accounting file pacct.

```
acctprc1 < pacct >ascii_ptacct
```

Normally, `acctprc1` gets login names from the password file /etc/passwd, which is sufficient on systems where each user has a unique user ID. However, on systems where different users share the same user ID, the ctmp file should be specified; it helps `acctprc1` distinguish different login names that share the same user ID.

acctprc2 command

This command reads from standard input records of the form created by `acctprc1`; it then summarizes the records by user ID

and name, and writes the sorted summaries to standard output as total accounting records. The following example creates total accounting records for all processes in the current process accounting file pacct; the total accounting records are stored in the file ptacct.

```
acctprc1 <pacct | acctprc2 >ptacct
```

Charging fees to users (chargefee)

System accounting provides the capability to charge fees to specific users; the `chargefee` command is used to accomplish this task. `chargefee` allows you to charge generic units to a specific login name. The syntax of this command is:

`chargefee` *login_name number*

where:

login_name

Is the login name of the user to whom *number* units are to be charged; *number* is the number of units to be charged to a particular user.

number

Can be any whole number in the range -32 768 to 32 767; when charging fees, keep in mind that the sum of each user's fees must also be within this range.

`chargefee` accumulates fee charge records in the file /usr/adm/fee. These records are then merged with other accounting records via `runacct`.

The following example charges 25 units to the user whose login name is julie:

```
chargefee julie 25
```

Suppose you inadvertently charged 247 units to the user named zimblits, and you want to return his charges to their original value. You would enter the following:

```
chargefee zimblits -247
```

Summarizing and reporting accounting information

This final group of commands summarizes and reports accounting information. Certain commands display and merge total accounting files, while others generate the daily and monthly reports used to analyze system performance and bill users for resource usage. The following commands are discussed in this section:

- `prtacct` — Displays total accounting records
- `acctmerg` — Merges total accounting files
- `runacct` — Generates daily summary files and reports
- `prdaily` — Displays the daily summary files and reports created by `runacct`
- `monacct` — Creates monthly summary files and reports

Displaying total accounting records (prtacct)

The `prtacct` command allows you to display the contents of a process accounting file. Its format is:

`prtacct` *file "heading"*

where:

file

Is the name of the total accounting file to display.

"heading"

Is a comment to be included in the standard report header produced by `prtacct`.

The format of the `prtacct` report is described next and is followed by an example.

`prtacct` produces a columnar report with one line per total accounting record. Descriptive column headings are included in the report. Descriptions of the columnar data produced by `prtacct` are shown in Table 15.

Table 15 `prtacct` report

Column	Description
UID	The user ID of the owner of the total accounting record (the ID of the user for whom the total accounting record was created).
LOGIN NAME	The login name of the owner of the total accounting record.

Table 15 `prtacct` report —(continued)

Column	Description
CPU (MINS)	The total amount of CPU time (in minutes) that the user has consumed. This column is divided into prime and nonprime columns (`PRIME` and `NPRIME`, respectively). Information in these columns is created through process accounting commands.
KCORE-MINS	A cumulative measure of memory and CPU time that a user consumed. Information in this column is also divided into PRIME and NPRIME columns. This information is created through process accounting commands.
CONNECT (MINS)	This identifies the real time used (in minutes). In essence, what this column identifies is the amount of time that the user was logged in to the system. This column is also subdivided into `PRIME` and `NPRIME` columns. The connect session accounting commands are the source of this information.
DISK BLOCKS	The total number of disk blocks allocated to the user. This information is created via disk space accounting commands.
# OF PROCS	The total number of processes spawned by the user. This information is created via the process accounting commands.
# OF SESS	The number of times the user logged in. Connect session accounting commands create this data.
# DISK SAMPLES	The number of times the disk accounting was run to obtain the average number of disk blocks listed in the `DISK BLOCKS` column.
`FEE`	The number of fee units charged via `chargefee`.

chargefee example

The following example displays disk total accounting records. First, the total accounting records are created via disk space accounting commands; then, they are displayed using `prtacct`. When examining this report, note the following:

- There are many similarities between this and the sample report produced by `diskusg` (refer to the section "Disk space usage accounting commands" on page 164).
- Only the columns relating to disk space usage have nonzero values, because the total accounting records were created only from disk space usage accounting commands.

Figure 12 shows a report produced by `prtacct`.

```
for file_system in `cat /etc/checklist`
   do
   diskusg $file_system >dtmp.`basename $file_system`
   done
diskusg -s dtmp.* | sort +0n +1 | acctdisk >diskacct
prtacct diskacct "DISK TOTAL ACCOUNTING RECORDS"
Mar 26 17:01 1985 DISK TOTAL ACCOUNTING RECORDS 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   0      0      0     0      0     0      11598    0     0     10     0
0    root    0      0      0     0      0     0      10616    0     0     1      0
1    bin     0      0      0     0      0     0      778      0     0     1      0
4    adm     0      0      0     0      0     0      96       0     0     1      0
350  fred    0      0      0     0      0     0      14       0     0     1      0
351  bill    0      0      0     0      0     0      32       0     0     1      0
352  mike    0      0      0     0      0     0      20       0     0     1      0
353  sarah   0      0      0     0      0     0      16       0     0     1      0
354  molly   0      0      0     0      0     0      22       0     0     1      0
355  julie   0      0      0     0      0     0      2        0     0     1      0
501  guest   0      0      0     0      0     0      2        0     0     1      0
```

Figure 12 Sample `prtacct` report

Merging total accounting files (acctmerg)

Normally executed by `runacct`, the `acctmerg` command merges separate total accounting files into a single total accounting file. All the total accounting records for a particular user name and ID are merged together to form one total accounting record for the given user name and ID. This command is useful for merging disk, connect session, and process total accounting files together to form a single, comprehensive total accounting file.

`acctmerg` reads standard input and up to nine additional files, all in total accounting record format. Its syntax is:

`acctmerg` [*options*] [*file...*]

options

Control the report format and the manner in which records are merged.

file

Is one of up to nine files (in addition to standard input) that are to be merged into a single total accounting file, written to standard output.

The following options may be used with `acctmerg` to control the report format and the manner in which the total accounting records are merged:

`-a`

`acctmerg` normally produces output as total accounting records. The `-a` option causes `acctmerg` to produce output in ASCII. Note that the output generated by using this option is the same as the report produced by `prtacct`, except that no report headings or totals are displayed — only the columnar data is shown.

`-i`

In the default case, `acctmerg` assumes that its input files contain total accounting records. If `-i` is specified, then `acctmerg` expects input files to be in the ASCII format created by the `-a` option.

`-p`

This option echoes input records — no merging or processing is done. The output is displayed in the format produced by the `-a` option.

`-t`

This option produces a single total accounting record that summarizes all input records. To see the ASCII version of this record, you must use the `-t` and `-a` options together:

```
acctmerg -t -a < tot_acct_recs
```

`-t` and `-a` can be specified in any order, but they must be specified separately as shown.

`-u`

Normally, `acctmerg` merges records that have the same user ID and user name. Using `-u` causes `acctmerg` to merge records on the basis of same user ID only — that is, the user name is disregarded as a key on which to merge records.

`-v`

This option causes `acctmerg` to produce output in verbose ASCII format. The same report is produced as the `-a` option, except that floating point numbers are displayed in more precise notation:

```
<mantissa>e<exponent>
```

Use the -a, -v, and -i options to edit total accounting records. For example, if you created a total accounting file (ptacct) containing process total accounting records, and you want to make some adjustments to these records, use the following sequence to repair this file:

```
acctmerg -v -a < ptacct > ptacct.ascii
```

Next, edit the ptacct.ascii file.

```
acctmerg -i < ptacct.ascii > ptacct
```

The following example creates disk, process, and connect session total accounting records, merges them together, and stores the merged file in the file tacct.

```
for fs in 'cat /etc/checklist'
   do
   diskusg $fs >dtmp.~basename $fs~
   done
diskusg -s dtmp.* | sort +0n +1 | acctdisk >dtacct
acctcon >/etc/wtmp | acctcon2 >ctacct
ptacct
for p_file in pacct*
   do
   acctpr1 <$p_file | acctprc2 >>ptacct
   done
acctmerg dtacct ctacct <ptacct >tacct
```

Creating daily accounting information (runacct)

runacct is the main daily accounting shell procedure. Start runacct via cron during nonprime hours, when users are logged off. This is because it does not correctly log time for users that log on before running runacct.

runacct processes disk, connect session, process, and fee accounting files. It prepares cumulative summary files for use by prdaily and for billing purposes. This section discusses the following aspects of runacct:

- Files processed by runacct
- The states that runacct progresses through while executing
- Recovery from runacct failure
- Restarting runacct
- Reports produced by runacct

Files processed by runacct

The following files, processed by `runacct`, are of particular interest to the reader. (File names are given relative to the directory /usr/adm/acct.)

- nite/lineuse contains usage statistics for each terminal line on the system. This report is especially useful for detecting bad lines. If the ratio of logoffs to logins on a particular line exceeds 3 to 1, then there is a good possibility that the line is failing.
- nite/daytacct contains total accounting records from the previous day.
- sum/tacct contains accumulated total accounting records for each day's total accounting records (nite/daytacct) and can be used for billing purposes. It is restarted each month or fiscal period by the `monacct` shell script.
- sum/daycms is produced by `acctcms`. It contains the daily command summary. The ASCII version of this file is in nite/daycms.
- sum/cms holds the accumulation of each day's command summaries (sum/daycms). A new sum/cms file is created each month by `monacct`. The ASCII version of this file is in nite/cms.
- sum/loginlog maintains a record of the last time each user logged in.
- sum/rprt*mmdd* is the main daily accounting report created by `runacct`. The name for this report is created automatically by the system with *mm* being the month and *dd* the day of the report. This report can be printed via `prdaily`.

`runacct` takes care not to damage files in the event of errors. A series of protection mechanisms is used to attempt to recognize errors, provide intelligent diagnostics, and terminate processing in such a way that `runacct` can be restarted with minimal intervention. To accomplish these goals, the following actions are performed by `runacct`:

- `runacct`'s progress is recorded by writing descriptive messages to the nite/active file.
- All diagnostics output during the execution of `runacct` are redirected to the file nite/fd2log.
- If the files lock and lock1 exist when `runacct` is invoked, an error message is displayed and execution terminates.
- The lastdate file contains the month and day that `runacct` was last run and is used to prevent more than one execution per day.

- If `runacct` detects an error, a message is written to /dev/console, mail is sent to root and adm, locks are removed, diagnostics files are saved, and execution is terminated.

The states of runacct

In order to allow `runacct` to restart, processing is broken down into separate reentrant states. As `runacct` executes, it records its progress by writing the name of the most recently completed state into the file called /usr/adm/statefile. After processing for a state is complete, `runacct` examines statefile to determine which state to enter next. When `runacct` reaches the final state (`CLEANUP`), the lock and lock1 files are removed, and execution terminates. `runacct`'s states are described in Table 16.

Table 16 `runacct` states

State	Description
SETUP	The command `turnacct switch` is executed. The process accounting files, pacct?, are moved to Spacct?.*mmdd*. The /etc/wtmp file is moved to nite/wtmp.*mmdd* with the current time added on the end.
WTMPFIX	nite/wtmp.*mmdd* is checked for correctness by `wtmpfix`. Some date changes causes `acctcon1` to fail, so `wtmpfix` attempts to adjust the time stamps in the nite/wtmp.*mmdd* file if a date change record appears.
CONNECT1	Connect session records are written to ctmp. The lineuse file is created, and the reboots file, showing all of the boot records found in nite/wtmp.*mmdd*, is created.
CONNECT2	ctmp is converted to connect session total accounting records in the file ctacct.*mmdd*.
PROCESS	The `acctprc1` and `acctprc2` programs are used to convert the process accounting files, Spacct?.*mmdd*, to the total accounting records in ptacct?.*mmdd*. The Spacct and ptacct files are correlated by number so that if `runacct` fails, the unnecessary reprocessing of Spacct files does not occur. One precaution should be noted: when restarting `runacct` in this state, remove the last ptacct file; if you do not, `runacct` does not finish.
MERGE	Merge the process and connect session total accounting records to form nite/daytacct.
FEES	Merge in any ASCII tacct records from the file fee into nite/daytacct.
DISK	On the day after the `dodisk` shell script runs, merge nite/disktacct with nite/daytacct.

Table 16 `runacct` states —(continued)

State	Description
MERGETACCT	Merge nite/daytacct with sum/tacct, the cumulative total accounting file. Each day, nite/daytacct is saved in sum/tacct*mmdd*, 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 summary file sum/cms. Produce ASCII and internal format command summary files.
`USEREXIT`	Any installation-dependent (local) accounting programs can be run in this state. For example, you might want to execute commands that generate daily billing data for individual users (the shell script acct_bill in "Sample accounting shell scripts" on page 211 could be used for this purpose). To have local accounting programs executed by `runacct`, enter the commands for the `USEREXIT` state of `runacct` in the local code.
`CLEANUP`	Clean up the temporary files, run `prdaily` and save its output in the file sum/rprt*mmdd*, remove the locks, then exit.

Recovering from failure

It is possible that `runacct` might fail and terminate abnormally. The primary reasons for `runacct` failure are:

- A system crash
- Not enough disk space remaining in /usr
- A corrupted wtmp file

If the nite/active*mmdd* file exists, check it first for error messages. If the nite/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 be restarted.

`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 before restarting `runacct`.

`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 have probably already run. Check the status of files, then run setups manually.

`ERROR: /usr/adm/acct/nite/wtmp.`*mmdd* `already exists, run setup manually`

You must perform the SETUP step manually, because the daily wtmp file already exists.

```
ERROR: wtmpfix errors see /usr/adm/acct/nite/wtmperror
```

wtmpfix detected a corrupted wtmp file. Refer to the section "Fixing corrupted files" on page 209 for details on fixing wtmp errors.

```
ERROR: connect acctg failed: check /usr/adm/acct/nite/log
```

acctcon1 encountered a bad wtmp file. Again, refer to "Fixing corrupted files" on page 209 information on how to fix the 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 is normally run via cron only once per day. However, if an error occurs while runacct is executing (as described above), it may be necessary to restart runacct. runacct has the following syntax:

runacct [*state*]

runacct assumes that it is being invoked for the first time on the current day. The entry point for processing is based on the contents of statefile. To override statefile, include the desired entry state on the command line.

For example, to start runacct, you would enter:

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

To restart runacct at state WTMPFIX:

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

All the above examples were run in the background (&) and use the nohup command so the process continues running even though you may log out.

Daily runacct reports

runacct generates five basic reports upon each invocation. Brief descriptions of each report follow.

Daily Line Usage Report

Summarizes connect session accounting since the last invocation of runacct. It provides a log of system reboots, power failure

recoveries, and any other records dumped into /etc/wtmp via `acctwtmp`. In addition, it provides a breakdown of line utilization.

Daily Resource Usage Report

Gives a summary of resource usage per individual user: it basically merges all the total accounting records for individual users and displays the records, one per user.

Daily Command Summary

Summarizes resource usage data for individual commands since the last invocation of `runacct`. The data included in this report is useful in determining the most heavily used commands; you can use these commands' characteristics of resource utilization when tuning your system.

This report is sorted by TOTAL KCOREMIN, an arbitrary but useful yardstick for calculating "drain" on a system.

Monthly Total Command Summary

This report is the same as the Daily Command Summary, except that the Daily Command Summary contains command summary information accumulated only since the last invocation of `runacct`, while the Monthly Total Command Summary summarizes commands from 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`.

Last Login

Gives the date each user last logged in to the system. This could be a good source for finding likely candidates for the archives, or getting rid of unused login directories.

Displaying runacct reports (prdaily)

As `runacct` finishes executing, it deposits a report of the current day's accounting in the file /usr/adm/acct/sum/rpt*mmdd*, where *mmdd* is the month and day the report was generated. The `prdaily` command is used to display the contents of any daily report file created by `runacct`. Its syntax is:

`prdaily [-l] [-c]` [*mmdd*]

where:

-l

Prints a report of exceptional usage by login name for the specified date. This option is used to determine which users are consuming excessive amounts of system resources. The limits for exceptional usage are kept in the file

/usr/lib/acct/ptelus.awk and can be edited to reflect your installation's requirements.

`-c`

Prints a report of exceptional resource usage by command. This option is used to determine which commands are using excessive amounts of system resources. The limits for exceptional usage are maintained in the file /usr/lib/acct/ptecms.awk and can be edited to reflect your system's needs. This option is valid only for the current day's accounting.

mmdd

Optional report date. If no date is specified, `prdaily` produces a report of the current day's accounting information. Previous days' accounting reports can be displayed by using the *mmdd* option and specifying the exact report date desired.

The reports produced by `runacct` were described briefly in the previous subsection. Now the reports are discussed in more detail.

Daily line usage report

In the first part of this report, the FROM/TO banner should alert you to which period is being reported. The times are the date-time that the last report was generated by `runacct`, and the date-time that the current report was generated. It is followed by a log of system reboots, shutdowns, power failure recoveries, and any other records dumped into wtmp by the `acctwtmp` command.

The second part of the report is a breakdown of line utilization. The `TOTAL DURATION` column shows how long the system was in a multiuser state. The columns of the report are defined as shown in Table 17.

Table 17 Daily line usage report

Column	Description
`LINE`	The terminal line or access port being reported on
`MINUTES`	The total number of minutes that the line was in use during the accounting period
`PERCENT`	The percentage of `TOTAL DURATION` that the line was in use: `PERCENT = (MINUTES / TOTAL DURATION)100`
`# SESS`	The number of times that this port was accessed for a login session

Table 17 Daily line usage report —(continued)

Column	Description
# ON	Historically, this column displayed the number of times that the port was used to log a user on; but since login can no longer be executed explicitly to log in a new user, this column should be identical to # SESS
# OFF	This column reflects not only the number of times a user logged off, but also any interrupts that occurred on the line. Interrupts occur on a port when getty is first invoked. getty is invoked when the system is brought to run-level 2. This column comes into play when # OFF exceeds # ON by a large factor. This usually indicates that the multiplexer, modem, or cable is going bad, or that there is a bad connection somewhere. The most common cause of this is an unconnected cable dangling from the multiplexer.

During real time, wtmp should be monitored, as this is the file from which connect session accounting is taken. If it grows rapidly, execute acctcon1 to determine which line is the noisiest. If the interrupting is occurring at a high rate, general system performance is affected.

Daily resource usage report

This report gives a breakdown of system resource usage by user. The format of this report is the same as that produced by the prtacct command. (For definitions of the data and format of this report, refer to the discussion of prtacct in "Displaying total accounting records (prtacct)" on page 196.)

Daily and monthly command summary

These two reports are the same, except that the Daily Command Summary reports information only for commands executed since the last invocation of runacct; the Monthly Command Summary contains information on commands executed since the last invocation of monacct.

The output of this report is identical to that produced by acctcms. For definitions of the data found in this report, refer to the discussion of acctcms in "Process accounting" on page 180.

Last login report

This report shows the last date and time that each user logged in. The longer it has been since a particular user logged in, the more likely it is that the user's files could be archived, or maybe even that the user could be removed from the system.

Creating monthly accounting reports (monacct)

`monacct` creates monthly summary files and reports; the resulting output is stored in the directory /usr/adm/acct/fiscal. After creating its monthly reports, it removes the old daily accounting files from the directory /usr/adm/acct/sum and replaces them with new summary accounting files.

`monacct` should be invoked once each month or accounting period. Its syntax is:

`monacct` *period*

period indicates the month or period (01=January, 12=December). If *period* is not specified, `monacct` assumes that it is being invoked for the current month; this default is useful if `monacct` is executed via `cron` on the first day of each month.

Descriptions of the files created in the acct/fiscal directory follow:

cmsperiod

Is the total command summary file for the accounting period denoted by *period*. The file is stored in internal summary format. Therefore, to display this file, you must use the `acctcms` command. The following example shows how to display this file for the month of June:

```
acctcms -a -s /usr/adm/acct/nite/fiscal/cms06
```

fiscrpt*period*

Is a report similar to that produced by `prdaily`. The report shows line and resource usage for the month represented by *period*. The following would display the fiscal accounting file for the month of November:

```
cat /usr/adm/acct/nite/fiscal/fiscrpt11
```

tacct*period*

Is the total accounting file for the month represented by *period*. To display this file, you must use the `prtacct` command. The following would display the total accounting summary file for the month of January:

```
prtacct /usr/adm/acct/nite/fiscal/tacct01 "JANUARY TOTAL ACCOUNTING"
```

Updating the holidays file

The file /usr/lib/acct/holidays contains the information that system accounting needs to distinguish between prime and nonprime time. It contains the following information:

Comment lines

Comment lines are entered by placing an asterisk (*) as the first character in the line; they may appear anywhere in the file.

Year designation line

This line should be the first noncomment line in the file and must appear only once. The line consists of three four-digit numbers, leading blanks and tabs are ignored. The first number designates the year; the second denotes the time, in 24-hour format, that prime time starts; the third gives the time that prime time ends and nonprime time starts.

For example, to specify the year as 1985, prime time at 9:00 a.m., and nonprime time at 4:30 p.m., the following entry would be appropriate:

```
1985 0900 1630
```

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

Company holiday lines

These entries follow the year designation line. Company holidays are days when few people should be using the computer. Therefore, system accounting assumes that nonprime time is in effect during the entire 24 hours of a specified holiday.

Company holiday lines have the following format:

day_of_year month day description_of_holiday

The *day_of_year* field is a number in the range 1 through 366, corresponding to the day of the year for the particular holiday, leading blanks and tabs are ignored. The remaining fields are commentary and are not used by other programs.

Fixing corrupted files

System accounting files may become corrupted or lost. Some of these files can simply be ignored or restored from the files saved through backup procedures. However, certain files must be fixed in order to maintain the integrity of system accounting. Two of the files that must be fixed are /etc/wtmp and /usr/adm/acct/sum/tacct.

Fixing wtmp errors

The wtmp files seem to cause the most problems in the daily operation of system accounting. When the date is changed and SPP-UX is switched into multiuser mode, a set of date change

records is written into /etc/wtmp. The `wtmpfix` command is designed to adjust the time stamps in the wtmp records when a date change is encountered. However, some combinations of date changes and reboots won't be caught by `wtmpfix` and cause `acctcon1` to fail. The following example shows how to patch a damaged wtmp file:

```
cd /usr/adm/acct/nite

fwtmp <wtmp.mmdd >wtmp.temp
```

Using an editor, delete corrupted records or delete all records from the beginning up to the date change.

```
fwtmp -ic <wtmp.temp >wtmp.mmdd

rm wtmp.temp
```

If the wtmp file is beyond repair, create a null wtmp file. This prevents any charging of connect time. `acctprc1` is not be able to determine which login owned a particular process, but it is charged to the login that is first in the password file for that user ID.

Fixing tacct errors

If your installation is using system accounting to charge users for system resource usage, the integrity of sum/tacct is quite important. If sum/tacct ever becomes corrupted, then check the contents of sum/tacctprev with the command `prtacct`. If it looks correct, then the latest sum/tacct.*mmdd* should be patched up, and sum/tacct should then be recreated. A simple patch procedure would be:

```
cd /usr/adm/acct/sum

acctmerg -a -v <tacct.mmdd >tacct.temp
```

Using an editor, remove the bad records and write duplicate UID records to a separate file.

```
acctmerg -i <tacct.temp >tacct.mmdd

acctmerg tacctprev <tacct.mmdd >tacct

rm tacct.temp
```

Remember that `monacct` removes all the tacct.*mmdd* files; therefore, sum/tacct can be recreated by merging these files together.

Sample accounting shell scripts

Accounting shell scripts automate the accounting functions. This section contains examples of scripts commonly used to perform common accounting functions.

grpdusg shell script

`grpdusg` shell script displays disk space usage totals for the users who are members of a specified group. The syntax of this command is:

`grpdusg` ***group_name***

group_name is the name of the group for which disk space accounting information is to be generated.

For example,

`grpdusg pseudo`

generates disk space usage information for all the users in the group pseudo.

The following sample shows source code for the `grpdusg` shell script.

```
# Check for the group-name parameter.
if [ $# -ne 1 ] then
  echo "\nUsage: grpdusg group-name\n"
  exit 1
fi
echo "\nOne moment please...\n"
# Use the find command to find all the files whose owners are members of
# group-name. Pipe the output from find into acctdusg which will accumulate
# disk space usage information for the users in group-name.
# NOTE:
# - accounting data is temporarily stored in _${1}_tmp
# - error messages are stored temporarily in _${1}_err
# - if files exist that have no owners, then the names of
# these files are stored in _no_owners
fn=_${1}_
find / -group $1 - hidden -print 2>${fn}err |acctdusg -u _no_owners >${fn}tmp
# Remove the _no_owners file if its size is not greater than zero.
if [ -s _no_owners ] then
  echo "\nFiles having no owners exist--check _no_owners\n"
```

```
  else rm _no_owners
  echo "\nAll files have owners-- _no_owners not created\n"
fi
# # Use echo and awk to display disk usage totals for this group.
echo "\nDisk space usage information (group is ${1}):\n"
awk 'BEGIN {print "\n_UIDUSER
NAME_____BLOCKS"}
{ sum += $3 ; # add up total disk blocks used
 print $0 # display information for user
} END { print "\nTOTAL DISC SPACE USAGE= ", sum, "blocks" }' ${fn}tmp
# Remove temporary files, then exit.
rm ${fn}*
```

acct_bill shell script

`acct_bill` takes as input a total accounting file and produces as output billing totals for all users found in the input file. The syntax of `acct_bill` is:

`acct_bill` [*mmdd*]

If the optional *mmdd* is not specified, `acct_bill` takes as input the current day's total accounting file (acct/nite/daytacct); if *mmdd* is given, then input is taken from the total accounting file for the date specified by *mmdd* (acct/sum/tacct*mmdd*). Output is written to the file bills*mmdd*, where *mmdd* is the date given with the command, or the current date if *mmdd* was not specified with the command.

Examples

To generate billing information for the current day, enter:

```
acct_bill
```

The billing information is stored in the file acct/sum/bills*mmdd*, where *mmdd* is the current date.

To create billing information for January 23rd, you would enter:

```
acct_bill 0123
```

after which the billing information would be stored in the file named acct/sum/bills0123.

To automatically generate daily billing totals for all users, you should call `acct_bill` without the date argument from the `USEREXIT` state of `runacct`.

Output produced by acct_bill

The output of `acct_bill` contains one line per user and has the following format:

user_ID user_name billing_amount

where:

user_ID and *user_name*

Identify the user who is being billed

billing_amount

Is the total amount that the user is to be charged. To compute multiply accounting coefficients (found in the shell script) by columns of the report generated by `prtacct`. Assuming that billing amounts are in dollars, the coefficients produce the following billing amounts:

- Ten cents for every minute of prime CPU time consumed
- Five cents for every minute of nonprime CPU time consumed
- A half cent for every prime kcore minute used
- Two-tenths of a cent for every nonprime kcore minute
- A half cent for every prime connect time minute
- Two-tenths of a cent for every nonprime connect minute
- Two-and-a-half cents for every block of disk space used
- Two-and-a-half cents for every process spawned by the user
- Ten cents for every connect session
- Each fee unit charged via chargefee counts as one cent

The following example shows source code for the `acct_bill` shell script.

```
_date=`date +%m%d`
_outfile=/usr/adm/acct/sum/bills
_infile=/usr/adm/acct
# Set _infile and _outfile, based on whether or not mmdd was given
if [ $# -eq 0 ] then # Generate billing data for current day.
  _infile=${_infile}/nite/daytacct
  _outfile=${_outfile}${_date}
else # Create billing data for date given (mmdd).
  _infile=${_infile}/sum/tacct${1}
  _outfile=${_outfile}${1}
fi
# Create a file containing the ASCII equivalent of the input total
# accounting file (tacct_ASC.tmp_). The file can then be supplied as input
# to awk, which will generate billing data for each user.
# acctmerg -a -t &tacct_ASC.tmp # output TOTAL amount first
# acctmerg -a &tacct_ASC.tmp # append users' total accounting records
```

```
#
# Using awk, compute billing totals for each user in the total
# accounting file.
#
awk 'BEGIN {
# ***************************************************
# A C C O U N T I N G C O E F F I C I E N T S
# ***************************************************
  cpu_P =0.10 # 0.10 monetary units per minute of prime CPU time
  cpu_NP=0.05 # 0.05 monetary units per non-prime CPU minute used
  kcm_P =0.005 # for prime kcore minutes consumed
  kcm_NP=0.002 # for non-prime kcore minutes used
  con_P =0.005 # prime connect (real) time
  con_NP=0.002 # non-prime connect time used
  blk = 0.025 # number of blocks used
  prc = 0.025 # number of processes spawned
  ses = 0.10 # number of connect sessions
  fee = 0.01 # 100 charge units per monetary unit
# ***************************************************
  }
# Start computing billing amounts for each user.
  { _sum = cpu_P*$3 + kcm_P*$5 + con_P*$7 # compute prime usage
  _sum+= cpu_NP*$4+ kcm_NP*$6+ con_NP*$8 # add non-prime usage
  _sum+= blk*$9 + prc*$10 + ses*$11 + fee*$13 # add remaining amounts
  printf "%-8s %-10s %10.3f\n", $1, $2, _sum # display user total
  }' tacct_ASC.tmp_ >$_outfile # write output from awk to appropriate file
rm tacct_ASC.tmp_ # remove the temporary ASCII file
```

System accounting files

This section contains descriptions of the different files processed by SPP-UX system accounting. The files are grouped according to the directory in which they are found.

Table 18 describes the files in the /usr/adm directory.

Table 18 Files in /usr/adm

File name	Description
diskdiag	Diagnostic output from the execution of disk space accounting commands
dtmp	Output from the `acctdusg` program
fee	Output from the `chargefee` command (ASCII total accounting records)
pacct	The current active process accounting file
pacct?	Process accounting files switched via `turnacct switch`

Table 19 describes the files contained in the /usr/adm/acct/nite directory.

Table 19 Files in /usr/adm/acct/nite

File name	Description
active	Used by `runacct` to record progress. It contains warning and error messages. active *mmdd* is the same as active after `runacct` detects an error.
ctacct. *mmdd*	Total accounting records created from connect session accounting where *mmdd* is the month and day the file was created.
ctmp	Output of `acctcon1` — connect session records.
daycms	ASCII daily command summary used by `prdaily`.
daytacct	Total accounting records for current day.
disktacct	Total accounting records created by the `dodisk` command.
fd2log	Diagnostic output from the execution of `runacct` (refer to crontab file entry).
lastdate	The last day that `runacct` was executed, in `date | # +%m%d` format. See the date(1) man page for a description of *+%m%d* date format.)
lock	Used to control serial use of `runacct`.
lock1	Used to control serial use of `runacct`.
lineuse	Terminal (tty) line usage report used by `prdaily`.
log	Diagnostics output from `acctcon1`.
log *mmdd*	Same as log after `runacct` detects an error.

Table 19 Files in /usr/adm/acct/nite —(continued)

File name	Description
reboots	Contains beginning and ending dates from wtmp, and a listing of reboots.
statefile	Used to record the current state being executed by `runacct`.
tmpwtmp	wtmp file, corrected by `wtmpfix`.
wtmperror	Error messages, if any, from `wtmpfix`.
wtmperror *mmdd*	Same as wtmperror after `runacct` detects an error.
wtmp. *mmdd*	The previous day's wtmp file.

Table 20 describes the files contained in the /usr/adm/acct/sum directory.

Table 20 Files in /usr/adm/acct/sum

File name	Description
cms	Total command summary file for current month in internal summary format.
cmsprev	Command summary file without latest update.
daycms	Command summary file for previous day in internal summary format.
loginlog	Shows the last login date for each user.
rpt mmdd	Daily accounting report for date *mmdd*.
tacct	Cumulative total accounting file for current month.
tacctprev	Same as tacct without latest update.
tacct mmdd	Total accounting file for date *mmdd*.
wtmp. mmdd	Saved copy of wtmp file for date *mmdd*. Removed after reboot.

Table 21 describes the files contained in the /usr/adm/acct/fiscal directory.

Table 21 Files in /usr/adm/acct/fiscal

File name	Description
cms*month*	Total command summary for month *month* in internal summary format.
fscrpt*month*	Report similar to the `prdaily` report for the month *month*.
tacct*month*	Total accounting file for the month *month*.

Open Boot PROM (OBP) and Restricted OBP

9

What is it?

Open Boot PROM (OBP) is a software architecture for the firmware that controls an Exemplar before the operating system begins executing. It runs entirely in the CPUs of the computer—not in the system utility board (MU) or the teststation.

OBP provides three main areas of functionality to the SPP:

- Autoconfiguration—OBP completely autoconfigures the I/O subsystem and provides a complete and accurate description of all available hardware. Information is stored in a hierarchical data structure called the *device tree* and provides a C-callable interface to the kernel for queries to the database.
- Boots the operating system—In the SPP, booting loads 3 different images into memory before starting
  - Kernel
  - Server and emulator
  - Tunables file
- Maintains configuration parameters—OBP accesses three kinds of *non-volatile configuration information*
  - Low-level hardware configuration—Managed by
    * Utilities firmware and primary loader
    * Utilities running on the test-station
  - Operating system boot configuration parameters—Managed by OBP, called NVRAM parameters or config options.
    * To display config options use `printenv`.
    * To change config options use `setenv`.
    * NVRAM parameters include: `auto-boot?` and `boot-device`

- Mach and Unix configuration parameters—Stored in an ASCII tunables file, which OBP loads into the *device tree* for later access by the Mach kernel or Unix server.

See "OBP configuration parameters" on page 227" for definitions of individual parameters and their defaults.

Booting

When the primary loader boots OBP and the auto-boot? parameter is false, OBP prints a startup message and enters interactive mode. See the section on "OBP configuration parameters" on page 227 for more detail about the auto-boot? NVRAM parameter.

OBP's interactive prompt is

[*X*:*Y*] `ok`

where

X

Is the hypernode's node id.

Y

Is the CPU number upon which OBP's interactive interpreter is running.

Boot commands

Common commands for booting include

- `boot`
- `load`
- `reset`

This section describes each command and its syntax.

`boot` [*device-spec*] [*boot-directory*] [*boot-arguments*]

Boots SPP-UX or another standalone program.

device-spec

Is a full OBP device pathname, beginning at /, or a device alias, created with the `devalias` or `nvalias` commands. You can see the default aliases by typing `devalias` without arguments at the `ok` prompt. If no *device-spec* is given, the current value of the NVRAM parameter `boot-device` is used.

boot-directory

If the boot device is a disk partition, *boot-directory* is a SPP-UX filesystem directory pathname residing in the filesystem stored upon that partition within which the files to be loaded are stored. *Boot-directory* may not be a symbolic link.

boot-arguments

Consist of everything else on the line. If not supplied on the command line, OBP uses the current value of the NVRAM parameter `boot-args`.

`load` [*device-spec*] [*boot-directory*] [*boot-arguments*]

Loads SPP-UX or a standalone program and returns to the `ok` prompt without starting the program. When you use load, you must type `go` to start the loaded program. Use load/go to set OBP breakpoints in the program being booted. The command arguments are the same as for `boot`.

`reset`

Reboots OBP. Similar to entering `do_reset` at the test-station prompt.

Boot related commands

Common commands related to booting include

- `setenv`
- `devalias`
- `nvalias`
- `nvunalias`

This section describes each command and its syntax

`setenv boot-directory` *full-unix-pathname*

Sets the directory in which OBP looks for the files mach, server, and tunables.

`setenv boot-device` [*full-obp-pathname* | *device-alias*]

Selects the boot-device which contains the program to be booted. If the boot-device is a disk device, it must be a partition of a Convex-labeled disk. If it is tape, the first file is loaded into memory.

`devalias` *alias-name full-obp-pathname*

Defines an alias representing the device path. If an alias with the same name already exists, the new value supersedes the old. For example,

```
boot sd4g
```

instead of

```
boot /mbus@0,ffec0000/sbus@f,fcffff00/Convex,afws@1,10000:narrow/sd@4,0:g
```

The alias is created as a property—name/value pair—under the software device /aliases, in OBP's device tree.

`nvalias` *alias-name full-obp-pathname*

Creates a device alias and adds a copy of the command to the nvramrc script executed at boot time—prior to probing the I/O system, so the device alias is non-volatile.

`nvunalias` *alias-name*

Deletes a device alias—created with `nvalias`–on *next* reboot of OBP, not immediately.

OBP's device tree

OBP's central data structure is the device tree, a hierarchical data structure constructed during the probing process.

The device tree describes the hardware devices attached to each hypernode; its organization is similar to an SPP-UX file system tree. The nodes referred to are not hypernodes, but OBP software data structures.

The device tree is made up of nodes. The command show-devs displays the device tree and the commands `.cd` and `.pwd` change and display the current node, or position, in the tree. Each device node may have

- Properties—Describe each node and its associated devices. A synonym for property is attribute. The command `.attributes` displays the current device's properties. `.properties` is an alias for `.attributes`.
- Methods—Software procedures to access devices.
- Children—Other device nodes attached to that node. The parent node is also called a hierarchical node.

Most hierarchical nodes represent busses and associated controllers. Each such node defines a physical address space that distinguishes its connected devices from one another.

Each device connected to a hierarchical device is assigned a physical address within that address space. The form of a physical address is a pair of 32-bit numbers. The interpretation of the two numbers is specific to the hierarchical device driver. In general, the numbers are based upon some physical characteristic unique to the device, such as the bus's address or slot number where the device is installed. This prevents device addresses from changing when another device is installed

Some devices in the device tree represent software objects, not hardware objects. These include

- The /options device—In which each property is a config option, or NVRAM parameter.
- The /packages device—Which has several children that are, in effect, shared subroutine libraries for other devices. These include disk-label, hfs file-reader, and deblocker packages.
- The /chosen and /server-load-map devices—Communicate one time booting information to the kernel. These devices are created as a side-effect of the `load` and `boot` commands.

The complete, unambiguous device path of a device node lists the node names of all devices in the path from the root of the tree to the desired device. A device path is represented as a list of node names separated by slashes (/). For example

```
/mbus@0,ffec0000/sbus@f,fcffff00/CRES,cddi@1,400000
```

The implied root of the tree is the machine node, which is not named explicitly but may be indicated by a leading slash (/).

Using the `ls` command shows you the device name of the children of the current node and the internal memory address of the device node's vocabulary in the device tree.

Device name syntax

Each node in the device tree is identified as follows

driver-name@unit-address:device-arguments

where

driver-name

Names the device. Contains between 1 and 31 letters, digits, and punctuation characters {, _ + -}. Upper and lower case are distinct. By convention, this name includes the name of the device's manufacturer and the device's model name, separated by a comma (,). For publicly traded companies, the recommended spelling of the manufacturer name is that company's stock symbol. Inclusion of the manufacturer name within *driver-name* is especially important for devices intended to plug into standard buses. It is less important for devices permanently attached to a particular machine.

unit-address

Is the physical address of the device within the address space defined by its parent node. *unit-address* consists of a pair of 32-bit numbers, represented as two hexadecimal numbers

separated by a comma (,). The interpretation of the two numbers depends upon the parent node.

device-arguments

Is an optional sequence of letters or digits. Upper and lower case are distinct. The length is arbitrary although sequences of more than eight characters are discouraged. Interpreted by the driver, *device-arguments* typically represents additional device information such as partition name or protocol. *device-arguments* and its preceding colon (:) may be omitted when specifying a node-name. *device-arguments* does not serve to identify the device node. Most devices do not use device-arguments, and ignore any that are passed in.

An example of *device-arguments*

```
/mbus@0,ffec0000/sbus@f,fcffff00/Convex,afws@1,10000/sd@2,0:a
```

Designates booting from the `a` partition on the disk at SCSI target 2.

The general form allows any SCSI disk to be specified:

```
/mbus@0,ffec0000/sbus@f,fcffff00/Convex,afws@1,10000/sd@T,U:P
```

Replace *T* with the SCSI target number, default is 0.

Replace *U* with the logical unit number, default is 0.

Replace *P* with the partition letter (`a-n`, `cntl`), default is `cntl`.

For example,

```
/mbus@0,ffec0000/sbus@f,fcffff00/Convex,afws@1,10000/sd@3,0:g
```

If you specify no partition,

```
/mbus@0,ffec0000/sbus@f,fcffff00/Convex,afws@1,10000/sd@3,0
```

or the control partition,

```
/mbus@0,ffec0000/sbus@f,fcffff00/Convex,afws@1,10000/sd@3,0:cntl
```

OBP assumes an unlabeled disk, and treats the entire disk device as a single unpartitioned device. Such usage is only applied for special stand alone disk utilities and scratch install programs. You cannot boot from the `cntl` partition.

Device path syntax

As the device tree diverges, a route, or device path, to individual devices becomes apparent. The device path represents the type of device and where that device is located within the device tree. A full device name is a series of device paths separated by slashes.

Device path syntax is

name@address:arguments

where

name

Represents an address on the main system bus. Valid names for Exemplar systems include:

`sd`

SCSI disk

`st`

SCSI tape

`le`

Ethernet

`CRES, cddi, fddi`

FDDI

`hippi`

HIPPI

`convex, afsw`

SCSI

@address

Represents a physical space unique to the device. *address* consists of two 32-bit numbers, usually in hexadecimal format, separated by a comma. The interpretation of these two numbers depends on the location of the device in the device tree.

:arguments

(Optional) Passes additional information to the device's software. Valid options are specific to each device package. *argument* usually shows additional device information, such as disk partition.

Traversing the device tree

From the interactive prompt, display the hypernode's device tree with the `show-devs` command. Other commands for looking at the device tree information include: `cd`, `ls`, and `pwd`. The following examples illustrate usage of `show-devs`, `cd`, and `.attributes` to examine the device tree and the properties of one of the CPU devices and the physical memory device.

```
[0:2] ok show-devs
```

```
/mbus@0,ffec0000
/mbus@0,ffed0000
/HP,PA71000@0,ffe60400
/HP,PA71000@0,ffe40400
/HP,PA71000@0,ffe20400
/HP,PA71000@0,ffe00400
/virtual-memory@0,0
/memory@0,0
/dart@0,f0200080
/console@0,f0200a00
/openprom
/chosen
/aliases
/options
/packages
/mbus@0,ffec0000/sbus@f,fcffff00
/mbus@0,ffec0000/sbus@f,fdffff00
/mbus@0,ffec0000/sbus@f,fcffff00/Convex,afws@1,10000
/mbus@0,ffec0000/sbus@f,fcffff00/Convex,afws@1,10000/st
/mbus@0,ffec0000/sbus@f,fcffff00/Convex,afws@1,10000/sd
/mbus@0,ffed0000/sbus@f,fcffff00
/mbus@0,ffed0000/sbus@f,fdffff00
/mbus@0,ffed0000/sbus@f,fcffff00/CRES,cddi@1,400000
/packages/obp-tftp
/packages/deblocker
/packages/tape-label
/packages/disk-label
/packages/hfs
```

To set the current device to CPU # 4—the path is from the `show-devs` output.

```
[0:2] ok cd /HP,PA71000@0,ffe40400
```

Check your current location with `pwd`:

```
[0:2] ok pwd
/HP,PA71000@0,ffe40400
```

To see the properties of CPU # 4, use the `.attributes` command.

```
[0:2] ok .attributes
```

```
name                    HP,PA71000
reg                     00000000  ffe40400  00000004
device_type             cpu
clock-frequency         05f5e100
page-size               00001000
icache-line-size        00000020
icache-nline            00008000
dcache-line-size        00000020
dcache-nline            00008000
tlb-purge-select        00000003
num-comb-btlb           00000010
num-data-btlb           00000000
num-inst-btlb           00000000
min-btlb-size           00000080
max-btlb-size           00004000
calibration-factor      000f4240
cpu-id                  00000004
```

Device short forms

When OBP searches for a particular node—and either the *driver-name* or *@unit-address* portion of the node-name is missing—it chooses the first node matching the portion that is present.

It is possible to use a short form of a device pathname when you move around the device tree. *If* there is only one device of a certain type in the device tree, you can omit the parents of the device to `cd` to it. So

```
cd /CRES,fddi
```

places the current device node at the same place as

```
cd /mbus@0,ffed0000/sbus@f,fcffff00/CRES,cddi@1,400000
```

Device aliases

A device specifier is either a device path or an alias. An alias represents an entire device path, not just a single component. Predefined aliases include the

- Default root device.
- Default install device.

Use the `devalias` command to see device aliases or to create new ones. These aliases disappear if the machine is reset.

You can make an alias permanent by using `nvalias`. This command stores the `devalias` command in nvramrc which causes the alias to be redefined when the machine is reset. `nvunalias` deletes the alias in the nvramrc, but only takes effect *after* OBP reboots.

Standard device properties

OBP defines some standard types of device properties or attributes. Each property has a name and a value, though the value may be zero-length. See Table 22 for a summary.

Table 22 OBP standard device properties

Device	Value
name	A string value that defines the name of a package or driver. Examples: mbus, sbus, pci, Convex,afws, fddi
reg	An array of three 32 bit integers that defines a package's registers. (64-bit hi-lo address + length). The first register entry is the package's unit-number in the device specifier for the node. Example: 00000000 ffec0000 00000040
device_type	A string value that specifies the implemented interface. Examples: block, byte, network, cpu, scsi-2, hierarchical
model	A string value that specifies a manufacturer dependent model name and number—including revision level—for this device. The format of the text string is arbitrary, although in conventional usage, the string begins with the name of the device's manufacturer as with the *name* property. Although there is no standard interpretation for the value of the *model* property, a specific device driver might use it to learn the revision level of its particular device.
address	An arbitrary number of integer (virt-addr) values that specify the virtual addresses of one or more memory-mapped regions on this device. This property typically reports the virtual address of regions that the firmware has already mapped, so that client programs can reuse those mappings. The correspondence between declared addresses and the set of mappable regions for a particular device is device dependent.
compatible	Defines alternate *name* property values. Examples: sun4m, sun4c, cnx-spp1
adr-mask	Describes byte-swapping, if any, done by the device. It is a single 32 bit integer and affects the behavior of the following OBP commands: rb@ - Reads an 8-bit byte, preserving bit order. rb! - Stores an 8-bit byte, preserving bit order. rw@ - Reads a 16-bit word, preserving bit order. rw! - Stores a 16-bit word, preserving bit order. rl@ - Reads a 32-bit longword, preserving bit order. rl! - Stores a 32-bit longword, preserving bit order.
ranges	Define a device's physical address. Use the `devinfo` command to display ranges. A single *range* is defined by `child-phys parent-phys` *size*. Each address is encoded with two 32-bit integers defining a 'high-low' 64-bit address. *size* is a 32 bit value representing the size in bytes of the *range*. Multiple ranges may exist.

OBP configuration parameters

The commands to set and display OBP configuration parameters are:

`printenv`

Lists the entire set of parameters along with the current and default values. Some values are truncated for display

`printenv` *parameter-name*

Displays only the named parameter.

`setenv` *parameter-name parameter-value*

Changes the current value of a parameter. The change is nonvolatile and is used the next time OBP boots. The default value cannot be changed, it is a compile-time value. When OBP compiles, all current values are set to their defaults.

`set-default` *parameter-name*

Restores the current value of an NVRAM configuration variable, *parameter-name*, to the default value.

`set-defaults`

Executes a `set-default` for all variables.

NVRAM configuration parameters

To list the current NVRAM configuration parameters and default values use the `printenv` command.

```
[0:2] ok printenv
Parameter Name          Value                          Default Value

log-device              /console                       /console
output-device           /console                       /console
input-device            /console                       /console
lu-hack?                true                           true
u0sb0
u0sb1
u0sb2
u0sb3
u1sb0
u1sb1
u1sb2
u1sb3
sbus-probe-list         U1SB1 U1SB0 U0SB1 U0SB0        U1SB1 U1SB0 U0SB1 U0SB0
fcode-debug?            false                          false
local-mac-address?      false                          false
tz                      CST6CDT,91/0300,301/0100       CST6CDT,91/0300,301/0100
diag-device             none                           none
diag-file               none                           none
boot-device             sd0a                           sd0a
boot-directory          /os                            /os
```

```
boot-file              mach                          mach
boot-args
ramdisk-file           ramdisk                       ramdisk
cluster-boot?          true                          true
load-tunables?         true                          true
load-ramdisk?          false                         false
auto-boot?             false                         false
watchdog-reboot?       false                         false
screen-#columns        80                            80
screen-#rows           24                            24
use-nvramrc?           true                          true
nvramrc
security-mode          none                          none
security-password                                    no default
security-#badlogins    0                             no default
last-hardware-update                                 no default
mfg-switch?            false                         false
diag-switch?           false                         false
```

`log-device`
`output-device`
`input-device`

Contain the OBP device tree pathname for the console device, /console, OBP uses when booting. The log-device is the device that OBP uses for log events.

`u0sb0 u0sb1 u0sb2 u0sb3`
`u1sb0 u1sb1 u1sb2 u1sb3`

Allow OBP to compile a different FCode device driver built into OBP—called a drop-in driver— instead of the one in the controller's ROM. sb2 and sb3 are currently undefined.

The default value for each of these is the null-string. A non-null value for one of these must be the name of an FCode driver compiled into OBP. It selects the driver that OBP compiles as it probes that SBus slot, assuming a controller actually exists in the slot.

Since OBP probes all implemented SBus slots automatically, there should be no reason for you to change these NVRAM parameters.

`sbus-probe-list`

Lists SBus controllers that OBP probes. The default list is all four controllers:

U1SB1 U1SB0 U0SB1 U0SB0

OBP probes the SBus slots named in left-to-right order. The default list is in decreasing order because the devices created in the device tree are listed out in most-recently-created order by `show-devs`.

`fcode-debug?`

Used during development of FCode drivers. It should be left `false`.

`local-mac-address?`

Tells the Ethernet FCode driver in an SBus ethernet controller to get the Ethernet address from its own property list or any parent device property list, instead of using the global system-mac-address.

`tz`

Allows OBP to know what time zone it is in. The value should be the same as the TZ environment variable for SPP-UX and HP-UX at the local site. The default is for Richardson, Texas, USA.

`diag-device diag-file`

Replace `boot-device` and `boot-file` when the parameter `diag-switch?` is set to true. Default values for both of these parameters are `none`.

`boot-device`
`boot-directory`
`boot-file`
`boot-args`

Names an OBP device path name or alias when you type the short version of the `load` or `boot` commands. The default value is `sd0a`—the alias for the root device. OBP attempts to open the named partition if the disk is correctly labeled.

If the `boot-file` is *mach*, OBP loads the files named *tunables* and *server* from the directory named by `boot-directory` into memory. The file named by the option `boot-file` is then loaded and the system is setup to start the program.

If the named device is a SCSI tape device—`st`, a different rule is used—the `boot-directory` is ignored. The named file, file 0 by default, loads into RAM at 0x00100000, and OBP sets up the cpu state for entry at address 0x00100000. The scratch install utility boots from tape.

`cluster-boot?`

Alters the behavior of the OBP system console. If true, each hypernode opens a separate X-term for the system console. If false, all use the same X-term.

`load-tunables?`

Boots `boot-file` from disk and loads a *tunables* and a *server* file. If you want to boot the `boot-file` without a *tunables* or

server file loaded—as with a standalone diagnostic program—set to false. Default is true.

`auto-boot?`

Determines if OBP automatically attempts to reboot the operating system after a reset. If true, OBP probes the I/O system and boots the hypernode as though you typed `boot` at the `ok` prompt. The NVRAM values of `boot-device`, `boot-directory`,`boot-file`,`server-file`,`tunables-file`, `load-tunables?`,`load-server?` and `diag-switch?` determine what program boots.

The default values boot SPP-UX from the root disk partition.

After OBP probes the I/O system, you may interrupt this process, by pressing and holding **ESCAPE** at the right moment. OBP must poll the keyboard, as interrupts are not used in OBP device drivers, and the polling window is brief, only a 10 seconds. The message

```
To interrupt autoboot, press and hold the ESCAPE
key.
```

prints after the first banner—OBP Hard Boot— when the polling window begins. You must be quick to interrupt autoboot.

`watchdog-reboot?`

Determines if OBP automatically reboots after a watchdog reset. Default is false, true enables automatic rebooting.

`screen-#columns`
`screen-#rows`

Sets the width and height of OBP's screen. OBP assumes it is using a dumb terminal though the test station is actually an X-term. For commands that do many lines of screen output, OBP has a built-in pager and the `screen-#rows` parameter controls when to pause output.

`use-nvramrc?`

Determines whether or not OBP attempts to execute commands in the nvramrc script during startup. The default, true, executes the script *before* the MBUS part of the device tree builds, so no commands can refer to I/O devices that have yet to be created—such as creating a property in a device node. It should be left true.

`nvramrc`

Executes when `use-nvramrc?` is true. Default value is empty, a zero-length file. The maximum size depends upon how much NVRAM is used by the rest of the NVRAM parameters.

Total reserved space is 16K; nvramrc gets the space at the end of the 16K not used by the other parameters.

Execute the nvramrc script at any time from the `ok` prompt with the command `nvrun`.

You can *not* modify nvramrc with `setenv`. You must edit the contents of nvramrc directly with the built in OBP line editor—`nvedit`. After editing the file—exit with ^C—save your changes with the command `nvstore`.

`security-mode`
`security-password`
`security-#badlogins`

Allow OBP to be configured so that a password is required to access most commands. Three access protection modes are provided

- Non-secure mode (security-mode = none)—does not require a password for any command.
- Command secure mode (security-mode = command)—Requires a password to execute any command except for `go` and `boot` from the default device and file. Automatic booting after power-on is allowed.
- Fully-secure mode (security-mode = full)—Requires a password for any command except `go`. Automatic booting is disabled—the machine does not automatically reboot after a power-failure).

Set the security level with `setenv security-mode`. Set or change the password—a printable ASCII string of up to 8 characters—with the `password` command, which prompts for the new password twice, without echoing the password. The new password takes effect immediately.

Issuing a command which requires a password causes the command interpreter to prompt for the password, without echoing, before executing the command. The command interpreter asks for the password again after an abort sequence, reset, or exit from a standalone program. Incorrect password entries increment `security-#badlogins`.

Do no use the `setenv` command to set `security-password`. Use the `password` command. This prevents display or echo of the password in a command history.

`last-hardware-update`

Records the last time the machine configuration was updated.

`mfg-switch?`

Resets all parameters to their defaults, at the *next* OBP reboot, if the NVRAM becomes corrupted.

`diag-switch?`

Sets `diagnostic-mode?` to true, which boots OBP from

- `diag-device` instead of `boot-device`.
- `diag-file` instead of `boot-file`.

This also prints out the names of new devices as they are being compiled into the device tree at I/O system probe time.

Using Restricted OBP

The Restricted OBP (ROBP) utility enables you to get information about OBP properties and nodes without having to shutdown the system and bring up the OBP prompt.

While you can retrieve information from the OBP database; you cannot change any OBP property with ROBP.

ROBP requires a special file for each hypernode in the complex. These special files are located in the /dev directory and have the form

`obp0`

where the number at the end of the special file corresponds to the hypernode number. The ROBP utility creates the required special files if you have root privileges.

You can use ROBP interactively or non-interactively.

Interactive interface

Type `robp` without arguments to invoke the interactive interface.

The interactive interface is identical to that of OBP. Entering

`help`

Prints the list of commands supported by ROBP.

`help` *command*

Obtains more information about a specific command.

Command line interface

Specify colon separated OBP commands you wish to execute as arguments on the command line.

`robp` *command1* : *command2* : *command3*:... : *commandn*

See the robp(1M) man page for more information.

Setting up licenses 10

Overview

Correct operation of your Exemplar system requires a license key for each software product. If your users do not acquire a license, they are instructed to contact you for assistance.

Convex products are licensed using the FlexLM licensing system. Refer to the SPP-UX distribution notice that accompanied your software release and the *FlexLM End User Manual* for licensing information specific to your version of the operating system.

Types of licenses

There are two types of licenses for Exemplar software products:

- Per CPU—The software is licensed for use on a system with no more than a given number of CPUs.
- Per User—The software is licensed for a specific number of simultaneous users. The same UID can run any number of copies, and it only counts as one user. The largest license allows unlimited simultaneous executions. The per user license is applied as follows
  - For SPP-UX—A user is a nonsystem UID with at least one process executing on the system. Nonsystem UIDs are generally UIDs greater than 99.
  - For layered products—A user is a unique UID running that program. A single user running multiple copies of a program consumes only a single license. This allows operations like a parallel make or multiple CXdb invocations for debugging PVM and MPIch programs.

Table 23lists Exemplar products, required license types, and capacities.

Table 23 Exemplar products and licensing policies

Product	Policy	Capacity
SPP-UX	Per User	2, 8, 16, 32, 64, unlimited
NQS+	Per CPU	4, 8, 16, 64, 12
Subcomplex manager	Per CPU	4, 8, 16, 64, 128
Fortran	Per User	1, 5, 10, 20, unlimited
C	Per User	1, 5, 10, 20, unlimited
AOP	Per User	1, 5, 10, 20, unlimited
Mlib	Per CPU	4, 8, 16, 64, 128
CXdb	Per User	1, 5, 10, 20, 50, unlimited
CXpa	Per User	1, 5, 10, 20, 50, unlimited
PVM/GSM and MPIch	Per User	1, 5, 10, 20, 50, unlimited

How licensing works

When you attempt to use a licensed product FlexLM receives a request in the form of a key—strings of printable ASCII characters generated by an encryption process. If the key is matched in the FlexLM database you are allowed to use the product. If the key is not located, you are refused service and a message prompts you to contact your system administrator.

You can obtain keys in any of the following ways:

1. Software shipments include keys printed on paper.
2. Keys are available online through Viper.
3. TAC provides keys by phone or fax.

License administration

The following license administration tools are located in the /usr/local/flexlm directory:

- lmcksum—Checksum the license file.
- lmdown—Graceful shutdown of all license daemons.
- lmgrd—Flexible license manager daemon.
- lmhostid—Report the hostid of a system.

- lmremove—Remove specific licenses and return them to license pool.
- lmreread—Tell the license daemon to reread the license file.
- lmstat—Report status on license manager daemons and feature usage.
- lmutil—Generic FlexLM utility program.
- lmver—Report the FlexLM version of a library or binary file.
- license.opt—Site administrator options file for FlexLM licensed applications.

Online man pages are available on systems running FlexLM.

Note

You must set the environment variable LM_LICENSE_FILE to /usr/local/flexlm/licenses/convex.dat

Configuring licenses

If FlexLM has not been previously configured on your Exemplar system you will receive the following message:

```
This release requires a license to work.  If you do
not have the FlexLm license keys for all the
products, contact the TAC. A license will only be
given to sites that are approved by SPP Product
Support.
```

Perform the following steps to enable licensing for your system.

Step 1 Add the following lines to /etc/rc above the localrc function:

```
#
# start_licenses
#
start_licenses()
{
START_FLEXLM=1
if [ "$START_FLEXLM" -eq 1 -a -x /usr/local/flexlm/rc.convex.flex ]; then
    echo "starting license system, log is in /usr/local/flexlm/log"
    /usr/local/flexlm/rc.convex.flex
fi
/etc/gscl

}
```

Step 2 Add a call to `start_licenses` in /etc/rc above localrc:

```
#       audit_start

        start_licenses
        localrc
```

Step 3 Enter license keys furnished for your site into the /usr/local/flexlm/license/convex.dat file. See "Activating a license" on page 236 for more information.

Step 4 Start the license daemon—/usr/local/flexlm/lmgrd. Enter

```
/usr/local/flexlm/rc.convex.flex
```

Adding a license key

Perform the following steps to enable a license key for a running system:

Step 1 Add the new license key to /usr/local/flexlm/licenses/convex.dat. See "Activating a license" for more information.

Step 2 Reread the license file. Enter

```
/usr/local/flexlm/lmreread
```

Step 3 Load the new license key into the kernel. Enter

```
/etc/gscl
```

Activating a license

To activate the license for a software product, edit the file /usr/local/flexlm/licenses/convex.dat. Add the license key to this file on a new line, after the SERVER statement and the DAEMON statement as shown in the following sample file:

```
SERVER demospp 12063 765
DAEMON convex_ls /usr/local/flexlm/convex_ls
FEATURE sppux convex_ls 99.990 1-jan-2099 0 EB2E4DFDFB40BC2F8F75 \
        VENDOR_STRING=UIDS=0;PART=710-98765-005 HOSTID=12063 ck=54
FEATURE scm convex_ls 99.990 1-jan-2099 0 9B6EEDFDED37F6C0621 \
        VENDOR_STRING=CPUS=128;PART=710-123450-004 HOSTID=12063 ck=34
FEATURE cc convex_ls 99.990 1-jan-2099 0 8B327A788A35FFED135B \
        VENDOR_STRING=;PART=123-000290-006 HOSTID=12063 DUP_GROUP=U ck=32
```

Each license key consists of two lines with a continuation symbol or backslash (\) at the end of the first line.

License messages

Unsuccessful attempts to use SPP-UX or layered products produce messages of the form

`Unable to obtain license for` *product*: *further detail*

where

product

Is the Exemplar product name.

further detail

Is either a description of the exact problem or detailed error information.

For example:

```
Unable to obtain license for spp-ux: Too many unique
uids active
```

The system that has reached the licensed user limit.

```
Unable to obtain license for nqs: Complex size (16)
exceeds license (8)
```

An NQS license for an 8-CPU complex is running on a 16-CPU complex.

```
Unable to obtain license for cxdb: Licensed number
of users already reached  (-4,132)
```

The licensed number of users are already using CXdb.

The numbers in parenthesis are FlexLM error codes. These are provided to help you diagnose problems.

Obtaining license keys via the WWW

If you are a Convex customer or are otherwise subscribed to our web service, you can access keys for all of your SPP systems at the following URL:

http://www.convex.com:90/cust-bin/keys

Enter your username and the password you received upon subscribing.

If you wish to become a subscriber, follow this URL:

http://www.convex.com:90/subscribe.html

SPP-UX system tunable parameters

A

SPP-UX system tunables file

The /os/tunables file contains tunable parameters that allow you to adjust system performance. The parameters from the /os/tunables file are read and set for the system each time SPP-UX boots. If you change the values for parameters in /os/tunables, the new values take effect the next time the system boots.

The /os/tunables file contains parameters for both the SPP-UX microkernel and the SPP-UX server.

When you install a new version of SPP-UX, your /os/tunables file is not automatically updated to add new tunables. Instead, a new tunables file is installed in the /etc/newconfig directory. After installing a new version of SPP-UX, check the /etc/newconfig/tunables file against your /os/tunables file to see if new tunables have been added or if the range of values for tunables has changed.

The following sample output illustrates the format of the /os/tunables file.

```
#       Microkernel tunables
#
# Event Logger internal buffer
Event Logger,buffer size:desc=Event Logger Buffer Size:1=64k:default=64k:

# Control the LCD processor heartbeat update (0:Off, 1:On)
LCD Heart Beat,control:desc=Control for LCD update:1=1:default=1:
#
#       Server tunables
#

# Percent of memory dedicated to the filesystem buffer cache
Fileserver,buffer_cache_percent:1[0..15]=10:
```

```
# Have all servers panic gracefully when any server panics (0:off, 1:on)
Server,distribute_panic:1[0..15]=1:

# Default stack size limit
Server,dflssiz:desc=Default Stack Size Limit:1=8M:

# Maximum stack size adjustable via rlimit
Server,maxssiz:desc=Maximum Stack Size Limit:1=766M:

# Default size of data segment
Server,dfldsiz:desc=Default Data Size Limit:1=512M:

# Maximum size of data segment adjustable via rlimit
Server,maxdsiz:desc=Maximum Data Size Limit:1=3072M:

# number of ptys
Server,number_pty:1=16:

# Tune the maximum number of mounted filesystems
Server,nmount:desc=number of fs to mount:1=40:
```

Microkernel tunables

Table 24 lists the microkernel tunable parameters.

Table 24 Microkernel tunable parameters

Parameter	Range of values/ default value	Description
Crashdump, do_compression	0 | 1 1	Determines whether crashdump output is compressed (1) or not (0)
Crashdump, do_crashreport	*0 | 1* 1	Determines whether crashdumps are enabled (1) or disabled (0)
Crashdump, do_serverdump	*0 | 1* 1	Determines whether server data is dumped for each node during a crashdump (1) or not (0)
Crashdump, do_kerneldump	*0 | 1* 1	Determines whether a crashdump of the microkernel is generated when the system crashes (1) or not (0)
Crashdump, do_swdump	*0 | 1* 1	Enables (1) or disables (0) the crashdump capability

Table 24 Microkernel tunable parameters (continued)

Parameter	Range of values/ default value	Description
Crashdump, panic_graceful	*0 | 1* 1	Determines whether messages are printed when a microkernel panic occurs (1) or not (0)
Crashdump, panic_query	*0 | 1* 1	Determines whether `crashdump` prints queries to the system console during a crashdump (1) or not (0)
Crashdump, panic_reboot	*0 | 1* 1	Determines whether to enter Open Boot after a crashdump (1) or not (0)
Event Logger, buffer size	*0 to 65536* 0	Determines the size of the event logger buffer in bytes; larger sizes improve event logging at the expense of available physical memory
LCD Heart Beat, control	*0 | 1* 1	Sets (1) or clears (0) a flag that determines whether the Exemplar LCD displays a heartbeat and the state of each CPU, updated every 1/4 second
Stripe Disk, max stripes	*1 to int_max* 24	Determines the maximum number of disk stripes per node. The maximum number of stripes per system is 256.

Server tunables

Table 25 lists the server tunable parameters.

Table 25 Server tunable parameters

Parameter	Range of values/ default value	Description
Fileserver, buffer_cache_percent	*0 to 100* 10	The buffer cache for each node to the specified percent of physical memory
Fileserver, disk_wdb_size	*0 to 128*1024* 1024	The maximum number of wired device buffers for hard disk drives

Table 25 Server tunable parameters (continued)

Parameter	Range of values/ default value	Description
Fileserver, tape_wdb_size	*0 to 16*1024*1024* 128*1024	The maximum number of wired device buffers for raw tape devices
Server, distribute_panic	*0 | 1* 1	Sets (1) or clears (0) a flag that determines whether a panic in the node's file server is distributed to all other nodes' file servers, or whether a panic in the node's file server causes only the local node's server to shut down
Server, acctresume	*0 to 100* 4	The percentage of file system space that must be free in order to reactivate process accounting after it is suspended due to insufficient free space
Server, acctsuspend	*0 to 100* 2	The percentage of file system space that must be free in order to allow process accounting
Server, dfldsiz	*512*1024 to 0x0c0000000* 64*1024*1024	The default size, in pages, of a process's data segment
Server, dflssiz	*512*1024 to 0x0c0001000* 64*1024*1024	The default size, in pages, of a process's stack
Server, dst	*0 | 1* 1	Sets (1) or clears (0) a flag that specifies whether to use daylight saving time
Server, incksum	*1 | 2 | 3* 2	Internet Protocol (IP) checksumming method. This parameter should be set to a value of 2 under normal circumstances. If the `netstat -s` command shows an abnormally high number of `ip`, `tcp`, or `udp` checksum errors, contact the Convex TAC for assistance in changing the value of this parameter.
Server, maxdsiz	*512*1024 to 0x0c0000000* 512*1024*1024	The maximum size, in pages, of a process's data segment
Server, maxfiles	*1-256* 256	The maximum number of files a process can have open at one time. This limit can be changed by using `setrlimit(2)`.

Table 25 Server tunable parameters (continued)

Parameter	Range of values/ default value	Description
Server, maxssiz	*512*1024 to0x0c0001000* 512*1024*1024	The maximum size, in pages, of a process's stack
Server, maxusers	*8 to 1024* 256	Tunes the number of user processes
Server, maxuprc	*8 to 1024* 256	The maximum number of child processes allowed per user
Server, msgmax	*0 to 65536* 8192	The maximum number of processes a user can have
Server, msgmnb	*0 to 65536* 16384	The maximum number of bytes allowed for all queued messages
Server, msgmni	*1 to int_max* 50	The number of message queue identifiers
Server, msgtql	*1 to int_max* 40	The number of message queue headers
Server, nbuf	*1 to 80* 10	The number of file system buffer cache headers
Server, ncallout	*26 to int_max* nproc+16	The number of timeouts that can be pending simultaneously
Server, nfile	*32 to int_max* (16*(nproc+16+max users)/10+32)	The maximum number of open files
Server, nflocks	*50 to 400* 200	The maximum number of file locks
Server, nmount	*1 to int_max* 20	The maximum number of mounted file systems
Server, nproc	*10 to int_max* (20+8*maxusers)	The maximum number of processes that can exist at one time.

Table 25 Server tunable parameters (continued)

Parameter	Range of values/ default value	Description
Server, npty	*16 to 27900* 60	The maximum number pseudo-terminals that can exist at one time
Server, semaem	*0 to int_max* 16384	The maximum value by which a semaphore can be adjusted due to the death of a process
Server, semmni	*2 to int_max* 64	The number of semaphore identifiers
Server, semmns	*2 to int_max* 128	The maximum number of semaphores
Server, semmnu	*1 to int_max* nproc_d	The maximum number of processes that can have `semaphore undo` requests on a semaphore
Server, semume	*1 to int_max* 10	The maximum number of semaphores on which a process can have can have a pending `semaphore undo` request
Server, semvmx	*1 to 65535* 32767	The maximum value of a semaphore
Server, shmmax	*2048 to 0xC00000000* 0x4000000	The maximum number of bytes in a shared memory segment
Server, shmmni	*3 to 1024* 200	The maximum number of shared memory segments
Server, shmseg	*3 to 1024* 120	The maximum number of shared memory segments that can be attached to a process at one time

The cnx_get_tunable system call

The cnx_get_tunable system call finds the *default* value of a system tunable stored in the /os/tunables file. It does not return the current value of a tunable if you have modified the default value.

Crashdump

B

Overview

The SPP-UX `crashdump` utility stores information about the state of the system to a raw disk partition in the event of a system crash. This information can be useful in some cases to help determine the cause of the system crash.

The `crashdump` utility is part of the SPP-UX microkernel. It runs when SPP-UX terminates abnormally. `crashdump` writes data to a raw disk partition that you create to hold `crashdump` information.

When a system crash occurs for a reason you do not understand, contact the Hewlett-Packard Convex Technical Assistance Center (TAC). If the TAC determines that you should send a crashdump file,

- Create a crashdump file using the `crashutil` command
- Make a tape containing the crashdump file
- Mail it to the TAC. See "Technical assistance" on page xxii for information about contacting the TAC.

See "SPP-UX system tunable parameters," on page 239 for related crashdump tunables.

Crashdump requirements

To use `crashdump`

- You must have a crashdump partition for each node you wish to dump. To see if a node already has a designated crashdump partition run

  /etc/crashutil -p

  The system returns a list of nodes and assigned disk partitions.
- Each node's crashdump partition must have the crashdump flag set with the `diskutil: set partition` command.

- You may only have one partition on a node marked as a crashdump partition. If you have more than one, delete the "C" flag with `diskutil`.
- You must have the following amount of space available:
  - A one hypernode system requires 40MB.
  - A multinode system requires 70MB.

  The amount of space needed depends on the amount of global shared memory on the SPP and the rate of compression achieved.

Creating a crashdump partition

You must create a crashdump partition on a disk drive on your Exemplar system to hold the data generated by `crashdump`. Use the `diskutil` command to create, modify, or delete a crashdump partition. The crashdump partition you choose should have 40MB of free space.

For more information about `diskutil` and the `diskutil` commands, see "The diskutil disk utility" on page 95.

Creating a crashdump partition

Use the following procedure to create a crashdump partition:

Step 1 Enter the `diskutil` command, followed by the `show disks`, `select disk`, and `show partition` commands, in order to see the available disks and partitions on your system:

```
# diskutil
DiskUtil: show disks
SD 0:0:2:0 mapped to sd0
SD 0:0:3:0 mapped to sd3
SD 0:0:4:0 mapped to sd1
DiskUtil: select disk sd0
DiskUtil: show partition
Logical disk name: sd0
partition table: (space available for file systems = 2098759)
part     offset      size    | partition description          | flags
-------------------------------------------------------------------
 a:          0K    819200K  |Root and /usr filesystem        | *
 b:     819200K   1048576K  |Default Pager Partition         | *D
 d:    1908736K    190023K  |/tmp filesystem                 | *
```

In the previous output, disk sd0 has free space between partitions b and d.

Step 2 Select the disk on which you wish to make a crashdump partition with the `select disk` command.

```
DiskUtil: select disk sdN
```

where *N* is the number for the logic disk on which you want to create the crashdump partition.

A one hypernode system should only require 40MB, while a multinode system may require as much as 70MB. The amount of space needed depends on the amount of global shared memory on the SPP and the rate of compression achieved.

Step 3 Use the `make partition` command to create a crashdump partition. For example:

```
DiskUtil: make partition c size size after b description "crash"
```

where *size* is appropriate size for your machine type.

Step 4 Set the `crashdump` flag for this partition with the `set partition` command. For example:

```
DiskUtil: set partition c flag crashdump
```

Step 5 Reboot the machine after exiting `diskutil`.

Changing or deleting a crashdump partition

Instead of creating a new crashdump partition, you can change an existing partition to a crashdump partition.

1. First make sure that all useful data has been removed from the partition, since it will be overwritten by `crashdump`.
2. Ensure that the selected partition is not mounted.
3. Use the `set partition` command to identify this partition as a crashdump partition:

   ```
   DiskUtil: set partition c flag crashdump
   ```

To stop using a partition as a crashdump partition, use the `unset partition` command:

```
DiskUtil: unset partition sd0c flag crashdump
```

Note

Do not create more than one crashdump partition on a single node; this causes the crashsystem to fail.

In order for your system to recognize any changes made to crash partitions, you must perform a system reboot.

Using crashsystem(1M)

The crashsystem(1M) command stores raw crashdump data on the designated partition for each hypernode. Execute crashsystem from the Exemplar test station. Enter

```
test_station % /spp/bin/crashsystem all
      crashsystem:  (version) (build date)
      Running this program will crash the SPP system
      Do you want to continue? [y/n] y
```

When the crashsystem command completes, the crashdump has *not* completed. The completion of crashsystem merely indicates that crashdump has been initiated on all nodes (or the node specified). Crashdump has completed when an 'ok' prompt appears on the system console.

To check the current status of a crashdump, you can spy on the crashdump virtual consoles. Output from crashdump on each node is directed to a different virtual console. You access virtual consoles from the test station with the `sn_cnsl` command. The crashdump virtual consoles start at 2090 for node 0 and increment by one for each node. To look at the crashdump output for node 1, you would execute the following command:

```
test_station% sn_cnsl -S 2091
```

The resulting output tells you whether the crashdump is

- In crashreport
- In kerneldump
- In serverdump
- Waiting for other nodes to complete
- Concluding by entering OBP

Monitoring the crashdump virtual consoles is not necessary for crashdump to work, only for debugging purposes.

Once all nodes have entered OBP, you may use the system console

```
sn_cnsl -F 1
```

and type 'boot' at the 'ok' prompt to boot your system in the usual manner.

Taking a crashdump on SPP usually takes only a few minutes. Even if the crashdump fails, informative messages notify you and you can proceed to reboot.

Creating a crashdump file with crashutil

If the TAC has determines that you should send a crashdump, use the `crashutil` command to create a crashdump report.

Each crashdump overwrites all crashdump data from previous crashdumps, if any exist. It is important to run `crashutil` on your SPP to retrieve crashdump data from the crashdump partition before initiating another crashdump. Due to the large amount of space they consume, crash files are compressed by default.

The `crashutil` command has the following format:

`crashutil [-d | -n | -p -s] -i` *block-partition* `-o` *outfile*

`-d`

Display the contents of the crashdump partition.

`-n`

Do not compress the output file. If this option is not specified, the crashdump file is compressed using `gzip` and a .z suffix is appended to the crashdump file.

`-p`

Display all possible crashdump partitions. You can have only one partition on a node marked as the crash partition.

`-s`

Calculate the size of an output crash file.

block-partition

The name of the raw disk partition containing the crashdump data.

outfile

The output file to which the crashdump file is to be written.

crashutil examples

To display possible crashdump partitions:

```
crashutil -p

 Current crashdump partitions:  last date written
        node 0 : sd0c   Tue May  7 15:50:37 1996
```

To write the compressed crashdump file to crashfile.o on partition `sd0c`, enter:

```
crashutil -i sd0c -o crashfile.o
```

```
gzip command: '/usr/contrib/bin/gzip > crashfile.0.z'
            Converting kernel address ranges
            Converting server address ranges
```

The .z suffix is automatically appended to the crashdump file name.

To display the contents of crashdump partition sd0c:

```
crashutil -d -i sd0c
```

```
Type: Crashreport   Date: Tue May  7 15:50:37 1996
  Size:   0.1 MB  Compressed:  0.1 MB  Offset: 0x00001000
Type: Kerneldump    Date: Tue May  7 15:50:37 1996
  Size:  60.8 MB  Compressed: 12.3 MB  Offset: 0x000149c9
Type: Serverdump    Date: Tue May  7 15:50:37 1996
  Size:  18.5 MB  Compressed:  3.2 MB  Offset: 0x00c66e2b
```

To calculate the size of an uncompressed output crash file:

```
crashutil -s -i sd0c
```

```
Generated file will be 47704396 bytes (45.5Meg)
```

Reading a crashdump file

SPP-UX crashdump files can be read using the `gdb` debugger. `gdb` is shipped with SPP-UX and is installed in /usr/bin on Exemplar systems.

To read a crashdump file in `gdb`, use the `gdb target` command, specifying the target crash and giving the name of the crashdump file.

In order to debug the SPP-UX kernel, enter the `gdb` command `attach 0` to attach to the kernel's address space. In order to attach to the SPP-UX server, use the `gdb` command `attach -3`.

gdb crashdump example

To start gdb and read the crashdump file crash.file, then attach to the SPP-UX kernel, enter the following commands:

```
% gdb mach_kernel
```

```
(gdb) target crash crash.file
#0 update_priority (thread=0x2da500) at
../../kern/sched_prim.c:1311/../../kern/sched_p
rim.c:1311: No such file or directory.
```

```
(gdb) attach 0
```

```
Attaching program:
/scratch/mach_kernel task 0
```

```
You can't do that without a process to
debug(gdb)
```

The message `You can't do that without a process to debug` is not an error message; you can ignore it.

Additional gdb crashdump commands

The `gdb` commands `info crash` and `info hardware` provide information while a crashdump file is attached to the debugger.

gdb info crash command

The `info crash` command prints information about the Exemplar system's state at the time the crashdump occurred. The following example illustrates the `info crash` command:

```
(gdb) info crash
Serial#:-1  Customer:krassh
Crash Time:Fri May 5 13:13:11 1995
Node id:0
Kernel:SPP-UX_mk   95.05.05.00 L23
firefox:/work3/tdavis/kernel
Server:Unknown
Panic message:
Hang
```

gdb info hardware command

The `info hardware` command prints information about the Exemplar system's hardware at the time the crashdump occurred. The following example illustrates the `info hardware` command:

```
(gdb) info hardware
CPU Rev T       CPA Rev B
CMC Dark        SCI REV Unknown
active_rings:4  memory_size:512 Meg
icache:1024 Kbytes      dcache:1024 Kbytes
```

Crashdump restrictions

Crashdump works by sending a high-priority machine check (HPMC) to gain control of the Exemplar processors. Any condition that would prevent a processor from receiving an HPMC can prevent crashdump from starting. For example, if you are unable to read and write to the Exemplar system's memory from the test station, crashdump will not work. Also, if an Exemplar processor has already received an HPMC, it is likely that crashdump will not work.

If a processor hangs due to network problems or space allocation problems, it may not be able to open the proper virtual console. In this case, the output for that virtual console is routed to the system console.

If you do not have a crash partition configured for a node, or do no reboot after a reconfiguration, the crashdump for that node will not work.

For more information and a list of common problems, see the crashsystem(1M) man page.

Glossary

This section defines terms used in this document.

A

ABI
Application binary interface. A software implementation that allows executable programs to run on computer systems with different hardware architectures. Convex SPP systems are implemented with an ABI that allows compatibility with other computer systems that use the Hewlett-Packard PA-RISC processors.

access permissions
Values associated with each file that control who has permission to read, write (modify), or execute the file.

alias
An alternative name for some object, especially an alternative variable name that refers to a memory location. Aliases can cause recurrences, which prevent the compiler from vectorizing or parallelizing parts of a program.

ASCII
American Standard Code for Information Interchange.

autoconfiguration
The process of determining, without human intervention, the set of devices that comprise a computer system and the characteristics of those devices, and of adjusting the firmware and software to behave appropriately with that set of devices.

B

block device
A hardware device that transmits and receives data in multiple-byte blocks (rather than by streams of individual bytes) or does block-buffered input/output.

block special file
A special file associated with a mass storage device (such as a hard disk or tape cartridge drive) that transfers data in multiple-byte blocks, rather than in a series of individual bytes. See *device file*.

buffer
A temporary storage area. Several types of buffers are used in computer systems, in both hardware and software. The most common types of buffers are those maintained by a computer operating system to mediate between processes and I/O devices.

bus
A data path shared by several components within a computer system. A typical example is an I/O bus, which can connect processors, memory, and I/O controllers.

byte (b)
A group of contiguous bits starting on an addressable boundary. In Convex machines, a byte is 8 bits in length.

C

cache, cache memory
A small, high-speed buffer memory used in modern computer systems to hold temporarily those portions of the contents of the main memory that are, or are believed to be, currently in use. Convex computers contain many separate caches, including logical caches, physical caches, and instruction caches.

cache purge
The act of invalidating or removing entries in a cache memory.

cacheable
Memory references which are eligible for cache *move ins* are said to be cacheable references.

cancel model script
The cancel model script, /usr/spool/lp/cmodel/rcmodel, is used to cancel a print request to a printer on a remote system.

CD-ROM file system
A Read Only Memory file system on Compact Disk. You can read data from a CD-ROM file system, but you cannot write to one.

character device file, character special file
A special file associated with I/O devices that transfer data byte-by-byte. Typical byte-mode I/O devices include printers, nine-track magnetic tape drives, and disk drives when accessed in *raw* mode. Disk drive access via character devices is typically

faster than via block devices. Character device files are sometimes called raw device files.

coherency
A term frequently applied to caches. If a data item is referenced by a particular processor on a multiprocessor system, the data is copied into that processor cache and is updated there if the processor modifies the data. If another processor references the data while a copy is still in the first processor cache, a mechanism is needed to ensure that the second processor does not use an outdated copy of the data from memory. The state that is achieved when both processors' caches always have the latest value for the data is called cache coherency.

communication line
A physical communication path, such as coaxial or fiber-optic cable.

complex
The complete set of processor and memory resources available on a Convex SPP system.

computer network
A system of interconnected computers that enables machines and their users to exchange information and share resources.

conventional compiler
A compiler that cannot perform interprocedural optimization. This term encompasses all Convex compilers except the Applications Compiler and virtually all compilers available from other vendors.

CTI cache
A cache within a node which contains coherent memory data fetched from other nodes in a Convex SPP system.

CTIRing
The ring interconnect that connects all the nodes of a multinode Convex SPP system together in a ring fashion. This is Convex implementation of SCI.

cylinder
On disk drives consisting of several disks, the arrangement of disk tracks under read/write heads that are in the same relative position.

D

DaRT
Diagnostic and Remote Test. The DaRT Bus is the Ethernet-like connection between the nodes of a Convex SPP system that is used for diagnostic purposes. The nodes are connected together in a bus fashion.

default group
A user can be a member of multiple groups, but only one of those groups is considered to be the user's primary or default group. In addition to being listed as a member of groups in the /etc/group file, an entry exists in the /etc/passwd file that indicates the user's primary group. When users first log in to the system, they are affiliated with their primary group.

default processor set
Each node of a Convex SPP system has a default processor set. All processors in a node are initially assigned to the default set when the node is booted. A processor remains in the default set until it is explicitly assigned to a subcomplex. When a processor is removed from a subcomplex, it is returned to the default set.

device alias
A shorthand representation of a *device path.*

device arguments
The component of a *device node name* that provides device-specific information.

device driver
The software responsible for managing low-level I/O operations for a particular hardware device of set of devices. A device driver contains all the device-specific code necessary to communicate with the device and provides a standard interface to the rest of the system. Device drivers can be implemented either in firmware or in the operating system.

device file
A file used by the computer to communicate with a device. The file tells the operating system the location of the device and what device driver to use. There are block device files (used for transmitting data in multiple-byte blocks) and character device files (used for transmitting data byte-by-byte). Device files are typically stored in the /dev directory. Block device files are stored in the directory /dev/dsk; character device file are stored in the directory /dev/rdsk.

device interface
One of the standard interfaces specified in the IEEE P1275 Boot Firmware standard, allowing devices to be identified,

characterized, and used to assist other firmware functions such as booting.

device node
An entry in a *device tree*, usually describing a single device or bus. A device node may have multiple child nodes and has only one parent node.

device node name
A text string of the form *driver-name@unit-address:device-arguments*, identifying a *device node* within the address space of its parent.

device path
A textual name identifying a *device node* by showing its position in the *device tree*.

device specifier
A *device path*, a *device alias*, or a hybrid path that begins with a *device alias* and ends with a *device path*.

device swap space
A disk or disk section reserved exclusively as swap space.

device tree
A data structure that describes the set of devices attached to a computer system, including both permanently installed devices and plug-in devices. The IEEE P1275 Boot Firmware standard specifies a structure for the device tree; this structure is used in Convex SPP systems.

device type
A set of properties and package classes that a *device node* is expected to implement.

direct memory access (DMA)
A procedure or method defined for gaining direct access to main storage and achieving data transfers without involving the Central Processing Unit.

disk quotas
Disk usage limits that a system administrator can assign to users of a file system.

disk section
A logical division or partition on a hard disk in which a file system or a swap location can be placed. Convex Exemplar systems use disks that are partitioned in numerous sections.

distributed memory
A memory architecture used in multi-CPU systems, in which the system memory is physically divided among the processors. In most distributed memory architectures, distributed memory is accessible from only a single processor; Convex GSM is an exception. See also *globally shared memory (GSM)*.

doublet
A unit of computer data consisting of sixteen bits.

drive
A mechanical device on which storage media, such as tapes or disks, are placed. A drive contains the physical devices used to position, read from, and write to the storage media

E

effective group
If a user changes their default or primary group with the `newgrp` command, the new current group is the effective group (see also *group* and *primary group*).

efficiency
The ratio of the amount of a resource that is used in practice vs. the amount of the resource that could theoretically be used under ideal conditions. In SPP systems, the efficiency of the processors for a particular parallel programming application can be measured as the average percentage of time each processor assigned to the application is performing useful work on the application.

enabled
A condition or bit is said to be enabled when it is true or set to a one.

end user
The person or program that requests a service.

environment variable
A shell variable that contains information about your environment.

erasable programmable read-only memory (EPROM)
A read-only memory module that can be programmed repeatedly by first erasing the previous contents of the memory module. Reprogramming the memory modules usually involves removing them and using special hardware to burn a new pattern into them.

error code
The status returned by a function call.

Ethernet
A popular local area network (LAN) technology developed by Xerox.

exception
A hardware-detected event that disrupts the running of a program, process, or system. See also *fault*.

execution stream
A series of instructions executed by a CPU.

expression
A combination of constants and symbolic names joined by operators.

F

fault
A type of *interruption* caused by an instruction which requests a legitimate action which cannot be carried out immediately due to a system problem.

FCode
A variant of the Forth programming language defined in the IEEE P1275 Boot Firmware standard. This language is used in the boot firmware for Convex SPP systems.

FCode driver
A *device driver*, written in *FCode*, intended for use in IEEE P1275 Boot Firmware and its client programs.

file system
The organization of files on storage devices. The term "file system" can refer either to the entire file system tree or to a subsection of that file system contained on an individual disk, which can be mounted or unmounted from the tree.

firmware
A computer program that controls the computer system from the time it is turned on until the time the operating system assumes control of the computer. Firmware typically resides in read-only memory.

flag
A status bit that is generally used to indicate the results of an operation. The results are in the form true or false.

FLOPS
Floating-point operations per second. A standard measure of computer processing power in the scientific community.

fragment
A part of a block. The end of a file that is not a whole block is typically stored as a fragment. The size of a fragment can be specified; the use of a small fragment size adversely affects performance but leaves less wasted space.

front end
A client application for presenting, inputting, and displaying data. It works with a back end or engine.

front-end network
The interconnection of workstations, PCs, facsimiles, terminals, and printers.

function
A procedure that returns a value. Any procedure in C or a procedure defined as a `FUNCTION` in FORTRAN.

G

gallium arsenide (GaAs)
An alloy of gallium and arsenic that is used as the base material for integrated circuits, or chips. It is several times faster than silicon, currently the most common base material for making chips.

gigabyte
One billion bytes.

globally shared memory (GSM)
A memory architecture in which memory is physically distributed among the processors of a computer system, but all memory can be accessed by all processors in the system. This architecture can also support virtual memory. This type of memory is sometimes referred to as shared virtual memory or global virtual memory.

global optimization
A restructuring of program statements that is not confined to a single basic block. Global optimization, unlike interprocedural optimization, is confined to a single procedure. Global optimization is done by all Convex compilers at optimization level `-O1` and above.

global memory
Memory that is accessible from several processors. See also *distributed memory, globally shared memory (GSM).*

granularity
A measure of the relative size of objects where performance is a function of size. In higher-level language programs, possible sizes

are routine, loop, block, statement, and expression. In I/O, granularity is often measured in terms of block size.

group
Users on an SPP-UX system can be grouped. If a group has access to a file, then any user who is defined as a member of that group will have access to the file. Users can be members of more than one group.

group_ID (GID)
A unique number associated with each group that identifies the group to SPP-UX. These group_ID numbers are defined in the /etc/group file.

group ownership
The secondary ownership associated with each file, associating the file with a group.

H

h
Abbreviation for halfword.

hard error
An uncorrectable data error.

hardware address
The device-dependent physical address of a node attached to a communication line.

HFS file system
A file system in which the files are arranged on a disk within hierarchical directories.

hierarchical model
An organizational model useful for object description that divides a system into levels of abstraction.

High-Performance Parallel Interface (HIPPI)
Currently the fastest industry standard for connecting high-performance computers. HIPPI hardware consists of a HIPPI channel control unit (CCU) that supports dual simplex connections (one input, one output) to provide a data rate of 800 megabits per second over distances of up to 25 meters.

host
A computer system into which communications hardware and software have been installed, and which is accessed as a separate entity by other hosts.

hypernode
In Convex SPP systems, a set of processors and physical memory organized as a symmetric multiprocessor (SMP) running a single image of the operating system microkernel. An SPP system consists of one or more nodes, with a high speed CTI connecting the nodes.

I

IEEE P1275 Boot Firmware standard
A software architecture for firmware that provides for *autoconfiguration* of a system and allows for customers to customize boot programs and *device drivers* without altering the firmware.

inode
An SPP-UX data structure containing information about a file, such as ownership, permissions, and the file location on disk. An inode exists for every file accessible to the operating system.

Institute for Electrical and Electronic Engineers (IEEE)
An international professional organization and a member of ANSI and ISO. IEEE created Project 802, the committee that developed a set of widely-used LAN standards known as the 802 standard.

instruction
One of the basic operations performed by a CPU.

instruction cache (Icache)
Accelerates the decoding of instructions to permit the simultaneous decoding on one instruction with the execution of another instruction.

instruction mnemonic
A symbolic name for a machine instruction.

interface
A logical path between any two modules or systems. (See *network interface.*)

interface script
A shell script, located in the /usr/spool/lp/interface directory, that is the final stage of the `lpss` print subsystem. Each printer that is configured in the `lpss` has an interface script that, under the control of the line printer scheduler, sends a print job to the printer.

interleaved memory
Memory that is divided into multiple banks to permit concurrent memory accesses. The number of separate memory banks is referred to as the memory stride. Programs can optimize memory accesses by using stride intervals so that each of the banks can be refreshed between memory accesses.

interrupt
An occurrence that changes the normal flow of instruction execution. An interruption originates from hardware, such as an I/O device. See also *maskable interrupt*.

interval timer
A device that generates an interrupt based on the passage of time.

I/O node
A node that contains a chassis filled with peripheral devices (e.g., disk drives) in place of a CPU chassis.

J

job scheduler
The operating system program that schedules and manages the execution of all processes.

K

kernel
The core of the operating system where basic system facilities, such as file access and memory management functions, are performed.

L

latency
The time delay between the issuing of an instruction and the completion of the operation. A common benchmark used for comparing SPP systems is the latency of coherent memory access instructions involving non-CPU private memory; Convex SPP system latency values are among the best (lowest) in the industry. This particular latency measurement is believed to be a good indication of the *scalability* of an SPP system; low latency equates to low system overhead as system size increases.

line printer scheduler
The line printer scheduler is the heart of the `lpss` print subsystem. It is the central program that must be running to ensure coordination of requests from users to printers.

linker
A software tool that combines separate software modules into a single module.

logical cache
A cache that is accessed with logical addresses for fast retrieval of data. It resides in the CPU.

logical memory
The memory space as seen by the program. Also called virtual memory. See also *page*.

log in
The process used to gain access to the computer by supplying a user name and (if required) a password.

long file names
File names using more than 14 (but not exceeding 255) characters. Long file names are incompatible with file systems configured for short file names.

local printer
A printer, configured into your `lpss` print subsystem, that is physically connected to your computer. Local printers are not supported on Convex Exemplar systems. See *remote printer*.

logical printer
Each printer that is defined in your `lpss` print subsystem is given a name that users will use to refer to it. You can create more than one definition (printer name) for any given printer. The logical printer name refers, not to the printer itself, but to one of the `lpss` definitions used to access the printer.

`lpss`
The SPP-UX software subsystem responsible for controlling output to the printers on your system. Its primary responsibility is to prevent intermixed listings. It can also prioritize print jobs and start and stop output to printers.

M

machine exception
A fatal error in the system that cannot be handled by the operating system. See also *exception*.

maskable interrupt
An interrupt to which the operating system may choose not to respond.

massively parallel processing (MPP)
A computer architecture in which tens to thousands of processors are combined to work simultaneously on solving complex problems.

MAUI
The Convex SPP system MPP Agent for Utilities and I/O gate array.

megabyte (Mbyte)
1 million bytes.

memory management
The hardware and software features that control memory page mapping and memory protection.

message
Data copied from one thread to another (or the same) thread. The copy is initiated by the sending thread, which specifies the receiving thread. The sending and receiving threads need not share a common address space.

Message passing
A type of programming in which program modules (often running on different processors or different hosts) communicate with each other by means of system library calls that package, transmit, and receive data.

A hardware architecture in which memory accesses involving physically distributed banks of memory are implemented by messages passed between the processors that own the respective banks of memory.

microcode
A program or set of assembly language instructions that reside within a storage device. Microcode is also used as a synonym for firmware.

microkernel
An operating system architecture in which only a small portion of the operating system kernel (the microkernel) is resident in each processor; the remainder of the operating system is provided by the operating system server. In Convex SPP systems, there is a single image of the microkernel resident in each *node* of the system.

MOD
Meaningful Optical Display. A term used to describe the Convex SPP system cabinet flashing lights.

model script
When a printer is added to the `lpss` print subsystem, an interface script must be created that can set up the communication to the printer and send data to it. Hewlett-Packard supplies models for

these interface scripts in the /usr/spool/lp/model directory. These models are used by the `lpadmin` command to build the interface script for a printer as it is being defined.

mount
To add an auxiliary (removable) file system to an active existing file system.

mount directory, mount point
A directory in an existing file system that serves as the root directory (the mount point) of a mounted file system.

MTV
Memory Tag Vortex. The Convex SPP system memory board.

MU
The Convex SPP system utility board.

Multiplexer (also known as a mux)
A device that takes multiple data streams and combines them into a single stream for transmission over a circuit. Another mux demultiplexes the data stream on the receiving end.

multiprocessing
The creation and scheduling of processes on any subset of CPUs in a system configuration.

multiuser mode
An SPP-UX mode of operation that allows multiple users to access the system simultaneously. This is the normal mode of operation for SPP-UX systems. See also *single-user mode*.

N

named pipe
A pipe that exists permanently in the file system with directory entries and path names. Because you can access these pipes by name, you can use them for a variety of applications that you cannot accomplish with ordinary pipes. Typically, named pipes are used to allow a number of processes to communicate with a daemon.

network
A system of interconnected computers that enables machines and their users to exchange information and share resources. Convex SPP systems provide support for *FDDI* networks.

network architecture
The logical organization of a networking system.

network-based printer
A printer that is directly connected to a local area network.

NFS client
A machine that mounts (via the network) a file system located on a remote NFS server.

NFS file system
A file system accessible over a network via the NFS Services product.

NFS server
A computer with local file systems that are being accessed via the network by remote computers, or NFS clients.

noncoherent memory reference
A memory reference which 1) does not cause a cache move in, or 2) causes a cache move in, but fails to obey cache coherency rules.

nonvolatile configuration information
OBP information which persists across machine reboots.

O

object management computing (OMC)
A way to represent information pictorially so a user does not need to understand the mechanics of the computer/networking operating system. OMC assigns icons to each network component, including programs, documents, and drawings, letting users manipulate images to access resources.

Object Management Group (OMG)
An industry consortium created to develop an easy-to-use graphic interface for distributed networks. The group adopted New Wave Object Management Facility from Hewlett-Packard as its core technology.

offset
An integer value that is added to a base address to calculate a memory address. Offsets in Convex SPP systems are 32-bit values, and must keep address values within a single 4-GB memory *space*.

Open Software Foundation (OSF)
A consortium of industry leaders working to standardize the UNIX operating system.

open systems
Systems that conform to any nonproprietary, publicly-available standards. In recent years the term has come to mean only those systems that use the international standards for network architecture, as specified by the OSI model.

Open Systems Interconnection (OSI)
The ISO definition of a system that provides reliable, data-transparent, host-independent services.

operand
A register or memory location referenced by an instruction. It is the code or sequence of bits in an instruction that determines the data to be operated on.

operating system
The program that manages the resources of a computer system. The Convex SPP system uses the SPP-UX operating system. SPP-UX is compatible with the Hewlett-Packard HP-UX operating system.

optimization
The refining of application software programs to minimize processing time. Optimization takes maximum advantage of a computer hardware features and minimizes input/output traffic and idle processor time.

optimization level
The degree to which source code is optimized by the compiler. The Convex FORTRAN and C compilers have five levels of optimization, level `-no`, `-O0`, `-O1`, `-O2`, and `-O3`.

ownership
Each file on the system has an owner. The owner controls access to the file by setting its access permissions. The owner is typically (but not always) the user who created the file.

P

PA-RISC
The Hewlett-Packard Precision Architecture reduced instruction set processor chip. This is the processor chip used in Convex SPP systems.

packet
A group of related items. A packet may refer to the arguments of a subroutine or to a group of bytes that is transmitted over a network.

page
A page is the unit of logical memory controlled by the memory management algorithms. A page is 4 K (4,096) contiguous bytes. See also *logical memory*.

pagefault
A pagefault occurs when a process requests data that is not currently in main memory. The machine first saves off the state of all controllers onto a context stack in main memory. The operating system creates a free page of physical memory to bring the data in from the disk. The appropriate Page Table Entries (PTEs) are set up so that the proper logical-to- physical translation occurs. The machine reads back from memory the state of the machine from the context stack, and restores the processor to the same state that it was in when it determined that the data it needed was nonresident. The CPU can then continue with normal operation of the process.

page frame
A page frame is the unit of physical (main) memory in which pages are placed. Referenced and modified bits associated with each page frame aid in memory management.

parallelization
The act of creating code that enables sections of code to run simultaneously on multiple CPUs. At optimization level `-O3`, the Convex FORTRAN compiler automatically parallelizes your program and recognizes compiler directives with which you can specify parallelization.

parent node
A node in any hierarchical system that has one or more child (immediately lower) nodes. A child node is said to *descend* from its parent node.

parity
Even parity is an even number of one bits, and is used throughout the Convex I/O system.

parity RAM
This register contains an odd parity bit for each byte of data in buffer memory 0 and 1.

path
An environment variable that you set in your shell configuration file that allows you to access commands in various directories without having to specify a complete path name.

PB0
Power board 0. The lower of the 2 front power boards in Convex SPP systems.

PB1
Power board 1. The upper of the 2 front power boards in Convex SPP systems.

PB2
Power board 2. The rear power board in Convex SPP systems.

PCC
Power concentrator card. The power board that is located in the power bay in Convex SPP systems.

PDIR
Physical page directory. PA-RISC CPUs that implement hardware TLB miss handlers, may fetch TLB entries from a PDIR entry in the event of a TLB miss. The PDIR serves as a cache of virtual-to-physical page translations, and is maintained by the operating system.

peripheral chassis
The chassis that is situated in the cabinet between the two nodes. Also referred to as the *I/O chassis*.

physical address
A unique identifier that selects a particular device from the set of all devices connected to a particular bus.

physical address space
The set of possible addresses for a particular bus.

PID
An SPP-UX process ID. See *process identification*.

pipe
A pair of file descriptors that provide the mechanism for a one-way flow of data.

pipeline
An overlapping operating cycle function that is used to increase the speed of computers. Pipelining provides a means by which multiple operations occur concurrently by beginning one instruction sequence before another has completed.

porting
Converting software from one type of machine to another.

POSIX standards
A family of open systems standards developed by the IEEE Technical Committee on Operating Systems.

Power bay
The area at the bottom of the cabinet that has input power components and the 48VDC power supplies. It is not called a chassis because it is not a self-contained assembly; for example, the power supplies are mounted directly to the bottom of the cabinet.

PRAM
Parallel random-access machine. An abstraction of a parallel computer that has the properties of (a) *p* serial processors, (b) a single, shared global memory for all of the *p* processors, (c) all *p* processors can read from and write to the global memory in parallel, and (d) the processors can perform arithmetic and logical operations in parallel with each other. The major difference between a PRAM model and a real machine is that the latter has different memory access times depending on the access patterns made by the processors' execution.

primary group
A user can be a member of multiple groups, but only one of those groups is considered to be the user's primary or default group. In addition to being listed as a member of groups in the /etc/group file, an entry exists in the /etc/passwd file that indicates the user's primary group. When users first log in to the system, they are affiliated with their primary group.

print destination
A print destination is a generic term used to describe a printer or printer class. Users can specify a print destination when they print something by using the `-d` option to the `lp` command. See *printer class.*

printer class
A defined group of printers. A printer class can be used as a print destination instead of a printer name. The first available printer in the printer class will print the next job queued to that printer class.

printer name
When a printer is configured into the `lpss` print subsystem, it is given a name that users can use to specify that printer as a print destination for their printout.

print queues
Also known as request directories, print queues are directories used by the `lpss` print subsystem to hold print jobs for each print destination until they can be printed.

print request
A term used to refer to a specific print job in the `lpss` print subsystem.

print request identification number
The number the `lpss` print subsystem uses to identify your print request. This identification number consists of the name of the printer or printer class followed by a sequence number.

priority
An ordering of events. May be applied to protection levels as well as to I/O interrupt levels.

A value associated with each printer and print request. Priorities are used to control which print requests can print on a given printer. Priorities can be adjusted and must have a range from 0 to 7.

privileged instruction
An instruction used by the operating system or privileged systems programs. It must execute at privilege level, or an exception occurs.

process
The running version of a program. The current state of the process is stored in the process image.
A process is the fundamental unit of a program that is managed by the job scheduler.
A collection of one or more execution streams within a single logical address space; an executable program. A process is made up of one or more threads.

process identification (PID)
A unique number assigned to a process when initiated. It may be found with the operating system command referenced by various commands, such as *kill*.

process memory
The portion of system memory that is used by an executing process.

process settings
The collection of settings that control various aspects of your process, such as its floating point mode. You can modify the process settings to match the needs of your application.

processor set
In Convex SPP systems, a collection of zero or more processors from a single node. Each microkernel task is assigned to a

particular processor set, and the threads of the task may only run on processors belonging to that processor set.

process stack
A dynamic, LIFO storage list of frames that the operating system uses to track the flow of execution in a process. Each frame stores the registers and local variables from the previous execution context, the addresses used to manage the current stack frame, and a pointer to the previous stack frame.

process working directory
The directory from which the process is run. It is also the base directory for the relative path names of all files accessed by your process.

prompt
A character or character string sent from a computer system to a terminal to indicate to the user that the system is ready to accept input.

protection
A mechanism provided by hardware and software that ensures that one user is protected from another user or to ensure that a user does not perform an unsafe computation.

Q

quadrant
One quarter of a space. In Convex SPP systems, each quadrant of memory is 1 GB. in size.

queue
A data structure in which entries are made at one end and deletions at the other. Often referred to as first-in, first-out (FIFO).

R

raw socket
An IPC facility that provides users with access to the underlying network protocols.

read
A memory operation in which the contents of a memory location are copied and passed to another part of the system.

reduced instruction set computer (RISC)
This is an architectural concept that applies to the definition of the instruction set of a processor. A RISC instruction set is an orthogonal instruction set that is easy to decode in hardware and for which a compiler can generate highly optimized code.

register
A hardware entity that contains an address, operand, or instruction status information.

remote print requests
A print request issued via the `lpss` print subsystem on your system to be printed on a printer that is attached to a remote computer.

remote printer
A printer that is defined in your `lpss` print subsystem but is physically connected to another computer (and accessed over a network).

remote spooling
The process used to allow printing to printers that are defined as part of your `lpss` print subsystem but physically connected to another computer.

Remote Procedure Call (RPC)
A facility provided by the Convex NFS product that allows a client process to have another process execute a procedure call, as if the caller had executed the procedure call in its own address space.

remote spooling daemon
A background program that runs on a remote computer. The remote spooling daemon receives print requests via a network and submits the print requests to its local `lpss` print subsystem on the network user's behalf.

root directory
The highest level directory in a file system. In any mountable file system (any file system other than the root file system) the root directory is the mount directory.
The / directory, also known as the root directory, is the highest level in the SPP-UX file system overall; the / file system cannot be unmounted because it contains the running operating system.

root file system
In SPP-UX, the file system associated with the root directory.

root node
A node in a hierarchy that has no *parent node*.

router
A hardware device that directs packets across networks. A router attempts to choose the most effective route to a packet destination if multiple routes exist.

A hardware device used in Convex SPP systems to pass data across *nodes*.

RPC
A facility provided by the Convex NFS product that allows a client process to have another process execute a procedure call, as if the caller had executed the procedure call in its own address space.

rsh
The SPP-UX remote shell command. This command enables you to execute a command on one computer from the command line of another computer.

run-level
An SPP-UX mode of operation. Modes of operation are defined in the file /etc/inittab. The /etc/inittab file defines which terminals and processes are active at each run-level.

S

Scalable parallel processing (SPP)
A computer architecture that permits an application to run with a relatively small number of processors or in a processor array containing hundreds to thousands of processors. A key design goal for SPP systems is to enable performance to increase linearly with respect to its number of processors.

SBus
The Sun SPARC Bus. This is the bus standard used in SPARC-compliant computers such as Sun 4. SPARC stands for Scalable Processor Architecture. Convex SPP systems use SBus for I/O connections to disk drives and external networks. SBus is both an electrical standard and a mechanical standard which includes defining the board size, connector type, and other physical details.

SCI
Scalable Coherent Interface. This is defined by IEEE standard 1596-1992. The interface is physically defined as a pair of 18-bit, differential ECL, unidirectional links which are clocked at 250 Mhz. Each link provides 16 bits of data with two control signals. Data is sampled on both the rising and falling edges of the clock. This interface is used in Convex SPP systems.

SCSI
Small computer system interface (SCSI).

server
A process that fulfills a request issued by a client process, and transmits a response back to the client. Also called a *service provider.*

server set
In a Convex SPP system, the set of processors on a node that will run server tasks for the node. Server tasks and operating system microkernel tasks run only on processors belonging to the server set. Every node has a server set, and it must contain at least one processor at all times.

server stub
Used in implementing remote procedure call (RPC) facilities, the server stub waits for an RPC request from a client, executes a local procedure call on behalf of the client, and returns results to the client. Server stubs are used to abstract the details of passing messages over the network.

s-field
A field in some PA-RISC instruction formats that specify the memory *space* the instruction operates on.

sharing list
A structure used by SCI for maintaining cache coherence state information. The sharing list is structured as a linked list.

short file names
Files with names consisting of 14 or fewer characters. Short file names are compatible with file systems configured for long file names.

signal
The standard SPP-UX entity used to control a program. Refer to the signal(3c) man page for a list of signal names.

single-user mode
A special SPP-UX mode of operation that restricts user input to the system console. It is usually used by the system administrator to prevent others from accessing the system during special system administration activities when it is not advisable to have other system activity (for example, when updating the operating system to a new revision). See also *multiuser mode.*

SMP
A symmetric multiprocessor.

soft error
A correctable single-bit memory error. May further be defined as transient (nonreproducible) or stuck (reproducible). Recorded in the softlog.

software device driver
A program that controls the operation of attached I/O peripheral devices.

space
A contiguous range of virtual addresses within the system wide virtual address space. Spaces are 4 GB. in size in MP-1.

space id
A 16, 24, or 32 bit value which occupies the upper portion of a virtual address and specifies the virtual space portion of the virtual address.

spawn
To create new tasks.

SPOK
System Power OK. The name of the cable in Convex SPP systems that connects the PPC to each of the nodes for the purpose of passing them the status of the power supplies in the power bay.

SPP
Scalable parallel processing. A computer architecture that permits an application to run with a relatively small number of processors or in a processor array containing hundreds to thousands of processors. A key design goal for SPP systems is to enable performance to increase linearly with respect to its number of processors.

SST
System Scan Test. A diagnostic procedure that uses the scan (diagnostic) bus.

stack
A data structure in which the last item entered is the first to be removed. Also referred to as last-in, first-out (LIFO). See also *process stack.*

status model script
The status model script, `/usr/spool/lp/smodel/rsmodel`, is used to return the status of the remote printer and print requests for the remote spooling daemon.

stream socket
IPC facility that provides for the bidirectional, reliable, sequenced, and unduplicated flow of data without record boundaries.

subcomplex
In Convex SPP systems, a logical entity that provides control over the allocation of processors and physical memory to different applications and users.

subcomplex manager
A utility that allows Convex SPP system administrators to maintain a database of subcomplex definitions, load subcomplex definitions onto the system, and modify subcomplexes currently running on the system.

subcomplex server
In Convex SPP systems, a subsystem within the process manager that provides the system call interface for creating and manipulating subcomplex and server sets on the system.

subroutine
A frequently used software module that is called from various places in a program.

superscalar
A class of *RISC* processors that allow multiple instructions to be issued on each clock period.

superuser
The SPP-UX term for a special privilege level that allows the system manager to perform system maintenance functions that ordinary users cannot perform.

swap space
Space on a disk used for storing the process image temporarily.

symbolic debugger
A debugger that can map source code to executable machine instructions. CXdb is a symbolic debugger with the added functionality and capability to debug optimized code symbolically.

symbolic link
A connection that associates a node with a tape device.
A feature present in most SPP-UX systems that allows a file to exist as a link to a file in another location.

system default printer
When a user issues a print request, the user can specify a print destination. A system default printer can be defined so that, if a print destination is not specified, the `lpss` print subsystem will use the system default printer.

system manager
The person responsible for the management and operation of a computer system. Also called the system administrator.

system console
The terminal or workstation that serves as a communication device between the system manager and the computer system. On Convex SPP systems, the test station serves as the system console.

system subcomplex
In a Convex SPP system, a subcomplex that is automatically created at boot time by the operating system to run system processes, including `init` and processes spawned by `init`. The Complex Manager will not allow users to destroy this subcomplex, nor to remove the last processor from this subcomplex.

T

task
A collection of system resources. A task itself does not execute, but at least one *thread* executes within each task.

task context
Relevant data for each task running on a host.

task ID (tid)
An integer that uniquely identifies a task across a virtual machine.

terabyte (Tbyte)
One trillion bytes.

TLB
Translation Lookaside Buffer. A hardware structure in each CPU in a Convex SPP system that contains the information necessary to translate a virtual memory reference to a physical page and to validate memory accesses.

TOC
Time of Century. This value must be consistent across an entire computer system. Convex SPP systems achieve this consistency by using cables that connect all of the nodes together in a bus fashion.

test station
The term that is used for the workstation that is used to diagnose problems and install system software on a Convex SPP system. The test station takes the place of the Service Processor Unit (SPU) on other Convex systems, and the system console on other computer systems.

thread
An independent execution stream that is fetched and executed by a CPU. One or more threads, each of which can execute on a different CPU, make up each process. Memory, files, signals, and other process attributes are generally shared among threads in a given process, enabling the threads to cooperate in solving the common problem. Threads are created and terminated by instructions that can be automatically generated by Convex compilers, inserted by adding compiler directives to source code, or coded explicitly in assembly-language programs.

thread data
Initialized data that is not shared among all threads comprising a process, but is unique to that one thread.

thread-private or thread-specific
Data that is accessible by a single thread only (not shared among the threads constituting a process). Thread-specific data allows the same virtual address to refer to different physical memory locations.

transaction
A data or control element, signal, event, or change of state that causes, triggers, or initiates an action or a sequence of actions.

trap
A type of *interruption* caused when either the function requested by the current instruction cannot or should not be carried out, or system intervention is desired by the user before or after the current instruction is executed. Typically, this condition is a result of unexpected arithmetic results. See also *exception*.

U

UID
A unique number that SPP-UX uses to identify a particular user. The user_ID number zero ("0") is used to identify the superuser. User_ID numbers between 1 and 99 are used by SPP-UX subsystems. User_ID numbers above 99 are used for normal users.

unit address
The component of a *device node name* that indicates the *device node* position within the address space defined for its *parent node.*

unmount
To remove an auxiliary file system from the existing file system.

unsigned
A value that is always positive.

use-def chain
A data structure created and used by optimizing compilers to track every site within a program where a particular variable or array subsection could possibly be defined, starting from the point where the item is used. See also *use-def chain.*

user account
The environment created on the system to allow the user access. Creating a user account involves updating the system to recognize the user's login name and password. You also need to give the user access to files, system resources, and applications.

user data
Data that originates from a service user.

user_ID
Also known as UID, is a unique number that SPP-UX uses to identify a particular user. The user_ID number zero ("0") is used to identify the superuser. User_ID numbers between 1 and 99 are used by SPP-UX subsystems. User_ID numbers above 99 are used for normal users.

user interface
The portion of a computer program that processes input entered by a human and provides output for human users.

utility
A software tool designed to perform a frequently used support function.

V

virtual machine
A collection of computers appearing as one heterogenous computer.

virtual memory
The memory space as seen by the program.

virtual page number (VPN).
The page number portion of a virtual address.

W

word
A contiguous group of bytes starting on an addressable boundary. In Convex computer systems, a word is four bytes (32 bits) in length.

working set
The portion of a user program currently in physical memory. Typically, the working set is much smaller than the user program

workstation
A stand-alone computer that has its own processor, memory, and possibly a disk drive.

X

X Window System
A graphic windowing system designed for networked computer systems with bitmap displays .

Index

Symbols

A

B

C

D

M

N

O

P

R

S

T

U

ORDER NUMBER
DSW-853

DOCUMENT NUMBER
710-029330-002