System Administration Guide, Volume I

2550 Garcia Avenue
Mountain View, CA 94043
U.S.A.

SunSoft
A Sun Microsystems, Inc. Business

Contents

Figures

Tables

Finding System and Network Administration Information

System Administration Guide, Volume I is part of a multibook set describing Solaris™ 2.5 system and network administration, which is shown in Figure P-1. The multibook set is provided with the *Solaris 2.5 System Administrator AnswerBook.*

Table P-1 lists what information is covered in each of the Solaris system and network administration books. Use this table as a high-level guide to find the right book for the information you need.

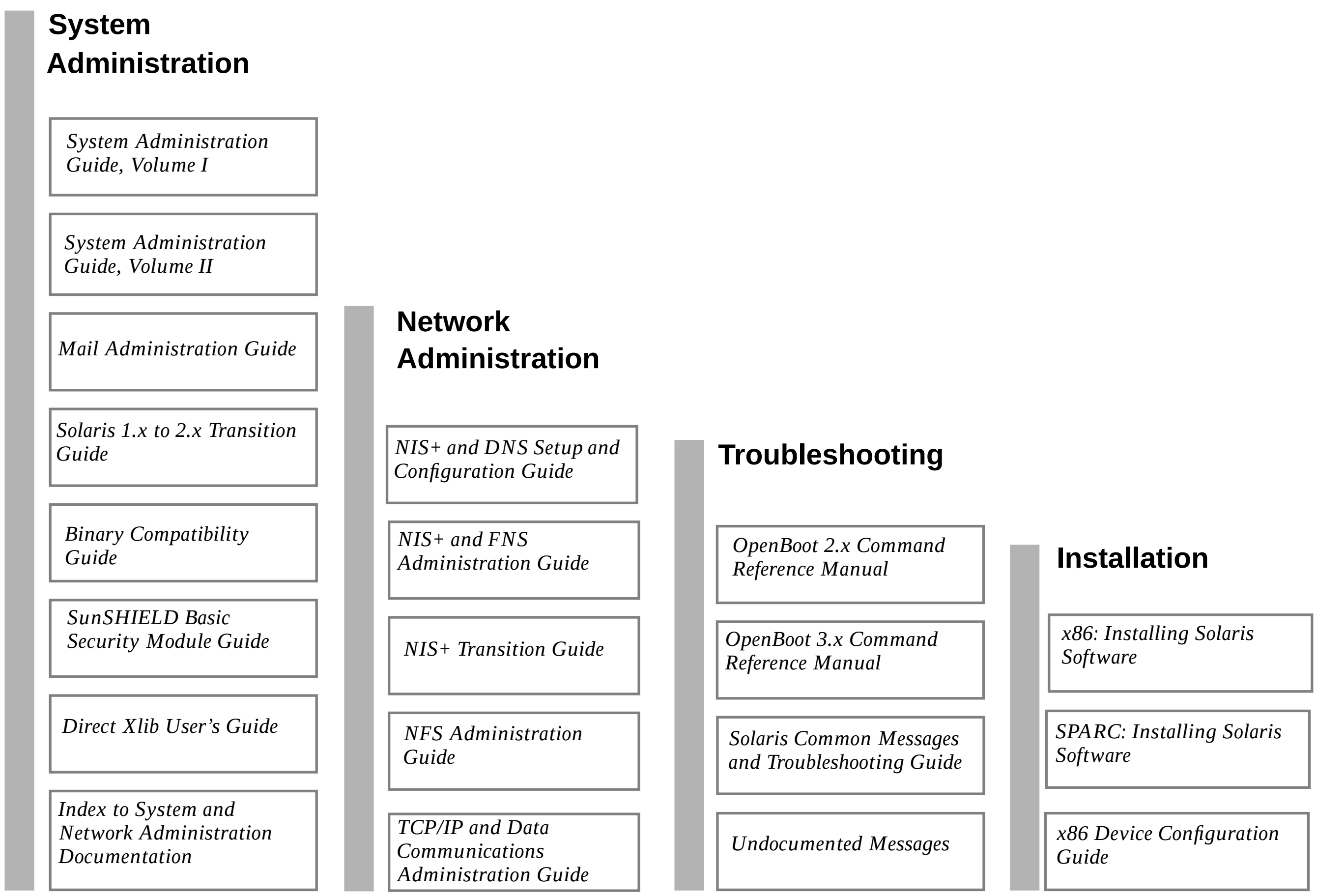


Figure P-1 System and Network Administration Books

Table P-1 Where to Find System and Network Administration Information

If You Need Information On ...	Then Go To ...	
• Backing up and restoring data • Shutting down and booting a system • Managing - Disks - File systems - Removable Media (CDs, diskettes, PCMCIA) - Software (packages, patches, AnswerBook™) - Server and client support - User accounts and groups • Working with remote files	*System Administration Guide, Volume I*	**System Administration**
• Managing - Printing Services - System resources (accounting, crash dumps, disk use and quotas, crontabs, system information) - System performance - Terminals and modems • System security (ACLs, file permissions, ASET)	*System Administration Guide, Volume II*	
• Managing mail	*Mail Administration Guide*	
• Transitioning SunOS™ 4.x systems to Solaris 2.5	*Solaris 1.x to 2.x Transition Guide*	
• Setting up binary compatibility	*Binary Compatibility Guide*	
• Setting up auditing	*SunSHIELD Basic Security Module Guide*	
• Managing runtime libraries	*Direct Xlib User's Guide*	

Table P-1 Where to Find System and Network Administration Information *(Continued)*

If You Need Information On ...	Then Go To ...	
• Managing NIS+, DNS, or FNS	*NIS+ and DNS Setup and Configuration Guide* *NIS+ and FNS Administration Guide*	**Network Administration**
• Transitioning from NIS to NIS+	*NIS+ Transition Guide*	
• Managing NFS	*NFS Administration Guide*	
• Configuring TCP/IP, PPP, or UUCP	*TCP/IP and Data Communications Administration Guide*	
• Testing hardware and software from the PROM	*OpenBoot 2.x Command Reference Manual* *OpenBoot 3.x Command Reference Manual*	**Troubleshooting**
• Error messages and troubleshooting	*Solaris Common Messages and Troubleshooting Guide* *Undocumented Messages*	
• Installing Solaris	*SPARC: Installing Solaris Software* *x86: Installing Solaris Software* *x86 Device Configuration Guide*	**Installing Solaris Software**

About This Book

System Administration Guide, Volume I is part of a two-volume set that covers a significant part of the Solaris™ system administration information. It includes both SPARC™ and x86 information and describes how to use the Solstice™ AdminSuite tools to perform some of the system administration tasks.

This book assumes that you have already installed the SunOS 5.5™ operating system and Solstice AdminSuite, and you have set up any networking software that you plan to use. The SunOS 5.x operating system is part of the Solaris 2.x product family, which also includes many utilities and OpenWindows™ Version 3.x. The SunOS 5.x operating system is compliant with AT&T's System V, Release 4 operating system.

System Administration Guide, Volume I and *System Administration Guide, Volume II* have replaced the following books previously released with the Solaris operating environment:

- *Security, Performance, and Accounting Administration*
- *User Accounts, Printers, and Mail Administration*
- *Administration Supplement for Solaris Platforms*
- *Common Administration Tasks*
- *File System Administration*
- *Administration Application Reference Manual*
- *Peripherals Administration*

Mail Administration is now covered in the *Mail Administration Guide*.

Who Should Use This Book

This book is intended for anyone responsible for administering one or more systems running the Solaris 2.x release. To use this book, you should have 1-2 years of UNIX® system administration experience and preferably a Computer Science B.S. degree or equivalent knowledge.

How This Book Is Organized

This book is split into parts that each cover a major system administration topic. Each part contains chapters that provide both overview and task information.

Most of the overview information about a topic is usually described in the beginning chapters of each part, and the other chapters provide step-by-step instructions on system administration tasks that you need to perform. Each set of steps is usually followed by a way to verify that the task was successfully performed and an example of how to perform the task.

Using AnswerBook to Read This Book

If you are reading this book from within the AnswerBook™ online document viewer, you can double-click on any cross reference, represented by text in a box, to quickly access the referenced information. To return to the previous display, click on the Go Back button.

SPARC and x86 Information

This book provides system administration information for both SPARC and x86 systems. Unless otherwise noted, information throughout this book applies to both types of systems. Table P-2 summarizes the differences between the SPARC and x86 system administration tasks.

Table P-2 SPARC and x86 System Administration Differences

Category	SPARC	x86
System operation before kernel is loaded	A programmable read-only memory (PROM) chip with a monitor program runs diagnostics and displays device information. It is also used to program default boot parameters and test the devices connected to the system.	The basic input/output system (BIOS) runs diagnostics and displays device information. A Solaris boot diskette with a program called Multiple Device Boot (MDB) is used to boot from non-default boot partitions, the network, or CD-ROM.
Booting the system	Commands and options at the PROM level.	Commands and options at the MDB, primary, and secondary boot subsystems level.

Boot programs	`bootblk` – the primary boot program, loads `ufsboot`
	`ufsboot` – the secondary boot program loads the kernel
		`mboot` – the master boot record, loads `pboot`
		`pboot` – the Solaris partition boot program, loads `bootblk`
		`bootblk` – the primary boot program, load `ufsboot`
		`ufsboot` – the secondary boot program, executes the `/etc/bootrc` script and loads the kernel

Reboot commands	The `shutdown`, `init 6`, or `reboot` commands can be used without additional operation intervention.	The `shutdown`, `init 6`, or `reboot` commands are used but requires operator intervention at the `type any key to continue` prompt.
Disk Controllers	SCSI, IPI, and Xylogics	SCSI and IDE
Disk slices and partitions	Maximum of eight slices, numbered 0-7.	Maximum of four `fdisk` partitions. The Solaris `fdisk` partition may contain up to ten slices, numbered 0-9, but only 0-7 can be used to store user data. You can only have one Solaris `fdisk` partition per disk.
Diskette drives	Desktop systems usually contain one 3.5-inch diskette drive.	Systems may contain two diskette drives: a 3.5-inch and a 5.25 inch drive.

What Typographic Changes Mean

The following table describes the typographic changes used in this book.

Table P-3 Typographic Conventions

Typeface or Symbol	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% You have mail.`
`AaBbCc123`	What you type, contrasted with on-screen computer output	`machine_name%` **`su`** `Password:`
AaBbCc123	Command-line placeholder: replace with a real name or value	To delete a file, type `rm` *filename*.
AaBbCc123	Book titles, new words or terms, or words to be emphasized	Read Chapter 6 in *User's Guide*. These are called *class* options. You *must* be root to do this.

Shell Prompts in Command Examples

The following table shows the default system prompt and superuser (root) prompt for the Bourne shell and Korn shell.

Table P-4 Shell Prompts

Shell	Prompt
Bourne shell and Korn shell prompt	`$`
Bourne shell and Korn shell superuser prompt	#

General Conventions

Be aware of the following conventions used in this book.

- Some code examples have a backslash (\) at the end of a line to specify line continuation, such as the following code example:

```
# pmadm -a -p tcp -s lpd -i root -m ‘nlsadmin -o \
/var/spool/lp/fifos/listenBSD -A \
'\x00020203819418 0e0000000000000000'‘ -v ‘nlsadmin -V‘
```

  If the line is an example of what to type, ignore the backslashes (don’t type them) and press Return at the end of the line that does not end with a backslash. In the example above, you would ignore the two backslashes when typing the `pmadm` command and press Return after the third line.

- When following steps or using examples, be sure to type double-quotes ("), left single-quotes (`), and right single-quotes (') exactly as shown.
- The key referred to as Return is labeled Enter on some keyboards.
- It is assumed that the root path includes the `/sbin`, `/usr/sbin`, `/usr/bin`, and `/etc` directories, so the steps in this book show the commands in these directories without absolute path names. Steps that use commands in other, less common, directories show the absolute path in the example.
- The examples in this book are for a basic SunOS 5.x software installation without the Binary Compatibility Package installed and without `/usr/ucb` in the path.

Caution – If `/usr/ucb` is included in a search path, it should always be at the end. Commands like `ps` or `df` are duplicated in `/usr/ucb` with different formats and options from the SunOS 5.x commands.

Part 1 — Managing User Accounts and Groups

This part provides instructions for managing users and groups.

1

Overview of Managing User Accounts and Groups
Provides overview information about setting up user accounts and groups in a network environment.

2

Setting Up and Maintaining User Accounts and Groups
Provides step-by-step instructions for setting up user accounts and groups with User Account Manager and Group Manager.

Overview of Managing User Accounts and Groups 1

This chapter provides guidelines and planning information for managing user accounts and groups, and it provides overview information about setting up user accounts and groups in a network environment. This chapter also includes information about the files used to store user account and group information and about customizing the user's work environment.

This is a list of the overview information in this chapter.

For instructions about how to manage users accounts and groups, see Chapter 2, "Setting Up and Maintaining User Accounts and Groups."

What Are User Accounts and Groups

One of the basic system administration tasks is to set up a user account for each user at a site. A typical user account includes the information a user needs to log in and use a system (without having the system's root password). User account information consists of four main components:

- **User name –** A name that a user uses to log in to a system (also known as a login name).
- **Password –** A secret combination of characters that a user must enter with a user name to gain access to a system.
- **User's home directory –** A directory that is usually the user's current directory at login. It typically contains most of the user's files.
- **User initialization files –** Shell scripts that control how the user's working environment is set up when a user logs in to a system.

Also, when you set up a user account, you can add the user to predefined groups of users. A typical use of groups is to set up file and directory access only to users who are part of a group (using the group permissions on a file or directory).

For example, you might have a directory containing top secret files that only a few users should be able to access. You could set up a group called `topsecret` that included the users working on the top secret project, and you could set up the top secret files with read permission for the `topsecret` group. That way, only the users in the `topsecret` group would be able to read the files.

Guidelines for Managing User Accounts

The following sections describe some guidelines and planning information for creating user accounts.

Name Services

If you are managing user accounts for a large site, you may want to consider using a name service such as NIS or NIS+. A name service enables you to store user account information in a centralized manner instead of storing user account information in every system's `/etc` files. When using a name service for user accounts, users can move from system to system using the same user account without having site-wide user account information duplicated in every system's `/etc` files. Using a name service also promotes centralized and consistent user account information.

User (Login) Names

User names, also called login names, let users access their own systems and remote systems that have the appropriate access privileges. You must choose a user name for each user account you create. User names must:

- Be unique within your organization, which may span multiple domains
- Contain from two to eight letters and numerals (the first character must be a letter and at least one character must be a lowercase letter)
- Not contain an underscore or space

It is helpful to establish a standard way of forming user names, and the names should be easy for users to remember. A simple scheme when selecting a user name is to use the first name initial and first seven letters of the user's last name. For example, Ziggy Ignatz becomes `zignatz`. If that scheme results in duplicate names, you can use the first initial, middle initial, and the first six characters of the user's last name. For example, Ziggy Top Ignatz becomes `ztignatz`. If that still results in duplicate names, you can use the first initial, middle initial, first five characters of the user's last name, and the number 1, or 2, or 3, and so on, until you have a unique name.

Note – Each new user name must be distinct from any mail aliases known to the system or to an NIS or NIS+ domain. Otherwise, mail may be delivered to the alias rather than to the actual user.

User ID Numbers

Associated with each user name is a user identification (UID) number. The UID number identifies the user name to any system on which the user attempts to log in, and it is used by systems to identify the owners of files and directories. If you create user accounts for a single individual on a number of different systems, always use the same user name and user ID. In that way, the user can easily move files between systems without ownership problems.

UID numbers must be a whole number less than or equal to 60000, and they are required for both regular user accounts and special system accounts. Table 1-1 lists the UID numbers reserved for user accounts and system accounts.

Table 1-1 Reserved UID Numbers

User ID Numbers	Login Accounts	Reserved For ...
0 - 99	root, daemon, bin, sys, etc.	System accounts
100 - 60000	regular users	General purpose accounts
60001	nobody	Unauthenticated users
60002	noaccess	Compatibility with previous Solaris 2.x and SVR4 releases

Although UID numbers 0 through 99 are reserved, you can add a user with one of these numbers. However, do not use them for regular user accounts. By definition, root always has UID 0, `daemon` has UID 1, and pseudo-user `bin` has UID 2. In addition, you should give `uucp` logins and pseudo user logins, like `who`, `tty`, and `ttytype`, low UIDs so they fall at the beginning of the `passwd` file.

As with user (login) names, you should adopt a scheme to assign unique UIDs. Some companies assign unique employee numbers, and administrators add 1000 to the employee number to create a unique UID number for each employee.

To minimize security risks, you should avoid reusing the UIDs from deleted accounts. If you must reuse a UID, "wipe the slate clean" so the new user is not affected by attributes set for a former user. For example, a former user may have been denied access to a printer—by being included in a printer deny list—but that attribute may not be appropriate for the new user. If need be, you can use duplicate UIDs in an NIS+ domain if the supply of unique UIDs is exhausted.

Passwords

Although user names are publicly known, passwords must be kept secret and known only to users. Each user account should be assigned a password, which is a combination of six to eight letters and numbers. You can set a user's password when you create the user account and have the user change it when logging in to a system for the first time.

To make your computer systems more secure, ask users to change their passwords periodically. For a high level of security, you should require users to change their passwords every six weeks. Once every three months is adequate for lower levels of security. System administration logins (such as `root` and `sys`) should be changed monthly, or whenever a person who knows the root password leaves the company or is reassigned.

Many breaches of computer security involve guessing a legitimate user's password. You should make sure that users avoid using proper nouns, names, login names, and other passwords that a person might guess just by knowing something about the user.

Good choices for passwords include:

- Phrase (`beammeup`)
- Nonsense words made up of the first letters of every word in a phrase (`swotrb` for SomeWhere Over The RainBow)
- Words with numbers or symbols substituted for letters (`sn00py` for snoopy)

Do not use these choices for passwords:

- Your name, forwards, backwards, or jumbled
- Names of family members or pets
- Car license numbers
- Telephone numbers
- Social Security numbers
- Employee numbers
- Names related to a hobby or interest
- Seasonal themes, such as Santa in December
- Any word in the dictionary

Password Aging

If you are using NIS+ or the `/etc` files to store user account information, you can set up password aging on a user's password. Password aging enables you to force users to change their passwords periodically or to prevent a user from changing a password before a specified interval. If you want to prevent an intruder from gaining undetected access to the system by using an old and inactive account, you can also set a password expiration date when the account will be disabled.

Home Directories

The home directory is the portion of a file system allocated to a user for storing private files. The amount of space you allocate for a home directory depends on the kinds of files the user creates and the type of work done. As a general rule, you should allocate at least 15 Mbytes of disk space for each user's home directory.

A home directory can be located either on the user's local system or on a remote file server. In either case, by convention the home directory should be created as `/export/home/`*username*. For a large site, you should store home directories on a server. Use a separate file system for each `/export/home`*n* directory to facilitate backing up and restoring home directories (for example, /export/`home1`, /export/`home2`).

Regardless of where their home directory is located, users usually access their home directories through a mount point named `/home/`*username*. When Autofs is used to mount home directories, you are not permitted to create any directories under the `/home` mount point on any system. The system recognizes the special status of `/home` when Autofs is active. For more information about automounting home directories, see "Mount Home Directories Automatically" on page 13.

To use the home directory anywhere on the network, you should always refer to it as `$HOME`, not as `/export/home/`*username*. The latter is machine-specific. In addition, any symbolic links created in a user's home directory should use relative paths (for example, `../../../x/y/x`), so the links will be valid no matter where the home directory is mounted.

The User's Work Environment

Besides having a home directory to create and store files, users need an environment that gives them access to the tools and resources they need to do their work. When a user logs in to a system, the user's work environment is determined by initialization files that are defined by the user's startup shell, such as the C, Korn, or Bourne shell.

A good strategy for managing the user's work environment is to provide customized user initialization files (`.login`, `.cshrc`, `.profile`) in the user's home directory. See "Customizing a User's Work Environment" on page 23 for detailed information about customizing user initialization files for users. After you create the customized user initialization files, you can add them to a user's home directory when you create a new user account.

A recommended one-time task is to set up separate directories, called skeleton directories, on a server (you can use the same server where the user's home directories are stored). The skeleton directories enable you to store customized user initialization files for different types of users.

Note – Do not use system initialization files (`/etc/profile`, `/etc/.login`) to manage a user's work environment, because they reside locally on systems and are not centrally administered. For example, if Autofs is used to mount the user's home directory from any system on the network, then you would have to modify the system initialization files on each system to ensure a consistent environment when a user moves from system to system.

Guidelines for Managing Groups

A *group* is a collection of users who can share files and other system resources. For example, the set of users working on the same project could be formed into a group. A group is traditionally known as a UNIX group.

Each group must have a name, a group identification (GID) number, and a list of user names that belong to the group. A GID identifies the group internally to the system. There are two types of groups that a user can belong to:

- **Primary group –** Specifies a group that the operating system will assign to files created by the user. Each user must belong to a primary group.

- **Secondary groups –** Specifies one or more groups to which a user also belongs. Users can belong to up to 16 secondary groups.

Sometimes a user's secondary group is not important. For example, ownership of files reflect the primary group, not any secondary groups. Other applications, however, may rely on a user's secondary memberships. For example, a user has to be a member of the sysadmin group (group 14) to use the Solstice AdminSuite software, but it doesn't matter if group 14 is his or her current primary group.

The `groups` command shows the groups a user belongs to. A user can have only one primary group at a time. However, the user can temporarily change the primary group (with the `newgrp` command) to any other group in which he or she is a member.

When adding a user account, you must assign a primary group for a user or accept the default: `staff` (group 10). The primary group should already exist (if it doesn't exist, specify the group by a GID number). User names are not added to primary groups. If they were, the list might become too long. Before you can assign users to a new secondary group, you must create the group and assign it a GID number.

Groups can be local to a system or can be managed through a name service. To simplify group administration, you should use a name service like NIS+, which enables you to centrally manage group memberships.

Tools for Managing User Accounts and Groups

In previous Solaris releases, you may have used the Solaris commands or Administration Tool to manage user accounts and groups. In the Solaris 2.5 release, there are three ways to manage user accounts and groups:

- **Solaris commands –** The commands `useradd` and `groupadd` are provided with Solaris to set up user accounts and groups on a local system only. The commands do not change name service maps or tables.
- **Admintool –** A new tool to manage `/etc` files on local systems. This tool should not be used in a name service environment.
- **Solstice AdminSuite –** Includes the tools User Manager and Group Manager to set up users and groups in the NIS or NIS+ name service and on a local system (`/etc` files).

You should use the Solstice AdminSuite tools User Manager and Group Manager to manage user account and group information in a networked and name service environment. User Account and Group Manager offer ease of use and provides support for the following name services:

- NIS+ tables
- NIS maps
- Local `/etc` files

To show you the benefits of the Solstice AdminSuite tools over using the commands, Table 1-2 lists the tasks that the User Manager and Group Manager can perform and the equivalent Solaris commands they replace.

Table 1-2 Managing User Accounts and Groups Without AdminSuite *(1 of 3)*

Task	If You Use This Name Service	Then Use These Commands
Add a User Account	NIS+	`nistbladm` `nisclient`
	NIS	`useradd` `make`
	None	`useradd`
Modify a User Account	NIS+	`nistbladm`
	NIS	`usermod` `make`
	None	`usermod`

Table 1-2 Managing User Accounts and Groups Without AdminSuite *(2 of 3)*

Task	**If You Use This Name Service**	**Then Use These Commands**
Delete a User Account	NIS+	`nistbladm` `nisclient`
	NIS	`userdel` `make`
	None	`userdel`
Copy an Existing User Account	NIS, NIS+ or None	*—not available—*
Set Up User Account Defaults	NIS+	*—not available—*
	NIS	`useradd -D` `make`
	None	`useradd -D`
Disable a User Account	NIS+	`nistbladm`
	NIS	`vipw` `make`
	None	`vipw`
Change a User's Password	NIS+	`nispasswd`
	NIS	`yppasswd`
	None	`passwd`
Sort User Accounts	NIS+	`niscat` `sort`
	NIS	`ypcat` `sort`
	None	`awk` `sort`
Find a User Account	NIS+	`nismatch`
	NIS	`ypmatch`
	None	`grep`
Add a Group	NIS+	`nistbladm`
	NIS	`groupadd` `make`
	None	`groupadd`

Table 1-2 Managing User Accounts and Groups Without AdminSuite *(3 of 3)*

Task	If You Use This Name Service	Then Use These Commands
Modify Users in a Group	NIS+	`nistbladm`
	NIS	`groupmod` `make`
	None	`groupmod`
Delete a Group	NIS+	`nistbladm`
	NIS	`groupdel` `make`
	None	`groupdel`

What You Can Do With User Manager

User Manager is a graphical user interface that enables you to set up user accounts on a local or remote system or in a name service environment. With a name service like NIS+, you can manage user accounts in a centralized manner so that important user account information, such as user names, do not have to be duplicated on every system in the network.

Mount Home Directories Automatically

One of User Manager's key features is setting up automounting of the user's home directory. The AutoHome Setup option enables users to log in to any system and have their home directories immediately available. To use this feature, make sure the `automount` entry in the target system's `/etc/nsswitch.conf` file is configured to use a name service. If you do not choose to set up automounting, you must manually set up the mounting required to make a user's home directory available on the network.

To support automatic mounting of home directories, the Solaris 2.x system software includes the `auto_master` file, which has the following entry:

```
/home       auto_home
```

This entry tells Autofs to mount the directories specified in the `auto_home` file on the `/home` mount point on the local system. The entries in `auto_home` have the following format:

```
username      system-name:/export/home/username
```

When a user logs in with *username*, Autofs mounts the specified directory (`/export/home/`*username*) from the server (*system-name*) on the `/home` mount point on the system where the user has just logged in. Autofs works even when the home directory is stored on the same system the user has logged in to.

Add New Users Quickly

Through the Copy User feature, User Manager can save you time when adding a new user. The Copy User feature enables you to copy parameters from an existing user account, such as a user account's group membership, default shell environment, home directory location, mailbox location, and other useful information. So, when you add a new user, you don't have to fill in the parameters that contain the copied values.

Set Default Settings for Adding Users

Through the Set Default feature, User Manager can save you time when adding a new user. The Set Default feature enables you to create default user account parameters such as group membership, default shell environment, password settings, home directory location, mailbox location, and other useful information. So, when you add a new user, you don't have to fill in the parameters that contain the default values.

Modify User Accounts

Unless you define a user name or UID number that conflicts with an existing one, you should never need to modify a user account's login name or UID number. If you do change a user name or UID number, the home directory's ownership is not changed (if a home directory exists for the user). You must manually change the ownership of all files and directories, including the mailbox, that have the old UID number.

One part of a user account that you can change is a user's group memberships. User Manager's Modify User option lets you add or delete a user's secondary groups. Alternatively, you can use Group Manager to directly modify a group's member list.

You can also modify the following parts of a user account:

- Comment
- Login shell
- Passwords
- Home directory
- AutoHome setup
- Credential table setup
- Mail server

Delete User Accounts

When you delete a user account with User Manager, the software deletes the entries in the `passwd`, `group`, `aliases`, `cred`, and `auto_home` files. In addition, you can delete the files in the user's home directory and delete the contents of the user's mailbox.

Only the single mail alias that directs mail to the user's mail box is deleted; the user name is not deleted from any other mail aliases. If you want to delete entries from mail aliases other than the one set up to direct mail to your mailbox, you must delete them by hand.

Add Customized User Initialization Files

Although you can't create customized user initialization files with User Manager, you can populate a user's home directory with user initialization files located in a specified "skeleton" directory.

When adding a user account with User Manager, you can specify a global path name (such as */net/machine-name/directory-name*) to a skeleton directory that contains the customized user initialization files. Autofs mounts the shared directory from the server, and the user's home directory is populated with the user initialization files.

Administer Passwords

You can use User Manager for password administration, which includes specifying a normal password for a user account, enabling users to create their own passwords during their first login, disabling or locking a user account, or specifying expiration dates and password aging information.

Note – Password aging is not supported by the NIS name service.

Disable User Accounts

Occasionally, you may need to temporarily or permanently disable a login account. Disabling or locking a user account means that an invalid password, `*LK*`, is assigned to the user account, preventing future logins.

The easiest way to disable a user account is to use User Manager to lock the password for an account. You can also enter an expiration date in the Expiration Date field to set how long the user account is disabled. In addition, if NIS+ is the selected name service and DES entries have been added to the `cred` table, User Manager removes the DES entries.

Other ways to disable a user account is to set up password aging or by just changing the user's password.

What You Can't Do With User Manager

Table 1-3 shows the limitations of User Manager and their suggested workarounds.

Table 1-3 User Manager Limitations and Workarounds

Limitation	Workaround
If you add a new user, User Manager does not automatically share the user's home directory so the user's system can remotely mount it.	Become root on the system that contains the home directory and either share the home directory or verify that it is already shared. Both of these workarounds can be done through the `share` command.
If you do not select AutoHome Setup (Autofs) for the user's home directory, User Manager does not set up the user's system to remotely mount the user's home directory.	Become root on the user's system and remotely mount the user's home directory. This can be done manually with the `mount` command or by editing the system's `/etc/vfstab` file.

What You Can Do With Group Manager

Group Manager is a graphical user interface that enables you to add or remove users from a group on a local or remote system or in a name service environment. With a name service like NIS+, you can manage group information in a centralized location so that it does not have to be duplicated on every system in the network.

Group Manager enables you to add and delete groups. When projects are finished, groups set up for those projects may no longer be needed, and you may want to delete these groups. Be careful to avoid conflicts if you reuse the GID numbers from deleted groups.

Where User Account and Group Information Is Stored

Depending on your site policy, you can store user account and group information in a name service or a local system's `/etc` files. In the NIS+ name service, information is stored in tables, and in the NIS name service, information is stored in maps.

Note – To avoid confusion, the location of the user account and group information will be generically referred to as a *file* rather than a *file, table*, or *map*.

Most of the user account information is stored in the `passwd` file. However, password encryption and password aging is stored in the `passwd` file when using NIS or NIS+ and in the `/etc/shadow` file when using `/etc` files Password aging is not available when using NIS.

Group information is stored in the `group` file.

Fields in the Password File

The fields in the `passwd` file are separated by colons and contain the following information:

```
username:password:uid:gid:comment:home-directory:login-shell
```

For example:

```
kryten:x:101:100:Kryten Series 4000:/export/home/kryten:/bin/csh
```

Table 1-4 describes the `passwd` file fields.

Table 1-4 Fields in the `passwd` File

Field Name	**Description**
username	Contains the user or (login) name. User names should be unique and consist of 2-8 letters (A-Z, a-z) and numerals (0-9). The first character must be a letter, and at least one character must be a lowercase letter. User names cannot contain underscores or spaces.
password	Contains an `x`, a placeholder for the encrypted password. (Password is an obsolete field.) The encrypted password is stored in the `shadow` file.
uid	Contains a user identification (UID) number that identifies the user to the system. UID numbers for regular users should range from 100 to 60000. All UID numbers should be unique.

Table 1-4 Fields in the `passwd` File *(Continued)*

Field Name	Description
gid	Contains a group identification (GID) number that identifies the user's primary group. Each GID number must be a whole number between 0 and 60002 (60001 and 60002 are assigned to `nobody` and `noaccess`, respectively).
comment	Usually contains the full name of the user. (This field is informational only.) It is sometimes called the GECOS field because it was originally used to hold the login information needed to submit batch jobs to a mainframe running GECOS (General Electric Computer Operating System) from UNIX systems at Bell Labs.
home-directory	Contains user's home directory path name.
login-shell	Contains the user's default login shell, which can be `/bin/sh`, `/bin/csh` or `/bin/ksh`. Table 62-8 on page 918 contains a description of shell features.

Fields in the Shadow File

The fields in the `shadow` file are separated by colons and contain the following information:

```
username:password:lastchg:min:max:warn:inactive:expire
```

For example:

```
rimmer:86Kg/MNT/dGu.:8882:0::5:20:8978
```

Table 1-5 describes the `shadow` file fields.

Table 1-5 Fields in the `shadow` File

Field Name	Description
username	Contains the user or (login) name.
password	May contain the following entries: a 13-character encrypted user password; the string `*LK*`, which indicates an inaccessible account; or the string `NP`, which indicates no password for the account.
lastchg	Indicates the number of days between January 1, 1970, and the last password modification date.
min	Contains the minimum number of days required between password changes.
max	Contains the maximum number of days the password is valid before the user is prompted to specify a new password.
inactive	Contains the number of days a user account can be inactive before being locked.
expire	Contains the absolute date when the user account expires. Past this date, the user cannot log in to the system.

Fields in the Group File

The fields in the `group` file are separated by colons and contain the following information:

```
group-name:group-password:gid:user-list
```

For example:

```
bin::2:root,bin,daemon
```

Table 1-6 describes the `group` file fields.

Table 1-6 Fields in the `group` File

Field Name	Description
group-name	Contains the name assigned to the group. For example, members of the chemistry department in a university may be called `chem`. Group names can have a maximum of nine characters.
group-password	Usually contains an asterik or is empty. The *group-password* field is a relic of earlier versions of UNIX. If a group has a password, the `newgrp` command prompts users to enter it. However, there is no utility to set the password.
gid	Contains the group's GID number. It must be unique on the local system, and should be unique across the entire organization. Each GID number must be a whole number between 0 and 60002. Numbers under 100 are reserved for system default group accounts. User defined groups can range from 100 to 60000. (60001 and 60002 are reserved and assigned to `nobody` and `noaccess`, respectively.)
user-list	Contains a list of groups and a comma-separated list of user names, representing the user's secondary group memberships. Each user can belong to a maximum of 16 secondary groups.

UNIX User Groups

By default, all Solaris 2.x systems have these groups:

```
root::0:root
other::1:
bin::2:root,bin,daemon
sys::3:root,bin,sys,adm
adm::4:root,adm,daemon
uucp::5:root,uucp
mail::6:root
tty::7:root,tty,adm
lp::8:root,lp,adm
nuucp::9:root,nuucp
staff::10:
daemon::12:root,daemon
sysadmin::14:
nobody::60001:
noaccess::60002:
```

Customizing a User's Work Environment

Part of setting up a user's home directory is providing user initialization files for the user's login shell. A *user initialization file* is a shell script that sets up a work environment for a user after the user logs in to a system. Basically, you can perform any task in a user initialization files that you can do in a shell script, but its primary job is to define the characteristics of a user's work environment, such as a user's search path, environment variables, and windowing environment. Each login shell has its own user initialization file (or files), which are listed in Table 1-7.

Table 1-7 User Initialization Files for Bourne, C, and Korn Shells

Shell	User Initialization File	Purpose
Bourne	`$HOME/.profile`	Defines user's environment at login
C	`$HOME/.cshrc`	Defines user's environment for all C shells invoked after login shell
	`$HOME/.login`	Defines user's environment at login
Korn	`$HOME/.profile`	Defines user's environment at login
	`$HOME/$ENV`	Defines user's environment at login in the file specified by the Korn shell's `ENV` environment variable

The Solaris 2.x system software provides default user initialization files for each shell in the `/etc/skel` directory on each system, as shown in Table 1-8.

Table 1-8 Default User Initialization Files

Shell	Default File
C	`/etc/skel/local.login`
	`/etc/skel/local.cshrc`
Bourne or Korn	`/etc/skel/local.profile`

You can use these as a starting point and modify them to create a standard set of files that will provide the work environment common to all users, or you can modify them to provide the working environment for different types of users. See “How to Customize User Initialization Files” on page 41 for step-by-step instructions on how to create sets of user initialization files for different types of users.

Use Site Initialization Files

When customizing a user initialization file, it is important that the user initialization files can be customized by both the administrator and the user. This important feature can be accomplished with centrally located and globally distributed user initialization files, called site initialization files. Site initialization files enable you to continually introduce new functionality to the user’s work environment, while enabling the user to also customize the user initialization file.

When you reference a site initialization file in a user initialization file, all updates to the site initialization file are automatically reflected when the user logs in to the system or when a user starts a new shell. Site initialization files are designed for you to distribute site-wide changes to users’ work environments that you did not anticipate when you added the users.

Any customization that can be done in a user initialization file can be done in a site initialization file. These files typically reside on a server (or set of servers), and appear as the first statement in a user initialization file. Also, each site initialization file must be the same type of shell script as the user initialization file that references it.

To reference a site initialization file in a C-shell user initialization file, place a line similar to the following at the beginning of the user initialization file:

```
source /net/machine-name/export/site-files/site-init-file
```

To reference a site initialization file in a Bourne- or Korn-shell user initialization file, place a line similar to the following at the beginning of the user initialization file:

```
. /net/machine-name/export/site-files/site-init-file
```

Avoid Local System References

You should not add specific references to the local system in the user's initialization file. You want the instructions in a user initialization file to be valid regardless of the system to which the user logs in. For example:

- To make a user's home directory available anywhere on the network, always refer to the home directory with the variable `$HOME`. For example, use `$HOME/bin`; do not use `/export/home/`*username*`/bin`. `$HOME` works when the user logs in to another system, when home directories are automounted.
- To access files on a local disk, use global path names, like `/net/`*machine-name*`/`*directory-name*. Any directory referenced by `/net/`*machine-name* can be mounted automatically on any system on which the user logs in, assuming the system is running Autofs.

Shell Features

Table 1-9 lists basic shell features that each shell provides, which can help you determine what you can and can't do when creating user initialization files for each shell.

Table 1-9 Basic Features of Bourne, C, and Korn Shells

Feature	**Bourne**	**C**	**Korn**
Known as the standard shell in UNIX	Yes	No	No
Compatible syntax with Bourne shell	–	No	Yes
Job control	Yes	Yes	Yes
History list	No	Yes	Yes
Command-line editing	No	Yes	Yes
Aliases	No	Yes	Yes
Single-character abbreviation for login directory	No	Yes	Yes
Protection from overwriting (`noclobber`)	No	Yes	Yes

Table 1-9 Basic Features of Bourne, C, and Korn Shells *(Continued)*

Feature	**Bourne**	**C**	**Korn**
Setting to ignore Control-d (`ignoreeof`)	No	Yes	Yes
Enhanced `cd`	No	Yes	Yes
Initialization file separate from `.profile`	No	Yes	Yes
Logout file	No	Yes	No

Shell Environment

A shell maintains an *environment* that includes a set of variables defined by the `login` program, the system initialization file, and the user initialization files. In addition, some variables are defined by default. A shell can have two types of variables:

- **Environment variables –** Variables that are exported to all processes spawned by the shell. Their settings can be seen with the `env` command. A subset of environment variables, like `PATH`, affects the behavior of the shell itself.
- **Shell (local) variables –** Variables that affect only the current shell. In the C shell, a set of these shell variables have a special relationship to a corresponding set of environment variables. These shell variables are `user`, `term`, `home`, and `path`. The value of the environment variable counterpart is initially used to set the shell variable.

In the C shell, you use the lowercase names with the `set` command to set shell variables and use uppercase names with the `setenv` command to set environment variables. If you set a shell variable, the shell sets the corresponding environment variable and vice versa. For example, if you update the `path` shell variable with a new path, the shell also updates the `PATH` environment variable with the new path.

In the Bourne and Korn shells, you use the uppercase names with the `setenv` command to set both shell and environment variables. You also have to use the `export` command to finish setting environment variables. For all shells, you generally refer to shell and environment variables by their uppercase names.

In a user initialization file, you can customize a user's shell environment by changing the values of the predefined variables or by specifying additional variables. Table 1-10 shows how to set environment variables in a user initialization file.

Table 1-10 Setting Environment Variables in a User Initialization File

If You Want to Set a User's Environment Variables for The ...	Then Add the Following Line to the User Initialization File ...

C shell	`setenv` *VARIABLE value*
	
	`Example:`
	`setenv MAIL /var/mail/ripley`

Bourne or Korn shell	*VARIABLE=value*`; export` *VARIABLE*
	
	`Example:`
	`MAIL=/var/mail/ripley;export MAIL`

Table 1-11 describes environment and shell variables you may want to customize in a user initialization file. For more information about variables used by the different shells, see `sh(1)`, `ksh(1)`, or `csh(1)`.

Table 1-11 Shell and Environment Variable Descriptions *(1 of 3)*

Variable	Description
`ARCH`	Sets the user's system architecture (for example, `sun4`, `i386`). This variable can be set with `ARCH = `uname -p`` (in Bourne or Korn shells) or `setenv ARCH `uname -p`` (in C shell). There is no built-in behavior of the shell that depends on this variable. It's just a useful variable for branching within shell scripts.
`CALENDAR`	Sets the path to the Calendar executables.
`CDPATH` (or `cdpath` in the C shell)	Sets a variable used by the `cd` command. If the target directory of the `cd` command is specified as a relative path name, the `cd` command will first look for the target directory in the current directory ("."). If the target is not found, the path names listed in the `CDPATH` variable are searched consecutively until the target directory is found and the directory change is completed. If the target directory is not found, the current working directory is left unmodified. For example, the `CDPATH` variable is set to `/home/jean`, and two directories exist under `/home/jean`: `bin` and `rje`. If you are in the `/home/jean/bin` directory and type `cd rje`, you change directories to `/home/jean/rje`, even though you do not specify a full path.
`DESKSET`	Sets the path to the DeskSet executables.

Table 1-11 Shell and Environment Variable Descriptions (2 of 3)

Variable	Description
history	Sets history for the C shell.
HOME (or home in the C shell)	Sets the path to the user's home directory.
LANG	Sets the locale.
LOGNAME	Defines the name of the user currently logged in. The default value of LOGNAME is automatically set by the login program to the user name specified in the passwd file. You should only need to refer to (not reset) this variable.
LPDEST	Sets the user's default printer.
MAIL	Sets the path to the user's mailbox.
MANPATH	Sets the hierarchies of man pages available.
MANSECTS	Sets the hierarchies of man pages available.
OPENWINHOME	Sets the path to the OpenWindows subsystem.
PATH (or path in the C shell)	Lists, in order, the directories that the shell searches to find the program to run when the user types a command. If the directory is not in the search path, users must type the complete path name of a command. The default PATH is automatically defined and set as specified in .profile (Bourne or Korn shell) or .cshrc (C shell) as part of the login process. The order of the search path is important. When identical commands exist in different locations, the first command found with that name is used. For example, suppose that PATH is defined (in Bourne and Korn shell syntax) as PATH=/bin:/usr/bin:/usr/sbin:$HOME/bin and a file named sample resides in both /usr/bin and /home/jean/bin. If the user types the command sample without specifying its full path name, the version found in /usr/bin is used.
prompt	Defines the shell prompt for the C shell.
PS1	Defines the shell prompt for the Bourne or Korn shell.
SHELL (or shell in the C shell)	Sets the default shell used by make, vi, and other tools.

Table 1-11 Shell and Environment Variable Descriptions *(3 of 3)*

Variable	Description
TERMINFO	Specifies the path name for an unsupported terminal that has been added to the terminfo file. Use the TERMINFO variable in /etc/profile or /etc/.login. When the TERMINFO environment variable is set, the system first checks the TERMINFO path defined by the user. If it does not find a definition for a terminal in the TERMINFO directory defined by the user, it searches the default directory, /usr/share/lib/terminfo, for a definition. If it does not find a definition in either location, the terminal is identified as "dumb."
TERM (or term in the C shell)	Defines the terminal. This variable should be reset in /etc/profile or /etc/.login. When the user invokes an editor, the system looks for a file with the same name as the definition of this environment variable. The system searches the directory referenced by TERMINFO to determine the terminal characteristics.
TZ	Sets the time zone, which is used to display dates, for example, in the ls -l command. If TZ is not set in the user's environment, the system setting is used; otherwise, Greenwich Mean Time is used.

The PATH *Variable*

When the user executes a command by using the full path, the shell uses that path to find the command. However, when users specify only a command name, the shell searches the directories for the command in the order specified by the PATH variable. If the command is found in one of the directories, the shell executes it.

A default path is set by the system, but most users modify it to add other command directories. Many user problems related to setting up the environment and accessing the right version of a command or a tool can be traced to incorrectly defined paths.

Guidelines

Here are some guidelines for setting up efficient PATH variables:

- If security is not a concern, put the current working directory (.) first in the path. However, including the current working directory in the path poses a security risk that you may want to avoid, especially for root.
- Keep the search path as short as possible. The shell searches each directory in the path. If a command is not found, long searches can slow down system performance.
- The search path is read from left to right, so you should put directories for commonly used commands at the beginning of the path.
- Make sure directories are not duplicated in the path.
- Avoid searching large directories, if possible. Put large directories at the end of the path.
- Put local directories before NFS™ mounted directories to lessen the chance of "hanging" when the NFS server does not respond and to reduce unnecessary network traffic.

Examples—Setting a User's Default Path

The following examples show how to set a user's default path to include the home directory and other NFS mounted directories (the current working directory is specified first in the path). In a C-shell user initialization file, you would add the following:

```
set path=(. /usr/bin $HOME/bin /net/glrr/files1/bin)
```

In a Bourne- or Korn-shell user initialization file, you would add the following:

```
PATH=.:/usr/bin:/$HOME/bin:/net/glrr/files1/bin
export PATH
```

The Locale Variables

The LANG and LC environment variables specify the locale-specific conversions and conventions for the shell, like time zones, collation orders, and formats of dates, time, currency, and numbers. In addition, you can use the `stty` command in a user initialization file to set whether the system will support multibyte characters.

LANG sets all possible conversions and conventions for the given locale. If you have special needs, you can set various aspects of localization separately through these LC variables: LC_COLLATE, LC_CTYPE, LC_MESSAGES, LC_NUMERIC, LC_MONETARY, and LC_TIME.

Table 1-12 describes the values for the LANG and LC environment variables.

Table 1-12 Values for LANG and LC Variables

Value	Locale
de	German
fr	French
iso_8895_1	English and European
it	Italian
japanese	Japanese
korean	Korean
sv	Swedish
tchinese	Taiwanese

Examples—Setting the Locale Using the LANG Variables

The following examples show how to set the locale using the LANG environment variables. In a C-shell user initialization file, you would add the following:

```
setenv LANG DE
```

In a Bourne- or Korn-shell user initialization file, you would add the following:

```
LANG=DE; export LANG
```

Default File Permissions (umask)

When you create a file or directory, the default file permissions assigned to the file or directory are controlled by the *user mask*. The user mask should be set by the umask command in a user initialization file. You can display the current value of the user mask by typing umask and pressing Return.

The user mask can be set with a three-digit octal value. The first digit sets permissions for the user; the second sets permissions for group; the third sets permissions for other (also referred to as world). Note that if the first digit is zero, it is not displayed. For example, if umask is set to 022, 22 is displayed.

To determine the umask value you want to set, subtract the value of the permissions you want from 666 (for a file) or 777 (for a directory). The remainder is the value to use with the umask command. For example, suppose you want to change the default mode for files to 644 (rw-r--r--). The difference between 666 and 644 is 022, which is the value you would use as an argument to the umask command.

You can also determine the umask value you want to set by using Table 1-13, which shows the file and directory permissions that are created for each of the octal values of umask.

Table 1-13 Permissions for umask Values

umask **Octal Value**	**File Permissions**	**Directory Permissions**
0	rw-	rwx
1	rw-	rw-
2	r--	r-x
3	r--	r--
4	-w-	-wx
5	-w-	-w-
6	--x	--x
7	--- (none)	--- (none)

The following line in a user initialization file sets the default file permissions to `rw-rw-rw-`.

```
umask 000
```

Examples of User and Site Initialization Files

The following sections provide examples of user and site initialization files that you can use to start customizing your own initialization files. Many of the examples use system names and paths that you will have to change for your particular site.

Example—`.profile` File

```
❶ PATH=$PATH:$HOME/bin:/usr/local/bin:/usr/ccs/bin:.
❷ MAIL=/var/mail/$LOGNAME
❸ NNTPSERVER=server1
❹ MANPATH=/usr/share/man:/usr/local/man
❺ PRINTER=printer1
❻ umask 022
❼ export PATH MAIL NNTPSERVER MANPATH PRINTER
```

Figure 1-1 Example `.profile` File

❶ Defines the user's shell search path.

❷ Defines the path to the user's mail file.

❸ Defines the environment variable for the user's Usenet news server.

❹ Defines the user's search path for man pages.

❺ Defines the user's default printer.

❻ Sets the user's default file creation permissions.

❼ Sets the listed environment variables.

Example— `.cshrc` *File*

```
❶ set path=($PATH $HOME/bin /usr/local/bin /usr/ccs/bin)
❷ setenv MAIL /var/mail/$LOGNAME
❸ setenv NNTPSERVER server1
❹ setenv PRINTER printer1
❺ alias h history
❻ umask 022
❼ source /net/server2/site-init-files/site.login
```

Figure 1-2 Example `.cshrc` File

❶ Sets the user's shell search path.

❷ Sets the path to the user's mail file.

❸ Sets the user's Usenet news server.

❹ Sets the user's default printer.

❺ Creates an alias for the `history` command (the user will need to type only `h` to run the `history` command).

❻ Sets the user's default file creation permissions.

❼ Runs the site intialization file shown in Figure 1-3 on page 35.

Example—Site Initialization File

shows an example site initialization file in which a user can choose a particular version of an application.

```
# @(#)site.login
main:
echo "Application Environment Selection"
echo ""
echo "1. Application, Version 1"
echo "2. Application, Version 2"
echo ""
echo -n "Type 1 or 2 and press Return to set your application
environment: "

set choice = $<
    if ( $choice !~ [1-2] ) then
    goto main
    endif
switch ($choice)

case "1":

    setenv APPHOME /opt/app-v.1
    breaksw

case "2":

    setenv APPHOME /opt/app-v.2
    endsw
```

Figure 1-3 Example Site Initialization File

This site initialization file could be referenced in a user's `.cshrc` file (C shell users only) with the following line:

```
source /net/server2/site-init-files/site.login
```

In this line, the site initialization file is named `site.login` and it is located on a server named `server2`. This line also assumes that the automounter is running on the user's system.

Setting Up and Maintaining User Accounts and Groups 2

This chapter describes the procedures for setting up and maintaining user accounts and groups. This is a list of the step-by-step instructions in this chapter.

2

For overview information about Managing User Accounts and Groups, see Chapter 1, “Overview of Managing User Accounts and Groups.”

Setting Up User Accounts

Table 2-1 Task Map: Setting Up User Accounts

Activity	Description	For Instructions, Go To	
Customize User Initialization Files	Optional. Set up user initialization files (`.cshrc`, `.profile`, `.login`), so you can provide new users with consistent environments. This is usually a one-time task.	▼ How to Customize User Initialization Files	page 41
Add Groups	Optional. To help administer users, add groups by choosing Add from the Group Manager’s Edit menu. This is usually a one-time task.	▼ How to Add a Group	page 45
Set User Account Defaults	Optional. Before you add several user accounts, set up defaults for the Add window by choosing Set Defaults from the User Manager’s Edit menu. Setting up defaults can increase the consistency and efficiency of adding new user accounts.	▼ How to Set Up User Account Defaults	page 48
Add a User Account	**Add a New User Account** Add a user account by choosing Add from the User Manager’s Edit menu.	▼ How to Add a New User Account	page 49
	Copy an Existing User Account Copy an existing user account by choosing Copy from the User Manager’s Edit menu. This is useful if you need to add a user account that is similar to an existing user account.	▼ How to Copy an Existing User Account	page 51
Share the User’s Home Directory	Share the user’s home directory, so the directory can be remotely mounted from the user’s system. If one system contains all the home directories, this is usually a one-time task.	▼ How to Share a User’s Home Directory	page 52

Table 2-1 Task Map: Setting Up User Accounts *(Continued)*

Activity	Description	For Instructions, Go To	
Mount the User's Home Directory	**Needed If Not Using AutoFS** If you did not select AutoFS when creating the user account (the `AutoHome Setup` field) and the user's home directory is located on another system, manually mount the user's home directory on the user's system.	▼ How to Mount a User's Home Directory	page 55

User Information Data Sheet

You may find it useful to create a form like the one below to gather information about users before adding their accounts. The items above the double line reflect the information specified when adding a user account with User Manager.

Item	Description
User Name:	
UID:	
Primary Group:	
Secondary Groups:	
Comment:	
Default Shell:	
Password Status and Aging:	
Home Directory Server Name:	
Home Directory Path Name:	
Mounting Method:	
Permissions on Home Directory:	
Mail Server:	
Department Name:	
Department Administrator:	
Manager:	
Employee Name:	
Employee Title:	
Employee Status:	
Employee Number:	
Start Date:	
Add to These Mail Aliases:	
Desktop System Name:	

How to Customize User Initialization Files

1. **Become root on the system where the users' home directories are created and shared.**

2. **Create a skeleton directory for each type of user.**

```
# mkdir /shared-directory/skel/user-type
```

In this command,

shared-directory	Is the name of a directory that is available to other systems on the network.
user-type	Is the name of a directory to store initialization files for a type of user.

3. **Copy the default user initialization files into the directories you created for different types of users.**

```
# cp /etc/skel/local.cshrc /shared-directory/skel/user-type/.cshrc
# cp /etc/skel/local.login /shared-directory/skel/user-type/.login
# cp /etc/skel/local.profile /shared-directory/skel/user-type/.profile
```

Note – You can use the `ls -a` command to list `.` (dot) files.

4. **Edit the user initialization files for each user type and customize them based on your site's needs.**

   See "Customizing a User's Work Environment" on page 23 for a detailed description on the ways to customize the user initialization files.

5. **Set the permissions for the user initialization files.**

```
# chmod 744 /shared-directory/skel/user-type/.*
```

Example—Customizing User Initialization Files

The following example customizes the C-shell user initialization file in the `/export/skel/enduser` directory designated for a particular type of user.

```
# mkdir /export/skel/enduser
# cp /etc/skel/local.cshrc /export/skel/enduser/.cshrc

Edit .cshrc file–see “Example—.cshrc File” on page 34

# chmod 744 /export/skel/enduser/.*
```

▼ How to Start Group Manager

1. **Verify that the following prerequisites are met. To use Group Manager, you must have:**
   - Solstice AdminSuite software installed.
   - A bit-mapped display monitor. The Solstice AdminSuite software can be used only on a system with a console that is a bit-mapped screen such as a standard display monitor that comes with a Sun workstation.

     If you want to perform administration tasks on a system with an ASCII terminal as the console, use Solaris commands instead.
   - OpenWindows™ software. Start this software with the following command:

```
$ /usr/openwin/bin/openwin
```

   - Membership in the `sysadmin` group (group 14) and the required access privileges for managing the NIS or NIS+ database.

Note – If your name service is NIS+, you must be a member of the NIS+ admin group.

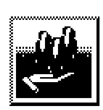
---

2. **Start the Solstice Launcher.**

```
$ solstice &
```

The Solstice Launcher is displayed.

3. **Click on the Group Manager icon.**



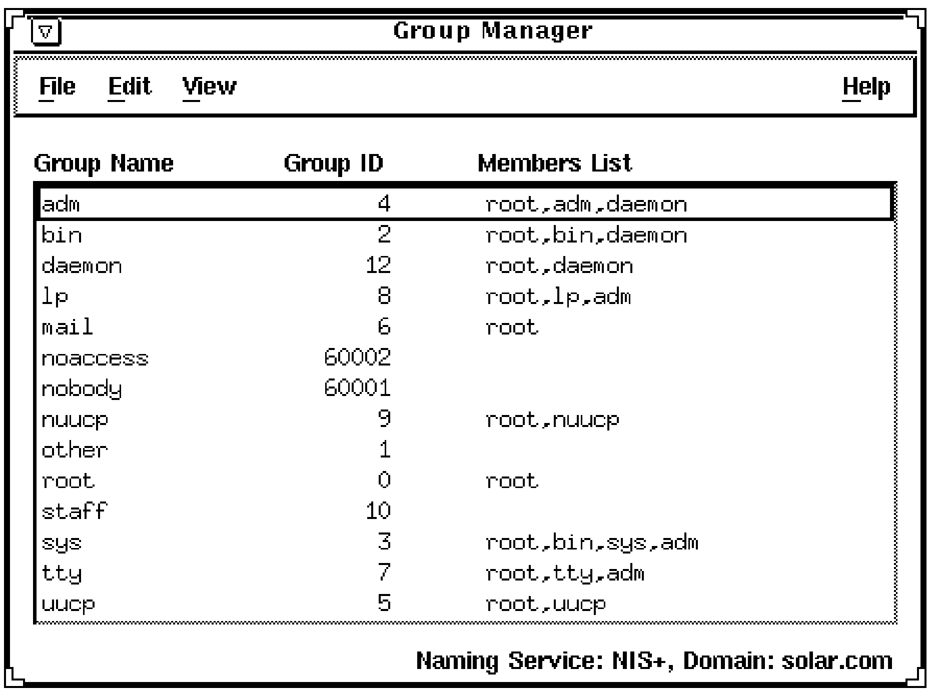
The Load window is displayed.

4. **Select the name service used in your network.**

5. **Check that the domain or host name is correct.**
   If not, type the name of the domain or host you need to access.

6. **Click on OK.**
   The Group Manager main window is displayed.

Example—Group Manager Main Window



▼ How to Add a Group

1. **Start Group Manager from the Solstice Launcher and select the name service, if not done already.**
   See "How to Start Group Manager" on page 43 for more information.

2. **Choose Add from the Edit menu on the Group Manager main window.**
   The Add window is displayed. If you need information to complete a field, click on the Help button to see field definitions for this window.

3. **Type the name of the new group in the Group Name text box.**

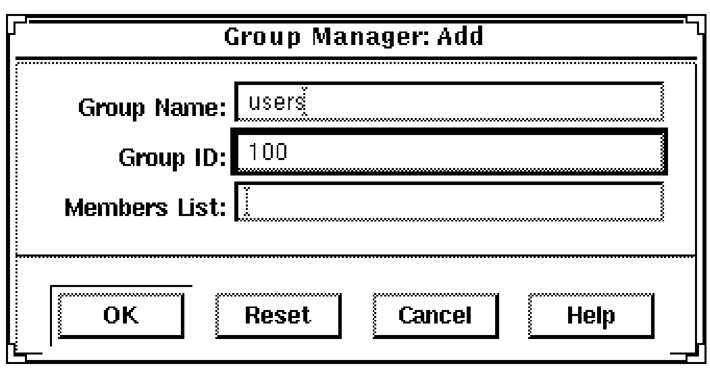
4. **Type the group ID for the new group in the Group ID text box.**
   The group ID should be unique.

5. **(Optional) Type user names in the Members List text box.**
   The list of users will be added to the group. User names must be separated by commas.

6. **Click on OK.**
   The list of groups displayed in the Group Manager main window is updated to include the new group.

Example—Completed Group Manager Add Window

The following example adds a group named `users` that has a group ID of 100.



▼ How to Start User Manager

1. **Verify that the following prerequisites are met. To use User Manager, you must have:**
   - Solstice AdminSuite software installed.
   - A bit-mapped display monitor. The Solstice AdminSuite software can be used only on a system with a console that is a bit-mapped screen such as a standard display monitor that comes with a Sun workstation.

     If you want to perform administration tasks on a system with an ASCII terminal as the console, use Solaris commands instead.
   - OpenWindows™ software. Start this software with the following command:

```
$ /usr/openwin/bin/openwin
```

   - Membership in the `sysadmin` group (group 14) and the required access privileges for managing the NIS or NIS+ database.

Note – If your name service is NIS+, you must be a member of the NIS+ admin group.

---

2. **Start the Solstice Launcher.**

```
$ solstice &
```

   The Solstice Launcher is displayed.

3. **Click on the User Manager icon from the Solstice Launcher.**

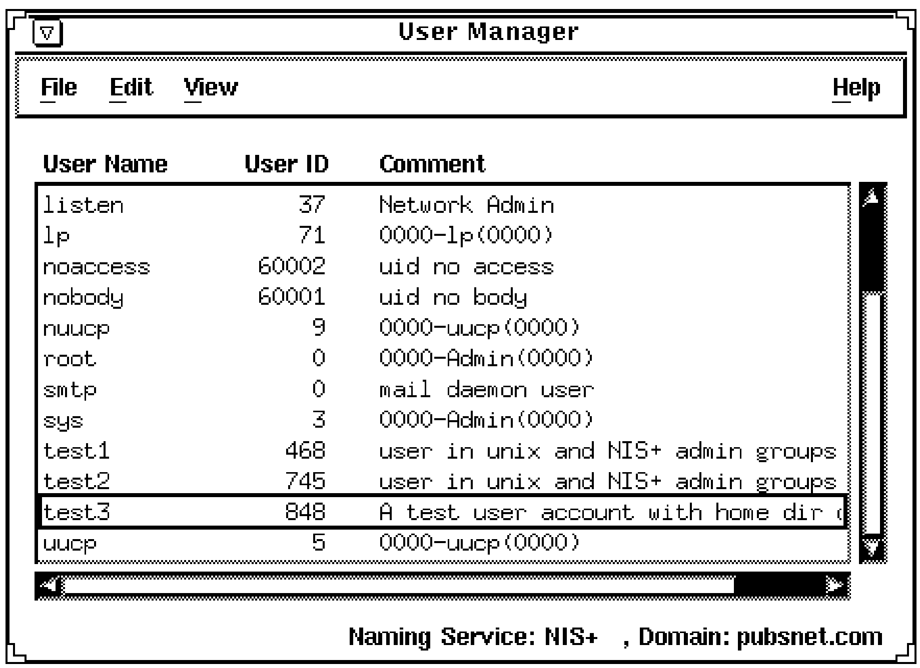
   The Load window is displayed.

4. **Select the name service used in your network.**

5. **Check that the domain or host name is correct.**
   If not, type the name of the domain or host you need to access.

6. **Click on OK.**
   The User Account main window is displayed.

Example—User Manager Main Window



▼ How to Set Up User Account Defaults

1. **Start User Manager from the Solstice Launcher and select the name service, if not done already.**
   See "How to Start User Manager" on page 46 for more information.

2. **Choose Set Defaults from the Edit menu.**
   The Set Add Defaults window is displayed.

3. **Fill in the Set Add Defaults window.**
   The defaults you select will be the initial defaults values in the Add window. If you need information to complete a field, click on the Help button to see field definitions for this window.

   You can set the following defaults:
   - Primary and Secondary Groups
   - Login Shell
   - Password Policy
   - Creating a Home Directory
   - Home Directory Server
   - Skeleton Path (Path to User Initialization Files)
   - Using AutoFS (AutoHome Setup)
   - Permissions in Home Directory
   - Mail Server

4. **Click on OK.**

▼ How to Add a New User Account

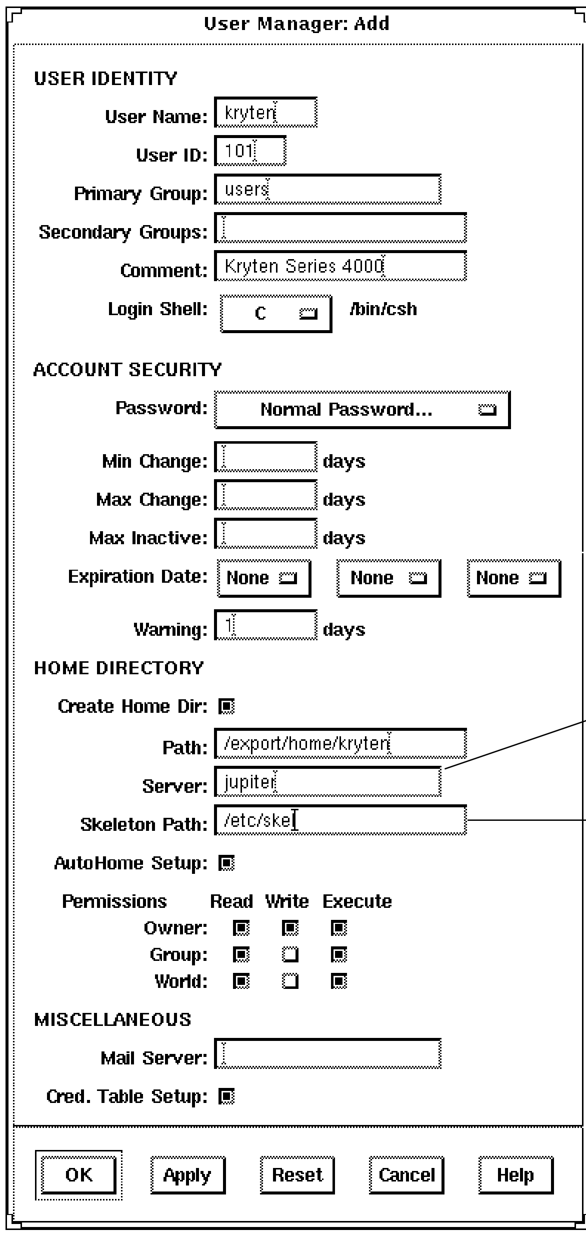
1. **(Optional) Fill out the user information data sheet on page 40.**
2. **Start User Manager from the Solstice Launcher and select the name service, if not done already.**
   See “How to Start User Manager” on page 46 for more information.
3. **Choose Add from the Edit menu.**
   The Add window is displayed.
4. **Fill in the Add window.**
   If you need information to complete a field, click on the Help button to see field definitions for this window.
5. **Click on OK.**
   The list of user accounts displayed in the User Manager main window is updated to include the new user account.

Where to Go From Here

If you created a user’s home directory, you must share the directory so the user’s system can remotely mount it. See “How to Share a User’s Home Directory” on page 52 for detailed instructions.

If disk space is limited, you can set up a disk quota for the user in the file system containing the user’s home directory. See "Managing System Resources" in *System Administration Guide, Volume II* for information on setting disk quotas.

Example—Completed User Manager Add Window



The server where the home directory is located.

The path (on the system running User Manager) from which to copy user initialization files into the user's home directory. You can also specify an automounted path to user initialization files on another system.

See zation Files” on page 41 Initialization Files” on page 41 for information about creating customized user initialization files.

▼ How to Copy an Existing User Account

1. **(Optional) Fill out the user information data sheet on page 40.**
2. **Start User Manager from the Solstice Launcher and select the name service, if not done already.**
   See “How to Start User Manager” on page 46 for more information.
3. **Select a user account entry to copy from the main window.**
4. **Choose Copy from the Edit menu.**
   The Copy window is displayed with the following fields copied from the selected user account:
   - Primary Group
   - Secondary Groups
   - Comment
   - Login Shell
   - Path
   - Server
   - Skeleton Path.
5. **Fill in the fields in the Copy window.**

   If you need information to complete a field, click on the Help button to see field definitions for this window.
6. **Click on OK.**
   The list of user accounts displayed in the User Manager main window is updated to include the new user account.

Where to Go From Here

If you created a user’s home directory, you must share the directory so the user’s system can remotely mount it. See “How to Share a User’s Home Directory” on page 52 for detailed instructions.

If disk space is limited, you can set up a disk quota for the user in the file system containing user’s home directory. See "Managing System Resources" in *System Administration Guide, Volume II* for information on setting disk quotas.

How to Share a User's Home Directory

1. **Become root on the system that contains the home directory.**
2. **Verify that the `mountd` daemon is running.**

```
# ps -ef | grep mountd
```

The following line is displayed if the `mountd` daemon is running.

```
/usr/lib/nfs/mountd
```

3. **If the `mountd` daemon is not running, start it.**

```
# /etc/init.d/nfs.server start
```

4. **List the file systems that are shared on the system.**

```
# share
```

5. **Determine your next step based on whether the file system containing the user's home directory is already shared.**

If the File System Containing the User's Home Directory Is ...	Then ...
Already shared	Go to "Verification—Sharing a User's Home Directory" on page 53.
Not shared	Go to Step 6.

6. **Edit the `/etc/dfs/dfstab` file and add the following line.**

```
share -F nfs /file-system
```

In this entry,

file-system	Is the file system containing the user’s home directory that you need to share. By convention, the file system is `/export/home`.

7. **Share the file systems listed in the `/etc/dfs/dfstab` file.**

```
# shareall -F nfs
```

This command executes all the `share` commands in the `/etc/dfs/dfstab` file, so you do not have to wait to reboot the system.

Verification—Sharing a User’s Home Directory

If you selected the `AutoHome Setup` field when creating the user account (enabled the automounting of the home directory), log in to a system as the new user to make sure that the user’s home directory is available. Otherwise, you have to manually mount the user’s home directory and then log in to see if it’s available.

Where to Go From Here

If you did not select the `AutoHome Setup` field when creating the user account (did not enable the automounting of the home directory) and the user’s home directory is not located on the user’s system, you have to mount the user’s home directory from the system where it is located. See “How to Mount a User’s Home Directory” on page 55 for detailed instructions.

Example—Sharing a User's Home Directory

```
# ps -ef | grep mountd
# /etc/init.d/nfs.server start
# share
# vi /etc/dfs/dfstab

The line share -F nfs /export/home is added.

# shareall -F nfs
```

▼ How to Mount a User's Home Directory

1. **Make sure that the user's home directory is shared. See "How to Share a User's Home Directory" on page 52 for more information.**

2. **Log in as root on the user's system.**

3. **Edit the `/etc/vfstab` file and create an entry for the user's home directory.**

```
system-name:/export/home/user-name - /export/home/user-name nfs - yes rw,intr
```

In this entry,

system-name	Is the name of the system where the home directory is located.
`/export/home/`*user-name*	Is the name of the user's home directory that will be shared. By convention, `/export/home` contains user's home directories, however, this could be a different file system.
`-`	Are required placeholders in the entry.
`/export/home/`*user-name*	Is the name of the directory where the user's home directory will be mounted.

See the Chapter , "Mounting and Unmounting File Systems," for more information about adding an entry to the `/etc/vfstab` file.

4. **Create the mount point for the user's home directory.**

```
# mkdir -p /export/home/user-name
```

5. Mount the user's home directory.

```
# mountall
```

All entries in the current `vfstab` file (whose `mount at boot` fields are set to `yes`) are mounted.

Verification—Mounting a User's Home Directory

Use the `mount` command to verify that the home directory is mounted.

Example—Mounting a User's Home Directory

```
# vi /etc/vfstab

The line venus:/export/home/ripley - /export/home/ripley nfs - yes rw,intr is added.

# mkdir -p /export/home/ripley
# mountall
# mount
/ on /dev/dsk/c0t2d0s0 read/write/setuid on Thu Nov 17 10:40:42 1994
/usr on /dev/dsk/c0t2d0s6 read/write/setuid on Thu Nov 17 10:40:42 1994
/proc on /proc read/write/setuid on Thu Nov 17 10:40:42 1994
/dev/fd on fd read/write/setuid on Thu Nov 17 10:40:42 1994
/tmp on swap read/write on Thu Nov 17 10:40:46 1994
/export/home/ripley on venus:/export/home/ripley /read/write/remote on Thu Nov 17 10:40:46
1994
#
```

Maintaining User Accounts

Table 2-2 Task Map: Maintaining User Accounts

Activity	Description	For Instructions, Go To	
Modify a Group	Modify a group's name or the users in a group by choosing Modify from the Group Manager's Edit menu.	▼ How to Modify a Group	page 58
Delete a Group	Delete a group by choosing Delete from the Group Manager's Edit menu.	▼ How to Delete a Group	page 59
Modify a User Account	If a user account needs to be changed, modify the user account by choosing Modify from the User Manager's Edit menu.	▼ How to Modify a User Account	page 60
	Disable a User Account If you want to temporarily disable a user account, lock the user account from the Password menu in the Modify window.	▼ How to Disable a User Account	page 62
	Change a User's Password If you want change a user's password, use the Password menu in the Modify window.	▼ How to Change a User's Password	page 64
	Change Password Aging for a User Account If you want to force users to change their passwords periodically, change the password aging fields for a user in the Modify window (Account Security category).	▼ How to Change Password Aging for a User Account	page 66
Delete a User Account	Delete a user account by choosing Delete from the User Manager's Edit menu.	▼ How to Delete a User Account	page 68

▼ How to Modify a Group

1. **Start Group Manager from the Solstice Launcher and select the name service, if not done already.**
   See "How to Start Group Manager" on page 43 for more information.

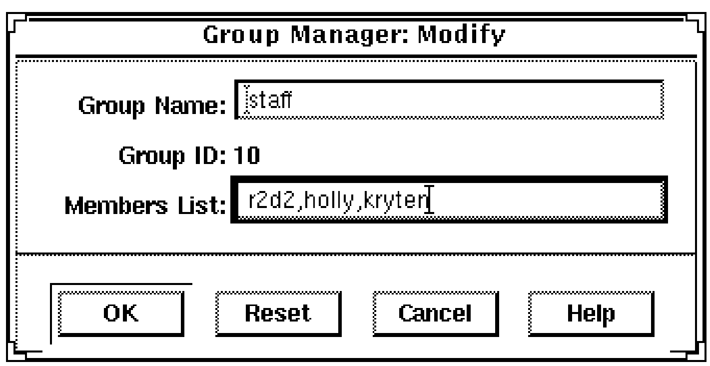
2. **Select the group entry to modify from the Group Manager main window.**

3. **Choose Modify from the Edit menu.**
   The Modify window is displayed containing the selected group entry.

4. **Either modify the group's name or the users in the group.**
   User names must be separated by commas. If you need information to complete a field, click on the Help button to see field definitions for this window.

5. **Click on OK.**
   The group information displayed in the main window is updated.

Example—Completed Group Manager Modify Window



How to Delete a Group

1. **Start Group Manager from the Solstice Launcher and select the name service, if not done already.**
   See "How to Start Group Manager" on page 43 for more information.

2. **Select the group entry you want to delete from the Group Manager main window.**

3. **Choose Delete from the Edit menu.**
   A window is displayed asking you to confirm the deletion.

4. **Click on OK.**
   The group entry is deleted from the Group Manager main window.

▼ How to Modify a User Account

1. **Start User Manager from the Solstice Launcher and select the name service, if not done already.**
   See "How to Start User Manager" on page 46 for more information.

2. **Select the user account entry to modify from the User Manager main window.**

3. **Choose Modify from the Edit menu.**
   The Modify window is displayed containing the selected user account entry.

4. **Modify the user account.**
   If you need information to complete a field, click on the Help button to see field definitions for this window.

   You can change any of the Account Security fields, which includes changing a password or changing password aging. See the following tasks for detailed step-by-step instructions:
   - "How to Disable a User Account" on page 62
   - "How to Change a User's Password" on page 64
   - "How to Change Password Aging for a User Account" on page 66

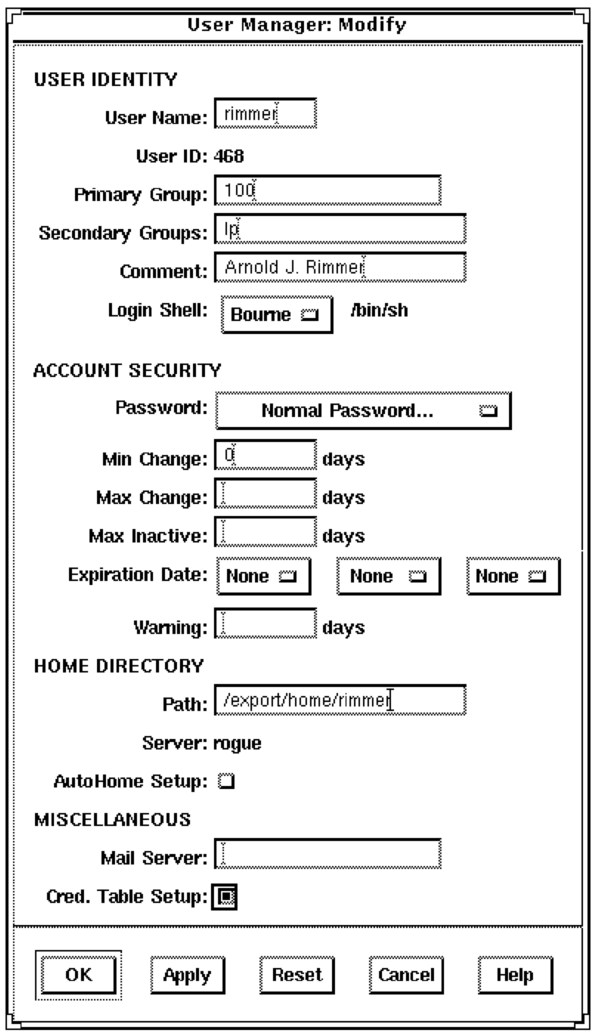
5. **Click on OK.**

Verification—Modifying a User Account

Double-click on the modified user account entry in the User Manager main window to verify that the modifications were made. Click on Cancel to close the window without making any modifications.

Example—Completed User Manager Modify Window

The following example adds the `lp` secondary group membership to the `rimmer` user account.



How to Disable a User Account

Note – You can enable the user account by changing the password status to `Normal Password` or `Cleared until first login`.

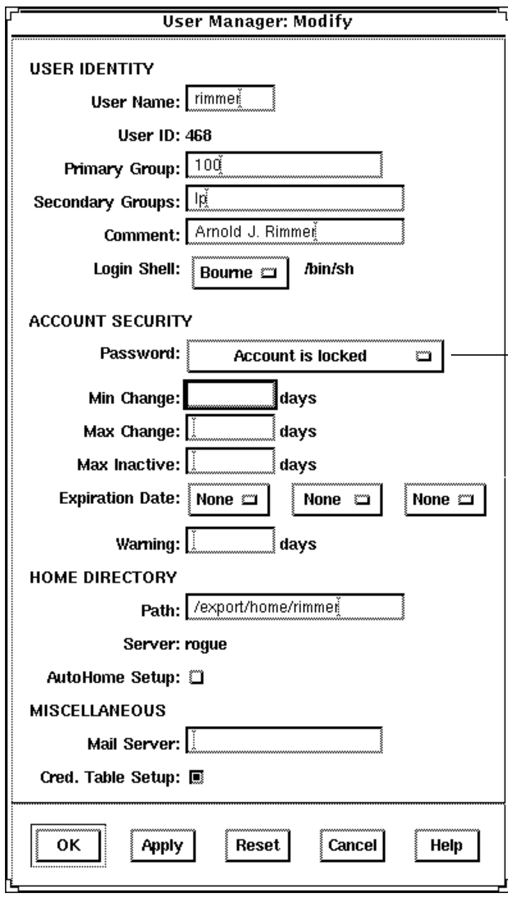
---

1. **Start User Manager from the Solstice Launcher and select the name service, if not done already.**
   See "How to Start User Manager" on page 46 for more information.
2. **Select the user account entry to be disabled.**
3. **Choose Modify from the Edit menu.**
   The Modify window is displayed containing the selected user account entry.
4. **Choose Account is Locked from the Password menu.**
   This selects the locked password status, which disables the user account.
5. **Click on OK.**

Verification—Disabling a User Account

Verify that you have disabled the user account by attempting to log in with the disabled user account.

Example—Disabling a User Account



Choosing this option disables the user account.

▼ How to Change a User's Password

1. **Start User Manager from the Solstice Launcher and select the name service, if not done already.**
   See "How to Start User Manager" on page 46 for more information.

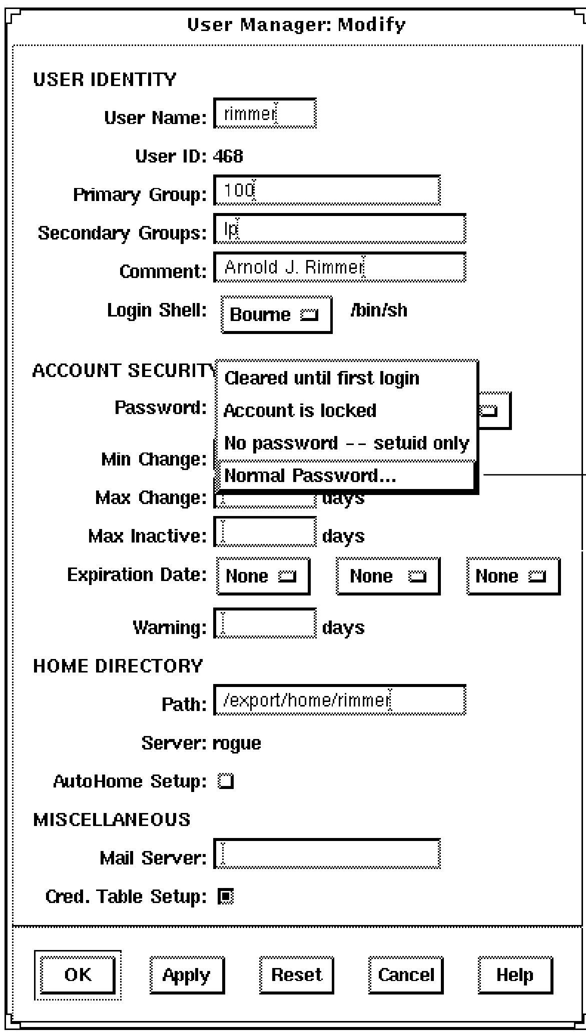
2. **Select the user account entry that needs the password changed.**

3. **Choose Modify from the Edit menu.**
   The Modify window is displayed containing the selected user account entry.

4. **Choose Cleared until first login or Normal password from the Password menu.**

5. **Click on OK.**

Example—Changing a User's Password



Choosing this option prompts you to create a new password for the user account.

How to Change Password Aging for a User Account

Note – Password aging is not supported for NIS. When you select NIS in User Manager, the password aging fields are not available.

1. **Start User Manager from the Solstice Launcher and select the name service, if not done already.**
   See "How to Start User Manager" on page 46 for more information.

2. **Select the user account entry that needs password aging changed.**

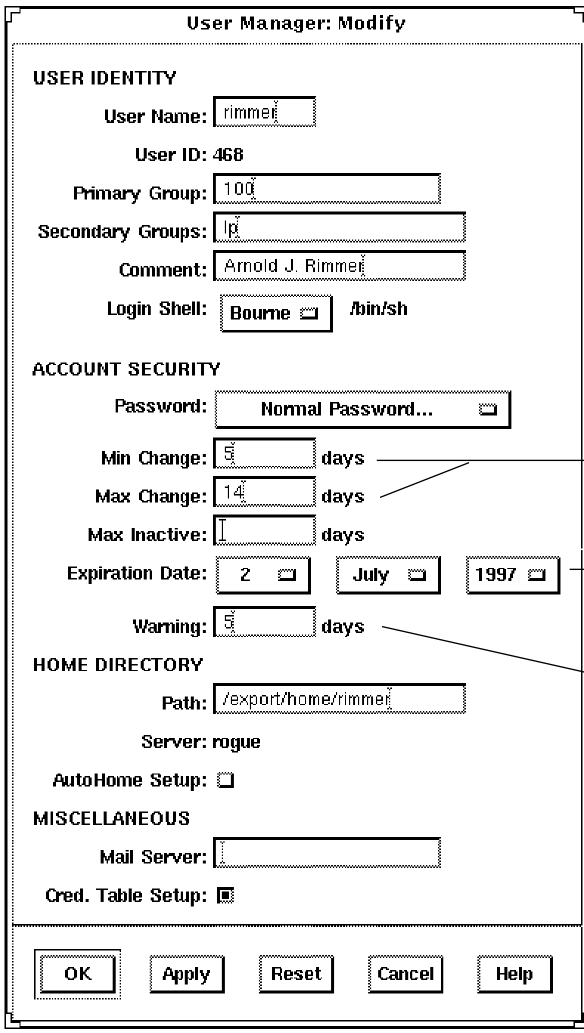
3. **Choose Modify from the Edit menu.**
   The Modify window is displayed containing the selected user account entry.

4. **Change the following fields that affect password aging:**
   - Min Change
   - Max Change
   - Max Inactive
   - Expiration Date
   - Warning

   If you need information about the password aging fields that are part of the Account Security category, click on the Help button.

5. **Click on OK.**

Example—Changing Password Aging for a User Account



The user must keep a new password for at least 5 days and the user must change the password every 14 days.

The user account will be disabled on July 2, 1997.

The user will be warned 5 days before the account is disabled.

▼ How to Delete a User Account

1. **Start User Manager from the Solstice Launcher and select the name service, if not done already.**
   See "How to Start User Manager" on page 46 for more information.

2. **Select the user account entry to remove from the main window.**

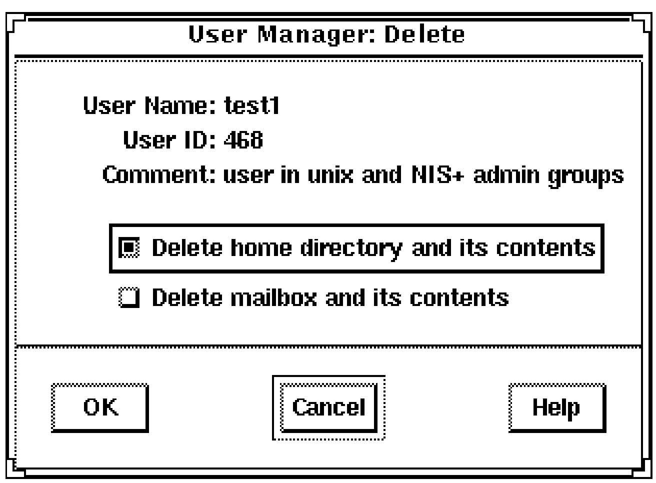
3. **Choose Delete from the Edit menu.**
   The Delete window is displayed to confirm the removal of the user account.

4. **(Optional) Click on the check box to delete the user's home directory and its contents.**

5. **(Optional) Click on the check box to delete the user's mailbox and its contents.**

6. **Click on OK when you are ready to delete the user account.**
   The user account entry is deleted from the User Manager main window.

Example—User Manager Delete Window



Part 2 — Managing Server and Client Support

This part provides instructions for managing server and client support in the Solaris environment.

Overview of Managing Server and Client Support 3

This chapter describes managing server and client support on a network, and it provides overview information about each system configuration (referred to as a *system type*) supported in the Solaris environment. This chapter also includes guidelines for selecting the appropriate system type to meet your needs.

This is a list of the overview information in this chapter.

For step-by-step instructions about how to add and maintain server and client support, see Chapter 4, "Adding and Maintaining Server and Client Support."

What Are Servers and Clients

Systems on the network can usually be described as one of the following:

- **Server –** A system that provides services to other systems in its network. There are file servers, boot servers, database servers, license servers, print servers, installation servers, and even servers for particular applications. This chapter uses the term server to mean a system that provides file systems and installation software for other systems on the network.
- **Client –** A system that uses remote services from a server. Some clients have limited disk storage capacity, or perhaps none at all, and they have to rely on remote file systems from a server to function. Diskless and dataless clients are examples of this type of client.

  Other clients may use remote services (such as installation software) from a server, but they don't rely on a server to function. A standalone system, which has its own hard disk containing the root (`/`), `/usr`, and `/home` file systems and swap space, is a good example of this type of client.

What Does Support Mean

Providing support for a system means providing software and services to help another system function. Support can include:

- Making a system known to the network (i.e., host name and ethernet address information)
- Providing installation services to remotely boot and install a system
- Providing operating system (OS) services to a system with limited or no disk space

Overview of System Types

System types are basically defined by how they access the root (/) and /usr file systems, including the swap area. For example, standalone and server systems mount these file systems from a local disk, while diskless and dataless clients mount the file systems remotely, relying on servers to provide these services. Table 3-1 lists these and other differences for each system type.

Table 3-1 System Type Overview

System Type	Local File Systems	Local Swap?	Remote File Systems	Network Use	Relative Performance
Server	root (/) /usr /home /opt /export/home /export/root	Yes	– *none* –	medium	high
Standalone System	root (/) /usr /export/home	Yes	– *none* –	low	high
Dataless Client	root (/)	Yes	/usr /home	medium	medium
Diskless Client	– *none* –	No	root (/) swap /usr /home	high	low
AutoClient™ System	cached root (/) cached /usr	Yes	/var /home	low	high

Servers

A server system has the following file systems:

- The root (/) and /usr file systems, plus swap space
- The /export, /export/swap, and /export/home file systems, which support client systems and provide home directories for users.
- The /opt directory or file system for storing application software.

Servers can also contain the following software to support other systems:

- Operating system (OS) services for diskless and dataless clients that want to run a different release or are a different platform than the server.
- Solaris CD image and boot software for networked systems to perform remote installations.
- JumpStart™ directory for networked systems to perform custom JumpStart installations.

Standalone Systems

A *networked standalone system* can share information with other systems in the network, but it could continue to function if detached from the network.

A standalone system can function autonomously because it has its own hard disk containing the root (/), /usr, and /home file systems and swap space. The standalone system thus has local access to operating system software, executables, virtual memory space, and user-created files.

Note – A standalone system requires sufficient disk space to hold the four necessary file systems.

A *non-networked standalone system* is a standalone system with all the characteristics listed above except that is not connected to a network.

Diskless Clients

A *diskless client* has no disk and depends on a server for all its software and storage area. A diskless client remotely mounts its root (/), /usr, and /home file systems from a server.

A diskless client generates significant network traffic due to its continual need to procure operating system software and virtual memory space from across the network. A diskless client cannot operate if it is detached from the network or if its server malfunctions.

Dataless Clients

A *dataless client* has local storage for its root (`/`) file system and swap space. The dataless client cannot function if detached from the network, because its executables (`/usr`) and user files (`/home`) are located across the network on the disk of a server.

Note – SunSoft plans to remove support for dataless clients after Solaris 2.5. You can add this system type now using Host Manager, but in future releases of the Solaris operating environment you will need to choose a different type.

A dataless client places far less demand on the server and the network than a diskless client does. Because dataless clients require less network access, a server can accommodate many more dataless clients than it can diskless clients.

Dataless clients are less expensive to manage than standalone systems. Also, the user files of all the dataless clients are stored centrally (on a server) and can be backed up and administered centrally.

However, if local security is an issue at your site, you need to weigh these conveniences against the relative lack of security of centralized files.

AutoClient Systems

An AutoClient system is nearly identical to a diskless client in terms of installation and administration. It has the following characteristics:

- Requires a 100-Mbyte local disk for swapping and for caching its individual root (/) file system and the /usr file system from a server

---

Note – The /var directory should not be cached on the AutoClient system. The /var directory is volatile, and therefore is not suitable for caching.

- Relies on a server to access other file systems and software applications
- Contains no permanent data, making it a field replaceable unit (FRU)

AutoClient systems use the Solstice AutoClient technology and must be added and maintained by using the AutoClient Manager.

Guidelines for Choosing System Types

Determining which system types are appropriate for your environment can be done by comparing each type based on the following characteristics:

- Centralized Administration
  - Can the system be treated as a field replaceable unit (FRU)? This means that a broken system can be quickly replaced with a new system without any lengthy backup/restore operations and no loss of system data.
  - Does the system need to be backed up? Large costs in terms of time and resources can be associated with backing up a large number of desktop systems.
  - Can the system's data be modified from a central server?
  - Can the system be installed from a centralized server, quickly and easily without handing the client system's hardware?
- Performance
  - Does this configuration perform well in desktop usage?
  - Does the addition of systems on a network affect the performance of other systems already on the network?
- Disk Usage
  - How much disk space is required to effectively deploy this configuration?

Table 3-2 describes how each system type scores in terms of each of these categories. A ranking of 1 is most efficient; a ranking of 4 is least efficient.

Table 3-2 Comparison of System Types

System Type	Centralized Administration	Performance	Disk Usage
Standalone System	4	1	4
Diskless Client	1	4	1
Dataless Client	3	3	2
AutoClient System	1	2	2

Tools for Managing Server and Client Support

In previous Solaris releases, you may have used Administration Tool to manage server and client support. In the Solaris 2.5 release, you must use the Solstice Host Manager tool, which offers ease of use and provides support for the following name services:

- NIS+ tables
- NIS maps
- Local `/etc` files

What You Can Do With Host Manager

Host Manager is a graphical user interface that enables you to add and maintain server and client support on a network. With a name service like NIS+, you can manage system information in a centralized manner so that important system information, such as host names, does not have to be duplicated on every system in the network.

Host Manager enables you to:

- Add and modify support
- Update system types
- Convert system types
- Add OS services
- Set up remote installation services

Add and Maintain Support

Host Manager enables you to add and modify support for the following Solaris system types:

- Diskless
- Dataless
- Standalone
- OS Server

Table 3-3 describes the server-client configurations that are supported by the Solstice AdminSuite 2.1 release of Host Manager.

Table 3-3 Supported Server-Client Configurations

			Client						
			SunOS 4.x	**Solaris 2.3**		**Solaris 2.4**		**Solaris 2.5**	
			SPARC	**x86**	**SPARC**	**x86**	**SPARC**	**x86**	**SPARC**
	Solaris 2.3	**x86**	Yes	-	-	Yes	Yes	Yes	Yes
		SPARC	Yes	-	Yes	Yes	Yes	Yes	Yes
Server	**Solaris 2.4**	**x86**	Yes	-	Yes	Yes	Yes	Yes	Yes
		SPARC	Yes	-	Yes	Yes	Yes	Yes	Yes
	Solaris 2.5	**x86**	Yes	-	Yes	Yes	Yes	Yes	Yes
		SPARC	Yes	-	Yes	Yes	Yes	Yes	Yes

Note – SunOS 4.x can only run on SPARC systems that are Sun4c and Sun4m platform groups.

Update System Types

Host Manager cannot recognize all previously added system types until you use the Update System Types option from the File menu. This option will probe client systems and identify their system types. Host Manager will mark systems as `generic` if they are not running the Solaris software or if they use local or loghost entries.

Note – Previously added systems running Solaris 2.5 must also have the Solstice AdminSuite software installed for Host Manager to update their system type.

The system type information is stored in the `bootparams` file in the local `/etc` files or a name service database. Host Manager will either modify an existing `bootparams` entry or add a new one such as the following for a Solaris standalone system named `mars`:

```
mars  boottype=:st
```

Convert System Types

Host Manager enables you to convert one system type to another. Currently, you can only convert:

- A generic system to a standalone system
- A standalone system to an OS server

You can add Solaris 2.x OS services during the standalone system to OS server conversion.

Add OS Services

A Solaris OS server is a server that provides operating system (OS) services to support diskless or dataless clients. By using Host Manager, you can add support for an OS server or convert a standalone system to an OS server.

By default, an OS server can support clients that are the same platform group and require the same Solaris release as the OS server. To support clients of a different platform group or clients that require a different Solaris release than the OS server, you must add the particular OS service to the OS server. You must have the appropriate Solaris CD image to add OS services.

For example, if you have a OS server running Solaris 2.5 and you want it to support diskless clients running Solaris 2.4, you must add Solaris 2.4 OS services to the OS server. You would also have to add services to a SPARC OS server if you want it to support x86 diskless clients running Solaris 2.4. In this example, both the server and client are running the same Solaris release, but they are different platform groups.

Note – Although Host Manager enables you to add support for diskless and dataless clients running the Sun4.x release, you cannot add SunOS 4.x OS services using Host Manager. You must use the `discover4x` and `install4x` commands to add OS services to an OS server, and then use Host Manager to add support for the SunOS 4.x client.

Set Up Remote Installation Services

Host Manager enables you to set up systems to provide Solaris 2.x installation services for other systems on the network. You can set up the following types of installation services on a system:

- **An install server –** A system on the network that provides a Solaris CD image (either from a CD-ROM drive or the copy on the hard disk) for other systems to install from.
- **A boot server –** A system that provides boot information to other systems on the network.
- **A profile server –** A system that contains Jumpstart files for systems to perform a custom JumpStart installation.

Note – A boot server and install server are typically the same system. However, if the system to be installed is on a different subnet than the install server, a boot server is required on that subnet.

What You Can't Do With Host Manager

Table 3-4 shows the limitations of Host Manager and their suggested workarounds.

Table 3-4 Host Manager Limitations and Workarounds

Limitation	Workaround
Host Manager cannot automatically recognize all previously added system types.	Use the Update System Type option from the File menu. This option will probe systems on the network and identify their system types.

Table 3-4 Host Manager Limitations and Workarounds *(Continued)*

Limitation	**Workaround**
Host Manager cannot add or maintain AutoClient systems.	Use the Solstice AutoClient Manager to create and maintain AutoClient systems. AutoClient Manager is provided with the Solstice AutoClient product.
Host Manager can't add SunOS 4.x services to an OS server.	Mount a SunOS 4.x CD image and add OS services by using the `discover4x` and `install4x` commands.
Host Manager can't provide remote installation services for SunOS 4.x systems.	Install SunOS 4.x systems from the local CD-ROM drive.
Host Manager does not enable you to add multiple host entries for a system with the same Ethernet address (needed for servers with multiple Ethernet interfaces, also known as multi-home hosts).	Use command-line equivalents to enter the host information into the name service.

Adding and Maintaining Server and Client Support 4

This chapter describes how to set up and maintain server and client support using the Solstice Host Manager.

This is a list of the step-by-step instructions in this chapter.

For overview information about managing server and client support, see Chapter 3, “Overview of Managing Server and Client Support.”

4

Adding Server and Client Support

Table 4-1 Task Map: Adding Server and Client Support

Activity	Description	For Instructions, Go To	
Update System Types	Optional. Make sure Host Manager recognizes all the previously added system types. This is usually a one-time task before using the AdminSuite 2.1 Host Manager for the first time.	▼ How to Update System Types	page 88
Set Defaults for Adding Support	Optional. Before you add support for several clients, set up defaults for the Add window by choosing Set Defaults from the Host Manager's Edit menu. Setting up defaults can increase the consistency and efficiency of adding support for systems.	▼ How to Set Defaults for Adding Support	page 88
Add Support for a Standalone System	Add support for a standalone system by choosing Add from the Host Manager's Edit menu. Once in the Add window, choose either Standalone System from the System Type menu.	▼ How to Add Support for a Standalone System or OS Server	page 89
Add Support for an OS Server	**Add Support for an OS Server** Add support for an OS server by choosing Add from the Host Manager's Edit menu. Once in the Add window, choose OS Server from the System Type menu.	▼ How to Add Support for a Standalone System or OS Server	page 89
	Add Support for an OS Server by Converting a Standalone System to an OS Server Convert a standalone system to an OS server by choosing Convert from the Host Manager's Edit menu. You can add Solaris 2.x OS services during the conversion.	▼ How to Convert a Standalone System to an OS Server	page 92

Table 4-1 Task Map: Adding Server and Client Support *(Continued)*

Activity	Description	For Instructions, Go To	
Add OS Services to an OS Server	**Add SunOS 4.x OS Services** If you need to add support for SunOS 4.x diskless or dataless clients, an OS server must have the appropriate SunOS 4.x services added.	▼ How to Add SunOS 4.x OS Services to an OS Server	page 94
	Add Solaris 2.x OS Services If you need to add support for Solaris 2.x diskless or dataless clients, an OS server must have the appropriate Solaris 2.x services added.	▼ How to Add Solaris 2.x OS Services to an OS Server	page 99
Add Support for Diskless or Dataless Clients	**Add Support for a Diskless Client** Add support for a diskless client by choosing Add from the Host Manager's Edit menu. Once in the Add window, choose diskless client from the System Type menu.	▼ How to Add Support for a Diskless Client	page 101
	Add Support for a Dataless Client Add support for a dataless client by choosing Add from the Host Manager's Edit menu. Once in the Add window, choose dataless client from the System Type menu.	▼ How to Add Support for a Dataless Client	page 104

▼ How to Start Solstice Host Manager

1. **Verify that the following prerequisites are met. To use Host Manager, you must have:**
   - Solstice AdminSuite installed.
   - A bit-mapped display monitor. The Solstice AdminSuite software can be used only on a system with a console that is a bit-mapped screen such as a standard display monitor that comes with a Sun workstation.

     If you want to perform administration tasks on a system with an ASCII terminal as the console, use Solaris commands instead.
   - OpenWindows software. Start this software with the following command:

```
$ /usr/openwin/bin/openwin
```

   - Root privilege or membership in the `sysadmin` group (group 14) and the required access privileges for managing the NIS or NIS+ database.

Note – If your name service is NIS+, you must be a member of the NIS+ admin group.

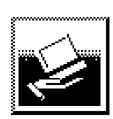
---

2. **Start the Solstice Launcher.**

```
$ solstice &
```

The Solstice Launcher is displayed.

3. **Click on the Host Manager icon.**



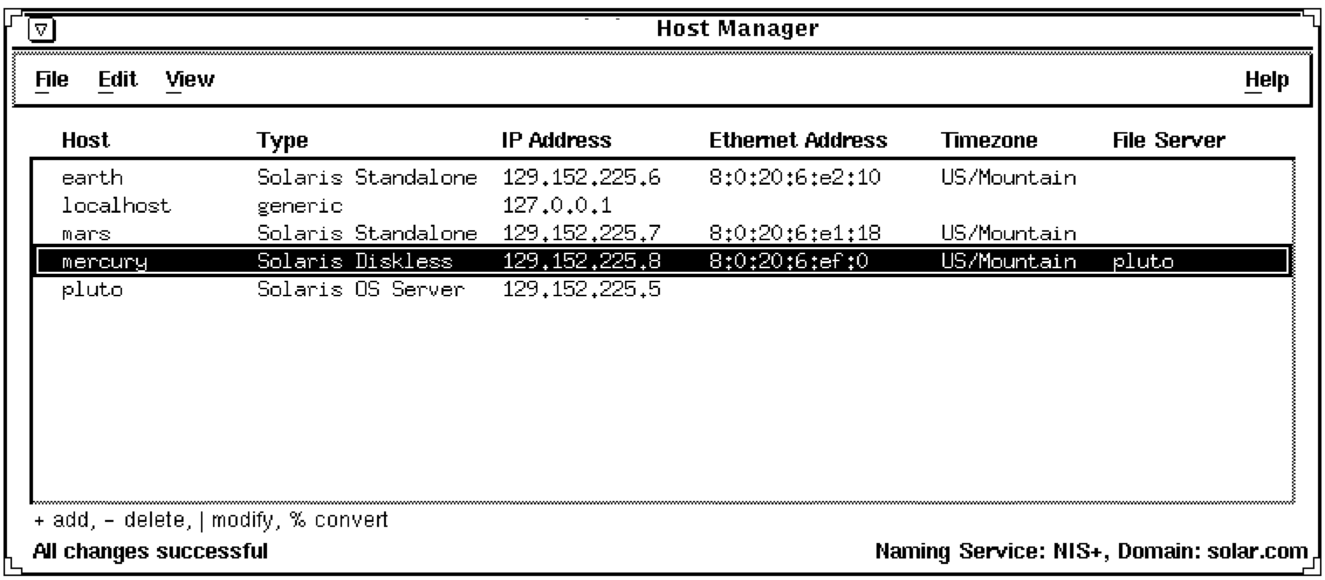
The Load window is displayed.

4. **Select the name service used in your network.**

5. **Check that the domain or host name is correct.**
   If not, type the domain or host name you need to access.

6. **Click on OK.**
   The Solstice Host Manager main window is displayed.

Example—Host Manager Main Window



How to Update System Types

This procedure converts system types that are identified as generic.

1. **Start Solstice Host Manager from the Solstice Launcher and select the name service, if not done already.**
   See "How to Start Solstice Host Manager" on page 86 for more information.

2. **Select Update System Types from the File menu on the Solstice Host Manager Window.**
   The Update Systems Types window is displayed after Host Manager searches for missing system types.

3. **Click on Update to identify unknown system types.**
   Host Manager will attempt to contact each host defined as a generic system type to determine if it is one of the following Solaris system types:

   - Standalone
   - OS Server
   - Dataless
   - Diskless

   If the system can be identified, Host Manager will update the host entry in the scrolling list with the new system type.

How to Set Defaults for Adding Support

1. **Start Host Manager from the Solstice Launcher and select the name service, if not done already.**
   See "How to Start Solstice Host Manager" on page 86 for more information.

2. **Choose Set Defaults from the Edit menu.**
   The Set Add Defaults window is displayed.

3. **Fill in the Set Add Defaults window.**
   The defaults you select will be the initial defaults values in the Add window. If you need information to complete a field, click on the Help button to see field definitions for this window.

4. **Click on OK.**

▼ How to Add Support for a Standalone System or OS Server

The high-level steps in this procedure are:

- Adding system information about the system
- (Optional) Setting up remote install capabilities for the system
- (Optional) Installing the system

1. **Start Solstice Host Manager from the Solstice Launcher and select the name service, if not done already.**
   See "How to Start Solstice Host Manager" on page 86 for more information.

2. **Select Add from the Edit menu on the Solstice Host Manager Window.**
   The Add window is displayed.

3. **Fill in the system information, selecting Solaris Standalone or OS Server as the system type.**
   The system information includes all the fields from the host name through the timezone. If you need information to complete a field, click on the Help button to see field definitions for this window.

4. **If you want to set up remote install capabilities for the system, continue to Step 5. If not, skip to Step 10.**

5. **Click on Enable Remote Install.**

6. **Select an Install Server.**
   The Install Server defaults to the current host. Select Other from the Install Server menu to specify another host as the install server.

7. **Click on Set Path to identify the path to the Solaris CD image on the install server.**

If You Are Using ...	And ...	Then Enter the Path ...
The Solaris CD as the Solaris CD image	The Solaris CD is managed by Volume Management	`/cdrom/cdrom0/s0` or `/cdrom/cdrom0/s2`
	The Solaris CD is not managed by Volume Management	Where you mounted the Solaris CD
A copy of the Solaris CD on the Install Server's hard disk (by using `setup_install_server`)		Where you specified `setup_install_server` to copy the Solaris CD

8. **Select the system's architecture type and OS release from the OS Release menu.**
   The architecture type must match the system's architecture and the OS release should match the Solaris release you want to remotely install on the system.

9. **If necessary, specify a boot server or profile server.**
   To specify another server other than the default, select Other from the menu. Select a Profile Server from the Profile Server pull-down menu. You must also specify a path to the boot software on the boot server or the custom JumpStart directory on the profile server.

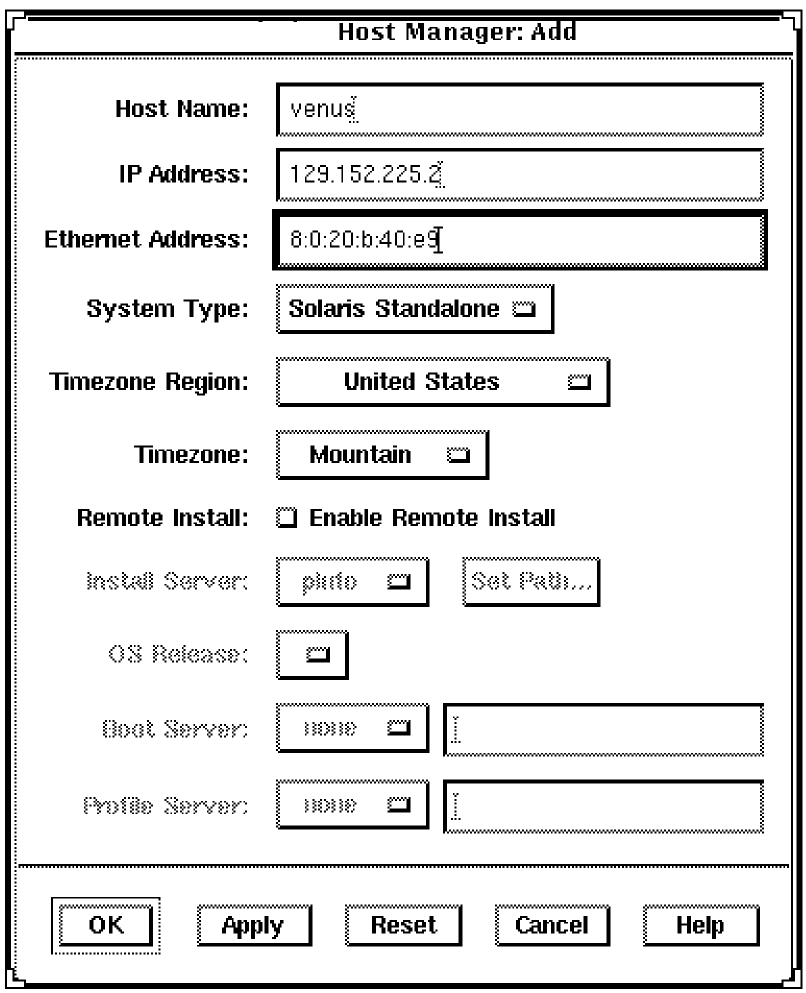
10. **Click on OK on the Add window.**

11. **Select Save Changes from the File menu to add support for the standalone system or OS server.**
    The standalone system or OS server is displayed in the Host Manager main window.

12. **(Optional) Boot and install the standalone system or OS server.**
    For more information about booting and installing, see *SPARC: Installing Solaris Software* for SPARC systems or *x86: Installing Solaris Software* for x86 systems.

Note – If you are installing an OS server, you must allocate space in `/export` and `/export/swap` for the desired number of clients.

Example—Completed Host Manager Add Window for a Standalone System



Where to Go From Here

If you want to add OS services after you install an OS server, see "How to Add Solaris 2.x OS Services to an OS Server" on page 99.

▼ How to Convert a Standalone System to an OS Server

1. **Start Solstice Host Manager from the Solstice Launcher and select the name service, if not done already.**
   See "How to Start Solstice Host Manager" on page 86 for more information.

2. **Select a standalone system host entry from Solstice Host Manager's main window.**

3. **Select Convert to OS Server from the Edit Menu.**
   The Convert window is displayed, and the selected standalone system is displayed in the host name field.

4. **Click on the Add button in the OS Services window to add services.**

5. **Click on Set Path to identify the path to the Solaris CD image from which to add the client services.**
   The Install Server defaults to the current host. Select Other from the Install Server menu to specify another host as the install server.

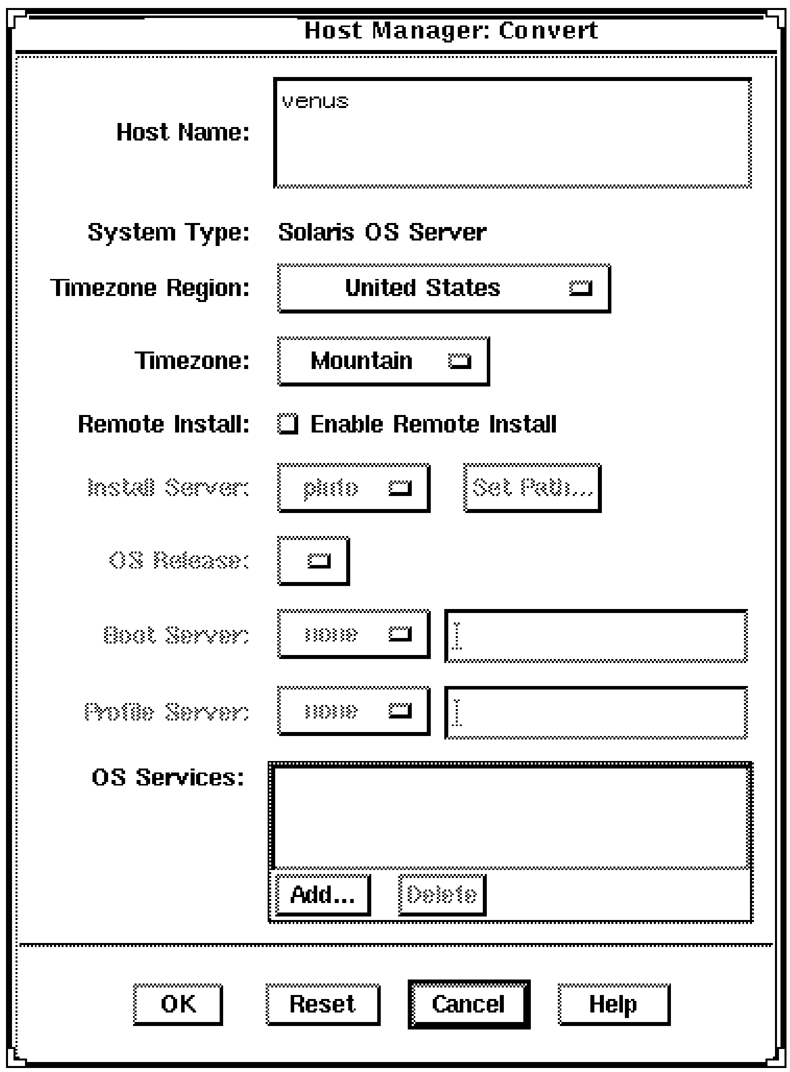
If You Are Using ...	And ...	Then Enter the Path ...
The Solaris CD as the Solaris CD image	The Solaris CD is managed by Volume Management	`/cdrom/cdrom0/s0` or `/cdrom/cdrom0/s2`
	The Solaris CD is not managed by Volume Management	Where you mounted the Solaris CD
A copy of the Solaris CD on the Install Server's hard disk (by using `setup_install_server`)		Where you specified `setup_install_server` to copy the Solaris CD

6. **Specify the type of services you want to add and click on Add.**

7. **Click on OK on the Convert window.**

8. **Select Save Changes from the File menu to convert the standalone system to an OS Server.**
   The converted standalone systems are displayed as OS servers in the Host Manager main window.

Example—Completed Host Manager Convert Window



▼ How to Add SunOS 4.x OS Services to an OS Server

The high-level steps in this procedure are:

- Determining the OS server to which you want to add SunOS 4.x OS services for diskless and dataless clients
- Verifying that the SunOS 4.x heterogeneous install software is installed
- Installing the required SunOS 4.x release software (`install4x` command)

1. **Log in as root on the OS server to which you want to add SunOS 4.x OS services for diskless and dataless clients.**

2. **Verify that the SunOS 4.x heterogeneous install software is installed.**

```
# pkginfo SUNWhinst
```

   If the package information is displayed, proceed to Step 7. If not, go to the next step.

3. **Insert the Solaris 2.x CD into your CD-ROM drive.**
   This step assumes that your system is running Volume Management.

Note – Use the Solaris 2.x CD that matches the Solaris 2.x release that is installed on the OS server. For example, use the Solaris 2.4 11/94 CD if the OS server is running the Solaris 2.4 11/94 release.

   To access the SunOS 4.x CD from another system running either the Solaris 2.x release or the SunOS 4.x release, see “Setting Up the CD-ROM Drive for `install4x`” in the *Solaris 1.x to 2.x Transition Guide*.

4. **Change directory to the location of the software.**

```
# cd /cdrom/cdrom0/s0/Solaris_2.x
```

5. **Install the SunOS 4.x heterogeneous install software.**

```
# pkgadd -d ‘pwd‘ SUNWhinst
```

6. **Eject the Solaris 2.x CD.**

```
# cd
# eject cd
```

7. **Insert the SunOS 4.x release CD into your CD-ROM drive.**
   This step assumes that your system is running Volume Management, and the CD-ROM drive is directly attached to the server. Volume Management automatically mounts the CD directory on `/cdrom/volume1/s0`.

8. **Start the SunOS 4.x release software installation program.**

```
# /usr/sbin/install4x -m /cdrom/volume1/s0 -e /export
```

   The main menu is displayed.

```
                        *** 4.1* Install Main Menu ***

      Choose an Architecture (then select modules to load):

                                          Modules
                                     Loaded    Selected
      [a] sun4.sun4c.sunos.4.1.2        0          0
      [b] sun4.sun4.sunos.4.1.2         0          0
      [c] sun4.sun4m.sunos.4.1.2        0          0


    or begin the loading process for all selected modules:

      [L] Load selected modules

    or abort without loading any modules:

      [Q] Quit without loading                            +-------------------+
                                                          | Disk Usage:       |
    Type any bracketed letter to select that function.    |       0K Selected |
                                                          |   53634K Free     |
    Type ? for help.                                      +-------------------+
```

9. **On the main menu, specify the architecture you want to support by typing the associated character that is shown in brackets.**
The Module Selection menu is displayed.

```
Select sun4.sun4c.sunos.4.1.2 modules:
  +[a] R proto root...............240K  |  [o]   User_Diag................6352K
  +[b] R usr....................26240K  |  [p]   Manual...................7456K
  +[c] R Kvm.....................4832K  | +[q] D TLI........................48K
  +[d] R Install..................936K  |  [r] D RFS.......................912K
   [e] D Networking..............1040K  |  [s] D Debugging................2928K
   [f] D System_V................4008K  |  [t]   SunView_Programmers......1840K
   [g] D Sys.....................5288K  |  [u]   Shlib_Custom.............1376K
   [h] C SunView_Users...........2664K  |  [v]   Graphics.................1784K
   [i]   SunView_Demo.............512K  | +[w]   uucp......................608K
  +[j]   Text.....................712K  | +[x]   Games....................3136K
   [k]   Demo....................4264K  |  [y]   Versatec.................5960K
   [l] C OpenWindows_Users.......25936K  |  [z]   Security..................312K
   [m] C OpenWindows_Demo.........4288K  |  [A]   OpenWindows_Programmers.10200K
   [n] C OpenWindows_Fonts........7840K  |

Module     + = already loaded       R = Required     C= Common
Legend:   ** = selected for loading  D = Desirable    Others are optional

Select [a-A] or a Quick-Pick Option:                       +-------------------+
    [1] All Required Modules     [4] All Optional Modules  | Disk Usage:       |
    [2] All Desirable  Modules   [5] All Modules           |        0K Selected |
    [3] All Common Modules                                 |   53634K Free     |
 or [D] (done) to return to the main screen                +-------------------+
```

10. **Select modules to load by typing the associated character that is shown in brackets.**
The Module Selection screen readily enables you to pick groups of modules to be loaded. When you enter a 1, it marks all required modules for loading. When you enter a 2, it marks all recommended modules. When you enter a 3, it marks all commonly loaded modules. When you enter a 4, it marks all optional modules. When you enter a 5, it marks all modules shown on the Module Selection screen.

11. Return to the main menu by typing `D`.
The main menu is displayed.

```
                    *** 4.1* Install Main Menu ***

     Choose an Architecture (then select modules to load):

                                          Modules
                                     Loaded    Selected
     [a] sun4.sun4c.sunos.4.1.2        0          4
     [b] sun4.sun4.sunos.4.1.2         0          3
     [c] sun4.sun4m.sunos.4.1.2        0          1


   or begin the loading process for all selected modules:

     [L] Load selected modules

   or abort without loading any modules:

     [Q] Quit without loading                               +--------------------+
                                                            | Disk Usage:        |
   Type any bracketed letter to select that function.       |       0K Selected  |
                                                            |   53634K Free      |
   Type ? for help.                                         +--------------------+
```

12. Type `L` to install the software.
The modules you previously selected are installed.

```
Installing module ‘proto root’ [size: 248K]
        in directory /export/exec/proto.root.sunos.4.1.2 ...

Updating server databases ...

Press any key to continue:
```

13. After the modules are installed, press any key to return to the main menu.
The loaded modules are displayed in the main menu.

14. If you want to add support for other architectures, repeat Step 9 through Step 12. Otherwise, type `Q` to exit.

Where to Go From Here

If you want to add SunOS 4.x support for a diskless client, see "How to Add Support for a Diskless Client" on page 101. If you want to add SunOS 4.x support for a dataless client, see "How to Add Support for a Dataless Client" on page 104.

▼ How to Add Solaris 2.x OS Services to an OS Server

1. **Start Solstice Host Manager from the Solstice Launcher and select the name service, if not done already.**
   See “How to Start Solstice Host Manager” on page 86 for more information.

2. **Select an OS server to modify from the Host Manager main window.**

3. **Select Modify from the Edit menu on the Solstice Host Manager Window.**
   The Modify window is displayed.

4. **Click on the Add button in the OS Services window to add services.**

5. **Click on Set Path to identify the path to the Solaris CD image from which to add the client services.**
   The Install Server defaults to the current host. Select Other from the Install Server menu to specify another host as the install server.

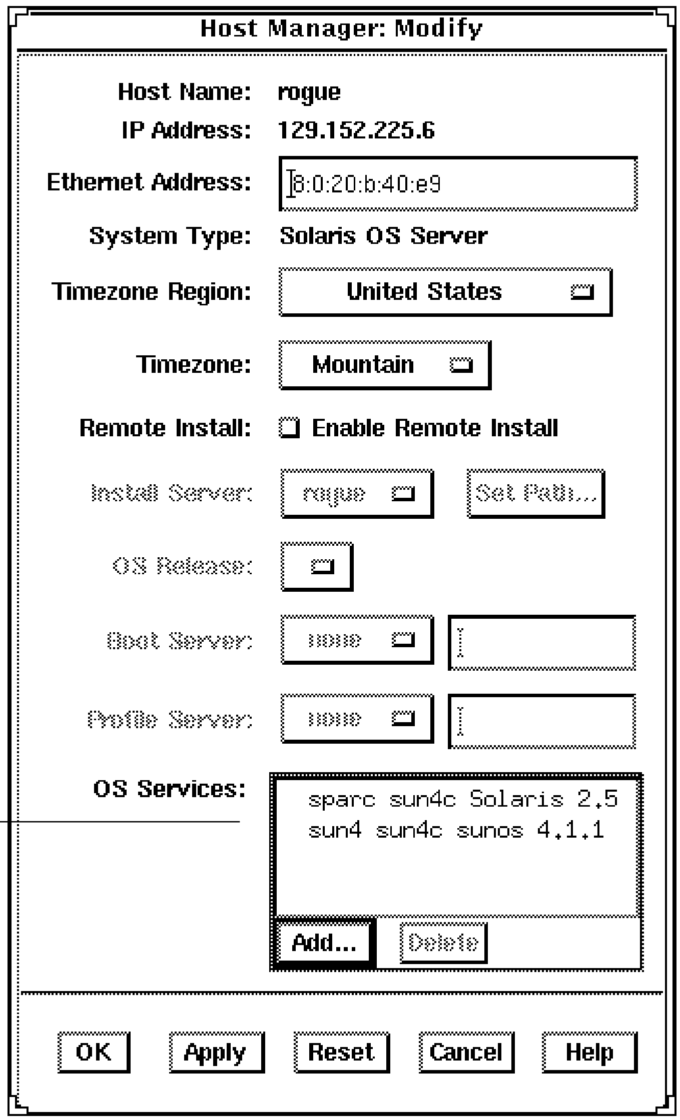
If You Are Using ...	And ...	Then Enter the Path ...
The Solaris CD as the Solaris CD image	The Solaris CD is managed by Volume Management	`/cdrom/cdrom0/s0` or `/cdrom/cdrom0/s2`
	The Solaris CD is not managed by Volume Management	Where you mounted the Solaris CD
A copy of the Solaris CD on the Install Server’s hard disk (by using `setup_install_server`)		Where you specified `setup_install_server` to copy the Solaris CD

6. **Specify the type of services you want to add and click on Add.**

7. **Click on OK on the Add window.**

8. **Select Save Changes from the File menu to add services.**

Example—Completed Add OS Services Window



List of OS services for each platform, platform group, and OS Release

▼ How to Add Support for a Diskless Client

The high-level steps in this procedure are:

- Adding system information about the diskless client
- Selecting OS services for the diskless client
- Booting the diskless client

Note – Before starting this procedure, make sure the system providing the services (the file server) has already been configured as an OS server with the `/export` and `/export/swap` file systems already created.

---

1. **Start Solstice Host Manager from the Solstice Launcher and select the name service, if not done already.**
   See "How to Start Solstice Host Manager" on page 86 for more information.
2. **Select Add from the Edit menu on the Solstice Host Manager main window.**
   The Add window is displayed.
3. **Fill in the system information, selecting Solaris Diskless as the system type.**
   The system information includes all the fields from the host name through the time zone. If you need information to complete a field, click on the Help button to see field definitions for this window.
4. **Select a File Server.**
   The File Server defaults to the current host. Select Other from the Install Server menu to specify another host as the install server.
5. **Select the client's architecture type and the OS release from the OS Release menu.**
   The architecture type must match the diskless client's architecture and the OS release should match the Solaris release you want the diskless client to run.
6. **Identify the system's root path, swap path, and swap size.**
7. **Click on OK on the Add window.**

8. **Select Save Changes from the File menu to add support for the diskless client.**
   The diskless client is displayed in the Host Manager main window. It takes several minutes to add the diskless client support, particularly to create the system's root and swap areas.

9. **(If necessary) Start the appropriate client daemons on the file server after the system information has been added successfully.**

```
# /etc/init.d/nfs.server start
```

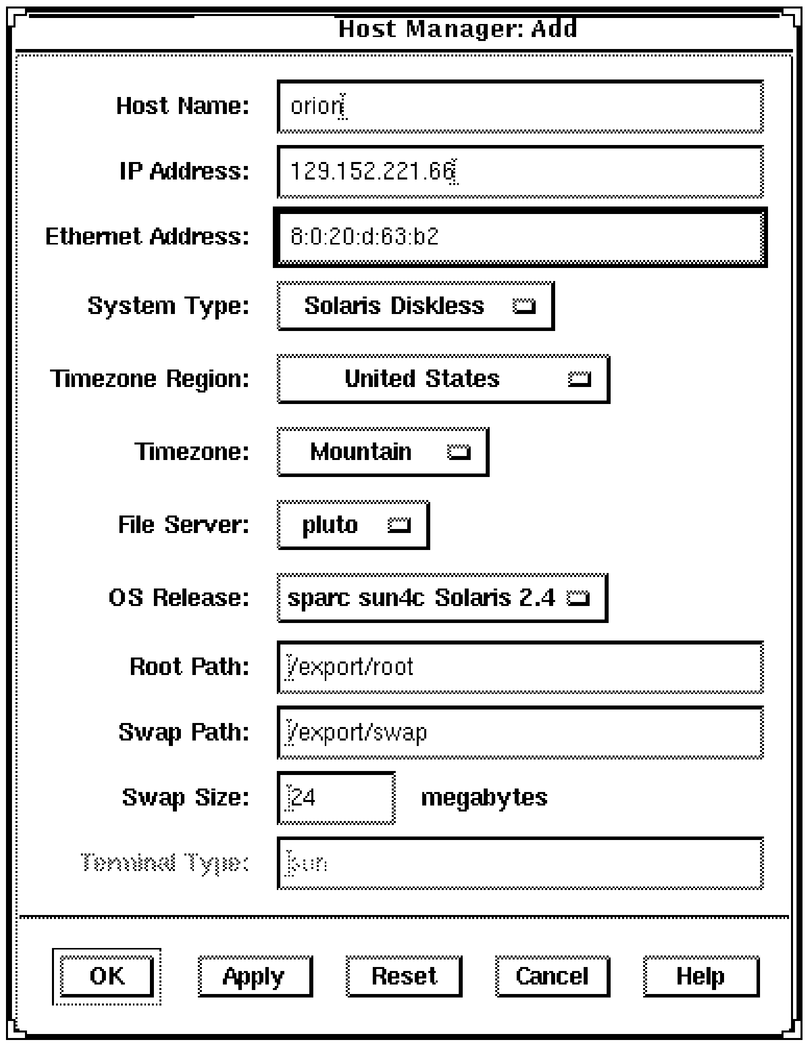
10. **Boot the diskless client.**

```
ok boot net
```

11. **Provide the following system configuration information for the diskless client during the initial boot process, if prompted.**
    - Geographic region
    - Time zone
    - Date and time

12. **Create a root password when prompted.**

Example—Completed Host Manager Add Window for a Diskless Client



▼ How to Add Support for a Dataless Client

The high-level steps in this procedure are:

- Adding system information about the dataless client
- Selecting OS services for the dataless client
- (Optional) Setting up remote install capabilities for the dataless client
- (Optional) Booting and installing the dataless client

Note – Before starting this procedure, make sure the system providing the services (the file server) has already been configured as an OS server.

---

1. **Start Solstice Host Manager from the Solstice Launcher and select the name service, if not done already.**
   See “How to Start Solstice Host Manager” on page 86 for more information.

2. **Select Add from the Edit menu on the Solstice Host Manager main window.**
   The Add window is displayed.

3. **Fill in the system information, selecting Solaris Dataless as the system type.**
   The system information includes all the fields from the host name through the time zone. If you need information to complete a field, click on the Help button to see field definitions for this window.

4. **Select a File Server.**
   The File Server defaults to the current host. Select Other from the Install Server menu to specify another host as the install server.

5. **Select the system’s architecture type and OS release from OS Release menu.**
   The architecture type must match the dataless client’s architecture and the OS release should match the Solaris release you want the dataless client to run.

6. **If you want to set up remote install capabilities for the system, continue to Step 7. If not, skip to Step 11.**

7. **Click on Enable Remote Install.**

8. **Select an Install Server.**
   The Install Server defaults to the current host. Select Other from the Install Server menu to specify another host as the install server.

9. **Click on Set Path to identify the path to the Solaris CD image on the install server.**

Note – The path to the Solaris CD image must be the same release that you specified in the OS Release menu.

If You Are Using ...	And ...	Then Enter the Path ...
The Solaris CD as the Solaris CD image	The Solaris CD is managed by Volume Management	`/cdrom/cdrom0/s0` or `/cdrom/cdrom0/s2`
	The Solaris CD is not managed by Volume Management	Where you mounted the Solaris CD
A copy of the Solaris CD on the Install Server's hard disk (by using `setup_install_server`)		Where you specified `setup_install_server` to copy the Solaris CD

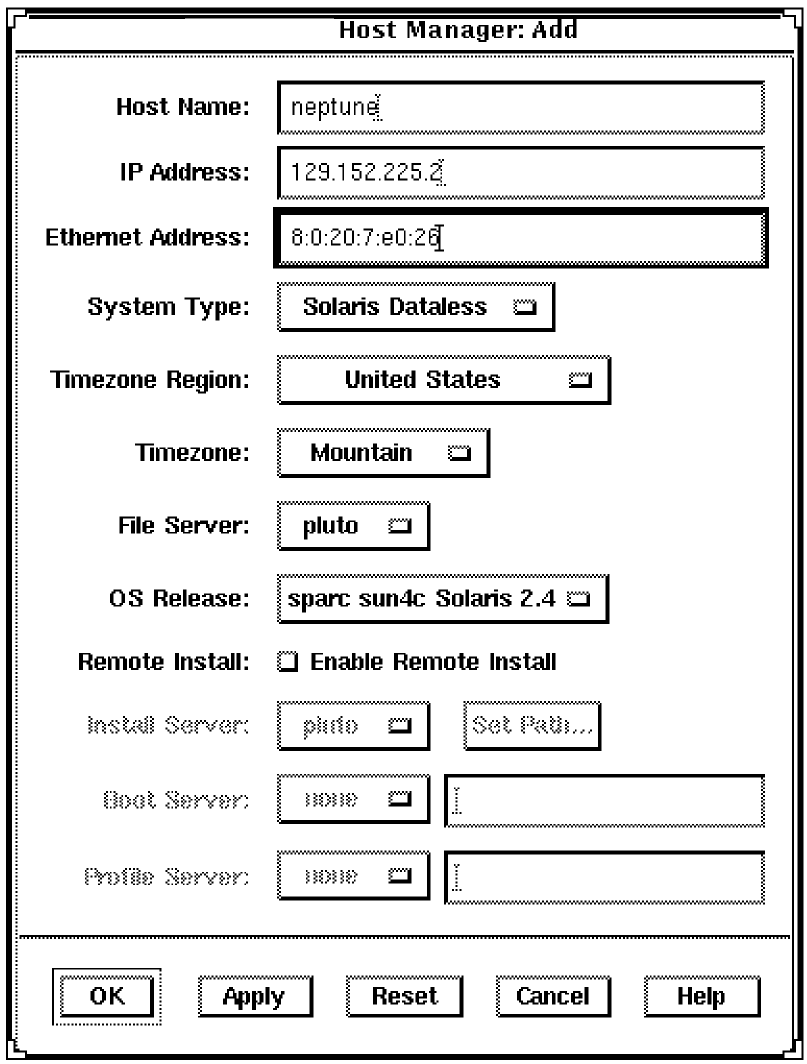
10. **If necessary, specify a boot server or profile server.**
    To specify another server other than the default, select Other from the menu. Select a Profile Server from the Profile Server pull-down menu. You must also specify a path to the boot software on the boot server or the custom JumpStart directory on the profile server.

11. **Click on OK on the Add window.**

12. **Select Save Changes from the File menu to add support for the dataless client.**
    The dataless client is displayed in the Host Manager main window.

13. **(Optional) Boot and install the dataless client.**
    For more information about booting and installing, see *SPARC: Installing Solaris Software* for SPARC systems or *x86: Installing Solaris Software* for x86 systems.

Example—Completed Host Manager Add Window for a Dataless Client



Maintaining Server and Client Support

Table 4-2 Task Map: Maintaining Server and Client Support

Activity	Description	For Instructions, Go To	
Modify Support for a System	Modify support for a system by choosing Modify from the Host Manager's Edit menu.	▼ How to Modify Support for a System	page 108
Delete Support for a System	Delete support for a system by choosing Delete from the Host Manager's Edit menu.	▼ How to Delete Support for a System	page 108

▼ How to Modify Support for a System

1. **Start Host Manager from the Solstice Launcher and select the name service, if not done already.**
   See "How to Start Solstice Host Manager" on page 86 for more information.
2. **Select a system entry to modify from the Host Manager main window.**
3. **Choose Modify from the Edit menu.**
   The Modify window contains the selected system entry.
4. **Modify support for the system.**
   If you need information to change a field, click on the Help button to see field definitions for this window.
5. **Click on OK on the Modify window.**
6. **Select Save Changes from the File menu to modify support for the system.**

▼ How to Delete Support for a System

1. **Start Solstice Host Manager from the Solstice Launcher and select the name service, if not done already.**
   See "How to Start Solstice Host Manager" on page 86 for more information.
2. **Select a system entry to delete from the Solstice Host Manager main window.**
3. **Select Delete from the Edit menu.**
   A window is displayed asking you to confirm the deletion.
4. **Click on OK.**
5. **Select Save Changes from the File menu to delete support for the system.**
   The system entry is deleted from the Host Manager main window.

Part 3 — Shutting Down and Booting a System

This part provides instructions for shutting down and booting systems running the Solaris 2.x release.

10

The Boot Process

Provides a high-level overview of the boot process including a description of the platform-specific hardware used to boot SPARC and x86 systems.

Overview of Shutting Down and Booting a System 5

This chapter provides guidelines for shutting down and booting a system.

This is a list of overview information in this chapter.

See Chapter 7, “Shutting Down a System,” for instructions on shutting down a system. See Chapter 8, “Booting a SPARC System,” or Chapter 9, “Booting an x86 System,” for instructions on booting a system.

5

The Solaris 2.x software environment is designed to run continuously so that electronic mail and network resources are available to users.

Occasionally, it is necessary to shut down or reboot a system because of a system configuration change, a scheduled maintenance event, or a power outage.

Terminology

This section describes the terminology used in shutting down and booting a system.

- **Run levels and init states –** A *run level* is a software configuration of processes and available services that describes how a system is shut down or booted. Run levels are also referred to as *init states* because the `init` process starts and stops the system processes that are available at each run level. This book refers to init states as run levels.
- **Boot Types –** A *boot type* describes how a system is booted, which may include a shut down of the operating system as well. Different boot types include:
  - **Interactive boot –** You are prompted to provide information about how the system is booted, such as the kernel and device path name.
  - **Reconfiguration boot –** The system is reconfigured to support newly added hardware or new pseudo devices.
  - **Recovery boot –** The system is hung or an invalid entry is prohibiting the system from booting successfully or from allowing users to log in.

Guidelines for Shutting Down a System

Keep the following in mind when shutting down a system:

- Use the `init` and `shutdown` commands to shut down a system. Both commands perform a clean system shutdown, which means all system processes are services are terminated normally.
- Use the `shutdown` command to shut down a server, because logged-in users and systems mounting resources from the server are notified before the server is shut down. Additional notification of system shutdowns via electronic mail is also recommended so that users can be prepared for system downtime.

- You need root privileges to use the `shutdown` or `init` command to shut down a system.
- Both `shutdown` and `init` commands take a run level as an argument. The three most common run levels are:
  - **Run level 3 –** Means that all system resources are available and users can login. By default, booting a system brings it to run level 3, which is used for normal day-to-day operations. Also known as multiuser level with NFS resources shared.
  - **Run level 6 –** Reboots the system from run level 3 (multiuser level with NFS resource shared) to run level 0, and back to run level 3. Rebooting is used to enable certain system configuration changes and to activate newly added software services.
  - **Run level 0 –** Means the operating system is shut down and it is safe to turn off power. Bringing a system to run level 0 is needed whenever the system is moved or hardware is added or removed.

  Run levels are fully described in Chapter 6, "Run Levels and Boot Files."

Guidelines for Booting a System

Keep the following in mind when booting a system:

- After a system is shut down, it is brought to a new run level using the `boot` command.
- A system can be rebooted by turning the power off and then back on. This is not a clean shutdown because system services and processes are terminated abruptly. However, turning a system's power off and back is an alternative for emergency situations.
- SPARC and x86 systems use different hardware components for booting. These differences are described in Chapter 10, "The Boot Process."

5

Performing a Reconfiguration Boot

Perform a reconfiguration boot when adding new hardware to the system or configuring support for pseudo devices, such as increasing the number of pseudo devices (ptys). Table 5-1 to determine which reconfiguration procedure to use.

Table 5-1 Reconfiguration Procedures

If You Are Reconfiguring the System To ...	See ...
Add a secondary disk	Chapter 29, “SPARC: Adding a Disk” or Chapter 30, “x86: Adding a Disk”
Add some other peripheral device	“How to Add a Peripheral Device” on page 391
Change the number of pseudo devices	"Managing System Performance" in *System Administration Guide, Volume II*

When to Shut Down a System

Table 5-2 provides a list of system administration tasks and the type of shut down needed to initiate the task.

Table 5-2 Shutting Down a System

If You Are ...	**Change to this Run Level ...**	**See a Example on ...**
Turning off system power due to anticipated power outage.	Run level 0, where it is safe to turn off power.	page 140
Changing kernel parameters in the `/etc/system` file.	Run level 6 (reboot the system)	page 141
Performing file system maintenance, such as performing a backup or restoring system data.	Run level S (single-user mode)	page 139
Repairing a system configuration file such as `/etc/system`.	N/A	N/A
Changing pseudo device parameters in the `/etc/system` file.	Reconfiguration boot	"Managing System Performance" in *System Administration Guide, Volume II*
Adding or removing hardware from the system.	Reconfiguration boot (plus turning off power when adding or removing hardware)	page 465
Repairing an important system file which is causing system boot failure.	N/A	N/A
Booting the kernel debugger (`kadb`) to track down a system problem.	Run level 0, if possible	page 140
Recovering from a hung system and you want to force a crash dump.	N/A	N/A

See Chapter 7, “Shutting Down a System,” for examples of shutting down a server or standalone system.

When to Boot a System

Table 5-3 provides a list of system administration tasks and the corresponding boot type used to complete the task.

Table 5-3 Booting a System

If You Are Rebooting the System After ...	Use This Boot Type ...	See a procedure for a SPARC System on ...	See a procedure for an x86 System on ...
Turning off system power due to anticipated power outage.	Turn system power back on	page 145	page 145
Changing kernel parameters in the `/etc/system` file.	Reboot the system to run level 3 (multiuser mode with NFS resources shared)	page 153	page 167
Performing file system maintenance, such as performing a backup or restoring system data.	Use Control-d from run level S to bring the system back to run level 3	page 155	page 168
Repairing a system configuration file such as `/etc/system`.	Interactive boot	page 156	page 170
Changing pseudo device parameters in the `/etc/system` file.	Reconfiguration boot	"Managing System Performance" in *System Administration Guide, Volume II*	"Managing System Performance" in *System Administration Guide, Volume II*
Adding or removing hardware from the system.	Reconfiguration boot (plus turning on system power after adding or removing hardware)	page 465	page 483
Booting the kernel debugger (`kadb`) to track down a system problem.	Booting `kabd`	page 162	page 175
Repairing an important system file which is causing system boot failure.	Recovery boot	page 172	page 172
Recovering from a hung system and you want to force a crash dump.	Recovery boot	See example on page 175	See example on page 175

See Chapter 8, “Booting a SPARC System,” or Chapter 9, “Booting an x86 System,” for examples of booting a system.

Run Levels and Boot Files 6

This chapter provides guidelines for shutting down and booting a system and information about run levels and boot files.

This is a list of the step-by-step instructions in this chapter.

This is a list of overview information in this chapter.

6

Run Levels

A system's *run level* (also known as an init state) defines what services and resources are available to users. A system can be in only one run level at a time.

The Solaris software environment has eight run levels, which are described in Table 6-1. The default run level 3 is specified in the `/etc/inittab` file.

Table 6-1 Solaris Run Levels

Run Level	**Init State**	**Type**	**Use This Level ...**
0	Power-down state	Power-down	To shut down the operating system so that it is safe to turn off power to the system.
1	Administrative state	Single-user	To access all available file systems with user logins allowed. The terminal from which you issue this command becomes the Console.
2	Multiuser state	Multiuser	For normal operations. Multiple users can access the system and the entire file system. All daemons are running except for NFS server and syslog.
3	Multiuser with NFS resources shared	Multiuser	For normal operations with NFS resource-sharing available.
4	Alternative multiuser state		This level is currently unavailable.
5	Power-down state	Power-down	To shut down the operating system so that it is safe to turn off power to the system. If possible, automatically turn off system power on systems that support this feature.
6	Reboot state	Reboot	To shut down the system to run level 0, and then reboot to multiuser state (or whatever level is the default in the `inittab` file).
s or S	Single-user state	Single-user	To run as a single user with all file systems mounted and accessible.

How to Determine a System's Run Level

Display run level information by using the `who -r` command to determine a system's run level.

```
$ who -r
```

Use the `who -r` command to determine a system's current run level for any level except run level 0.

Example—Determining a System's Run Level

```
$ who -r
.       run-level 3  Oct 26 15:04      3      0  S
$
```

In this example,

`run level 3`	Identifies the current run level.
`Oct 26 15:04`	Identifies the date of last run level change.
`3`	Is the current run level.
`0`	Identifies the number of times at this run level since the last reboot.
`S`	Identifies the previous run level.

6

The /etc/inittab File

When you boot the system or change run levels with the `init` or `shutdown` command, the `init` daemon starts processes by reading information from the `/etc/inittab` file. This file defines three important items for the `init` process:

- The system's default run level
- What processes to start, monitor, and restart if they terminate
- What actions to be taken when the system enters a new run level

Each entry in the `/etc/inittab` file has the following fields:

id:*rstate*:*action*:*process*

Table 6-2 describes the fields in an `inittab` entry.

Table 6-2 Fields in the `inittab` File

Field	Description
id	A unique identifier for the entry.
rstate	The run level, which corresponds to the command or (script) to be processed.
action	How the process specified in the process field is to be run.
process	The name of the process (or command) to execute.

Example—Default `inittab` *File*

The following example shows an annotated default `inittab` file:

```
1   ap::sysinit:/sbin/autopush -f /etc/iu.ap
2   fs::sysinit:/sbin/bcheckrc             >/dev/console 2>&1 </dev/console
3   is:3:initdefault:
4   p3:s1234:powerfail:/sbin/shutdown -y -i0 -g0 >/dev/console 2>&1 </dev/console
5   s0:0:wait:/sbin/rc0 off                >/dev/console 2>&1 </dev/console
6   s1:1:wait:/sbin/shutdown -y -iS -g0    >/dev/console 2>&1 </dev/console
7   s2:23:wait:/sbin/rc2                   >/dev/console 2>&1 </dev/console
8   s3:3:wait:/sbin/rc3                    >/dev/console 2>&1 </dev/console
9   s5:5:wait:/sbin/rc5 ask                >/dev/console 2>&1 </dev/console
10  s6:6:wait:/sbin/rc6 reboot             >/dev/console 2>&1 </dev/console
11  of:0:wait:/sbin/uadmin 2 0             >/dev/console 2>&1 </dev/console
12  fw:5:wait:/sbin/uadmin 2 2             >/dev/console 2>&1 </dev/console
13  RB:6:wait:/sbin/sh -c 'echo "\nThe system is being restarted."' >/dev/console 2>&1
14  rb:6:wait:/sbin/uadmin 2 1             >/dev/console 2>&1 </dev/console
15  sc:234:respawn:/usr/lib/saf/sac -t 300
16  co:234:respawn:/usr/lib/saf/ttymon -g -h -p "‘uname -n‘ console  login: "\
    -T terminal_type -d /dev/console -l console -m ldterm,ttcompat
```

1 STREAMS module initialization
2 File system check
3 Default run level
4 Power fail shutdown
5 Run level 0
6 Run level 1
7 Run level 2
8 Run level 3
9 Run level 5
10 Run level 6
11 Off
12 Firmware
13 Reboot
14 Reboot single-user
15 Service access controller initialization
16 Console initialization

What Happens When the System is Brought to Run Level 3

1. The `init` process is started and reads the `/etc/default/init` file to set any environment variables. By default, only the `TIMEZONE` variable is set.
2. Then `init` reads the `inittab` file to do the following:
   a. Identify the `initdefault` entry, which defines the default run level (3).
   b. Execute any process entries that have `sysinit` in the action field so that any special initializations can take place before users login.
   c. Execute any process entries that have 3 in the `rstate` field, which matches the default run level, 3.

See `init(1M)` for a detailed description of how the `init` process uses the `inittab` file.

Table 6-3 describes the key words used for run level 3's action field.

Table 6-3 Run Level 3 Action Key Word Descriptions

Key Word	**Starts the Specified Process ...**
`powerfail`	Only when the system receives a power fail signal.
`wait`	And waits for its termination.
`respawn`	If it does not exist. If the process already exists, continue scanning the `inittab` file.

Table 6-4 describes the processes (or commands) executed at run level 3.

Table 6-4 Run Level 3 Command Descriptions

Command or Script Name	Description
`/usr/sbin/shutdown`	Shuts down the system. The `init` process runs the `shutdown` command only if the system has received a `powerfail` signal.
`/sbin/rc2`	Sets up the time zone, then starts the standard system processes, bringing the system up into run level 2 (multiuser mode).
`/sbin/rc3`	Starts NFS resource sharing for run level 3.
`/usr/lib/saf/sac -t 30`	Starts the port monitors and network access for UUCP. This process is restarted if it fails.
`/usr/lib/saf/ttymon -g -h -p "`uname -n` console login: " -T` *terminal_type* `-d /dev/console -l console`	Starts the `ttymon` process that monitors the console for login requests. This process is restarted if it fails. The *terminal_type* on a SPARC system is `sun` The *terminal_type* on a x86 system is `AT386`

6

Run Control Scripts

The Solaris software environment provides a detailed series of run control (`rc`) scripts to control run level changes. Each run level has an associated `rc` script located in the `/sbin` directory:

- `rc0`
- `rc1`
- `rc2`
- `rc3`
- `rc5`
- `rc6`
- `rcS`

For each `rc` script in the `/sbin` directory, there is a corresponding directory named `/etc/rc`*n*`.d` that contains scripts to perform various actions for that run level. For example, `/etc/rc2.d` contains files used to start and stop processes for run level 2.

```
# ls /etc/rc2.d
K20lp            S30sysid.net     S74autofs        S88sendmail
K20spc           S69inet          S74syslog        S88utmpd
K60nfs.server    S71rpc           S75cron          S92volmgt
README           S71sysid.sys     S76nscd          S93cacheos.finish
S01MOUNTFSYS     S72autoinstall   S80PRESERVE      S99audit
S05RMTMPFILES    S72inetsvc       S80lp
S20sysetup       S73nfs.client    S80spc
#
```

The `/etc/rc`*n*`.d` scripts are always run in ASCII sort order. The scripts have names of the form:

[K,S][0-9][0-9][A-Z][0-99]

Files beginning with `K` are run to terminate (kill) a system process. Files beginning with `S` are run to start a system process.

Run control scripts are also located in the `/etc/init.d` directory. These files are linked to corresponding run control scripts in the `/etc/rc*.d` directories.

The actions of each run control script are summarized in Table 6-5 through Table 6-11, starting on page 129.

Using a Run Control Script to Stop or Start Services

One advantage of having individual scripts for each run level is that you can run scripts in the /etc/init.d directory individually to turn off functionality without changing a system's run level.

How to Use a Run Control Script to Stop or Start a Service

1. **Become root.**
2. **Turn off functionality.**

```
# /etc/init.d/filename stop
```

3. **Restart functionality.**

```
# /etc/init.d/filename start
```

Verification—Using a Run Control Script to Stop or Start a Service

Use the ps and grep commands to identify whether the service has been stopped or started.

```
# ps -ef | grep service
```

Example—Using a Run Control Script to Stop or Start a Service

Turn off NFS server functionality by typing:

```
# /etc/init.d/nfs.server stop
# ps -ef | grep nfs
```

Restart the NFS services by typing:

```
# /etc/init.d/nfs.server start
# ps -ef | grep nfs
    root   141     1 40   Jul 31 ?        0:00 /usr/lib/nfs/statd
    root   143     1 80   Jul 31 ?        0:01 /usr/lib/nfs/lockd
    root   245     1 34   Jul 31 ?        0:00 /usr/lib/nfs/nfsd -a 16
    root   247     1 80   Jul 31 ?        0:02 /usr/lib/nfs/mountd
    root  1324  1318 11 13:29:52 pts/0    0:00 grep nfs
```

Adding a Run Control Script

If you want to add a run control script to start and stop a service, copy the script into the `/etc/init.d` directory and create links in the `rc*.d` directory you want the service to start and stop.

See the `README` file in each `/etc/rc*.d` directory for more information on naming run control scripts. The procedure below describes how to add a run control script.

▼ How to Add a Run Control Script

1. **Become root.**
2. **Add the script to the** `/etc/init.d` **directory.**

```
# cp filename /etc/init.d
```

3. **Create links to the appropriate** `rc*.d` **directory.**

```
# cd /etc/init.d
# ln filename /etc/rc2.d/Snnfilename
# ln filename /etc/rcn.d/Knnfilename
```

Verification—Adding a Run Control Script

Use the `ls` command to verify that the script has links in the specified directories.

```
# ls /etc/init.d/filename /etc/rc2.d/Snnfilename /etc/rcn.d/Knnfilename
/etc/init.d/filea   /etc/rc2.d/Snnfilename /etc/rcn.d/Knnfilename
```

Example—Adding a Run Control Script

```
# cp xyz /etc/init.d
# cd /etc/init.d
# ln xyz /etc/rc2.d/S100xyz
# ln xyz /etc/rc0.d/K100xyz
```

Disabling a Run Control Script

Disable a run control script by renaming it with a dot (.) at the beginning of the new file name. Files that begin with a dot are not executed. If you copy a file by adding a suffix to it, both files will be run.

▼ How to Disable a Run Control Script

1. **Become root.**

2. **Rename the script by adding a dot (.) to the beginning of the new file.**

```
# cd /etc/rcn.d
# cp filename .filename
```

Example—Disabling a Run Control Script

The following example changes the K00ANNOUNCE script name but saves the original script.

```
# cd /etc/rc0.d
# cp K00ANNOUNCE .K00ANNOUNCE
```

Run Control Script Summaries

Table 6-5 summarizes the /sbin/rc0 script.

Table 6-5 The /sbin/rc0 Script

Script Name	Description
/sbin/rc0	Performs the following tasks: • Stops system services and daemons • Terminates all running processes • Unmounts all file systems

Table 6-6 summarizes the /sbin/rc1 script.

Table 6-6 The /sbin/rc1 Script

Script Name	Description
/sbin/rc1	Runs the /etc/rc1.d scripts to perform the following tasks: • Stops system services and daemons • Terminates all running processes • Unmounts all file systems • Brings the system up in single-user mode

Table 6-7 summarizes the /sbin/rc2 script.

Table 6-7 The /sbin/rc2 Script

Script Name	Description
/sbin/rc2	Runs the /etc/rc2.d scripts to perform the following tasks: • Mounts all local file systems • Enables disk quotas if at least one file system was mounted with the quota option • Saves editor temporary files in /usr/preserve • Removes any files in the /tmp directory • Rebuilds device entries for reconfiguration boots • Configures system accounting • Configures default router • Sets NIS domain and ifconfig netmask • Reboots the system from the installation media or a boot server if either /.PREINSTALL or /AUTOINSTALL exists • Starts inetd and rpcbind and named, if appropriate • Starts Kerberos client-side daemon, kerbd • Starts NIS daemons (ypbind) and NIS+ daemons (rpc.nisd), depending on whether the system is configured for NIS or NIS+, and whether the system is a client or a server • Starts keyserv, statd, lockd, and utmpd • Mounts all NFS entries • Starts ncsd (name service cache daemon) • Starts automount, cron, LP, sendmail, utmpd, and vold daemons

Note – Many of the system services and applications that are started at run level 2 depend on what software is installed on the system.

Table 6-8 summarizes the /sbin/rc3 script.

Table 6-8 The /sbin/rc3 Script

Script Name	Description
/sbin/rc3	Runs the /etc/rc3.d scripts to perform the following tasks: • Cleans up sharetab • Starts nfsds • Starts mountd • If boot server, starts rarpd, rpc.bootparamd, and rpld

Table 6-9 summarizes the /sbin/rc5 script.

Table 6-9 The /sbin/rc5 Script

Script Name	Description
/sbin/rc5	Runs the /etc/rc0.d scripts to perform the following tasks: • Kills the printer and syslog daemons • Unmounts local and remote file systems • Stops NFS server and client services • Stops NIS, RPC, and cron services Kills all active processes and initiates an interactive boot

Table 6-10 summarizes the /sbin/rc6 script.

Table 6-10 The /sbin/rc6 Script

Script Name	Description
/sbin/rc6	Performs the following tasks: • Runs the /etc/rc0.d/K* scripts to stop system processes • Kills all active processes • Unmounts the file systems • Runs the initdefault entries in the /etc/inittab file.

Table 6-11 summarizes the /sbin/rcS script.

Table 6-11 The /sbin/rcS Script

Script Name	Description
/sbin/rcS	Runs the /etc/rcS.d scripts to bring the system up to run level S. The following tasks are performed from these scripts: • Establishes a minimal network • Mounts /usr, if necessary • Sets the system name • Checks the root (/) and /usr file systems • Mounts pseudo file systems (/proc and /dev/fd) • Rebuilds the device entries for reconfiguration boots • Checks and mounts other file systems to be mounted in single-user mode

Shutting Down a System 7

This chapter describes the procedures for shutting down systems. This is a list of the step-by-step instructions in this chapter.

This is a list of the overview information in this chapter.

For overview information about the available run levels, see Chapter 6, "Run Levels and Boot Files."

When to Shut Down the System

The Solaris system software is designed to be left running continuously so that the electronic mail and network software can work correctly. However, some system administration tasks and emergency situations require that the system is shut down to a level where it is safe to remove power or brought to an intermediate level, where not all system services are available, such as:

- Adding or removing hardware
- Preparing for an expected power outage
- Performing file system maintenance, such as a backup

See Chapter 5, "Overview of Shutting Down and Booting a System" for a complete list of system administration tasks requiring a system shutdown.

How to Shut Down a System

Using the `init` and `shutdown` commands are the primary ways to shut down a system. Both commands perform a *clean shutdown* of the system, which means all file system changes are written to the disk, and all system services, processes, and the operating system are terminated normally.

Using a system's stop key sequence or turning a system off and then on are not clean shutdowns because system services are terminated abruptly. However, is it sometimes necessary to use these actions in emergency situations. See Chapter 8, "Booting a SPARC System," or Chapter 9, "Booting an x86 System," for instructions on system recovery techniques.

Note – There is no clean way to bring a system to run level 2 or S from run level 3 (multiuser state with NFS resources shared). The best way to bring a system to an intermediate run level is to bring the system to run level 0, and then boot the system to run level S.

Table 7-1 describes the various shut down commands and provides recommendations for using them.

Table 7-1 Shut Down Commands

Command	Description	This Command Is ...
`shutdown`	An executable shell script that calls the `init` program to shut down the system. The system is brought to run level S by default.	Recommended for servers running at run level 3 because users are notified of the impending shut down as are the systems that are mounting resources from the server being shut down.
`init {0,1,2,3,6,S,s}`	An executable that kills all active process and syncs the disks before changing run levels.	Recommended for standalone systems when other users will not be affected. It provides a faster system shutdown because users are not notified of the impending shutdown.
`reboot`	An executable that syncs the disks and passes booting instructions to the `uadmin` system call, which, in turn, stops the processor.	Not recommended; use the `init` command instead.
`halt`	An executable that syncs the disks and stops the processor.	Not recommended because it doesn't execute the `/etc/rc0` script, which stops all processes, syncs the disks, and unmounts any remaining file systems.

Note – The `/usr/sbin/shutdown` command, not the `/usr/ucb/shutdown` command, is used in this chapter and throughout this book.

When to Turn Off Power to Devices

Turning off power to all system devices is necessary when you need to:

- Replace or add hardware
- Move the system from one location to another
- Prepare for an expected power outage or natural disaster like an approaching electrical storm

All system devices include the CPU, the monitor, and external devices such as disks, tapes, and printers.

The steps for turning off power to all devices are performed in addition to shutting down the system.

Notifying Users of System Down Time

When the `shutdown` command is initiated, it will notify all logged-in users and all systems that are mounting resources from it of the impending shutdown with a warning and then a final message.

This is why the `shutdown` command is recommended over the `init` command when used on a server. When using either command, you may want to give users more notice by sending a mail message about any scheduled system shut down.

Use the `who` command to determine which users on the system need to be notified. This command is also useful for determining a system's current run level, which is described on page 119.

▼ How to Determine Who is Logged in to a System

1. **Log into the system to be shut down.**
2. **Display logged-in users with the `who` command.**

```
$ who
```

Example—Determining Who Is Logged in to a System

The following example displays the output of the `who` command.

```
$ who
holly ❶   console ❷   Jul 31 08:04 ❸
kryten     pts/0        Jul 28 10:04(starbug)❹
lister     pts/1        Jul 31 08:42(bluemidget)
```

In this example,

❶ Identifies the *username* of the logged-in user.

❷ Identifies the *terminal line* of the logged-in user.

❸ Identifies the *date* and *time* the user logged in.

❹ (Optional) Identifies the *host name* if a user is logged in from a remote system.

▼ How to Shut Down a Server

1. **Become root.**
2. **Find out if users are logged into the system.**

```
# who
```

A list of all logged-in users is displayed. You may want to send mail or broadcast a message to let users know that the system is being shut down.

3. **Shut down the system by using the** `shutdown` **command.**

```
# shutdown -iinit-state -ggrace-period -y
```

In this command,

`-i`*init-state*	Allows you to bring the system to an init state different from the default of S. The choices are 0, 1, 2, 5, and 6.
`-g`*grace-period*	Indicates a time (in seconds) before the system is shut down. The default is 60 seconds.
`-y`	Continues to shut down the system without intervention; otherwise, you are prompted to continue the shutdown process after 60 seconds.

4. **If you are asked for confirmation, type** `y`**.**

```
Do you want to continue? (y or n): y
```

If you used the `shutdown -y` command, you will not be prompted to continue.

5. **Type the root password, if prompted.**

```
Type Ctrl-d to proceed with normal startup,
(or give root password for system maintenance): xxx
```

6. **After you have finished the system administration tasks, press Control-d to return to the default run system level.**

Verification—Shutting Down a Server

Use the following table to verify the system is at the run level specified in the `shutdown` command.

If the System was Brought To ...	The SPARC System Prompt Should Be ...	The x86 System Prompt Should Be ...
run level S (single-user state)	#	#
run level 0 (power-down state)	`ok` or `>`	`type any key to continue`
run level 3 (multiuser state with remote resources shared)	*hostname* `console login:`	*hostname* `console login:`

Example—Bringing a SPARC System to Run Level S

In the following example, the `shutdown` and `boot` commands are used to bring a SPARC system to run level S (single-user state) in 3 minutes.

```
# who
root       console      Aug  1 08:35
# shutdown -i0 -g180 -y

Shutdown started.    Tue Aug  1 08:39:17 MDT 1995

Broadcast Message from root (console) on mars Tue Aug  1 08:39:18
The system will be shut down in 1 minute
Broadcast Message from root (console) on mars Tue Aug  1 08:39:50
The system will be shut down in 30 seconds
.
.
.
INIT: New run level: 0
The system is coming down.  Please wait.
syncing file systems... [7] [7] [5] done
Program terminated
ok boot -s
Booting from: sd(0,0,0) -s
SunOS Release 5.5 Version generic [UNIX(R) System V Release 4.0]
Copyright (c) 1983-1995, Sun Microsystems, Inc.
configuring network interfaces: le0.
Hostname: mars

INIT: SINGLE USER MODE

Type Ctrl-d to proceed with normal startup,
(or give root password for system maintenance): xxx
Entering System Maintenance Mode
#
```

Example—Bringing a SPARC System to Run Level 0

In the following example, the `shutdown` command is used to bring a SPARC system to run level 0 in 5 minutes without requiring additional confirmation.

```
# who
kryten     console      Jul 31 16:05
rimmer     pts/1        Jul 31 16:06    (starbug)
pmorph     pts/2        Jul 31 16:06    (bluemidget)
Send mail message to logged-in users
# shutdown -i0 -g300 -y
Shutdown started.    Fri Nov  4 11:07:33 MST 1994
Broadcast Message from root (console) on pluto Fri Nov  4 11:07:34
The system will be shut down in 3 minutes
.
.
.
INIT: New run level: 0
The system is coming down.  Please wait.
.
.
.
The system is down.
syncing file systems... [11] [9] [5] done
Program terminated
Type  help  for more information
ok
```

See “How to Turn Off Power to All Devices” on page 145 if you are bringing the system to run level 0 to turn off power to all devices,

Example—Rebooting a SPARC System to Run Level 3

In the following example, the `shutdown` command is used to reboot a SPARC system to run level 3 in 2 minutes without requiring additional confirmation.

```
# who
kryten     console      Jul 31 16:05
rimmer     pts/1        Jul 31 16:06    (starbug)
pmorph     pts/2        Jul 31 16:06    (bluemidget)
Send mail message to logged-in users
# shutdown -i6 -g120 -y
Shutdown started.    Fri Nov  4 16:10:27 MST 1994
Broadcast Message from root (console) on pluto Fri Nov  4 16:10:29
The system will be shut down in 1 minute
Changing to init state 6 - please wait
#
INIT: New run level: 6
The system is coming down.  Please wait.
.
.
.
The system is down.
syncing file systems... [11] [9] [5] done
rebooting...
.
.
.
pluto console login:
```

Where to Go From Here

Regardless of the reason for shutting down the system, you'll probably want to return to run level 3 where all file resources are available and users can log in. See Chapter 8, "Booting a SPARC System," or Chapter 9, "Booting an x86 System," for instructions on bringing a system back to a multiuser state.

How to Shut Down a Standalone System

1. **Become root.**
2. **Shut down the system by using the `init` command.**

```
# init run-level
```

In this command,

run-level	Identifies the new run level.

Verification—Shutting Down a Standalone System

Use the following table to verify the system is at the run level specified in the `init` command.

If the System was Brought To ...	The SPARC System Prompt Should Be ...	The x86 System Prompt Should Be ...
run level S (single-user state)	#	#
run level 2 (multiuser state)	#	#
run level 0 (power-down state)	`ok` or `>`	`type any key to continue`
run level 3 (multiuser state with remote resource shared)	*hostname* `console login:`	*hostname* `console login:`

Example—Bringing an x86 System to Run Level 0

In the following example, the `init` command is used to bring an x86 system to the level where it is safe to turn off power.

```
# init 0
#
INIT: New run level: 0
The system is coming down.  Please wait.
.
.
.
The system is down.
syncing file systems... [11] [10] [3] done
Type any key to continue
```

See “How to Turn Off Power to All Devices” on page 145 if you are bringing the system to run level 0 to turn off power to all devices.

Example—Bringing a SPARC System to Run Level S

In the following example, the `init` and `boot` commands are used to bring a SPARC system to run level S (single-user state).

```
# init 0
#
INIT: New run level: 0
The system is coming down.  Please wait.
.
.
.
syncing file systems... [7] [7] [5] done
Program terminated
ok boot -s
Booting from: sd(0,0,0) -s
SunOS Release 5.5 Version generic [UNIX(R) System V Release 4.0]
Copyright (c) 1983-1994, Sun Microsystems, Inc.
configuring network interfaces: le0.
Hostname: venus

INIT: SINGLE USER MODE

Type Ctrl-d to proceed with normal startup,
(or give root password for system maintenance): xxx
Entering System Maintenance Mode

#
```

Where to Go From Here

Regardless of the reason for shutting down the system, you'll probably want to return to run level 3 where all file resources are available and users can log in. See Chapter 8, "Booting a SPARC System," or Chapter 9, "Booting an x86 System," for instructions on bringing a system back to a multiuser state.

▼ How to Turn Off Power to All Devices

1. **Use the following table to determine which procedure to use for shutting down the system.**

If You Are Shutting Down a Server ...	If You Are Shutting Down a Standalone System ...
See "How to Shut Down a Server" on page 137.	See "How to Shut Down a Standalone System" on page 142.

2. **Turn off power to all devices after the system is shutdown. If necessary, also unplug the power cables.**

3. **After power can be restored, use the following steps to turn on the system and devices.**

   a. **Plug in the power cables.**

   b. **Turn on the monitor.**

   c. **Turn on disk drives, tape drives, and printers.**

   d. **Turn on the CPU.**
   The system to brought to run level 3 after the CPU is turned on.

Booting a SPARC System 8

This chapter describes procedures for using the OpenBoot™ PROM monitor and procedures for booting a SPARC system to different run levels.

This is a list of the step-by-step instructions in this chapter.

For overview information about the boot process, see Chapter 10, “The Boot Process.”

8

Using the Boot PROM

System administrators typically use the PROM level to boot a system but occasionally may need to change the way the system works, such as setting which device to boot from or running hardware diagnostics, before the system is brought to a multiuser state.

Changing the default boot device is necessary when you want to add a new drive to the system either permanently or temporarily, or if you convert a standalone system to a diskless client that needs to boot from the network.

See `monitor(1M)` or `eeprom(1M)` for a complete list of PROM commands.

▼ SPARC: How to Switch to the `ok` Prompt

When the system is halted, the PROM monitor prompt is either the greater than sign (>) or `ok`.

Switch from the > prompt to the `ok` prompt on SPARC systems by typing the following command.

```
> n
ok
```

All examples in this section use the `ok` prompt.

▼ SPARC: How to Find the PROM Release for a System

Display a system's PROM release level with the `banner` command.

```
ok banner
SPARCstation 2, Type 4 Keyboard
ROM Rev. 2.2, 16 MB memory installed, Serial #nnnnnn
Ethernet address 8:0:20:f:fd:6c HostID nnnnnnnn
```

Hardware configuration information, including the release number of the PROM, is displayed. The PROM release level is indicated by the ROM Rev. number.

▼ SPARC: How to Change the Default Boot Device

Use this procedure when you need to change the default boot device.

1. **Become root.**
2. **Halt the system by using the `init` command.**

```
# init 0
```

The PROM prompt is displayed.

3. **If the > PROM prompt is displayed, type `n` and press Return.**

```
> n
ok
```

The `ok` PROM prompt is displayed.

4. **Change the `boot-device` setting by using the `setenv` command.**

```
ok setenv boot-device disk[n]
```

In this command,

`boot-device`	Identifies the parameter for setting the device from which to boot.
`disk`[*n*]	Identifies the `boot-device` value and in this case, *n* is the disk number.

Use the `probe-scsi-all` command if you need help identifying the disk number.

5. **Verify the change by using the `printenv` command.**

```
ok printenv boot-device
```

6. **Save the new** `boot-device` **value by using the** `reset` **command.**

```
ok reset
```

The `boot-device` setting is written to the PROM.

Example—Changing the Default Boot Device

```
# init 0
#
INIT: New run level: 0
.
.
.
The system is down.
syncing file systems... [11] [10] [5] done
Program terminated
Type  help  for more information
ok setenv boot-device disk
boot-device =          disk
ok printenv boot-device
boot-device              disk                          disk
ok reset
SPARCstation IPC, No Keyboard
ROM Rev. 2.9, 12 MB memory installed, Serial #32522.
Ethernet address 8:0:20:b:40:e9, Host ID: 52007f0a.
Testing   1 megs of memory. Still to go    0
Boot device: /sbus/esp@0,800000/sd@3,0   File and args:
.
.
.
pluto console login:
```

▼ SPARC: How to Reset the System

Run the `reset` command from the `ok` prompt.

```
ok reset
```

The self-test program, which runs diagnostic tests on the hardware, is executed and the system is rebooted.

8

Booting a SPARC System

Table 8-1 describes the boot scenarios covered in this chapter.

Table 8-1 Boot Type Descriptions

Booting the System ...	Is Usually Done ...	See an Example On ...
To run level 3 (multiuser state with NFS resource shared)	After halting the system or performing some system hardware maintenance task. This is the default boot level where all resources are available and users can log into the system.	page 154
To run level S (single-user state)	After performing some system maintenance task such as backing up a file system. At this level only some file systems are mounted and users cannot log into the system.	page 155
Interactively	After making temporary changes to a system file or the kernel for testing purposes. This type of boot allows you to recover easily if there are problems with the system file or kernel by supplying an alternative pathname to these files when prompted. Use the default settings for the other system prompts.	page 157
From local CD-ROM or the network for recovery purposes	To repair an important system file that is preventing the system from booting successfully. This type of boot is also used for installing (or upgrading) a new release of the operating system.	page 159
Using `kadb`	To troubleshoot system problems by running the kernel debugger.	page 162

If a system is turned off, turning it on starts the multiuser boot sequence. The following procedures show how to boot to different run levels from the `ok` PROM prompt.

Use the `who -r` command to verify that the system is brought to the specified run level.

See Chapter 6, "Run Levels and Boot Files," for a description of run levels.

▼ SPARC: How to Boot a System to Run Level 3 (Multiuser State)

Boot to run level 3 by using the `boot` command.

```
ok boot
```

The automatic boot procedure displays a series of startup messages, and brings the system to run level 3.

Verification—Booting a System to Run Level 3 (Multiuser State)

If the system successfully boots to run level 3 if the login prompt is displayed.

```
hostname console login:
```

Example—Booting a System to Run Level 3 (Multiuser State)

The following example displays the messages from booting a system to run level 3.

```
ok boot
Resetting ...
SPARCstation 10 (1 X 390Z50), Keyboard Present
ROM Rev. 2.14, 32 MB memory installed, Serial #number.
Ethernet address 8:0:20:1f:21:be, Host ID: number.
Rebooting with command:
Boot device: /iommu/sbus/espdma@f,400000/esp@f,800000/sd@3,0
File and args:
SunOS Release 5.5 Version [UNIX(R) System V Release 4.0]
Copyright (c) 1983-1995, Sun Microsystems, Inc.
configuring network interfaces: le0.
Hostname: venus
The system is coming up.  Please wait.
checking ufs filesystems
/dev/rdsk/c0t3d0s7: is clean.
/dev/rdsk/c0t3d0s5: is clean.
NIS domainname is solar.com
starting rpc services: rpcbind keyserv nis_cachemgr kerbd done.
Setting netmask of le0 to 255.255.255.0
Setting default interface for multicast: add net 224.0.0.0:
gateway venus
syslog service starting.
volume management starting.
The system is ready.
venus console login:
```

▼ SPARC: How to Boot a System to Run Level S (Single-User State)

1. **Boot the system to run level S by using the `boot -s` command.**

```
ok boot –s
```

2. **Enter the root password when the following message is displayed.**

```
INIT: SINGLE USER MODE
Type Ctrl-d to proceed with normal startup,
(or give root password for system maintenance):   xxx
```

3. **To bring the system up to multiuser state after the system maintenance task is performed, press Control-d.**

Verification—Booting a System to Run Level S (Single-User State)

Use the `who -r` command to verify the system is at run level S.

Example—Booting a System to Run Level S (Single-User State)

The following example displays a system booted to run level S.

```
ok boot -s
.
.
.
SunOS Release 5.5 Version [UNIX(R) System V Release 4.0]
Copyright (c) 1983-1995, Sun Microsystems, Inc.
configuring network interfaces: le0.
Hostname: mars
INIT: SINGLE USER MODE
Type Ctrl-d to proceed with normal startup,
(or give root password for system maintenance): xxx
Sun Microsystems Inc.   SunOS 5.5 October 1995
# who -r
   .       run-level S  Aug  4 13:11     S      0  ?
Perform some maintenance task
# <Control-d>
```

SPARC: How to Boot a System Interactively

1. Boot the system interactively by using the `boot -a` command.

```
ok boot -a
```

2. Answer the system prompts as described in Table 8-2.

Table 8-2 Interactive Boot Procedure Steps

If the System Displays ...	Do the Following ...
`Enter filename [kernel/unix]:`	Provide the name of another kernel to use for booting. Or, press Return to use the default kernel (`/platform/‘uname -m‘/kernel/unix`).
`Name of default directory for modules [/platform/‘uname -m‘/kernel /kernel /usr/kernel]:`	Provide an alternate path for the modules directory and press Return. Or, press Return to use the default modules directory path.
`Name of system file [/etc/system]:`	Provide the name of an alternate system file and press Return. Or, press Return to use the default `/etc/system` file.
`root filesystem type [ufs]:`	Press Return to use the default root file system type: UFS for local disk booting or NFS for diskless clients.
`Enter physical name of root device [`*physical_device_name*`]:`	Provide an alternate device name and press Return. Or, press Return to use the default physical name of the root device.

Verification—Booting a System Interactively

The system booted interactively if you are prompted to answer the questions listed in Table 8-2.

Example—Booting a System Interactively

In following example, the default choices (shown in square brackets [ ]) are accepted.

```
ok boot -a
.
.
.
Resetting ...
Rebooting with command: -a
Boot device: /iommu/sbus/espdma@f,400000/esp@f,800000/sd@3,0
File and args: -a
Enter filename [kernel/unix]: Retur
Enter default directory for modules
[/platform/SUNW,SPARCstation-10/kernel /kernel /usr/kernel]:
Return
SunOS Release 5.5 Version [UNIX(R) System V Release 4.0]
Copyright (c) 1983-1995, Sun Microsystems, Inc.
Name of system file [etc/system]:  Return
root filesystem type [ufs]: Return
Enter physical name of root device
[/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000
/sd@3,0:a]: Return
configuring network interfaces: le0.
Hostname: earth
The system is coming up.  Please wait.
.
.
.
The system is ready.
earth console login:
```

SPARC: How to Boot a System for Recovery Purposes

This procedure is needed when an important file, such as `/etc/passwd`, has an invalid entry and is causing the boot process to fail.

If you need help identifying a system's device names, refer to Chapter 26, "Accessing Devices."

1. **Follow the instructions below depending on whether you are booting from the Solaris 2.x installation CD or the network.**

If You are Booting From ...	Then ...
Solaris 2.x installation CD	1. Insert the Solaris 2.x installation CD into the CD caddy. 2. Insert the CD caddy into the CD-ROM drive. 3. Boot from the installation CD in single-user mode: `ok` **`boot cdrom -s`**
The network and an installation server or remote CD drive are available	Use the following command: `ok` **`boot net -s`**

2. **Mount the file system that has the file with an invalid entry.**

```
# mount /dev/dsk/device-name /a
```

3. **Change to the newly mounted directory.**

```
# cd /a/directory
```

4. **Set the terminal type.**

```
# TERM=sun
# export TERM
```

5. **Remove the invalid entry from the file using an editor.**

```
# vi filename
```

6. **Change to the root (/) directory.**

```
# cd /
```

7. **Unmount the /a directory.**

```
# umount /a
```

8. **Reboot the system.**

```
# init 6
```

Verification—Booting a System for Recovery Purposes

The procedure is successful if the system successfully boots to run level 3.

Example—Booting a System for Recovery Purposes

The following example uses the `/etc/passwd` as an example of repairing an important system file, after booting from a local CD-ROM.

```
ok boot cdrom -s
# mount /dev/dsk/rdsk/c0t3d0s0 /a
# cd /a/etc
# TERM=sun
# export TERM
# vi passwd
# cd /
# umount /a
# init 6
```

SPARC: How to Stop the System for Recovery Purposes

The specific stop key sequence depends on your keyboard type. For example, you can press Stop-A or L1-A. On terminals, press the Break key.

1. **Type the abort key sequence for your system.**
   The monitor displays the `ok` PROM prompt.

```
ok
```

2. **Use the `sync` command to synchronize the disks.**

```
ok sync
```

3. **When you see the `syncing file systems...` message, press the abort key sequence for your system again.**

4. **Type the appropriate `boot` command to start the boot process.**

Verification—Stopping the System for Recovery Purposes

Stopping the system was successful when you see the `syncing file systems...` message and the `ok` prompt.

Example—Stopping the System for Recovery Purposes

```
Press <Stop-A>
ok sync
syncing file systems...
Press <Stop-A>
ok boot
```

Forcing a Crash Dump and Rebooting the System

Saving crash dumps of the operating system is sometimes necessary for troubleshooting purposes. The `savecore` command is used to enable this feature. It can be turned on automatically by editing the `/etc/init.d/sysetup` script.

The `savecore` feature and how to set it up is described in "Managing System Resources" the *System Administration Guide, Volume II*. This section only describes how to reboot the system if the `savecore` feature is enabled.

▼ SPARC: How to Force a Crash Dump and Reboot the System

1. **Type the stop key sequence for your system. The specific stop key sequence depends on your keyboard type. For example, you can press Stop-A or L1-A. On terminals, press the Break key.**
   The monitor displays the `ok` PROM prompt.

2. **Use the** `sync` **command at the** `ok` **prompt to synchronize the disk and write the crash dump.**

```
> n
ok sync
```

Verification—Forcing a Crash Dump and Rebooting the System

After the crash dump is written to disk, the system will continue to reboot.

Example—Forcing a Crash Dump and Rebooting the System

```
Press <Stop-A>
ok sync
```

▼ SPARC: How to Boot the System Using the Kernel Debugger (`kadb`)

1. **Type the stop key sequence for your system. The specific stop key sequence depends on your keyboard type. For example, you can press Stop-A or L1-A. On terminals, press the Break key.**
   The monitor displays the `ok` PROM prompt.

2. **Use the `sync` command at the `ok` prompt to synchronize the disk and write the crash dump.**

```
> n
ok sync
```

3. **When you see the `syncing file systems...` message, press the abort key sequence for your system again.**

4. **Boot the system using the kernel debugger.**

```
ok boot kadb
```

Example—Booting the System using the Kernel Debugger (`kadb`)

```
Press <Stop-A>
ok sync
syncing file systems...
Press <Stop-A>
ok boot kadb
```

SPARC: Troubleshooting System Problems

Problem — The System is Hung ...	How to Fix the Problem
At run level 2 or 3.	Log in remotely and attempt to identify and kill the process that is hanging the system.
And you can't log in remotely from another machine to attempt recovery.	See "SPARC: How to Stop the System for Recovery Purposes" on page 160.

Problem — The System Won't Boot Because ...	How to Fix the Problem
It can't find `/platform/‘uname -m‘/kernel/unix` , you may need to change the `boot-device` setting in the PROM.	See "SPARC: How to Change the Default Boot Device" on page 149.
A process or an external event like a power failure has caused the system to "hang."	See "SPARC: How to Reset the System" on page 151.
An important system file like `/etc/passwd` has an invalid entry.	See "SPARC: How to Boot a System for Recovery Purposes" on page 158.

Booting an x86 System 9

This chapter describes the procedures for booting an x86 system.

This is a list of the step-by-step instructions in this chapter.

For overview information about the boot process, see Chapter 10, “The Boot Process.”

9

Booting an x86 System

Table 9-1 describes the boot types covered in this chapter.

Table 9-1 Boot Type Descriptions

Booting the System ...	**Is Usually Done ...**	**See an Example On ...**
To run level 3 (multiuser state)	After shutting down the system or performing some system hardware maintenance task. This is the default boot level where all resources are available and users can log into the system.	page 167
To run level S (single-user state)	After performing some system maintenance task such as backing up a file system. At this level only some file systems are mounted and users cannot log into the system.	page 169
Interactively	After making temporary changes to the system file or the kernel for testing purposes. This type of boot allows you to recover easily if there are problems with the system file or kernel by supplying an alternative pathname to these files when prompted. Use the default settings for the other system prompts.	page 171
From local CD-ROM or the network for recovery purposes	To repair an important system file that is preventing the system from booting successfully. This type of boot is also used for installing (or upgrading) a new release of the operating system.	page 174
Using `kadb`	To troubleshoot system problems by using the kernel debugger and saving core dumps of the operating system.	page 175

The following procedures use the reset button to restart the system. If your system does not have a reset button, use the on/off switch to restart the system. You might be able to press the Control-Alt-Del keys to interrupt system operation, depending upon the state of the system.

x86: How to Boot a System to Run Level 3 (Multiuser State)

1. **Press any key to reboot the system if the system displays the** `type any key to reboot` **prompt. Or, use the reset button to restart the system if the system is shut down.**
   The Primary Boot Subsystem menu is displayed after a few minutes.

2. **Select the Active Solaris slice (Part #2 in the example on page 183) as the boot device from the Primary Boot Subsystem menu. Press Return.**
   If you do not make a selection within 30 seconds, the default boot slice is selected automatically. The Secondary Boot Subsystem menu is displayed.

3. **Type** `b` **to boot the system to run level 3. Press Return.**
   If you do not make a selection within 5 seconds, the system is automatically booted to run level 3.

Verification—Booting a System to Run Level 3 (Multiuser State)

If the system successfully boots to run level 3 the login prompt is displayed.

```
hostname console login:
```

Example—Booting a System to Run Level 3 (Multiuser State)

```
type any key to reboot
Please select the partition you wish to boot: 2
Select (b)oot or (i)nterpreter: b
              .
              .
              .
venus console login:
```

▼ x86: How to Boot a System to Run Level S (Single-User State)

1. **Press any key to reboot the system if the system displays the** `type any key to reboot` **prompt. Or, use the reset button to restart the system if the system is shutdown.**
   The Primary Boot Subsystem menu is displayed after a few minutes.

2. **Select the Active Solaris slice (Part #2 in the example on page 183) as the boot device from the Primary Boot Subsystem menu. Press Return.**
   If you do not make a selection within 30 seconds, the default boot slice is selected automatically.

   The Secondary Boot Subsystem menu is displayed.

3. **Type** `b -s` **to boot the system to run level S. Press Return.**
   If you do not make a selection within 5 seconds, the system is automatically booted to run level 3.

4. **Type the root password, if prompted.**

5. **Perform the maintenance task that needed the run level change to s.**

6. **Press Control-d to bring the system back to run level 3.**

Verification—Booting a System to Run Level S (Single-User State)

Use the `who -r` command to verify the system is at run level S.

Example—Booting a System to Run Level S (Single-User State)

```
type any key to reboot
Please select the partition you wish to boot: 2
Select (b)oot or (i)nterpreter: b -s
              .
              .
              .
INIT: SINGLE USER MODE
Type Ctrl-d to proceed with normal startup,
(or give root password for system maintenance):   xxx
# who -r
.       run-level S  Aug  4 13:11     S      0  ?
Perform some maintenance task
#  <Press Control-d>
```

x86: How to Boot a System Interactively

1. Boot the system interactively.

```
ok boot -a
```

2. Answer the system prompts as described in Table 9-2.

Table 9-2 Interactive Boot Procedure Steps

If the System Displays ...	Do the Following ...
`type any key to reboot`	Press any key to reboot the system, or use the reset button to restart the system. The Primary Boot Subsystem menu is displayed after a few minutes.
The Primary Boot Subsystem menu	Select the Active Solaris slice as the boot device. Press Return. If you do not make a selection within 30 seconds, the default boot slice is selected automatically.
`Select (b)oot or (i)nterpreter:`	Type `b -a` and press Return.
`Enter filename [kernel/unix]:`	Provide the name of another kernel to use for booting and press Return. Or, press Return to use the default kernel (`/platform/i86pc/kernel/unix`).
`Name of system file [etc/system]:`	Provide the name of an alternate system file and press Return, or press Return to use the default `/etc/system` file.
`Name of default directory for modules [/platform/i86pc/kernel /kernel /usr/kernel]:`	Provide an alternate path for the modules directory and press Return, or press Return to use the default modules directory path.
`root filesystem type [ufs]:`	Press Return to use the default root file system type: UFS for local disk booting or NFS for diskless clients.
`Enter physical name of root device[`*physical_device_name*`]:`	Provide an alternate device name and press Return, or press Return to use the default physical name of the root device.

Verification—Booting a System Interactively

The system booted interactively if you are prompted to answer the questions listed in Table 9-2.

Example—Booting a System Interactively

In the following example, the default choices (shown in square brackets []) are accepted.

```
Select (b)oot or (i)nterpreter: b -a
(Copyright notice)
Enter filename [kernel/unix]:Return
Name of system file [/etc/system]:Return
Name of default directory for modules [platform/i86pc/kernel /kernel
/usr/kernel]:> Return
root filesystem type [ufs]: Return
Enter physical name of root device
[/eisa/dpt@5c88,0/cmdk@0,0:a]: Return
Configuring network interfaces:  smc0
Hostname: venus
(fsck messages)
The system is coming up.  Please wait
(More messages)
venus console login:
```

▼ x86: How to Boot a System for Recovery Purposes

Recovering from a invalid `/etc/passwd` file is used as an example of how to boot a system for recovery purposes.

Substitute the device name of the file system to be repaired for the *devicename* variable identified in the procedures below. If you need help identifying a system's device names, refer to Chapter 26, "Accessing Devices" in *System Administration Guide, Volume I.*

Follow the instructions below depending on whether you are booting from the Solaris 2.x installation CD or the network.

1. **Boot from the Solaris 2.x installation CD (or the network) to single-user mode using steps a-f.**
   If you are booting from the network, skip steps a and b.

   a. **Insert the Solaris 2.x installation CD into the CD caddy.**

   b. **Insert the CD caddy into the CD-ROM drive.**

   c. **Insert the Solaris boot diskette into the primary diskette drive (DOS drive A).**

   d. **Press any key to reboot the system if the system displays the** `type any key to reboot` **prompt. Or, use the reset button to restart the system if the system is shutdown.**
   The Multiple Device Boot Subsystem menu is displayed after a few minutes.

   e. **Select the CD-ROM drive or net(work) as the boot device from the Multiple Device Boot menu.**
   The Secondary Boot Subsystem menu is displayed.

   f. **Type** `b -s` **at the** `Select the type of installation:` **prompt.**

   After a few minutes, the single-user mode # prompt is displayed.

2. **Mount the root (/) file system that has the invalid** `passwd` **file.**

```
# mount /dev/dsk/devicename /a
```

3. **Change to the newly mounted `etc` directory.**

```
# cd /a/etc
```

4. **Set the terminal type.**

```
# TERM=AT386
# export TERM
```

5. **Make the necessary change to the `passwd` file using an editor.**

```
# vi passwd
```

6. **Change to the root (`/`) directory.**

```
# cd /
```

7. **Unmount the `/a` directory.**

```
# umount /a
```

8. **Reboot the system.**

```
# init 6
```

Verification—Booting a System for Recovery Purposes

The procedure is successful if the system successfully boots to run level.

Example—Booting a System for Recovery Purposes

```
type any key to reboot
Enter the boot device code: 11
Select the type of installation: b -s
# mount /dev/dsk/rdsk/c0t3d0s0 /a
# cd /a/etc
# TERM=AT386
# export TERM
# vi passwd
# cd /
# umount /a
# init 6
```

▼ x86: How to Stop the System for Recovery Purposes

The specific stop key sequence depends on your system type. For example, press the reset button to stop the system. If your system doesn't have a reset button, turn the power off and back on again.

Forcing a Crash Dump and Rebooting the System

Saving core dumps of the operating system is sometimes necessary for troubleshooting purposes. The `savecore` command is used to enable this feature. It can be turned on automatically by editing the `/etc/init.d/sysetup` script.

The `savecore` feature and how to set it up is described in "Managing System Resources" in *System Administration Guide, Volume II*. This section only describes how to reboot the system if the `savecore` feature is enabled.

▼ x86: How to Force a Crash Dump and Reboot the System

1. **Press Control-Alt-d.**

```
kadb>
```

The `kadb>` prompt is displayed.

2. **Type** `0:c` **at the** `kadb>` **prompt.**

```
kadb> 0:c
```

3. **Type** `:c` **at the** `kadb>` **prompt.**

```
kadb> :c
```

Verification—Forcing a Crash Dump and Rebooting the System

After the crash dump is written to disk, the system will continue to reboot.

Example—Forcing a Crash Dump and Rebooting the System

```
<Press Control-Alt-d>
kadb> 0:c
kadb> :c
```

9

x86: Troubleshooting System Problems

Problem — The System is Hung ...	How to Fix the Problem
At run level 2 or 3.	Log in remotely and attempt to identify and kill the process that is hanging the system.
And you can't log in remotely from another machine to attempt recovery.	See "x86: How to Stop the System for Recovery Purposes" on page 174.

Problem — The System Won't Boot Because ...	How to Fix the Problem
There is no default boot device. The message displayed is: `Not a UFS filesystem`	Boot the system using the Solaris boot diskette and select the disk from which to boot.
A process or an external event like a power failure has caused the system to "hang."	See "x86: How to Stop the System for Recovery Purposes" on page 174.
An important system file like `/etc/passwd` has an invalid entry.	See "x86: How to Boot a System for Recovery Purposes" on page 172.

The Boot Process 10

This chapter describes the hardware used for booting on SPARC and x86 systems and a conceptual overview of the boot process on each platform.

This is a list of overview information in this chapter.

For instructions on booting a system, see Chapter 8, “Booting a SPARC System,” or Chapter 9, “Booting an x86 System.”

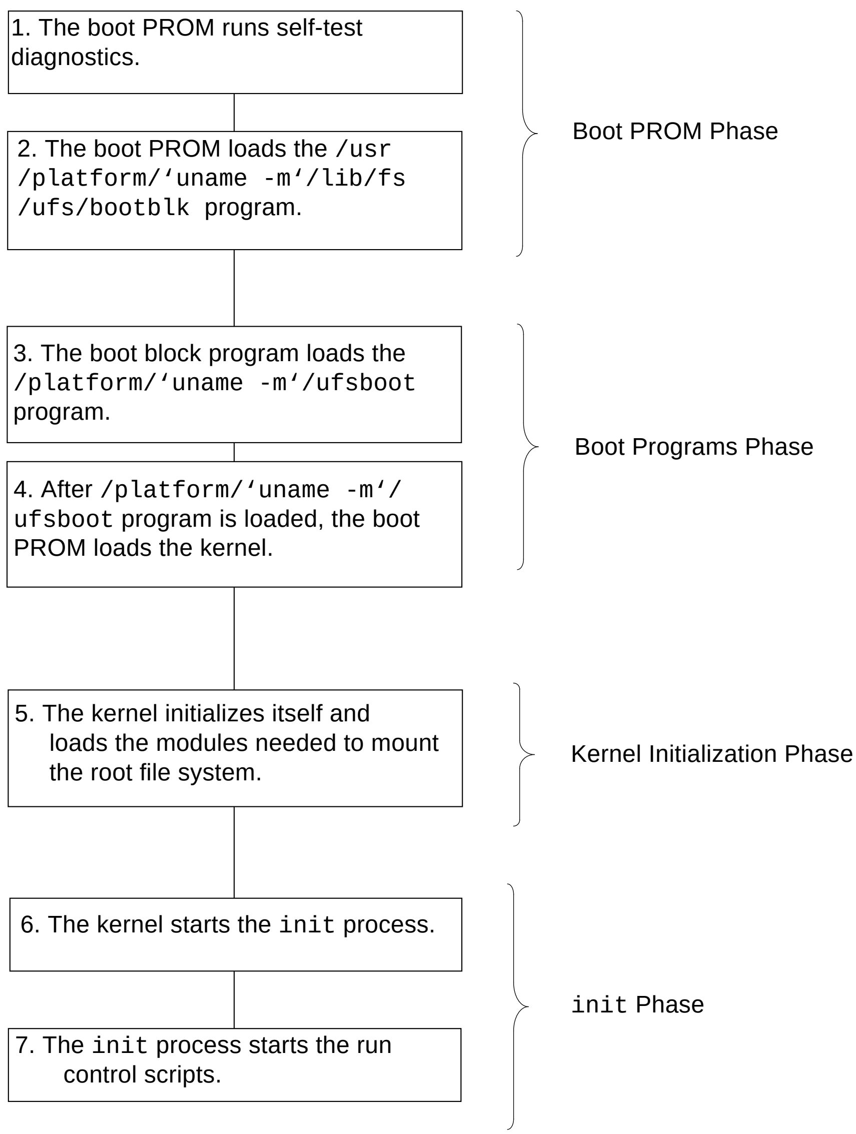
10

SPARC: The Boot PROM

Each SPARC system has a PROM (programmable read-only memory) chip with a program called the *monitor*. The monitor controls the operation of the system before the kernel is available. When a system is turned on, the monitor runs a quick self-test procedure that checks things such as the hardware and memory on the system. If no errors are found, the system begins the automatic boot process.

Note – Some older systems may require PROM upgrades before they will work with the Solaris system software. Contact your local service provider for more information.

SPARC: The Boot Process



SPARC: The Boot Process Details

Table 10-1 describes the SPARC boot process illustrated on the previous page.

Table 10-1 Description of SPARC Boot Process

Boot Phase	Description
Boot PROM	1. The PROM displays system identification information and then runs self-test diagnostics to verify the system's hardware and memory.
	2. Then the PROM loads the primary boot program, `/usr/platform/‘uname m‘/lib/fs/ufs/bootblk`, whose purpose is to load the secondary boot program located in the `ufs` file system from the default boot device.
Boot Programs	3. The `bootblk` program finds and executes the secondary boot program, `/platform/‘uname -m‘ /ufsboot`, and loads it into memory.
	4. After `/platform/‘uname -m‘/ufsboot` program is loaded, the boot PROM loads the kernel.
Kernel Initialization	5. The kernel initializes itself and begins loading modules, using `ufsboot` to read the files. When the kernel has loaded enough modules to mount the root file system, it unmaps the `/ufsboot` program and continues, using its own resources.
`init`	6. The kernel creates a user process and starts the `/sbin/init` process, which starts other processes by reading the `/etc/inittab` file.
	7. The `/sbin/init` process starts the run control (`rc`) scripts, which execute a series of other scripts. These scripts (`sbin/rc*`) check and mount file systems, start various processes, and perform system maintenance tasks.

x86: The PC BIOS

Before the kernel is started, the system is controlled by the read-only-memory (ROM) Basic Input/Output System (BIOS), the firmware interface on a PC.

Hardware adapters can have an onboard BIOS that displays the physical characteristics of the device and can be used to access the device.

During the startup sequence, the PC BIOS checks for the presence of any adapter BIOS and if found, loads and executes each one. Each individual adapter's BIOS runs self-test diagnostics and displays device information.

x86: Boot Subsystems

Booting an x86 system uses these subsystems:

- Primary boot subsystem
- Secondary boot subsystem

Additionally, booting from the network or a local CD-ROM drive uses this subsystem:

- Solaris boot diskette

Table 10-2 describes the three boot interfaces that are used to boot all levels on an x86 system.

Table 10-2 x86 Boot Subsystems

Boot Subsystem	**This Subsystem Menu Displays ...**
Solaris Boot Diskette	A list of bootable devices such as disk, network, or CD-ROM.
Primary Boot Subsystem	A list of `fdisk` partitions to boot from if there is more than one bootable `fdisk` partition.
Secondary Boot Subsystem	A list of boot options. The system automatically boots to run level 3 if you don't select an option (after a five-second time out) from this menu. The other options enable you to specify boot options or enter the boot interpreter (see `boot(1M)`).

During the boot process, the boot subsystem menus display different device and booting options. If the system receives no response after several time-out periods, it continues to boot automatically using default selections. You can stop the boot process when the boot subsystem menus are displayed or let it continue automatically.

The following section provides examples of each subsystem screen. Screen displays will vary based on system configurations.

Solaris Boot Diskette

This diskette is also known as the multiple device boot (MDB) diskette.

```
Solaris x86 -  MDB                                          Version 1.55

                 Solaris/x86 Multiple Device Boot Menu

            Code   Device    Vendor    Model/Desc          Rev
            ====================================================

            10     DISK      MAXTOR    LXT-535S            8.75
            11     CD        SONY      CD-ROM CDV-8012     3.1d
            12     NET       SMC/WD    I/O=300 IRQ=5

            Enter the boot device code:

30
```

During this phase, the system probes currently connected devices and displays the devices in the Multiple Device Boot menu.) The first device listed is the default boot device.

Primary Boot Subsystem

```
SunOS 5.5 for x86                            Primary Boot Subsystem, vsn 2.0

     Loading master boot record from specified device....

                  Current Disk Partition Information

            Part#    Status    Type        Start        Length
            =====================================================
              1                DOS16          32         61408
              2      Active    SOLARIS     61440        983040
              3
              4

            Please select the partition you wish to boot: 2
30
```

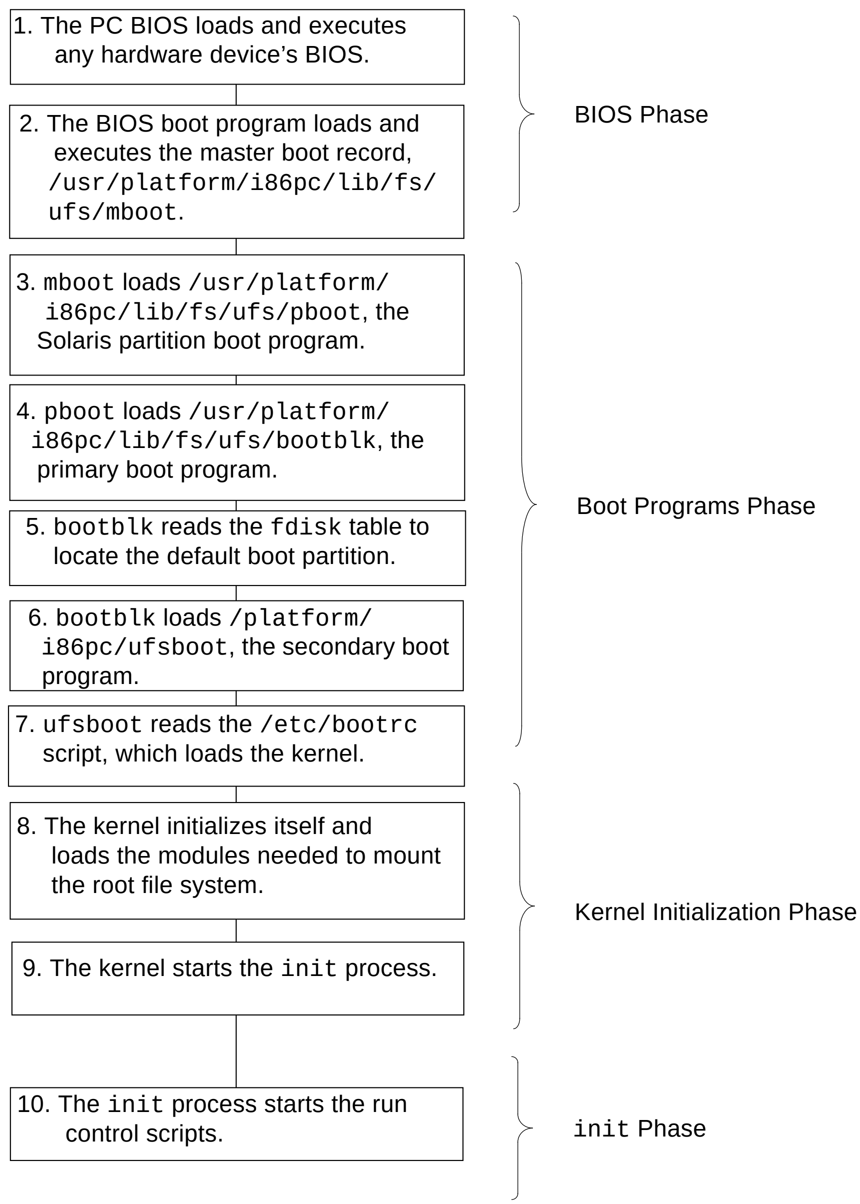
The active `fdisk` partition is automatically selected if you press Return or do not select an alternate slice from which to boot (after a 30-second time out).

Secondary Boot Subsystem

The second boot subsystem menu displays available boot options.

```
Solaris 2.5 for x86                    Secondary Boot Subsystem, vsn 2.11

                    <<< Current Boot Parameters >>>
 Boot path: /eisa/dpt@5c88,0/cmdk@0,0:a
 Boot args: /kernel/unix

 Type    b [file-name] [boot-flags] <ENTER>   to boot with options
 or      i <ENTER>                            to enter boot interpreter
 or      <ENTER>                              to boot with default

                      <<< timeout in 5 seconds >>>

 Select (b)oot or (i)nterpreter: b
```

x86: The Boot Process



x86: The Boot Process Details

Table 10-3 describes the x86 boot process illustrated on the previous page.

Table 10-3 Description of x86 Boot Process

Boot Phase	Description
BIOS	1. When the system is turned on, the PC BIOS runs self-test diagnostics to verify the system's hardware and memory. The system begins to boot automatically if no errors are found. If errors are found, error messages are displayed describing recovery options. Additional hardware devices' BIOS are run at this time.
	2. The BIOS boot program tries to read the first physical sector from the boot device—either a diskette or hard drive. This first disk sector on the boot device contains the master boot record `/usr/platform/i86pc/lib/fs/ufs/mboot`, which is loaded and executed. If no `mboot` file is found, an error message is displayed.
Boot Programs	3. `mboot`, which contains disk information needed to find the active partition and the location of the Solaris boot program, `/usr/platform/i86pc/lib/fs/ufs/pboot`, loads and executes `pboot`.
	4. `pboot` loads `/usr/platform/i86pc/lib/fs/ufs/bootblk`, the primary boot program, whose purpose is to load the secondary boot program located in the `ufs` file system.
	5. If there is more than one bootable partition, `bootblk` reads the `fdisk` table to locate the default boot partition, and builds and displays a menu of available partitions. This step only occurs if there is more than one bootable partition present on the system.
	6. `bootblk` finds and executes `/platform/i86pc/ufsboot`, the secondary boot program in the root file system.
	7. `ufsboot` starts a command interpreter that executes the `/etc/bootrc` script, which provides a menu of choices for booting the system. The default action is to load and execute the kernel.
Kernel Initialization	8. The kernel initializes itself and begins loading modules, using `ufsboot` to read the files. When the kernel has loaded enough modules to mount the root file system, it unmaps the `/platform/i86pc/ufsboot` program and continues, using its own resources.
	9. The kernel creates a user process and starts the `/sbin/init` process, which starts other processes by reading the `/etc/inittab` file.
`init`	10. The `/sbin/init` process starts the run control (`rc`) scripts, which execute a series of other scripts. These scripts (`sbin/rc*`) check and mount file systems, start various processes, and perform system maintenance tasks.

Part 4 — Managing Removable Media

This part provides instructions for using removable media in the Solaris environment.

Guidelines for Using CDs and Diskettes 11

This chapter provides general guidelines for using diskettes and CDs in the Solaris environment.

This is a list of overview information in this chapter.

Features and Benefits

The Solaris environment gives users and software developers a standard interface for dealing with diskettes and CDs. Referred to as Volume Management, this interface provides three major benefits:

- By automatically mounting diskettes and CDs, it simplifies their use. (For a comparison between manual and automatic mounting, see Figure 11-1 on page 190.)
- It enables you to access diskettes and CDs without having to act as root.
- It allows you to give other systems on the network automatic access to any diskettes and CDs you insert into your system (see Chapter 13, “Using CDs From the Command Line” and Chapter 14, “Formatting and Using Diskettes From the Command Line”).

Comparison of Automatic and Manual Mounting

Figure 11-1 compares the steps involved in manual mounting (without Volume Management) and automatic mounting (with Volume Management).

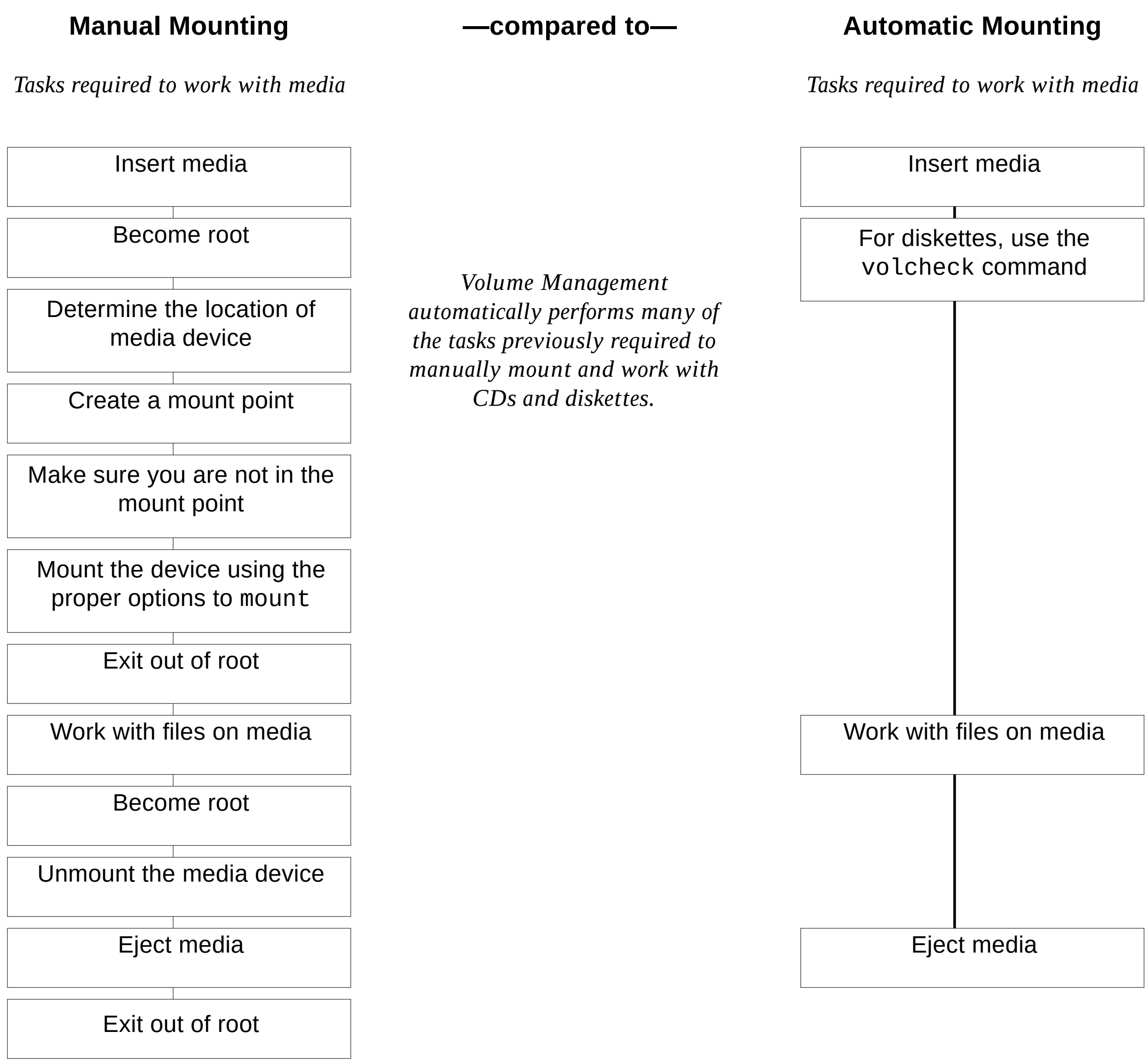
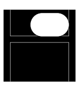
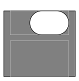
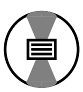


Figure 11-1 Comparison of Manual and Automatic Mounting

What You Can Do With Diskettes and CDs

Essentially Volume Management enables you to access diskettes and CDs just as manual mounting does, but more easily and without the need for root access. To make diskettes and CDs easier to work with, they are mounted in easy-to-remember locations. As a result, ...

To Access ...	Insert ...	And Find The Files In ...
 *Files on a diskette*	*The diskette and enter* `volcheck`	/floppy/floppy0
 *Raw characters on a diskette*	*The diskette and enter* `volcheck`	/vol/dev/aliases/floppy0
 *Files on a CD*	*The CD and wait a few seconds*	/cdrom/cdrom0

If your system has more than one diskette or CD-ROM drive, see Table 11-1 for their access points.

Table 11-1 Where to Access Diskettes and CDs

Media Device	Access File Systems On	Access Raw Data On
First diskette drive	`/floppy/floppy0`	`/vol/dev/aliases/floppy0`
Second diskette drive	`/floppy/floppy1`	`/vol/dev/aliases/floppy1`
First CD-ROM drive	`/cdrom/cdrom0`	`/vol/dev/aliases/cdrom0`
Second CD-ROM drive	`/cdrom/cdrom1`	`/vol/dev/aliases/cdrom1`

You can perform most diskette- and CD-related tasks knowing only the above information. However, step-by-step instructions for a variety of tasks are provided in subsequent chapters. A summary of those tasks, including which can be performed with File Manager and which must be performed from the command line, is provided in Figure 11-2 on page 192.

Available with File Manager?	Tasks Related to CDs	Available through Command Line?
page 195	▼ How to Load a CD	page 203
page 197	▼ How to Examine the Contents of a CD	page 204
page 198	▼ How to Copy Information from a CD	page 204
No	▼ How to Find Out If a CD Is Still in Use	page 206
page 195	▼ How to Eject a CD	page 207
No	▼ How to Access CDs on Other Systems	page 207
No	▼ How to Make Local CDs Available to Other Systems	page 209
No	▼ How to Configure a System to Play Musical CDs	page 213
No	▼ How to Prepare a System for a New CD-ROM Drive	page 214
No	▼ How to Stop Volume Management	page 215
No	▼ How to Restart Volume Management	page 215
	Tasks Related to Diskettes	
No	▼ How to Format a UFS Diskette	page 222
No	▼ How to Place a UFS File System on a Diskette	page 225
No	▼ How to Format a DOS Diskette	page 228
page 194	▼ How to Load a Diskette	page 232
page 197	▼ How to Examine the Contents of a Diskette	page 233
page 199	▼ How to Copy or Move Information From a Diskette	page 234
page 199	▼ How to Copy or Move Information To a Diskette	page 235
No	▼ How to Find Out If a Diskette Is Still in Use	page 237
page 195	▼ How to Eject a Diskette	page 238
No	▼ How to Access Diskettes on Other Systems	page 239
No	▼ How to Make Local Diskettes Available to Other Systems	page 241

Figure 11-2 What You Can Do With CDs and Diskettes

Using CDs and Diskettes Through File Manager 12

Although you can perform only some media-related tasks with File Manager, if you are accustomed to graphical user interfaces, you may prefer to access your media through its menus. This is a list of the step-by-step instructions in this chapter.

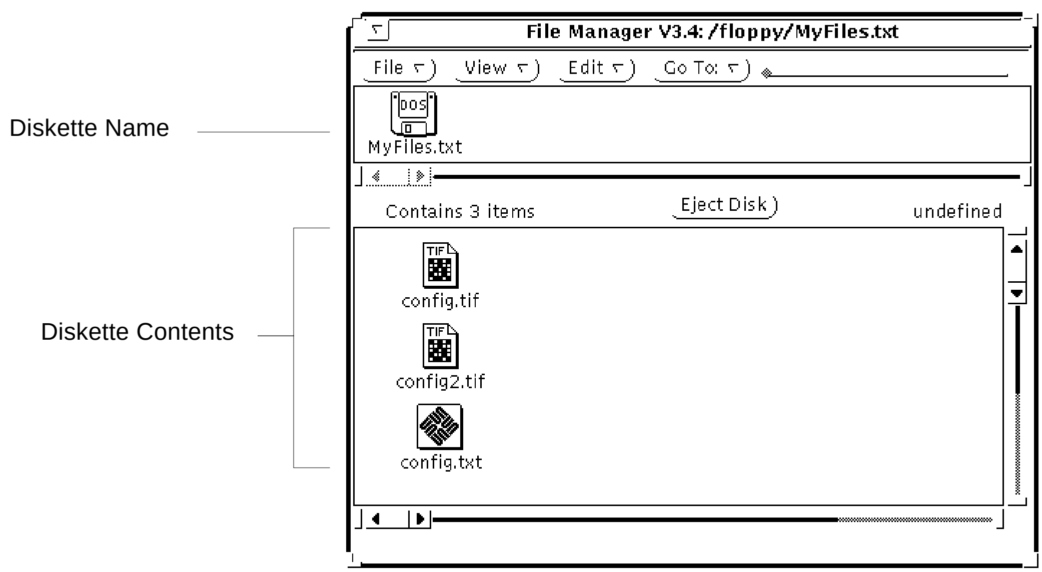
For additional tasks, see Chapter 13, “Using CDs From the Command Line” or Chapter 14, “Formatting and Using Diskettes From the Command Line.”

▼ How to Load a Diskette

1. **Make sure the diskette is formatted.**
   If you aren't sure, don't worry. When you insert the diskette, File Manager will display a large screen notifying you if the diskette is unformatted. To format the diskette, see "How to Format a UFS Diskette" on page 222 or "How to Format a DOS Diskette" on page 228.

2. **Insert the diskette.**
   Make sure the diskette is completely inserted. It needs to drop down into the drive.

3. **Select Check for Floppy from the File menu.**
   If the floppy is inserted properly, a window displaying the contents of the floppy disk is displayed.

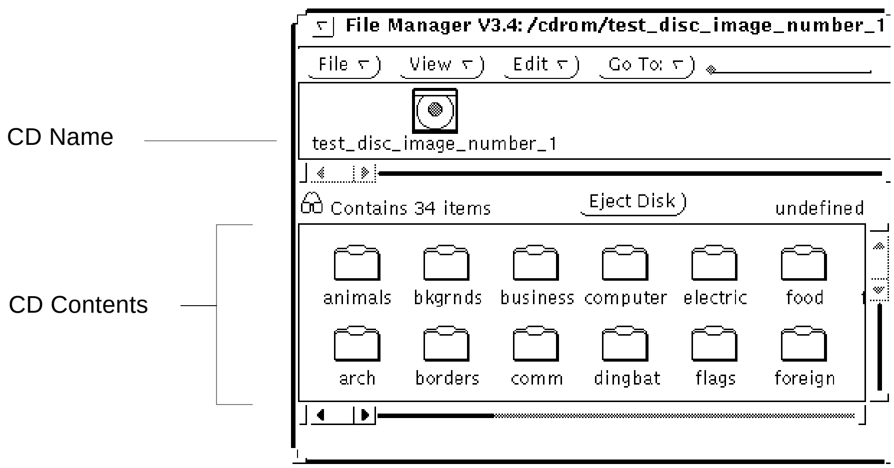


If the floppy is not inserted properly, this message appears at the bottom of the File Manager window:

```
No floppy disk in drive
```

▼ How to Load a CD

Simply insert the CD, in its caddy, into the CD-ROM drive. The system may take a few seconds to recognize the CD. Once it does, a window displaying the contents of the CD automatically appears:



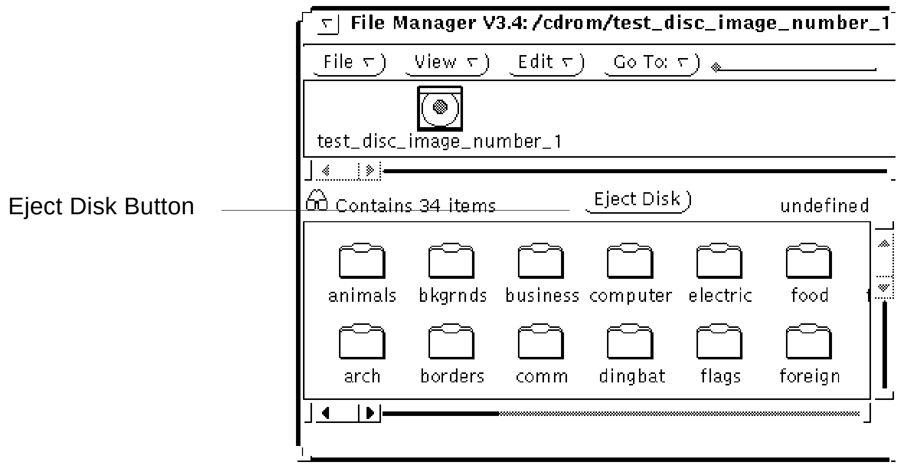
▼ How to Eject a Diskette or CD

The processes for ejecting a diskette and a CD are almost identical. The only difference is minor, as described in Step 2.

1. **If ejecting a diskette, make sure the diskette is not being used.**
   If ejecting a CD, skip this step. Remember, a diskette is “being used” if a shell or an application is accessing any of its files or directories. File Manager doesn’t count; even if File Manager is displaying the contents of the media, you can still eject it without harm—File Manager will revert to displaying the contents of your home directory.

   If you are not sure whether you have found all users of the diskette (a renegade shell hidden behind a desktop tool may be accessing it), use the `fuser` command, as described in “How to Find Out If a Diskette Is Still in Use” on page 237.

2. **Select the Eject Disk button in the Diskette Window or CD Window.**



If the CD Window no longer appears on the screen, enter `eject cdrom` in a shell tool. If the Diskette Window no longer appears, enter `eject diskette`.

The CD is automatically ejected from its drive on both SPARC and x86 systems. On a SPARC system the diskette is physically ejected from its drive. On an x86 system, you'll have to eject the diskette by hand. Look for an onscreen message that says you can now eject the diskette. If the diskette jams, eject it manually by inserting an unfolded paper clip about an inch into the small hole in the front of the drive.

After the media is ejected, its window will disappear automatically. If the contents of the media were being displayed in File Manager's main window, they are replaced by the contents of your home directory.

Accessing Files on CDs and Diskettes

Once they are recognized by Volume Management, you can access the files on diskettes and CDs just as you would files on any other file system. The only restrictions to be aware of are ownership and permissions. For instance, if you are not the owner of a file on a diskette, you won't be able to overwrite it on the diskette. Or, if you copy a file from a CD into your file system, you'll be the owner, but the file won't have write permissions (because it never had them on the CD); you'll have to change the permissions yourself.

Taking these restrictions into account, you can perform the following operations with the files and directories on CDs and diskettes:

- Examine the contents of a CD or diskette
- Copy information from a CD or diskette
- Move information between a diskette and a system

▼ How to Examine the Contents of a Diskette or CD

Simply load the diskette or CD, according to the instructions in the following tasks:

- "How to Load a Diskette" on page 194
- "How to Load a CD" on page 195.

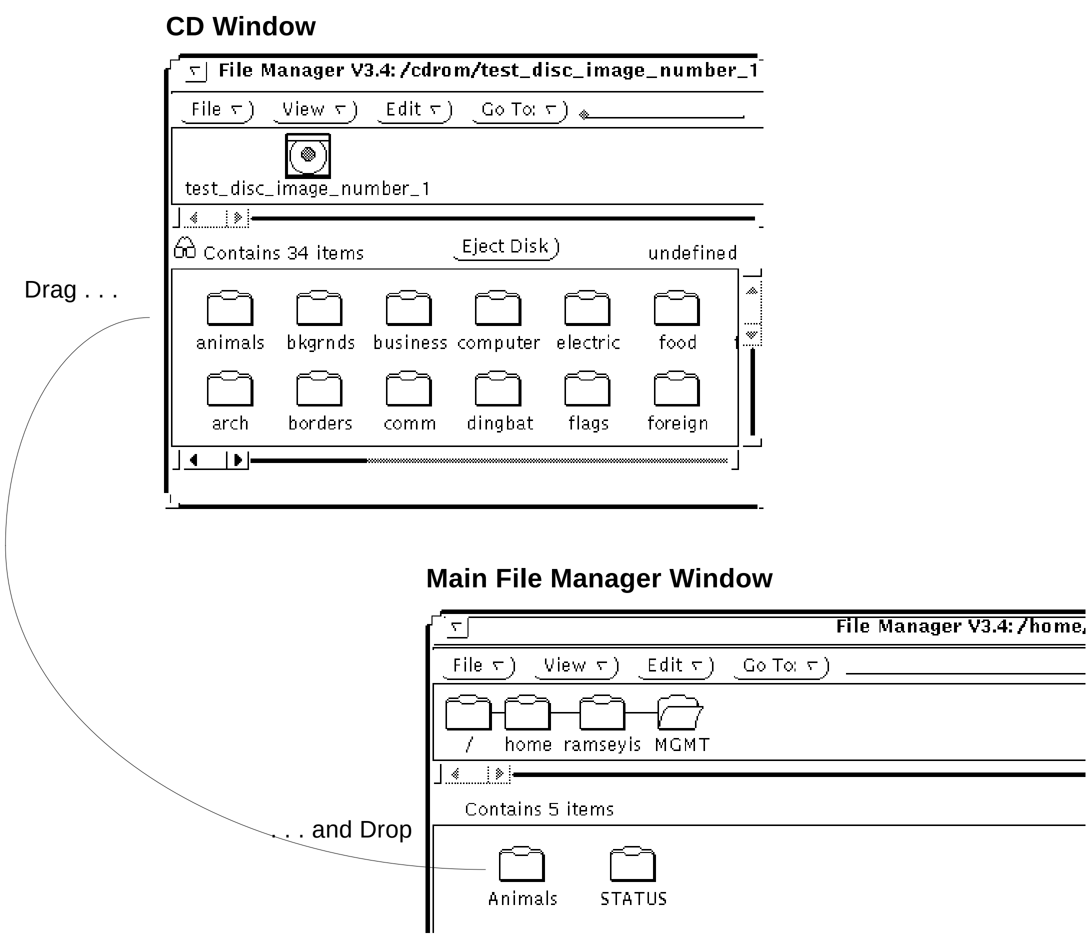
The contents of the diskette or CD are displayed in their own individual windows.

▼ How to Copy Information from a CD

1. **Check permissions.**
   Make sure you have write permission for the directory into which you will copy the CD information.

2. **Insert the CD.**
   For instructions, see "How to Load a CD" on page 195. The CD Window appears.

3. **Drag and drop.**
   Simply drag and drop the files or directories from the CD to a location in File Manager.



▼ How to Move Information Between a Diskette and a System

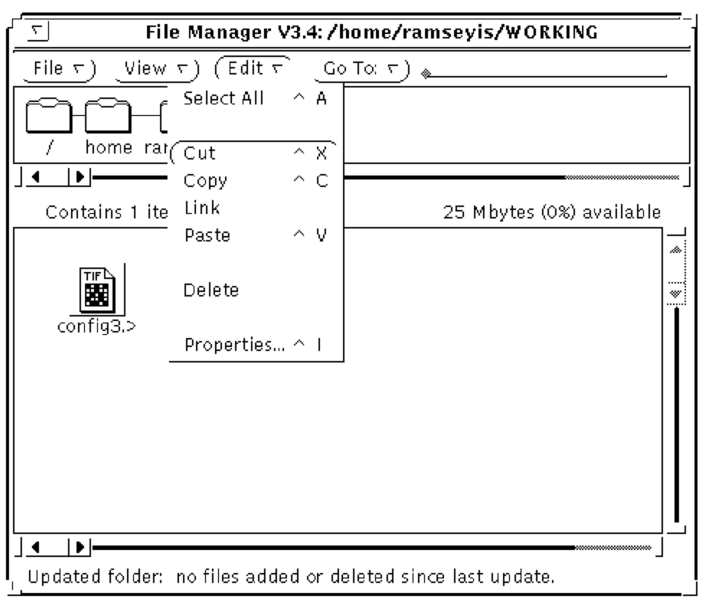
This procedure can be used to move files or folders into or out of a diskette. It does not apply to CDs since you cannot write to a CD.

1. **Check permissions.**
   Make sure you have read permission for the source (the old location of the information) and write permission for the target (the new location of the information).

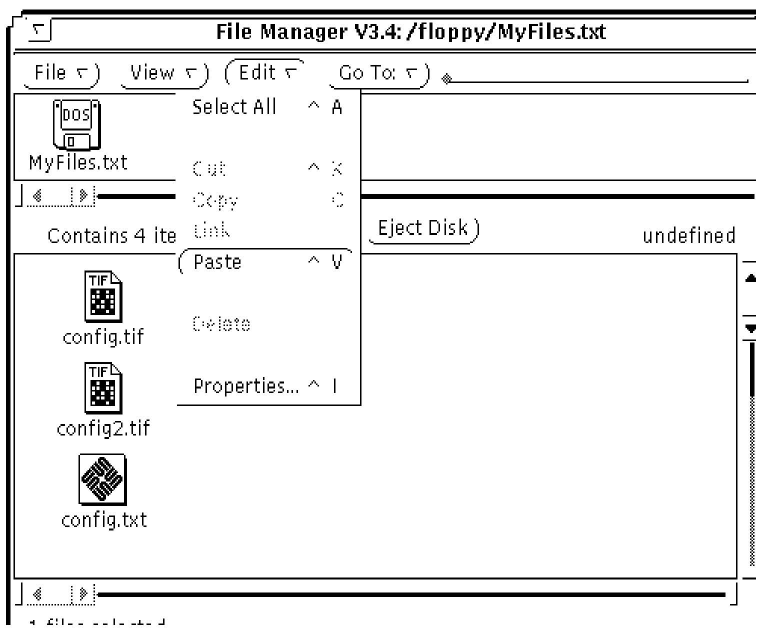
2. **Open the diskette window.**
   This window appears when you insert a diskette and select the Check for Floppy command.

3. **Select the file or folder you want to move.**
   Simply click it with the left mouse button.

4. **Select Cut from the Edit menu in the source window.**
   The source window is simply the window that contains the file or folder you want to move. In this example, it is the main File Manager window:



5. **Select Paste from the Edit menu of the target window.**
The target window is the window into which you want to move the file or folder. In this example it is the diskette window.



The new file or folder is displayed in the target window.

Using CDs From the Command Line 13

This chapter describes all the tasks required to use CDs in the Solaris environment from the command line. This is a list of the step-by-step instructions in this chapter.

13

Using CDs

Table 13-1 Task Map: How CDs Are Typically Used

Activity	Description	For Instructions, Go To	
Load the CD	Insert the CD into the CD-ROM drive.	▼ How to Load a CD	page 203
Examine its Contents	Optional. To examine the contents of the CD, look in the appropriate directory under `/cdrom`.	▼ How to Examine the Contents of a CD	page 204
Copy Files or Directories	Optional. Copy files or directories from the CD as you would from any other location in the file system.	▼ *How to Copy Information From a CD*	*page 201*
Is CD Still in Use?	Optional. Before ejecting the CD, find out if it is still in use.	▼ How to Find Out If a CD is Still in Use	page 206
Eject the CD	When you finish, eject the CD.	▼ How to Eject a CD	page 207

Using CD Names

When working with CDs, you can identify them by name or with a designator from Table 13-2, below. For brevity, task descriptions use `cdrom0`, but you can replace it with either the CD name or a different designator.

Table 13-2 How to Identify CDs

CD	Alternate Name
First CD-ROM drive	`cdrom0`
Second CD-ROM drive	`cdrom1`
Third CD-ROM drive	`cdrom2`

▼ How to Load a CD

Insert the CD. Shortly after the light stops flashing (about five to ten seconds), the CD is mounted to `/cdrom`. To verify that the CD is mounted, perform the task titled, "How to Examine the Contents of a CD," below.

Note – Most CDs are formatted to the ISO 9660 standard, which is portable, so most CDs can be mounted by Volume Management. However, as described in Chapter 16, "How Volume Management Works," UFS CDs are not portable between architectures, so they must be used on the architecture for which they were designed. If you are having trouble mounting a CD, particularly if it is an installation CD, make sure its UFS file system is appropriate for your system's architecture (check the label on the CD).

▼ How to Examine the Contents of a CD

Use the `-L` option to the `ls` command, because some directories under `/floppy` and `/cdrom` are symbolic links.

```
$ ls -L [-l] /cdrom/cdrom0
```

In this command,

`-L`	Includes symbolic links in the output.
`-l`	Long format. Includes permissions and owners in the output.

Example—Examining the Contents of a CD

The following example lists the contents of the CD loaded into the first CD-ROM directory, `/cdrom/cdrom0`.

```
$ ls -L -l /cdrom/cdrom0
dr-xr-xr-x   2  root   sys  2048  Dec 31  1993  tools
dr-xr-xr-x   2  root   sys  2048  Dec 31  1993  graphics
```

▼ How to Copy Information From a CD

You can access a CD's files and directories just like any other file system. The only restrictions to be aware of are ownership and permissions. For instance, if you copy a file from a CD into your file system, you'll be the owner, but you won't have write permissions (because the file never had them on the CD); you'll have to change the permissions yourself.

1. Make sure the CD is mounted.

```
$ ls /cdrom
```

The `ls` command displays the contents of a mounted CD. If no contents are displayed, see the task titled, "How to Load a CD" on page 203.

2. Copy the files or directories.

To Copy ...	Use ...
A file	`cp`
A directory	`cp -r`

Example—Copying Information From a CD

Following are two examples. The first example uses `cp` to copy a single file from the `animals` directory of `cdrom0` into the system's working directory (denoted by the "`.`"). The second example uses `cp -r` to copy the entire `animals` directory into the system's working directory. In both instances, the `ls` command is used to verify the copy operation.

```
$ cp /cdrom/cdrom0/animals/dog.vec .
$ ls -l
-r-xr-xr-x   2  smith   sys  2048  Dec 31  1994  dog.vec

$ cp -r /cdrom/cdrom0/animals .
$ ls -l
dr-xr-xr-x   2  smith   sys  512   Dec 31  1994  animals
-r-xr-xr-x   2  smith   sys  2048  Dec 31  1994  dog.vec
```

Note that when the CD file or directory is copied into your file system, you become its owner, but it retains the permissions it had on the CD:

```
-r-xr-xr-x
```

To overwrite it, you'll need to change the permissions with `chmod`.

▼ How to Find Out If a CD is Still in Use

1. **Become root.**
2. **Enter the `fuser` command.**
   The `fuser` command lists the processes that are currently accessing the CD that you specify.

```
# fuser -u [-k] /cdrom/cdrom0
```

In this command,

`-u`	Displays the user of the CD.
`-k`	Kills the process accessing the CD.

Example—Finding Out If a CD is Still in Use

In the following example, the processes `6400c` and `6399c` are accessing the `/cdrom/cdrom0` directory, and the process owners are `root` and `smith`, respectively.

```
# fuser -u /cdrom/cdrom0
/cdrom/cdrom0: 6400c(root)  6399c(smith)
```

You can kill the processes individually (as root), or you can use the `fuser` command with the `-k` option, which kills all the processes accessing that file system, as shown in the following example.

```
# fuser -u -k /cdrom/cdrom0
/cdrom/cdrom0: 6400c(root)Killed  6399c(smith)Killed
```

The `fuser` command may not always identify all the killed processes. To be sure, run it again with the `-u` option.

▼ How to Eject a CD

1. **Make sure the CD is not being used.**
   Remember, a CD is "being used" if a shell or an application is accessing any of its files or directories. If you are not sure whether you have found all users of a CD (a renegade shell hidden behind a desktop tool may be accessing it), use the `fuser` command, as described in "How to Find Out If a CD is Still in Use" on page 206.

2. **Eject the CD.**

```
$ eject cdrom0
```

▼ How to Access CDs on Other Systems

You can access a CD on another system by mounting it manually into your file system—provided the other system has exported its CD-ROM according to the instructions in "How to Make Local CDs Available to Other Systems" on page 209.

1. **Select an existing directory to serve as the mount point or create one.**

```
$ mkdir directory
```

   In this command,

directory	Is the name of the directory that you create to serve as a mount point for the other system's CD.

2. **Find the name of the CD you want to mount.**
   When you manually mount a remote CD, you cannot use the `cdrom0` or `cdrom1` variables available with your local CDs. You must use the exact CD name. To find it, use the `ls` command on the remote system's `/cdrom` directory. If the automounter is running, you can simply `cd` to the system whose CD you want to mount and then use the `ls` command. If the automounter is not running, you'll have to use another method, such as logging in remotely.

3. As root, mount the CD.

```
# mount -F nfs -o ro system-name:/cdrom/cd-name local-mount-point
```

In this command,

system-name	Is the name of the system whose CD you will mount.
cd-name	Is the name of the CD you want to mount.
local-mount-point	Is the local directory onto which you will mount the remote CD.

Verification—Accessing CDs on Other Systems

To verify that the CD is indeed mounted, use the `ls` command to list the contents of the mount point.

```
$ ls /cdrom
```

Example—Accessing CDs on Other Systems

This example mounts the CD named `Solaris_2.5_Install` from the remote system `mars` onto the `/cdrom` directory of the local system.

```
$ cd /net/mars
$ ls /cdrom
cdrom0     Solaris_2.5_Install
$ su
Password: password
# mount -F nfs ro mars:/cdrom/Solaris_2.5_Install /cdrom
# exit
$ ls /cdrom
Solaris_2.5_Install
```

How to Make Local CDs Available to Other Systems

You can configure your system to share its CD-ROM drives; in other words, make any CDs[1] in those drives available to other systems. Once your CD-ROM drives are shared, other systems can access the CDs they contain simply by mounting them, as described in "How to Access CDs on Other Systems" on page 207.

1. **Become root.**

2. **Find out whether the NFS daemon (`nfsd`) is running.**

```
# ps -ef | grep nfsd
root 14533    1 17 10:46:55 ?      0:00 /usr/lib/nfs/nfsd -a 16
root 14656  289  7 14:06:02 pts/3 0:00 /grep nfsd
```

If the daemon is running, a line for `/usr/lib/nfs/nfsd` will appear, as shown above. If the daemon is not running, only the `/grep nfsd` line will appear.

3. **Select an option from the following table.**

If ...	Then ...
`nfsd` is running	Go to Step 8
`nfsd` is *not* running	Continue with Step 4

4. **Create a dummy directory for `nfsd` to share.**

```
# mkdir /dummy-dir
```

In this command,

dummy-dir — Can be any directory name; for example, `dummy`. This directory will not contain any files. Its only purpose is to "wake up" the NFS daemon so that it notices your shared CD-ROM.

1. Except musical CDs.

5. **Add the following entry into** `/etc/dfs/dfstab`**.**

```
share -F nfs -o ro [-d comment] /dummy-dir
```

When you start the NFS daemon, it will see this entry, "wake up," and notice the shared CD-ROM drive. Note that the comment (preceded by `-d`) is optional.

6. **Start the NFS daemon.**

```
# /etc/init.d/nfs.server start
```

7. **Verify that the NFS daemon is indeed running.**

```
# ps -ef | grep nfsd
root 14533    1 17 10:46:55 ?     0:00 /usr/lib/nfs/nfsd -a 16
root 14656  289  7 14:06:02 pts/3 0:00 /grep nfsd
```

8. **Eject any CD currently in the drive.**

```
# eject cdrom0
```

9. **Assign root write permissions to** `/etc/rmmount.conf`**.**

```
# chmod 644 /etc/rmmount.conf
```

10. **Add the following lines to** `/etc/rmmount.conf`**.**

```
# File System Sharing
share cdrom*
```

These lines share any CD loaded into your system's CD-ROM drive. You can, however, limit sharing to a particular CD or series of CDs, as described in `share(1M)`.

11. **Remove write permissions from** `/etc/rmmount.conf`**.**

```
# chmod 444 /etc/rmmount.conf
```

This step returns the file to its default permissions.

12. **Load a CD.**
The CD you now load, and all subsequent CDs, will be available to other systems. Remember to wait until the light on the drive stops blinking before you verify this task.

To access the CD, the remote user must mount it by name, according to the instructions in "How to Access CDs on Other Systems" on page 207.

Verification—Making Local CDs Available to Other Systems

To verify that the CD is indeed available to other systems, use the `share` command. If the CD is available, its share configuration will be displayed. (The shared dummy directory will also be displayed.)

```
# share
-    /dummy  ro "dummy dir to wake up NFS daemon"
-    /Solaris_2.5_Install  ro  ""
```

Example—Making Local CDs Available to Other Systems

The following example makes any CD loaded into the local system's CD-ROM drive available to other systems on the network.

```
$ su
Password password
# ps -ef | grep nfsd
root 14522  236  7 14:06:02 pts/3 0:00 /grep nfsd
# mkdir /dummy
# vi /etc/dfs/dfstab

        # /etc/dfs/dfstab
        share -F nfs -o ro "fake dir" /dummy

# /etc/init.d/nfs.server start
# ps -ef | grep nfsd
root 14533    1 17 14:10:55 ?     0:00 /usr/lib/nfs/nfsd -a 16
root 14577  289  7 14:11:55 pts/3 0:00 /grep nfsd
# eject cdrom0
# chmod 644 /etc/rmmount.conf
# vi /etc/rmmount.conf

        # /etc/rmmount.conf
        # File System Sharing
        share cdrom*

# chmod 444 /etc/rmmount.conf

—Load the CD—

# share
-    /dummy  ro "dummy dir to wake up NFS daemon"
-    /Solaris_2.5_Install  ro  ""
```

▼ How to Configure a System to Play Musical CDs

You can play musical CDs from a CD-ROM attached to your Solaris system. You'll need to access Workman, which is public domain software, and you must attach external speakers or headphones independently to the CD-ROM drive; speakers attached to the system hardware will not work.

Once you configure your system, you can play a musical CD simply by inserting it into the CD-ROM drive. The workman control panel is automatically displayed on your desktop.

1. **Become root.**
2. **Edit `/etc/rmmount.conf`.**
   Add the following line under `# Actions`, before the `cdrom` action, as shown in the example below.

```
# Actions
action cdrom action_workman.so path/workman workman-options
```

In this entry,

path	The directory in which you have placed the Workman software
workman-options	The options allowed by the Workman software

Example—Configuring a System to Play Musical CDs

```
❶  # @(#)rmm.conf 1.2     92/09/23 SMI
   #
   # Removable Media Mounter configuration file.
   #

   # File system identification
   ident hsfs ident_hsfs.so cdrom
   ident ufs ident_ufs.so cdrom floppy
   ident pcfs ident_pcfs.so floppy

   # Actions
❷  action cdrom action_workman.so /home/smith/programs/workman
   action cdrom action_filemgr.so
   action floppy action_filemgr.so
```

❶ Default information for `rmmount.conf` file

❷ Workman line

▼ How to Prepare a System for a New CD-ROM Drive

On a system that is properly booted with the `boot -r` command, Volume Management will automatically recognize a new CD-ROM drive. However, to make sure Volume Management always recognizes a new drive, create the `/reconfigure` file.

1. **Become root.**

2. **Create a file called `/reconfigure`.**

```
# touch /reconfigure
```

The `/reconfigure` file makes the Solaris environment check for the presence of any newly installed peripheral devices when you power on or boot your system. After that, Volume Management locates the new device and automatically manages it for you.

Configuring Volume Management

Occasionally, you may want to manage diskettes or CDs without the help of Volume Management. This section describes how to stop and restart it.

▼ How to Stop Volume Management

1. **Make sure no diskettes or CDs are being used.**
   If you are not sure whether you have found all users of the diskette or CD, use the `fuser` command, as described in “How to Find Out If a CD is Still in Use” on page 206.

2. **Become root.**

3. **Enter the `stop` command.**

```
# /etc/init.d/volmgt stop
```

▼ How to Restart Volume Management

1. **Become root.**

2. **Enter the `start` command.**

```
# /etc/init.d/volmgt start
volume management starting.
```

Formatting and Using Diskettes From the Command Line 14

This chapter describes all the tasks required to format and use diskettes from the command line in the Solaris environment. This is a list of the step-by-step instructions in this chapter.

14

Formatting Diskettes

Table 14-1 Task Map: How Diskettes Are Typically Formatted

Activity	Description	For Instructions, Go To	
Load Unformatted Diskette	Insert the diskette into the drive and enter the `volcheck` command.	▼ How to Load a Diskette	page 232
Format the Diskette	Format the diskette for either UFS or DOS.	▼ How to Format a UFS Diskette	page 222
		▼ How to Format a DOS Diskette	page 228
Add a UFS Filesystem	**UFS Only.** Optional. To use the diskette for files, add a UFS filesystem. To use for characters, skip this step.	▼ How Place a UFS File System on a Diskette	page 225
Eject the Diskette	When finished formatting, always eject the diskette, even if you are going to use it again right away.	▼ How to Eject a Diskette	page 238

Using Diskette Names

When working with diskettes, you can identify them by name or with a designator from Table 14-2, below. For brevity, task descriptions use `floppy0`, but you can replace it with either the diskette name or a different designator.

Table 14-2 How to Identify Diskettes

CD	Alternate Name
First CD-ROM drive	`cdrom0`
Second CD-ROM drive	`cdrom1`
Third CD-ROM drive	`cdrom2`

Also note that diskettes that are not named (that is, have no "label") are assigned the default name of `noname`.

Hardware Considerations

A Solaris system can format diskettes for use on both Solaris and DOS systems. However, the hardware platform imposes some limitations. They are summarized in the table below.

Solaris on this Platform ...	Can Format Diskettes For ...
Solaris on SPARC	Solaris on SPARC (UFS)
	MS-DOS or NEC-DOS (PCFS)
Solaris on x86	Solaris on x86 (UFS)
	MS-DOS or NEC-DOS (PCFS)

Diskettes formatted for UFS are restricted to the hardware platform on which they were formatted. In other words, a UFS diskette formatted on a SPARC system cannot be used for UFS on an x86 system, nor can a diskette formatted on an x86 system be used on a SPARC system. This is because the SPARC and x86 UFS formats are different.[1]

1. SPARC uses little-endian bits coding, x86 uses big-endian.

A complete format for SunOS file systems consists of the basic “bit” formatting plus the structure to support a SunOS file system. A complete format for a DOS file system consists of the basic “bit” formatting plus the structure to support either an MS-DOS or an NEC-DOS file system. The procedures required to prepare a diskette for each type of file system are different. Therefore, before you format a diskette, consider which procedure to follow. See Table 14-1 on page 218.

On a Solaris system (either SPARC or x86), you can format diskettes of seven different densities (provided you have the appropriate drive).

Diskette Size	Diskette Density	Capacity
3.5”	Extended Density	2.88 Mbytes
3.5”	High Density (HD)	1.44 Mbytes
3.5”	Medium Density (DD)	1.2 Mbytes
3.5”	Low Density	720 Kbytes
5.25”	High Density (HD)	1.2 Mbytes
5.25	Medium Density (DD)	720 Kbytes
5.25”	Low Density	360 Kbytes

By default, the diskette drive formats a diskette to a like density. In other words, a 1.44 Mbyte drive attempts to format a diskette for 1.44 Mbytes, whether the diskette is in fact a 1.44 Mbyte diskette or not —unless you instruct it otherwise. You can tell a 1.44 Mbyte drive to format a diskette to, for instance, 720 Kbytes. You cannot, however, instruct a 720 Kbyte drive to format a diskette to 1.44 Mbyte. In other words, a diskette can be formatted to its capacity or lower, and a drive can format to its capacity or lower.

To instruct a drive to format a diskette to a non-default density, use the `fdformat` command as instructed in the following tasks, but use the appropriate *density option* from Table 14-3, below.

Table 14-3 Density Options

To Format a Diskette with this Density ...	In a Drive with this Default Density ...	Use this *density option* to the `fdformat` Command ...
2.88 Mbytes	2.88 Mbytes	`-E`
1.44 Mbytes	2.88 Mbytes	`-H`
1.44 Mbytes	1.44 Mbytes	`none`
1.2 Mbytes	1.44 Mbytes	`-t nec -M`
720 Kbytes	1.44 Mbytes	`-D` or `-t dos -D`
1.2 Mbytes	1.2 Mbytes	`none`
720 Kbytes	1.2 Mbytes	`-D`
720 Kbytes	720 Kbytes	`none`
360 Kbytes	720 Kbytes	`-D`

To view all the options to the `fdformat` command, either see `fdformat(1)` or enter `fdformat -z`. The `-z` option displays all the options to the command.

If you don't know the default density of your drive, begin the formatting process with the default setting (that is, no *density options*) and observe the configuration message. It will look something like this:

```
Formatting 1.44 M in /vol/dev/rdiskette0/unformatted
Press return to start formatting floppy.
```

The confirmation message indicates the drive's default density. For instance, in the example above, the default density of the drive is 1.44 Mbytes. If the density is not what you expected, use Control-c to escape the formatting process and start over.

▼ How to Format a UFS Diskette

As mentioned in "Hardware Considerations" on page 219, a UFS diskette formatted on a SPARC system can only be used on another SPARC system, and a UFS diskette formatted on an x86 system can only be used on an x86 system running Solaris.

Caution – Formatting a diskette erases any pre-existing content.

---

1. **Quit File Manager.**
   File Manager automatically displays a formatting window when you insert an unformatted diskette. Unfortunately, File Manager formatting is unreliable. To avoid the window, quit from File Manager. If you prefer to keep File Manager open, quit the formatting window when it appears.

2. **Make sure the diskette is write-enabled.**
   On both 3.5-inch and 5.25 inch diskettes, write-protection is controlled by a small tab in either the lower left or lower right corner. If you can see through the square hole behind the tab, the diskette is write-protected. If the hole is covered by the tab, the diskette is write-enabled. (If you need to eject the diskette to examine it, simply type `eject floppy` in a shell.)

3. **Insert the diskette.**
   Make sure the diskette is completely inserted.

4. Invoke formatting.

```
$ fdformat -v -U [density-options  convenience-options]
```

In this command,

`-v`	Verifies whether the diskette was formatted correctly.
`-U`	Unmounts the diskette if it is mounted.
density-options	If the drive density is 1.44 Mbytes, *density-options* are:
—*none*—	Formats a 1.44 Mbyte diskette.
`-D`	Formats a 720 Kbyte diskette.
	A complete list of *density-options* appears in Table 14-3 on page 221.
convenience-options	
`-e`	Ejects the diskette when done formatting.
`-f`	Forces formatting without asking for confirmation.
`-b` *label*	Names the diskette. *label* must be eight characters or less. Case is ignored.
`-z`	Lists all the options to the `fdformat` command, but does not format the diskette.

Note – If you try to format a 720Kbyte (DD) diskette for 1.44 Mbytes, `fdformat` will not stop you unless you include the `-v` option. With the `-v` option, `fdformat` will format the diskette, but the verification will catch the error and notify you with the following message:
`fdformat: check diskette density, I/O error`

The `fdformat` command displays a confirmation message (unless you used the `-f` option), indicating the type of formatting to be performed:

```
Formatting 1.44 M in /vol/dev/rdiskette0/unformatted
Press return to start formatting floppy.
```

5. Select one of the options in the table below.

To ...	Press ...
Confirm the type of formatting	Return (unless you used the `-f` option in the previous step, in which case no confirmation is necessary).
Cancel formatting	Control-c.

As the formatting progresses, a series of dots is displayed. As the verification progresses, a series of V's appears beneath the dots. When the series stops, the formatting is complete.

6. Eject the diskette.

```
$ eject floppy0
```

The diskette is now ready for raw character operations such as `tar` and `cpio`. To prepare the diskette for Solaris file system operation, add a UFS file system to the diskette, as described in "How to Format a UFS Diskette" on page 222.

Examples—Formatting a UFS Diskette

Following are several examples of UFS formatting. The first example formats a 1.44 Mbyte diskette on a 1.44 Mbyte drive:

```
$ fdformat -v -U
Formatting 1.44 M in /vol/dev/rdiskette0/unformatted
Press return to start formatting floppy. [ Return ]
.............................................................
vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
```

The following example performs the same job, but assigns the diskette the name `myfiles`:

```
$ fdformat -v -U -b myfiles
Formatting 1.44 M in /vol/dev/rdiskette0/unformatted
Press return to start formatting floppy. [ Return ]
.............................................................
vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
```

The following example formats a 720Kbyte diskette on a 1.44 Mbyte drive, and names it `myfiles`:

```
$ fdformat -v -U -D -b myfiles
Formatting 720 KB in /vol/dev/rdiskette0/unformatted
Press return to start formatting floppy. [ Return ]
.............................................................
vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
```

▼ How Place a UFS File System on a Diskette

Even though the procedure for adding a UFS file system is the same for UFS diskettes formatted on x86 systems and SPARC systems, a UFS diskette formatted on a SPARC system can only be used on another SPARC system, and a UFS diskette formatted on an x86 system can only be used on an x86 system running Solaris.

1. **Format the diskette for a UFS file system.**
   Use "How to Format a UFS Diskette" on page 222.

2. **Re-insert the diskette.**

3. **Notify Volume Management.**

```
$ volcheck -v
media was found
```

The status message "media was found" is displayed in the shell, but the following error messages appear in the Console:

```
fd0: unformatted diskette or no diskette in the drive
fd0: read failed (40 1 0)
fd0: bad format
```

Ignore this messages. It appears because the diskette has no file system yet.

4. Create a SunOS file system on the diskette.

```
$ newfs -v /vol/dev/aliases/floppy0
```

In this command,

`-v`	Prints status messages.
`/vol/dev/aliases/floppy0`	Indicates the location of the floppy.

The `newfs` command displays message asking you to confirm the creation of the file system.

5. Confirm the creation of the file system.

```
newfs: construct a new file system  \
      /vol/dev/aliases/floppy0:(y/n)? y
```

A status message is displayed, indicating the particulars of the file system and the diskette's formatting:

```
mkfs -F ufs /vol/dev/aliases/floppy0 2880 18 2 8192 1024 16 \
    10 5 2048 t 0 -1 8 -1
/vol/dev/aliases/floppy0:  2880 sectors in 80 cylinders of \
    2 tracks, 18 sectors
1.4 MB in 5 cyl groups (16 c/g, 0.28MB/g, 128 i/g)
super-block backups (for fsck -F ufs -o b=#) at:
    32, 640, 1184, 1792, 2336
```

The diskette is now ready to be used on a SPARC system. However, before Volume Management recognizes it, you must eject and reinsert it, as described in the following steps.

6. **Eject the diskette.**

```
$ eject floppy0
```

7. **Reinsert the diskette and notify Volume Management.**

```
$ volcheck -v
media was found
```

The diskette should now be mounted under `/floppy/floppy0`.

Verification—Placing a UFS File System on a Diskette

Use the `ls` command on the `/floppy` directory.

```
$ ls /floppy
floppy0
```

If the `floppy0` subdirectory appears, the diskette has a UFS file system and has been mounted properly.

Example—Placing a UFS File System on a Diskette

```
$ volcheck -v
media was found
$ newfs -v /vol/dev/aliases/floppy0
newfs: construct a new file system  \
       /vol/dev/aliases/floppy0:(y/n)? y
mkfs -F ufs /vol/dev/aliases/floppy0 ...

$ eject floppy0

—re-insert the diskette–

$ volcheck -v
media was found
```

▼ How to Format a DOS Diskette

You can format a DOS diskette on a SPARC or x86 Solaris system. The steps are similar, except that instead of a SunOS file system being placed on the diskette, a DOS file system, either MS-DOS or NEC-DOS, is put on the file system.

Caution – Formatting a diskette erases any pre-existing content.

---

1. **Quit File Manager.**
   File Manager automatically displays a formatting window when you insert an unformatted diskette. Unfortunately, File Manager formatting is unreliable. To avoid the window, quit from File Manager. If you prefer to keep File Manager open, quit the formatting window when it appears.

2. **Make sure the diskette is not write-protected.**
   On both 3.5-inch and 5.25 inch diskettes, write-protection is controlled by a small tab in either the lower left or lower right corner. If you can see through the square hole behind the tab, the diskette is write-protected. If the hole is covered by the tab, the diskette is write-enabled. (If you need to eject the diskette to examine it, simply type `eject floppy` in a shell.)

3. **Insert the diskette.**
   Make sure the diskette is completely inserted. It must drop down into the drive.

4. Invoke formatting.

```
$ fdformat -v -U  [density-options  convenience-options]
```

In this command,

`-v`	Verifies whether the diskette was formatted correctly.
`-U`	Unmounts the diskette if it is mounted.
density-options	If the drive density is 1.44 Mbytes, *density-options* are:
`-d`	Formats at 1.44 Mbytes for MS-DOS.
`-d -D`	Formats at 720 Kbytes for MS-DOS.
`-t nec -M`	Formats at 1.2 Mbytes for NEC-DOS.
	A complete list of *density-options* appears in Table 14-3 on page 221.
convenience-options	
`-e`	Ejects the diskette when done formatting.
`-f`	Does not ask for confirmation before formatting.
`-b` *label*	Name the diskette. *label* must be eight characters or less. Case is ignored.
`-z`	Lists all the options to the `fdformat` command, but does not format the diskette.

Note – If you try to format a 720Kbyte (DD) diskette for 1.44 Mbytes, `fdformat` will not stop you unless you include the `-v` option. With the `-v` option, `fdformat` will format the diskette, but the verification will catch the error and notify you with the following message:
`fdformat: check diskette density, I/O error`

The `fdformat` command displays a confirmation message, indicating the type of formatting to be performed:

```
Formatting 1.44 M in /vol/dev/rdiskette0/unformatted
Press return to start formatting floppy.
```

5. **Select one of the options in the table below.**

To ...	Press ...
Confirm the type of formatting	Return (unless you used the `-f` option in the previous step, in which case no confirmation is necessary).
Cancel formatting	Control-c.

As the formatting progresses, a series of dots is displayed. As the verification progresses, a series of V's appears beneath the dots. When the series stops, the formatting is complete and the diskette is ready for use on a DOS system.

6. **Eject the diskette.**

```
$ eject floppy0
```

7. **Reinsert the diskette and notify Volume Management.**

```
$ volcheck -v
media was found
```

Volume Management mounts the diskette under `/floppy/floppy0`.

Using Diskettes

Table 14-4 Task Map: How Diskettes Are Typically Used

Activity	Description	For Instructions, Go To	
Load the Diskette	Insert the diskette into its drive and enter the `volcheck` command.	▼ How to Load a Diskette	page 232
Examine its Contents	Optional. To examine the contents of the diskette, look in the appropriate directory under `/diskette`.	▼ How to Examine the Contents of a Diskette	page 233
Exchange Files	Optional. Copy files or directories between the diskette and your filesystem.	▼ How to Copy or Move Information From a Diskette	page 234
		▼ How to Copy or Move Information to a Diskette	page 235
Is Diskette Still in Use?	Optional. Before ejecting the diskette, use the `fusers` command to find out if the diskette is still in use.	▼ How to Find Out If a Diskette Is Still in Use	page 237
Eject the Diskette	When you finish, eject the diskette.	▼ How to Eject a Diskette	page 238

How to Load a Diskette

1. **Make sure the diskette is formatted.**
   If you aren't sure, insert it and check the status messages in the console, as described in Step 3, below. If you need to format the diskette, go to "How to Format a UFS Diskette" on page 222 or "How to Format a DOS Diskette" on page 228.

2. **Insert the diskette.**
   Make sure the diskette is completely inserted. It must drop down into the drive. If the drive has a door, close it.

3. **Notify Volume Management.**

```
$ volcheck -v
media was found
```

Two status messages are possible:

`media was found`	Volume Management detected the diskette and will attempt to mount it in the `/floppy` directory.
	
	If the diskette is formatted properly, no error messages appear in the console.
	
	If the diskette is not formatted, the "`media was found`" message is still displayed, but the following error messages appear in the Console:
	
	`fd0: unformatted diskette or no diskette in the drive`
	`fd0: read failed (40 1 0)`
	`fd0: bad format`
	
	You must format the diskette before Volume Management can mount it. Instructions are provided on page 222 (for UFS) and page 228 (for DOS).

`no media was found`	Volume Management did not detect a diskette. Make sure the diskette is inserted properly and run `volcheck` again. If unsuccessful, check the diskette; it could be damaged. You can also try to mount the diskette manually.

Verification—Loading a Diskette

Verify that the diskette was mounted by listing its contents.

```
$ ls /floppy/floppy0
floppy0 myfiles
```

As described earlier, `floppy0` is a symbolic link to the actual name of the diskette; in this case, `myfiles`. If the diskette has no name but is formatted correctly, the system will refer to it as `unnamed_floppy`.

If nothing appears under the `/floppy` directory, the diskette was either not mounted or is not formatted properly. To find out, run the `mount` command and look for the line that begins with `/floppy` (usually at the end of the listing):

`/floppy/`*name* `on /vol/dev/diskette0/`*name* `...`

If the line does not appear, the diskette was not mounted. Check the Console for error messages.

▼ How to Examine the Contents of a Diskette

Use the `-L` option to the `ls` command, because some directories under `/floppy` are symbolic links.

```
$ ls -L [-l] floppy0
```

In this command,

`-L`	Includes symbolic links in the output
`-l`	Long format. Includes permissions and owners in the output.

Example—Examining the Contents of a Diskette

The following example lists the contents of the diskette in the first floppy drive, identified by `floppy0`.

```
$ ls -L -l /floppy/floppy0
-rwxrwxrwx  1 smith  staff 362284  Nov 16 20:54  text.doc
-rwxrwxrwx  1 smith  staff  24562  Nov 16 12:20  art.gif
```

Once you have inserted a diskette, you can access its files and directories just as you would those of any other file system. The only restrictions to be aware of are ownership and permissions. For instance, if you are not the owner of a file on a diskette, you won't be able to overwrite it on the diskette. Or, if you copy a file into your file system, you'll be the owner, but it won't have write permissions (because it never had them on the diskette); you'll have to change the permissions yourself.

▼ How to Copy or Move Information From a Diskette

1. **Make sure the diskette is formatted and mounted.**

```
$ ls /floppy
floppy0   diskette-name
```

If the diskette is properly formatted and mounted, its name and the symbolic link will appear under `/floppy`.

If nothing appears under the `/floppy` directory, the diskette is not mounted. See the task titled, "How to Load a Diskette" on page 232. The diskette might also need to be formatted. See the task titled, "How to Format a UFS Diskette" on page 222 or "How to Format a DOS Diskette" on page 228.

2. **Copy the files or directories.**

To Copy ...	Use ...
A file	`cp`
A directory	`cp -r`

Verification—Copying or Moving Information from a Diskette

To verify a copy or move operation, use the `ls` command.

Examples—Copying or Moving Information from a Diskette

The first example, below, moves a file (`readme.doc`) from the diskette to the current directory (indicated by the "`.`" symbol). The second example copies a file (`readme2.doc`) from the diskette to the current directory. The third example copies a directory (`morefiles`) and everything below it from the diskette to the current directory.

```
$ mv /floppy/floppy0/readme.doc .

$ cp /floppy/floppy0/readme2.doc .

$ cp -r /floppy/floppy0/morefiles .
```

▼ How to Copy or Move Information to a Diskette

1. **Make sure the diskette is not write-protected.**
   On both 3.5-inch and 5.25 inch diskettes, write-protection is controlled by a small tab in either the lower left or lower right corner. If you can see through the square hole behind the tab, the diskette is write-protected. If the hole is covered by the tab, the diskette is write-enabled.

2. **Make sure the diskette is formatted and mounted.**

   ```
   $ ls /floppy
   floppy0   diskette-name
   ```

   If the diskette is properly formatted and mounted, its name and the symbolic link, `floppy0`, will appear under `/floppy`.

   If nothing appears under the `/floppy` directory, the diskette is not mounted. See the task titled, "How to Load a Diskette" on page 232. The diskette might also need to be formatted. See the task titled, "How to Format a UFS Diskette" on page 222 or "How to Format a DOS Diskette" on page 228.

3. Move or copy the files or directories.

To ...	Use ...
Copy a file	`cp`
Copy a directory	`cp -r`
Move a file or directory	`mv`

Verification—Copying or Moving Information to a Diskette

To verify a move or copy operation, use the `ls` command.

Examples—Copying or Moving Information to a Diskette

The first example, below, moves a file (`readme.doc`) from the current directory to the diskette loaded into the first floppy drive (indicated by `/floppy/floppy0`). The second example copies a file (`readme2.doc`) from the current directory to the diskette loaded into the second floppy drive (indicated by `/floppy/floppy1`). The third example copies a directory (`morefiles`) and its contents from the `/home/smith/`directory to the diskette loaded into the first floppy drive.

```
$ mv readme.doc /floppy/floppy0

$ cp readme.doc /floppy/floppy1

$ cp -r /home/smith/morefiles /floppy/floppy0
```

How to Find Out If a Diskette Is Still in Use

1. **Become root.**

2. **Invoke the `fuser` command.**
   The `fuser` command lists the processes that are currently accessing the CD that you specify.

```
# fuser -u [ -k ] floppy0
```

In this command,

`-u`	Displays the user of the diskette.
`-k`	Kills the process accessing the diskette.

Example—Finding Out If a Diskette Is Still In Use

In the following example, the processes `6400c` and `6399c` are accessing the `/floppy/floppy0` directory, and the process owners are `root` and `smith`, respectively.

```
# fuser -u /floppy/floppy0
/floppy/floppy0: 6400c(root)  6399c(smith)
```

You can kill the processes individually (as root), or you can use the `fuser` command with the `-k` option, which kills all the processes accessing that file system:

```
# fuser -u -k /floppy/floppy0
/floppy/floppy0: 6400c(root)Killed  6399c(smith)Killed
```

The `fuser` command may not always identify all the killed processes. To be sure, run it again with the `-u` option.

How to Eject a Diskette

1. **Make sure the diskette is not being used.**
   Remember, a diskette is "being used" if a shell or an application is accessing any of its files or directories.

   If you are not sure whether you have found all users of a diskette (a renegade shell hidden behind a desktop tool may be accessing it), use the `fuser` command, as described in "How to Find Out If a Diskette Is Still in Use" on page 237.

2. **Eject the diskette.**

```
$ eject floppy0
```

   On a SPARC system the floppy is physically ejected from its drive, but on an x86 system you'll have to eject the diskette by hand. If you are running Windows, look for an onscreen message that says you can now eject the diskette.

   If the diskette is still in use, the following message appears:

```
/vol/dev/rdiskette0/noname: Device busy
```

   In this case, return to Step 1 and make sure no one is using the diskette, then eject it again.

   If the diskette jams, eject it manually by inserting an unfolded paper clip about an inch into the small hole in the front of the drive.

How to Access Diskettes on Other Systems

You can access a diskette on another system by mounting it manually into your file system—provided the other system has exported its diskette drive according to the instructions in “How to Make Local Diskettes Available to Other Systems” on page 241.

1. **Select an existing directory to serve as the mount point or create one.**

```
$ mkdir directory
```

In this command,

directory	Is the name of the directory that you create to serve as a mount point for the other system’s diskette.

2. **Find the name of the diskette you want to mount.**
When you manually mount a remote diskette, you cannot use the `floppy0` or `floppy1` variables available with your local diskettes. You must use the exact diskette name. To find it, use the `ls` command on the remote system’s `/floppy` directory. If the automounter is running, you can simply `cd` to the system whose diskette you want to mount and then use the `ls` command. If the automounter is not running, you’ll have to use another method, such as logging in remotely.

3. **As root, mount the diskette.**

```
# mount -F nfs -o rw system-name:/floppy/diskette-name local-mount-point
```

In this command,

system-name	Is the name of the system whose diskette you will mount.
diskette-name	Is the name of the diskette you want to mount
local-mount-point	Is the local directory onto which you will mount the remote diskette.

Verification—Accessing Diskettes on Other Systems

To verify that the diskette is indeed mounted, use the `ls` command to list the contents of the mount point.

```
$ ls /floppy
```

Example—Accessing Diskettes on Other Systems

This example mounts the diskette named `MyFiles` from the remote system `mars` onto the `/floppy` directory of the local system.

```
$ cd /net/mars
$ ls /floppy
floppy0     MyFiles
$ su
Password: password
# mount -F nfs rw mars:/floppy/MyFiles /floppy
# exit
$ ls /floppy
MyFiles
```

How to Make Local Diskettes Available to Other Systems

You can configure your system to share its diskettes; in other words, make any diskettes in those drives available to other systems. Once your diskette drives are shared, other systems can access the diskettes they contain simply by mounting them, as described in "How to Access Diskettes on Other Systems" on page 239.

1. **Become root.**

2. **Find out whether the NFS daemon (`nfsd`) is running.**

```
# ps -ef | grep nfsd
root 14533    1 17 10:46:55 ?     0:00 /usr/lib/nfs/nfsd -a 16
root 14656  289  7 14:06:02 pts/3 0:00 /grep nfsd
```

If the daemon is running, a line for `/usr/lib/nfs/nfsd` will appear, as shown above. If the daemon is not running, only the `/grep nfsd` line will appear.

3. **Select an option from the following table.**

If ...	Then ...
`nfsd` is running	Go to Step 8
`nfsd` is *not* running	Continue with Step 4

4. **Create a dummy directory for `nfsd` to share.**

```
# mkdir /dummy-dir
```

In this command,

dummy-dir — Can be any directory name; for example, `dummy`. This directory will not contain any files. Its only purpose is to "wake up" the NFS daemon so that it notices your shared diskettes.

5. **Add the following entry into** `/etc/dfs/dfstab`**.**

```
share -F nfs -o ro [-d comment] /dummy-dir
```

When you start the NFS daemon, it will see this entry, "wake up," and notice the shared diskette drive. Note that the comment (preceded by `-d`) is optional.

6. **Start the NFS daemon.**

```
# /etc/init.d/nfs.server start
```

7. **Verify that the NFS daemon is indeed running.**

```
# ps -ef | grep nfsd
root 14533    1 17 10:46:55 ?     0:00 /usr/lib/nfs/nfsd -a 16
root 14656  289  7 14:06:02 pts/3 0:00 /grep nfsd
```

8. **Eject any diskette currently in the drive.**

```
# eject floppy0
```

9. **Assign root write permissions to** `/etc/rmmount.conf`**.**

```
# chmod 644 /etc/rmmount.conf
```

10. **Add the following lines to** `/etc/rmmount.conf`**.**

```
# File System Sharing
share floppy*
```

These lines share any diskette loaded into your system's diskette drives.

11. **Remove write permissions from** `/etc/rmmount.conf`**.**

```
# chmod 444 /etc/rmmount.conf
```

This step returns the file to its default permissions.

12. **Load a diskette.**

```
—Insert the diskette—
# volcheck -v
media was found
```

The diskette you now load, and all subsequent diskettes, will be available to other systems. To access the diskette, the remote user must mount it by name, according to the instructions in "How to Access Diskettes on Other Systems" on page 239.

Verification—Making Local Diskettes Available to Other Systems

To verify that the diskette is indeed available to other systems, use the `share` command. If the diskette is available, its share configuration will be displayed. (The shared dummy directory will also be displayed.)

```
# share
-    /dummy  ro "dummy dir to wake up NFS daemon"
-    /MyFiles rw  ""
```

Example—Making Local Diskettes Available to Other Systems

The following example makes any diskette loaded into the local system's diskette drive available to other systems on the network.

```
$ su
Password password
# ps -ef | grep nfsd
root 14522  236  7 14:06:02 pts/3 0:00 /grep nfsd
# mkdir /dummy
# vi /etc/dfs/dfstab
```

```
# /etc/dfs/dfstab
share -F nfs -o ro “fake dir” /dummy
```

```
# /etc/init.d/nfs.server start
# ps -ef | grep nfsd
root 14533    1 17 14:10:55 ?     0:00 /usr/lib/nfs/nfsd -a 16
root 14577  289  7 14:11:55 pts/3 0:00 /grep nfsd
# eject floppy0
# chmod 644 /etc/rmmount.conf
# vi /etc/rmmount.conf
```

```
# /etc/rmmount.conf
# File System Sharing
share floppy*
```

```
# chmod 444 /etc/rmmount.conf

—Load the diskette—

# volcheck -v
media was found
# share
-    /dummy  ro "dummy dir to wake up NFS daemon"
-    /MyFiles rw  ""
```

Using PCMCIA Memory Cards From the Command Line 15

This chapter describes all the tasks required to format and use PCMCIA memory cards from the command line in the Solaris environment.

This is a list of the step-by-step instructions in this chapter.

15

Formatting PCMCIA Memory Cards

Table 15-1 Task Map: How PCMCIA Memory Cards Are Typically Formatted

Activity	Description	For Instructions, Go To	
Load Unformatted PCMCIA Memory Card	Insert the PCMCIA memory card into the drive and enter the `volcheck` command.	▼ How to Load a PCMCIA Memory Card	page 258
Format the PCMCIA Memory Card	Format the PCMCIA memory card for either UFS or DOS.	▼ How to Format a UFS PCMCIA Memory Card	page 248
		▼ How to Format a DOS PCMCIA Memory Card	page 254
Add a UFS File System	**UFS Only.** Optional. To use the PCMCIA memory card for files, add a UFS file system. To use for characters, skip this step.	▼ How to Place a UFS File System on a PCMCIA Memory Card	page 251
Eject the PCMCIA MemoryCcard	When finished formatting, always eject the PCMCIA memory card, even if you are going to use it again right away.	▼ How to Eject a PCMCIA Memory Card	page 265

Using PCMCIA Memory Cards Names

When working with PCMCIA memory cards, you can identify them by name or with a designator from Table 15-2, below. For brevity, task descriptions use `pcmem0`, but you can replace it with either the PCMCIA memory card's name or a different designator.

Table 15-2 How to Identify PCMCIA Memory Cards

PCMCIA Card	Alternate Name
First PCMCIA drive	`pcmem0`
Second PCMCIA drive	`pcmem1`
Third PCMCIA drive	`pcmem2`

Also note that PCMCIA memory cards that are not named (that is, have no "label") are assigned the default name of `noname`.

Hardware Considerations

A Solaris system can format PCMCIA memory cards for use on both Solaris and DOS systems. However, the hardware platform imposes some limitations. They are summarized in the table below.

Solaris on This Platform ...	Can Format PCMCIA Memory Cards For ...
Solaris on SPARC	Solaris on SPARC (UFS)
	MS-DOS or NEC-DOS (PCFS)
Solaris on x86	Solaris on x86 (UFS)
	MS-DOS or NEC-DOS (PCFS)

PCMCIA memory cards formatted for UFS are restricted to the hardware platform on which they were formatted. In other words, a UFS PCMCIA memory card formatted on a SPARC system cannot be used for UFS on an x86 system. Likewise, PCMCIA memory cards formatted on an x86 system cannot be used on a SPARC system. This is because the SPARC and x86 UFS formats are different.

A complete format for UFS file systems consists of the basic “bit” formatting plus the structure to support a UFS file system. A complete format for a DOS file system consists of the basic “bit” formatting plus the structure to support either an MS-DOS or an NEC-DOS file system. The procedures required to prepare a PCMCIA memory card for each type of file system are different. Therefore, before you format a PCMCIA memory card, consider which route to take. See Table 15-1 on page 246.

To view all the options to the `fdformat` command, either see the `fdformat(1)` or enter `fdformat -z`. The `-z` option displays all the options to the command.

▼ How to Format a UFS PCMCIA Memory Card

As mentioned in the introduction, a UFS PCMCIA memory card formatted on a SPARC system can be used only on another SPARC system, and a UFS PCMCIA memory card formatted on an x86 system can be used only on an x86 system running the Solaris release.

Caution – Formatting a PCMCIA memory card erases any pre-existing content.

1. **Quit File Manager.**
   File Manager automatically displays a formatting window when you insert an unformatted PCMCIA memory card. Unfortunately, File Manager formatting is unreliable. To avoid the window, quit File Manager. If you prefer to keep File Manager open, quit the formatting window when it appears.

2. **Make sure the PCMCIA memory card is write-enabled.**
   Write-protection is controlled by a small slide switch in the end of the PCMCIA memory card.

3. **Insert the PCMCIA memory card.**
   Make sure the PCMCIA memory card is completely inserted.

4. Invoke formatting.

```
$ fdformat -v -U [convenience-options]
```

In this command,

`-v`	Verifies whether the PCMCIA memory card was formatted correctly.
`-U`	Unmounts the PCMCIA memory card if it is mounted.

convenience-options

`-e`	Ejects the PCMCIA memory card when done formatting.
`-f`	Forces formatting without asking for confirmation.
`-b` *label*	Names the PCMCIA memory card. *label* must be eight characters or less. Case is ignored.
`-z`	Lists all the options to the `fdformat` command, but does not format the PCMCIA memory card.

The `fdformat` command displays a confirmation message (unless you used the `-f` option), indicating the type of formatting to be performed:

```
Formatting in /vol/dev/aliases/pcmem0
Press return to start formatting pcmem0.
```

5. **Select one of the options in the table below.**

To ...	Press ...
Confirm the type of formatting	Return (unless you used the `-f` option in the previous step, in which case no confirmation is necessary).
Cancel formatting	Control-c.

As the formatting progresses, a series of dots is displayed. As the verification progresses, a series of V's appears beneath the dots. When the series stops, the formatting is complete.

6. **Eject the PCMCIA memory card.**

```
$ eject pcmem0
```

The PCMCIA memory card is now ready for raw character operations such as `tar` and `cpio`. To prepare the PCMCIA memory card for Solaris file system operation, add a UFS file system to the PCMCIA memory card, as described in "How to Format a UFS PCMCIA Memory Card" on page 248.

Examples—Formatting a UFS PCMCIA Memory Card

Following are several examples of UFS formatting.

```
$ fdformat -v -U
Formatting in /vol/dev/aliases/unformatted
Press return to start formatting pcmem0. [ Return ]
.............................................................
vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
```

The following example performs the same job, but assigns the PCMCIA memory card the name `myfiles`:

```
$ fdformat -v -U -b myfiles
Formatting in /vol/dev/aliases/unformatted
Press return to start formatting pcmem0. [ Return ]
..............................................................
vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
```

▼ How to Place a UFS File System on a PCMCIA Memory Card

Even though the procedure for adding a UFS file system is the same for UFS PCMCIA memory cards formatted on x86 systems and SPARC systems, a UFS PCMCIA memory card formatted on a SPARC system can only be used on another SPARC system, and a UFS PCMCIA memory card formatted on an x86 system can only be used on an x86 system running Solaris.

1. **Format the PCMCIA memory card for a UFS file system.**
   Use the procedure titled, "How to Format a UFS PCMCIA Memory Card" on page 248.

2. **Re-insert the PCMCIA memory card.**

3. **Notify Volume Management.**

```
$ volcheck -v
media was found
```

The status message "`media was found`" is displayed in the shell, but the following error messages appear in the Console:

```
fd0: unformatted diskette or no diskette in the drive
fd0: read failed (40 1 0)
fd0: bad format
```

Ignore these messages. It appears because the PCMCIA memory card has no file system yet.

4. **Create a UFS file system on the PCMCIA memory card.**

```
$ newfs -v /vol/dev/aliases/pcmem0
```

In this command,

`-v`	Prints status messages.
`/vol/dev/aliases/pcmem0`	Indicates the location of the floppy.

The `newfs` command displays a message asking you to confirm the creation of the file system.

5. **Confirm the creation of the file system.**

```
newfs: construct a new file system  \
       /vol/dev/aliases/pcmem0:(y/n)? y
```

A status message is displayed, indicating the particulars of the file system and the PCMCIA memory card's formatting:

```
mkfs -F ufs /vol/dev/aliases/pcmem0 2880 18 2 8192 1024 16 \
    10 5 2048 t 0 -1 8 -1
/vol/dev/aliases/pcmem0:  2880 sectors in 80 cylinders of \
    2 tracks, 18 sectors
1.4 MB in 5 cyl groups (16 c/g, 0.28MB/g, 128 i/g)
super-block backups (for fsck -F ufs -o b=#) at:
    32, 640, 1184, 1792, 2336
```

The PCMCIA memory card is now ready to be used on a SPARC system. However, before Volume Management recognizes it, you must eject and reinsert it, as described in the following steps.

6. **Eject the PCMCIA memory card.**

```
$ eject pcmem0
```

7. Reinsert the PCMCIA memory card and notify Volume Management.

```
$ volcheck -v
media was found
```

The PCMCIA memory card should now be mounted under `/pcmem/pcmem0`.

Verification—Placing a UFS File System on a PCMCIA Memory Card

Use the `ls` command on the `/pcmem` directory.

```
$ ls /pcmem
pcmem0
```

If the `pcmem0` subdirectory appears, the PCMCIA memory card has a UFS file system and has been mounted properly.

Example—Placing a UFS File System on a PCMCIA Memory Card

```
$ volcheck -v
media was found
$ newfs -v /vol/dev/aliases/pcmem0
newfs: construct a new file system  \
       /vol/dev/aliases/pcmem0:(y/n)? y
mkfs -F ufs /vol/dev/aliases/pcmem0 ...

$ eject pcmem0

—re-insert the PCMCIA memory card–

$ volcheck -v
media was found
```

▼ How to Format a DOS PCMCIA Memory Card

You can format a DOS PCMCIA memory card on a SPARC or x86 Solaris system. The steps are similar, except that instead of a SunOS file system being placed on the PCMCIA memory card, a DOS file system, either MS-DOS or NEC-DOS, is put on the file system.

Caution – Formatting a PCMCIA memory card erases any pre-existing content.

1. **Quit File Manager.**
   File Manager automatically displays a formatting window when you insert an unformatted PCMCIA memory card. Unfortunately, File Manager formatting is unreliable. To avoid the window, quit File Manager. If you prefer to keep File Manager open, quit the formatting window when it appears.

2. **Make sure the PCMCIA memory card is not write-protected.**
   Write-protection is controlled by a small slide switch in the end of the PCMCIA memory card.

3. **Insert the PCMCIA memory card.**
   Make sure the PCMCIA memory card is completely inserted. It must drop down into the drive.

4. **Invoke formatting.**

```
$ fdformat -v -U  [density-options  convenience-options]
```

In this command,

`-v`	Verifies whether the PCMCIA memory card was formatted correctly.
`-U`	Unmounts the PCMCIA memory card if it is mounted.
density-options	If the drive density is 1.44 Mbytes, *density-options* are:
`-d`	Formats at 1.44 Mbytes for MS-DOS.

`-d -D`	Formats at 720 Kbytes for MS-DOS.
`-t nec -M`	Formats at 1.2 Mbytes for NEC-DOS
	A complete list of *density-options* appears in `fdformat(1)`.

convenience-options

`-e`	Ejects the PCMCIA memory card when done formatting.
`-f`	Does not ask for confirmation before formatting.
`-b` *label*	Name the PCMCIA memory card. *Label* must be eight characters or less. Case is ignored.
`-z`	Lists all the options to the `fdformat` command, but does not format the PCMCIA memory card.

Note – If you try to format a 720Kbyte (DD) diskette for 1.44 Mbytes, `fdformat` will not stop you unless you include the `-v` option. With the `-v` option, `fdformat` will format the diskette, but the verification will catch the error and notify you with the following message:
`fdformat: check diskette density, I/O error`

The `fdformat` command displays a confirmation message, indicating the type of formatting to be performed:

```
Formatting 1.44 M in /vol/dev/rdiskette0/unformatted
Press return to start formatting floppy.
```

5. **Select one of the options in the table below.**

To ...	Press ...
Confirm the type of formatting	Return (unless you used the `-f` option in the previous step, in which case no confirmation is necessary).
Cancel formatting	Control-c.

As the formatting progresses, a series of dots is displayed. As the verification progresses, a series of V's appears beneath the dots. When the series stops, the formatting is complete and the PCMCIA memory card is ready for use on a DOS system.

6. **Eject the PCMCIA memory card.**

```
$ eject pcmem0
```

7. **Reinsert the PCMCIA memory card and notify Volume Management.**

```
$ volcheck -v
media was found
```

Volume Management mounts the PCMCIA memory card under `/pcmem/pcmem0`.

Using PCMCIA Memory Cards

Table 15-3 Task Map: How PCMCIA Memory Cards Are Typically Used

Activity	Description	For Instructions, Go To	
Load the PCMCIA Memory Card	Insert the PCMCIA memory card into its drive and enter the `volcheck` command.	▼ How to Load a PCMCIA Memory Card	page 258
Display the Contents of a PCMCIA Memory Card	Optional. To examine the contents of the PCMCIA memory card, look in the appropriate directory under */PCMCIAmemorycard*	▼ How to Display the Contents of a PCMCIA Memory Card	page 260
Exchange Files	Optional. Copy files or directories between the PCMCIA memory card and your filesystem.	▼ How to Copy or Move Information From a PCMCIA Memory Card	page 261
		▼ How to Copy or Move Information to a PCMCIA Memory Card	page 262
Is PCMCIA memory card Still in Use?	Optional. Before ejecting the PCMCIA memory card, use the `fusers` command to find out if the PCMCIA memory card is still in use.	▼ How to Find Out If a PCMCIA Memory Card Is Still In Use	page 264
Eject the PCMCIA memory card	When you finish, eject the PCMCIA memory card.	▼ How to Eject a PCMCIA Memory Card	page 265

▼ How to Load a PCMCIA Memory Card

1. **Make sure the PCMCIA memory card is formatted.**
   If you aren't sure, insert it and check the status messages in the Console, as described in Step 3, below. If you need to format the PCMCIA memory card, go to "How to Format a UFS PCMCIA Memory Card" on page 248 or "How to Format a DOS PCMCIA Memory Card" on page 254.

2. **Insert the PCMCIA memory card.**
   Make sure the PCMCIA memory card is completely inserted. It must drop down into the drive. If the drive has a door, close it.

3. **Notify Volume Management.**

```
$ volcheck -v
media was found
```

Two status messages are possible:

`media was found` Volume Management detected the PCMCIA memory card and will attempt to mount it in the `/pcmem` directory.

If the PCMCIA memory card is formatted properly, no error messages appear in the Console.

If the PCMCIA memory card is not formatted, the "`media was found`" message is still displayed, but the following error messages appear in the Console:

```
  fd0: unformatted diskette or no diskette in
the drive
   fd0: read failed (40 1 0)
   fd0: bad format
```

You must format the PCMCIA memory card before Volume Management can mount it. Instructions are provided on page 248 (for UFS) and page 254 (for DOS).

`no media was found`	Volume Management did not detect a PCMCIA memory card. Make sure the PCMCIA memory card is inserted properly and run `volcheck` again. If unsuccessful, check the PCMCIA memory card; it could be damaged. You can also try to mount the PCMCIA memory card manually.

Verification—Loading a PCMCIA Memory Card

Verify that the PCMCIA memory card was mounted by listing its contents.

```
$ ls /pcmem/pcmem0
pcmem0 myfiles
```

As described earlier, `pcmem0` is a symbolic link to the actual name of the PCMCIA memory card; in this case, `myfiles`. If the PCMCIA memory card has no name but is formatted correctly, the system will refer to it as `unnamed_floppy`.

If nothing appears under the `/pcmem` directory, the PCMCIA memory card was either not mounted or is not formatted properly. To find out, run the `mount` command and look for the line that begins with `/pcmem` (usually at the end of the listing):

```
/pcmem/name on /vol/dev/diskette0/name ...
```

If the line does not appear, the PCMCIA memory card was not mounted. Check the Console for error messages.

▼ How to Display the Contents of a PCMCIA Memory Card

Use the -L option to the ls command, because some directories under /pcmem are symbolic links:

```
$ ls -L [-l] pcmem0
```

In this command,

-L	Includes symbolic links in the output.
-l	Long format. Includes permissions and owners in the output.

Example—Displaying the Contents of a PCMCIA Memory Card

The following example lists the contents of the PCMCIA memory card in the first floppy drive, identified by pcmem0.

```
$ ls -L -l /pcmem/pcmem0
-rwxrwxrwx  1 smith  staff 362284  Nov 16 20:54  text.doc
-rwxrwxrwx  1 smith  staff  24562  Nov 16 12:20  art.gif
```

Once you have inserted a PCMCIA memory card, you can access its files and directories just as you would those of any other file system. The only restrictions to be aware of are ownership and permissions. For instance, if you are not the owner of a file on a PCMCIA memory card, you won't be able to overwrite it on the PCMCIA memory card. Or, if you copy a file into your file system, you'll be the owner, but it won't have write permissions (because it never had them on the PCMCIA memory card); you'll have to change the permissions yourself.

▼ How to Copy or Move Information From a PCMCIA Memory Card

1. **Make sure the PCMCIA memory card is formatted and mounted.**

```
$ ls /pcmem
pcmem0   PCMCIA memory card-name
```

If the PCMCIA memory card is properly formatted and mounted, its name and the symbolic link will appear under `/pcmem`.

If nothing appears under the `/pcmem` directory, the PCMCIA memory card is not mounted. See “How to Load a PCMCIA Memory Card” on page 258. The PCMCIA memory card might also need to be formatted. See “How to Format a UFS PCMCIA Memory Card” on page 248 or “How to Format a DOS PCMCIA Memory Card” on page 254.

2. **Copy the files or directories.**

To Copy ...	Use ...
A file	`cp`
A directory	`cp -r`

Verification—Copying or Moving Information from a PCMCIA Memory Card

To verify a copy or move operation, use the `ls` command.

Examples—Copying or Moving Information from a PCMCIA Memory Card

The first example, below, moves a file (`readme.doc`) from the PCMCIA memory card to the current directory (indicated by the "`.`" symbol). The second example copies a file (`readme2.doc`) from the PCMCIA memory card to the current directory. The third example copies a directory (`morefiles`) and everything below it from the PCMCIA memory card to the current directory.

```
$ mv /pcmem/pcmem0/readme.doc .

$ cp /pcmem/pcmem0/readme2.doc .

$ cp -r /pcmem/pcmem0/morefiles .
```

▼ How to Copy or Move Information to a PCMCIA Memory Card

1. **Make sure the PCMCIA memory card is not write-protected.**
   Write-protection is controlled by a small slide switch in the end of the PCMCIA memory card.

2. **Make sure the PCMCIA memory card is formatted and mounted.**

```
$ ls /pcmem
pcmem0   PCMCIA memory card-name
```

   If the PCMCIA memory card is properly formatted and mounted, its name and the symbolic link, `pcmem0`, will appear under `/pcmem`.

   If nothing appears under the `/pcmem` directory, the PCMCIA memory card is not mounted. See "How to Load a PCMCIA Memory Card" on page 258. The PCMCIA memory card might also need to be formatted. See "How to Format a UFS PCMCIA Memory Card" on page 248 or "How to Format a DOS PCMCIA Memory Card" on page 254.

3. **Move or copy the files or directories.**

To ...	Use ...
Copy a file	`cp`
Copy a directory	`cp -r`
Move a file or directory	`mv`

Verification—Copying or Moving Information to a PCMCIA Memory Card

To verify a move or copy operation, use the `ls` command.

Examples—Copying or Moving Information to a PCMCIA Memory Card

The first example, below, moves a file (`readme.doc`) from the current directory to the PCMCIA memory card loaded into the first floppy drive (indicated by `/pcmem/pcmem0`). The second example copies a file (`readme2.doc`) from the current directory to the PCMCIA memory card loaded into the second floppy drive (indicated by `/pcmem/pcmem1`). The third example copies a directory (`morefiles`) and its contents from the `/home/smith/`directory to the PCMCIA memory card loaded into the first floppy drive.

```
$ mv readme.doc /pcmem/pcmem0

$ cp readme.doc /pcmem/pcmem1

$ cp -r /home/smith/morefiles /pcmem/pcmem0
```

▼ How to Find Out If a PCMCIA Memory Card Is Still In Use

1. **Become root.**
2. **Invoke the `fuser` command.**
   The `fuser` command lists the processes that are currently accessing the CD that you specify.

```
# fuser -u [ -k ] pcmem0
```

In this command,

`-u`	Displays the user of the PCMCIA memory card.
`-k`	Kills the process accessing the PCMCIA memory card.

Example—Finding Out If a PCMCIA Memory Card Is Still In Use

In the following example, the processes `6400c` and `6399c` are accessing the `/pcmem/pcmem0` directory, and the process owners are `root` and `smith`, respectively.

```
# fuser -u /pcmem/pcmem0
/pcmem/pcmem0: 6400c(root)  6399c(smith)
```

You can kill the processes individually (as root), or you can use the `fuser` command with the `-k` option, which kills all the processes accessing that file system:

```
# fuser -u -k /pcmem/pcmem0
/pcmem/pcmem0: 6400c(root)Killed  6399c(smith)Killed
```

The `fuser` command may not always identify all the killed processes. To be sure, run it again with the `-u` option.

How to Eject a PCMCIA Memory Card

1. **Make sure the PCMCIA memory card is not being used.**
   Remember, a PCMCIA memory card is "being used" if a shell or an application is accessing any of its files or directories.

   If you are not sure whether you have found all users of a PCMCIA memory card (a renegade shell hidden behind a desktop tool may be accessing it), use the `fuser` command, as described in "How to Find Out If a PCMCIA Memory Card Is Still In Use" on page 264.

2. **Eject the PCMCIA memory card.**

```
$ eject pcmem0
```

   You'll have to eject the PCMCIA memory card by hand. If you are running Windows, look for an onscreen message that says you can now eject the PCMCIA memory card.

   If the PCMCIA memory card is still in use, the following message appears:

```
/vol/dev/pcmem/noname: Device busy
```

   In this case, return to Step 1 and make sure no one is using the PCMCIA memory card, then eject it again.

How to Access PCMCIA Memory Cards on Other Systems

You can access a PCMCIA memory card on another system by mounting it manually into your file system—provided the other system has shared its PCMCIA memory card drive according to the instructions in "How to Make Local PCMCIA Memory Cards Available to Other Systems" on page 268.

1. **Select an existing directory to serve as the mount point or create one.**

```
$ mkdir directory
```

In this command,

directory	Is the name of the directory that you create to serve as a mount point for the other system's PCMCIA memory card.

2. **Find the name of the PCMCIA memory card you want to mount.**
When you manually mount a remote PCMCIA memory card, you cannot use the `pcmem0` or `floppy1` variables available with your local PCMCIA memory cards. You must use the exact PCMCIA memory card name. To find it, use the `ls` command on the remote system's `/pcmem` directory. If the automounter is running, you can simply `cd` to the system whose PCMCIA memory card you want to mount and then use the `ls` command. If the automounter is not running, you'll have to use another method, such as logging in remotely.

3. **As root, mount the PCMCIA memory card.**

```
# mount -F nfs -o rw system-name:/pcmem/PCMCIA memory card-name local-mount-point
```

In this command,

system-name	Is the name of the system whose PCMCIA memory card you will mount.
PCMCIA memory card-name	Is the name of the PCMCIA memory card you want to mount.
local-mount-point	Is the local directory onto which you will mount the remote PCMCIA memory card.

Verification—Accessing PCMCIA Memory Cards on Other Systems

To verify that the PCMCIA memory card is indeed mounted, use the `ls` command to list the contents of the mount point.

```
$ ls /pcmem
```

Example—Accessing PCMCIA Memory Cards on Other Systems

This example mounts the PCMCIA memory card named `MyFiles` from the remote system `mars` onto the `/pcmem` directory of the local system.

```
$ cd /net/mars
$ ls /pcmem
pcmem0     MyFiles
$ su
Password: password
# mount -F nfs rw mars:/pcmem/MyFiles /pcmem
# exit
$ ls /pcmem
MyFiles
```

How to Make Local PCMCIA Memory Cards Available to Other Systems

You can configure your system to share its PCMCIA memory cards; in other words, you can make any PCMCIA memory cards in those drives available to other systems. Once your PCMCIA memory card drives are shared, other systems can access the PCMCIA memory cards they contain simply by mounting them, as described in “How to Access PCMCIA Memory Cards on Other Systems” on page 265.

1. **Become root.**
2. **Find out whether the NFS daemon (`nfsd`) is running.**

```
# ps -ef | grep nfsd
root 14533    1 17 10:46:55 ?     0:00 /usr/lib/nfs/nfsd -a 16
root 14656  289  7 14:06:02 pts/3 0:00 /grep nfsd
```

If the daemon is running, a line for `/usr/lib/nfs/nfsd` will appear, as shown above. If the daemon is not running, only the `/grep nfsd` line will appear.

3. **Select an option from the following table.**

If ...	Then ...
`nfsd` is running	Go to Step 8
`nfsd` is *not* running	Continue with Step 4

4. **Create a dummy directory for `nfsd` to share.**

```
# mkdir /dummy-dir
```

In this command,

dummy-dir — Can be any directory name; for example, `dummy`. This directory will not contain any files. Its only purpose is to “wake up” the NFS daemon so that it notices your shared PCMCIA memory cards.

5. **Add the following entry into the** `/etc/dfs/dfstab` **file.**

```
share -F nfs -o ro [-d comment] /dummy-dir
```

When you start the NFS daemon, it will see this entry, “wake up,” and notice the shared PCMCIA memory card drive. Note that the comment (preceded by `-d`) is optional.

6. **Start the NFS daemon.**

```
# /etc/init.d/nfs.server start
```

7. **Verify that the NFS daemon is indeed running.**

```
# ps -ef | grep nfsd
root 14533    1 17 10:46:55 ?     0:00 /usr/lib/nfs/nfsd -a 16
root 14656  289  7 14:06:02 pts/3 0:00 /grep nfsd
```

8. **Eject any PCMCIA memory card currently in the drive.**

```
# eject pcmem0
```

9. **Assign write permissions to** `/etc/rmmount.conf`**.**

```
# chmod 644 /etc/rmmount.conf
```

10. **Add the following lines to** `/etc/rmmount.conf`**.**

```
# File System Sharing
share floppy*
```

These lines share any PCMCIA memory card loaded into your system's PCMCIA memory card drives.

11. **Remove write permissions from** `/etc/rmmount.conf`**.**

```
# chmod 444 /etc/rmmount.conf
```

This step returns the file to its default permissions.

12. **Load a PCMCIA memory card.**

```
—Insert the PCMCIA memory card—
# volcheck -v
media was found
```

The PCMCIA memory card you now load, and all subsequent PCMCIA memory cards, will be available to other systems. To access the PCMCIA memory card, the remote user must mount it by name, according to the instructions in "How to Access PCMCIA Memory Cards on Other Systems" on page 265.

Verification—Making Local PCMCIA Memory Cards Available to Other Systems

To verify that the PCMCIA memory card is indeed available to other systems, use the `share` command. If the PCMCIA memory card is available, its share configuration will be displayed. (The shared dummy directory will also be displayed.)

```
# share
-    /dummy  ro "dummy dir to wake up NFS daemon"
-    /MyFiles rw  ""
```

Example—Making Local PCMCIA Memory Cards Available to Other Systems

The following example makes any PCMCIA memory card loaded into the local system's PCMCIA memory card drive available to other systems on the network.

```
$ su
Password password
# ps -ef | grep nfsd
root 14522  236  7 14:06:02 pts/3 0:00 /grep nfsd
# mkdir /dummy
# vi /etc/dfs/dfstab
```

```
# /etc/dfs/dfstab
share -F nfs -o ro “fake dir” /dummy
```

```
# /etc/init.d/nfs.server start
# ps -ef | grep nfsd
root 14533    1 17 14:10:55 ?     0:00 /usr/lib/nfs/nfsd -a 16
root 14577  289  7 14:11:55 pts/3 0:00 /grep nfsd
# eject pcmem0
# chmod 644 /etc/rmmount.conf
# vi /etc/rmmount.conf
```

```
# /etc/rmmount.conf
# File System Sharing
share floppy*
```

```
# chmod 444 /etc/rmmount.conf
```

—Load the PCMCIA memory card—

```
# volcheck -v
media was found
# share
-    /dummy  ro "dummy dir to wake up NFS daemon"
-    /MyFiles rw  ""
```

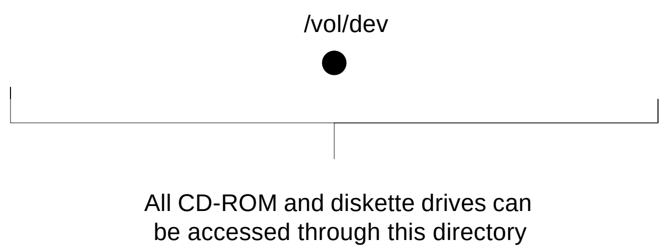
How Volume Management Works 16

This chapter describes the mount points and symbolic links that Volume Management creates to accommodate removable media.

This is a list of overview information in this chapter.

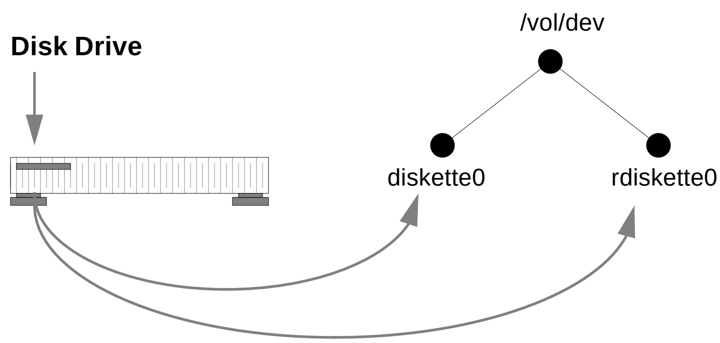
Volume Management Mounts All Removable Media

Volume Management provides access to all CD-ROM and diskette drives under `/vol/dev`:



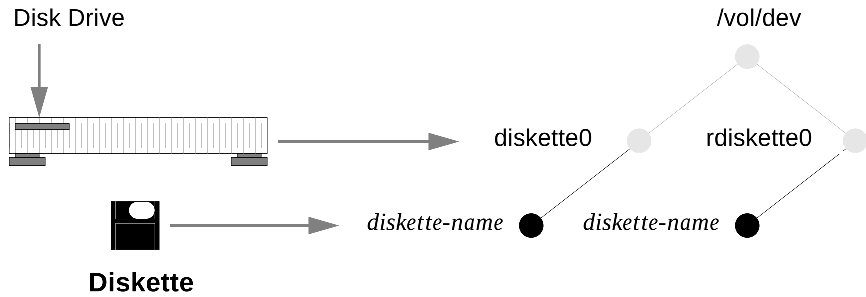
It Provides Access to Diskettes

Volume Management provides access to a system's diskette drive through subdirectories of `/vol/dev`; namely, `diskette0` and `rdiskette0`.



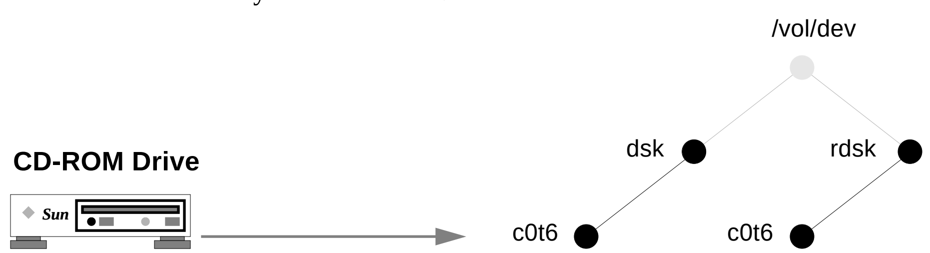
If a system has a second diskette drive, Volume Management creates a second pair of directories named `diskette1` and `rdiskette1`. For a third diskette drive, it would create `diskette2` and `rdiskette2`. And so on for additional drives.

The diskette directories provide access to file systems, and the rdiskette directories provide access to raw characters. The diskettes themselves appear in subdirectories beneath the drive directories[1]:



It Provides Access to CDs

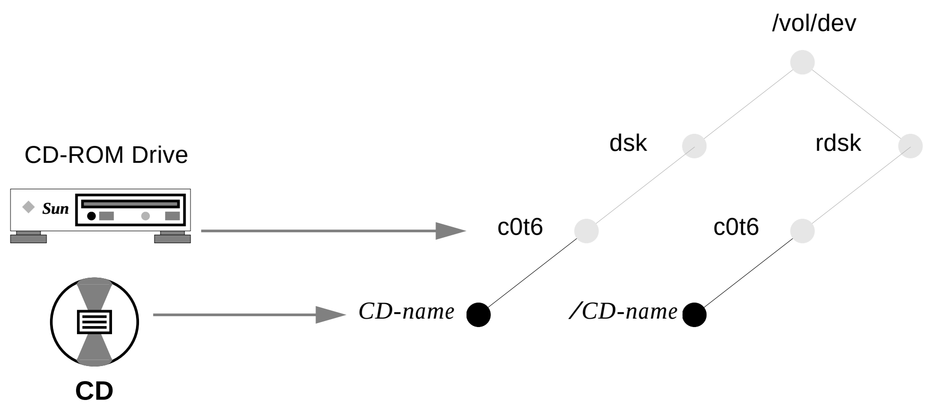
The arrangement for CDs is similar, except that the block and raw directories are labelled /dsk and /rdsk, respectively, and the CD-ROM device is actually located one directory beneath them.



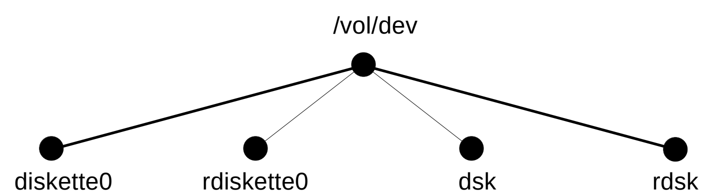
In the illustration above, the additional directory is named c0t6. That simply reflects a particular system's device naming conventions. The directory name on your system could be different, though it would have the same format.

1. In this and subsequent illustrations, some nodes are "grayed out" to draw attention to the other nodes. There is no structural significance to this convention; it is simply a means of highlighting.

The CDs themselves, however, follow a convention similar to diskettes, in that they are mounted beneath the directory belonging to their device:



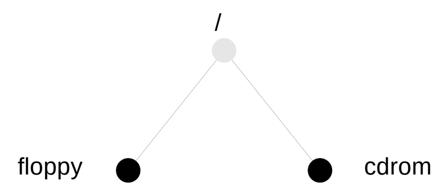
As a result of this arrangement, a system with one diskette drive and one CD-ROM drive would have the following `/vol/dev` file system:



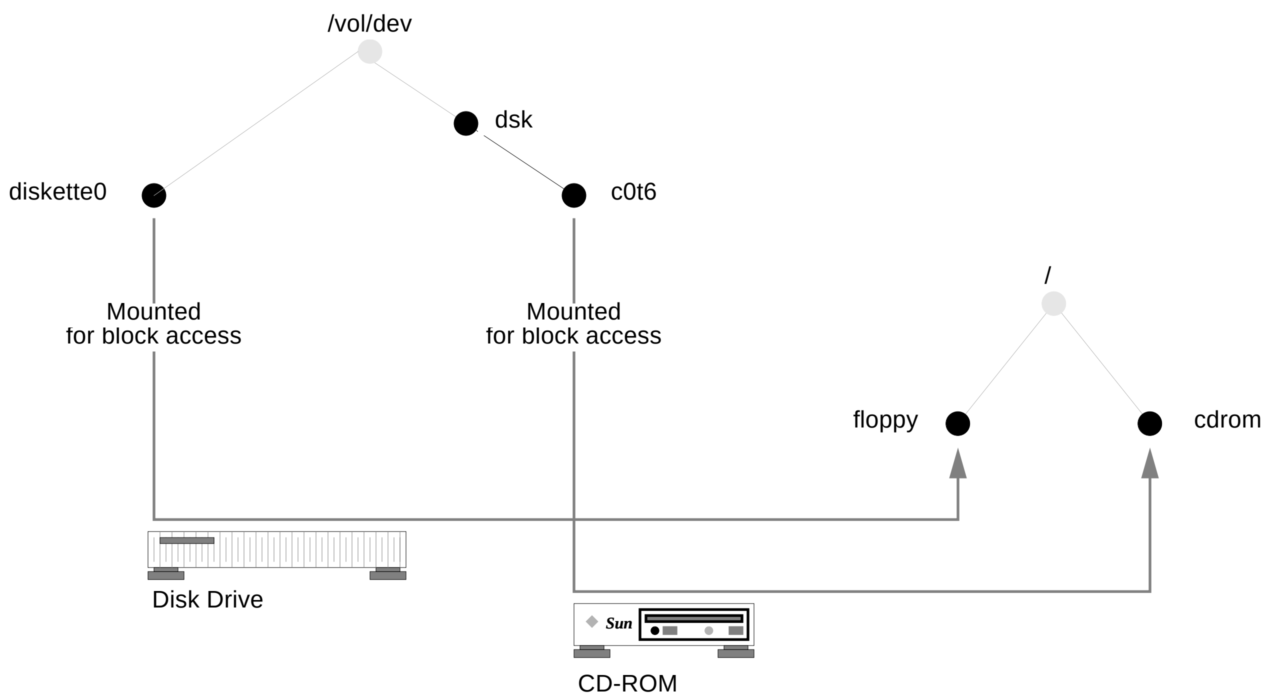
(Actually, `/vol/dev` includes an additional subdirectory named `aliases`, but that is described later in this section.)

It Supplies Convenient Mount Points for Easier Access

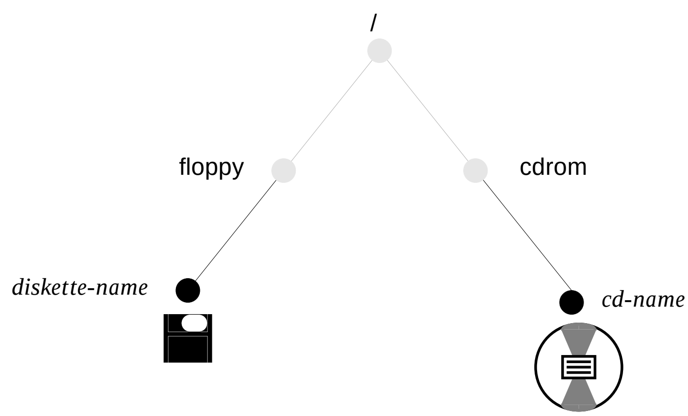
To make access more convenient, Volume Management uses two special mount points, `/floppy` and `/cdrom`.



Volume Management mounts the `/vol/dev/diskette0` and `/vol/dev/dsk/c0t6` directories onto `/floppy` and `/cdrom`:



Because of these mount points, when you insert a diskette, you can access it under `/floppy/`*diskette-name*. Likewise, when you insert a CD, you can access it under `/cdrom/`*cd-name*.



However, these mount points depend on proper formatting. If a diskette is formatted, the mount succeeds, but if it is unformatted, the mount fails and the diskette is only available under `/vol/dev/diskette0`. You can format diskettes according to the instructions in the tasks "How to Format a UFS Diskette" on page 222 or "How to Format a DOS Diskette" on page 228.

If a system has multiple drives, they are mounted onto parallel directories such as `/floppy/floppy0`, `/floppy/floppy1`, `/cdrom/cdrom0`, etc.

It Creates Two Sets of Symbolic Links

As an additional convenience, Volume Management creates two separate sets of symbolic links:

- One for file system access
- One for raw device access

Symbolic Links for File System Access

The symbolic links for file system access simply link the directories `/floppy/floppy0` and `/cdrom/cdrom0` to the diskette inserted into the first diskette drive and the CD inserted into the first CD-ROM drive:

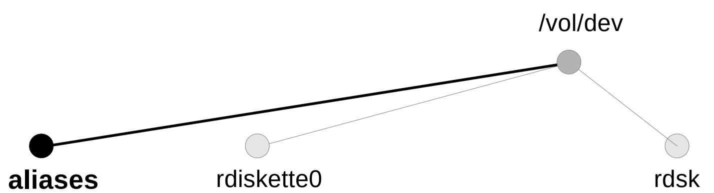
```
/floppy/floppy0 --> /floppy/name --> /vol/dev/diskette0/name
/cdrom/cdrom0   --> /cdrom/name --> /vol/dev/dsk/c0t6/CD-name
```

These links enable you to access floppies and CDs without knowing their names. You can use the link names, `floppy0` or `cdrom0`, instead.

Diskettes and CDs inserted into subsequent drives would follow the naming conventions summarized in Table 11-1 on page 191.

Symbolic Links for Raw Device Access

To make raw device access more convenient, Volume Management creates the `aliases` directory, under `/vol/dev`:

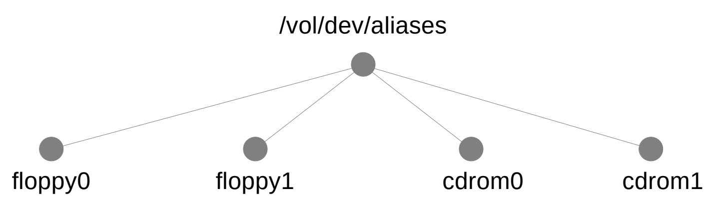


Beneath the `aliases` directory, Volume Management creates a set of symbolic links similar to those used for block access. In other words, for character access, these directories are equivalent:

```
/vol/dev/aliases/floppy0   -->   /vol/dev/rdiskette0/diskette-name
/vol/dev/aliases/cdrom0    -->   /vol/dev/rdsk/c0t6/CD-name
```

Like the symbolic links for file system access, the purpose of these links is to enable you to access a raw-character diskette or CD without knowing its name; in other words, by using the `/vol/dev/aliases/floppy0` and `/vol/dev/aliases/cdrom0` link names.

The example above shows only one symbolic link for diskettes and one for CDs. If your system had two diskettes or two CDs, there would be one symbolic link for each:



It Can Be Limited by UFS Formats

UFS formats are not portable between architectures, so they must be used on the architecture for which they were formatted. For instance, a UFS CD formatted for a SPARC system cannot be recognized by an x86 system[1]. Likewise, an x86 UFS CD cannot be mounted by Volume Management on a SPARC system. The same limitation applies to diskettes.

Therefore, Volume Management cannot recognize and mount x86 UFS media on a SPARC system—or SPARC UFS media on an x86 system.

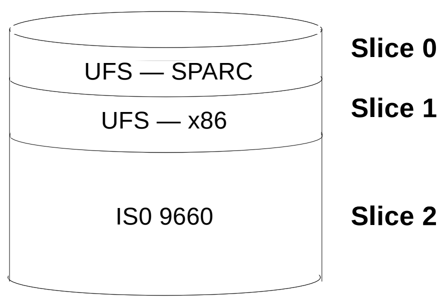
Most CDs are formatted according to the ISO 9660 standard (High Sierra File System—HSFS), which imposes no limitations on Volume Management, so incompatibility is seldom a problem with CDs.

With diskettes, UFS incompatibility can occur more often because formats can be established by the user. Be aware that if you format a UFS diskette on one architecture, you won't be able to use it on a different architecture. (For instructions, see "How to Format a UFS Diskette" on page 222.)

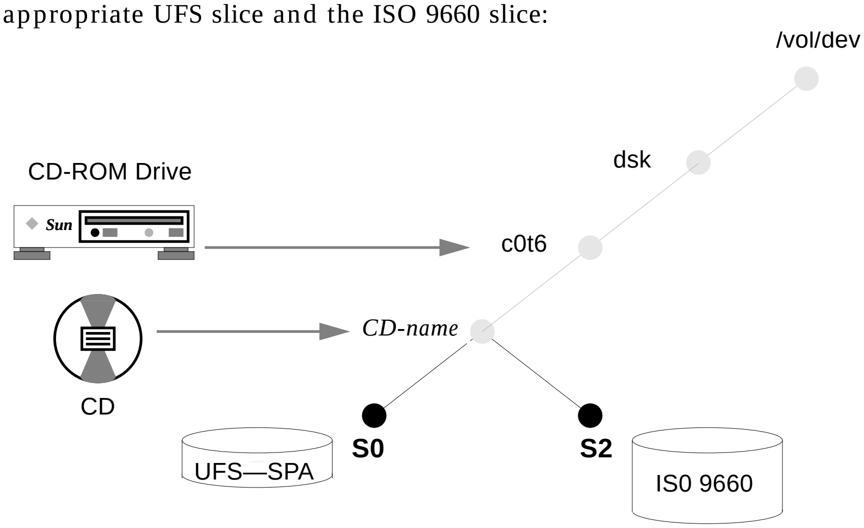
1. Actually, some architectures share the same bit structure, so occasionally a UFS format specific to one architecture will be recognized by another architecture, but the UFS file system structure was not designed to guarantee this compatibility.

What About Mixed Formats?

Some CDs, particularly installation CDs, contain mixed formats; that is, part UFS, part ISO 9660. To accommodate the different formats, the CD is split into slices, which are similar in effect to partitions on hard disks. The 9660 portion is portable, but the UFS portion is architecture-specific. Furthermore, to make the CD usable by several different architectures (as in the case of installation, when different PROM architectures might be used to boot the system), more than one UFS format is loaded onto the CD:



When Volume Management encounters this arrangement, it simply ignores the UFS formats not specific to the local system's architecture and mounts the appropriate UFS slice and the ISO 9660 slice:



These slices appear as subdirectories both under /vol/dev/dsk/c0t6 and /cdrom/cdrom0:

```
$ ls /cdrom/cdrom0
S0  S2
$ ls /vol/dev/dsk/c0t6
S0 S2
```

Part 5 — Managing Software

This part provides instructions for managing software. This includes managing Solaris, product, and AnswerBook software packages and managing patches in the Solaris environment.

Overview of Software Administration 17

Software administration involves installing and removing software from standalone systems, servers, and their clients. This chapter describes background and other useful information about installing and managing software. This chapter does not describe installing the Solaris software, which has its own installation and setup procedures.

This is a list of the overview information in this chapter.

For instructions about how to install and manage software, see Chapter 18, “Adding and Removing Packages.”

17

Software Packages

For the purpose of this discussion, software administration involves installing or removing software products. Sun and its third-party vendors deliver products in a form called a software *package*. (The term *packaging* generically refers to the method for distributing and installing software products to systems where the products will be used.) In its simplest form, you can think of a package as a collection of files and directories in a defined format. This format conforms to the Application Binary Interface, which is a supplement to the System V Interface Definition. Solaris provides a set of utilities that interpret this format and provide the means to install or remove a package or to verify its installation.

Tools for Managing Software

There are two tools for adding and removing software from a system:

- The `pkgadd` and `pkgrm` commands
- Admintool

Although either of these are appropriate to use, each has its merits.

Using the `pkgadd` and `pkgrm` commands offers flexibility. For example, you can incorporate these commands into scripts, set up optional files to avoid user interaction or perform special checks, and copy software packages to spool directories. If you're already familiar with adding and removing packages with the `pkgadd` and `pkgrm` commands, it's probably easiest for you to continue using them.

Using Admintool to add and remove software offers ease of use, because it is a graphical interface to the `pkgadd` and `pkgrm` commands and it includes online help that provides general information on using the tool. Using the Admintool graphical browser is an especially nice way to view software already installed on a system or the software that resides on the installation media. If you're unfamiliar with software package naming conventions, you're uncomfortable using command line options, and you're managing software only on one system at time, it's probably easiest for you to use Admintool to add and remove software.

Table 17-1 suggests some of the relative merits of using Admintool as opposed to using the `pkgadd` and `pkgrm` commands to manage software.

Table 17-1 Admintool Software Management Capabilities

Software Management Tasks	Performed with Admintool?
Add and remove packages on standalone or server systems, diskless or dataless clients	Yes
Easily view all installed software	Yes
Easily view and select packages from an installation media	Yes
Add packages to a spool directory	No
Eliminate user interaction by using an administration file	No

Note that prior to Solaris 2.5, Software Manager (accessed with the `swmtool` command) was the graphical tool for adding and removing software. With the release of Solaris 2.5, Admintool provides that capability. If you use the `swmtool` command on a Solaris 2.5 system, it will start Admintool.

What Happens When You Add or Remove a Package

When you add a package, the `pkgadd` software uncompresses and copies files from the installation media to a local system's disk. When you remove a package, `pkgrm` deletes all files associated with that package, unless those files are also shared with other packages. (This is true whether you use the `pkgadd` and `pkgrm` commands or Admintool to add and remove software, because Admintool is a graphical front-end to the `pkgadd` and `pkgrm` commands.)

Package files are unusable as they are delivered. The `pkgadd` command interprets the software package's control files, and then uncompresses and installs the product files onto the system's local disk.

Although the `pkgadd` and `pkgrm` commands do not log their output to a standard location, they do keep track of the product installed or removed. `pkgadd` and `pkgrm` store information about a package that has been installed or removed in a software product database.

By updating this database, the `pkgadd` and `pkgrm` commands keep a record of all software products installed on the system.

What You Should Know Before Adding or Removing Packages

Before installing or removing packages on your system, you should know:

- **Package naming conventions –** Sun packages always begin with the prefix `SUNW`, as in `SUNWvolr`, `SUNWadmap`, `SUNWabe`. Third-party packages usually begin with a prefix that corresponds to the company's stock symbol.
- **Software already installed –** You can use the `pkginfo` command to determine the software already installed on a system or you can use Admintool to view already installed software.
- **How servers and clients share software –** Clients may have software that resides partially on a server and partially on the client. If this is the case, adding software for the client requires adding packages to both the server and the client. ("Guidelines for Client Software Administration" describes in more detail how to manage client software.)

Guidelines for Client Software Administration

Managing software on a standalone system is fairly straightforward, after you're familiar with the package installation tools and conventions. You install the software package on a system's local disk and that software is then available for use. However, managing software on diskless and dataless clients can be more difficult—especially when the software resides partially on the server and partially on the client. (For example, a piece of software may have a package with files that are installed on the client's root file system and a package with files that are installed on the `/usr` file system, which the client typically mounts from a server.)

Solaris supports diskless clients, Solstice AutoClient systems, and dataless clients. On diskless clients and AutoClient systems, all software resides on the server. For example, when you add a software package to a diskless client, you don't actually install the package on the client, because it doesn't have any local disk storage device. Instead, you add the package either to the server or to the client's root file system (which resides on the server), or both. A diskless client's root file system is typically in `/export/root/`*hostname* on the server.

AutoClient systems have their own disk storage, but it is only used for caching. Software resides on a server. (See the *Solstice AutoClient 1.0 Administration Guide* for more information.)

Dataless clients have their own disk storage, but do not have the `/usr` file system installed locally. Instead, a dataless client mounts the `/usr` file system from a server.

Because diskless and dataless clients may have software partially installed on their root file system and partially installed on a server's `/usr` (or some other shared file system), adding software packages to these clients requires that you know where (in what file systems) a software package is supposed to be installed.

Installing Sun Packages on Servers and Clients

When adding packages for diskless and dataless clients, it is important to know where those packages' files are installed—in the client's root file system or in a server's `/usr` file system (or any other file system shared with the client).

Many Sun software packages are named to indicate where they are installed. For example, the `SUNWvolr` package is installed in the root file system and the `SUNWvolu` package is installed in the `/usr` file system. The "r" suffix stands for root, and the "u" suffix stands for `/usr`. However, the surest way to determine where a Sun package's files are installed is to examine the `SUNW_PKGTYPE` parameter, which is set in the package's `pkginfo` file. An interface for examining the `pkginfo` file is described in the procedure "How to Determine Where a Package's Files Will Be Installed" on page 317.

Some Sun packages do not have a `SUNW_PKGTYPE` parameter associated with them. These packages are usually set up to be installed in `/opt`. If a Sun package does not have a `SUNW_PKGTYPE` parameter value, treat it as a third-party package when installing it. (See "Installing Third-Party Packages on Servers and Clients" on page 290 for more information.)

When installing Sun packages on diskless or dataless clients, follow the general guidelines in Table 17-2.

Table 17-2 Installing Sun Packages on Clients

If The Package's Files Are Installed in The ...	Then Install the Package on The ...
root (/) file system	Client or the client's root file system
/usr (or any other shared file system)	Server

Installing Third-Party Packages on Servers and Clients

Third-party packages do not use the SUNW_PKGTYPE parameter. As a result, there is no convenient way to determine where the package's files are installed. The surest way is to examine the package's pkgmap file. Based on that information, you can install according to the guidelines in Table 17-2. However, if you want avoid having to examine a package's pkgmap file, you can use the following two-step process, which is the safest way to install third-party packages on diskless and dataless clients:

1. Install software on the server. Everything the server shares with clients is updated. (This assumes the server and clients are running the same version of Solaris software and are the same a hardware platform: for example, either both x86 systems or both SPARC systems.)

2. Install the software on the client. The pkgadd command or Admintool, whichever you're using, will install only those files appropriate for the client. The pkgadd command or Admintool will not install software already available from file systems that are mounted from a server, because that software is already available to the client.

Installing Packages in Heterogeneous Environments

There are two cases in which software management on clients/servers is further complicated:

- When the server is running a different Solaris release than the client: for example, the server is running Solaris 2.5 and it is serving Solaris 2.3 diskless and dataless clients.

- When the server and the clients are different hardware platforms: for example, the server is a SPARC system serving diskless and dataless clients that are x86 systems.

These are generically referred to as heterogeneous environments. When managing software in heterogeneous environments, you must first add the proper Solaris and architecture services appropriate for the server's clients. To do this, you use Host Manager to "add services" to the server (for detailed information, see Chapter 4, "Adding and Maintaining Server and Client Support.

For detailed information about how to add packages in a heterogeneous environment, see "Adding Packages in a Heterogeneous Client/Server Environment" on page 305.

Guidelines for Removing Packages

Because the `pkgadd` and `pkgrm` commands update information in a software products database, it is important when you remove a package to use the `pkgrm` command—even though you might be tempted to use `rm` instead. For example, you could use the `rm` command to remove a binary executable file, but that is not the same as using `pkgrm` to remove the software package that includes that binary executable. Using `rm` to remove a package's files will corrupt the software products database. (If you really only want to remove one file, you can use the `removef` command, which will update the software product database correctly. See the `removef(1M)` man page for more information.)

If you intend to keep multiple versions of a package (for example, multiple versions of a document processing application), install new versions into a different directory than the already installed package. The directory where a package is installed is referred to as the basedir, and you can manipulate the basedir by setting the `basedir` keyword in a special file called an administration file. See "Avoiding User Interaction When Adding Packages" on page 295 and the `admin(4)` man page for more information on use of an administration file and setting the base directory.

Note – If you use the upgrade option when installing the Solaris software, the Solaris installation software consults the software product database to determine the products already installed on the system.

Adding and Removing Packages 18

This chapter describes how to install, remove, and administer software packages with Solaris commands. You can also use the Admintool™ software management capability to install or remove software packages.

This is a list of step-by-step instructions in this chapter.

Commands for Handling Software Packages

Table 18-1 shows commands to use for adding, removing, and checking the installation of software packages.

Table 18-1 Commands for Adding and Removing Packages

Command	**Description**
`pkgadd(1M)`	Install a software package.
`pkgrm(1M)`	Remove a software package.
`pkgchk(1M)`	Check the installation of a software package.
`pkginfo(1M)`	List software packages installed on a system.
`pkgparam(1)`	List software packages installed on a system.

Prerequisites

All software package installation assumes that there are software packages available on some media. Typically, packages reside on a CD image, tape, diskette, or a hard disk.

A software package is a directory of files that include a `pkgmap` and a `pkginfo` file. If these files exist in the specified software package directory, `pkgadd` or Admintool, whichever you are using to install packages, will attempt to install that package.

In general, before adding new packages, remove any already installed packages with the same name. This ensures that the system keeps a proper record of software that has been added and removed. There may be times when you want to maintain multiple versions of the same application on the system. For strategies on how to do this, see the information in “Guidelines for Removing Packages” on page 291.

If you are installing a package on diskless or dataless clients, determine where the package’s files are actually installed. Most importantly, determine whether the package’s files are installed in the root (`/`) file system or in a file system such as `/usr`, which resides on the server and is shared with the client. Because diskless and dataless clients rely on a server for some of their file systems, a package may need to be installed on the server as well as the client.

Problem With Adding and Removing Packages

There is a known problem with adding or removing some packages developed before Solaris 2.5. If adding or removing the package fails during user interaction, or if you are prompted for user interaction and your responses are ignored, set the following environment variable:

```
NONABI_SCRIPTS=TRUE
```

Avoiding User Interaction When Adding Packages

When you use `pkgadd` with the `-a` option, `pkgadd` consults a special *administration* file for information about how the installation should proceed. Normally, `pkgadd` performs several checks and prompts the user for confirmation before actually adding the specified package. You can, however, create an administration file that indicates to `pkgadd` it should bypass these checks and install the package without user confirmation.

The `pkgadd` command, by default, looks in the current working directory for an administration file. If `pkgadd` doesn't find an administration file in the current working directory, `pkgadd` looks in the `/var/sadm/install/admin` directory for the specified administration file. The `pkgadd` command also accepts an absolute path to the administration file.

Caution – Use administration files judiciously. You should know where a package's files are installed and how a package's installation scripts run before using an administration file to avoid the checks and prompts `pkgadd` normally provides.

Figure 18-1 shows an example administration file that will prevent `pkgadd` from prompting the user for confirmation before installing the package.

```
mail=
instance=overwrite
partial=nocheck
runlevel=nocheck
idepend=nocheck
rdepend=nocheck
space=nocheck
setuid=nocheck
conflict=nocheck
action=nocheck
basedir=default
```

Figure 18-1 Sample Administration File

Besides using administration files to avoid user interaction when adding packages, you can use them in several other ways. For example, you can use an administration file to quit a package installation (without user interaction) if there's an error or to avoid interaction when removing packages with `pkgrm`.

You can also assign a special installation directory for a package. (It would make sense to do this if you wanted to maintain multiple versions of a package on a system.) To do this, set an alternate base directory in the administration file (using the `basedir` keyword), which specifies where the package will be installed. For more information, see the `admin(4)` man page.

Adding Packages

▼ How to Add Packages to a Standalone System

1. **Make sure you have met the prerequisites listed on page 294.**
2. **Log in as root.**
3. **Add a software package to the system.**

```
# pkgadd -a admin-file -d device-name pkgid...
```

In this command,

`-a` *admin-file*	(Optional) Specifies an administration file `pkgadd` should consult during the installation. (For details about using an administration file, see "Avoiding User Interaction When Adding Packages" on page 295.)
`-d` *device-name*	Specifies the absolute path to the software packages. *device-name* can be a the path to a device, a directory, or a spool directory. If you do not specify the path where the package resides, `pkgadd` checks the default spool directory (`/var/spool/pkg`). If the package is not there, the package installation fails.
pkgid	(Optional) Is the name of one or more packages (separated by spaces) to be installed. If omitted, `pkgadd` displays all available packages.

If `pkgadd` encounters a problem during installation of the package, it displays a message related to the problem, followed by this prompt:

```
Do you want to continue with this installation?
```

Respond with yes, no, or quit. If more than one package has been specified, type no to stop the installation of the package being installed. pkgadd continues to install the other packages. Type quit to stop the installation.

Verification—Installing Software From Mounted CD

To verify that the package has been installed successfully, use the pkgchk command.

```
# pkgchk -v pkgid
```

If pkgchk determines there are no errors, it returns a list of installed files. Otherwise, it reports the error.

Example—Installing Software From Mounted CD

The following example shows a command to install the SUNWaudio package from a mounted Solaris 2.5 CD. The example also shows use of pkgchk to verify that the packages files were installed properly.

```
# pkgadd -d /cdrom/cdrom0/s0/Solaris_2.5 SUNWaudio
    .
    .
    .
Installation of <SUNWaudio> complete.
# pkgchk -v SUNWaudio
/usr
/usr/bin
/usr/bin/audioconvert
/usr/bin/audioplay
/usr/bin/audiorecord
#
```

Example—Installing Software From Remote Package Server

If packages you want to install packages that reside on a remote system, you can manually mount the package directory and install packages on the local system. The following example shows the commands to do this. In this example, assume the remote system named `package-server` has software packages in the `/latest-packages` directory. The `mount` command mounts the packages locally on `/mnt`, and the `pkgadd` command installs the `SUNWaudio` package.

```
# mount -F nfs -o ro package-server:/latest-packages /mnt
# pkgadd -d /mnt SUNWaudio
    .
    .
    .
Installation of <SUNWaudio> was successful.
#
```

If the automounter is running at your site, you do not need to mount the remote package server manually. Instead, use the automounter path (in this case, `/net/package-server/latest-packages`) as the argument to the `-d` option.

```
# pkgadd -d /net/package-server/latest-packages SUNWaudio
    .
    .
    .
Installation of <SUNWaudio> was successful.
#
```

The following example is similar to the previous one, except it uses the `-a` option and specifies an administration file named `noask-pkgadd`, which is illustrated in Figure 18-1 on page 296. In this example, assume the `noask-pkgadd` administration file is in the default location, `/var/sadm/install/admin`.

```
# pkgadd -a noask-pkgadd -d /net/package-server/latest-packages SUNWaudio
    .
    .
    .
Installation of <SUNWaudio> was successful.
#
```

Using a Spool Directory

For convenience, you can copy frequently installed packages to a spool directory. If you copy packages to the default spool directory, `/var/spool/pkg`, you do not need to specify the source location of the package (`-d` *device-name* argument) when using `pkgadd`. The `pkgadd` command, by default, looks in the `/var/spool/pkg` directory for any packages specified on the command line.

▼ How to Add a Package to a Spool Directory

1. **Make sure you have met the prerequisites listed on page 294.**
2. **Log in as root to the server or standalone system.**
3. **Add a software package to a spool directory.**

```
# pkgadd -d device-name -s spooldir pkgid...
```

In this command,

`-d` *device-name*	Specifies the absolute path to the software packages. *device-name* can be a the path to a device, a directory, or a spool directory.
`-s` *spooldir*	Specifies the name of the spool directory where the package will be spooled. You must specify a *spooldir*.
pkgid	(Optional) Is the name of one or more packages (separated by spaces) to be added to the spool directory. If omitted, `pkgadd` displays all available packages.

Verification—Adding a Package to a Spool Directory

To verify that the package has been copied successfully to the spool directory, use the `pkginfo` command.

```
$ pkginfo -d spooldir | grep pkgid
```

If *pkgid* is installed, the `pkginfo` command returns a line of information about it. Otherwise, `pkginfo` returns the system prompt.

Example—Setting Up a Spool Directory From a Mounted CD

The following example shows a command to copy the `SUNWaudio` and `SUNWabe` packages from a mounted SPARC Solaris 2.5 CD to the default spool directory (`/var/spool/pkg`).

```
# pkgadd -d /cdrom/cdrom0/s0/Solaris_2.5 -s /var/spool/pkg SUNWaudio SUNWabe
Transferring <SUNWaudio> package instance
Transferring <SUNWabe> package instance
#
```

Example—Setting Up a Spool Directory From a Remote Package Server

If packages you want to install reside on a remote system, you can manually mount the package directory and copy it to a local spool directory. The following example shows the commands to do this. In the following example, assume the remote system named `package-server` has software packages in the `/latest-packages` directory. The `mount` command mounts the package directory locally on `/mnt`, and the `pkgadd` command copies the `SUNWman` package from `/mnt` to the default spool directory (`/var/spool/pkg`).

```
# mount -F nfs -o ro package-server:/latest-packages /mnt
# pkgadd -d /mnt -s /var/spool/pkg SUNWman
Transferring <SUNWman> package instance
#
```

If the automounter is running at your site, you do not have to mount the remote package server manually. Instead, use the automounter path (in this case, /net/package-server/latest-packages) as the argument to the -d option.

```
# pkgadd -d /net/package-server/latest-packages -s /var/spool/pkg SUNWman
Transferring <SUNWman> package instance
#
```

Example—Installing a Package From the Default Spool Directory

The following example shows a command to install the SUNWman package from the default spool directory. (When no options are used with pkgadd, it searches /var/spool/pkg for the named packages.)

```
# pkgadd SUNWman
    .
    .
    .
Installation of <SUNWman> was successful.
#
```

Adding Packages in a Homogeneous Client/Server Environment

For the purposes of this discussion, a homogeneous client/server means the clients and servers are running the same version of Solaris and are the same hardware platform (either all SPARC or all x86 platforms).

This section describes how to install packages that place files in a client's root file system. If you are installing a package for clients, and that package does not place files on the client's root file system, the package can be installed directly on the server and shared. (This assumes that the package is installed to a file system such as /usr on the server.)

Use the pkgadd command with the -R option to specify the location of the client's root file system for the client installation. (There's a common misconception that you can use the -R option to specify an alternate base directory for a package installation. That is not the case. The -R option is specifically for defining the client's root file system. To specify an alternate base directory, use `pkgadd` with the `-a` option and provide an administration file that has the `basedir` keyword set to the new installation directory.)

Note – Packages installed on the server for diskless or dataless clients are read only to the client and are shared with other clients.

Although there are several ways to install and maintain packages in a client/server environment, this section provides instructions on how to do this from a server. This is a centralized software administration model. Note, however, that you can log in to clients and install software directly on them.

Adding Sun Packages on Clients

In general, when installing Sun packages on clients in a homogeneous environment, follow the guidelines in Table 18-2.

Table 18-2 Installing Sun Packages on Clients in a Homogeneous Environment

If the Package's Files Are Installed in The ...	Then ...
root (/) file system	Add the package by using one of the following procedures, whichever is appropriate: • "How to Add a Package to a Diskless Client's Root File System" on page 305 • "How to Add a Package to a Dataless Client's Root File System" on page 310.
`/usr`	Add the package using the procedure "How to Add Packages to a Standalone System" on page 297.

You can determine where a Sun package's files are installed by using the procedure "How to Determine Where a Package's Files Will Be Installed" on page 317.

Adding Third-Party Packages on Clients

When installing third-party packages on clients, follow these guidelines:

1. Install the package on the server using the procedure "How to Add Packages to a Standalone System" on page 297.
2. Install the package on the client using the procedure "How to Add a Package to a Diskless Client's Root File System" on page 305 or "How to Add a Package to a Dataless Client's Root File System" on page 310, whichever is appropriate.

Adding Packages in a Heterogeneous Client/Server Environment

For the purposes of this discussion, a a heterogeneous client/server environment means the clients and servers are either running different versions of Solaris or are different hardware platforms (for example, a Solaris 2.3 server of Solaris 2.5 clients, or an x86 server with SPARC clients). Adding packages in a heterogeneous client/server environment presents its own difficulties. The server will have multiple `/usr` file systems for the heterogeneous clients it supports. For example, it might have an x86 `/usr` file system for its x86 clients, a Solaris 2.4 `/usr` file system for its Solaris 2.4 clients, and so on. In general, when installing packages in a heterogeneous client/server environment, follow the guidelines in Table 18-3.

Table 18-3 Installing Packages in a Heterogeneous Environment

If the Package's Files Are Installed in The ...	Then ...
root (/) file system	Add the package by using one of the following procedures, whichever is appropriate: • "How to Add a Package to a Diskless Client's Root File System" on page 305 • "How to Add a Package to a Dataless Client's Root File System" on page 310.
`/usr`	Add the package using the procedure "How to Add Packages to a Server" on page 313.

▼ How to Add a Package to a Diskless Client's Root File System

When you add a package to a diskless client, you don't actually install the package on the client, because it doesn't have any local secondary storage device. Instead, you add the package to the client's root file system, which resides on a server. A diskless client's root file system is typically in `/export/root/`*hostname* on the server.

Note – If the package's files are installed into the /usr file system, you need to install the package on the server. If you are working in a homogeneous client/server environment, use Table 18-2 to determine how to install the package. If you are working in a heterogeneous client/server environment, use Table 18-3 to determine how to install the package.

---

1. **Make sure you have met the prerequisites listed on page 294.**
2. **Log in to the server as root.**
3. **Add a software package to the diskless client's root file system.**

```
server# pkgadd -R rootpath -d device-name pkgid...
```

In this command,

-R *rootpath*	Specifies the path name of the client's root file system.
-d *device-name*	Specifies the absolute path to the software packages. *device-name* can be a the path to a device, a directory, or a spool directory. If you do not specify the path where the package resides, pkgadd checks the default spool directory (/var/spool/pkg). If the package is not there, the package installation fails.
pkgid	(Optional) Is the name of one or more packages (separated by spaces) to be installed. If omitted, pkgadd displays all available packages.

Caution – During the installation, you may see the following message:

`WARNING:` *filename* `<not present on Read Only file system>`

This indicates that not all of the package's files have been installed. The client may not have access to all files necessary for the software to work correctly. If you see this warning message, you must also install the package on the server as well as the client's root file system.

Verification—Installing a Package From a Mounted CD to a Diskless Client's Root File System

To verify that the package has successfully been installed, log in to the server, become root, and use the `pkgchk` command.

```
server# pkgchk -R rootpath -v pkgid
```

If `pkgchk` determines there are no errors, it returns a list of installed files. Otherwise, it reports the error.

You can also verify the package has been installed by logging in as root and using the `pkginfo` command.

```
server# pkginfo -R rootpath | egrep pkgid
```

The `pkginfo` command returns a line of information about the installed *pkgid*. If *pkgid* is not installed, `pkginfo` returns the system prompt.

Example—Installing a Package From a Mounted CD to a Diskless Client's Root File System

The following example shows a command to install the `SUNWadmr` (software to support system and network administration) package from a server onto a diskless client's root file system. In this case, the diskless client's root file system is `/export/root/client-1`. This example assumes the `SUNWadmr` package is available from a mounted SPARC 2.5 Solaris CD (`/cdrom/cdrom0/s0/Solaris_2.5`). The example also shows use of `pkginfo` and `pkgchk` to verify that the package's files were installed properly.

```
server# pkgadd -R /export/root/client-1 -d /cdrom/cdrom0/s0/Solaris_2.5 SUNWadmr
          .
          .
          .
Installation of <SUNWadmr> complete.
server# pkginfo -R /export/root/client-1 | egrep SUNWadmr
system     SUNWadmr   System & Network Administration Root
server# pkgchk -v -R /export/root/client-1 SUNWadmr
/etc
/etc/init.d
/etc/init.d/autoinstall
/etc/init.d/sysid.net
/etc/init.d/sysid.sys
/etc/rc2.d
/etc/rc2.d/S30sysid.net
/etc/rc2.d/S71sysid.sys
/etc/rc2.d/S72autoinstall
/sbin
/sbin/bpgetfile
server#
```

Example—Installing a Package From a Package Server to a Diskless Client’s Root File System

The following example shows a command to install the `SUNWcg6` package from a server onto a diskless client’s root file system. In this case, the diskless client’s root file system is `/export/root/client-2`. This example assumes the `SUNWcg6` package is available from a package server on the network (`/net/package-server/latest-packages`).

```
server# pkgadd -R /export/root/client-2 -d /net/package-server/latest-packages SUNWcg6
        .
        .
        .
Installation of <SUNWcg6> complete.
server#
```

How to Add a Package to a Dataless Client's Root File System

Dataless clients have their own disk storage devices, but do not have the /usr file system installed locally. Instead, a dataless client mounts the /usr file system from a server. Since part of the dataless client's file system is local and part is mounted from a remote system, adding software packages to dataless clients requires that you know where (in what file systems) a software package is supposed to be installed.

Note – If the package's files are installed into the /usr file system, you need to install the package on the server. If you are working in a homogeneous client/server environment, use Table 18-2 on page 303 to determine how to install the package. If you are working in a heterogeneous environment, use Table 18-3 on page 305 to determine how to install the package.

---

1. **Make sure you have met the prerequisites listed on page 294.**
2. **Log in to the client as root.**
3. **Use the** `share` **command so that the dataless client's root file system can be remotely mounted by the server.**

```
client# share -F nfs -o rw,anon=0 /
```

4. **Log in to the server as root.**
5. **Use the** `mount` **command to mount the client's root file system (/) on the server's** `/mnt` **directory.**

```
server# mount -F nfs client-name:/ /mnt
```

In this command,

client-name	Is the name of the dataless client.

6. Use the `pkgadd` command to add a software package to the system.

```
server# pkgadd -R /mnt -d device-name pkgid...
```

In this command,

`-R /mnt`	Specifies the mount point of the client's root file system.
`-d` *device-name*	Specifies the absolute path to the software packages. *device-name* can be the path to a device, a directory, or a spool directory. If you do not specify the path where the package resides, `pkgadd` checks the default spool directory (`/var/spool/pkg`). If the package is not there, the package installation fails.
pkgid	(Optional) Is the name of one or more packages (separated by spaces) to be installed. If omitted, `pkgadd` displays all available packages.

Caution – During the installation, you may see the following message:

`WARNING:` *filename* `<not present on Read Only file system>`

This indicates that not all of the package's files have been installed. The client may not have access to all files necessary for the software to work correctly. If you see this warning message, you must also install the package on the server as well as the client's root file system.

Verification—Adding a Package to a Dataless Client's Root File System

To verify that the package has been installed successfully on the dataless client, use the `pkgchk` or `pkginfo` commands on either the server or the dataless client. In these verification examples, the `pkgchk` and `pkginfo` commands are issued from the server.

```
server# pkgchk -v -R /mnt pkgid
```

If `pkgchk` determines there are no errors, it returns a list of installed files. Otherwise, it reports the error.

You can also verify the package has been installed by logging in as root and using the `pkginfo` command.

```
server# pkginfo -R /mnt | egrep pkgid
```

The `pkginfo` command returns a line of information about the installed *pkgid*. If *pkgid* is not installed, `pkginfo` returns the system prompt.

Example—Adding a Package to a Dataless Client's Root File System

The following example shows the commands to add a package to a dataless client's root file system. The share command is executed on the dataless client so the rest of the package installation can be accomplished from the server. This example also shows use of the pkgchk command to verify the package was installed successfully.

```
client# share -F nfs -o rw,anon=0 /

server# mount -F nfs client:/ /mnt
server# pkgadd -R /mnt -d /cdrom/cdrom0/s0/Solaris_2.5 SUNWvolr
    .
    .
    .
Installation of <SUNWvolr> was successful.
server# pkgchk -v -R /mnt SUNWvolr
/etc
/etc/init.d
/etc/init.d/volmgt
/etc/rc2.d
/etc/rc2.d/K92volmgt
/etc/rc2.d/S92volmgt
/etc/rmmount.conf
/etc/vold.conf
server# pkginfo | egrep SUNWvolr
system      SUNWvolr       Volume Management, (Root)
server#
```

▼ How to Add Packages to a Server

1. **Log in to the server as root.**

2. **Make sure the server has the OS services necessary for its diskless and dataless clients.**
   Use Host Manager to verify the OS services available on the server. If you need to add OS services, you can do that using the "Add Services" capability of Host Manager. For detailed information, see Chapter 4, "Adding and Maintaining Server and Client Support.

3. **Determine your next step based on whether the server and the diskless or dataless clients are the same Solaris release and the same hardware platform.**

If the Diskless or Dataless Clients and Server Are ...	Then ...
The same Solaris release and the same hardware architecture	Do not use this procedure. Instead, use the procedure "How to Add Packages to a Standalone System" on page 297.
Either different Solaris releases or different hardware platforms (for example, a Solaris 2.3 server of Solaris 2.5 diskless clients, or an x86 server of SPARC diskless clients)	Go to Step 4.

4. **Make a copy of the default administration file.**

```
# cp /var/sadm/install/admin/default /var/sadm/install/admin/admin-file
```

5. **Edit the new administration file and set the** `basedir` **keyword.**

   Use a text editor to edit the new administration file and set the `basedir` keyword to the correct path for the OS services supporting the client.

```
basedir=/export/exec/Solaris_2.x_platform.all/usr
```

   In this line,

`Solaris_2.`*x*	Is the Solaris version number: for example, `Solaris_2.5`
platform	Is the hardware architecture of the client: for example, i386 or sparc, as in `Solaris_2.5_i386.all` or `Solaris_2.5_sparc.all`.

6. **Add a software package to the server.**
   The administration file will specify to install the package into the `/usr` file system appropriate for the client.

```
# pkgadd -a admin-file -d device-name pkgid...
```

   In this command,

`-a` *admin-file*	(Optional) Specifies an administration file `pkgadd` should consult during the installation. By default, `pkgadd` looks in the `/var/sadm/install/admin` directory for the specified administration file. You can also specify an absolute path to an administration file.

`-d` *device-name*	Specifies the absolute path to the software packages. *device-name* can be the path to a device, a directory, or a spool directory. If you do not specify the path where the package resides, `pkgadd` checks the default spool directory (`/var/spool/pkg`). If the package is not there, the package installation fails.
pkgid	(Optional) Is the name of one or more packages (separated by spaces) to be installed. If omitted, `pkgadd` displays all available packages.

If `pkgadd` encounters a problem during installation of the package, it displays a message related to the problem, followed by this prompt:

```
Do you want to continue with this installation?
```

Respond with `yes`, `no`, or `quit`. If more than one package has been specified, type `no` to stop the installation of the package being installed. `pkgadd` continues to install the other packages. Type `quit` to stop the installation.

Verification—Adding Packages to a Server

To verify that the package has successfully been installed, use the `pkgchk` command.

```
# pkgchk -v pkgid
```

If the `pkgchk` command determines there are no errors for the specified package instance, it returns a list of installed files. Otherwise, it reports the error.

You can also use the `pkginfo` command to verify a package installation.

```
# pkginfo pkgid*
```

The pkginfo command will return all instances of the installed package. Typically, pkgadd installs duplicate versions of an already installed package as *pkgid*.1, *pkgid*.2, and so on.

Example—Installing Software From a Mounted CD

The following example shows a command to install a fictitious package SUNWtoolu, which will install files into a /usr file system. Assume that the package resides on a mounted product CD, which is mounted on /cdrom/cdrom0 by default. The pkgadd command uses an administration file named new-basedir, which specifies a new installation directory for the package. The example also shows use of pkgchk to verify that the package's files were installed properly.

```
# pkgadd -a new-basedir /cdrom/cdrom0 SUNWtoolu
    .
    .
    .
Installation of <SUNWtoolu> complete.
# pkgchk -v SUNWtoolu
/usr
/usr/bin
/usr/bin/toolconvert
/usr/bin/toolplay
/usr/bin/toolrecord
#
```

▼ How to Determine Where a Package's Files Will Be Installed

This procedure is valid only for Sun software packages. For third-party software products, the surest way to determine where the package's files will be installed is to look in the package's directory at the pkgmap file.

1. **Log in to any system.**
   You must be able to access the directory where the packages reside.

2. Determine where a Sun package's files will be installed.

```
$ pkgparam -d device-name pkgid SUNW_PKGTYPE
```

In this command,

`-d` *device-name*	Specifies the absolute path to the software packages. *device-name* can be the path to a device, a directory, or a spool directory. If you do not use the `-d` option, the `pkgparam` command will return the default installation directory of the specified *pkgid* installed on the local system.
pkgid	Is the name of a software package.
`SUNW_PKGTYPE`	Is a special parameter that reports where a Solaris software package will be installed. If the package does not have the `SUNW_PKGTYPE` parameter set, the `pkgparam` command returns an empty string. For Sun packages, this usually means the package will be installed in `/opt`.

Example—Determining Where a Package's Files Will Be Installed

```
$ pkgparam -d /cdrom/cdrom0/s0/Solaris_2.4 SUNWvolr SUNW_PKGTYPE
root
$ pkgparam -d /cdrom/cdrom0/s0/Solaris_2.4 SUNWvolu SUNW_PKGTYPE
usr
```

Checking the Installation of Packages

You use the pkgchk command to check installation completeness, path name, file contents, and file attributes of a package. See the pkgchk(1M) man page for more information on all the options.

Use the pkginfo command to display information about the packages that are installed on the system.

▼ How to Check the Integrity of an Installed Package

1. **Log in to a system as root.**

2. **Check the status of an installed package with the pkgchk command.**

```
# pkgchk -a | -c -v pkgid ...
# pkgchk -d spooldir pkgid ...
```

In this command,

-a	Specifies to audit only the file attributes (that is, the permissions), rather than the file attributes and contents, which is the default for pkgchk.
-c	Specifies to audit only the file contents, rather than the file contents and attributes, which is the default for pkgchk.
-v	Specifies verbose mode, which displays file names as pkgchk processes them.
-d *spooldir*	Specifies the absolute path of the spool directory.
pkgid	(Optional) Is the name of one or more packages (separated by spaces). If you do not specify a *pkgid*, pkgchk checks all the software packages installed on the system. If omitted, pkgchk displays all available packages.

Example—Checking the Contents of an Installed Package

The following example shows how to check the contents of a package.

```
# pkgchk -c SUNWadmfw
```

If `pkgchk` determines there are no errors, it returns the system prompt. Otherwise, it reports the error.

Example—Checking the File Attributes of an Installed Package

The following example shows how to check the file attributes of a package.

```
# pkgchk -a SUNWadmfw
```

If `pkgchk` determines there are no errors, it returns the system prompt. Otherwise, it reports the error.

Example—Checking Packages Installed in a Spool Directory

The following example shows how to check a software package copied to a spool directory (`/export/install/packages`).

```
# pkgchk -d /export/install/packages
## checking spooled package <SUNWadmap>
## checking spooled package <SUNWadmfw>
## checking spooled package <SUNWadmc>
## checking spooled package <SUNWsadml>
```

Note – The checks made on a spooled package are limited because not all information can be audited until a package is installed.

▼ How to List Information About All Installed Packages

List information about installed packages with the `pkginfo` command.

```
$ pkginfo
```

Example—Listing All Packages Installed

The following example shows the `pkginfo` command to list all packages installed on a local system, whether that system is a standalone, server, diskless client, or dataless client. The output shows the primary category, package name, and a description of the package.

```
$ pkginfo
applicationSUNWabe    Solaris 2.4 User AnswerBook
system     SUNWaccr   System Accounting, (Root)
system     SUNWaccu   System Accounting, (Usr)
system     SUNWadmap  System & Network Administration Applications
system     SUNWadmfw  System & Network Administration Framework
    .
    .
    .
```

Example—Listing All Packages Installed on a Diskless Client

In a diskless client/server setup, you may want to manage software from a central location. Since the server is the place to do this, you would need to use a variation of the `pkginfo` command. The following example shows the `pkginfo -R` command to list all packages installed on a diskless client named `io`. This command is executed from the diskless client's server.

```
server$ pkginfo -R /export/root/io
system     SUNWaccr   System Accounting, (Root)
system     SUNWaccu   System Accounting, (Usr)
system     SUNWadmap  System & Network Administration Applications
system     SUNWadmfw  System & Network Administration Framework
    .
    .
    .
```

Example—Listing All Packages Installed on a Dataless Client

Similarly, in a dataless client/server setup, you may want to monitor software packages installed on the client, but do it from the server. The following example shows the `mount` and `pkginfo` commands to list all packages installed on a dataless client. (This assumes that the dataless client named `andromeda` shares its root file system; otherwise, the mount will fail.) This command is executed from the dataless client's OS server.

```
server# mount -F nfs andromeda:/ /mnt
server# pkginfo -R /mnt
system     SUNWadmr   System and Network Administration, (Root)
system     SUNWcar    Core Architecture, (Root)
system     SUNWcsd    Core Solaris Devices, (Root)
system     SUNWcsr    Core Solaris, (Root)
    .
    .
    .
```

▼ How to Display Detailed Information About a Package

List information about installed packages with the `pkginfo -l` command.

```
$ pkginfo -l pkgid ...
```

In this command,

`-l`	Specifies to display output in long format, which includes all available information about the package.
pkgid	(Optional) Is the name of one or more packages (separated by spaces). If omitted, `pkginfo` displays information about all available packages.

Example—Displaying Detailed Information About a Package

```
$ pkginfo -l SUNWcar
PKGINST:       SUNWcar
NAME:          Core Architecture, (Root)
CATEGORY:      system
ARCH:          sparc.sun4c
VERSION:       10.0.2
BASEDIR:       /a
VENDOR:        Sun Microsystems, Inc.
DESC:          Core Architecture, (Root)
PSTAMP:        dive920521215828
INSTDATE:      Jun 03 1992  03:16
HOTLINE:       Please contact your local service provider
STATUS:        completely installed
FILES:         39 installed path names
               7 shared path names
               1 linked files
               7 directories
               21 executables
               3603 blocks used (approx)
```

Removing Packages From Servers and Standalone Systems

Caution – Always use the `pkgrm` command to remove installed packages. Do not use the `rm(1)` command, which will corrupt the system's record-keeping of installed packages.

▼ How to Remove a Package

1. **Log in to the system as root.**
2. **Remove an installed package.**

```
# pkgrm pkgid...
```

In this command,

pkgid	(Optional) Is the name of one or more packages (separated by spaces). If omitted, `pkgrm` displays all available packages. If omitted, `pkgrm` displays all available packages.

▼ How to Remove a Spooled Package

1. **Log in as root.**
2. **Remove an installed package from a spool directory with the** `pkgrm -s` **command.**

```
# pkgrm -s spooldir pkgid...
```

In this command,

`-s` *spooldir*	Specifies the name of the spool directory where the package was spooled.

pkgid	(Optional) Is the name of one or more packages (separated by spaces). If no *pkgid* is supplied, `pkgrm` prompts the user to remove each package listed in the spool directory. If omitted, `pkgrm` displays all available packages.

▼ How to Remove a Diskless Client's Package

1. **Log in to the server and become root.**
2. **Remove a software package from a dataless client's OS server with the** `pkgrm -R` **command.**

```
server# pkgrm -R rootpath pkgid...
```

In this command,

`-R` *rootpath*	Specifies the mount point of the client's root file system.
pkgid	(Optional) Is the name of one or more packages (separated by spaces) to be removed. If omitted, `pkgrm` displays all available packages.

Files in the client's package database that are marked shared are not removed from the server, but are removed from the client's database. If all clients have removed the package, you can remove the shared files from the server using a separate invocation of `pkgrm` on the server.

Verification—Removing a Diskless Client's Package

To verify that the package has successfully been removed, use the `pkginfo` command.

```
server$ pkginfo -R rootpath | egrep pkgid
```

If *pkgid* is installed, the `pkginfo` command returns a line of information about it. Otherwise, `pkginfo` returns the system prompt.

Example—Removing a Diskless Client's Package

In the following example, assume the client's root file system is shared. Also, assume these commands are executed on the client's server.

```
server# pkgrm -R /export/root/client-1 SUNWaudio
The following package is currently installed.
SUNWaudio
Do you want to remove this package? y/n/q?
y
    .
    .
    .
server#
```

▼ How to Remove a Dataless Client's Package

1. **Log in to the dataless client and become root.**
2. **Share the dataless client's root (/) file system so that it can be remotely mounted by the server.**

```
client# share -F nfs -o rw,anon=0 /
```

3. **Log in to the server as root.**
4. **On the server, use the `mount` command to mount the client's root file system (/) on the server's `/mnt` directory.**

```
server# mount -F nfs client-name:/ /mnt
```

5. **Use the `pkgrm` command to remove a software package from the system.**

```
server# pkgrm -R /mnt pkgid...
```

In this command,

`-R /mnt`	Specifies the mount point of the client's root file system.
pkgid	(Optional) Is the name of one or more packages (separated by spaces) to be removed. If omitted, `pkgrm` displays all available packages.

Files in the client's package database that are marked shared are not removed from the server, but are removed from the client's database. If all clients have removed the package, you can remove the shared files from the server using a separate invocation of `pkgrm` on the server.

Verification—Removing a Dataless Client's Package

To verify that the package has successfully been removed, use the `pkginfo` command.

```
server$ pkginfo -l | grep pkgid
```

If *pkgid* is installed, the `pkginfo` command returns a line of information about it. Otherwise, `pkginfo` returns the system prompt.

Example—Removing a Dataless Client's Package

In the following example, assume the client's root file system is shared. Also, assume these commands are executed on the client's server.

```
server# mount -F nfs client-1:/ /mnt
server# pkgrm -R /mnt SUNWvolr
The following package is currently installed.
SUNWvolr
Do you want to remove this package? y/n/q?
y
    .
    .
    .
server#
```

Managing Software Packages With Admintool

Solaris 2.5 software includes a new version of Admintool, which is a graphical user interface for performing several administration tasks, including adding and removing software packages. Specifically, you can use Admintool to:

- Add software packages to a local system
- Remove software packages from a local system
- View software already installed on the local system
- Customize software packages to be installed
- Specify an alternate installation directory for a software package

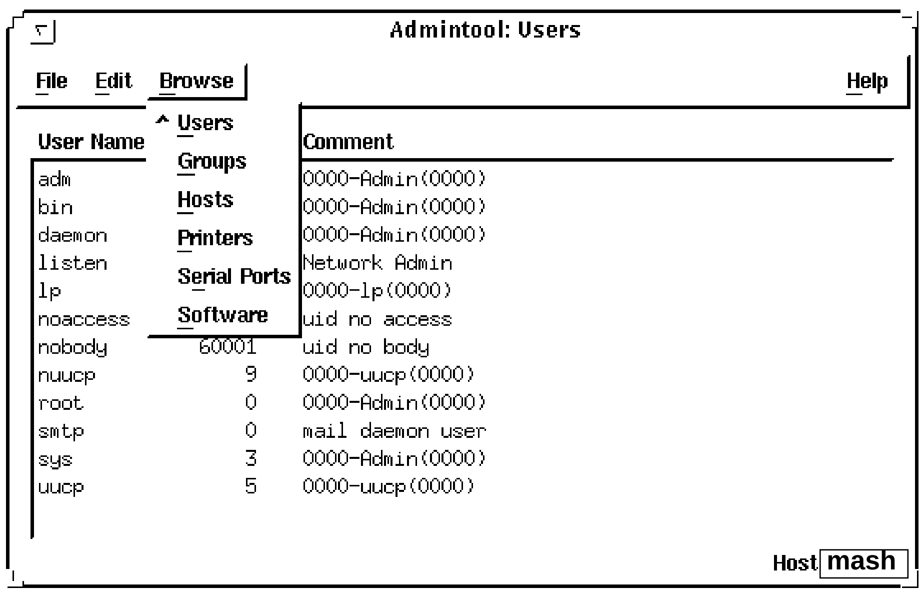
▼ How to Start Admintool

1. **Log in as root.**
   Unless you are a member of the UNIX sysadmin group (group 14), you must become root on your system to add or remove software packages with Admintool.

2. **Start Admintool.**

```
# admintool &
```

3. **Select Software from the Admintool Browse menu.**

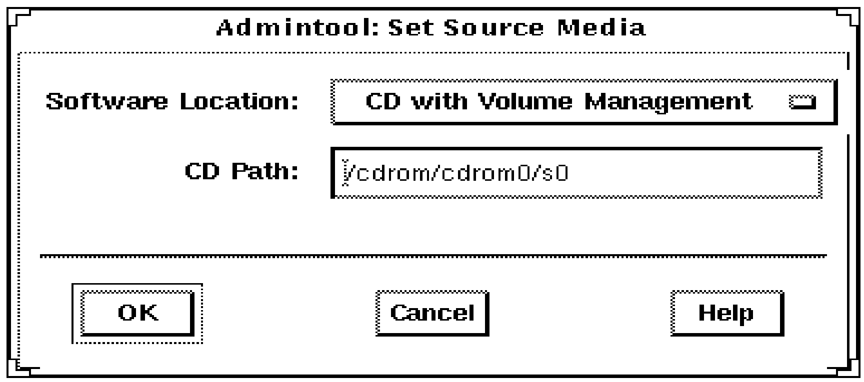


▼ How to Add Software With Admintool

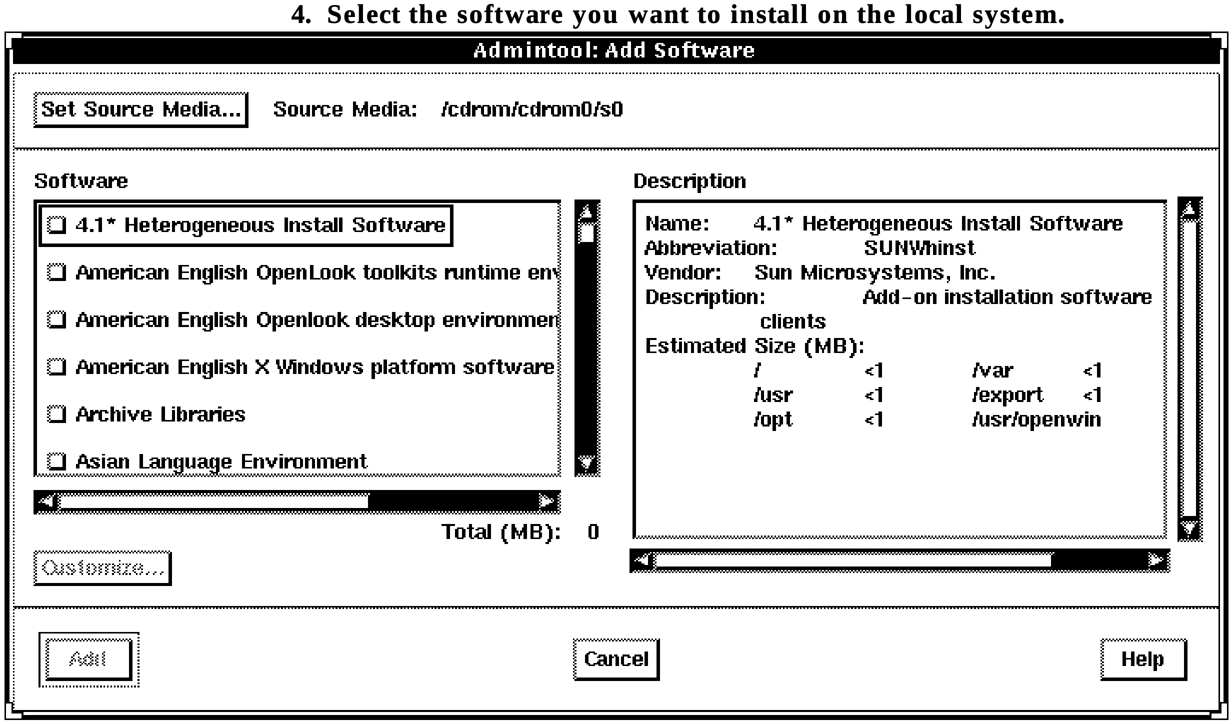
1. **Start Admintool.**

2. **Select Software from the Admintool Browse menu.**

3. **Select Add from the Edit menu.**
   The Admintool: Set Source Media window may appear. If so, specify the path to the installation media. The default path is a mounted SPARC Solaris CD.



4. **Select the software you want to install on the local system.**



5. **Click the Add button.**
   For each package selected, a window will appear prompting you for information about the installation.

Software Administration Troubleshooting 19

This chapter describes problems you may encounter when installing or removing software packages. There are two sections: Specific Software Administration Errors, which describes package installation and administration errors you might encounter, and General Software Administration Problems, which describes behavioral problems that might not result in a particular error message.

The following table shows common error messages and the page number where you can find causes and possible solutions to the related errors.

WARNING: filename <not present on Read Only file system>	*page 334*

Specific Software Administration Errors

```
WARNING: filename <not present on Read Only file system>
```

Reason Error Occurred	How to Fix the Problem
This error message indicates that not all of a package's files could be installed. This usually occurs when you are using pkgadd to install a package on a client. In this case, pkgadd attempts to install a package on a file system that is mounted from a server, but pkgadd doesn't have permission to do so.	If you see this warning message during a package installation, you must also install the package on the server. See "How to Add Packages to a Server" on page 313 for details.

General Software Administration Problems

Problem	How to Fix the Problem
There is a known problem with adding or removing some packages developed prior to Solaris 2.5. Sometimes, when adding or removing these packages, the installation fails during user interaction or you are prompted for user interaction and your responses are ignored.	Set the following environment variable and try to add the package again. NONABI_SCRIPTS=TRUE

Overview of AnswerBook Administration 20

AnswerBook software is the desktop application that delivers online documentation to the screen. Like the printed books, the AnswerBook page-by-page interface is familiar to users, and it offers the added advantages of full-text searching capability, navigation via hypertext links, and an electronic form of bookmarking.

AnswerBook documentation is shipped on a CD in a unit of software called a *package*. There are many AnswerBook packages available as part of Solaris related products.

This chapter contains background and other useful information about installing and managing AnswerBooks. Although AnswerBook packages can be installed on the desktop and run locally, this chapter focuses on issues of setting up AnswerBook on the network and making it available to clients.

This is a list of the overview information in this chapter.

For instructions about how to install and manage AnswerBook software, see Chapter 21, “Installing AnswerBook” and Chapter 22, “Setting Up AnswerBook on the Network.”

A User’s View

The AnswerBook software enables viewers to access AnswerBook titles and swap them in or out of a library, as they might use books from a shelf. Users can browse, search, set bookmarks, and print from their AnswerBook Viewer window.

Typically the user starts AnswerBook from a command line (by entering `answerbook`) or from file manager. When the new AnswerBook Navigator opens, it displays the contents of the user’s personal library. Figure 20-1 shows the AnswerBook interface displayed after a user starts the AnswerBook software.

Note – The contents of the user’s library varies according to which AnswerBook packages are installed, and, in the case of users accessing AnswerBook titles on a network, which ones have been made available by the system administrator.

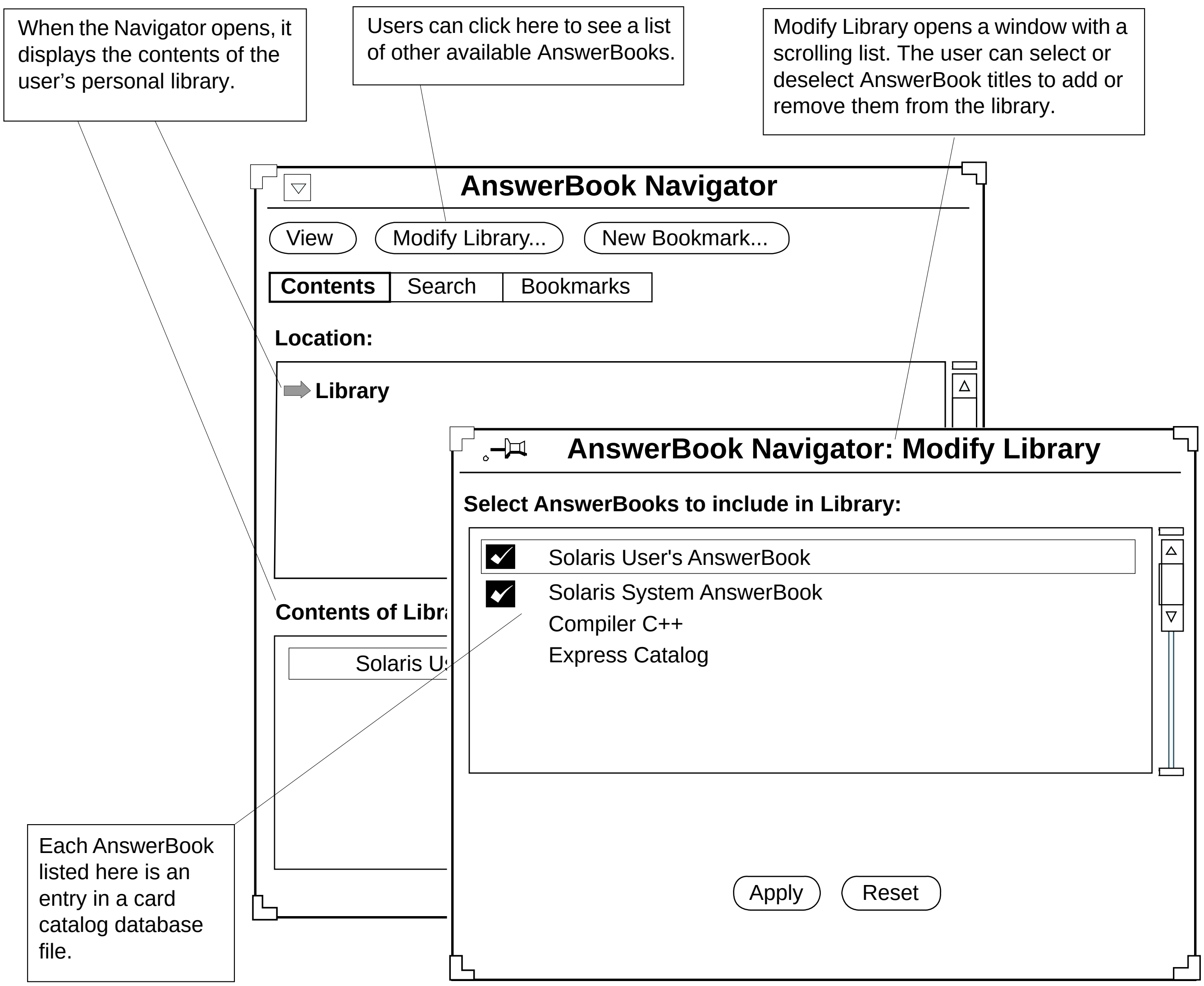


Figure 20-1 AnswerBook User Interface

A System Administrator's View

Two files and one environment variable support a user's AnswerBook library interface: the ~/.ab_library file, the master ab_cardcatalog file, and the AB_CARDCATALOG environment variable.

The personal library file is named ~/.ab_library. The AnswerBook startup software creates it the first time the user starts AnswerBook. This file stores the user's selected AnswerBook titles and bookmarks. When AnswerBook packages have been installed locally, the answerbook startup script automatically recognizes and configures them for viewing. No special procedure is necessary to enable use of AnswerBooks on a local system. (See the ab_cardcatalog(4) man page for more details.)

The ab_cardcatalog file is required when you want a central repository of AnswerBooks that you want to make available to clients on the network. Whereas the AnswerBook startup software creates and updates a user's personal ~/.ab_library file, you must create and manage the ab_cardcatalog file.

When enabling clients on a network to access a central AnswerBook server (a single system with all available AnswerBooks), you must also set an environment variable on all clients. The AnswerBook software consults the AB_CARDCATALOG environment variable to determine where on the network to look for AnswerBooks.

Guidelines for AnswerBook Administration

To enable all users on a network to browse the same set of AnswerBook titles, you can choose *either* one of the following approaches:

- **Server-side AnswerBook administration –** In this administration model, you create a network-wide AnswerBook server with a master ab_cardcatalog file. You share the master ab_cardcatalog file with all users and make its location known to clients by setting their AB_CARDCATALOG environment variable.
- **Client-side AnswerBook administration –** In this model, you create individual ~/.ab_cardcatalog) files for each user and update those personal card catalogs to reflect network-wide AnswerBook titles.

Because it's generally easier to manage information from a single server than from multiple clients, it is recommended you use the server-side model for AnswerBook administration. The following sections describe use of a master `ab_cardcatalog` file and `AB_CARDCATLOG` environment variable as they relate to the server-side administration model.

The Master `ab_cardcatalog` *File*

The `ab_cardcatalog` file is a database of AnswerBooks available to the user. The `ab_cardcatalog` file has a series of records that describe an AnswerBook. Each AnswerBook record includes its title, unique ID, and location. For example, Figure 20-2 shows a sample record for `SUNWAxg` (the Solaris XGL 3.1 AnswerBook) in a master `ab_cardcatalog` file.

```
:id=SUNWab_10_4: \
:version=: \
:title=Solaris XGL 3.1 AnswerBook: \
:tocpath=/net/marvin/opt/SUNWAxg/toc: \
:pspath=/net/marvin/opt/SUNWAxg/ps: \
:indexpath=/net/marvin/SUNWAxg/index:\
.
.
.
```

Figure 20-2 Sample AnswerBook Record in the `ab_cardcatalog` File

Every installed AnswerBook package has its own `ab_cardcatalog` file. (Before Solaris 2.2, the AnswerBook card catalog file was called the `bookinfo` file.) The master `ab_cardcatalog` file must contain all of the information from each AnswerBook `ab_cardcatalog` file. The master `ab_cardcatalog` file must be updated whenever an AnswerBook package is added or removed from the network.

The `ab_admin` command is the interface for creating and managing the `ab_cardcatalog` file. (See the `ab_admin(1)` man page for more details.) It resides in `/usr/openwin/bin` and can be used to add, remove, and modify AnswerBook records in the `ab_cardcatalog` file.

The AB_CARDCATALOG Environment Variable

To access a central AnswerBook server, each client on the network must have a local environment variable named AB_CARDCATALOG. This environment variable must be set to the path the master ab_cardcatalog file on the AnswerBook server. Then, when a user types the answerbook command, the AnswerBook startup software knows where to find all the AnswerBooks available on the network.

There are several ways to set this environment variable for users. Three are described in this section:

- Creating a site-specific initialization file and sourcing it from the users' initialization file. (This is the recommended way to set up the AB_CARDCATALOG environment variable.) In this model, you use a site-specific initialization file, and then modify the /etc/skel initialization files to source the site-specific file. This model is the easiest to administer and maintain. For a detailed procedure, see "Use Site Initialization Files" on page 24 and"How to Enable Clients to Access an AnswerBook Server" on page 369.)
- Modifying the /etc/skel initialization files. In this model, you modify the default initialization files in /etc/skel and distribute it to all new users. For example, you could set the AB_CARDCATALOG environment variable in the /etc/skel/local.profile and /etc/skel/local.cshrc files. Then, you could use these default initialization files for each new user. However, this method doesn't provide a way to set the AB_CARDCATALOG environment variable for existing users.
- Writing a shell script and making it available to all users. In this model, you write a simple shell script and distribute it from a central software server. This assumes the software server shares software and its path is in every user's initialization files. This script must be named answerbook_setup and be executable (have permissions set to 755).

  Following are sample scripts for sh and ksh:

```
#!/bin/sh
AB_CARDCATALOG=AB_net_path/ab_cardcatalog:${AB_CARDCATALOG}
export AB_CARDCATALOG
```

```
#!/bin/ksh
AB_CARDCATALOG=AB_net_path/ab_cardcatalog:${AB_CARDCATALOG}
```

In these scripts,

AB_net_path Is the path to the `ab_cardcatalog` file that will be shared with clients on the network.
Note: You must use the automounter path for clients to be able to access the server's `ab_cardcatalog` correctly. (An automounter path typically begins with `/net/`*hostname*`/`).

AnswerBook Startup Process

When the user starts AnswerBook on the desktop, the AnswerBook Navigator consults the following in this order:

1. The user's own card catalog (`~/.ab_cardcatalog`)
2. The `AB_CARDCATALOG` environment variable, which points to one or more card catalog files of available AnswerBook packages)
3. The user's AnswerBook library file (`~/.ab_library`)
4. The `pkginfo(1)` database (to find locally installed AnswerBooks)

Environment Variables Useful in AnswerBook Administration

Several environment variables support AnswerBook use and setup, as shown in Table 20-1.

Table 20-1 AnswerBook Environment Variables

Environment Variables and Settings	**Notes**
PATH=/usr/openwin/bin:${PATH}	The answerbook startup script and the ab_admin utility reside in /usr/openwin/bin.
DISPLAY=*hostname*:0.0	The DISPLAY setting is helpful when displaying AnswerBook packages on a remote system.
AB_CARDCATALOG=*AB-net-path*	*AB-net-path* is the automounter path to the master ab_cardcatalog file.

Compatibility Issues With Older Versions of AnswerBook

Before Solaris 2.2, the central database file for AnswerBooks available on the network was called the bookinfo file. This file was managed with the abmerge utility. With the Solaris 2.2 release, the bookinfo file was replaced with the ab_cardcatalog file, which is managed with the ab_admin utility.

If an old bookinfo file is on your network, convert it to an ab_cardcatalog file. You can do this with the ab_admin command. (This procedure is described in “How to Convert a bookfinfo File To an ab_cardcatalog File” on page 371.)

If your site used any individual or customized AnswerBook startup scripts *before* Solaris 2.2, they should be removed in favor of the generic AnswerBook startup script (see answerbook(1)), delivered as part of the OpenWindows package with Solaris 2.2 and subsequent releases.

Compatibility of AnswerBook Viewers and Navigators

The current AnswerBook viewer enables users to view all versions of AnswerBook packages, including those preceding Solaris 2.2. However, if you have AnswerBook software that was delivered *before* Solaris 2.2, that viewer cannot be used to view AnswerBook packages published in Solaris 2.2 or any later releases.

Bookmarks that users have created in AnswerBook titles that predate Solaris 2.2 are not compatible with the current AnswerBook navigator.

AnswerBook on X Terminals

AnswerBook pages are rendered using the Display PostScript™ system from Adobe® Systems. AnswerBook can be displayed remotely on any Solaris system that is running the Display PostScript system, which is installed by default with the appropriate fonts and software. (Display PostScript is installed as part of Solaris.)

In addition, AnswerBook may be displayed remotely on other X-based systems if they meet the following requirements:

- The remote X-based system supports the Display PostScript extension to X or supports Adobe Display PostScript NX software's remote display capabilities.
- The remote X-based system (or appropriate host) is installed with a complete LWII Type1 font set.

To determine if your system or terminal is running Display PostScript, see the procedure "How to Determine If Display PostScript Is Available" on page 374.

For additional information about the Display PostScript system, send email with your name, company, address, and phone number to:

`dps-info@adobe.com`

Note – AnswerBook sets can also be displayed on remote systems that are running NeWS.

Installing AnswerBook 21

This chapter describes how to install and administer AnswerBook packages with Solaris commands. You can also use the Admintool software management capability to install or remove AnswerBook packages.

This is a list of step-by-step instructions in this chapter.

21

Managing AnswerBook on the Network

Table 21-1 Task Map: AnswerBook Administration

Activity	Description	For Instructions, Go To	
Before You Install AnswerBook	Verify you have enough disk space for the AnswerBook you want to install.	▼ How to Check Required Disk Space for AnswerBook Packages	page 349
		▼ How to Check the Size of AnswerBook Packages	page 350
Install AnswerBook	Install AnswerBook from a product CD image, an AnswerBook CD image, or from a Solaris CD image.	▼ How to Install an AnswerBook From an AnswerBook or Product CD	page 351
		▼ How to Install the User AnswerBook From the Solaris CD	page 357
Set Up an AnswerBook Server	Set up a central `ab_cardcatalog` file. Modify it to point to all installed AnswerBook packages on the AnswerBook server.	Chapter 22, "Setting Up AnswerBook on the Network"	page 361
Set Up AnswerBook Clients	Set the `AB_CARDCATALOG` environment variable and make it available to all users wanting to access the AnswerBook server.	Chapter 22, "Setting Up AnswerBook on the Network"	page 361

Prerequisites

To install a typical AnswerBook package, you need:

- A system or server running Solaris system software.
- Access to the AnswerBook software package you want to install. This can be an AnswerBook package on a mounted CD-ROM drive or an AnswerBook package on a spool directory.
- Available disk space. The disk space required varies depending on the particular AnswerBook. The best way to calculate the space required is to change to the directory where the AnswerBook package resides (this can be a mounted AnswerBook CD or a spool directory) and use the `du -s` command.

Once you've installed the AnswerBook package, you should have the following system setup to view the AnswerBook:

- A system or server running Solaris system software and a black-and-white or color bitmap graphics monitor
- OpenWindows™ Version 3.2 or later
- For printing pages from AnswerBook: A laser printer with Palatino™ style fonts resident (such as the Sun LaserWriter™ or Sun SPARCprinter™)
- For displaying AnswerBook on X terminals, the remote X-based system supports the Display PostScript extension to X or supports Adobe Display PostScript NX software's remote display capabilities and the remote X-based system (or appropriate host) is installed with a complete LWII Type1 font set.

Considerations Before Installing AnswerBook on a Standalone System or Server

Before you install an AnswerBook package, determine whether or not there is an existing site policy for AnswerBook installation and administration. In particular, does your site use a central server for all AnswerBooks? If so, be sure to install AnswerBook packages on that server and share them with users on the network. If your site doesn't have such a policy, you may want to consider it. If you want to install AnswerBook to run only on a local system, you just need access to the AnswerBook package and enough disk space on the local system for the AnswerBook.

Also, before you install an AnswerBook, you need to know where you want to install the AnswerBook and what AnswerBook installation option you want to use. To install an AnswerBook, you can use either the `pkgadd` command or the Solaris Desktop Admintool. After you initiate the installation with one of these tools, an interactive AnswerBook installation script takes control. This script prompts you to choose an installation option (the choices are `nil` or `heavy`) and an installation directory (called the *parent* directory in the installation script).

AnswerBook Installation Options

The AnswerBook installation script also prompts you for the installation option you want to use. For each AnswerBook package you install, the installation script prompts you to choose either the `nil` or `heavy` installation option, as illustrated here:

```
Copyright information....
The installation options are as follows:
Option:      Description:
--------------------------------------------
1. nil:   less than X Megabyte disk space required
          [slowest performance].
2. heavy: XX Megabytes disk space required
          [best performance].
Enter the number of an installation option from the list above (1
or 2).

Make sure to choose a parent directory on a file system big enough
to accommodate all the files to be moved for the INSTALL OPTION
you selected.
```

A choice of `nil` takes up less disk space, but choosing `heavy` results in better AnswerBook performance. If your site policy is to use a central AnswerBook server, then you'll probably want to use the `heavy` option and make sure there's enough disk space for all the AnswerBooks you want to install on it. It makes sense to use the `nil` option when you're installing an AnswerBook on a desktop system with a dedicated CD-ROM drive.

Table 21-2 explains more about the AnswerBook installation options.

Table 21-2 AnswerBook Installation Options

Option	Description	Required Disk Space (Mbytes)	Install Time
`nil`	Leaves almost all AnswerBook files on the CD. Saves disk space but requires that the AnswerBook CD remain in the CD-ROM drive, dedicated to AnswerBook use. (This is not an option when installing the Solaris User AnswerBook.)	< 1	< 5 min.
`heavy`	Stores all AnswerBook files on hard disk. Optimizes AnswerBook performance. This configuration is recommended if the AnswerBook package is to be shared by multiple systems and users.	Varies with the package, from < 1 up to 50 or more	< 15-30 min.

AnswerBook Parent Installation Directory

AnswerBook software typically resides in `/opt.` Unless there's not enough disk space in `/opt` for the AnswerBook or your site installs AnswerBook software in another directory, choose `/opt` as the parent installation directory when prompted by the AnswerBook installation script.

If you're concerned about available disk space for the installation, you can check the available disk space and determine the *approximate* size of the contents of the CD before you install the AnswerBook package(s).

▼ How to Check Required Disk Space for AnswerBook Packages

1. **Determine whether or not there is sufficient disk space in the AnswerBook installation directory.**
   Typically AnswerBook packages are installed in `/opt`, but they can be installed in any directory with enough disk space.

```
# df -kl /opt
```

2. **Compare the available space in the list with the sizes listed for the** `nil` **and** `heavy` **installations.**
   In the example below, the system has enough space in the `/opt` partition to do a heavy installation of an AnswerBook package under 18 Mbytes.

```
Filesystem      kbytes  used   avail capacity mounted on
/dev/dsk/c0t0d0s531966  10837  17939 38%      /opt
```

▼ How to Check the Size of AnswerBook Packages

1. **Change directory to directory where the AnswerBook package resides.**
   This AnswerBook package can be on a mounted AnswerBook CD or a spool directory.

```
# cd AB-path
```

2. **Determine the approximate amount of space required by the AnswerBook by using the** `du -s` **command.**
   This will tell you how much disk space you'll need when you install.

```
# du -s
```

Installing Any AnswerBook From an AnswerBook or Product CD

AnswerBook software packages are generally distributed on their own CDs. For example, the System Administration AnswerBook is on its own CD. AnswerBook software packages may also be part of another product CD. For example, the Online Disksuite™ product includes its own AnswerBook package. (The exception is the User AnswerBook, which is distributed as part of the Solaris software CD.) The following section describes how to install AnswerBook from an AnswerBook or product CD.

These instructions generically refer to installing from a CD. However, if an AnswerBook software package has been copied from a CD to a spool directory, you can also install the package from the spool directory.

How to Install an AnswerBook From an AnswerBook or Product CD

1. **Log in as root.**
2. **If you are installing an AnswerBook package from a spool directory, skip to Step 3. If you are installing an AnswerBook package from a CD, insert the CD into the CD-ROM drive.**

Note – The Volume Management software [1] automatically mounts CDs on /cdrom/cdrom0 [2].

---

3. **Use the** `pkgadd` **command to add the AnswerBook software package to the system.**
   After you initiate the installation with the `pkgadd` command, an interactive AnswerBook installation script will control the rest of the installation.

```
# pkgadd -d AB-path pkgid...
```

In this command,

`-d` *AB-path*	Specifies the absolute path to AnswerBook software packages. This path is a path to one of the following: • Mounted AnswerBook or other product CD (`/cdrom/cdrom0`). • Spool directory (If the spool directory is a remote system, either mount the spool directory locally and use that path or use an automounter path to it: for example, `/net/system/spool/packages`.
pkgid	Is the name of one or more packages to be installed.

1. If you have disabled volume management, mount the CD using the `mount(1M)` command.
2. Volume management mounts SPARC Solaris CDs on `/cdrom/cdrom0/s0` and x86 Solaris CDs on `/cdrom/cdrom0/s2`.

Note – All CD paths in this table assume the Volume Management software is running. If you have disabled the Volume Management software, mount the CD by using the `mount(1M)` command and use the mount path as the AB-path.

4. **If you do not specify a *pkgid* on the command line, the `pkgadd` command presents a numbered list of all the packages available in *AB-path*. Specify one or all AnswerBook packages by number and press Return.**
   Each AnswerBook package will be installed successively. Prior to each package installation you will be prompted for information. Alternatively, you can install a single AnswerBook package by typing its number from the list.

5. **You are prompted to choose an AnswerBook installation option. Type `1` for `nil`, `2` for `heavy`.**
   (If you're concerned about the ramifications of this decision, see "AnswerBook Installation Options" on page 348.)

```
Enter the number of an installation option from the list above
(1 or 2).
2
```

6. **You are prompted for the name of the parent directory for the package.**
   AnswerBook packages are typically installed in `/opt`.

```
Specify the parent of the AnswerBook home directory:
/opt
```

7. **You are prompted to complete the installation.**

```
Do you want to continue with the installation of this package?
[y, n,?]
y
```

When the installation completes, you see this message:

```
Installation was successful.
```

If you used the `pkgadd` command without giving it a *pkgid* argument, and there are other AnswerBook packages that can be installed, you also see this prompt:

```
Select package(s) you wish to process (or 'all' to process all
packages). (default: all) [?,??,q]:
```

Type `q` to quit the installation or specify another AnswerBook package and continue installing.

When you are done installing AnswerBooks, verify that the installation completed successfully.

Verification—Installing an AnswerBook From an AnswerBook or Product CD

To verify that the AnswerBook package or packages have been installed successfully, use the `pkginfo` and `pkgchk` commands.

```
# pkginfo | egrep pkgid ...
    .
    .
    .
# pkgchk pkgid ...
```

If `pkgchk` determines there are no errors, it returns the system prompt. Otherwise, it reports the error.

Example—Installing an AnswerBook From an AnswerBook or Product CD

This example shows use of `pkgadd` on an AnswerBook CD with multiple AnswerBook packages on it. Assume the AnswerBook CD is mounted on a local CD-ROM drive. The default volume management mount point is `/cdrom/cdrom0`. Because no *pkgid* is specified on the command line, the `pkgadd` command list all packages on the CD. In this example, the AnswerBook package is installed in `/opt`, which is the typical AnswerBook installation directory.

```
# pkgadd -d /cdrom/cdrom0
1   SUNWabc    An AnswerBook
       (pltfrm) 1.2.1
2   SUNWcde    Another AnswerBook
       (pltfrm) 40.5.2
3   SUNWefg    Yet Another AnswerBook
       (pltfrm) 78.9.3

Select package(s) you wish to process (or ‘all’ to process all packages). (default: all)
[?,??,q]:
3
Copyright information....
The installation options are as follows:
Option:      Description:
--------------------------------------------
1. nil:   less than X Megabyte disk space required
          [slowest performance].
2. heavy: XX Megabytes disk space required
          [best performance].
Enter the number of an installation option from the list above (1 or 2).

Make sure to choose a parent directory on a file system big enough to accommodate all the
files to be moved for the INSTALL OPTION you selected.

Enter the number of an installation option from the list above (1 or 2).
2
Specify the parent of the AnswerBook home directory:
/opt
Do you want to continue with the installation of this package? [y, n,?]
y
    .
    .
    .
Installation was successful.
    .
    .
    .
Select package(s) you wish to process (or 'all' to process all packages). (default: all)
[?,??,q]:
q
```

Where to Go Next

If you are planning to run the AnswerBook only on the system where it is installed, you're done; it will be accessed automatically the next time you run the AnswerBook software.

If you want to make this AnswerBook available to users on other systems, follow the procedures in Chapter 22, "Setting Up AnswerBook on the Network."

Installing the End User AnswerBook From the Solaris CD

The Solaris User AnswerBook is a package of online documentation written for users of the Solaris desktop. It is included on the Solaris CD and consumes about 20 Mbytes of available disk space when installed.

There are two ways to install the Solaris User AnswerBook:

- As part of the Solaris installation, you can install it into a predetermined directory on the local system. To do so, you choose "customize" at the appropriate point during Solaris installation and include the Solaris User AnswerBook package (`SUNWabe`). This package is installed in the `/opt` directory during Solaris installation.

  The Solaris User AnswerBook package is installed on the local system by default if you have chosen the "Entire Distribution" software group. (It is also installed automatically by certain configurations of JumpStart installation.)

- You can install the Solaris User AnswerBook *after* Solaris installation, using `pkgadd`, as shown in the following section. Many sites prefer to install the Solaris User AnswerBook this way, installing a single copy in the same place as other AnswerBook packages on their network.

Note – There is no `nil` installation option for the Solaris User AnswerBook.

▼ How to Install the User AnswerBook From the Solaris CD

Because the Solaris User AnswerBook package is part of the Solaris CD, installation steps are slightly different from those of other AnswerBook packages.

1. **Determine your first step based on which directory you want to install AnswerBook in.**

You Want to Install AnswerBook In ...	Then ...
`/opt` (the default installation directory)	Go to Step 2.
A directory of your choice	Go to Step 4.

2. **Use the `pkgadd` command to install the User AnswerBook package (`SUNWabe`) into the `/opt` directory.**

```
# pkgadd -d AB-path SUNWabe
```

In this command,

AB-path	Specifies the absolute path to the User AnswerBook on a Solaris CD image. This path is a path to one of the following: • Mounted SPARC Solaris CD (`/cdrom/cdrom0/s0/Solaris_2.5`) • Mounted x86 Solaris CD (`/cdrom/cdrom0/s2/Solaris_2.5`) • Copy of either a SPARC or x86 Solaris CD on a hard disk

---

Note – All CD paths in this table assume the Volume Management software is running. If you have disabled the Volume Management software, mount the CD by using the `mount(1M)` command and use the mount path as the *AB-path*.

---

3. **Go to "Verification—Installing the User AnswerBook From the Solaris CD" on page 359.**
   Step 4 is required only if you want to install the AnswerBook in a directory other than `/opt`.

4. **Use the** `pkgadd` **command to install the** `SUNWabe` **(AnswerBook) package into the directory of your choice.**

```
# pkgadd -d AB-path -a none SUNWabe
Enter the path to the package base directory [?,q] AB-installpath
```

In this command,

AB-path	Specifies the absolute path to the User AnswerBook on a Solaris CD. This path is a path to one of the following: • Mounted SPARC Solaris CD (`/cdrom/cdrom0/s0/Solaris_2.5`) • Mounted x86 Solaris CD (`/cdrom/cdrom0/s2/Solaris_2.5`) • Copy of either a SPARC or x86 Solaris CD on a hard disk
AB-installpath	Specifies the path where the AnswerBook will be installed.

Note – All CD paths in this table assume the Volume Management software is running. If you have disabled the Volume Management software, mount the CD by using the `mount(1M)` command and use the mount path as the *AB-path*.

A list of files is displayed as the package is installed. When the package installation is complete you will see:

```
Installation of <SUNWabe> was successful.
```

When you are done installing the User AnswerBook, verify that the installation was completed successfully.

Verification—Installing the User AnswerBook From the Solaris CD

To verify that the AnswerBook package has been installed successfully, use the `pkginfo` and `pkgchk` commands.

```
# pkginfo | egrep SUNWabe
application SUNWabe   Solaris 2.5 User AnswerBook
# pkgchk SUNWabe
#
```

The `pkgchk` command will return a shell prompt if the `SUNWabe` files have been installed correctly. Otherwise, it will report errors.

To verify the locally installed AnswerBook can be accessed, verify the installation with a trial run.

To start the AnswerBook application, either double-click on its icon in File Manager or enter:

```
$ /usr/openwin/bin/answerbook &
```

If OpenWindows has been installed somewhere other than `/usr/openwin` (which is the default installation directory), then use the command:

```
$ /openwin-install-path/openwin/bin/answerbook &
```

When the AnswerBook Navigator opens, click the Modify Library button and select AnswerBook titles from the scrolling list so they are included in the Navigator Contents.

Where to Go Next

If you are planning to run the Solaris User AnswerBook only on the system where it is installed, you're done; it will be accessed automatically the next time you run the AnswerBook software.

If you want to make the User AnswerBook available to users on other systems, follow the procedures in Chapter 22, "Setting Up AnswerBook on the Network."

Removing an AnswerBook Package

You may decide you want to remove an AnswerBook package. For example, during the installation you may have chosen the `nil` option, but now you want to install the complete `heavy` AnswerBook package.

To remove an AnswerBook software package, follow the instructions for removing software packages on Chapter 18, "Adding and Removing Packages."

Caution – Always use the `pkgrm` command to remove installed AnswerBook packages. Do not use the `rm(1)` command, which will corrupt the system's record-keeping of installed packages.

Setting Up AnswerBook on the Network 22

This chapter describes how to set up the network so that clients can access AnswerBook from a central server. This chapter also describes issues with providing client's access to older versions of AnswerBook and with displaying AnswerBook.

This is a list of step-by-step instructions in this chapter.

22

Setting Up an AnswerBook Server and AnswerBook Clients

Table 22-1 Task Map: AnswerBook Network Setup

Activity	Description	For Instructions, Go To	
Install AnswerBook	Install AnswerBook from a product CD image or from an AnswerBook CD image.	Chapter 21, “Installing AnswerBook”	page 345
Set Up an AnswerBook Server	Set up a central `ab_cardcatalog` file. Modify it to point to all installed AnswerBook packages on the AnswerBook server.	▼ How to Set Up an AnswerBook Server on the Network	page 363
Set Up AnswerBook Clients	Set the `AB_CARDCATALOG` environment variable and make it available to all users wanting to access the AnswerBook server.	▼ How to Enable Clients to Access an AnswerBook Server	page 369
Manage the ab_cardcatalog File	Convert old-style bookinfo files to `ab_cardcatalog` files, add entries and maintain the `ab_cardcatalog` file.	▼ How to Convert a bookfinfo File To an ab_cardcatalog File ▼ How to Manage the ab_cardcatalog File	page 371 page 372
Manage the AnswerBook Display	Verify that Display PostScript is available for displaying AnswerBook documentation.	▼ How to Determine If Display PostScript Is Available	page 374

Prerequisites

To set up an AnswerBook server on the network, you need:

- A system or server running Solaris system software.
- Disk space to hold numerous AnswerBooks. (The disk space required for any given AnswerBook varies, so there's no sure recommendation.)
- The automounter running. (There are setup files on the server that the `answerbook` startup script expects to access via the automounter.)

Once you've installed the AnswerBook package, network clients should have the following system setup to view the AnswerBook documentation:

- A system or server running Solaris system software and a black-and-white or color bitmap graphics monitor.
- OpenWindows™ Version 3.2 or later.
- For printing pages from AnswerBook: a laser printer with Palatino™ style fonts resident (such as the Sun LaserWriter™ or Sun SPARCprinter™).
- The automounter running. (There are setup files on the server that the `answerbook` startup script expects to access via the automounter.)

Setting Up an AnswerBook Server

For networked sites, it makes sense to install all AnswerBook packages on a single system and create an AnswerBook server. This centralizes AnswerBook installation and administration, and frees disk space and the CD-ROM drive on desktop systems. The remainder of this section shows the procedure to set up an AnswerBook server.

▼ How to Set Up an AnswerBook Server on the Network

This procedure assumes the AnswerBook package is already installed on a system. For details on how to install an AnswerBook package, see Chapter 21, "Installing AnswerBook."

1. **Choose a system to be an AnswerBook server for other clients on the network.**

2. **Log in to the AnswerBook server as root.**

3. **Create or update the master** `ab_cardcatalog` **file with information about the new AnswerBook.**

```
# /usr/openwin/bin/ab_admin -file AB-net-path -merge
AB-install-dir/ab_cardcatalog
```

In this command,

AB-net-path	Is the path to the `ab_cardcatalog` file that will be shared with clients on the network, for example: `/export/share/ab_cardcatalog`
AB-install-dir	Is the path to the installed AnswerBook. The `/opt` directory is typically the default AnswerBook installation directory: for example, `/opt/SUNWaadm`

Clients accessing the AnswerBook server will reference this `ab_cardcatalog` file to determine the AnswerBooks that are available for viewing.

The `ab_admin` command copies information from the `ab_cardcatalog` in the AnswerBook installation directory into the master `ab_cardcatalog` file.

4. **Edit the master** `ab_cardcatalog` **file so that the** `tocpath`**,** `pspath`**, and** `indexpath` **entries use automounter path names. (Automounter path names typically begin with** `/net/`*hostname***.)**

By default, ab_admin sets the tocpath, pspath, and indexpath to the location of the installed AnswerBook packages on the AnswerBook server. For example, if the SUNWAxg AnswerBook were installed in /opt, the ab_cardcatalog file would look like this:

```
id=SUNWAxg_10_6
title=...
tocpath=/opt/SUNWAxg/toc
pspath=/opt/SUNWAxg/ps
indexpath=/opt/SUNWAxg/index
```

These paths must have automounter paths (for example, tocpath=/net/*hostname*/opt/SUNWAxg/toc) so that clients can access the AnswerBook server's AnswerBook packages successfully. Manually edit the master ab_cardcatalog file according to the following conventions.

```
id=SUNWAxg_10_6
title=...
tocpath=/net/hostname/AB-install-dir/AB-name/toc
pspath=/net/hostname/AB-install-dir/AB-name/ps
indexpath=/net/hostname/AB-install-dir/AB-name/index
```

Note – This assumes your site uses the default automounter mount point, which is /net. If your site uses a different automounter mount point, substitute that mount point for /net.

In this ab_cardcatalog file,

/net	Is the default mount point for the automounter. If your site uses another automounter mount point, use that mount point instead of /net.
hostname	Is the name of the AnswerBook server.
AB-install-dir	Is the path to the installed AnswerBook. The /opt directory is typically the default AnswerBook installation directory.

AB-name — Is the name of the installed AnswerBook package, for example: `SUNWAxg`.

5. **Repeat Step 3 and Step 4 for all AnswerBooks you want to be accessed from the AnswerBook server.**

6. **Edit the `/etc/dfs/dfstab` file and add these lines.**

```
share -F nfs -o ro AB-net-path
share -F nfs -o ro AB-install-dir
```

In this file,

AB-net-path — Is the directory with the master `ab_cardcatalog` file, for example: `/export`.

AB-install-dir — Is the directory with the installed AnswerBook. The `/opt` directory is typically the default. For example, for the SUNWAxg AnswerBook installed in `/opt/SUNWAxg`, share `/opt`.

7. **Share the file systems with the *AB-net-path* and *AB-install-dir* file systems.**
This makes the master `ab_cardcatalog` and related files available when clients on the network start the AnswerBook software.

```
# shareall
```

When you are done setting up the AnswerBook server, verify that the master `ab_cardcatalog` is correct and that the `ab_cardcatalog` and AnswerBook installation directories are shared.

Verification—Setting Up an AnswerBook Server on the Network

To verify that the paths in the master `ab_cardcatalog` file can be mounted by other clients on the network, examine the master `ab_cardcatalog` file. In this case, assume the master `ab_cardcatalog` file is on a system named `mars` and in the /export/share directory. The tocpath, pspath, and indexpath are correctly set to a path for clients running the automounter (a path that begins with /net/*hostname*/).

```
mars# more /export/share/ab_cardcatalog
::::::::::::::
ab_cardcatalog
::::::::::::::
#<Card Catalog> version 1
#
# This file was generated by cardcatadmin.
# DO NOT EDIT THIS FILE BY HAND.
:id=SUNWab_10_4: \
:version=: \
:title=Solaris XGL 3.1 AnswerBook: \
:tocpath=/net/mars/opt/SUNWAxg/toc:
:pspath=/net/mars/opt/SUNWAxg/ps:
:indexpath=/net/mars/opt/SUNWAxg/index:
:id=SUNWab_8_22:
:version=: \
:title=Solaris 2.4 User AnswerBook: \
:tocpath=/net/mars/opt/SUNWabe/toc:
:pspath=/net/mars/opt/SUNWabe/ps:
:indexpath=/net/mars/opt/SUNWabe/index:
```

To verify that the tocpath, pspath, and indexpath are set correctly and are readable, use the ab_admin command. If there is an error, the ab_admin command will report it. Otherwise, the command will return the prompt.

To verify a single AnswerBook entry, use this ab_admin command.

```
# ab_admin -file /net/mars/export/share/ab_cardcatalog -verify
SUNWab_10_4
#
```

To verify all AnswerBooks in the ab_cardcatalog file, use these commands.

```
# /bin/sh
$ for AB in ‘ab_admin -list -file $AB_CARDCATALOG‘
> do
> echo Verifying $AB
> ab_admin -verify $AB -file $AB_CARDCATALOG
> done
Verifying SUNWab_10_4
Verifying SUNWab_8_22
    .
    .
    .
```

To verify that the *AB-net-path* and *AB-install-path* are shared so that client's trying to access the AnswerBook server can mount them, use the `share` command. For example, assume an AnswerBook server named `mars` has AnswerBooks installed in `/opt` and has a master `ab_cardcatalog` file in `/export/share`. `/opt` and `/export` need to be shared.

```
mars# share
-      /export    ro ""
-      /opt       ro ""
```

Where to Go Next

After setting up the AnswerBook server, there is still some work required before clients on the network can access AnswerBook from the server. For details, see "Enabling Clients to Access the AnswerBook Server" on page 368.

Enabling Clients to Access the AnswerBook Server

To access a central AnswerBook server, each client on the network must set a local environment variable named `AB_CARDCATALOG`. This environment must be set to the path the master `ab_cardcatalog` file on the AnswerBook server. Then, when a user types the `answerbook` command, the AnswerBook startup software knows where to find all the AnswerBooks available on the network.

There are several ways to set this environment variable for users. Three are described in detail in "The AB_CARDCATALOG Environment Variable" on page 340. This section describes only the recommended way to distribute the `AB_CARDCATALOG` environment variable. This method involves creating a site-specific initialization file and sourcing it from the users's initialization file.

▼ How to Enable Clients to Access an AnswerBook Server

Enabling clients to access an AnswerBook server involves making a special AnswerBook environment variable to all clients. This section shows one model for distributing that environment variable. However, this model for distributing the environment variable has implications beyond AnswerBook administration. It is discussed in detail in the section titled "Use Site Initialization Files" on page 24.

1. **Make sure your site has site-specific initialization files that are somehow available to all users.**

   For detailed information on setting up site-specific initialization files and making information in those files available to users, see the section titled "Use Site Initialization Files" on page 24.

2. **Modify the site-specific `csh` and `sh` initialization files.**
   For `csh` users, add the following line:

```
setenv AB_CARDCATALOG AB-net-path
```

   For `sh` users, add the following line:

```
AB_CARDCATALOG=AB-net-path:${AB_CARDCATALOG}
export AB_CARDCATALOG
```

In these site-specific initialization files,

AB-net-path	Is the path to the `ab_cardcatalog` file that will be shared with clients on the network, for example: `/net/moon/export/share/ab_cardcatalog` **Note:** You must use the automounter path for clients to be able to access the server's `ab_cardcatalog` correctly. (An automounter path begins with `/net/`*hostname*`/`).

Verification—Enabling Clients to Access an AnswerBook Server

Verify the setup with a trial run. Log in to a system on the network and start the AnswerBook application. Either double-click on its icon in File Manager or enter:

```
$ /usr/openwin/bin/answerbook &
```

If OpenWindows has been installed somewhere other than `/usr/openwin` (which is the default installation directory), then use the command:

```
$ /openwin-install-path/openwin/bin/answerbook &
```

The `answerbook` command will search for a local `ab_cardcatalog` file and then for a remote `ab_cardcatalog` file (which it finds according to the `AB_CARDCATALOG` environment variable).

When the AnswerBook Navigator opens, click the Modify Library button and select AnswerBook titles from the scrolling list so they are included in the Navigator Contents.

Converting a bookinfo *File to an* ab_cardcatalog *File*

If an old (pre-Solaris 2.2) bookinfo file is on your network, convert it to ab_cardcatalog file. You can do this with the ab_admin command, as described in the next procedure.

▼ How to Convert a bookfinfo File To an ab_cardcatalog File

To convert an older bookinfo file to card-catalog format, use this ab_admin command:

```
# ab_admin -file $AB_CARDCATALOG -convert /opt/AB-name/bookinfo
```

In this command,

AB-name	Is the name of the installed AnswerBook package.

Using ab_admin *to Manage the* ab_cardcatalog *File*

One way to view and manage the ab_cardcatalog file is simply to view it with the cat command and edit it. However, the $AB_CARDCATALOG/ab_cardcatalog file can grow to be quite large, possibly making it unwieldy to manage in this way. For this reason, the ab_admin command is the recommended interface for managing the ab_cardcatalog file.

How to Manage the ab_cardcatalog File

Following is the syntax for using the ab_admin command. For more details, see the ab_admin(1) man page.

```
ab_admin [-verify AB-ID ] [-listpaths] [-match AB-ID] [-remove
AB-ID] [-merge AB-install-dir] [-file $AB_CARDCATALOG]
```

In this command,

AB-ID	Is the AnswerBook ID number as listed in the ab_cardcatalog file.
AB-install-dir	Is the AnswerBook installation directory. AnswerBook packages are typically installed in /opt.
$AB_CARDCATALOG	Is the environment variable that is set to a master ab_cardcatalog file that is shared on an AnswerBook server.

Example—Validating ab_cardcatalog *Entries*

This ab_admin command verifies the AnswerBook entries are valid in an ab_cardcatalog file.

```
% /bin/sh
$ for AB in ‘ab_admin -list -file $AB_CARDCATALOG‘
> do
> echo Verifying $AB
> ab_admin -verify $AB -file $AB_CARDCATALOG
> done
Verifying SUNWab_8_6
Verifying SUNWab_2_6
```

The ab_admin command will report any invalid entries.

Example—Finding an AnswerBook ID Number

The ab_admin command often takes the AnswerBook ID as an argument. If the ab_cardcatalog file has a lot of AnswerBook entries, you can use ab_admin with an AnswerBook ID to view information specific to that AnswerBook.

This ab_admin command shows the AnswerBook ID number for the SUNWabe AnswerBook installed in /opt.

```
% ab_admin -list -file /opt/SUNWabe/ab_cardcatalog
SUNWab_2_6
```

Example—Listing Card Catalogs

This ab_admin command list card catalogs and packages in the current environment:

```
# ab_admin -listpaths
/net/AB-server/export/ab_cardcatalog
/home/mickey/.ab_cardcatalog
```

Example—Listing AnswerBook Title and Paths

This ab_admin command lists a specific AnswerBook title and paths as listed in ab_cardcatalog file. In this example, ab_admin displays information about the AnswerBook corresponding to ID SUNWab_10_4:

```
% ab_admin -match SUNWab_10_4 -file $AB_CARDCATALOG
id=SUNWab_10_4
title= Solaris XGL 3.1 AnswerBook
tocpath=/net/AB-server/export/share/ab_cardcatalog
pspath=/net/AB-server/export/share/ab_cardcatalog
indexpath=/net/AB-server/export/share/ab_cardcatalog
```

Example—Merging New Card Catalog Data with the Master

This ab_admin command merges a specific AnswerBook ab_cardcatalog file with the site's master ab_cardcatalog file. (You would do this after installing a new AnswerBook. Before users on the network can access it, you need to merge its ab_cardcatalog with the site's master ab_cardcatalog).

```
# ab_admin -file $AB_CARDCATALOG -merge \
/opt/AB-name/ab_cardcatalog
```

Example—Removing Entries From ab_cardcatalog

This ab_admin command removes a specific AnswerBook entry (in this case, the AnswerBook with an ID of SUNWab_2_6) from the ab_cardcatalog file.

```
# ab_admin -file $AB_CARDCATALOG -remove SUNWab_2_6
```

▼ How to Determine If Display PostScript Is Available

To determine if your system or terminal is running Display PostScript, set the DISPLAY environment variable to the appropriate display and then use the xdpyinfo(1) command.

At the command line, enter:

```
$ xdpyinfo | grep -i dps
```

If Display PostScript is running you will see:

```
Adobe-DPS-Extension
DPSExtension
```

If Display PostScript is not running, no information will be displayed.

AnswerBook Troubleshooting 23

This chapter describes problems you may encounter when setting up AnswerBook and possible solutions for those problems. There are two sections: Specific AnswerBook Errors, which describes specific error messages you might encounter, and General AnswerBook Problems, which describes setup problems that might not result in a particular error message.

The following table shows common error messages and the page number where you can find causes and possible solutions for the related errors.

AnswerBook "AB-name" is not accessible. or *Can't add AnswerBook '<bs=SUNWab_AB-ID;vr=>' invalid AnswerBook: <bs=SUNWab_AB-ID;vr=>*	*page 376*
X Error of failed request: BadAccess (attempt to access private resource denied) or *X Error of failed request: BadMatch (invalid parameter attributes)*	*page 377*
Looking for locally installed AnswerBooks...	*page 377*

Specific AnswerBook Errors

```
AnswerBook "AB-name" is not accessible.
Verify that the card catalog entry for this
AnswerBook is correct.
```

```
Can't add AnswerBook '<bs=SUNWab_AB-ID;vr=>'
invalid AnswerBook: <bs=SUNWab_AB-ID;vr=>
Can't view document. View Document failed
Can't start new Viewer. Error executing link
```

Reason Error Occurred

These error messages indicate that the master `ab_cardcatalog` file has incorrect entries for this AnswerBook.

How to Fix the Problem

Examine the master `ab_cardcatalog` file. Verify that the `tocpath`, `pspath`, and `indexpath` entries for this AnswerBook are automounter path names (for example, they typically begin with `/net/`*hostname*).

Also, verify that the file systems that have the master `ab_cardcatalog` file and the installed AnswerBook packages are shared so that clients on the network can access them.

There may also be discrepancies between the local `~/.ab_library` file and the master `ab_cardcatalog` file. Examine these to make sure the `~/.ab_library` files list books available on the master AnswerBook server.

```
X Error of failed request:  BadAccess (attempt to access private resource denied)
   Major opcode of failed request:  88 (X_FreeColors)
   Serial number of failed request:  287
   Current serial number in output stream:  288
```

```
X Error of failed request:  BadMatch (invalid parameter attributes)
   Major opcode of failed request:  129 (Adobe-DPS-Extension)
   Minor opcode of failed request:  2 ()
   Serial number of failed request:  338
   Current serial number in output stream:  338
```

Reason Error Occurred

These problem occur when you try to open a specific AnswerBook from the AnswerBook navigator. The AnswerBook doesn't open. It reports "Can't start new Viewer. Could not launch new viewer."

This happens because other applications are already using desktop resources required for viewing an AnswerBook.

How to Fix the Problem

Close other applications on the desktop and try again to open the AnswerBook.

```
Looking for locally installed AnswerBooks...

Could not find any AnswerBook Card Catalog files in your environment.
The AnswerBook Navigator cannot access AnswerBooks
that are not listed in a Card Catalog file.
```

Reason Error Occurred

These error messages indicate that the AnswerBook startup software cannot locate the site's master `ab_cardcatalog` file.

How to Fix the Problem

Verify that the `AB_CARDCATALOG` environment variable is properly set on the system trying to access AnswerBook. (For information about distributing the `AB_CARDCATALOG` environment variable, see "The AB_CARDCATALOG Environment Variable" on page 340.)

General AnswerBook Problems

Problem	How to Fix the Problem
Users cannot print AnswerBooks successfully from the AnswerBook viewer.	The number of pages that can be printed from AnswerBook per user request varies with the capacity of the printer. Print the AnswerBook in smaller units, for example, print AnswerBook section by section.

Problem	How to Fix the Problem
The installation script prompts for user interaction, but ignores user responses. The package add or remove operation then fails.	There is a known problem with adding or removing some packages developed prior to Solaris 2.4. If you are prompted for user interaction and your responses are ignored, set the following environment variable and try the package add or remove operation again. `NONABI_SCRIPTS=TRUE`

Overview of Patch Administration 24

For the purpose of this discussion, patch administration involves installing or removing Solaris patches from a running Solaris system. It may also involve removing (called *backing out*) unwanted or faulty patches.

This is a list of the overview information in this chapter.

What Is a Patch

In its simplest form, you can think of a patch as a collection of files and directories that replace or update existing files and directories that are preventing proper execution of the software. The existing software is derived from a specified *package* format, which conforms to the Application Binary Interface. (For details about packages, see Chapter 17, "Overview of Software Administration.")

Tools For Managing Patches

There are two utilities for managing patches:

- `installpatch` – use to install directory-format patches to a Solaris system
- `backoutpatch` – use to remove patches installed on a Solaris system. This script restores the file system to its state before a patch was applied.

Detailed information about how to install and back out a patch is provided in the `README` file that comes with every patch.

Before installing patches, you might also want to know more about patches that have previously been installed. Table 24-1 shows commands that provide useful information about patches already installed on a system.

Table 24-1 Helpful Commands for Patch Administration

Command	**Function**
`showrev -p`	Shows all patches applied to a system.
`pkgparam` *pkgid* `PATCHLIST`	Shows all patches applied to the package identified by *pkgid*.
`pkgparam` *pkgid* `PATCH_INFO_`*patch-number*	Shows the installation date and name of the host from which the patch was applied. *pkgid* is the name of the package: for example, `SUNWadmap`.

Patch Distribution

All Sun customers can access security patches and other recommended patches via the World-Wide Web or anonymous `ftp`. Sun customers who have purchased a service contract can access an extended set of patches and a complete database of patch information. This information is available via the World-Wide Web, anonymous `ftp`, and it is regularly distributed on a CD (See Table 24-2.

Table 24-2 Customer Patch Access Information

If You Are ...	Then ...
A Sun Service customer	You have access to the SunSolve database of patches and patch information. These are available via the world-wide-web or anonymous ftp, as described in "Patch Access Via the World-Wide Web" on page 382 and "Patch Access Via ftp" on page 382. These patches are updated nightly. You also receive a patch CD every 6 to 8 weeks.
Not a Sun Service customer	You have access to a general set of security patches and other recommended patches. These are available via the world-wide-web or anonymous ftp, as described in "Patch Access Via the World-Wide Web" on page 382 and "Patch Access Via ftp" on page 382.

What You Need to Access Sun Patches

You can access Sun patches via the World-Wide Web or anonymous `ftp`. If you have purchased a Sun service contract, you will also be able to get patches from the patch CD that is regularly distributed.

To access patches on the world-wide web, you need a machine that is:

- Connected to the Internet
- Capable of running Web browsing software such as Mosaic or Netscape

To access patches via anonymous `ftp`, you need a machine that is:

- Connected to the Internet
- Capable of running the `ftp` program

Patch Access Via the World-Wide Web

To access patches via the world-wide web, use this uniform resource locator (URL):

`http://www.sun.com/`

After reaching the Sun home page, click on the Sales and Service button and navigate your way to the SunSolve patch database.

The patch database for publicly available patches are labeled “Public patch access.” The patch database for the comprehensive set of patches and patch information available to contract customers is labeled “Contract customer patch access.” You will be prompted for a password to access this contract customer database.

You can also access publicly available patches using this URL:

`http://sunsite.unc.edu/`

Patch Access Via `ftp`

To access patches via `ftp`, you can use `ftp` to connect to either the `sunsolve1.sun.com` (provided by Sun Service) or `sunsite.unc.edu` (maintained by the University of North Carolina). When `ftp` prompts you for a login, enter `anonymous` as the login name. Use your complete email address when prompted for a password. After you have connected, you can find publicly available patches in the `/pubs/patches` directory.

Note – To transfer patches, you will need to change the `ftp` transfer mode to binary. To do this, enter `bin` at the `ftp` prompt.

Patch Numbering

Patches are identified by unique alphanumeric strings, with the patch base code first, a hyphen, and a number that represents the patch revision number. For example, patch 101977-02 is a Solaris 2.4 patch to correct the `lockd` daemon.

What Happens When You Add a Patch

When you add a patch, the `installpatch` script copies files from the patch directory to a local system's disk. More specifically, `installpatch`:

- Determines the Solaris version number of the managing host and the target host.
- Updates the patch's `pkginfo` file with information about patches obsoleted by the patch being installed, other patches required by this patches, patches incompatible with this patch.

During the patch installation, `installpatch` keeps a log of the patch installation in `/tmp/log_`*patch-number*

The `installpatch` script will not install a patch under the following conditions:

- The package is not fully installed on the host.
- There is already an installed patch with the same base code and a higher version number.
- The patch is incompatible with another, already installed patch. (Each installed patch keeps this information in its `pkginfo` file.)

What Happens When You Remove a Patch

When you back out a patch, `backoutpatch` restores all files modified by that patch, unless:

- The patch was installed with `installpatch -d` (which instructs `installpatch` not to save copies of files being updated or replaced).
- The patch has been obsoleted by a later patch.

The `backoutpatch` script calls `pkgadd` to restore packages that were saved from the initial patch installation.

During the patch installation, `backoutpatch` keeps a log of the patch installation in `/tmp/log_`*patch-number*.

Part 6 — Managing Devices

This part provides instructions for managing devices in the Solaris 2.x environment.

Overview of Device Management 25

The chapter provides overview information about managing peripheral devices in the Solaris 2.x environment.

This is a list of overview information in this chapter.

This is a list of step-by-step instructions in this chapter.

For information about accessing devices, see Chapter 26, “Accessing Devices.”

Device management in the Solaris 2.x environment usually includes adding and removing peripheral devices from systems, possibly adding a third-party device driver to support a device, and displaying system configuration information.

Table 25-1 describes where to find step-by-step procedures for adding serial devices, such as printers and modems, and peripheral devices, such as a disk, CD-ROM, or tape drive to your system.

Table 25-1 Where to Find Instructions for Adding a Device

For Information On ...	See the Following
Adding a disk	Chapter 29, "SPARC: Adding a Disk" or Chapter 30, "x86: Adding a Disk"
Adding a CD-ROM or tape device	"How to Add a Peripheral Device" on page 391
Adding a modem	"Managing Terminals and Modems" in *System Administration Guide, Volume II*
Adding a printer	"Managing Printer Servers" in *System Administration Guide, Volume II*

About Device Drivers

A computer typically uses a wide range of peripheral and mass-storage devices. Your system, for example, probably has a SCSI disk drive, a keyboard and a mouse, and some kind of magnetic backup medium. Other commonly used devices include CD-ROM drives, printers and plotters, light pens, touch-sensitive screens, digitizers, and tablet-and-stylus pairs.

The Solaris software does not directly communicate with all these devices. Each type of device requires different data formats, protocols, and transmission rates.

A *device driver* is a low-level program that allows the operating system to communicate with a specific piece of hardware. The driver serves as the operating system's "interpreter" for that piece of hardware.

Automatic Configuration of Devices

The kernel, consisting of a small generic core with a platform-specific component and a set of modules, is configured automatically in the Solaris 2.x environment.

A kernel module is a hardware or software component that is used to perform a specific task on the system. An example of a *loadable* kernel module is a device driver that is loaded when the device is accessed.

The platform-independent kernel is `/kernel/genunix`. The platform-specific component is `/platform/‘uname -m‘/kernel/unix`.

The kernel modules are described in Table 25-2.

Table 25-2 Description of Kernel Modules

Location	**This Directory Contains ...**
`/platform/‘uname -m‘/kernel`	Platform-specific kernel components
`/kernel`	Kernel components common to all platforms that are needed for booting the system
`/usr/kernel`	Kernel components common to all platforms within a particular instruction set

The system determines what devices are attached to it at boot time. Then the kernel configures itself dynamically, loading needed modules into memory. At this time, device drivers are loaded when devices, such as disk and tape devices, are accessed for the first time. This process is called *autoconfiguration* because all kernel modules are loaded automatically when needed.

You can customize the way in which kernel modules are loaded by modifying the `/etc/system` file. See "Managing System Performance" in *System Administration Guide, Volume II*, for instructions on modifying this file.

Features and Benefits

The benefits of autoconfiguration are:

- Main memory is used more efficiently because modules are loaded when needed.
- There is no need to reconfigure the kernel when new devices are added to the system.
- Drivers can be loaded and tested without having to rebuild the kernel and reboot the system.

The autoconfiguration process is used by a system administrator when adding a new device (and driver) to the system. At this time, the administrator performs a reconfiguration boot so the system will recognize the new device.

What You Need For Unsupported Devices

Device drivers needed to support a wide range of standard devices are included in the Solaris 2.x environment. These drivers can be found in the `/kernel/drv` and `/platform/‘uname -m‘/kernel/drv` directories.

However, if you've purchased an unsupported device, the manufacturer should provide the software needed for the device to be properly installed, maintained, and administered.

At a minimum, this software includes a device driver and its associated configuration `.conf`) file. The `.conf` files reside in the `drv` directories. In addition, the device may be incompatible with Solaris 2.x utilities, and may require custom maintenance and administrative utilities.

Contact your device manufacturer for more information.

Adding a Peripheral Device to a System

Adding a new peripheral device usually involves:

- Shutting down the system
- Connecting the device to the system
- Rebooting the system

Use the procedure below to add the following devices to a system:

- CD-ROM
- Secondary disk drive
- Tape drive
- SBUS card

In some cases, you may have to add a third-party device driver to support the new device.

▼ How to Add a Peripheral Device

1. **Become root.**

2. **Follow steps 2 and 3 of "How to a Add a Device Driver" on page 393 if you need to add a device driver to support the device.**

3. **Create the `/reconfigure` file.**

```
# touch /reconfigure
```

The `/reconfigure` file will cause the Solaris software to check for the presence of any newly installed devices the next time you turn on or boot your system.

4. **Shut down the system.**

```
# shutdown -i0 -g30 -y
```

In this command,

`-i0`	Brings the system to the 0 init state, which is the appropriate state for turning the system power off for adding and removing devices.
`-g30`	Shuts the system down in 30 seconds. The default is 60 seconds.
`-y`	Continues the system shutdown without user intervention; otherwise, you are prompted to continue the shutdown process.

5. **Turn off power to the system after it is shut down.**

On SPARC Systems ...	On x86 Systems ...
It is safe to turn off power if the `ok` or `>` prompt is displayed.	It is safe to turn off power if the type `any key to continue prompt` is displayed.

Refer to the hardware installation guide that accompanies your system for the location of the power switch.

6. **Turn off power to all external devices.**
For location of power switches on any peripheral devices, refer to the hardware installation guides that accompany your peripheral devices.

7. **Install the peripheral device making sure the device you are adding has a different target number than the other devices on the system.**
You will often find a small switch located at the back of the disk for this purpose.

Refer to the hardware installation guide that accompanies the peripheral device for information on installing and connecting the device.

8. **Turn on the power to the system.**
   The system will boot to multiuser mode and the login prompt will be displayed.

Verification—Adding a Peripheral Device

Verify that the peripheral device has been added by attempting to access the device. See Chapter 26, “Accessing Devices,” for information on accessing the device.

▼ How to a Add a Device Driver

This procedure assumes that the device has already been added to the system. If not, see “How to Add a Peripheral Device” on page 391.

1. **Become root.**
2. **Place the tape, diskette, or CD-ROM into the drive.**
3. **Use the** `pkgadd` **command install the driver.**

```
# pkgadd -d device package-name
```

In this command,

`-d` *device*	Identifies the device pathname.
package-name	Identifies the package name that contains the device driver.

Verification—Adding a Device Driver

Verify that the package has been added correctly by using the `pkgchk` command. The system prompt returns with no response if the package is installed correctly.

```
# pkgchk packagename
#
```

Example—Adding a Device Driver

The following example installs and verifies a package called `XYZdrv`.

```
# pkgadd XYZdrv
(licensing messages displayed)
.
.
.
Installing XYZ Company driver as <XYZdrv>
.
.
.
Installation of <XYZdrv> was successful.
# pkgchk XYZdrv
```

Displaying Device Configuration Information

Three commands are used to display system and device configuration information:

- `prtconf` – Displays system configuration information, including total amount of memory and the device configuration as described by the system's device hierarchy.

  The output displayed by this command depends upon the type of system.

- `sysdef` – Displays device configuration information including system hardware, pseudo devices, loadable modules, and selected kernel parameters.
- `dmesg` – Displays system diagnostic messages as well as a list of devices attached to the system since the last reboot.

See "Device Naming Conventions" on page 404 for information on the device names used to identify devices on the system.

`driver not attached` *Message*

The following driver-related message may be displayed by the `prtconf` and `sysdef` commands:

```
device, instance #number (driver not attached)
```

This message does not always mean that a driver is unavailable for this device. It means that no driver is *currently* attached to the device instance because there is no device at this node or the device is not in use. Drivers are loaded automatically when the device is accessed and unloaded when the device is not in use.

Identifying a System's Devices

Use the output of `prtconf` and `sysdef` commands to identify which disk, tape, and CD-ROM devices are connected to the system. The output of these commands display the `driver not attached` messages next to the device instances. Since these devices are always being monitored by some system process, the `driver not attached` message is usually a good indication that there is no device at that device instance.

For example, the following `prtconf` output identifies a device at `instance #3` and `instance #6`, which is probably a disk device at target 3 and a CD-ROM device at target 6 of the first SCSI host adapter (`esp, instance #0`).

```
$ /usr/sbin/prtconf
.
.
.

esp, instance #0
            sd (driver not attached)
            st (driver not attached)
            sd, instance #0 (driver not attached)
            sd, instance #1 (driver not attached)
            sd, instance #2 (driver not attached)
            sd, instance #3
            sd, instance #4 (driver not attached)
            sd, instance #5 (driver not attached)
            sd, instance #6
.
.
.
```

The same device information can be gleaned from the `sysdef` output.

How to Display System Configuration Information

Use the `prtconf` command to display system configuration information.

```
$ /usr/sbin/prtconf
```

Use the `sysdef` command to display system configuration information including pseudo devices, loadable modules, and selected kernel parameters

```
$ /usr/sbin/sysdef
```

Examples—Displaying System Configuration Information

The following `prtconf` output is displayed on a SPARC system.

```
# prtconf
System Configuration:  Sun Microsystems  sun4c
Memory size: 32 Megabytes
System Peripherals (Software Nodes):
SUNW,Sun 4_50
    packages (driver not attached)
        disk-label (driver not attached)
        deblocker (driver not attached)
        obp-tftp (driver not attached)
    options, instance #0
    aliases (driver not attached)
    openprom (driver not attached)
    zs, instance #0
    zs, instance #1
    audio (driver not attached)
    eeprom (driver not attached)
    counter-timer (driver not attached)
    memory-error (driver not attached)
    interrupt-enable (driver not attached)
    auxiliary-io (driver not attached)
    sbus, instance #0
        dma, instance #0
        esp, instance #0
            sd (driver not attached)
            st (driver not attached)
            sd, instance #0 (driver not attached)
            sd, instance #1 (driver not attached)
            sd, instance #2 (driver not attached)
            sd, instance #3
            sd, instance #4 (driver not attached)
            sd, instance #5 (driver not attached)
            sd, instance #6
.
.
.
```

The following `sysdef` output is displayed from an x86 system.

```
# sysdef
*
* Hostid
*
  24b02304
*
* i86pc Configuration
*
*
* Devices
*
eisa, instance #0
    kd, instance #0
    eha, instance #0
    cmdk, instance #0
    cmdk, instance #1 (driver not attached)
    cmdk, instance #2 (driver not attached)
    cmdk, instance #3 (driver not attached)
    cmdk, instance #4 (driver not attached)
    cmdk, instance #5 (driver not attached)
    cmdk, instance #6
.
.
.
```

▼ How to Display Device Information

Display device information with the `dmesg` command.

```
$  /usr/sbin/dmesg
```

The `dmesg` output is displayed as messages on the system console and identifies which devices are connected to the system since the last reboot.

Examples—Displaying Device Information

The following `dmesg` output is displayed from a SPARC system.

```
# dmesg
May 31 10:32
SunOS Release 5.4 Version[UNIX(R) System V Release 4.0]
Copyright (c) 1983-1994, Sun Microsystems, Inc.
mem = 28672K (0x1c00000)
avail mem = 26701824
Ethernet address = 8:0:20:9:5:d9
root nexus = Sun 4_65
sbus0 at root: obio 0xf8000000
dma0 at sbus0: SBus slot 0 0x400000
esp0 at sbus0: SBus slot 0 0x800000 SBus level 3 sparc ipl 3
sd1 at esp0: target 1 lun 0
sd1 is /sbus@1,f8000000/esp@0,800000/sd@1,0
    <SUN1.3G cyl 1965 alt 2 hd 17 sec 80>
sd3 at esp0: target 3 lun 0
sd3 is /sbus@1,f8000000/esp@0,800000/sd@3,0
    <SUN0207 cyl 1254 alt 2 hd 9 sec 36>
root on /sbus@1,f8000000/esp@0,800000/sd@1,0:a fstype ufs
zs0 at root: obio 0xf1000000 sparc ipl 12
zs0 is /zs@1,f1000000
zs1 at root: obio 0xf0000000 sparc ipl 12
zs1 is /zs@1,f0000000
cgsix0 at sbus0: SBus slot 2 0x0 SBus level 5 sparc ipl 7
cgsix0 is /sbus@1,f8000000/cgsix@2,0
cgsix0: screen 1152x900, single buffered, 1M mappable, rev 2
le0 at sbus0: SBus slot 0 0xc00000 SBus level 4 sparc ipl 5
le0 is /sbus@1,f8000000/le@0,c00000
le1 at sbus0: SBus slot 1 0xc00000 SBus level 4 sparc ipl 5
le1 is /sbus@1,f8000000/le@1,c00000
dump on /dev/dsk/c0t1d0s1 size 33308K
pseudo-device: vol0
vol0 is /pseudo/vol@0
fd0 at root: obio 0xf7200000 sparc ipl 11
```

The following dmesg output is displayed from an x86 system.

```
# dmesg
May 31 10:30
SunOS Release 5.4 Version [UNIX(R) System V Release 4.0]
Copyright (c) 1983-1994, Sun Microsystems, Inc.
mem = 15992K (0xf9e000)
avail mem = 12070912
root nexus = i86pc
eisa0 at root
EISA-device: dpt6
Disk0:<Vendor ‘MAXTOR  ’ Product ‘LXT-535S        ’>
cmdk0 at dpt6 target 0 lun 0
cmdk0 is /eisa/dpt@5c88,0/cmdk@0,0
Disk6:<Vendor ‘SONY    ’ Product ‘CD-ROM CDU-8012 ’>
cmdk6 at dpt6 target 6 lun 0
cmdk6 is /eisa/dpt@5c88,0/cmdk@6,0
EISA-device: asy0
asy0 is /eisa/asy@3f8,0
Ethernet address = 0:0:c0:68:14:5d
SMC WD8003/WD8013 driver: type=WD8013W   addr=00 00 c0 68 14 5d
EISA-device: smc0
smc0 is /eisa/smc@0,c0000
dump on /dev/dsk/c0t0d0s1 size 32756K
NOTICE: GIO_KEYMAP type 0
NOTICE: PIO_KEYMAP type 0
NOTICE: INSTALLING new map of type USL FORMAT
NOTICE: IN i8042_acquire
NOTICE: out i8042_acquire
NOTICE: rv was 1
NOTICE: IN i8042_release
NOTICE: about to enable keyboard
NOTICE: out i8042_release
Nov 30 17:19:31 sendmail[171]: network daemon starting
```

Accessing Devices 26

This chapter provides information about how system administrators access the devices on their systems.

This is a list of overview information in this chapter.

For overview information about configuring devices, see Chapter 25, “Overview of Device Management.”

26

Accessing Devices

System administrators need to know how to specify device names when using commands to manage disks, file systems, and other devices. In most cases, system administrators use logical device names to represent devices connected to the system. Both logical and physical device names are represented on the system by logical and physical device files.

How Device Information Is Created

When a system is booted for the first time, a device hierarchy is created to represent all the devices connected to the system. The kernel uses the device hierarchy information to associate drivers with their appropriate devices, and provides a set of pointers to the drivers that perform specific operations. See *OpenBoot 3.x Command Reference Manual* for more information on device hierarchy information.

Device Naming Conventions

Devices are referenced in three ways in the Solaris 2.x environment.

- **Physical device name –** Represents the full device pathname in the device information hierarchy. Physical device names are displayed using the following commands:
  - `dmesg`
  - `format`
  - `sysdef`
  - `prtconf`

  Physical device files are found in the `/devices` directory.

- **Instance name –** Represents the kernel's abbreviation name for every possible device on the system. For example, `sd0` and `sd1` represent the instance names of two disk devices. Instance names are mapped in the `/etc/path_to_inst` file and are displayed using the following commands:
  - `dmesg`
  - `sysdef`
  - `prtconf`

- **Logical device name –** Used by system administrators with most file system commands to refer to devices. See Table 26-1 on page 407 for a list of file commands that use logical device names. Logical device files in the `/dev` directory are symbolically linked to physical device files in the `/devices` directory.

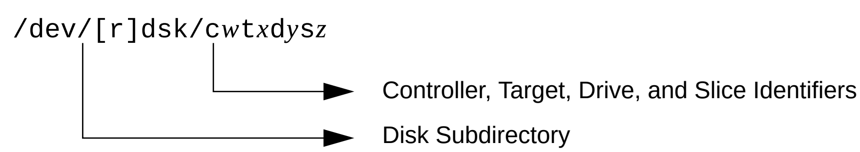
Logical Disk Device Names

Logical device names are used to access disk devices when you:

- Add a new disk to the system.
- Move a disk from one system to another.
- Access (or mount) a file system residing on a local disk.
- Back up a local file system.

Many administration commands take arguments that refer to a disk slice or file system.

Refer to a disk device by specifying the subdirectory to which it is symbolically linked (either `/dev/dsk` or `/dev/rdsk`), followed by a string identifying the particular controller, disk, and slice.



Specifying the Disk Subdirectory

Disk and file administration commands require the use of either a *raw* (or *character*) device interface, or a *block* device interface. The distinction is made by how data is read from the device.

Raw device interfaces transfer only small amounts of data at a time. Block device interfaces include a buffer from which large blocks of data are read at once.

Different commands require different interfaces.

- When a command requires the raw device interface, specify the `/dev/rdsk` subdirectory. (The “r” in `rdsk` stands for “raw.”)
- When a command requires the block device interface, specify the `/dev/dsk` subdirectory.
- When you’re not sure whether a command requires use of `/dev/dsk` or `/dev/rdsk`, check the reference man page for that command.

Table 26-1 shows which interface is required for a few commonly used disk and file system commands.

Table 26-1 Device Interface Type Required by Some Frequently Used Commands

Command	Interface Type	Example of Use
`df`	Block	`df /dev/dsk/c0t3d0s6`
`fsck`	Raw	`fsck -p /dev/rdsk/c0t0d0s0`
`mount`	Block	`mount /dev/dsk/c1t0d0s7 /export/home/ziggy`
`newfs`	Raw	`newfs /dev/rdsk/c0t0d1s1`
`prtvtoc`	Raw	`prtvtoc /dev/rdsk/c0t0d0s2`

Specifying the Slice

The string you use to identify a specific slice on a specific disk depends on the controller type, either direct or bus-oriented. Table 26-2 describes the different types of direct or bus-oriented controllers on different platforms.

Table 26-2 Controller Types

Direct controllers	Bus-Oriented Controllers
Xylogics (SPARC)	SCSI (SPARC/x86)
IDE (x86)	IPI (SPARC)

The conventions for both types of controllers are explained in the following subsections.

Note – Controller numbers are assigned automatically at system initialization. The numbers are strictly logical and imply no direct mapping to physical controllers.

SPARC: Disks With Direct Controllers

To specify a slice on a disk with a direct controller on a SPARC system, follow the naming convention shown in Figure 26-1.

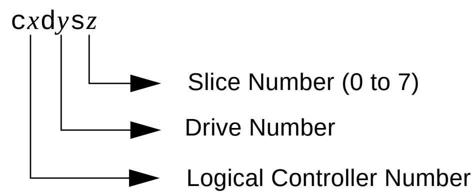


Figure 26-1 Naming Convention for Disks With Direct Controllers on SPARC Systems

To indicate the whole disk, specify slice 2 (`s2`).

If you have only one controller on your system, *x* will always be 0.

x86: Disks With Direct Controllers

To specify a slice on a disk with an IDE controller on an x86 system, follow the naming convention shown in Figure 26-2.

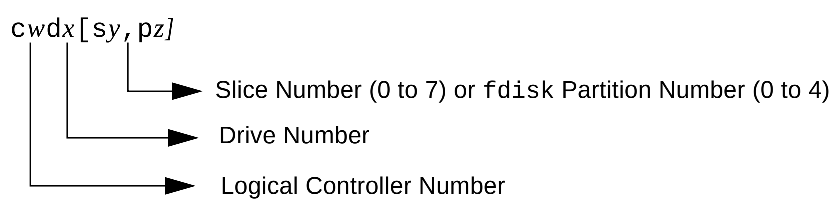


Figure 26-2 Naming Convention for Disks With IDE Controllers on x86 Systems

To indicate the entire Solaris `fdisk` partition, specify slice 2 (`s2`).

If you have only one controller on your system, *w* will always be 0.

SPARC: Disks With Bus-Oriented Controllers

To specify a slice on a disk with a bus-oriented controller (SCSI, for instance) on a SPARC system, follow the naming convention shown in Figure 26-3.

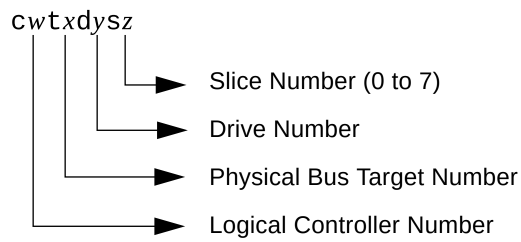


Figure 26-3 Naming Convention for Disks With Bus-Oriented Controllers on SPARC Systems

If you have only one controller on your system, *w* will always be 0.

For SCSI controllers, *x* is the target address as set by the switch on the back of the unit, and *y* is the logical unit number (LUN) of the drive attached to the target. If the disk has an embedded controller, *y* is usually 0.

To indicate the whole disk, specify slice 2 (`s2`).

x86: Disks With SCSI Controllers

To specify a slice on a disk with a SCSI controller on an x86 system, follow the naming convention shown in Figure 26-4.

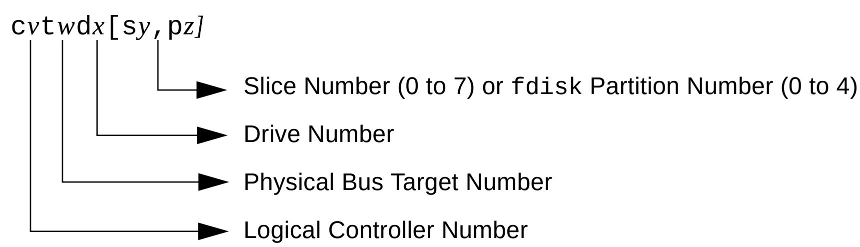


Figure 26-4 Naming Convention for Disks With SCSI Controllers on x86 Systems

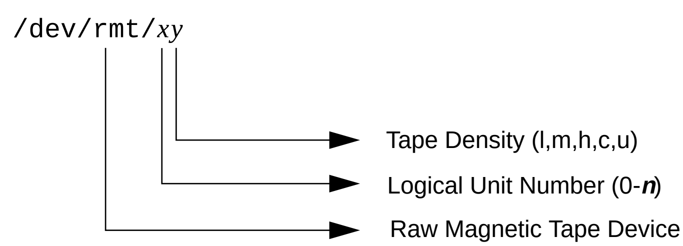
If you have only one controller on your system, *v* will always be 0.

For SCSI controllers, *w* is the target address as set by the switch on the back of the unit, and *x* is the logical unit number (LUN) of the drive attached to the target. If the disk has an embedded controller, *x* is usually 0.

To indicate the entire Solaris `fdisk` partition, specify slice 2 (`s2`).

Logical Tape Device Names

Logical tape device files are found in the `/dev/rmt/*` directory as symbolic links from the `/devices` directory.



The first tape device connected to the system is 0 (`/dev/rmt/0`), which may be one of the following types: QIC-11, QIC-24, QIC-150, or Exabyte. Tape density values (`l`, `m`, `h`, `c`, and `u`) are described in Chapter 46, “Managing Tape Drives.”

Logical CD-ROM Device Names

The logical device name that represents the first CD-ROM device on a system is `/dev/dsk/c0t6d0s0`.

Since CD-ROMS are managed by Volume Management, the logical CD-ROM device name is usually not used unless you want to mount the CD manually. See Chapter 11, “Guidelines for Using CDs and Diskettes,” for information on accessing your CD-ROM device.

Part 7 — Managing Disks

This part provides instructions for managing disks in the Solaris environment.

Overview of Disk Management 27

This overview chapter provides conceptual information about Solaris disk slices and introduces the `format` utility.

This is a list of the overview information in this chapter.

For instructions on how to add a disk drive to your system, see Chapter 29, "SPARC: Adding a Disk," or Chapter 30, "x86: Adding a Disk."

27

Introduction

Managing disks in the Solaris 2.x environment usually involves setting up the system and running the Solaris installation program to create the appropriate disk slices and install the operating system. Occasionally, you may need to use the `format` utility to add a new disk drive or replace a defective one.

Disk Terminology

Before you can effectively use the information in this section, you should be familiar with basic disk architecture. In particular, you should be familiar with the following terms:

- Track
- Cylinder
- Sector
- Disk controller
- Disk label
- Device drivers

If you are unfamiliar with these terms, refer to the glossary (for a brief definition) or product information from the disk's manufacturer.

About Disk Slices

Files stored on a disk are contained in file systems. Each file system on a disk is assigned to a *slice*—a group of cylinders set aside for use by that file system. Each disk slice appears to the operating system (and to the system administrator) as though it were a separate disk drive.

See "Managing File Systems" in *System Administration Guide, Volume I* for information about file systems.

Note – Slices are sometimes referred to as partitions. This book uses *slice*, but, certain interfaces, such as the `format` utility, refer to slices as partitions.

When setting up slices, remember these rules:

- Each disk slice holds only one file system.
- No file system can span multiple slices.

Slices are set up slightly differently on SPARC and x86 platforms. Table 27-1 summarizes the differences:

Table 27-1 Slice Differences on Platforms

SPARC	**x86**
Whole disk is devoted to Solaris environment	Disk is divided into `fdisk` partitions, one per operating environment
Disk is divided into eight slices, numbered 0-7	The Solaris `fdisk` partition is divided into 10 slices, numbered 0-9

SPARC: Disk Slices

On SPARC systems, Solaris defines eight disk slices and assigns to each a conventional use. These slices are numbered 0 through 7. Table 27-2 summarizes the contents of the eight Solaris slices on a SPARC system.

Table 27-2 SPARC: Customary Disk Slices

Slice	File System	Usually Found on Client or Server Systems?	Purpose
0	`root`	both	Holds files and directories that make up the operating system.
1	`swap`	both	Provides virtual memory, or *swap space*. Swap space is used when running programs are too large to fit in a computer's memory. The Solaris operating environment then "swaps" programs from memory to the disk and back as needed.
2	–	both	Refers to the entire disk, by convention. It is defined automatically by Sun's `format` and the Solaris installation programs. The size of this slice should not be changed.
3	`/export`	server only	Holds alternative versions of the operating system. These alternative versions are required by client systems whose architectures differ from that of the server. Clients with the same architecture type as the server obtain executables from the `/usr` file system, usually slice 6.
4	`/export/swap`	server only	Provides virtual memory space for client systems.
5	`/opt`	both	Holds application software added to a system. If a slice is not allocated for this file system during installation, the `/opt` directory is put in slice 0.
6	`/usr`	both	Holds operating system commands—also known as *executables*—designed to be run by users. This slice also holds documentation, system programs (`init` and `syslogd`, for example) and library routines.
7	`/home` or `/export/home`	both	Holds files created by users.

x86: Disk Slices

On x86 systems, disks are divided into `fdisk` partitions. An `fdisk` partition is a section of the disk reserved for a particular operating environment, such as Solaris.

Solaris places ten slices, numbered 0-9, on the Solaris `fdisk` partition on a disk in an x86 system, as shown in Table 27-3.

Table 27-3 x86: Customary Disk Slices

Slice	File System	Usually Found on Client or Server Systems?	Purpose
0	`root`	both	Holds the files and directories that make up the operating system.
1	swap	both	Provides virtual memory, or *swap space*. Swap space is used when running programs are too large to fit in a computer's memory. The Solaris operating environment then "swaps" programs from memory to the disk and back as needed.
2	—	both	Refers to the entire disk, by convention. It is defined automatically by Sun's `format` and the Solaris installation programs. The size of this slice should not be changed.
3	`/export`	server only	Holds alternative versions of the operating system. These alternative versions are required by client systems whose architectures differ from that of the server.
4	`/export/swap`	server only	Provides virtual memory space for the client systems.
5	`/opt`	both	Holds application software added to a system. If a slice is not allocated for this file system during installation, the `/opt` directory is put in slice 0.
6	`/usr`	both	Holds operating system commands—also known as *executables*— that are run by users. This slice also holds documentation, system programs (`init` and `syslogd`, for example) and library routines.
7	`/home` or `/export/home`	both	Holds files created by users.
8	—	both	Contains information necessary for Solaris to boot from the hard disk. It resides at the beginning of the Solaris partition (although the slice number itself does not indicate this), and is known as the boot slice.
9	—	both	Provides an area reserved for alternate disk blocks. Slice 9 is known as the alternate sector slice.

Slice Arrangements on Multiple Disks

Although a single disk that is large enough can hold all slices and their corresponding file systems, two or more disks are often used to hold a system's slices and file systems.

Note – A slice cannot be split between two or more disks. However, multiple swap slices on separate disks are allowed.

For instance, a single disk might hold the root (`/`) file system, a swap area, and the `/usr` file system, while a separate disk is provided for the `/export/home` file system and other file systems containing user data.

In a multiple disk arrangement, the disk containing the operating system software and swap space (that is, the disk holding the root (`/`) or `/usr` file systems or the slice for swap space) is called the *system disk*. Disks other than the system disk are called *secondary disks* or *non-system disks*.

Locating a system's file systems on multiple disks allows you to modify file systems and slices on the secondary disks without having to shut down the system or reload operating system software.

Having more than one disk also increases input-output (I/O) volume. By distributing disk load across multiple disks, you can avoid I/O bottlenecks.

Determining Which Slices to Use

When you set up a disk's file systems, you choose not only the size of each slice, but also which slices to use. Your decisions about these matters depend on the configuration of the system to which the disk is attached and the software you want to install on the disk.

There are five system configurations:

- Servers
- Diskless clients
- Standalone systems
- Dataless clients
- AutoClient systems

Each system configuration requires the use of different slices. Table 27-4 lists these requirements.

Table 27-4 System Configurations and Slice Requirements

Slice	Servers	Diskless Clients	Standalone Systems	Dataless Clients	AutoClient Systems
0	`root`	(on server)	`root`	`root`	`root`
1	`swap`	(on server)	`swap`	`swap`	`swap`
2	—	—	—	—	—
3	`/export`	—	—	—	—
4	`/export/swap`	—	—	—	—
5	`/opt`	(on server)	`/opt`	`/opt`	(on server)
6	`/usr`	(on server)	`/usr`	(on server)	(on server)
7	`/export/home`	(on server)	`/home`	(on server)	(on server)

See Chapter 4, “Adding and Maintaining Server and Client Support,” for more information about system configurations.

Note – The Solaris installation program provides slice size recommendations based on the software you select for installation.

The `format` Utility

Read the following information if you want to see a conceptual view of the `format` utility and what it is used for before proceeding to the “how-to” or reference sections.

Definition

The `format` utility is a system administration tool used to prepare hard disk drives for use on your Solaris system. The `format` utility cannot be used on diskette drives, CD-ROM drives, or tape drives.

Features and Benefits

Table 27-5 shows the features and associated benefits that the `format` utility provides.

Table 27-5 Features and Benefits of the `format` Utility

Feature	**Benefit**
Searches your system for all attached disk drives	Reports • Target location • Disk geometry • Whether the disk is formatted • If the disk has mounted partitions
Retrieves disk labels	Used in repair operations
Repairs defective sectors	Allows administrators to repair disk drives with recoverable errors instead of sending the drive back to the manufacturer
Formats and analyzes a disk	Creates sectors on the disk and verifies each sector
Partitions a disk	Divides a disk so individual file systems can be created on separate slices
Labels a disk	Writes disk name and configuration information to the disk for future retrieval (usually for repair operations)

All of the options of the `format` utility are fully described in Chapter 31, “The format Utility.”

When to Use the `format` *Utility*

Disk drives are partitioned and labeled by the Solaris installation program as part of installing the Solaris release. You may need to use the `format` utility when:

- Displaying slice information
- Dividing a disk into slices
- Adding a disk drive to an existing system
- Formatting a disk drive
- Repairing a disk drive

The main reason a system administrator uses the `format` utility is to divide a disk into disk slices. These steps are covered in Chapter 29, "SPARC: Adding a Disk" and Chapter 30, "x86: Adding a Disk."

See Table 27-6 for guidelines on using the `format` utility.

Guidelines for Using the `format` *Utility*

Table 27-6 The `format` utility Guidelines

Use the `format` Utility To ...	**Considerations ...**	**Where to Go ...**
Format a disk	• Any existing data will be destroyed when a disk is reformatted. • The need for formatting a disk drive has dropped as more and more manufacturers ship their disk drives formatted and partitioned. You may not need to use the `format` utility when adding a disk drive to an existing system. • If a disk has been relocated and is displaying a lot of disk errors, you can attempt to reformat it, which will automatically remap any bad sectors.	"How to Format a Disk" on page 435
Replace a system disk	• Data from the damaged system disk must be restored from a backup medium; otherwise the system will have to be reinstalled by using the installation program.	Chapter 29, "SPARC: Adding a Disk" or Chapter 30, "x86: Adding a Disk" or if the system must be reinstalled, *SPARC: Installing Solaris Software* or *x86: Installing Solaris Software*
Divide a disk into slices	• Any existing data will be destroyed when a disk with existing slices is repartitioned and relabeled. • Existing data must be copied to backup media before the disk is repartitioned and restored after the disk is relabeled.	Chapter 29, "SPARC: Adding a Disk" or Chapter 30, "x86: Adding a Disk"
Add a secondary disk to an existing system	• Any existing data must be restored from backup media if the secondary disk is reformatted or repartitioned.	Chapter 29, "SPARC: Adding a Disk" or Chapter 30, "x86: Adding a Disk"
Repair a disk drive	• Some customer sites prefer to replace rather than repair defective drives. If your site has a repair contract with the disk drive manufacturer, you may not need to use the `format` utility to repair disk drives. • Repairing a disk drive usually means that a bad sector is added to a defect list. New controllers remap bad sectors automatically with no system interruption. • If the system has an older controller, you may need to remap a bad sector and restore any lost data.	Chapter 31, "The format Utility"

Formatting a Disk

In most cases, disks are formatted by the manufacturer or reseller and do not need to be reformatted when you install the drive. To determine whether or not a disk is formatted, use the `format` utility. See "How to Determine If a Disk Is Formatted" on page 434 for more information.

If you determine that a disk is not formatted, use the `format` utility to format the disk.

Formatting a disk accomplishes two steps:

- Preparing disk media for use
- Compiling a list of disk defects based on a surface analysis

---

Caution – Formatting is a destructive process—it overwrites data on the disk. For this reason, disks are usually formatted only by the manufacturer or reseller. If you think disk defects are causing recurring problems, you can use the `format` utility to do a surface analysis, but be careful to use only the commands that do not destroy data. See "How to Format a Disk" on page 435 for details.

A small percentage of total disk space available for data is used to store defect and formatting information. This percentage varies according to disk geometry, and decreases as the disk ages and develops more defects.

Formatting may take anywhere from a few minutes to several hours, depending on the type and size of the disk.

About Disk Labels

A special area of every disk is set aside for storing information about the disk's controller, geometry, and slices. That information is called the disk's *label*. Another term used to described the disk label is the VTOC (Volume Table of Contents). To *label* a disk means to write slice information onto the disk. You usually label a disk after changing its slices.

If you fail to label a disk after creating slices, the slices will be unavailable because the operating system has no way of "knowing" about the slices.

Partition Table

An important part of the disk label is the *partition table* which identifies a disk's slices, the slice boundaries (in cylinders), and total size of the slices. A disk's partition table can be displayed using the `format` utility. Table 27-7 describes partition table terminology.

Table 27-7 Partition Table Terminology

Partition Term	Value	Description
Number	`0-7`	Partition or (slice number). Valid numbers are 0-7.

Tag	`0=UNASSIGNED`
	`1=BOOT`
	`2=ROOT`
	`3=SWAP`
	`4=USR`
	`5=BACKUP`
	`7=VAR`
	`8=HOME`
		A numeric value that usually describes the file system mounted on this partition.

Flags	`wm`	The partition is writable and mountable.
	`wu`	The partition is writable and unmountable. This is the default state of partitions dedicated for swap areas. However, the mount command does not check the "not mountable" flag.
	`rm`	The partition is read only and mountable.

Partition flags and tag are assigned by convention and require no maintenance.

See "How to Display Disk Slice Information" on page 438 or "How to Examine a Disk Label" on page 444 for more information on displaying the partition table.

Examples—Partition Tables

The following partition table example is displayed from a 535-Mbyte disk using the `format` utility:

```
Part      Tag    Flag     Cylinders        Size            Blocks
 0       root    wm        0 -  292       80.12MB    (293/0/0)   164080
 1       swap    wu      293 -  410       32.27MB    (118/0/0)    66080
 2     backup    wm        0 - 1865      510.23MB    (1866/0/0) 1044960
 3 unassigned    wm        0                0        (0/0/0)          0
 4 unassigned    wm        0                0        (0/0/0)          0
 5       home    wm      411 - 1311      246.37MB    (901/0/0)   504560
 6        usr    wm     1312 - 1718      111.29MB    (407/0/0)   227920
 7 unassigned    wm     1719 - 1865       40.20MB    (147/0/0)    82320
```

The partition table contains the following information:

Column Name	Description
`Part`	Partition (or slice number). See Table 27-7 for a description of this column.
`Tag`	Partition tag. See Table 27-7 for a description of this column.
`Flags`	Partition flag. See Table 27-7 for a description of this column.
`Cylinders`	The starting and ending cylinder number for the slice.
`Size`	The slice size in Mbytes.
`Blocks`	The total number of cylinders and the total number of sectors per slice in the far right column.

The following example displays a disk label using the `prtvtoc` command.

```
# prtvtoc /dev/rdsk/c0t3d0s0
* /dev/rdsk/c0t3d0s0 partition map
*
* Dimensions:❶
*     512 bytes/sector
*      80 sectors/track
*       9 tracks/cylinder
*     720 sectors/cylinder
*    2500 cylinders
*    1151 accessible cylinders
*
* Flags:❷
*   1: unmountable
*  10: read-only
*
*                          First     Sector    Last
* Partition  Tag  Flags    Sector     Count    Sector  Mount Directory ❸
       0      2    00          0     66240     66239   /
       1      3    01      66240    131760    197999
       2      5    00          0    828720    828719
       3      0    00     198000     66240    264239   /export
       4      0    00     264240     66240    330479   /export/swap
       5      0    00     330480     72000    402479   /opt
       6      4    00     402480    385200    787679   /usr
       7      8    00     787680     41040    828719   /export/home
#
```

The disk label includes the following information:

❶ **Dimensions –** This section describes the physical dimensions of the disk drive.

❷ **Flags –** This section describes the flags listed in the partition table section. See Table 27-7 for a description of partition flags.

❸ **Partition (or Slice) Table –** This section contains the following information:

Column Name	Description
`Partition`	Partition (or slice number). See Table 27-7 for a description of this column.
`Tag`	Partition tag. See Table 27-7 for a description of this column.
`Flags`	Partition flag. See Table 27-7 for a description of this column.
`First Sector`	The first sector of the slice.
`Sector Count`	The total number of sectors in the slice.
`Last Sector`	The last sector number in the slice.
`Mount Directory`	The last mount point directory for the file system.

Dividing a Disk Into Slices

The `format` utility is most often used by system administrators to divide a disk into slices. The steps are:

- Determining which slices are needed
- Determining the size of each slice
- Using the `format` utility to divide the disk into slices
- Labeling the disk with new slice information
- Creating the file system for each slice

The easiest way to divide a disk into slices is to use the `modify` command from the `partition` menu. The `modify` command allows you to create slices by specifying the size of each slice in megabytes without having to keep track of starting cylinder boundaries. It also keeps tracks of any disk space remainder in the “free hog” slice.

Using the Free Hog Slice

When you use the `format` utility to change the size of one or more disk slices, you designate a temporary slice that will expand and shrink to accommodate the resizing operations.

This temporary slice donates, or “frees,” space when you expand a slice, and receives, or “hogs,” the discarded space when you shrink a slice. For this reason, the donor slice is sometimes called the *free hog*.

The donor slice exists only during installation or when you run the `format` utility. There is no permanent donor slice during day-to-day, normal operations.

See “SPARC: How to Create Disk Slices and Label a Disk” on page 467 or “x86: How to Create Disk Slices and Label a Disk” on page 487 for information on using the free hog slice.

Administering Disks 28

This chapter contains disk administration procedures. Many of the procedures described in this chapter are optional if you are already familiar with how disks are managed on systems running the Solaris 2.x release.

This is a list of the step-by-step instructions in this chapter.

For overview information about disk management, see Chapter 27, “Overview of Disk Management.”

28

Administering Disks

Table 28-1 Task Map: Administering Disks

Activity	Description	For Instructions, Go To	
Identify the Disks on a System	If you are not sure of the types of disks on a system, use the `format` utility to identify the disk types.	▼ How to Identify the Disks on a System	page 431
Format the Disk	Determine whether a disk is already formatted by using the `format` utility.	▼ How to Determine If a Disk Is Formatted	page 434
	In most cases, disks are already formatted. Use the `format` utility if you need to format a disk.	▼ How to Format a Disk	page 435
Display Slice Information	Display slice information by using the `format` utility.	▼ How to Display Disk Slice Information	page 438
Label the Disk	Create the disk label by using the `format` utility.	▼ How to Label a Disk	page 442
	Examine the disk label by using the `prtvtoc` command.	▼ How to Examine a Disk Label	page 444
Create a `format.dat` Entry	Create a `format.dat` entry to support a third-party disk.	▼ How to Create a format.dat Entry	*page 451
Repair a Defective Disk Sector	Identify a defective disk sector by using the `format` utility.	▼ How to Identify a Defective Sector by Using Surface Analysis	page 456
	Fix a defective disk sector by using the `format` utility.	▼ How to Repair a Defective Sector	page 458

Identifying Disks on a System

Use the format utility to discover the types of disks that are connected to a system. You can also use the format utility to verify that a disk is known to the system.

See Chapter 31, “The format Utility” for information on using the format utility.

▼ How to Identify the Disks on a System

1. **Become root.**
2. **Identify the disks that are recognized on the system with the format utility.**

```
# format
```

The format utility displays a list of disks that is recognizes under AVAILABLE DISK SELECTIONS.

Verification—Identifying the Disks on a System

The disks recognized by the format utility are displayed under the AVAILABLE DISK SELECTIONS heading.

Examples—Identifying the Disks on a System

The following format output is from a system with two disks.

```
# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
       0. c0t1d0 <SUN0535 cyl 1866 alt 2 hd 7 sec 80>
          /sbus@1,f8000000/esp@0,800000/sd@1,0
       1. c0t2d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
          /sbus@1,f8000000/esp@0,800000/sd@2,0
Specify disk (enter its number):
```

The `format` output associates a disk's physical and logical device name to the disk's marketing name which appears in brackets <>. This is an easy way to identify which logical device names represent the disks connected to your system. See Chapter 26, "Accessing Devices," for a description of logical and physical device names.

The following example uses a wildcard to display the disks connected to a second controller.

```
# format /dev/rdsk/c2*
AVAILABLE DISK SELECTIONS:
      0. /dev/rdsk/c2t0d0s0 <SUN2.1G cyl 2733 alt 2 hd 19 sec 80>
          /io-unit@f,e0200000/sbi@0,0/QLGC,isp@2,10000/sd@0,0
      1. /dev/rdsk/c2t1d0s0 <SUN2.1G cyl 2733 alt 2 hd 19 sec 80>
          /io-unit@f,e0200000/sbi@0,0/QLGC,isp@2,10000/sd@1,0
      2. /dev/rdsk/c2t2d0s0 <SUN2.1G cyl 2733 alt 2 hd 19 sec 80>
          /io-unit@f,e0200000/sbi@0,0/QLGC,isp@2,10000/sd@2,0
      3. /dev/rdsk/c2t3d0s0 <SUN2.1G cyl 2733 alt 2 hd 19 sec 80>
          /io-unit@f,e0200000/sbi@0,0/QLGC,isp@2,10000/sd@3,0
      4. /dev/rdsk/c2t5d0s0 <SUN2.1G cyl 2733 alt 2 hd 19 sec 80>
          /io-unit@f,e0200000/sbi@0,0/QLGC,isp@2,10000/sd@5,0
Specify disk (enter its number):
```

The following example identifies the disks on a SPARC system.

```
# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
       0. c0t1d0 <SUN0535 cyl 1866 alt 2 hd 7 sec 80>
          /sbus@1,f8000000/esp@0,800000/sd@1,0
Specify disk (enter its number):
```

The `format` output identifies that disk `0` (target `1`) is connected to the first SCSI host adapter (`esp@0...`), which is connected to the first SBus device (`sbus@1...`). The output also associates both the physical and logical device name to the disk's marketing name, `SUN0535`.

The following example identifies the disks on an x86 system.

```
# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
       0. c0t0d0 <DEFAULT cyl 507 alt 2 hd 64 sec 32>
          /eisa/dpt@5c88,0/cmdk@0,0
       1. c0t3d0 <DEFAULT cyl 1852 alt 2 hd 15 sec 74>
          /eisa/dpt@5c88,0/cmdk@3,0
Specify disk (enter its number):
```

The `format` output identifies that disk `0`, target `0` (`cmdk@0,0`) is connected to the first DPT host adapter (`dpt@5...`), which is connected to the EISA device (`eisa`). The `format` output on an x86 system does not identify disks by their marketing names.

Where to Go From Here

Check the following table if the `format` utility did not recognize the disk.

If the Disk ...	Then ...
Is newly added and you didn't perform a reconfiguration boot	Go to Chapter 29, "SPARC: Adding a Disk," or Chapter 30, "x86: Adding a Disk."
Is a third-party disk	Go to "Creating a format.dat Entry" on page 450.
Label was corrupted by a system problem, such as a power failure	Go to "How to Label a Disk" on page 442.
Is not properly connected to the system	Connect the disk to the system using your disk hardware documentation.

Formatting a Disk

Disks are formatted by the manufacturer or reseller and usually do not need to be reformatted when you install the drive.

A disk must be formatted before ...

- You can write data to it. However, most disks are already formatted.
- You can use the Solaris installation program to install the system.

Caution – Formatting is a destructive process—it overwrites data on the disk. For this reason, disks are usually formatted only by the manufacturer or reseller. If you think disk defects are causing recurring problems, you can use the `format` utility to do a surface analysis, but be careful to use only the commands that do not destroy data.

▼ How to Determine If a Disk Is Formatted

1. **Become root.**
2. **Enter the** `format` **utility.**

```
# format
```

3. **Enter the number of the disk that you want to check from the list displayed on your screen.**

```
Specify disk (enter its number): 0
```

Verification—Determining If a Disk Is Formatted

If the disk you chose is formatted, you will see the following message:

```
[disk formatted]
```

Example—Determining If a Disk Is Formatted

The following example shows that disk `c0t3d0` is formatted.

```
# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
       0. c0t3d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
          /sbus@1,f8000000/esp@0,800000/sd@3,0
Specify disk (enter its number): 0
selecting c0t3d0
[disk formatted]
```

▼ How to Format a Disk

1. **Become root.**
2. **Enter the** `format` **utility.**

```
# format
```

3. **Enter the number of the disk that you want to format from the list displayed on your screen.**

```
Specify disk (enter its number): 0
```

Warning – Do not select the system disk. Formatting your system disk deletes your operating system and any data that you may have on this disk.

4. **To begin formatting the disk, enter** `format` **at the** `format>` **prompt. Confirm the command by typing** `y`**.**

```
format> format
Ready to format. Formatting cannot be interrupted
and takes 10 minutes (estimated). Continue? y
```

Verification—Formatting a Disk

To verify that the disk is formatted, go to "How to Determine If a Disk Is Formatted" on page 434.

Example—Formatting a Disk

The following example formats the disk `c0t2d0`.

```
# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
       0. c0t1d0 <SUN0535 cyl 1866 alt 2 hd 7 sec 80>
          /sbus@1,f8000000/esp@0,800000/sd@1,0
       1. c0t2d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
          /sbus@1,f8000000/esp@0,800000/sd@2,0
Specify disk (enter its number):1
Selecting c0t2d0
[disk unformatted]
format> format
Ready to format. Formatting cannot be interrupted
and takes 23 minutes (estimated). Continue? y
Beginning format. The current time is Tue May  3 17:44:42 1994
Formatting ...
done
Verifying media ...
pass 0 - pattern = 0xc6dec6de
2035/12/18
pass 1 - pattern = 0x6db6db6d
2035/12/18
total of 0 defective blocks repaired.
format>
```

Displaying Disk Slices

You can use the `format` utility to check whether or not a disk has the appropriate disk slices. If you determine that a disk does not contain the slices you want to use, use the `format` utility to re-create them and label the disk. See "SPARC: How to Create Disk Slices and Label a Disk" on page 467 or "x86: How to Create Disk Slices and Label a Disk" on page 487 for information on creating disk slices.

Note – The `format` utility uses the term *partition* in place of *slice.*

▼ How to Display Disk Slice Information

1. **Become root.**
2. **Enter the** `format` **utility.**

```
# format
```

3. **Identify the disk for which you want to display slice information by selecting a disk listed under** `AVAILABLE DISK SELECTIONS`**.**

```
Specify disk (enter its number):1
```

4. **Enter the partition menu by typing** `partition` **at the** `format>` **prompt.**

```
format> partition
```

5. **Display the slice information for the current disk drive by typing** `print` **at the** `partition>` **prompt.**

```
partition> print
```

6. **Exit the** `format` **utility by typing** `q` **at the** `partition>` **prompt and typing** `q` **at the** `format>` **prompt.**

```
partition> q
format> q
#
```

Verification—Displaying Disk Slice Information

If slices have been assigned to the disk, the screen output shows specific slice tags and sizes. If the screen output shows that no slice sizes are assigned, the disk probably does not have slices.

Examples—Displaying Disk Slice Information

The following example displays slice information for disk `/dev/rdsk/c0t3d0`.

```
# format
Searching for disks...done
Specify disk (enter its number):1
Selecting c0t3d0
format> partition
partition> print
Current partition table (original):
Total disk cylinders available: 1866 + 2 (reserved cylinders)
Part      Tag    Flag     Cylinders        Size            Blocks
  0       root    wm       0 -  292       80.12MB    (293/0/0)   164080
  1       swap    wu     293 -  410       32.27MB    (118/0/0)    66080
  2     backup    wm       0 - 1865      510.23MB    (1866/0/0) 1044960
  3 unassigned    wm       0               0         (0/0/0)          0
  4 unassigned    wm       0               0         (0/0/0)          0
  5       home    wm     411 - 1311      246.37MB    (901/0/0)   504560
  6        usr    wm    1312 - 1718      111.29MB    (407/0/0)   227920
  7 unassigned    wm    1719 - 1865       40.20MB    (147/0/0)    82320
partition> q
format> q
#
```

See Chapter 27, “Overview of Disk Management,” for a detailed description of the slice information displayed in these examples.

The following example displays the slice information on disk `/dev/rdsk/c0t0d0`.

```
# format
Searching for disks...done
Specify disk (enter its number): 0
selecting c0t0d0
[disk formatted]
format> partition
partition> print
Current partition table (original):
Total disk cylinders available: 1479 + 2 (reserved cylinders)

Part      Tag    Flag     Cylinders        Size            Blocks
  0       root    wm       1 -  400      108.01MB    (400/0/0)  221200
  1       swap    wu     401 -  556       42.12MB    (156/0/0)   86268
  2     backup    wu       0 - 1479      399.63MB    (1480/0/0) 818440
  3 unassigned    wm       0                0        (0/0/0)         0
  4 unassigned    wm       0                0        (0/0/0)         0
  5 unassigned    wm     557 -  734       48.06MB    (178/0/0)   98434
  6        usr    wm     735 - 1401      180.10MB    (667/0/0)  368851
  7       home    wm    1402 - 1476       20.25MB    (75/0/0)    41475
  8       boot    wu       0 -    0        0.27MB    (1/0/0)       553
  9 unassigned    wm       0                0        (0/0/0)         0
partition> q
format> q
```

Creating and Examining a Disk Label

Labeling a disk is usually done during system installation or when you are creating new disk slices. You may need to relabel a disk if the disk label is corrupted (for example, from a power failure).

The `format` utility will attempt to automatically configure any unlabeled SCSI disk. If `format` is able to automatically configure an unlabeled disk, it will display a message like the following:

```
    c1t0d0: configured with capacity of 404.65MB
```

▼ How to Label a Disk

1. **Become root.**
2. **Enter the `format` utility.**

```
# format
```

3. **Enter the number of the disk that you want to label from the list displayed on your screen.**

```
Specify disk (enter its number):1
```

4. **Use the table below to determine how to label the disk.**

If the Disk Is Unlabeled and Was Successfully Configured ...	If the Disk Was Labeled and You Want to Change the Type, or Format Was Not Able to Automatically Configure the Disk ...
Format will ask you if you want to label the disk. Go to step 5 to label the disk.	You must specify the disk type. Go to steps 6-7 to set the disk type and label the disk.

5. **Label the disk by typing** `yes` **at the** `Label it now?` **prompt.**

```
Disk not labeled. Label it now? yes
```

The disk is now labeled. Go to step 9 to exit the `format` utility.

6. **Enter** `type` **at the** `format>` **prompt.**

```
format> type
```

Format displays the Available Drive Types menu.

7. **Select a disk type from the list of possible disk types.**

```
Specify disk type (enter its number)[8]: 8
```

8. **Label the disk. If the disk is not labeled, the following message is displayed.**

```
Disk not labeled. Label it now? yes
```

Otherwise you are prompted with this message:

```
Ready to label disk, continue? yes
```

9. **Exit the** `format` **utility by typing** `q` **at the** `format>` **prompt.**

```
partition> q
format> q
#
```

Verification—Labeling a Disk

Use the `verify` command from the `format` main menu to verify the disk label.

Example—Labeling a Disk

The following example automatically configures and labels a 424-Mbyte disk.

```
# format
   c1t0d0: configured with capacity of 404.65MB

AVAILABLE DISK SELECTIONS:
       0. c0t3d0 <SUN0424 cyl 1151 alt 2 hd 9 sec 80>
          /sbus@1,f8000000/esp@0,800000/sd@3,0
       1. c1t0d0 <SUN0424 cyl 1151 alt 2 hd 9 sec 80>
          /sbus@1,f8000000/QLGC,isp@1,10000/sd@0,0
Specify disk (enter its number): 1
Disk not labeled.  Label it now?  yes
format> q
#
```

▼ How to Examine a Disk Label

Examine disk label information by using the `prtvtoc(1M)` command. See Chapter 27, "Overview of Disk Management," for a detailed description of the disk label and the information displayed by the `prtvtoc` command.

1. **Become root.**
2. **Display the disk label information by using the `prtvtoc` command.**

```
# prtvtoc /dev/rdsk/device-name
```

In this command,

device-name	Is the raw disk device you want to examine.

Verification—Examining a Disk Label

The disk label can be verified using the `verify` command from the `format` main menu.

Example—Examining a Disk Label

The following example shows the disk label information for disk `/dev/rdsk/c0t0d0s0`.

```
# prtvtoc /dev/rdsk/c0t0d0s0
* c0t0d0s0 partition map
*
* Dimensions:
*     512 bytes/sector
*      36 sectors/track
*       9 tracks/cylinder
*     324 sectors/cylinder
*    1272 cylinders
*    1254 accessible cylinders
*
* Flags:
*   1: unmountable
*  10: read-only
*
*                          First     Sector    Last
* Partition  Tag  Flags    Sector    Count     Sector  Mount Directory
       0      2    00          0     37260     37259   /
       1      3    01      37260     77760    115019
       2      5    00          0    406296    406295
       6      4    00     115020    283824    398843   /usr
       7      6    00     398844      7452    406295   /export/home
#
```

28

Recovering a Corrupted Disk Label

Sometimes a power or system failure will cause a disk's label to become unrecognizable. This doesn't always mean that the slice information or the disk's data will have be to re-created or restored.

The first step to recovering a corrupted disk label is to label the disk with the correct geometry and disk type information. This can be done through the normal disk labeling method, either automatic configuration or manual disk type specification.

If format recognizes the disk type, the next step is to search for a backup label to label the disk. Labeling the disk with the backup label will label the disk with the correct partitioning information, the disk type, and disk geometry.

▼ How to Recover a Corrupted Disk Label

1. **Boot the system to single-user mode. If necessary, boot the system from a local CD-ROM or the network in single-user mode to access the disk.**
   See Chapter 8, "Booting a SPARC System" or Chapter 9, "Booting an x86 System" for information on booting the system.

2. **Use the `format` utility to relabel the disk.**

```
# format
```

   At this point, `format` attempts to automatically configure any unlabeled SCSI disk. If `format` is able to configure the unlabeled and corrupted disk, it will display:

   `c`*n*`t`*n*`d`*n*`: configured with capacity of `*nnn*`MB`

   The `format` utility then displays the list of disks on the system.

3. **Enter the number of the disk that you need to recover from the list displayed on your screen.**

```
Specify disk (enter its number): 1
```

4. **Use the table below to determine how to label the disk.**

If the Disk was Successfully Configured ...	If the Disk was not Successfully Configured ...
Follow steps 5 and 6. Then go to step 12.	Follow steps 7-11. Then go to step 12.

5. **Search for the backup label by using the** `verify` **command.**

```
format> verify
Warning: Could not read primary label.
Warning: Check the current partitioning and 'label' the disk or use the
'backup' command.

Backup label contents:
Volume name = <        >
ascii name  = <SUN0424 cyl 1151 alt 2 hd 9 sec 80>
pcyl        = 2500
ncyl        = 1151
acyl        =    2
nhead       =    9
nsect       =   80
Part      Tag    Flag     Cylinders        Size            Blocks
  0       root    wm       0 -   91       32.34MB    (92/0/0)    66240
  1       swap    wu      92 -  183       32.34MB    (92/0/0)    66240
  2     backup    wu       0 - 1150      404.65MB    (1151/0/0) 828720
  3 unassigned    wm       0                0        (0/0/0)         0
  4 unassigned    wm       0                0        (0/0/0)         0
  5 unassigned    wm       0                0        (0/0/0)         0
  6        usr    wm     184 - 1150      339.96MB    (967/0/0)  696240
  7 unassigned    wm       0                0        (0/0/0)         0
```

6. **If** `format` **was able to find a backup label and the backup label contents appear satisfactory, use the** `backup` **command to label the disk with the backup label.**

```
format> backup
Disk has a primary label, still continue? y

Searching for backup labels...found.
Restoring primary label
```

The disk label has been recovered. Go to step 12.

7. **If** `format` **was not able to automatically configure the disk, specify the disk type using the** `type` **command.**

```
format> type
```

Format will display the Available Drives Type menu.

8. **Select** `0` **to automatically configure the disk, or select a disk type from the list of possible disk types.**

```
Specify disk type (enter its number)[8]: 8
```

9. **If the disk was successfully configured, reply with** `no` **when** `format` **asks if you want to label the disk.**

```
Disk not labeled.  Label it now?  no
```

10. **Use the `verify` command to search for backup labels.**

```
format> verify
Warning: Could not read primary label.
Warning: Check the current partitioning and 'label' the disk
or use the 'backup' command.
.
.
.
```

11. **If format was able to find a backup label and the backup label contents appear satisfactory, use the `backup` command to label the disk with the backup label.**

```
format> backup
Disk has a primary label, still continue? y

Searching for backup labels...found.
Restoring primary label
```

The disk label has been recovered.

12. **Exit the `format` utility by typing `q`.**

```
format> q
```

13. **Verify the file systems on the recovered disk by using the `fsck` command.**
See Chapter 39, “Checking File System Integrity,” for information about using the `fsck` command.

Adding a Third-Party Disk

The Solaris environment supports many third-party disks. However, you may need to supply either a device driver, a `format.dat` entry, or both of these.

If the third-party disk was designed to work with standard SunOS operating system-compatible device drivers, creating an appropriate `format.dat` entry should be enough to allow the disk to be recognized by the `format` utility. In other cases, you'll need to load a third-party device driver to support the disk.

Note – Sun cannot guarantee that its `format` utility will work properly with all third-party disk drivers. If the disk driver is not compatible with the Solaris `format` utility, the disk drive vendor should supply you with a custom format program.

This section discusses what to do if some of this software support is missing. Typically, this occurs when you invoke the `format` utility and find that the disk type is not recognized.

Supply the missing software as described in this section, and then refer to the appropriate configuration procedure for system disks or secondary disks in "SPARC: Adding a System or Secondary Disk" on page 463, or "Adding a System or Secondary Disk" on page 479.

Creating a `format.dat` *Entry*

Unrecognized disks cannot be formatted without precise information about the disk's geometry and operating parameters. This information is supplied in the `/etc/format.dat` file.

Note – SCSI-2 drives do not require a `format.dat` entry. Starting in Solaris 2.3, the `format` utility automatically configures the SCSI-2 drives if the drives are powered on during a reconfiguration boot. See "How to Automatically Configure a SCSI Disk Drive" on page 453 for step-by-step instructions on configuring a SCSI disk drive automatically.

If your disk was not recognized, use a text editor to create an entry in `format.dat` for the disk. You'll need to gather all the pertinent technical specifications about the disk and its controller before you start. This information should have been provided with the disk. If not, contact the disk manufacturer or your supplier. See Chapter 31, “The format Utility” for more information on adding an entry to the `/etc/format.dat` file.

▼ How to Create a `format.dat` Entry

1. **Become root.**
2. **Make a copy of the** `/etc/format.dat` **file.**

```
# cp /etc/format.dat /etc/format.dat.gen
```

3. **Modify the** `/etc/format.dat` **file to include an entry for the third-party disk using the** `format.dat` **information described in Chapter 31, “The format Utility.”**
   Use the disk’s hardware product documentation to gather the required information.

Automatically Configuring SCSI Disk Drives

In Solaris 2.3 and subsequent releases, the `format` utility automatically configures SCSI disk drives even if that specific type of drive is not listed in the `/etc/format.dat` file. This feature enables you to format, slice, and label any disk driver compliant with SCSI-2 specification for disk device mode sense pages.

Use the following steps to configure a SCSI drive via autoconfiguration:

- Shut down the system
- Attach the SCSI disk drive to the system and verify it is attached correctly
- Turn on the disk drive
- Perform a reconfiguration boot
- Use the `format` utility to automatically configure the SCSI disk drive

After the reconfiguration boot, invoke the `format` utility. The `format` utility will attempt to configure the disk and, if successful, alert the user that the disk was configured. See "How to Automatically Configure a SCSI Disk Drive" on page 453 for step-by-step instructions on configuring a SCSI disk drive automatically.

Here's the default slice rules that `format` uses to create the partition table.

Table 28-2 SCSI Disk Slice Rules

Disk size	`Root`	`Swap`
0 - 180 Mbytes	16 Mbytes	16 Mbytes
180 Mbytes - 280 Mbytes	16 Mbytes	32 Mbytes
280 Mbytes - 380 Mbytes	24 Mbytes	32 Mbytes
380 Mbytes - 600 Mbytes	32 Mbytes	32 Mbytes
600 Mbytes - 1.0 Gbytes	32 Mbytes	64 Mbytes
1.0 Gbytes - 2.0 Gbytes	64 Mbytes	128 Mbytes
2.0 Gbytes -	128 Mbytes	128 Mbytes

In all cases, slice 6 (`/usr`) gets the remainder of the space on the disk.

Here's an example of a format-generated partition table for a 1.3-Gbyte SCSI disk drive.

```
Part      Tag    Flag     Cylinders        Size       Blocks
0         root    wm       0 -   96       64.41MB    (97/0/0)
1         swap    wu      97 -  289      128.16MB    (193/0/0)
2       backup    wu       0 - 1964        1.27GB    (1965/0/0)
6          usr    wm     290 - 1964        1.09GB    (1675/0/0)
```

See Chapter 31, "The format Utility," for more information about using SCSI automatic configuration.

▼ How to Automatically Configure a SCSI Disk Drive

1. **Become root.**
2. **Create the /reconfigure file that will be read when the system is booted.**

```
# touch /reconfigure
```

3. **Shut down the system.**

```
# shutdown -i0 -g30 -y
```

In this command,

-i0	Brings the system down to init state 0 (zero), the power-down state.
-g30	Notifies logged-in users that they have *n* seconds before the system begins to shut down.
-y	Specifies the command should run without user intervention.

The ok or > prompt is displayed after the operating environment is shut down.

4. **Turn off power to the system and all external peripheral devices.**

5. **Make sure the disk you are adding has a different target number than the other devices on the system.**
   You will often find a small switch located at the back of the disk for this purpose.

6. **Connect the disk to the system and check the physical connections.**
   Refer to the disk's hardware installation guide for installation details.

7. **Turn on the power to all external peripherals.**

8. **Turn on the power to the system.**
   The system will boot and display the login prompt.

9. **Login as root, invoke the `format` utility, and select the disk to be configured automatically.**

```
# format
Searching for disks...done

c1t0d0: configured with capacity of 404.65MB

AVAILABLE DISK SELECTIONS:
       0. c0t3d0 <SUN0424 cyl 1151 alt 2 hd 9 sec 80>
          /sbus@1,f8000000/esp@0,800000/sd@3,0
       1. c1t0d0 <SUN0424 cyl 1151 alt 2 hd 9 sec 80>
          /sbus@1,f8000000/QLGC,isp@1,10000/sd@0,0
Specify disk (enter its number): 1
```

10. **Reply `yes` to the prompt to label the disk.**
    Replying `yes` will cause the disk label to be generated and written to the disk by the autoconfiguration feature.

```
Disk not labeled. Label it now? yes
```

11. **Verify the disk label with the `verify` command.**

```
format> verify
```

12. **Exit the `format` utility.**

```
format> quit
```

Repairing a Defective Sector

If a disk on your system has a defective sector, you can repair it by using the instructions in the following procedures. You may become aware of defective sectors when you:

- Run surface analysis on a disk.

  See “The analyze Menu” on page 500 for more information on the analysis functionality of `format`.

  The defective area reported while your system is running may not be accurate. Since the system does disk operations many sectors at a time, it is often hard to pinpoint exactly which sector caused a given error. Use “How to Identify a Defective Sector by Using Surface Analysis” on page 456 to find the exact sector(s).

- Get multiple error messages from the disk driver concerning a particular portion of the disk while your system is running.

  Messages related to disk errors look like the following:

```
WARNING: /io-unit@f,e0200000/sbi@0,0/QLGC,isp@1,10000/sd@3,0
(sd33):
    Error for command 'read' Error Level: Retryable
    Requested Block 126, Error Block: 179
    Sense Key: Media Error
    Vendor 'SEAGATE':
    ASC = 0x11 (unrecovered read error), ASCQ = 0x0, FRU = 0x0
```

  The above console message indicates that block 179 may be bad. Relocate the bad block by using the `format` utility’s `repair` command or use the `analyze` command with the `repair` option enabled.

▼ How to Identify a Defective Sector by Using Surface Analysis

1. **Become root.**
2. **Unmount the file system in the slice that contains the defective sector.**
   See `mount(1M)` for more information.

```
# umount /dev/dsk/device-name
```

3. **Enter the `format` utility by typing `format`.**

```
# format
```

4. **Select the affected disk.**

```
Specify disk (enter its number):1
selecting c0t2d0:
[disk formatted]
Warning: Current Disk has mounted partitions.
```

5. **Enter the analyze menu by typing `analyze` at the `format>` prompt.**

```
format> analyze
```

6. **Set up the analysis parameters for the search step. Use the parameters shown here:**

```
analyze> setup
Analyze entire disk [no]? n
Enter starting block number [0, 0/0/0]: 12330
Enter ending block number [584159, 1216/9/47]: 12360
Loop continuously [no]? y
Repair defective blocks [yes]? n
Stop after first error [no]? n
Use random bit patterns [no]? n
Enter number of blocks per transfer [31, 0/0/31]: 1
Verify media after formatting [yes]? y
Enable extended messages [no]? n
Restore defect list [yes]? y
Create defect label [yes]? y
```

7. **Use the** `read` **command to find the defect.**

```
analyze> read
Ready to analyze (won't harm SunOS). This takes a long time,
but is interruptible with Control-C. Continue? y
        pass 0
   25/7/24
        pass 1
Block 12354  (18/4/18), Corrected media error (hard data ecc)
   25/7/24
^C
Total of 1 defective blocks repaired.
```

How to Repair a Defective Sector

1. **Become root.**
2. **Enter the `format` utility and select the disk that contains the defective sector.**

```
# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
       0. c0t2d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd@2,0
       1. c0t3d0 <SUN0535 cyl 1866 alt 2 hd 7 sec 80>
/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd@3,0
Specify disk (enter its number): 1
selecting c0t3d0
[disk formatted]
format>
```

3. **Enter the `repair` command at the `format>` prompt.**

```
format> repair
```

4. **Enter the defective block number.**
   If you are unsure of the format used to identify the defective sector, see "How to Identify a Defective Sector by Using Surface Analysis" on page 456 for more information.

```
 Enter absolute block number of defect: 12354
   Ready to repair defect, continue? y
   Repairing block 12354 (18/4/18)...ok.
format>
```

Tips and Tricks

Use the following tips to help you manage disks more efficiently.

Debugging `format` Sessions

Invoke `format -M` to enable extended and diagnostic messages for using the `format` utility with SCSI devices only.

In this example, the series of numbers below `Inquiry:` represent the hexadecimal value of the inquiry data displayed to the right of the numbers.

```
# format -M
Searching for disks...done
AVAILABLE DISK SELECTIONS:
       0. c0t3d0 <SUN0535 cyl 1866 alt 2 hd 7 sec 80>
/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/
sd@3,0
Specify disk (enter its number): 0
selecting c0t3d0
[disk formatted]
format> inquiry
Inquiry:
00 00 02 02 27 00 00 12 43 4f 4e 4e 45 52 20 20    ....'...CONNER
43 50 33 30 35 34 30 20 20 53 55 4e 30 35 33 35    CP30540  SUN0535
42 30 42 42 39 33 30 38 46 39 30                       B0BB9308F90
Vendor:   CONNER
Product:  CP30540  SUN0535
Revision: B0BB
format>
```

Label Multiple Disks by Using the prtvtoc and fmthard Commands

Use the prtvtoc and fmthard commands to label multiple disks with the same disk geometry.

Use this for loop in a script to copy a disk label from one disk and replicate it on multiple disks.

```
# for i in x y z
> do
> prtvtoc /dev/rdsk/cntndnsn | fmthard -s - /dev/rdsk/cnt${i}d0s2
> done
```

Example—Labeling Multiple Disks

In this example, the disk label from c2t0d0s0 is copied to four other disks.

```
# for i in 1 2 3 5
> do
> prtvtoc /dev/rdsk/c2t0d0s0 | fmthard -s - /dev/rdsk/c2t${i}d0s2
> done
fmthard:  New volume table of contents now in place.
fmthard:  New volume table of contents now in place.
fmthard:  New volume table of contents now in place.
fmthard:  New volume table of contents now in place.
#
```

SPARC: Adding a Disk 29

This chapter provides the procedures for adding a disk to a SPARC system.

This is a list of the step-by-step instructions in this chapter.

For overview information about disk management, see Chapter 27, “Overview of Disk Management.”

29

SPARC: About System and Secondary Disks

A system disk contains the root (/) or /usr file systems, or both. If the disk containing either of these file system becomes damaged, you have two ways to recover:

- You can reinstall the entire Solaris environment
- Or, you can replace the system disk and restore your file systems from a backup medium.

A secondary disk doesn't contain the root (/) and /usr file systems. It usually contains space for user files. You can add a secondary disk to a system for more disk space or you can replace a damaged secondary disk. If you replace a secondary disk on a system, you can restore the old disk's data on the new disk.

SPARC: Adding a System or Secondary Disk

Table 29-1 SPARC: Adding a System or Secondary Disk

Activity	Description	For Instructions, Go To	
Connect the Disk and Boot	**System Disk** Connect the new disk and boot from a local or remote Solaris 2.x CD. **Secondary Disk** Connect the new disk and perform a reconfiguration boot, so the system will recognize the disk.	▼ SPARC: How to Connect a System Disk and Boot ▼ SPARC: How to Connect a Secondary Disk and Boot	page 464 page 465
Create Slices and Label the Disk	Create disk slices and label the disk if it has not already been done by the disk manufacturer.	▼ SPARC: How to Create Disk Slices and Label a Disk	page 467
Create File Systems	Create UFS file systems on the disk slices with the `newfs` command. You must create the root (`/`) or `/usr` file system (or both) for a system disk.	▼ SPARC: How to Create File Systems	page 474
Restore File Systems	Restore the root (`/`) or `/usr` file system (or both) on the system disk. If necessary, restore file systems on the secondary disk.	▼ SPARC: How to Restore File Systems	page 475
Install Boot Block	**System Disk Only** Install the boot block on the `root` (`/`) file system, so the system can boot.	▼ SPARC: How to Install a Boot Block on a System Disk	page 476

▼ SPARC: How to Connect a System Disk and Boot

This procedure assumes that the system is shut down.

1. **Disconnect the damaged system disk from the system.**
2. **Make sure the disk you are adding has a different target number than the other devices on the system.**
   You will often find a small switch located at the back of the disk for this purpose.
3. **Connect the replacement system disk to the system and check the physical connections.**
   Refer to the disk's hardware installation guide for installation details.
4. **Follow the instructions in the table below depending on whether you are booting from a local or remote Solaris 2.x CD.**

If You Are Booting From ...	Then ...
A Solaris 2.x CD from a local CD-ROM drive	1. Make sure the CD is in the CD-ROM drive. 2. Boot from the CD to single-user mode: `ok` **`boot cdrom -s`**
A Solaris 2.x CD from a CD-ROM drive over the network	Boot from the net to single-user mode: `ok` **`boot net -s`**

After a few minutes, the root prompt (#) is displayed.

Where to Go From Here

After you boot the system, you can create slices and a disk label on the disk. Go to "SPARC: How to Create Disk Slices and Label a Disk" on page 467.

▼ SPARC: How to Connect a Secondary Disk and Boot

1. **Become root.**
2. **If the disk type is unsupported by the Solaris software, add the device driver for the disk by following the instructions included with the hardware.**
   If necessary, see "How to Create a format.dat Entry" on page 451 for information on creating a `format.dat` entry for the disk.
3. **Create the `/reconfigure` file that will be read when the system is booted.**

```
# touch /reconfigure
```

   The `/reconfigure` file will cause the SunOS software to check for the presence of any newly installed peripheral devices when you power on or boot your system later.

4. **Shut down the system.**

```
# shutdown -i0 -g30 -y
```

   In this command,

`-i0`	Brings the system down to init state 0 (zero), the power-down state.
`-g30`	Notifies logged-in users that they have *n* seconds before the system begins to shut down.
`-y`	Specifies the command should run without user intervention.

   The `ok` or `>` prompt is displayed after the operating environment is shut down.

5. **Turn off power to the system and all external peripheral devices.**

6. **Make sure the disk you are adding has a different target number than the other devices on the system.**
   You will often find a small switch located at the back of the disk for this purpose.

7. **Connect the disk to the system and check the physical connections.**
   Refer to the disk's hardware installation guide for installation details.

8. **Turn on the power to all external peripherals.**

9. **Turn on the power to the system.**
   The system will boot and display the login prompt.

Where to Go From Here

After you boot the system, you can create slices and a disk label on the disk. Go to "SPARC: How to Create Disk Slices and Label a Disk" on page 467.

SPARC: How to Create Disk Slices and Label a Disk

1. **Become root.**

2. **Start the `format` utility.**

```
# format
```

A list of available disks is displayed.

3. **Enter the number of the disk that you want to repartition from the list displayed on your screen.**

```
Specify disk (enter its number): disk-number
```

At this prompt,

disk-number	Is the number of the disk that you want to repartition.

4. **Go into the `partition` menu (which lets you set up the slices).**

```
format> partition
```

5. **Display the current partition (slice) table.**

```
partition> print
```

6. **Start the modification process.**

```
partition> modify
```

7. **Set the disk to all free hog.**

```
Choose base (enter number) [0]? 1
```

See "Using the Free Hog Slice" on page 428 for more information about the free hog slice.

8. **Create a new partition table by answering** `yes` **when prompted to continue.**

```
Do you wish to continue creating a new partition
table based on above table[yes]? yes
```

9. **Identify the free hog partition (slice) and the sizes of the slices when prompted.**

When adding a system disk, you must set up slices for:

- root (slice 0), swap (slice 1) and/or
- `/usr` (slice 6)

After you identify the slices, the new partition table is displayed.

10. **Make the displayed partition table the current partition table by answering** `yes` **when asked.**

```
Okay to make this the current partition table[yes]? yes
```

If you don't want the current partition table and you want to change it, answer `no` and go to Step 6.

11. **Name the partition table.**

```
Enter table name (remember quotes): "partition-name"
```

At this prompt,

partition-name	Is the name for the new partition table.

12. **Label the disk with the new partition table when you have finished allocating slices on the new disk.**

```
Ready to label disk, continue? yes
```

13. **Quit the** `partition` **menu.**

```
partition> quit
```

14. **Quit the** `format` **menu.**

```
format> quit
```

Example—Creating Disk Slices and Labeling a System Disk

The following example uses the `format` utility to divide a 200-Mbyte disk into two slices: one for the root (`/`) file system and one for the swap area.

```
# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
       0. c0t2d0 <SUN0207 cyl 1254 alt 2 hd 9 sec 36>
          /sbus@1,f8000000/esp@0,800000/sd@2,0
       1. c0t3d0 <SUN0424 cyl 1151 alt 2 hd 9 sec 80>
          /sbus@1,f8000000/esp@0,800000/sd@3,0
Specify disk (enter its number): 0
selecting c0t2d0
[disk formatted]
format> partition
partition> print
partition> modify
Select partitioning base:
    0. Current partition table (original)
    1. All Free Hog
Choose base (enter number) [0]? 1

Part      Tag    Flag     Cylinders        Size       Blocks
  0       root    wm       0               0         (0/0/0)
  1       swap    wu       0               0         (0/0/0)
  2     backup    wu       0 - 1253      198.39MB    (1254/0/0)
  3 unassigned    wm       0               0         (0/0/0)
  4 unassigned    wm       0               0         (0/0/0)
  5 unassigned    wm       0               0         (0/0/0)
  6        usr    wm       0               0         (0/0/0)
  7 unassigned    wm       0               0         (0/0/0)
```

Example—Creating Disk Slices and Labeling a System Disk (Continued)

```
Do you wish to continue creating a new partition
table based on above table[yes]? yes
Free Hog partition[6]? 5
Enter size of partition ‘0’ [0b, 0c, 0.00mb]: 100mb
Enter size of partition ‘1’ [0b, 0c, 0.00mb]: 98mb
Enter size of partition ‘3’ [0b, 0c, 0.00mb]:
Enter size of partition ‘4’ [0b, 0c, 0.00mb]:
Enter size of partition ‘6’ [0b, 0c, 0.00mb]:
Enter size of partition ‘7’ [0b, 0c, 0.00mb]:

Part      Tag    Flag     Cylinders        Size       Blocks
  0       root    wm       0 -  632      100.14MB    (633/0/0)
  1       swap    wu     633 - 1233       95.08MB    (601/0/0)
  2     backup    wu       0 - 1253      198.39MB    (1254/0/0)
  3 unassigned    wm       0               0         (0/0/0)
  4 unassigned    wm       0               0         (0/0/0)
  5 unassigned    wm    1234 - 1253        3.16MB    (20/0/0)
  6        usr    wm       0               0         (0/0/0)
  7 unassigned    wm       0               0         (0/0/0)

Okay to make this the current partition table[yes]? yes
Enter table name (remember quotes): "disk2"
Ready to label disk, continue? yes

partition> quit
format> quit
```

Example—Creating Disk Slices and Labeling a Secondary Disk

The following example uses the `format` utility to divide a 200-Mbyte disk into one slice for the `/export/home1` file system.

```
# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
       0. c0t2d0 <SUN0207 cyl 1254 alt 2 hd 9 sec 36>
          /sbus@1,f8000000/esp@0,800000/sd@2,0
       1. c0t3d0 <SUN0424 cyl 1151 alt 2 hd 9 sec 80>
          /sbus@1,f8000000/esp@0,800000/sd@3,0
Specify disk (enter its number): 0
selecting c0t2d0
[disk formatted]
format> partition
partition> print
partition> modify
Select partitioning base:
    0. Current partition table (original)
    1. All Free Hog
Choose base (enter number) [0]? 1

Part      Tag    Flag     Cylinders        Size       Blocks
  0       root    wm       0               0        (0/0/0)
  1       swap    wu       0               0        (0/0/0)
  2     backup    wu       0 - 1253      198.39MB    (1254/0/0)
  3 unassigned    wm       0               0        (0/0/0)
  4 unassigned    wm       0               0        (0/0/0)
  5 unassigned    wm       0               0        (0/0/0)
  6        usr    wm       0               0        (0/0/0)
  7 unassigned    wm       0               0        (0/0/0)
Do you wish to continue creating a new partition
table based on above table[yes]? yes
Free Hog partition[6]? 5
Enter size of partition ‘0’ [0b, 0c, 0.00mb]:
Enter size of partition ‘1’ [0b, 0c, 0.00mb]:
Enter size of partition ‘3’ [0b, 0c, 0.00mb]:
Enter size of partition ‘4’ [0b, 0c, 0.00mb]: 198mb
Enter size of partition ‘6’ [0b, 0c, 0.00mb]:
Enter size of partition ‘7’ [0b, 0c, 0.00mb]:
```

Example—Creating Disk Slices and Labeling a Secondary Disk (Continued)

```
Part      Tag    Flag     Cylinders        Size       Blocks
  0       root    wm       0                0         (0/0/0)
  1       swap    wu       0                0         (0/0/0)
  2     backup    wu       0 - 1253      198.39MB    (1254/0/0)
  3 unassigned    wm       0                0         (0/0/0)
  4 unassigned    wm       0 - 1251      198.07MB    (1252/0/0)
  5 unassigned    wm    1252 - 1253        0.32MB    (2/0/0)
  6        usr    wm       0                0         (0/0/0)
  7 unassigned    wm       0                0         (0/0/0)

Okay to make this the current partition table[yes]? yes
Enter table name (remember quotes): "home1"

Ready to label disk, continue? yes

partition> quit
format> quit
#
```

Where to Go From Here

After you create disk slices and label the disk, you can create file systems on the disk. Go to “SPARC: How to Create File Systems” on page 474.

SPARC: How to Create File Systems

1. **Become root.**
2. **Create a file system for each slice with the `newfs` command.**

```
# newfs /dev/rdsk/cntndnsn
```

In this command,

`/dev/rdsk/cntndnsn`	Is the raw device for the file system to be created.

See Chapter 33, "Creating File Systems," for more information about the `newfs` command.

Where to Go From Here

If You Are Adding A ...	Then ...
System Disk	You need to restore the root (`/`) and `/usr` file systems on the disk. Go to Chapter 43, "Restoring Files and File Systems."
Secondary Disk	You may need to restore file systems on the new disk. Go to Chapter 43, "Restoring Files and File Systems." If you are not restoring file systems on the new disk, you are finished adding a secondary disk. See Chapter 34, "Mounting and Unmounting File Systems" for information on making the file systems available to users.

SPARC: How to Restore File Systems

To restore the root (/) and `/usr` file systems on a system disk or to restore other file systems on a secondary disk, see Chapter 43, "Restoring Files and File Systems."

Where to Go From Here

If you are adding a system disk, you must install a boot block on the disk. Go to "SPARC: How to Install a Boot Block on a System Disk" on page 476.

▼ SPARC: How to Install a Boot Block on a System Disk

1. **Become root.**
2. **Install a boot block on a system disk using the `installboot` command.**

```
# /usr/sbin/installboot /usr/platform/‘uname -i‘/lib/fs/ufs/bootblk
/dev/rdsk/cntndns0
```

In this command,

`/usr/platform/‘uname -i‘/lib/fs/ufs/bootblk`	Is the boot block code.
`/dev/rdsk/`*cntndn*`s0`	Is the raw device of the root (/) file system.

3. **Reboot the system to multi-user mode.**

```
# init 6
```

SPARC: Example—Installing a Boot Block on a System Disk

The following example installs the boot block on a SPARCstation 10.

```
# installboot /usr/platform/sun4m/lib/fs/ufs/bootblk /dev/rdsk/c0t0d0s0
```

SPARC: Verification—Installing a Boot Block on a System Disk

If the system boots to multi-user mode, the boot block has been installed correctly.

x86: Adding a Disk 30

This chapter provides the procedures for adding a disk on an x86 system. This is a list of the step-by-step instructions in this chapter.

For overview information about disk management, see Chapter 27, "Overview of Disk Management."

30

About System and Secondary Disks

A system disk contains the root (/) or /usr file systems, or both. If the disk containing either of these file system becomes damaged, you have two ways to recover:

- You can reinstall the entire Solaris environment
- Or, you can replace the system disk and restore your file systems from a backup medium.

A secondary disk doesn't contain the root (/) and /usr file systems. It usually contains space for user files. You can add a secondary disk to a system for more disk space or you can replace a damaged secondary disk. If you replace a secondary disk on a system, you can restore the old disk's data on the new disk.

Adding a System or Secondary Disk

Table 30-1 x86: Adding a System or Secondary Disk

Activity	Description	For Instructions, Go To	
Connect the Disk and Boot	**System Disk** Connect the new disk and boot from a local or remote Solaris 2.x CD.	▼ "x86: How to Connect a System Disk and Boot"	page 481
	Secondary Disk Connect the new disk and perform a reconfiguration boot, so the system will recognize the new disk.	▼ "x86: How to Connect a Secondary Disk and Boot"	page 483
Create a Solaris `fdisk` Partition	**System Disk Only** Create the `fdisk` partition to hold the Solaris operating environment.	▼ "x86: How to Create a Solaris fdisk Partition"	page 485
Create Slices and Label the Disk	Create disk slices and label the disk if it has not already been done by the disk manufacturer.	▼ "x86: How to Create Disk Slices and Label a Disk"	page 487
Create File Systems	Create UFS file systems on the disk slices with the `newfs` command. You must create the root (`/`) or `/usr` file system (or both) for a system disk.	▼ x86: How to Create File Systems	page 490
Restore File Systems	Restore the root (`/`) or `/usr` file system (or both) on the system disk. If necessary, restore file systems on the secondary disk.	▼ x86: How to Restore File Systems	page 490
Install Boot Block	**System Disk Only** Install the boot block on the `root` (`/`) file system, so the system can boot.	▼ "x86: How to Install a Boot Block on a System Disk"	page 492

x86: Guidelines for Creating an fdisk Partition

Follow these guidelines when setting up the fdisk partition.

- The disk can be divided into a maximum of four fdisk partitions. One of these partitions must be a Solaris partition.
- The Solaris partition must be made the active partition on the disk. The active partition is the one whose operating system will be booted by default at system start-up.
- Solaris fdisk partitions must begin on cylinder boundaries.
- Solaris fdisk partitions must begin at cylinder 1, not cylinder 0, on the first disk because additional boot information, including the master boot record, is written in sector 0.
- The Solaris fdisk partition can be the entire disk or you may want to make it smaller to allow room for a DOS partition. You can also make a new fdisk partition on a disk without disturbing existing partitions (if there is enough room to create a new one).

For x86 systems – Solaris slices are sometimes called partitions. This user guide uses the term slice, but some Solaris documentation and programs may refer to a slice as a partition. To avoid confusion, Solaris 2.x documentation tries to distinguish between fdisk partitions (which are supported only on Solaris for x86) and the divisions within the Solaris fdisk partition, which may be called slices or partitions.

▼ x86: How to Connect a System Disk and Boot

This procedure assumes that the system is down.

1. **Disconnect the damaged system disk from the system.**
2. **Make sure the disk you are adding has a different target number than the other devices on the system.**
   You will often find a small switch located at the back of the disk for this purpose.
3. **Connect the replacement system disk to the system and check the physical connections.**
   Refer to the disk's hardware installation guide for installation details. Also, refer to the *x86 Device Configuration Guide* about hardware configuration requirements specific to the disk.
4. **Follow steps a-e if you are booting from a local or remote Solaris 2.x CD.**
   If you are booting from the network, skip step a.
   a. **Insert the Solaris 2.x installation CD into the CD-ROM drive.**
   b. **Insert the Solaris boot diskette into the primary diskette drive (DOS drive A).**
   c. **Press any key to reboot the system if the system displays the** `type any key to reboot` **prompt. Or, use the reset button to restart the system if the system is shut down.**

      The Multiple Device Boot Subsystem menu is displayed after a few minutes.
   d. **Select the CD-ROM drive or net(work) as the boot device from the Multiple Device Boot menu.**
      The Secondary Boot Subsystem menu is displayed.
   e. **Boot the system in single-user mode.**

```
Select the type of installation: b -s
```

After a few minutes, the root prompt (#) is displayed.

Where to Go From Here

After you boot the system, you can create slices and a disk label on the disk. Go to "x86: How to Create Disk Slices and Label a Disk" on page 487.

x86: How to Connect a Secondary Disk and Boot

1. **Become root.**

2. **If the disk is unsupported by the Solaris software, add the device driver for the disk by following the instructions included with the hardware.**

3. **Create the** `/reconfigure` **file that will be read when the system is booted.**

```
# touch /reconfigure
```

The `/reconfigure` file will cause the SunOS software to check for the presence of any newly installed peripheral devices when you power on or boot your system later.

4. **Shut down the system.**

```
# shutdown -i0 -g30 -y
```

In this command,

`-i0`	Brings the system down to init state 0 (zero), the power-down state.
`-g30`	Notifies logged-in users that they have *n* seconds before the system begins to shut down.
`-y`	Specifies the command should run without user intervention.

The `Type any key to reboot` prompt is displayed.

5. **Turn off power to the system and all external peripheral devices.**

6. **Make sure the disk you are adding has a different target number than the other devices on the system.**
You will often find a small switch located at the back of the disk for this purpose.

7. **Connect the disk to the system and check the physical connections.**
Refer to the disk's hardware installation guide for installation details. Also, refer to the *x86 Device Configuration Guide* about hardware configuration requirements specific to the disk.

8. **Turn on the power to all external peripherals.**

9. **Turn on the power to the system.**
The system will boot and display the login prompt.

Where to Go From Here

After you boot the system, you can create slices and a disk label on the disk. Go to "x86: How to Create Disk Slices and Label a Disk" on page 487.

x86: How to Create a Solaris `fdisk` Partition

1. **Make sure you have read "x86: Guidelines for Creating an fdisk Partition" on page 480.**

2. **Become root.**

3. **Start the `format` utility.**

```
# format
```

4. **Enter the number of the disk on which to create a Solaris `fdisk` partition from the list displayed on your screen.**

```
Specify disk (enter its number): disk-number
```

At this prompt,

disk-number	Is the number of the disk on which to create a Solaris `fdisk` partition.

5. **Go into the `fdisk` menu.**

```
format> fdisk
```

6. **Select the option 1, `Create a partition`.**

```
Selection: 1
```

7. **Create a Solaris fdisk partition by selecting `1(=Solaris)`.**

```
Indicate the type of partition you want to create
  (1=SOLARIS, 2=UNIX, 3=PCIXOS, 4=Other, 8=DOSBIG)
  (5=DOS12, 6=DOS16, 7=DOSEXT, 0=Exit) ? 1
```

8. **Activate a Solaris `fdisk` partition spanning the entire disk by specifying `100` at the prompt.**

```
Indicate the percentage of the disk you want this partition
to use (or enter "c" to specify in cylinders). 100
```

9. **Activate a Solaris `fdisk` partition spanning the entire disk by specifying `y` at the prompt.**

```
The recommended default partitioning for your disk is:

  a 100% “SOLARIS System” partition.

To select this, please type “y”.  To partition your disk
differently, type “n” and the “fdisk” program will let you
select other partitions. y
```

The `Selection` menu displays after the `fdisk` partition is created.

10. **Update the disk configuration and exit the `fdisk` menu from the selection menu.**

```
Selection: 4
```

11. **Quit the `format` menu.**

```
format> quit
```

Where to Go From Here

After you create a Solaris `fdisk` partition on the disk, you can create slices on the disk. Go to “x86: How to Create Disk Slices and Label a Disk” on page 487.

x86: How to Create Disk Slices and Label a Disk

1. **Become root.**
2. **Start the `format` utility.**

```
# format
```

3. **Enter the number of the disk that you want to repartition from the list displayed on your screen.**

```
Specify disk (enter its number): disk-number
```

At this prompt,

disk-number	Is the number of the disk that you want to repartition.

4. **Go into the `partition` menu (which lets you set up the slices).**

```
format> partition
```

5. **Display the current partition (slice) table.**

```
partition> print
```

6. **Start the modification process.**

```
partition> modify
```

7. **Set the disk to all free hog.**

```
Choose base (enter number) [0]? 1
```

See "Using the Free Hog Slice" on page 428 for more information about the free hog slice.

8. **Create a new partition table by answering** `yes` **when prompted to continue.**

```
Do you wish to continue creating a new partition
table based on above table[yes]? yes
```

9. **Identify the free hog partition (slice) and the sizes of the slices when prompted.**

   When adding a system disk, you must set up slices for:
   - root (slice 0), swap (slice 1) and/or
   - `/usr` (slice 6)

   After you identify the slices, the new partition table is displayed.

10. **Make the displayed partition table the current partition table by answering** `yes` **when asked.**

```
Okay to make this the current partition table[yes]? yes
```

   If you don't want the current partition table and you want to change it, answer `no` and go to Step 6.

11. **Name the partition table.**

```
Enter table name (remember quotes): "partition-name"
```

   At this prompt,

   partition-name Is the name for the new partition table.

12. **Label the disk with the new partition table when you have finished allocating slices on the new disk.**

```
Ready to label disk, continue? yes
```

13. **Quit the** `partition` **menu.**

```
partition> quit
```

14. **Quit the** `format` **menu.**

```
format> quit
```

Where to Go From Here

After you create disk slices and label the disk, you can create file systems on the disk. Go to “x86: How to Create File Systems” on page 490.

▼ x86: How to Create File Systems

1. **Become root.**
2. **Create a file system for each slice with the `newfs` command.**

```
# newfs /dev/rdsk/cntndnsn
```

In this command,

`/dev/rdsk/cntndnsn`	Is the raw device for the file system to be created.

See Chapter 34, “Mounting and Unmounting File Systems,” for more information about the `newfs` command.

Where to Go From Here

If You Are Adding A ...	Then ...
System Disk	You need to restore the root (`/`) and `/usr` file systems on the disk. Go to Chapter 43, “Restoring Files and File Systems.”
Secondary Disk	You may need to restore file systems on the new disk. Go to Chapter 43, “Restoring Files and File Systems.” If you are not restoring file systems on the new disk, you are finished adding a secondary disk. See Chapter 34, “Mounting and Unmounting File Systems” for information on making the file systems available to users.

▼ x86: How to Restore File Systems

To restore the root (`/`) and `/usr` file systems on a system disk or to restore other file systems on a secondary disk, see Chapter 43, “Restoring Files and File Systems.”

Where to Go From Here

If you are adding a system disk, you must install a boot block on the disk. Go to "x86: How to Install a Boot Block on a System Disk" on page 492.

▼ x86: How to Install a Boot Block on a System Disk

1. **Become root.**

2. **Type the following command:**

```
# /usr/sbin/installboot /usr/platform/‘uname -i‘/lib/fs/ufs/pboot
/usr/platform/‘uname -i‘/lib/fs/ufs/bootblk /dev/rdsk/cntndns2
```

In this command:

`/usr/platform/‘uname -i‘/lib/fs/ufs/pboot`	Is the partition boot file.
`/usr/platform/‘uname -i‘/lib/fs/ufs/bootblk`	Is the boot block code.
`/dev/rdsk/`*cntndn*`s2`	Is the raw device name that represents the whole disk.

3. **Reboot the system to multi-user mode.**

Example—Installing a Boot Block on a System Disk

```
# /usr/sbin/installboot /usr/platform/‘uname -i‘/lib/fs/ufs/pboot /usr/platform/‘uname -i‘/
lib/fs/ufs/bootblk /dev/rdsk/cntndns2
```

Verification—Installing a Boot Block on a System Disk

If the system boots to multi-user mode, the boot block has been installed correctly.

The `format` *Utility* 31

This chapter describes the `format` utility's menu and commands.

This is a list of the overview information in this chapter.

See Chapter 27, "Overview of Disk Management," for a conceptual overview of when to use the `format` utility.

Requirements or Restrictions for Using the format Utility

You must be root to use the format utility. If you are not root, you will see the following error message when you try to use format.

```
% format
   Searching for disk...done
   No permission (or no disk found)!
```

Recommendations for Preserving Information When Using format

- Back up all files on the disk drive before doing anything else.
- Save all your defect lists in files by using the format dump command. The file name should include the drive type, model number, and serial number.
- Save the paper copies of the manufacturer's defect list shipped with your drive.

Format Menu and Command Descriptions

The format main menu looks like the following:

```
FORMAT MENU:
          disk      - select a disk
          type      - select (define) a disk type
          partition - select (define) a partition table
          current   - describe the current disk
          format    - format and analyze the disk
          fdisk     - run the fdisk program (x86 systems only)
          repair    - repair a defective sector
          label     - write label to the disk
          analyze   - surface analysis
          defect    - defect list management
          backup    - search for backup labels
          verify    - read and display labels
          save      - save new disk/partition definitions
          inquiry   - show vendor, product and revision
          volname   - set 8-character volume name
          quit
```

Table 31-1 describes the `format` main menu items.

Table 31-1 The `format` Main Menu Item Descriptions

Item	Command or Menu?	Allows You To ...
`disk`	command	Choose the disk that will be used in subsequent operations (known as the current disk). All of the system's drives are listed.
`type`	command	Identify the manufacturer and model of the current disk. A list of known drive types is displayed. Choose the `Auto configure` option for all SCSI-2 disk drives.
`partition`	menu	Create and modify slices. See "The partition Menu" on page 497 for more information.
`current`	command	Display the following information about the current disk: • Device name and type • Number of cylinders, alternate cylinders, heads and sectors • Physical device name
`format`	command	Format the current disk, using one of these sources of information in this order: • Information found in the `format.dat` file • Information from the automatic configuration process • Information prompted for if there is no `format.dat` entry
`fdisk`	menu	Run the `fdisk` program to create a Solaris `fdisk` partition.
`repair`	command	Repair a specific block on the disk.
`label`	command	Write a new label to the current disk
`analyze`	menu	Run read, write, compare tests. See "The analyze Menu" on page 500 for more information.
`defect`	menu	Retrieve and print defect lists. See "The defect Menu" on page 502 for more information.
`backup`	command	Search for backup labels.
`verify`	command	Print the following information about the disk: • Device name and type • Number of cylinders, alternate cylinders, heads and sectors • Partition table
`save`	command	Save new disk and partition information.

Table 31-1 The `format` Main Menu Item Descriptions *(Continued)*

Item	Command or Menu?	Allows You To ...
`inquiry`	command	Print the vendor, product name, and revision level of the current drive (SCSI disks only).
`volname`	command	Label the disk with a new eight-character volume name.
`quit`	command	Exit the `format` menu.

The `partition` *Menu*

The partition menu looks like this.

```
format> partition
PARTITION MENU:
         0       - change ‘0' partition
         1       - change ‘1' partition
         2       - change ‘2' partition
         3       - change ‘3' partition
         4       - change ‘4' partition
         5       - change ‘5' partition
         6       - change ‘6' partition
         7       - change ‘7' partition
         select - select a predefined table
         modify - modify a predefined partition table
         name   - name the current table
         print  - display the current table
         label  - write partition map and label to the disk
         quit
partition>
```

Table 31-2 describes the `partition` menu items.

Table 31-2 The `partition` Menu Item Descriptions

The Command ...	Allows You To ...
`change` ‘*x*’ `partition`	Specify new slice: • Identification tag • Permission flags • Starting cylinder • Size
`select`	Choose a predefined slice table.
`modify`	Change all the slices in the slice table. This command is preferred over the individual `change` ‘*x*’ `partition` commands.
`name`	Specify a name for the current slice table.

Table 31-2 The `partition` Menu Item Descriptions *(Continued)*

The Command ...	Allows You To ...
`print`	View the current slice table.
`label`	Write the slice map and label to the current disk.
`quit`	Exit the partition menu.

x86: The `fdisk` Menu

The `fdisk` menu appears on x86 systems only and looks like this.

```
format> fdisk
             Total disk size is 1855 cylinders
             Cylinder size is 553 (512 byte) blocks
                                               Cylinders
      Partition   Status    Type      Start   End   Length    %
      =========   ======    ========  =====   ===   ======   ===
          1                 DOS-BIG       0   370     371     20
          2       Active    SOLARIS     370  1851    1482     80

SELECT ONE OF THE FOLLOWING:

     1.   Create a partition
     2.   Change Active (Boot from) partition
     3.   Delete a partition
     4.   Exit (Update disk configuration and exit)
     5.   Cancel (Exit without updating disk configuration)
Enter Selection:
```

Table 31-3 describes the `fdisk` menu items.

Table 31-3 The `fdisk` Menu Item Descriptions

The Command ...	**Allows You To ...**
`Create a partition`	Create an `fdisk` partition. You must create a separate partition for each operating system such as Solaris or DOS. There is a maximum of 4 partitions per disk. You will be prompted for the size of the `fdisk` partition as a percentage of the disk.
`Change Active partition`	Specify which partition will be used for booting. This identifies where the first stage boot program will look for the second state boot program.
`Delete a partition`	Delete a previously created partition. This command will destroy all the data in the partition.
`Exit`	Write a new version of the partition table and exit the `fdisk` menu.
`Cancel`	Exit the `fdisk` menu without modifying the partition table.

The `analyze` Menu

The `analyze` menu looks like this.

```
format> analyze

ANALYZE MENU:
     read      - read only test   (doesn't harm SunOS)
     refresh  - read then write  (doesn't harm data)
     test      - pattern testing  (doesn't harm data)
     write     - write then read      (corrupts data)
     compare  - write, read, compare (corrupts data)
     purge     - write, read, write   (corrupts data)
     verify    - write entire disk, then verify (corrupts data)
     print     - display data buffer
     setup     - set analysis parameters
     config    - show analysis parameters
     quit
analyze>
```

Table 31-4 describes the `analyze` menu items.

Table 31-4 The `analyze` Menu Item Descriptions

The Command ...	Allows You To ...
`read`	Read each sector on this disk. Repairs defective blocks as a default.
`refresh`	Read then write data on the disk without harming the data. Repairs defective blocks as a default.
`test`	Write a set of patterns to the disk without harming the data. Repairs defective blocks as a default.
`write`	Write a set of patterns to the disk then read the data on the disk back. Destroys existing data on the disk. Repairs defective blocks as a default.
`compare`	Write a set of patterns to the disk, read the data back, and compare it to the data in the write buffer. Destroys existing data on the disk. Repairs defective blocks as a default.

Table 31-4 The `analyze` Menu Item Descriptions *(Continued)*

The Command ...	Allows You To ...
`purge`	Remove all data from the disk so that the data can't be retrieved by any means. Data is removed by writing three distinct patterns over the entire disk (or section of the disk), then writing an hex-bit pattern if the verification passes. Repairs defective blocks as a default.
`verify`	Write unique data to each block on the entire disk in the first pass. Read and verify the data in the next pass. Destroys existing data on the disk. Repairs defective blocks as a default.
`print`	View the data in the read / write buffer.
`setup`	Specify the following analysis parameters `Analyze entire disk?` **`yes`** `Starting block number:` *depends on drive* `Ending block number:` *depends on drive* `Loop continuously?` **`no`** `Number of passes:` **`2`** `Repair defective blocks?` **`yes`** `Stop after first error?` **`no`** `Use random bit patterns?` **`no`** `Number of blocks per transfer:` **`126 (0/`*`n`*`/`*`nn`*`)`** `Verify media after formatting?` **`yes`** `Enable extended messages?` **`no`** `Restore defect list?` **`yes`** `Restore disk label?` **`yes`** Defaults are shown in bold.
`config`	View the current analysis parameters.
`quit`	Exit the `analyze` menu.

The `defect` Menu

The `defect` menu looks like this.

```
format> defect

DEFECT MENU:
        primary  - extract manufacturer's defect list
        grown     - extract manufacturer's and repaired defects lists
        both      - extract both primary and grown defects lists
        print     - display working list
        dump      - dump working list to file
        quit
defect>
```

Table 31-5 describes the `defect` menu items.

Table 31-5 The `defect` Menu Item Descriptions

The Command ...	Allows You To ...
`primary`	Read the manufacturer's defect list from the disk drive and update the in-memory defect list.
`grown`	Read the grown defect list (defects that have been detected during analysis) and update the in-memory defect list.
`both`	Read both the manufacturer's and grown defect list and update the in-memory defect list.
`print`	View the in-memory defect list.
`dump`	Save the in-memory defect list to a file.
`quit`	Exit the defect menu.

Files Used by `format`—`format.dat`

The `format` data, `/etc/format.dat`, contains:

- Disk types
- Default slice tables

The format.dat file shipped with the Solaris operating system supports many standard disks. If your disk drive is not listed in the format.dat file, you can choose to add an entry for it or allow format to prompt you for the information it needs while it is performing operations.

Adding an entry to the format.dat file can save time if the disk drive will be used throughout your site. To use the format.dat file on other systems, copy the file to each system that will use the specific disk drive you added to the format.dat file.

You should modify the data file for your system if you have one of the following:

- A disk that is not supported by the Solaris operating system
- A disk with a slice table that is different from the Solaris operating system default configuration

---

Note – Do not alter default entries. If you want to alter the default entries, copy the entry, give it a different name, and make the modification to avoid confusion.

Structure

The format data file (format.dat) contains specific disk drive information used by the format utility. Three items are defined in the format.dat file:

- Search paths
- Disk types
- Slice tables

Syntax

The following syntax rules apply to the data file:

- The pound sign (#) is the comment character. Any text on a line after a pound sign is not interpreted by format.
- Each definition in the format.dat file appears on a single logical line. If the definition is more than one line long, all but the last line of the definition must end with a backslash (\).

- A definition consists of a series of assignments that have an identifier on the left side and one or more values on the right side. The assignment operator is the equal sign (=). The assignments within a definition must be separated by a colon (:).
- White space is ignored by format. If you want an assigned value to contain white space, enclose the entire value in double quotes ("). This will cause the white space within the quotes to be preserved as part of the assignment value.
- Some assignments can have multiple values on the right hand side. Separate values by a comma.

Keywords

The data file contains disk definitions that are read in by format when it is started. Each definition starts with one of the following keywords: search_path, disk_type, and partition, which are described in Table 31-6.

Table 31-6 format.dat Keyword Descriptions

Keyword	**Use**
search_path	This keyword is no longer used in the format.dat file. Starting with the Solaris 2.0 release, the format utility searchs the logical device hierarchy (/dev) so there is no need to set this keyword to find a system's disks.
disk_type	Defines the controller and disk model. Each disk_type definition contains information concerning the physical geometry of the disk. The default data file contains definitions for the controllers and disks that the Solaris operating system supports. You need to add a new disk_type only if you have an unsupported disk. You can add as many disk_type definitions to the data file as you want.
partition	Defines a slice table for a specific disk type. The slice table contains the slice information, plus a name that lets you refer to it in format. The default data file contains default slice definitions for several kinds of disk drives. Add a slice definition if you re-created slices on any of the disks on your system. Add as many slice definitions to the data file as you need.

Disk Type

disk_type defines the controller and disk model. Each disk_type definition contains the physical geometry of the disk. The default data file contains definitions for the controllers and disks that the Solaris operating system supports. You need to add a new disk_type only if you have an unsupported disk. You can add as many disk_type definitions to the data file as you want.

The keyword itself is assigned the name of the disk type. This name appears in the disk's label, and is used to identify the disk type whenever format is run. Enclose the name in double quotes to preserve any white space in the name. Table 31-7 describes the identifiers that must also be assigned values in all disk_type definitions.

Table 31-7 Required disk_type Identifiers

Identifier	Description
ctlr	Valid controller type for the disk type. Currently, the supported values for this assignment are SCSI and ISP-80 (IPI controller).
ncyl	The number of data cylinders in the disk type. This determines how many logical cylinders of the disk the system will be allowed to access.
acyl	The number of alternate cylinders in the disk type. These cylinders are used by format to store information such as the defect list for the drive. You should always leave at least two cylinders for alternates.
pcyl	The number of physical cylinders in the disk type. This number is used to calculate the boundaries of the disk media. This number is usually equal to ncyl plus acyl.
nhead	The number of heads in the disk type. This number is used to calculate the boundaries of the disk media.
nsect	The number of data sectors per track in the disk type. This number is used to calculate the boundaries of the disk media. Note that this is only the data sectors, any spares are not reflected in the assignment.
rpm	The rotations per minute of the disk type. This information is put in the label and later used by the file system to calculate the optimal placement of file data.

Other assignments may be necessary depending on the controller. Table 31-8 describes the assignments required for SCSI controllers.

Table 31-8 `disk_type` Identifiers for SCSI Controllers

Identifier	Description
`fmt_time`	A number indicating how long it takes to format a given drive. See the controller manual for more information.
`cache`	A number that controls the operation of the onboard cache while `format` is operating. See the controller manual for more information.
`trks_zone`	A number that specified how many tracks you have per defect zone, to be used in alternate sector mapping. See the controller manual for more information.
`asect`	The number assigned to this parameter specifies how many sectors are available for alternate mapping within a given defect zone. See the controller manual for more information.

Below are some examples of `disk_type` definitions:

```
disk_type = "SUN0535" \
    : ctlr = SCSI : fmt_time = 4 \
    : ncyl = 1866 : acyl = 2 : pcyl = 2500 : nhead = 7 : nsect = 80 \
    : rpm = 5400
disk_type = "SUN0669" \
    : ctlr = SCSI : fmt_time = 4 \
    : trks_zone = 15 : asect = 5 : atrks = 30 \
    : ncyl =  1614 : acyl = 2 : pcyl = 1632 : nhead = 15 : nsect = 54 \
    : rpm = 3600 : bpt = 31410
disk_type = "SUN1.0G" \
    : ctlr = SCSI : fmt_time = 4 \
    : trks_zone = 15 : asect = 5 : atrks = 30 \
    : ncyl =  1703 : acyl = 2 : pcyl = 1931 : nhead = 15 : nsect = 80 \
    : rpm = 3597 : bpt = 41301
```

Slice Tables

A `partition` definition keyword is assigned the name of the slice table. Enclose the name in double quotes to preserve any white space in the name. Table 31-9 describes the identifiers that must be assigned values in all slice tables.

Table 31-9 Required Identifiers for Slice Tables

Identifier	Description
`disk`	The name of the `disk_type` that this slice table is defined for. This name must appear exactly as it does in the `disk_type` definition.
`ctlr`	The disk controller type this slice table can be attached to. Currently, the supported values for this assignment are ISP-80 for IPI controllers and SCSI for SCSI controllers. The controller type specified here must also be defined for the `disk_type` chosen above.

The other assignments in a slice definition describe the actual slice information. The identifiers are the numbers 0 through 7. These assignments are optional. Any slice not explicitly assigned is set to 0 length. The value of each of these assignments is a pair of numbers separated by a comma. The first number is the starting cylinder for the slice, and the second is the number of sectors in the slice. Below are some examples of slice definitions:

```
partition = "SUN0535" \
    : disk = "SUN0535" : ctlr = SCSI \
    : 0 = 0, 64400 : 1 = 115, 103600 : 2 = 0, 1044960 : 6 = 300, 876960
partition = "SUN0669" \
    : disk = "SUN0669" : ctlr = SCSI \
    : 0 = 0, 32400 : 1 = 40, 64800 : 2 = 0, 1307340 : 6 = 120, 1210140
partition = "SUN1.0G" \
    : disk = "SUN1.0G" : ctlr = SCSI \
    : 0 = 0, 32400 : 1 = 27, 64800 : 2 = 0, 2043600 : 6 = 81, 1946400
```

Location

The `format` utility learns of the location of your data file by the following methods.

1. If a filename is given with the `-x` command line option, that file is always used as the data file.

2. If the `-x` option is not specified, then `format` looks in the current directory for a file named `format.dat`. If the file exists, it is used as the data file.

3. If neither of these methods yields a data file, `format` uses `/etc/format.dat` as the data file. This file is shipped with the Solaris operating system and should always be present.

Rules for Input to `format` Commands

When using format, you need to provide information in various formats. This section describes the rules for the formats. See "Help" on page 511 for information on using `format`'s help facility when inputting data.

Numbers

Several places in `format` require an integer as input. You must either specify the data or select one from a list of choices. In either case, the `help` facility causes `format` to print the upper and lower limits of the integer expected. Simply enter the number desired. The number is assumed to be in decimal format unless a base is explicitly specified as part of the number (for example, 0x for hexadecimal).

The following are examples of integer input:

```
Enter number of passes [2]: 34
Enter number of passes [34] 0xf
```

Block Numbers

Whenever you are required to specify a disk block number, there are two ways to input the information:

- Block number as an integer
- Block number in the cylinder / head / sector format

You can specify the information as an integer representing the logical block number. You can specify the integer in any base, but the default is decimal. The maximum operator (a dollar sign, `$`) can also be used here to let `format` select the appropriate value. Logical block format is used by the SunOS disk drivers in error messages.

The other way to specify a block number is the cylinder / head / sector format. In this format, you must specify explicitly the three logical components of the block number, the cylinder, head, and sector values. These values are still logical, but they allow you to define regions of the disk related to the layout of the media.

If any of the cylinder / head / sector numbers are not specified, the appropriate value is assumed to be zero. You can also use the maximum operator in place of any of the numbers and let `format` select the appropriate value. Below are some examples of cylinder, head, and sector entries:

```
Enter defective block number: 34/2/3
Enter defective block number: 23/1/
Enter defective block number: 457//
Enter defective block number: 12345
Enter defective block number: Oxabcd
Enter defective block number: 334/$/2
Enter defective block number: 892//$
```

`format` always prints block numbers, in both of the above formats. Also, the `help` facility shows you the upper and lower bounds of the block number expected, in both formats.

Command Names

Command names are needed as input whenever `format` is displaying a menu prompt. You can *abbreviate* the command names, as long as what is entered is sufficient to uniquely identify the command desired.

For example, use `p` to enter the `partition` menu from the `format` menu. Then enter `p` to display the current slice table.

```
format> p
PARTITION MENU:
        0      - change `0' partition
        1      - change `1' partition
        2      - change `2' partition
        3      - change `3' partition
        4      - change `4' partition
        5      - change `5' partition
        6      - change `6' partition
        7      - change `7' partition
        select - select a predefined table
        modify - modify a predefined partition table
        name   - name the current table
        print  - display the current table
        label  - write partition map and label to the disk
        quit
partition> p
```

Other Names

There are certain times in `format` when you must name something. In these cases, you are free to specify any string desired for the name. If the name has white space in it, the entire name must be enclosed in double quotes ("). Otherwise, only the first word of the name is used.

Help

The `format` utility provides a help facility you can use whenever `format` is expecting input. You can request help about what information is expected by entering a question mark (?). The `format` utility displays a brief description of what type of input is needed.

If you enter a `?` at a menu prompt, a list of available commands is displayed.

Associated Man Pages

The man pages associated with the `format` utility is `format(1M)`, which describes the basic `format` utility capabilities and provides descriptions of all command line variables, and `format.dat(4)`, which describes disk drive configuration information for the `format` utility.

Part 8 — Managing File Systems

This part provides instructions for managing file systems in the Solaris environment.

Overview of File Systems 32

This is a list of the overview information in this chapter.

A *file system* is a structure of directories used to organize and store files. The term "file system" is used in several different ways:

- To describe a particular type of file system: disk-based, network-based, or virtual
- To describe the entire file tree from the root directory downward
- To describe the data structure of a disk slice or other media storage device
- To describe a portion of a file tree structure that is attached to a mount point on the main file tree so that it is accessible

Usually, you can tell from context which meaning is intended.

The Solaris system software uses the *virtual file system* (VFS) architecture, which provides a standard interface for different file system types. The VFS architecture enables the kernel to handle basic operations, such as reading, writing, and listing files, without requiring the user or program to know about the underlying file system type.

The file system administrative commands provide a common interface that enables you to maintain file systems of differing types. These commands have two components: a generic component and a component specific to each type of file system. The generic commands apply to most types of file systems, while the specific commands apply to only one type of file system.

Administering file systems is one of your most important system administration tasks. The file system story is a complex one, and understanding it can help you more effectively administer file systems. Read this chapter for background and planning information. Refer to other chapters in this part or to other parts in the *System Administration Guide* for instructions about these tasks:

- **Setting up new UFS and virtual file systems –** See Chapter 33, “Creating File Systems,” and Chapter 35, “Setting Up and Maintaining the Cache File System,” for detailed information. The various types of virtual file systems are described later in this chapter.
- **Making local and remote files available to users –** See Chapter 34, “Mounting and Unmounting File Systems,” for detailed information.
- **Connecting and configuring new storage devices** – See "Managing Disks" in the *System Administration Guide, Volume I* for detailed information.
- **Designing and implementing a backup schedule and restoring files and file systems as needed –** See Chapter 6, “Planning a Backup Strategy,” for information on designing a backup schedule. See Chapter 7, “Backing Up Files and File Systems,” for detailed information about doing backups. See Chapter 8, “Restoring Files and File Systems,” for detailed information about restoring files and file systems.
- **Checking for and correcting file system damage –** See Chapter 39, “Checking File System Integrity,” for detailed information on how to proceed if the automatic (boot time) checking fails.

Types of File Systems

The Solaris system software supports three types of file systems:

- Disk-based
- Network-based
- Virtual

To identify the type for a particular file system, see "Determining a File System's Type" on page 541.

Disk-based File Systems

Disk-based file systems are stored on physical media such as hard disks, CD-ROMs, and diskettes. Disk-based file systems can be written in different formats. The available formats are:

- **UFS –** UNIX file system (based on the BSD Fat Fast File system that was provided in the 4.3 Tahoe release). UFS is the default disk-based file system in SunOS system software.
- **HSFS –** High Sierra and ISO 9660 file system. High Sierra is the first CD-ROM file system; ISO 9660 is the official standard version of the High Sierra File System. The HSFS file system is used on CD-ROM, and is a read-only file system. Solaris HSFS supports Rock Ridge extensions to ISO 9660, which, when present on a CD-ROM, provide all UFS file system semantics and file types except for writability and hard links.
- **PCFS –** PC file system, which allows read / write access to data and programs on DOS-formatted disks written for DOS-based personal computers.

Each type of disk-based file system is customarily associated with a particular media device:

- UFS with hard disk
- HSFS with CD-ROM
- PCFS with diskette

These associations are not, however, restrictive. For example, CD-ROMs and diskettes can have UFS file systems put on them.

Network-based File Systems

Network-based file systems are file systems accessed over the network. Typically, network-based file systems reside on one system and are accessed by other systems across the network. NFS is the only available network-based file system.

NFS is the default distributed file system for Solaris. You administer distributed file systems by exporting them from a server and mounting them on individual systems. See “Mounting and Unmounting” on page 533 for more information.

Virtual File Systems

Virtual file systems are memory-based file systems that provide access to special kernel information and facilities. Most virtual file systems do not use file system disk space. However, Cache File Systems use a file system on the disk to contain the cache, and some virtual file systems, such as the temporary file system, use the swap space on a disk.

The Cache File System

The Cache File System (CacheFS™) can be used to improve performance of remote file systems or slow devices such as CD-ROM drives. When a file system is cached, the data read from the remote file system or CD-ROM is stored in a cache on the local system. See Chapter 35, “Setting Up and Maintaining the Cache File System,” for detailed information on setting up and administering Cache File Systems.

The Temporary File System

The Temporary File System (TMPFS) uses local memory for file system reads and writes. Access to files in a TMPFS file system is typically much faster than to files in a UFS file system. Files in the temporary file system are not permanent. They are deleted when the file system is unmounted and when the system is shut down or rebooted.

TMPFS is the default file system type for the `/tmp` directory in the Solaris system software. You can copy or move files into or out of the `/tmp` directory, just as you would in a UFS `/tmp` file system.

Using TMPFS file systems can improve system performance by saving the cost of reading and writing temporary files to a local disk or across the network. For example, temporary files are created when you compile a program. The operating system generates a lot of disk or network activity while manipulating these files. Using TMPFS to hold these temporary files may significantly speed up their creation, manipulation, and deletion.

The TMPFS file system uses swap space as a temporary backing store. If a system with a TMPFS file system does not have adequate swap space, two problems can occur:

- The TMPFS file system can run out of space, just as a regular file system can fill up.
- Because TMPFS allocates swap space to save file data (if necessary), some programs may not be able to execute because there is not enough swap space.

See Chapter 33, "Creating File Systems," for information on how to create TMPFS file systems. See Chapter 37, "Configuring Additional Swap Space," for information about increasing swap space.

The Loopback File System

The Loopback File System (LOFS) lets you create a new virtual file system so you can access files by using an alternative path name. For example, you can create a loopback mount of `/` onto `/tmp/newroot`. The entire file system hierarchy looks like it is duplicated under `/tmp/newroot`, including any file systems mounted from NFS servers. All files are accessible either with a path name starting from `/`, or with a path name starting from `/tmp/newroot`.

See Chapter 33, "Creating File Systems," for information on how to create LOFS file systems.

The Process File System

The Process File System (PROCFS) resides in memory. It contains a list of active processes, by process number, in the `/proc` directory. Information in the `/proc` directory is used by commands like `ps`. Debuggers and other development tools can also access the address space of the processes by using file system calls. The following example shows a partial listing of the contents of the /proc directory:

```
jupiter% ls -l /proc
total 144944
-rw-------   1 root     root          0 Dec 19 15:45 00000
-rw-------   1 root     root     196608 Dec 19 15:45 00001
-rw-------   1 root     root          0 Dec 19 15:45 00002
-rw-------   1 root     root    1028096 Dec 19 15:46 00073
-rw-------   1 root     root    1445888 Dec 19 15:46 00091
-rw-------   1 root     root    1142784 Dec 19 15:46 00093
-rw-------   1 root     root    1142784 Dec 19 15:46 00095
-rw-------   1 ignatz   staff   1576960 Dec 19 15:50 00226
-rw-------   1 ignatz   staff    192512 Dec 19 15:51 00236
-rw-------   1 ignatz   staff   1269760 Dec 19 15:52 00240
-rw-------   1 ignatz   staff   6090752 Dec 19 15:52 00241
-rw-------   1 ignatz   staff    188416 Dec 19 15:52 00247
-rw-------   1 ignatz   staff   2744320 Dec 19 15:52 00256
```

Caution – Do not delete the files in the /proc directory. Deleting processes from the /proc directory is not the best way to kill them. Remember, /proc files do not use disk space, so there is little reason to delete files from this directory.

The /proc directory does not require system administration.

Additional Virtual File Systems

These additional types of virtual file systems are listed for your information. They do not require administration.

- **FIFOFS (first-in first-out) –** Named pipe files that give processes common access to data
- **FDFS (file descriptors) –** Provides explicit names for opening files using file descriptors
- **NAMEFS –** Used mostly by STREAMS for dynamic mounts of file descriptors on top of files

- **SPECFS (special) –** Provides access to character special and block devices
- **SWAPFS –** File system used by the kernel for swapping

File System Administration Commands

Most file system administration commands have a *generic* and a *file system–specific* component. Use the generic commands, which call the file system–specific component. Table 32-1 lists the generic file system administrative commands, which are located in the /usr/bin directory.

Table 32-1 Generic File System Administrative Commands

Command	Description
clri(1M)	Clears inodes
df(1M)	Reports the number of free disk blocks and files
ff(1M)	Lists file names and statistics for a file system
fsck(1M)	Checks the integrity of a file system and repairs any damage found
fsdb(1M)	Debugs the file system
fstyp(1M)	Determines the file system type
labelit(1M)	Lists or provides labels for file systems when copied to tape (for use by the volcopy command only)
mkfs(1M)	Makes a new file system
mount(1M)	Mounts file systems and remote resources
mountall(1M)	Mounts all file systems specified in a file system table
ncheck(1M)	Generates a list of path names with their i-numbers
umount(1M)	Unmounts file systems and remote resources
umountall(1M)	Unmounts all file systems specified in a file system table
volcopy(1M)	Makes an image copy of a file system

Most of these commands also have a file system-specific counterpart.

Note – If you specify an operation on a file system that does not support it, the generic command displays this error message:
command: Operation not applicable for FSType *type*

Syntax of Generic Commands

Most of these commands use this syntax:

command [-F *type*] [-V] [*generic-options*] [-o *specific-options*] [*special* | *mount-point*] [*operands*]

Table 32-2 Options and Arguments to Generic Commands

Option/Argument	Description
-F *type*	The type of file system. If you do not use this option, the command looks for an entry which matches special or mount point in the /etc/vfstab file. Otherwise, the default is taken from the file /etc/default/fs for local file systems and from the file /etc/dfs/fstypes for remote file systems.
-V	An intruction to echo the completed command line. The echoed line may include additional information derived from /etc/vfstab. Use this option to verify and validate the command line. The command is not executed.
generic-options	Options common to different types of file systems.
-o *specific-options*	A list of options specific to the type of file system. The list must have the following format: -o followed by a space, followed by a series of *keyword* [=*value*] pairs separated by commas with no intervening spaces.
special | *mount-point*	The file system indentification. This name must be either the mount point or the special device file for the slice holding the file system. For some commands, the *special* file must be the raw (character) device and for other commands it must be the block device. See "Understanding Disk Device Names" on page 287 for more information about disk device names. In some cases, this argument is used as a key to search the file /etc/vfstab for a matching entry from which to obtain other information. In most cases, this argument is required and must come immediately after *specific-options*. However, it is not required when you want a command to act on all the file systems (optionally limited by type) listed in the /etc/vfstab file.
operands	Arguments specific to a type of file system. See the specific manual page of the command (for example, mkfs_ufs(1M)) for a detailed description.

How the File System Commands Determine the File System Type

The generic file system commands determine the file system type by following this sequence:

1. From `-F`, if supplied.
2. By matching a special device with an entry in `/etc/vfstab` (if `special` is supplied). For example, `fsck` first looks for a match against the `fsck device` field; if no match is found, it then checks against the special device field.
3. By using the default specified in `/etc/default/fs` for local file systems and in `/etc/dfs/fstypes` for remote file systems.

Manual Pages for Generic and Specific Commands

Both the generic and specific commands have manual pages in the *man Pages(1M): System Administration Commands*. The specific manual page is a continuation of the generic manual page. To look at a specific manual page, append an underscore and the file system type abbreviation to the generic command name. For example, to see the specific manual page for mounting an HSFS file system, type `man mount_hsfs(1M)`.

The Default Solaris File Systems

The Solaris file system is hierarchical, starting with the root directory (`/`) and continuing downwards through a number of directories. The system software installs a default set of directories and uses a set of conventions to group similar types of files together. Table 32-3 provides a summary of the default

Solaris file systems, and shows the type of each file system. See Chapter 40, “File System Reference” for more detailed information on the default directories for the root and /usr file systems.

Table 32-3 The Default Solaris File Systems

File System or Directory	File System Type	Description
root (/)	UFS	The top of the hierarchical file tree. The root directory contains the directories and files critical for system operation, such as the kernel, the device drivers, and the programs used to boot the system. It also contains the mount point directories where local and remote file systems can be attached to the file tree.
/usr	UFS	System files and directories that can be shared with other users. Files that run only on certain types of systems are in the /usr directory (for example SPARC executables). Files (such as man pages) that can be used on all types of systems are in /usr/share.
/export/home or /home	NFS, UFS	The mount point for users’ home directories, which store users’ work files. By default /home is an automounted file system. On standalone systems, /home may be a UFS file system on a local disk slice.
/var	UFS	System files and directories that are likely to change or grow over the life of the local system. These include system logs, vi and ex backup files, and uucp files.
/opt	NFS, UFS	Mount point for optional, third-party software. On some systems /opt may be a UFS file system on a local disk slice.
/tmp	TMPFS	Temporary files, cleared each time the system is booted or unmounted.
/proc	PROCFS	A list of active processes, by number.

The root (`/`) and `/usr` file systems are both needed to run a system. Some of the most basic commands from the `/usr` file system (like `mount`) are included in the root (`/`) file system so that they are available when the system boots or is in single-user mode and `/usr` is not mounted. See Chapter 40, “File System Reference,” for a complete list of the default directories.

Swap Space

The Solaris system software uses some disk slices for temporary storage rather than for file systems. These slices are called *swap* slices. Swap slices are used as virtual memory storage areas when the system does not have enough physical memory to handle current processes.

The Solaris 2.x virtual memory system maps physical copies of files on disk to virtual addresses in memory. Physical memory pages which hold the data for these mappings can be backed by regular files in the file system or by swap space. If the memory is backed by swap space it is referred to as *anonymous* memory because the user doesn’t know the names of the files backing the memory.

For a complete conceptual overview of swap space, instructions for adding more swap space, and information on the differences between Solaris 1.x (SunOS 4.x) and Solaris 2.x (SunOS 5.x) swap requirements, see Chapter 37, “Configuring Additional Swap Space.”

The UFS File System

UFS is the default disk-based file system in Solaris system software. Most of the time, when you administer a disk-based file system, you will be administering UFS. UFS provides the following features:

- **State flags –** Show the state of the file system: clean, stable, active, or unknown. These flags eliminate unnecessary file system checks. If the file system is “clean” or “stable,” file system checks are not run.
- **Extended fundamental types (EFT) –** 32-bit user ID (UID), group ID (GID), and device numbers.

- **Large file systems –** A UFS file system can be as large as 1 Tbyte (terabyte) and can have regular files up to 2 Gbytes (gigabytes). The Solaris system software does not provide *striping*, which is required to make a logical slice large enough for a 1-Tbyte file system. However, the Solstice DiskSuite™ software, available from SunSoft, provides this capability.

Note – Solaris device names use the term *slice* (and the letter “s” in the device name) to refer to the slice number.

Figure 32-1 shows how a disk slice can be formatted to contain a UFS file system.

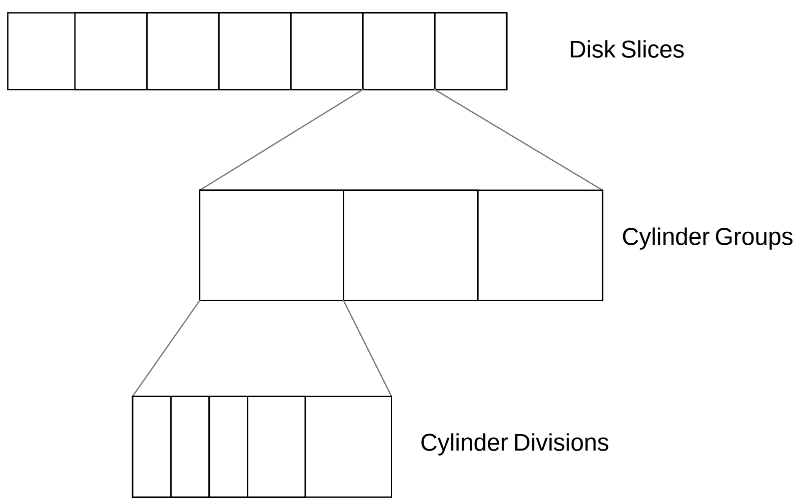


Figure 32-1 How a Disk Slice Is Formatted for a UFS File System

The following sections describe how a disk is divided into slices and cylinder groups, and structured as a UFS file system.

Disk Slices

A slice is composed of a single range of contiguous blocks. It is a physical subset of the disk (except for slice 2, which customarily represents the entire disk). Before you can create a file system on a disk, you must format it and divide it into slices. See "Managing Disks" in the *System Administration Guide, Volume I* for complete information on installing and formatting disks, and dividing disks into slices.

A slice can be used as a raw device for swap space or to hold one UFS file system. A disk can be divided into as many as eight slices. See "Managing Disks" in the *System Administration Guide, Volume I* for a list of customary disk slice assignments.

This example shows disk information for `/dev/dsk/c0t3d0`. Note that the `format` command does not show slice information as part of the available disk selections. It shows the controller, target, and disk number for each disk.

Code Example 32-1 Using the `format` Command

```
jupiter% su
Password:
# format
AVAILABLE DISK SELECTIONS:
 0. c0t0d0 at scsibus0 slave 24
 sd0: <SUN0207 cyl 1254 alt 2 hd 9 sec 36>
 1. c0t3d0 at scsibus0 slave 0: veryloud
 sd3: <SUN0207 cyl 1254 alt 2 hd 9 sec 36>
Specify disk (enter its number): 1
FORMAT MENU:
 disk - select a disk
 type - select (define) a disk type
 partition - select (define) a partition table
 current - describe the current disk
 format - format and analyze the disk
 repair - repair a defective sector
 label - write label to the disk
 analyze - surface analysis
 defect - defect list management
 backup - search for backup labels
 verify - read and display labels
 save - save new disk/partition definitions
 inquiry - show vendor, product and revision
 volname - set 8-character volume name
 quit
format> partition
PARTITION MENU:
 0 - change ‘0’ partition
```

Code Example 32-1 Using the `format` Command *(Continued)*

```
 1 - change '1' partition
 2 - change '2' partition
 3 - change '3' partition
 4 - change '4' partition
 5 - change '5' partition
 6 - change '6' partition
 7 - change '7' partition
 select - select a predefined table
 modify - modify a predefined partition table
 name - name the current table
 print - display the current table
 label - write partition map and label to the disk
 quit
partition> print
Volume: veryloud
Current partition table (original sd3):
Part      Tag   Flag     Cylinders        Size       Blocks
  0       root    wm       0 - 39        14.06MB     (40/0/0)
  1       swap    wu       40 - 199      56.25MB    (160/0/0)
  2     backup    wm       0 - 1150     404.65MB   (1151/0/0)
  3 unassigned    wm       0               0          (0/0/0)
  4 unassigned    wm       0               0          (0/0/0)
  5          -    wm       0             10.20MB     (29/0/0)
  6        usr    wm       200 - 228    121.29MB    (345/0/0)
  7       home    wm       574 - 1150   202.85MB    (577/0/0
partition> quit
format> quit
#
```

If you know the disk and slice number, you can display information for a disk using the `prtvtoc` (print volume table of contents) command. You can specify the volume by specifying any non-zero-size slice defined on the disk (for example, `/dev/rdsk/c0t3d0s2` for all of disk 3, or `/dev/rdsk/c0t3d0s7` for the eighth slice of disk 3). If you know the target number of the disk, but do not know how it is divided into slices, you can show information for the entire disk by specifying either slice 2 or slice 0.

Cylinder Groups

You create a UFS file system on a disk slice, which is divided into one or more areas called *cylinder groups*. A cylinder group is composed of one or more consecutive disk cylinders (the set of tracks on a group of platters that have the same radial distance from the center of the platter). See "Managing Disks" for a complete description of disk geometry.

A *cylinder group map* is created for each cylinder group. The cylinder group map records the block usage and available blocks.

Types of Blocks

Cylinder groups are divided into blocks to control and organize the structure of the files within the cylinder group. Each type of block has a specific function in the file system. A UFS file system has four types of addressable blocks and additional information management disk areas. The four types of blocks are:

- **Boot block –** Used to store information used when booting the system
- **Superblock –** Used to store much of the information about the file system
- **Inode –** Used to store all information about a file except its name
- **Storage or data block –** Used to store data for each file

See Chapter 40, “File System Reference,” for more detailed information about each type of block.

If you want to customize a file system using arguments to the `newfs` command or with the `mkfs` command, see Chapter 40, “File System Reference,” for information about altering these parameters.

Size Restrictions on UFS File Systems

Total Size

The limit on the total size of a file system is 1 Tbyte. A UFS file system can be as big as the slice that holds it.

Maximum File Size

The maximum size for any one file in a UFS file system is 2 Gbytes.

Maximum Number of Files

The maximum number of files per UFS file system is determined by the number of inodes allocated for a file system. The number of inodes depends on how much disk space is allocated for each inode and the total size of the file system. By default, one inode is allocated for each 2 Kbyte of data space. You can change the default allocation using the `-i` option of the `newfs` command.

Logical Block and Fragment Size

Fragments are small logical blocks that are created to save space by reducing unused portions of logical blocks. The maximum logical block and fragment size is 8192 bytes, although fragments are typically less than or equal to the size of logical blocks.

Planning UFS File Systems

After disks are formatted and divided into slices, you need to make a file system on each slice that will contain UFS files. See "Managing Disks" for detailed information on how to format disks and divide them into slices.

When laying out file systems, you need to consider possible conflicting demands. Here are some suggestions:

- Distribute the work load as evenly as possible among different I/O systems and disk drives. Distribute `/home` and `swap` directories evenly across disks.
- Keep pieces of projects or members of groups within the same file system.
- Use as few file systems per disk as possible. On the system (or boot) disk you usually have three slices: `/`, `/usr`, and a swap area. On other disks create one or, at most, two slices. Fewer, roomier slices cause less file fragmentation than many small, over-crowded slices. Higher-capacity tape drives and the ability of `ufsdump` to handle multiple volumes make it easier to back up larger file systems.
- If you have some users who consistently create very small files, consider creating a separate file system with more inodes. However, most sites do not need to be concerned about keeping similar types of user files in the same file system.

See Chapter 33, “Creating File Systems,” for information on default file system parameters as well as prerequisites and procedures for creating new file systems.

UFS Fix-On-Panic

UFS fix-on-panic improves UFS recovery mechanisms by detecting, and where possible, automatically correcting errors on line. Thus:

- Faults that currently require a forced system shutdown have the potential for becoming recoverable failures.
- The entire server is not forced to shut down when only one file system has errors.
- The impact on users is limited to the failing file service only, rather than all services provided by the server.

Fix-on-panic repairs occasional file system inconsistencies that can occur on heavily loaded, multi-user servers. These types of inconsistencies include corrupt file pointers, failed asynchronous writes, and intermittent disk I/O. Repairing file system damage on a single file system does not remove the cause of the error. Therefore, if a second error is detected within a certain amount of time, the system reverts to panic mode as a way to force administrative repair of the problem.

Fix-on-panic does not fix system failures (such as recurrent data faults caused by hardware errors), nor does it increase data integrity. There is some possibility of data loss if fix-on-panic is used on a system that experiences low-level problems over time because the cause of the damage might still be active. For this reason, this feature should only be used when overall system availability needs outweigh the needs of data integrity.

Fix-on-panic can be used on any UFS file system except:

- Root (`/`) or `/usr`
- A file system that contains the system accounting file
- A file system that contains a file being used as `swap`

When fix-on-panic detects a problem, it takes the following actions:

Table 32-4 Fix-On-Panic Overview

From The ...	What Happens
Remote user's view	If a file system is accessed via the network, the user receives a "server not responding" message. Access is denied to this file service until fix-on-panic repairs the error. The user can still access other services and resources on that server.
Local user's view	If a file system is being accessed locally, for example, at the console, the file service process stops responding. All other running processes remain usable. The file service becomes available after fix-on-panic completes its repair of the error.
System's view	Fix-on-panic determines if an online fix is possible. (If not, a forced system shutdown occurs.) Next, it locks the file system and repairs the file system damage with `fsck`. Finally, it unlocks the file system for use. The server's other file systems and services remain available during this process, thus increasing overall availability of services to users.

UFS fix-on-panic requires three components, which are packaged separately:

Table 32-5 Fix-On-Panic Components

Component	Description	Packaged With...
kernel UFS module changes	Enables detection of UFS problems	Solaris 2.5 and greater
`mount_ufs` command	Provides file system unmount, lock, repair, and unlock capabilities	Solaris 2.5 and greater
`ufsd` daemon	Enables `fsck` to be run automatically after the file system is locked by `mount_ufs`	Solstice DiskSuite 4.1 and greater

Mounting and Unmounting

File systems can be attached to the hierarchy of directories available on a system. This process is called *mounting*. A mounted file system is attached to the system directory tree at the specified *mount point* (a directory to which the mounted file system is attached), and becomes available to the system. The root (`/`) file system is always mounted. Any other file system can be connected or disconnected from the root (`/`) file system.

You need to determine:

- Whether the file system should be entered in the `/etc/vfstab` file to be mounted each time the system is booted
- Whether the file system can be appropriately mounted using AutoFS
- Whether the file system will be used only temporarily
- Whether the file system can be mounted from the command line

To mount a file system you need:

- To be root
- A mount point on the local system
- The resource name of the file system to be mounted (for example, `/usr`)

As a general rule, local disk slices should always be included in the `/etc/vfstab` file. Any software from servers, such as OpenWindows software or man pages, and home directories from a server can be either included in the `/etc/vfstab` file or automatically mounted with the `automount` command, depending on the policy at your site.

When you mount a file system, any files or directories in the mount point directory are unavailable as long as the file system is mounted. These files are not permanently affected by the mounting process, and become available again when the file system is unmounted. However, mount directories are usually empty, because you usually do not want to obscure existing files.

Figure 32-2 shows the root (`/`) file system with subdirectories `sbin`, `etc`, and `home`:

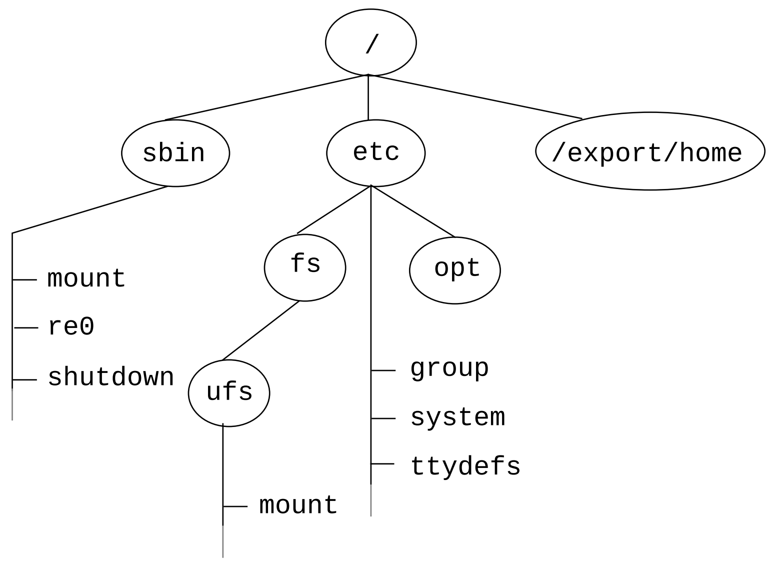


Figure 32-2 A File System

To attach a user's home directory to the empty `/export/home` directory mount point, first create a directory for the new user. For a user named `ignatz`, create a directory in `/export/home` named `ignatz`, giving it the appropriate permissions and ownership. Then mount the file system. When the `ignatz` file system is mounted, all of the files and directories in `/export/home/ignatz` are available, as shown in Figure 32-3. You can also create other user directories in the `/export/home` directory and use those directories as mount points for other user file systems. See Chapter 34, "Mounting and Unmounting File Systems," for information on how to perform these tasks.

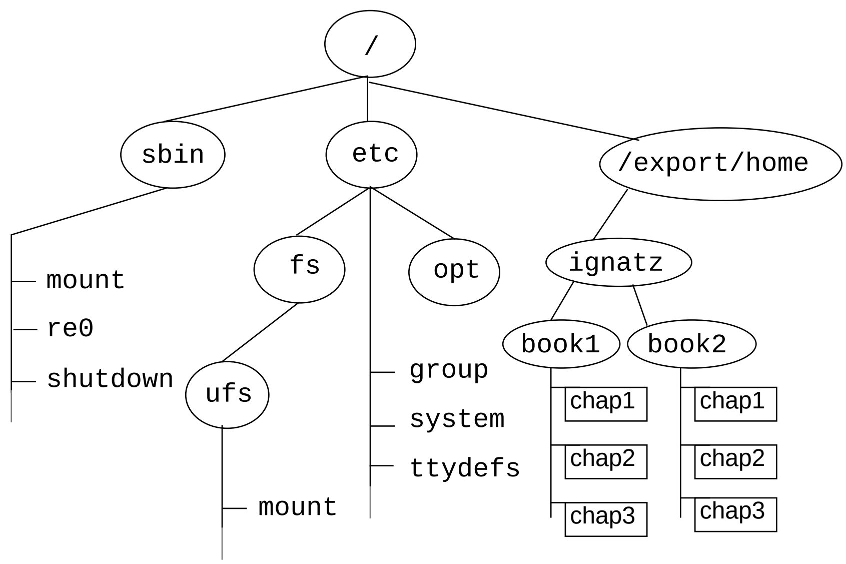


Figure 32-3 Mounting a Home Directory File System

Note – This example illustrates the concept of mounting. Because `/home` is, by default, an AutoFS mount point directory, home directory files would be mounted by AutoFS rather than the `mount` command.

Whenever you mount or unmount a file system, the /etc/mnttab (mount table) file is modified with the list of currently mounted file systems. You can display the contents of the mount table using the cat or more commands, but you cannot edit it as you would the /etc/vfstab file. Here is an example of a mount table file:

```
mars% more /etc/mnttab
/dev/dsk/c0t3d0s0       /       ufs     rw,suid 693186371
/dev/dsk/c0t1d0s6       /usr    ufs     rw,suid 693186371
/proc   /proc   proc    rw,suid 693186371
swap    /tmp    tmpfs   ,dev=0 693186373
swsvr4-50:/export/svr4/openwinV3.jpb1   /usr/openwin    nfs     rw,dev=21c0000  693186443
swsvr4-50:/export/svr4/man      /usr/man        nfs     rw,dev=21c0001  693186447
mars:(pid127)    /nse  nfs    ro,ignore,map=/etc/auto.nse,indirect,dev=21c0002 693186449
mars:(pid127)    /net    nfs     ro,ignore,map=-hosts,indirect,dev=21c0003   693186449
mars:(pid127)       /home   nfs     ro,ignore,map=/etc/auto_home,indirect,dev=21c0004
693186449
neptune:/export/home/neptune  /tmp_mnt/home/neptune  nfs     rw,dev=21c0005  693186673
mars%
```

Unmounting a file system removes it from the file system mount point, and deletes the entry from the /etc/mnttab file. Some file system administration tasks cannot be performed on mounted file systems. You should unmount a file system when:

- It is no longer needed or has been replaced by a file system that contains more current software.
- You check and repair it using the fsck command. See Chapter 39, “Checking File System Integrity,” for more information about the fsck command.

It is a good idea to unmount a file system before doing a complete backup of it. See Chapter 7, “Backing Up Files and File Systems,” for more information about doing backups.

Note – File systems are automatically unmounted as part of the system shutdown procedure.

The Virtual File System Table

The default file system configuration table (the `/etc/vfstab` file) depends on the selections you make when installing system software. You should edit the `/etc/vfstab` file for each system to automatically mount local UFS file systems, essential NFS file systems, and any other appropriate file systems.

The following is an example of the `/etc/vfstab` file. The file system table is an ASCII file. Comment lines begin with #. This example shows an `/etc/vfstab` file for a system with two disks and two NFS file systems mounted.

```
# more /etc/vfstab
#device         device          mount           FS      fsck   mount    mount
#to mount       to fsck         point           type    pass   at boot  options
/dev/dsk/c0t0d0s0 /dev/rdsk/c0t0d0s0 /          ufs     1       no      -
/proc           -               /proc           proc    -       no      -
/dev/dsk/c0t0d0s1 -                -            swap    -       no      -
swap            -               /tmp            tmpfs   -       yes     -
/dev/dsk/c0t0d0s6 /dev/rdsk/c0t0d0s6 /usr       ufs     2       no      -
/dev/dsk/c0t3d0s7 /dev/rdsk/c0t3d0s7 /files7    ufs     2       no      -
cheers:/export/svr4/man.ja5  -  /usr/man        nfs     -       yes     hard
cheers:/export/svr4/openwinV3.ja4 -     /usr/openwin    nfs     -       yes     hard
#
```

Note that, for `/` and `/usr`, the `mount at boot` field value is specified as `no` because these file systems are mounted as part of the boot sequence before the `mountall` command is run. If the `automount` field value is specified as `yes`, the `mountall` program redundantly (and unnecessarily) tries to mount these already mounted file systems. See "Shutting Down and Booting a System" for a description of the booting procedure.

See Chapter 34, “Mounting and Unmounting File Systems,” for descriptions of each of the `/etc/vfstab` fields and information on how to edit and use the file.

Automounting Directories

You can mount file systems shared through NFS by using a method called *automounting*. AutoFS runs in the background and mounts and unmounts remote directories as they are needed. Whenever a user on a client system running AutoFS accesses a remote file or directory available through AutoFS, it mounts the file system on the user's system. The remote file system remains mounted as long as the user remains in the directory and is using a file. If the remote file system is not accessed for a certain period of time, it is automatically unmounted. AutoFS mounts and unmounts file systems as required without any intervention on the part of the user other than changing into or out of a directory.

You can mount some file hierarchies with AutoFS, and others using the `/etc/vfstab` file and the `mount` command. A diskless machine *must* have entries for `/` (root), `/usr`, and `/usr/kvm` in the `/etc/vfstab` file. Because shared file systems should always remain available, do not use AutoFS to mount `/usr/share`.

AutoFS works with the file systems specified in *maps*. These maps can be maintained as NIS, NIS+, or local files.

The AutoFS maps can specify several remote locations for a particular file. This way, if one of the servers is down, AutoFS can try to mount from another machine. You can specify which servers are preferred for each resource in the maps by assigning each server a weighting factor.

AutoFS starts automatically when a system enters run level 3. You can also start it from a command line. See the *NFS Administration Guide* for complete information on how to set up and administer AutoFS.

By default, the SunOS system software automounts `/home`.

Sharing Files From a Server

NFS is a distributed file system that can be used to "tie together" computers that are running different operating systems. For example, systems running DOS can share files with systems running UNIX.

NFS makes the actual physical location of the file system irrelevant to the user. You can use NFS to allow users to see all the relevant files, regardless of location. Instead of placing copies of commonly used files on every system, NFS allows you to place one copy on one system's disk and let all other systems access it across the network. Under NFS, remote file systems are virtually indistinguishable from local ones.

A system becomes an NFS server if it has file systems to *share* or *export* over the network. A server keeps a list of currently exported file systems and their access restrictions (such as read / write or read-only).

You may want to share resources, such as files, directories, or devices from one system on the network (typically, a server) with other systems. For example, you might want to share third-party applications or source files with users on other systems.

When you share a resource, you make it available for mounting by remote systems.

You can share a resource in these ways:

- By using the `share` or `shareall` command
- By adding an entry to the `/etc/dfs/dfstab` (distributed file system table) file

The default `/etc/dfs/dfstab` file shows you the syntax and an example of entries:

```
venus% more /etc/dfs/dfstab

#  place share(1M) commands here for automatic execution
#  on entering init state 3.
#
#  share [-F fstype] [ -o options] [-d "<text>"] <pathname> [resource]
#  .e.g,
#  share  -F nfs  -o rw=engineering  -d "home dirs"  /export/home2
share -F nfs /var/mail

venus%
```

Add one entry to the `/etc/dfs/dfstab` file for each resource that you want to have shared automatically. Each entry must be on a separate line, using this syntax:

`share [-F nfs] [-o` *specific-options*`] [-d "`*description*`"]` *pathname resource*

Table 32-6 describes these variables.

Table 32-6 Variables for `/etc/dfstab` Entry

Option	Description
`-F nfs`	Indicates that the file system type is NFS. If you have only one distributed file system package installed, `nfs` is the default, and you can omit the `-F` option.
`-o` *specific-options*	Regulates how the resource is shared. Specific options, seperated by commas, that can follow the `-o` flag include: • `rw` - Shares *pathname* read / write to all clients (by default), except those that are specified under `ro=`. • `ro` - Shares *pathname* read-only to all clients, except those that are specified under `rw=`. • `ro=`*client*[:*client*] - Shares *pathname* read-only to the listed client machines or netgroup names (overriding `rw`). • `rw=`*client*[:*client*] - Shares *pathname* read / write to the listed client machines or netgroup names (overriding `ro`). • `anon=`*uid* - Lets you specify a different *uid* for ‘‘anonymous’’ users—users whose *uid* is 0, the UID of root on Solaris systems—when accessing *pathname*. By default, anonymous users are mapped to user `nobody`, which has the UID `UID_NOBODY`. User `nobody` has ordinary user privileges, not root privileges. • `root=`*host*[:*host*] - Lets a user from host *host*, whose *uid* is 0, access *pathname* as root; root users from all other hosts become `anon`. If this option is not specified, no user from any host is granted access to *pathname* as root. • `secure -` Shares a resource with additional user authentication required (see *Security, Performance, and Accounting Administration* for more information). • `kerberos` - Shares a resource with Kerberos authentication.
`-d` *description*	Is a comment that describes the resource to be shared. If you use the `-d` option, the description is stored in the `sharetab` file. However, clients do not see the description displayed when they use the `dfshares` command to list the resources shared on that system.
pathname	Is the full name of the resource to be shared, starting at root (/).

You cannot specify both `rw` and `ro` without arguments, and you cannot specify the same client in the `rw=` list and the `ro=` list. If no read/write option is specified, the default is read/write for all clients.

Caution – Granting root access to other hosts has far-reaching security implications; use the `root=` option with extreme caution.

See Chapter 34, "Mounting and Unmounting File Systems," for information on how to share files and file systems. See the *NFS Administration Guide* for a complete description of NFS.

Note – Arguments that accept a client or host list (`ro=`, `rw=`, and `root=`) are guaranteed to work over UDP, but may not work over other transport providers.

Under NFS, a server shares resources it owns so clients can mount them. However, a user who becomes root at a client is denied access as root to mounted remote resources. When a user logged in as root on one host requests access to a remote file shared through NFS, the user's ID is changed from 0 to the user ID of the user name `nobody`. The access rights of user `nobody` are the same as those given to the public for a particular file. For example, if the public has only execute permission for a file, then user `nobody` can execute only that file.

Determining a File System's Type

You can determine a file system's type by using the following:

- The `FS type` field in the file system table (`/etc/vfstab`)
- The `/etc/default/fs` file for local file systems
- The `/etc/dfs/fstypes` file for remote file systems

▼ How to Determine the Type of a File System

Use the `grep` command to find a file system's type in the `/etc/vfstab` file:

Information for the mount point is displayed.

```
$ grep /tmp /etc/vfstab
swap          -             /tmp          tmpfs   -     yes    -
$
```

If vfstab does not have an entry for a file system, use one of the following procedures to determine the file system's type.

▼ How to Determine a Mounted File System's Type

Type `grep` *mount-point* `/etc/mnttab` and press Return.

Information about the mount point is displayed.

```
$ grep /home /etc/mnttab
mars:(pid129)  /home nfs  ro,ignore,map=/etc/auto_home,indirect,dev=21c0004 693606637
neptune:/export/home/neptune  /tmp_mnt/home/neptune  nfs     rw,dev=21c0005  695409833
```

Or

1. **Use the `devnm` command to display the raw device name of the mounted file system.**

```
$ devnm /filesystem
```

In this command,

/filesystem	Is the mounted file system whose raw device name you want to identify.

2. **Become root.**

3. **Use the `fstyp` command to display the file system type.**

```
# fstyp dev/rdsk/device-name
```

In this command,

`dev/rdsk/`*device-name*	Is the name of the raw device containing the file system you want to identify.

Example—Determining a Mounted File System's Type

The following example uses the `devnm` command to identify the name of the device that contains `/usr`, a mounted file system. Next, the `fstyp` command is used to identify what type of file system `/usr` is.

```
$ devnm /usr
/dev/dsk/cot3d0s6 /usr
$ su
# fstyp /dev/rdsk/c0t3d0s6
ufs
```

▼ How to Find Out Which File Systems are Mounted on What Disk Slices

If you know the disk and slice number, you can display information for a disk using the `prtvtoc` (print volume table of contents) command. You can specify the volume by specifying any non-zero-size slice defined on the disk (for example, `/dev/rdsk/c0t3d0s2` for all of disk 3, or `/dev/rdsk/c0t3d0s7` for the eighth slice of disk 3). If you know the target number of the disk, but do not know how it is divided into slices, you can show information for the entire disk by specifying either slice 2 or slice 0.

1. **Become root.**
2. **Display disk and slice information.**

```
# prtvtoc /dev/rdsk/devicename
```

In this command,

dev/rdsk/*device-name*	Is the name of the raw device containing the file system you want to identify.

Example—Finding Out Which File Systems are Mounted on What Disk Slices

In this example, information is displayed for all of disk 3:

```
$ su
Password:
# prtvtoc /dev/rdsk/c0t3d0s2
* /dev/rdsk/c0t3d0s2 (volume "") partition map
*
* Dimensions:
*     512 bytes/sector
*      36 sectors/track
*       9 tracks/cylinder
*     324 sectors/cylinder
*    1272 cylinders
*    1254 accessible cylinders
*
* Flags:
*   1: unmountable
*  10: read-only
*
*                           First     Sector    Last
* Partition  Tag  Flags    Sector    Count     Sector  Mount
Directory
       2      5    01          0    406296    406295
       6      4    00          0    242352    242351
       7      0    00     242352    163944    406295   /files7
```

32

Creating File Systems 33

This chapter describes how to create and preserve UFS, temporary, and loopback file systems on hard disks. This is a list of the step-by-step instructions in this chapter.

Creating a UFS File System

You need to create UFS file systems only occasionally, because the system software automatically creates file systems as part of the installation process. You need to create (or re-create) a UFS file system when you:

- Add or replace disks
- Change the existing partitioning structure
- Do a full restoration on a file system

The following procedure uses the `newfs` command to create UFS file systems. The `newfs` command is a convenient front-end to the `mkfs` command, which creates the new file system. On Solaris 2.x systems, `newfs` parameter defaults, such as tracks per cylinder and sectors per track, are read from the disk label that will contain the new file system, and the options you choose are passed to the `mkfs` command to build the file system.

File System Parameters

To make a new file system on a disk slice, you almost always use the `newfs` command. Table 33-1 shows the default parameters the `newfs` command uses.

Table 33-1 Default Parameters Used by the `newfs` Command

Parameter	**Default Value**
Block size	8 Kbytes
Fragment size	1 Kbyte
Minimum free space	10%
Rotational delay	Device-dependent
Optimization type	Space
Number of inodes	1 for each 2 Kbytes of disk space

Prerequisites

- The disk must be formatted and divided into slices before you can create UFS file systems on it.
- You need to know the raw device file name of the slice that will contain the file system. See Chapter 28, “Administering Disks,” for information on finding disks and disk slice numbers.
- If you are re-creating an existing UFS file system, unmount it.
- You must be root.

▼ How to Create a UFS File System

1. **Make sure you have met the prerequisites listed on page 548.**
2. **Create the file system.**

```
# newfs [-N] [-b size] [-i bytes] /dev/rdsk/device-name
```

In this command,

`-N`	Displays all of the parameters `newfs` would pass to `mkfs`. No file system is created.
`-b` *size*	Sets the file system block size. Default is 8192 blocks.
`-i` *bytes*	Sets the number of bytes per inode. Default is 2048 bytes.
device-name	Is the device that will contain the new file system.

Caution – Be sure you have specified the correct device name for the slice before performing the next step. If you specify the wrong slice, you will erase its contents when the new file system is created.

The system asks for confirmation.

Verification—Creating a UFS File System

Check the new file systems with the `fsck` command.

```
# fsck /dev/rdsk/device-name
```

In this command,

device-name	Is the name of the device containing the new file system.

This utility checks the consistency of the new file system, reports problems it finds, and prompts you before repairing the problems. See Chapter 39, "Checking File System Integrity," for more information on `fsck`.

Example—Creating a UFS File System

This example creates a file system on `/dev/rdsk/c0t3d0s7`.

```
# newfs /dev/rdsk/c0t3d0s7
newfs: construct a new file system /dev/rdsk/c0t3d0s7 (y/n)? y
/dev/rdsk/c0t3d0s7:     163944 sectors in 506 cylinders of 9 tracks, 36 sectors
        83.9MB in 32 cyl groups (16 c/g, 2.65MB/g, 1216 i/g)
super-block backups (for fsck -b #) at:
 32, 5264, 10496, 15728, 20960, 26192, 31424, 36656, 41888,
 47120, 52352, 57584, 62816, 68048, 73280, 78512, 82976, 88208,
 93440, 98672, 103904, 109136, 114368, 119600, 124832, 130064, 135296,
 140528, 145760, 150992, 156224, 161456,
```

Where to Go From Here

If You Want to Make the File System Available By ...	Then ...
Mounting the file system with the `mount` command	Go to "How to Mount a UFS File System" on page 566.
Creating an entry in `/etc/vfstab` to make the system automatically mount the file system when it boots	Go to "How to Add an Entry in the /etc/vfstab File" on page 562.

Creating a Temporary File System (TMPFS)

The most common use for a TMPFS is the `/tmp` directory. By default, the `/tmp` directory for the SunOS system software is a TMPFS, and an entry is provided for it in the default `/etc/vfstab` file.

By default, files in TMPFS directories do not survive across reboots or unmounts, including file systems under `/var/tmp`. Note that the `vi -r` command expects to find preserved files in the `/var/tmp` directory after a system is rebooted. You can preserve a TMPFS across reboots or unmounts by adding an entry to the `/etc/vfstab` file.

If you create multiple temporary file systems, be aware that they all use the same system resources. Files created under one TMPFS directory use up the space available for any other TMPFS, unless you limit TMPFS sizes using the `-o` option of the `mount` command.

▼ How to Create a Temporary File System

1. **Become root.**
2. **If necessary, create the directory where you want to mount the TMPFS and set permissions and ownership as necessary.**
3. **Create a temporary file system.**

```
# mount -F tmpfs swap mount-point
```

In this command,

`-F tmpfs swap`	Is a TMPFS file system.
mount-point	Is the directory where the TMPFS file system is mounted.

Verification—Creating a Temporary File System

To verify that a temporary file system called /visitors has been created and mounted in /usr/tmp, use the mount and ls commands.

```
$ mount
/ on /dev/md/dsk/d0 read/write/setuid on Mon Oct 10 08:36:49 1994
/usr on /dev/md/dsk/d1 read/write/setuid on Mon Oct 10 08:36:49 1994
/proc on /proc read/write/setuid on Mon Oct 10 08:36:49 1994
/dev/fd/ on fd read/write/setuid on Mon Oct 10 08:36:49 1994
/var on /dev/md/dsk/d2 read/write/setuid on Mon Oct 10 08:36:49 1994
/tmp on swap read/write/setuid on Mon Oct 10 08:36:49 1994
/opt on /dev/md/dsk/d3
/scratch on /dev/rdsk/c0t3d0s7 read/write/setuid on Mon Oct 10 08:36:49 1994
/var/spool/news on /dev/dsk/c2t1d0s2 read/write/setuid on Mon Oct 10 08:36:49 1994
/home/ganymede on site:/export/home/ganymede read/write/remote on Mon Oct 10 08:36:49 1994
/visitors on /usr/tmp on Mon Oct 10 08:36:49 1994
$ ls /usr/tmp
NTa0006h    ps_data      visitors
```

Example—Creating a Temporary File System

This example creates a new directory, /export/reports, and mounts a temporary file system at that point.

```
# mkdir /export/reports
# chmod 777 /export/reports
# mount -F tmpfs swap /export/reports
```

See the tmpfs(7FS) manual page for more information.

▼ How to Preserve a Temporary File System

You can preserve a TMPFS across reboots or unmounts by adding an entry to the `/etc/vfstab` file. Edit the file to add an entry like this, separating each field with a Tab:

```
swap	-	mount-point	tmpfs	-	yes	-
```

In this command,

mount-point	Is the default mount point directory.

For more information about editing the `/etc/vfstab` file, see Chapter 34, "Mounting and Unmounting File Systems.

Creating a Loopback File System (LOFS)

A LOFS is a virtual file system which provides an alternate path to an existing file system. When other file systems are mounted onto a loopback file system, the original file system does not change.

You can preserve a LOFS across reboots or unmounts by adding an entry to the `/etc/vfstab` file.

Caution – Be careful when creating loopback mounts. Because these are virtual file systems, the potential for confusing both users and applications is enormous.

▼ How to Create a Loopback File System

1. **Become root.**
2. **Create the directory where you want to mount the loopback file system and give it the appropriate permissions and ownership.**

3. **Create a loopback file system.**

```
# mount -F lofs lo-directory mount-point
```

In this command,

lo-directory	Is the file system to be mounted at the loopback mount point.
mount-point	Is the directory where the LOFS is mounted.

Example—Creating a Loopback File System

This example shows how to use a new loopback file system with the `chroot` command to provide a complete virtual file system view to a process or family of processes:

```
# mount -F lofs / /tmp/newroot
# chroot /tmp/newroot command
```

See `lofs(7FS)` or `mount(1M)` for more information.

▼ How to Preserve a Loopback File System

You can preserve a loopback file system across reboots or unmounts by adding an entry to the `/etc/vfstab` file. Edit the file to add an entry like this at the end of the file, separating each field with a tab.

```
/	-	mount-point	lofs	-	yes	-
```

In this command,

mount-point	Is the default mount point directory (for example, `/tmp/newroot`).

Caution – Make sure the loopback entry is the last entry in the `/etc/vfstab` file. Otherwise, if the `/etc/vfstab` entry for the loopback file system precedes the file systems to be included in it, the loopback file system cannot be created.

For more information about the `/etc/vfstab` file, refer to Chapter 34, "Mounting and Unmounting File Systems."

Mounting and Unmounting File Systems 34

This chapter describes how to mount and unmount file systems individually through the `/etc/vfstab` file. This is a list of the step-by-step instructions in this chapter.

Mounting File Systems

After you create a file system, you need to make it available. You make file systems available by mounting them. A mounted file system is attached to the system directory tree at the specified mount point, and becomes available to the system. The root (`/`) file system is always mounted. Any other file system can be connected or disconnected from the root (`/`) file system.

You can mount a local file system in these ways:

- By creating an entry in the `/etc/vfstab` (virtual file system table) file. The `/etc/vfstab` file contains a list of file systems that are automatically mounted when the system is booted to multiuser state.
- From the command line by using the `mount` command.

When file systems are shared from a server, a client can mount them as NFS file systems in any of these three ways:

- By adding an entry to the `/etc/vfstab` file so that the file system is automatically mounted when the system is booted to multiuser state.
- By using AutoFS to automatically mount or unmount the file system when a user changes into (`mount`) or out of (`umount`) the directory. See the *NFS Administration Guide* for information about AutoFS.
- By using the `mount` command from the command line.

CD-ROMs containing file systems are mounted when the CD-ROM is inserted. Diskettes containing file systems are mounted by running the `volcheck(1)` command.

Prerequisites

The prerequisites to mount file systems are:

- You must be root.
- There must be a mount point on the local system to mount a file system. A mount point is a directory to which the mounted file system is attached.

Verifying a Mounted File System

To verify that you mounted a file system or a number of file systems, look at the output from the mount command. This is described in "How to See Which File Systems Are Mounted" on page 562.

Commands Used to Mount and Unmount File Systems

Table 34-1 lists the commands in the /usr/sbin directory that you use to mount and unmount file systems.

Table 34-1 Commands for Mounting and Unmounting File Systems

Command	Description
mount(1M)	Mounts file systems and remote resources.
mountall(1M)	Mounts all file systems specified in the /etc/vfstab file. The mountall command is run automatically when entering multiuser run states.
umount(1M)	Unmounts file systems and remote resources.
umountall(1M)	Unmounts all file systems specified in the /etc/vfstab file.

The mount commands will not mount a read/write file system that has inconsistencies. If you receive an error message from the mount or mountall command, you may need to check the file system. See Chapter 39, "Checking File System Integrity," for information on how to check the file system.

The umount commands will not unmount a file system that is busy. A file system is considered busy if a user is in a directory in the file system, or if a program has a file open in that file system.

General Mount Options

Table 34-2 describes the general mount options that you can specify with the `-o` option of the `mount` command. If you specify multiple options, separate them with commas (no spaces). For example, `-o ro,nosuid`.

Table 34-2 General Mount Options

Option	File System	Description
`rw | ro`		Specifies read / write or read-only. If you do not specify this option, the default is read / write.
`nosuid`		Disallows `setuid` execution and prevents devices on the file system from being opened. The default is to allow `setuid` execution and allow devices to be opened.
`remount`		With `rw`, remounts a file system with read / write access.
`f`	UFS, S5FS	Fakes an entry in `/etc/mnttab`, but doesn't really mount any file systems.
`n`	UFS, S5FS	Mounts the file system without making an entry in `/etc/mnttab.`
`bg | fg`	NFS	If the first attempt fails, retries in the background (`bg`) or in the foreground (`fg`). The default is `fg`.
`soft | hard`	NFS	Specifies the procedure if the server does not respond. `Soft` indicates that an error is returned. `Hard` indicates that the retry request is continued until the server responds. The default is `hard`.
`intr | nointr`	NFS	Specifies whether keyboard interrupts are allowed to kill a process hung while waiting for a response on hard-mounted file systems. The default is `intr` (interrupts allowed).
`retry=`*n*	NFS	Retries the mount operation when it fails. *n* is the number of times to retry.

Field Descriptions of the `/etc/vfstab` File

An entry in the `/etc/vfstab` file has seven fields, which are described in Table 34-3.

Table 34-3 Field Descriptions of the `/etc/vfstab` File

Field Name	Description
`device to mount`	The device to mount can be: • The block special device for local UFS file systems (for example, `/dev/dsk/c0t0d0s0`). • The resource name for remote file systems (for example, `myserver:/export/home` for an NFS file system). For more information on remote file systems, see *NFS Administration Guide*. • The name of the slice on which to swap (for example, `/dev/dsk/c0t3d0s1).` • The `/proc` directory and `proc` file system type.
`device to fsck`	The raw (character) special device that corresponds to the file system identified by the `device to mount` field (for example, `/dev/rdsk/c0t0d0s0`). This determines the raw interface that is used by `fsck`. Use a dash (`-`) when there is no applicable device, such as for a read-only file system or a remote file system.
`mount point`	The default mount point directory (for example, `/usr` for `/dev/dsk/c0t0d0s6`).
`FS type`	The type of file system identified by the `device to mount` field.
`fsck pass`	The pass number used by `fsck` to decide whether to check a file system. When the field contains a dash (`-`), the file system is not checked. When the field contains a value greater than zero, the file system is checked; non-UFS file systems with a zero value for `fsck pass` are checked. For UFS file systems only, when the field contains a zero, the file system is not checked. When `fsck` is run on multiple UFS file systems that have `fsck pass` values greater than one and the preen option (`-o p`) is used, `fsck` automatically checks the file systems on different disks in parallel to maximize efficiency. When the field contains a value of `1`, the file system is checked sequentially. Otherwise, the value of the pass number does not have any effect. In SunOS system software, the `fsck pass` field does not explicitly specify the order in which file systems are checked.
`mount at boot`	`yes` or `no` for whether the file system should be automatically mounted by `mountall` when the system is booted. Note that this field has nothing to do with AutoFS.
`mount options`	A list of comma-separated options (with no spaces) that are used in mounting the file system. Use a dash (`-`) to show no options. See `mount(1M)` for a list of the available options.

Note – You must have an entry in each field in the /etc/vfstab file. If there is no value for the field, be sure to enter a dash (-).

▼ How to See Which File Systems Are Mounted

To see which file systems are mounted:

```
$ mount
```

Example—Seeing Which File Systems Are Mounted

```
mars$ mount
/ on /dev/dsk/c0t3d0s0 read/write on Tue Dec 24 12:29:22 1993
/usr on /dev/dsk/c0t1d0s6 read/write on Tue Dec 24 12:29:22 1993
/proc on /proc read/write on Tue Dec 24 12:29:22 1993
/tmp on swap read/write on Tue Dec 24 12:29:24 1993
/export/home on /dev/dsk/c0t3d0s7 read/write on Tue Dec 24 12:29:22 1993
/usr/openwin on pluto:/export/openwin read/write/remote on Tue Dec 24 12:30:32 1993
/usr/man on pluto:/export/man read/write/remote on Tue Dec 24 12:30:35 1993
/cdrom/ptf_1_2a on /vol/dev/dsk/c1t5/ptf_1_2a read only on Tue Dec 24 12:30:39
mars$
```

▼ How to Add an Entry in the /etc/vfstab File

1. **Make sure you have met the prerequisites listed on page 558.**
2. **Edit the /etc/vfstab file and add an entry.**
   See Table 34-3 on page 561 for detailed information about the /etc/vfstab field entries. Make sure you:
   - Separate each field with white space (a space or a Tab).
   - Enter a dash (-) if a field has no contents.
3. **Save the changes.**

Examples—Adding an Entry in the `/etc/vfstab` *File*

```
   #device          device           mount            FS      fsck    mount      mount
   #to mount        to fsck          point            type    pass    at boot    options
   #
❶  /dev/dsk/c0t3d0s7 /dev/rdsk/c0t3d0s7 /files1    ufs     2       yes        -
❷  pluto:/export/man   -           /usr/man         nfs     -       yes        ro,soft
❸  /              -             /tmp/newroot    lofs    -       yes        -
```

❶ This example mounts the disk slice `/dev/dsk/c0t3d0s7` as a UFS file system attached to the mount point directory `/files1` with the default mount options (read / write). It specifies the raw character device `/dev/rdsk/c0t3d0s7` as the `device to fsck`. The `fsck pass` value of `2` means that the file system will be checked, but not sequentially.

❷ This example mounts the directory `/export/man` from the system `pluto` as an NFS file system on mount point `/usr/man`. It does not specify a `device to fsck` or a `fsck pass` for NFS file systems. In this example, `mount options` are `ro` (read-only) and `soft`. For greater reliability, specify the `hard` mount option for read / write NFS file systems.

❸ This example mounts the root (`/`) file system on a loopback mount point named `/tmp/newroot`. It specifies `yes` for `mount at boot`, no `device to fsck`, and no `fsck pass` number. Loopback file systems must always be mounted after the file systems used to make up the loopback file system. Be sure that the loopback entry is the last entry in the `/etc/vfstab` file so that it follows the entries that it depends on.

▼ How to Mount Multiple File Systems Listed in the /etc/vfstab File

1. **Make sure you have met the prerequisites listed on page 558.**

2. **Mount the file systems listed in the /etc/vfstab file.**

```
# mountall [-l | -r] [-F fstype]
```

If no options are specified, all file systems listed in the /etc/vfstab file with yes in the mount at boot field are mounted.

In this command,

-l	Mounts all the local file systems listed in the /etc/vfstab file with yes in the mount at boot field.
-r	Mounts all the remote file systems listed in the /etc/vfstab file with yes in the mount at boot field.
-F *fstype*	Mounts all file systems of the type *fstype* listed in the /etc/vfstab file with yes in the mount at boot field.

All the file systems with a device to fsck entry are checked and fixed, if necessary, before mounting.

Examples—Mounting Multiple File Systems Listed in the /etc/vfstab File

This example shows the messages displayed if file systems are already mounted when you use the mountall command.

```
# mountall
mount: /tmp already mounted
nfs mount: mount: /usr/openwin: Device busy
nfs mount: mount: /usr/man: Device busy
```

This example mounts all the local systems listed in the `/etc/vfstab` file.

```
# mountall -l
# mount
/ on /dev/dsk/c0t3d0s0 read/write on Tue Dec 24 12:29:22 1993
/usr on /dev/dsk/c0t1d0s6 read/write on Tue Dec 24 12:29:22 1993
/proc on /proc read/write on Tue Dec 24 12:29:22 1993
/tmp on swap read/write on Mon Dec 30 12:37:33 1993
```

This example mounts all the remote file systems listed in the `/etc/vfstab` file.

```
# mountall -r
# mount
/ on /dev/dsk/c0t3d0s0 read/write on Tue Dec 24 12:29:22 1993
/usr on /dev/dsk/c0t1d0s6 read/write on Tue Dec 24 12:29:22 1993
/proc on /proc read/write on Tue Dec 24 12:29:22 1993
/tmp_mnt/home/neptune on neptune:/export/home/neptune read/write/remote on Mon Dec 30
12:27:41 1993
/usr/openwin on pluto:/export/openwin read/write/remote on Mon Dec 30 12:37:53 1993
/usr/man on pluto:/export/man read/write/remote on Mon Dec 30 12:37:55 1993
```

This example mounts all the NFS file systems listed in the `/etc/vfstab` file:

```
# mountall -F nfs
# mount
/ on /dev/dsk/c0t3d0s0 read/write on Tue Dec 24 12:29:22 1993
/usr on /dev/dsk/c0t1d0s6 read/write on Tue Dec 24 12:29:22 1993
/proc on /proc read/write on Tue Dec 24 12:29:22 1993
/tmp_mnt/home/neptune on neptune:/export/home/neptune read/write/remote on Mon Dec 30
12:27:41 1993
/usr/openwin on pluto:/export/openwin read/write/remote on Mon Dec 30 12:49:09 1993
/usr/man on pluto:/export/man read/write/remote on Mon Dec 30 12:49:11 1993
```

▼ How to Mount a File System Listed in the /etc/vfstab File

1. **Make sure you have met the prerequisites listed on page 558.**
2. **Mount a file system listed in the /etc/vfstab file.**

```
# mount mount-point
```

In this command,

mount-point	Is an entry in the mount point or device to mount field in the /etc/vfstab file. It is usually easier to specify the mount point.

Example—Mounting a File System Listed in the /etc/vfstab *File*

This example mounts the /usr/openwin file system listed in the /etc/vfstab file.

```
# mount /usr/openwin
```

▼ How to Mount a UFS File System

1. **Make sure you have met the prerequisites listed on page 558.**
2. **Mount the UFS file system.**

```
# mount [-o mount-options] /dev/dsk/device-name mount-point
```

In this command,

-o *mount-options*	Specifies mount options that you can use to mount a UFS file system. See Table 34-2 for the list of general mount options or mount_ufs(1M) for a complete list of options.

`/dev/dsk/`*device-name*	Is the special block device file for the disk slice holding the file system (for example, `/dev/dsk/c0t3d0s7`). See Chapter 1, “Planning File Systems,” for information about how to find out disk device names.
mount-point	Is the directory where the file system is mounted.

Example—Mounting a UFS File System

In this example, `/dev/dsk/c0t3d0s7` is mounted on the `/files1` directory.

```
# mount /dev/dsk/c0t3d0s7 /files1
```

▼ How to Mount an NFS File System

1. **Make sure you have met the prerequisites listed on page 558.**
2. **Make sure the file system is available from a server.**

   To mount an NFS file system, it must be made available from the server system. The `share(1M)` command creates a list of file systems in the file `/etc/dfs/sharetab` than can be shared across the network. See [cross-reference to be supplied] for information on how to export file systems.
3. **Mount the NFS file system.**

```
# mount -F nfs [-o mount-options] server:/directory mount-point
```

In this command,

`-o` *mount-options*	Specifies mount options that you can use to mount a NFS file system. See Table 34-2 for the list of general mount options or `mount_nfs(1M)` for a complete list of options.
server`:/`*directory*	Is the server's host name on which the file system is mounted, and the name of the file system to mount.
mount-point	Is the directory where the file system is mounted.

Example—Mounting an NFS File System

In this example, packages from the server `pluto` in the directory `/export/packages` are mounted on `/mnt`.

```
# mount -F nfs pluto:/export/packages /mnt
```

x86: How to Mount an S5FS File System

1. **Make sure you have met the prerequisites listed on page 558.**
2. **Mount the S5FS file system.**

```
# mount -F s5fs [-o mount-options] /dev/dsk/device_name mount-point
```

In this command,

`-o` *mount-options*	Specifies mount options that you can use to mount a S5FS file system. See Table 34-2 for the list of common mount options or `mount_s5fs(1M)` for a complete list of options.
`/dev/dsk/`*device-name*	Is the device name of the disk slice holding the file system (for example, `/dev/dsk/c0t3d0s7`). See Chapter 1, “Planning File Systems,” for information about how to find out disk device names.
mount-point	Is the directory where the file system is mounted.

Example—Mounting an S5FS File System

In this example, `/dev/dsk/c0t3d0s7` is mounted on the `/files1` directory.

```
# mount -F s5fs /dev/dsk/c0t3d0s7 /files1
```

x86: How to Mount a PCFS (DOS) File System From a Hard Disk

Use the following procedure to mount a PCFS (DOS) file system from a hard disk.

1. **Make sure you have met the prerequisites listed on page 558.**
2. **Mount the PCFS file system.**

```
# mount -F pcfs [-o rw | ro] /dev/dsk/device-name:logical-drive mount-point
```

In this command,

`-o rw	ro`	Specifies that you can mount a PCFS file system read/write or read-only. If you do not specify this option, the default is read/write.
`/dev/dsk/`*device-name*	Is the device name of the whole disk (for example, `/dev/dsk/c0t0d0p0`).
logical-drive	Specifies either the DOS logical drive letter (c through z) or a drive number 1 through 24. Drive c is equivalent to drive 1 and represents the Primary DOS slice on the drive; all other letters or numbers represent DOS logical drives within the Extended DOS slice.
mount-point	Is the directory where the file system is mounted.

Note that the *device-name* and *logical-drive* must be separated by a colon.

Examples—Mounting a PCFS (DOS) File System From a Hard Disk

In this example, the logical drive in the Primary DOS slice is mounted on the `/pcfs/c` directory.

```
# mount -F pcfs /dev/dsk/c0t0d0p0:c /pcfs/c
```

In this example, the first logical drive in the Extended DOS slice on the disk is mounted read-only on /pcfs/d.

```
# mount -F pcfs -o ro /dev/dsk/c0t0d0p0:2 /pcfs/d
```

Unmounting File Systems

Unmounting a file system removes it from the file system mount point, and deletes the entry from the /etc/mnttab file. Some file system administration tasks cannot be performed on mounted file systems. You should unmount a file system when:

- It is no longer needed or has been replaced by a file system that contains more current software.
- You check and repair it using the fsck command. See Chapter 39, “Checking File System Integrity,” for more information about the fsck command.

It is a good idea to unmount a file system before doing a complete backup of it. See Chapter 42, “Backing Up Files and File Systems,” for more information about doing backups.

Note – File systems are automatically unmounted as part of the system shutdown procedure.

Prerequisites

The prerequisites to unmount file systems are:

- You must be root.
- A file system must be available for unmounting. You cannot umount a file system that is busy. A file system is considered busy if a user is in a directory in the file system, or if a program has a file open in that file system. You can make a file system available for unmounting by:
  - Changing to a directory in a different file system.
  - Logging out of the system.

- Using the `fuser` command to list all processes accessing the file system and to stop them if necessary. See "How to Stop All Processes for a File System" on page 572 for more details.

Notify users if you need to unmount a file system they are using.

Verifying an Unmounted File System

To verify that you unmounted a file system or a number of file systems, look at the output from the `mount` command. This is described in "How to See Which File Systems Are Mounted" on page 562.

▼ How to Stop All Processes for a File System

1. **Become root.**

2. **List all the processes that are using the file system, so you know which processes you are going to stop.**

```
# fuser -c [-u] mount-point
```

In this command,

-c	Reports on files that are mount points for file systems, and any files within those mounted file systems.
`-u`	Displays the user login name for each process ID.
mount-point	Is the name of the file system for which you want to stop processes.

3. **Stop all processes for the file system.**

Note – You should not stop a user's processes without warning.

```
# fuser -c -k mount-point
```

A `SIGKILL` is sent to each process using the file system.

Verification—Stopping All Processes for a File System

Verify that there are no processes using the file system.

```
# fuser -c mount-point
```

Example—Stopping All Processes for a File System

In this example, process `4006c` that is using the `/export/home` file system is stopped.

```
# fuser -c /export/home
/export/home:     4006c
# fuser -c -k /export/home
/export/home:     4006c
# fuser -c /export/home
/export/home:
```

▼ How to Unmount a File System

Use the following procedure to unmount a file system (except / or /usr):

Note – The root (/) and /usr UFS file systems are special cases. The root (/) file system can be unmounted only during a shutdown, since the system needs root to function.

---

1. **Make sure you have met the prerequisites listed on page 571.**
2. **Unmount the file system.**

```
# umount mount-point
```

In this command,

mount-point	Is the name of the file system that you want to unmount. This can either be the directory name where the file system is mounted, the device name path of the file system, the resource for an NFS file system, or the loopback directory for LOFS file systems.

Examples—Unmounting a File System

In this example, a local home file system is unmounted.

```
# umount /export/home
```

In this example, the file system on slice 7 is unmounted.

```
# umount /dev/dsk/c0t0d0s7
```

▼ How to Unmount File Systems Listed in the /etc/vfstab File

Use the following procedure to unmount all the file systems listed in the /etc/vfstab file, except for the /, /proc, /var, and /usr file systems.

1. **Make sure you have met the prerequisites listed on page 571.**
2. **Unmount all the file systems listed in the /etc/vfstab file.**

```
# umountall
```

   All systems that can be unmounted are unmounted. File systems that are busy are not unmounted.

3. **For the file systems that were busy and not unmounted, make them available to be unmounted as described in “How to Stop All Processes for a File System” on page 572.**
4. **Repeat Step 2 as needed until all file systems are unmounted.**

Setting Up and Maintaining the Cache File System

35

The Cache File System (CacheFS) is a general purpose file system caching mechanism that improves NFS server performance and scalability by reducing server and network load. Designed as a layered file system, CacheFS provides the ability to cache one file system on another. In an NFS environment, CacheFS increases the client per server ratio, reduces server and network loads and improves performance for clients on slow links, such as Point-to-Point Protocol (PPP).

This is a list of the step-by-step instructions in this chapter.

How CacheFS Works

You create a cache, using the `cfsadmin(1M)` command, on the client so that file systems you specify to be mounted in the cache can be accessed by the user locally instead of across the network. Figure 35-1 on page 579 shows the relationship of the components involved in using CacheFS. The back file system is the file system that you specify to be mounted in the cache. Typically, this is an NFS or HSFS (High Sierra File System) file system. When the user attempts to access files that are part of the back file system, those files are placed in the cache. So the cache does not get filled until the user requests access to a file or files. To the user, the initial request to access a file may seem slow, but subsequent uses of the same file will be faster.

Note – You can mount only file systems that are shared. See the `share(1M)`command. Also, there is no performance gain in caching a local UFS disk file system.

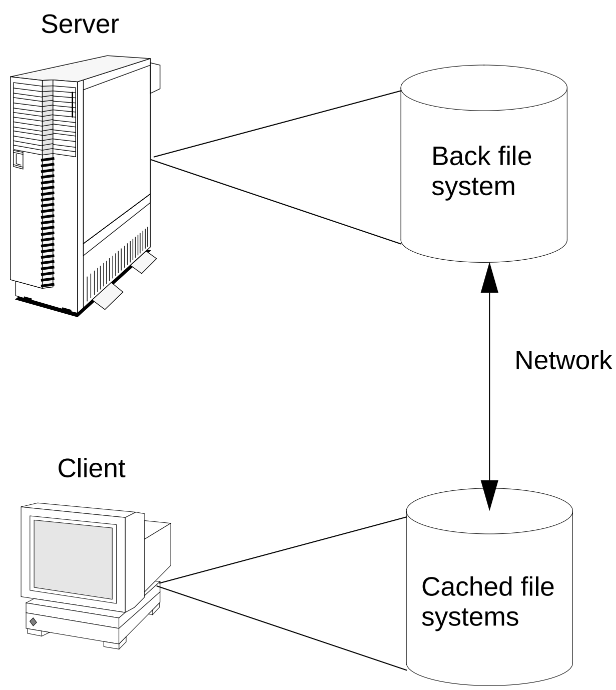


Figure 35-1 How CacheFS Works

Setting Up a Cached File System

Table 35-1 shows the steps involved to set up a cached file system.

Table 35-1 Task Map for Setting Up a Cached File System

Activity	Description	For Instructions, Go To	
Create a Cache	Use the `cfsadmin` command to create a cache	▼ How to Create a Cache	page 581
Mount File Systems in the Cache	Mount a file system in a cache by using the `mount` command	▼ How to Specify a File System to be Mounted in a Cache With mount(1M)	page 583
	or		
	Cache a file system by editing the `/etc/vfstab` file	▼ How to Mount a File System in a Cache Editing the /etc/vfstab File	page 586
	or		
	Cache a file system by using AutoFS	▼ How to Mount a File System in a Cache With AutoFS	page 587

How to Create a Cache

1. **Become root.**

2. **Create a cache using the** `-c` **option of the** `cfsadmin(1M)` **command.**

```
# cfsadmin -c cache-directory
```

In this command,

cache-directory	Is the name of the directory where the cache resides. For more information about the `cfsadmin` command, see the man page.

Note – After you have created the cache, do not perform any operations within the cache directory itself. This causes conflicts within the CacheFS software.

Example—Creating a Cache

The following example creates a cache in the `/local/mycache` directory by using the default cache parameter values.

```
# cfsadmin -c /local/mycache
#
```

Verification—Creating a Cache

If you performed the steps in this procedure successfully, you will receive a system prompt after each step, with no system response, as shown in the above example.

Specifying a File System to Be Mounted in the Cache

You specify file systems to be mounted in the cache so that users can locally access files in the file system you've specified. The files do not actually get placed in the cache until the user accesses the files.

There are three ways to specify file systems to be cached:

- Using the `mount(1M)` command – you need to use the `mount(1M)`command every time the system reboots in order to access the same file system.
- Editing the `/etc/vfstab` file – you only need to do this once. The `/etc/vfstab` file will remain unaltered after the system reboots.
- Using AutoFS – you only need to do this once. AutoFS maps remain unaltered after the system reboots.

Choose the method of mounting file systems you are currently familiar with.

Note – Caching of the `/` (root) and `/usr` file systems is not supported in CacheFS. To cache the `/` and `/usr` file systems, you must purchase the Solstice AutoClient product. For more information about the AutoClient product, see the *Solstice AutoClient Administration Guide.*

▼ How to Specify a File System to be Mounted in a Cache With `mount(1M)`

1. **Become root.**
2. **Create a mount point.**
   The mount point allows user access to the file system specified under that mount point. You can create the mount point from anywhere. The CacheFS options used with the `mount(1M)`command, as shown in the next step, will determine that the mount point you created will be cached in the cache directory you specified.
3. **Mount a file system in a cache with the `mount(1M)` command.**

```
# mount -F cachefs -o backfstype=fstype,cachedir=cache-directory[,options] back-filesystem  mount-point
```

In this command,

fstype	Is the file system type of the back file system, for example `nfs`.
cache-directory	Is the name of the directory where the cache resides. This is the same name you specified when you created the cache in "How to Create a Cache" on page 581.
options	Specifies other mount options that you can include when mounting a file system in a cache. See `mount_cachefs(1M)` for a list of CacheFS mount options.
back-filesystem	Is the mount point of the back file system to cache. If the back file system is an NFS file system, you must specify the host name of the server from which you are mounting the file system and the name of the file system to cache (separated by a colon), for example, *merlin: /usr/openwin*
mount-point	Is the directory where the file system is mounted.

Examples—Specifying a File System to be Mounted in a Cache With `mount(1M)`

The following example creates the mount point `/docs`, and mounts the NFS file system `merlin:/docs` as a cached file system named `/docs` in the cache named `/local/mycache`.

```
# mkdir /docs
# mount -F cachefs -o backfstype=nfs,cachedir=/local/mycache merlin:/docs /docs
```

The following example makes a CD-ROM (HSFS file system) available as a cached file system named `/docs`. Because you cannot write to the CD-ROM, the `ro` argument is specified to make the cached file system read-only. You must specify the `backpath` argument because Volume Management automatically mounts the CD-ROM when it is inserted. The mount point is in the `/cdrom` directory and is determined by the name of the CD-ROM. The special device to mount is the same as the value for the `backpath` argument.

```
# mount -F cachefs -o backfstype=hsfs,cachedir=/local/mycache,ro,backpath=/cdrom/cdrom_name \
/cdrom/cdrom_name /docs
```

The following example uses the `demandconst` option to specify consistency checking on demand for the NFS cached file system `/docs`, whose back file system is `merlin:/docs`. See "Consistency Checking of Cached File Systems With the Back File System" on page 611 for more information.

```
# mount -F cachefs -o backfstype=nfs,cachedir=/local/mycache,demandconst merlin:/docs /docs
```

Verification—Specifying a File System to be Mounted in a Cache with `mount(1M)`

When you mount the file system in the cache using the `mount` command, you will receive a prompt with no system response if the file system was mounted correctly. Or, to verify that the cache you created was actually mounted, use the `cachefsstat(1M)`command, as follows:

```
# cachefsstat mount-point
```

For example:

```
# cachefsstat /docs
/docs
             cache hit rate:   100% (0 hits, 0 misses)
         consistency checks:      1 (1 pass, 0 fail)
                   modifies:      0
         garbage collection:      0
```

The mount point is the cached file system you created. For more information about the `cachefsstat` command, see “How to View Cache Statistics” on page 609.

If the file system was not mounted in the cache, you will receive an error message similar to the following:

```
# cachefsstat mount-point
cachefsstat: mount-point: not a cachefs mountpoint
```

▼ How to Mount a File System in a Cache Editing the `/etc/vfstab` File

1. **Become root.**

2. **Using an editor, specify the file systems to be mounted in the `/etc/vfstab` file:**

```
#device                    device             mount        FS       fsck mount   mount
#to mount                  to fsck            point        type     pass at boot  options
#
/dev/dsk/c0t1d0s0          /dev/rdsk/c0t1d0s0 /cache       ufs      2    yes     -
merlin:/docs               /local/mycache     /usr/local   cachefs  3    yes   backfstype=nfs,cachedir=/local/mycache
```

   The last line represents the new entry.

3. **Mount the cached file system using the `mount` command, as follows:**

```
# mount  /docs
```

   or reboot.

   The above example shows the file system `/usr/local` mounted in the cache directory.

How to Mount a File System in a Cache With AutoFS

Enable caching of automounts by specifying the *fstype=cachefs* mount option in your automount map. Note that CacheFS mount options (for example, *backfstype* and *cachedir*) are also specified in the automount map. See `automount(1M)` the for details on automount maps. Also see the *NFS Administration Guide.*

1. **Become root.**
2. **Using an editor, add the following line to the** `auto_direct` **map:**

```
/docs -fstype=cachefs,cachedir=/local/mycache,backfstype=nfs merlin:/docs
```

3. **Using an editor, add the following line to the** `auto_master` **map:**

```
/-
```

The `/-` entry is a pointer to check the `auto_direct` map.

4. **Reboot the system.**

Verification—Mounting a File System in a Cache With AutoFS

To verify that the entry was made correctly, change to the file system you modified, and then list the contents, as follows:

```
# cd filesystem
# ls filesystem
```

For example:

```
# cd /usr/openwin
# ls
bin demo etc include
lib man server share
```

For more information about AutoFS and how to edit the maps, refer to the AutoFS chapter of the *NFS Administration Guide*.

Maintaining a Cached File System

Table 35-2 shows the steps involved to maintain a cached file system.

Table 35-2 Maintaining a Cached File System

Activity	Description	For Instructions, Go To	
Modify the Cache	Modify the cache behavior.	▼ How to Modify File Systems in a Cache	page 590
Display Cache Information	Display information about cached file systems by using the `cfsadmin` command.	▼ How to Display Information About Cached File Systems	page 592
Perform Consistency Checking	Perform consistency checking on demand by using the `cfsadmin` command.	▼ How to Specify Consistency Checking on Demand	page 593
Delete a Cache	Delete cached file systems by using the `umount` command and the `cfsadmin` command.	▼ How to Delete a Cached File System	page 594
Check File System Integrity	Check the integrity of cached file systems by using the `fsck_cachefs` command.	▼ How to Check the Integrity of Cached File Systems	page 596

Maintaining the Cache

After you set up the cache, you can perform the following maintenance tasks on it:

- Modify file systems in the cache (by unmounting, deleting, recreating, and remounting the cache)
- Display cache information
- Check cache consistency
- Delete a file system from the cache
- Check cached file system integrity

Note – If you are using the `/etc/vfstab` file to mount file systems, you modify the cache by editing the file systems options in the `etc/vfstab` file. If you are using AutoFS, you modify the cache by editing the file systems options in the AutoFS maps.

▼ How to Modify File Systems in a Cache

For information on how to modify specific options of a file system, refer to the Chapter 34, "Mounting and Unmounting File Systems". When you modify a file system in the cache, you need to delete the cache and then recreate it. You may also need to reboot your machine in single user mode, depending on how your file systems are shared and accessed.

Following is an example of what this procedure may entail:

Example—Modifying File Systems in a Cache

In the following example, the cache is deleted, then re-created, and then mounted again with the `demandconst` option specified for the file system `/docs`. This example shows the steps including rebooting to single user mode. You may have other commands you prefer to use to accomplish some of the tasks shown in this example.

```
$ su
password:
# halt
.
.
.
ok boot -s
.
.
.
Type Cntrl-d to proceed with normal startup,
(or give root password for system maintenance):
# enter password:
.
.
.
Here is where you may be prompted from system to run fsck on the file system where the cache is located.
# fsck /local
# mount /local
# cfsadmin -d all /local/mycache
# cfsadmin -c /local/mycache
# reboot
.
.
.
login:
password:
# mount -F cachefs -o backfstype=nfs,cachedir=/local/cache1,demandconst merlin:/docs /docs
```

Verification—Modifying File Systems in a Cache

If you performed the steps in this procedure successfully, you will receive a prompt after each step, with no system output.

If you did not successfully mount the file system in the cache again, you would get an error message similar to the following:

```
cachefsstat: /doc: not a cachefs mount point
```

▼ How to Display Information About Cached File Systems

1. **Become root.**
2. **Display information about all file systems cached under a specified cache.**

```
# cfsadmin -l cache-directory
```

In this command, *cache-directory* is the name of the directory where the cache resides.

Example—Displaying Information About Cached File Systems

The following example shows information about the cache directory named `/local/mycache`. In this example, the file system `/docs` is cached in `/local/mycache`. The last line displays the cached file system.

```
# cfsadmin -l /local/mycache
cfsadmin: list cache FS information
   maxblocks     90%
   minblocks      0%
   threshblocks  85%
   maxfiles      90%
   minfiles       0%
   threshfiles   85%
   maxfilesize    3MB
merlin:_docs:_docs
#
```

▼ How to Specify Consistency Checking on Demand

1. **Become root.**
2. **Mount the file system in the cache specifying the** `demandconst` **option of the** `mount(1M)` **command, as follows:**

```
# mount -F cachefs -o backfstype=nfs,cachedir=/local/mycache,demandconst merlin:/docs /docs
```

3. **To enable consistency checking on a specific cached file system, use the** `cfsadmin(1M)` **command with the** `-s` **option as follows:**

```
# cfsadmin -s /docs
```

For more information about consistency checking, see “Consistency Checking of Cached File Systems With the Back File System” on page 611.

Verification—Specifying Consistency Checking on Demand

If you performed the steps in this procedure successfully, you will receive a prompt after each step, with no system output. If the check was not successful, you will receive an error message.

▼ How to Delete a Cached File System

1. **Become root.**

2. **Unmount all the file systems in the cache.**

```
# umount mount-point
```

In this command, *mount-point* specifies the directory where a file system or file systems are mounted.

3. **Determine the cache ID from the `cfsadmin -l` output, as follows:**

```
# cfsadmin -l cache-directory
cfsadmin: list cache FS information
   maxblocks     90%
   minblocks      0%
   threshblocks  85%
   maxfiles      90%
   minfiles       0%
   threshfiles   85%
   maxfilesize    3MB
merlin:_docs:_docs
#
```

4. **Delete a cached file system from a specified cache.**

```
# cfsadmin -d cache-id cache-directory
```

In this command,

cache-id	Is the name of the cached file system, which is the last line of the output from the `cfsadmin -l` command. See “How to Display Information About Cached File Systems” for more information. You can delete all the cached file systems in a particular cache by specifying `all` for *cache-id*.

cache-directory Is the directory where the cache resides.

Examples—Deleting a Cached File System

The following example unmounts a cached file system and deletes the cached file system from the cache.

```
# umount /usr/openwin
# cfsadmin -d merlin:docs /docs /local/mycache
```

The following example deletes all the cached file systems in the `/local/mycache` cache. This also deletes the cache.

```
# cfsadmin -d all /local/mycache
```

Verification—Deleting a Cached File System

The cache ID of the file system you just deleted should be missing from the output of the following command. Refer to the `cfsadmin(1M)` manpage for more information about the fields specified in the command output.

```
# cfsadmin -l /local/mycache
cfsadmin: list cache FS information
   maxblocks     90%
   minblocks      0%
   threshblocks  85%
   maxfiles      90%
   minfiles       0%
   threshfiles   85%
   maxfilesize    3MB
#
```

▼ How to Check the Integrity of Cached File Systems

Use the `fsck(1M)` command to check the integrity of cached file systems. The CacheFS version of `fsck(1M)` automatically corrects problems without requiring user interaction. You should not need to run `fsck(1M)` manually for cached file systems; `fsck(1M)` is run automatically at boot time or when the file system is mounted. If you want to manually check the integrity, you can use the following procedure.

See the `fsck_cachefs(1M)` man page for more information.

1. **Become root.**
2. **Check the cached file systems under a specified cache.**

```
# fsck -F cachefs [-m] [-o noclean] cache-directory
```

In this command,

`-m`	Causes `fsck` to check the cached file systems without making any repairs.
`-o noclean`	Forces a check on the cached file systems only. Does not make any repairs.
cache-directory	The name of the directory where the cache resides.

Example—Checking the Integrity of Cached File Systems

The following example checks the cached file systems that are part of the `/local/mycache` cache.

```
# fsck -F cachefs /local/mycache
```

Verification—Checking the Integrity of Cached File Systems

If you performed this procedure successfully, you will receive a prompt back with no system output. If the check was not successful, you will receive an error message.

Setting Up and Viewing CacheFS Statistics 36

This is a list of step-by-step instructions on how to set up and view the CacheFS statistics:

36

CacheFS Statistics

CacheFS statistics enable you to:

- Determine an appropriate cache size
- Observe the performance of the cache

These two pieces of information will help you determine the trade-off between your cache size and the desired performance of the cache.

The CacheFS statistics consist of three commands:

`cachefslog`	Specifies the location of the log file. This command also displays where the statistics are currently being logged, and enables you to halt logging. See the `cachefslog(1M)` man page for more information.
`cachefswssize`	Interprets the log file to give a recommended cache size. See the `cachefswssize(1M)` man page for more information.
`cachefsstat`	Displays statistical information about a specific file system or all cached file systems. The information provided in the output of this command is taken directly from the cache. See the `cachefsstat(1M)` man page for more information.

Note – The CacheFS statistics commands can be issued from any directory. You must be root to issue the `cachefswssize(1M)` command.

The statistics begin accumulating when you create the log file. When the work session length of time is up, stop the logging by using the `cachefslog -h` command, as described in “How to Stop the Logging Process” on page 606.

Prerequisites for Setting Up and Viewing the CacheFS Statistics

Before using the CacheFS statistics commands, you must:

- Set up your cache using the `cfsadmin(1M)` command.
- Decide on an appropriate length of time to allow statistical information to collect in the log file you create. The length of time should equal a typical work session; for example, a day, a week, or a month.
- Select a location or path for the log file. Make sure there is enough space to allow for the growth of the log file. The longer you intend to allow statistical information to collect in the log file, the more space you will need.

Note – The following procedures are presented in a recommended order. The order is not required.

Setting Up CacheFS Statistics

Table 36-1 shows the steps involved to set up CacheFS statistics.

Table 36-1 Setting Up CacheFS Statistics

Activity	Description	For Instructions, Go To	
Set Up Logging	Set up logging on a cached file system using the `cachefslog` command.	▼ How to Set Up the Logging Process	page 603
Locate the Log File	Locte the log file with the `cachefslog` command.	▼ How to Locate the Log File	page 605
Stop the Logging Process	Stop the logging process with the `cachefslog` command.	▼ How to Stop the Logging Process	page 606
View the Cache Size	View the cache size using the `cachefswssize` command.	▼ How to View the Working Set (Cache) Size	page 607
View the Cache Statistics	View the statistics using the `cachefsstat` command.	▼ How to View Cache Statistics	page 606

Setting Up the Logging Process

The first step in the process of determining your ideal cache size is to set up the log file. The statistics are collected in the log file. You can then locate the log file in case you need to verify its location. You stop the logging process on a particular cached file system after you have gathered the necessary information about the cache.

▼ How to Set Up the Logging Process

Set up the logging process with the `cachefslog` command.

```
$ cachefslog -f log-file-path mount-point
```

In this command,

`-f` *option*	Sets up the logging process.
log-file-path	Represents the location of the log file. The log file is a standard file you create with an editor, such as `vi`.
mount-point	Designates the mount point (cached file system) for which statistics are being collected.

Example—Setting Up the Logging Process

The following example sets up the log file `samlog` to collect statistics about `/home/sam`. The location of `samlog` is `/var/tmp/samlog`.

```
$ cachefslog -f /var/tmp/samlog /home/sam
  /var/tmp/samlog: /home/sam
```

Verification—Setting Up the Logging Process

1. **To verify that you set up the log file correctly, use the** `cachefslog` **command, as follows:**

```
$ cachefslog mount-point
```

For example:

```
$ cachefslog /home/sam
/var/tmp/samlog: /home/sam
```

How to Locate the Log File

You can also use the `cachefslog(1M)` command with no options to locate a log file for a particular mount point.

```
$ cachefslog mount-point
```

In this command,

mount-point	Is the cached file system for which you want to view the statistics.

Examples—Locating the Log File

The following example shows that no log file has been set up for the specified file system.

```
$ cachefslog /home/zap
   not logged: /home/zap
```

▼ How to Stop the Logging Process

Use the `-h` option of the `cachefslog(1M)` command to stop the logging process.

```
$ cachefslog -h mount-point
```

Example—Stopping the Logging Process

The following example shows what you would see if a log file has been set up. The location of the log file is `/var/tmp/stufflog`.

```
$ cachefslog /home/stuff
    /var/tmp/stufflog: /home/stuff
```

The following example halts logging on `/home/stuff`.

```
$ cachefslog -h /home/stuff
not logged: /home/stuff
```

Verification—Stopping the Logging Process

If you get a system response other than the one specified in the above example, you did not successfully stop the logging process. Check to see if you are using the correct log file name and mount point.

Viewing the Cache Size

You may want to check if you need to increase the size of the cache or determine what the ideal cache size is based on your activity since you last used the `cachefslog(1M)` command for a particular mount point.

▼ How to View the Working Set (Cache) Size

1. **Become root.**
2. **View the current and highest logged cache size with the** `cachefswssize(1M)` **command.**

```
# cachefswssize log-file-path
```

Example—Viewing the Working Set (Cache) Size

In the following example, the end size is the size of the cache at the time you issued the `cachefswssize` command. The high water size is the largest size of the cache during the time frame in which logging has occurred. The end size is the current size of the cache.

```
# cachefswssize /var/tmp/samlog

    /home/sam
           end size:  10688k
    high water size:  10704k

    /
           end size:   1736k
    high water size:   1736k

    /opt
           end size:    128k
    high water size:    128k

    /nfs/saturn.dist
           end size:   1472k
    high water size:   1472k

    /usr/openwin
           end size:   7168k
    high water size:   7168k

    /nfs/venus.svr4
           end size:   4688k
    high water size:   5000k

    /usr
           end size:   4992k
    high water size:   4992k

    total for cache
       initial size: 110960k
           end size:  30872k
    high water size:  30872k
```

Viewing the Statistics

You may want to view certain information about a specific cached file system. The following table explains the terminology displayed in the statistics output.

Table 36-2 Statistics Output Terminology

Output Term	Description
hit rate	The rate of cache hits versus cache misses, followed by the actual number of hits and misses. A cache hit occurs when the user wants to perform an operation on a file or files, and the file or files are actually in the cache. A cache miss occurs when the file was not in the cache. The load on the server is the sum of cache misses, consistency checks, and modifications (modifies).
checks	The number of consistency checks performed, followed by the number that passed, and the number that failed.
modifies	The number of modify operations; for example, writes or creates.

▼ How to View Cache Statistics

View the statistics with the `cachefsstat(1M)` command. You can do this at any time. For example, you do not have to set up logging in order to view the statistics.

```
$ cachefsstat mount-point
```

In this command,

mount-point	Is the cached file system for which you want to view the statistics.

If you do not specify the mount point, statistics for all mounted CacheFS file systems will be displayed.

Example—Viewing Cache Statistics

```
$ cachefsstat /home/sam
    cache hit rate: 73% (1234 hits, 450 misses)
    consistency checks: 700 (650 pass, 50 fail)
    modifies: 321
```

The Cache Structure and Behavior

Each cache has a set of parameters that determine how it behaves and its structure. The parameters are set to default values which are listed in Table 36-3. The default values specify that the entire front file system is used for caching, which is the recommended method of caching file systems.

Table 36-3 Cache Parameters and Their Default Values

Cache Parameter	Default Value	Definition
maxblocks	90%	Sets the maximum number of blocks that CacheFS is allowed to claim within the front file system.
minblocks	0%	Sets the minimum number of blocks that CacheFS is allowed to claim within the front file system.
threshblocks	85%	Sets the number of blocks that must be available in the front file system before CacheFS can claim more than the blocks specified by minblocks.
maxfiles	90%	Sets the maximum number of available inodes (number of files) that CacheFS is allowed to claim within the front file system.
minfiles	0%	Sets the minimum number of available inodes (number of files) that CacheFS is allowed to claim within the front file system.
threshfiles	85%	Sets the number of inodes (number of files) that must be available in the front file system before CacheFS can claim more than the files specified in minfiles.

Typically, you should not change any of these parameter values. They are set to default values to achieve optimal cache behavior. However, you may want to modify the `maxblocks` and `maxfiles` settings if you have some room in the front file system that is not used by the cache, and you wish to use it for some other file system. You do this using the `cfsadmin(1M)` command. For example:

```
$ cfsadmin -o maxblocks=60
```

Consistency Checking of Cached File Systems With the Back File System

To ensure that the cached directories and files are kept up to date, CacheFS periodically checks consistency of files stored in the cache. To check consistency, CacheFS compares the current modification time to the previous modification time. If the modification times are different, all data and attributes for the directory or file are purged from the cache and new data and attributes are retrieved from the back file system.

When a user requests an operation on a directory or file, CacheFS checks if it is time to verify consistency. If so, CacheFS obtains the modification time from the back file system and performs the comparison.

Consistency Checking on Demand

By specifying the `demandconst` option of the `mount(1M)` command, consistency checks can be performed only when you explicitly request them for file systems mounted with this option. After specifying the `demandconst` option when you mount a file system in a cache, you use the `cfsadmin(1M)` command with the `-s` option to request a consistency check. By default, consistency checking is performed file by file as the files are accessed. If no files are accessed, no checks are performed. Use of the `demandconst` option will avoid the situation where the network is flooded with consistency checks. For more information about consistency checking on demand, refer to the `cfsadmin(1M)` command.

Configuring Additional Swap Space 37

This is a list of the overview conceptual information and step-by-step instructions in this chapter.

About Swap Space

This section provides a conceptual overview of swap space and briefly discusses the differences between Solaris 1.x (SunOS 4.x) and Solaris 2.x (SunOS 5.x) swap requirements. If you are already familiar with the Solaris 2.x swap mechanism, proceed to the section called "Planning for Swap Space" on page 618.

It is important for administrators to understand the features of the Solaris 2.x swap mechanism in determining:

- Swap space requirements
- The relationship with the `tmpfs` file system
- Recovery from error messages related to swap space

Swap Space and Virtual Memory

The Solaris 2.x system software uses some disk slices for temporary storage rather than for file systems. These slices are called *swap* slices. Swap slices are used as virtual memory storage areas when the system does not have enough physical memory to handle current processes.

The Solaris 2.x virtual memory system maps physical copies of files on disk to virtual addresses in memory. Physical memory pages which hold the data for these mappings can be backed by regular files in the file system or by swap space. If the memory is backed by swap space it is referred to as *anonymous* memory because the user doesn't know the names of the files backing the memory.

Solaris 1.x anonymous memory pages are mapped using randomly assigned names from the system's swap space pool. These memory pages are used for:

- Private copies of data created during copy-on-write operations
- Process and stack segments
- The `tmpfs` file system storage resources

The limitations of the Solaris 1.x anonymous memory implementation are:

- Physical storage (disk-backed swap) must always be reserved for anonymous memory mappings even if the application doesn't use it.

  For example, applications with large data segments must be configured with lots of swap space even if the pages are not written out to physical storage.

- The formula used to associate an anonymous memory page with physical storage is limited and inflexible because the backing store is chosen at random and can never be changed.

The Solaris 2.x software environment introduces the concept of *virtual swap space*, a layer between anonymous memory pages and the physical storage (or disk-backed swap space) that actually back these pages. A system's virtual swap space is equal to the sum of all its physical (disk-backed) swap space plus a portion of the currently available physical memory.

Virtual swap space has these advantages:

- The need for large amounts of physical swap space is reduced because virtual swap space does not necessarily correspond to physical (disk) storage.
- A pseudo file system called `swapfs` provides addresses for anonymous memory pages. Because `swapfs` controls the allocation of memory pages, it has greater flexibility in deciding what happens to a page. For example, it may change the page's requirements for disk-backed swap storage.

Swap Space and the `tmpfs` *File System*

The `tmpfs` file system is activated automatically in the Solaris 2.x environment by an entry in the `/etc/vfstab` file. The `tmpfs` file system stores files and their associated information in memory (in the `/tmp` directory) rather than on disk, which speeds access to those files. This results in a major performance enhancement for applications such as compilers and DBMS products that use `/tmp` heavily.

The `tmpfs` file system allocates space in the `/tmp` directory from the system's swap resources. This means that as you use up space in `/tmp`, you are also using up swap space. So if your applications use `/tmp` heavily and you do not monitor swap space usage, your system could run out of swap space.

Use the following if you want to use `tmpfs` but your swap resources are limited:

- Mount the `tmpfs` file system with the size option (`-o` *size*) to control how much of the swap resources `tmpfs` can use.
- If you are close to running out of swap space, you can use your compiler's `TMPDIR` environment variable to point to a larger temporary directory.

  Using your compiler's `TMPDIR` variable only controls whether the compiler is using `/tmp` or not. It has not effect on other programs' use of `/tmp`.

How Do I Know If I Need More Swap Space?

This section lists several possible error messages displayed when you run out of swap space.

Swap-Related Error Messages

These messages indicate that an application was trying to get more anonymous memory and there was no swap space left to back it.

```
application is out of memory
```

```
malloc error O
```

```
WARNING: Sorry, no swap space to grow stack for pid
```

`tmpfs`-Related Error Messages

`directory: File system full, swap space limit exceeded`

This message is displayed if a page could not be allocated when writing a file. This can occur when `tmpfs` tries to write more than it is allowed or if currently executed programs are using a lot of memory.

`directory: File system full, memory allocation failed`

This messages means `tmpfs` ran out of physical memory while attempting to create a new file or directory.

See `tmpfs(7FS)` for information on recovering from the `tmpfs`-related error messages.

How Swap Space Is Allocated

Initially, swap space is allocated as part of the Solaris installation process. If you use the installation program's automatic layout of disk slices and do not manually change the size of the swap slice, the Solaris installation program allocates default swap slices as shown in Table 37-1.

Table 37-1 Default Swap Space Allocations

If Your System Has *n* Mbytes of Physical Memory...	Then the Default Swap Space Allocated Is ...
16-64	32 Mbytes
64-128	64 Mbytes
128-512	128 Mbytes
greater than 512	256 Mbytes

The `/etc/vfstab` File

After the system is installed, swap slices and files are listed in the `/etc/vfstab` file and are activated by the `/sbin/swapadd` script when the system is booted.

An entry for a swap device in the `/etc/vfstab` file contains:

- The full path name of the swap slice or file
- File system type of `swap`

Because the file system containing a swap file must be mounted before the swap file is activated, make sure that the entry that mounts the file system comes before the entry that activates the swap file in the `/etc/vfstab` file.

37

Planning for Swap Space

The most important factors in determining swap space size are the requirements of the system's software applications. For example, large applications such as computer-aided-design simulators, database-management products, transaction monitors, and geologic analysis systems can consume as much as 200-1000 Mbytes of swap space in very large memory systems.

Consult your application vendor for swap space requirements for any application whose data files typically exceed 10-20 Mbytes in size.

If you are unable to determine swap space requirements from the application vendor, use the following guidelines to allocate swap space:

- To support your applications, allocate:
  - 1 Mbyte per trivial application such as `xterm`.
  - 2-3 Mbytes per lightweight application such as a calendar or mail application.
  - 20-50 Mbytes for large applications such as desktop publishing software.
- To save crash dumps, allocate 100% of physical memory to save a worst-case crash dump.
- If you are unsure of system or application requirements, allocate 50 to 100% of the system's physical memory. For example, allocate 16-32 Mbytes of swap space for a system with 32 Mbytes of physical memory. This will provide 48-64 Mbytes of total virtual swap space.
- Determine whether large applications (like compilers) will be using the `/tmp` directory. Then allocate additional swap space to be used by `tmpfs`. See "Swap Space and the tmpfs File System" for information about `tmpfs`.

Monitoring Swap Resources

The `/usr/sbin/swap` command is used to manage swap areas. Two options, `-l` and `-s`, are used to display information about swap resources.

Use the `swap -l` command to identify a system's swap areas. Activated swap devices or files are listed under the `swapfile` column.

```
# swap -l
swapfile             dev  swaplo blocks   free
/dev/dsk/c0t2d0s1 32,17      8 205624 192704
```

Use the `swap -s` command to monitor swap resources.

```
# swap -s
total: 10492k bytes allocated + 7840k reserved = 18332k used, \
21568k available
```

The `used` plus `available` figures equals total swap space on the system, which includes a portion of physical memory and swap devices (or files).

You can use the amount of swap space available and used (in the `swap -s` output) as a way to monitor swap space usage over time. If a system's performance is good, use `swap -s` to see how much swap space is available. When the performance of a system slows down, check the amount of swap space available to see if it has decreased. Then you can identify what changes to the system might have caused swap space usage to increase.

Keep in mind when using this command that the amount of physical memory available for swap usage changes dynamically as the kernel and user processes lock down and release physical memory.

Note – The `swap -l` command displays swap space in 512-byte blocks and the `swap -s` command displays swap space in 1024-byte blocks. If you add up the blocks from `swap -l` and convert them to Kbytes, it will be less than `used` + `available` (in the `swap -s` output) because `swap -l` does not include physical memory in its calculation of swap space.

The output from the `swap -s` command is summarized in Table 37-2.

Table 37-2 Output of the `swap -s` Command

Keyword	Description
`bytes allocated`	The total amount of swap space in 1024-byte blocks that is currently allocated as backing store (disk-backed swap space).
`reserved`	The total amount of swap space in 1024-byte blocks not currently allocated, but claimed by memory for possible future use.
`used`	The total amount of swap space in 1024-byte blocks that is either allocated or reserved.
`available`	The total amount of swap space in 1024-byte blocks that is currently available for future reservation and allocation.

Adding More Swap Space

As system configurations change and new software packages are installed, you might need to add more swap space. The preferred way to add more swap space is to use the `mkfile` and `swap` commands to designate a part of an existing UFS or NFS file system as a supplementary swap area. These commands, described below, enable you to add more swap space without repartitioning a disk.

An alternative way to add more swap space is to repartition a disk. See "Managing Disks" for your system for information on how to repartition a disk.

Creating a Swap File

The following steps are used to create a swap file:

- Create a swap file using the `mkfile` command.
- Activate the swap file with the `swap` command.
- Add an entry for the swap file in the `/etc/vfstab` file so that its activated automatically when the system is booted.

The `mkfile` Command

The `mkfile` command creates a file that is suitable for use either as an NFS-mounted or local swap area. The sticky bit is set, and the file is padded with zeros. You can specify the size of the swap file in bytes (the default) or in kilobytes, blocks, or megabytes using the `k`, `b`, or `m` suffixes, respectively.

Table 37-3 shows the options to the `mkfile` command.

Table 37-3 Options to the `mkfile` Command

Option	Description
`-n`	Creates an empty file. The size is noted, but the disk blocks are not allocated until data is written to them.
`-v`	Verbose. Reports the names and sizes of created files.

Caution – Use the `-n` option only when creating an NFS swap file.

▼ How to Create a Swap File and Make It Available

1. **Become root.**
   You can create a swap file without root permissions, but it is a good idea to have root be the owner of the swap file so that the swap file cannot be inadvertently overwritten.

2. **Create the swap file.**

```
# mkfile nnn[k|b|m] filename
```

   The swap file of the size *nnn* (in Kbytes, bytes, or Mbytes) and file name you specify is created.

   In this example, a 20-Mbyte swap file named `swapfile` is created on a UFS file system named `/files`:

```
# mkfile 20m /files/swapfile
```

3. **Activate the swap file.**

```
# /usr/sbin/swap -a /path/filename
```

You must use the absolute path name to specify the swap file. The swap file is added and available until the file system is unmounted or the system is rebooted.

4. **Add an entry for the swap file to the** `/etc/vfstab` **file that specifies the full path name of the file, and designates** `swap` **as the file system type.**
For example, to make the swap file `/files/swapfile` available, add this entry to the `/etc/vfstab` file:

```
/files/swapfile        -      -       swap     -      no     -
```

5. **To verify that the swap file is added, type.**

```
/usr/sbin/swap -l
```

Example—Creating a Swap File and Making It Available

```
# swap -a /files/swapfile
# swap -l
swapfile             dev  swaplo blocks   free
/dev/dsk/c0t2d0s1   32,17      8 205624 192704
/files/swapfile       -        8  40952  40952
```

Removing a Swap File From Use

If the user no longer needs the extra swap space, you can remove it.

▼ How to Remove Extra Swap Space

1. **Become root.**
2. **Use the** `swap -d` **command to remove swap space..**

```
# /usr/sbin/swap -d /path/filename
```

   The swap file name is removed from the list so that it is no longer available for swapping. The file itself is not deleted.

3. **Edit the** `/etc/vfstab` **file and delete the entry for the swap file.**
4. **Recover the disk space so that you can use it for something else.**

```
rm swap-filename
```

   If the swap space is a file, remove it. Or, if the swap space is on a separate slice and you are sure you will not need it again, make a new file system and mount the file system.

   See Chapter 34, “Mounting and Unmounting File Systems,” for more information.

Example—Removing Extra Swap Space

```
# swap -d /files/swapfile
# swap -l
swapfile             dev  swaplo  blocks   free
/dev/dsk/c0t2d0s1   32,17      8  205624  192720
# ls -l /files/swapfile
-rw------T   1 root   staff   20971520  Jun 2 12:54 /files/swapfile
```

Recognizing File Access Problems 38

This is a list of the step-by-step instructions in this chapter.

Users frequently experience problems—and call on a system administrator for help—because they cannot access a program, a file, or a directory that they used to be able to use. Whenever you encounter such a problem, investigate one of three areas:

- The user's search path may have been changed, or the directories in the search path may not be in the proper order.
- The file or directory may not have the proper permissions or ownership.
- The configuration of a system accessed over the network may have changed.

This chapter briefly describes how to recognize problems in each of these three areas and suggests possible solutions.

Solving Problems With Search Paths

A message of `Command not found` indicates one of the following:

- The command is not available on the system.
- The command directory is not in the search path.

If the wrong version of the command is found, a directory that has a command of the same name is in the search path. In this case, the proper directory may be later in the search path or may not be present at all.

▼ How to Diagnose and Correct Search Path Problems

1. **Display the current search path.**

```
$ echo $PATH
```

2. **Check the following:**
   - Is the search path correct?
   - Is the search path listed before other search paths where another version of the command is found?
   - Make sure the command is in one of the search paths.

   If the path needs correction, go to step 3. Otherwise, go to step 4.

3. **Set the path as follows:**

Shell	File	Syntax	Notes
Bourne and Korn	`$HOME/.profile`	`path=.:/usr/bin:`	A colon separates path names.
C	`$HOME/.cshrc` or `$ HOME/.login`	`set path`	A blank space separates the two words.

4. **Activate the new path as follows:**

Shell	File Where Path Is Located	Command
Bourne and Korn	`.profile.`	`. .profile`
C	`.cshrc`	`source .cshrc`
	`.login`	`source .login`

Verification—Diagnosing and Correcting Search Path Problem

To verify the path use the command shown in Table 38-1:

Table 38-1 Path Verification

Shell	Command
Bourne and Korn	`$ echo $PATH`
C	`hostname% echo $PATH`

Example—Diagnosing and Correcting Search Path Problems

```
$ command
command not found
$ echo $PATH
```

Type `which` *command-name* and press Return.

If the command is found in the path, the path and the name of the command are displayed.

Note – The `which` command looks in the `.cshrc` file for information. The `which` command may give misleading results if you execute it from the Bourne or Korn shell and you have a `.cshrc` file that contains aliases for `which`. To ensure accurate results, use the `which` command in a C shell, or, in the Korn shell, use the `whence` command.

This example shows that the OpenWindows executable is not in any of the directories in the search path:

```
venus% which openwin
no openwin in . /home/ignatz /sbin /usr/sbin /usr/bin /etc \
/home/ignatz/bin /bin /home/bin /usr/etc
venus%
```

This example shows that the executable for OpenWindows is found among the directories in the search path:

```
venus$ which openwin
/usr/openwin
venus$
```

If you cannot find a command, look at the manual page. For example, if you cannot find the `lpsched` command (the `lp` printer daemon), `lpsched(1M)` tells you the path is `/usr/lib/lp/lpsched`.

Solving Problems With Permission and Ownership

When users cannot access files or directories that they used to be able to access, the most likely problem is that permissions or ownership of the files or directories has changed.

Frequently, file and directory ownerships change because someone edited the files as root. When you create home directories for new users, be sure to make the user the owner of the dot (`.`) file in the home directory. When users do not own "`.`" they cannot create files in their own home directory.

Access problems can also arise when the group ownership changes or when a group of which a user is a member is deleted from the `/etc/groups` database.

▼ How to Change File Permissions

Table 38-2 shows the octal values for setting file permissions. You use these numbers in sets of three to set permissions for owner, group, and other. For example, the value 644 sets read / write permissions for owner, and read-only permissions for group and other.

Table 38-2 Octal Values for File Permissions

Value	Description
0	No permissions
1	Execute-only
2	Write-only

Table 38-2 Octal Values for File Permissions *(Continued)*

Value	Description
3	Write, execute
4	Read-only
5	Read, execute
6	Read, write
7	Read, write, execute

1. List the file permissions.

```
# ls -l filename
```

The long listing shows the current permissions for the file.

2. Change the file permissions.

```
# chmod nnn filename
```

Permissions are changed using the numbers you specify.

Note – You can change permissions on groups of files or on all files in a directory using meta characters such as (*?) in place of file names or in combination with them.

Example—Changing File Permissions

This example shows changing the permissions of a file from 666 (read / write, read / write, read / write) to 644 (read / write, read-only, read-only).

```
$ ls -l quest
-rw-rw-rw-  1 ignatz   staff    6023 Aug  5 12:06 quest
$ chmod 644 quest
```

Verification—Changing File Permissions

To verify that the permissions have been changed, use the `ls -l` command.

```
$ ls -l
-rw-r--r--  1 ignatz   staff    6023 Aug  5 12:06 quest
$
```

▼ How to Change File Ownership

Note – You must own a file or directory (or have root permission) to be able to change its owner.

1. **Become root.**
2. **List the file permissions.**

```
# ls -l filename
```

The owner of the file is displayed in the third column.

3. **Change file owner.**

```
# chown new-owner filename
```

Ownership is assigned to the new owner you specify.

Example—Changing File Ownership

```
# ls -l quest
-rw-r--r--  1 fred   staff    6023 Aug  5 12:06 quest

# chown ignatz quest
```

Verification—Changing File Ownership

```
# ls -l quest
-rw-r--r--  1 ignatz   staff    6023 Aug  5 12:06 quest
#
```

▼ How to Change File Group Ownership

Change a file's group ownership.

```
$ chgrp gid filename
```

The group ID for the file you specify is changed.

Example—Changing File Group Ownership

```
$ ls -lg junk
-rw-r--r-- 1 other 3138 Oct 31 14:49 junk
$ chgrp 10 junk
```

Verification—Changing File Group Ownership

```
$ ls -lg junk
-rw-r--r-- 1 staff  3138 Oct 31 14:49 junk
$
```

See "Managing System Security" in *System Administration Guide, Volume II* for information about how to edit group accounts.

Recognizing Problems With Network Access

If users have problems using the `rcp` remote copy command to copy files over the network, the directories and files on the remote system may have restricted access by setting permissions. Another possible source of trouble is that the remote system and the local system are not configured to allow access.

See *NFS Administration Guide* for information about problems with network access and problems with accessing systems through AutoFS.

Checking File System Integrity 39

This is a list of the conceptual information and step-by-step instructions in this chapter.

The UFS file system relies on an internal set of tables to keep track of inodes and used and available blocks. When these internal tables are not properly synchronized with data on a disk, inconsistencies result and file systems need to be repaired.

File systems can be damaged or become inconsistent because of abrupt termination of the operating system in these ways:

- Power failure
- Accidental unplugging of the system
- Turning the system off without proper shutdown procedure
- A software error in the kernel

File system corruption, while serious, is not common. When a system is booted, a file system consistency check is automatically performed (with the `fsck` program). Most of the time, this file system check repairs problems it encounters.

This chapter describes what the `fsck` program checks and repairs, the `fsck` options, and `fsck` error messages. It also describes the following tasks:

- How to modify the automatic checking done during booting
- How to find out if a file system needs to be checked
- How to check and repair a UFS file system interactively
- How to restore a bad superblock
- How to fix a UFS file system that `fsck` cannot repair

The `fsck` program places files and directories that are allocated but unreferenced in the `lost+found` directory. The inode number of each file is assigned as the name. If the `lost+found` directory does not exist, `fsck` creates it. If there is not enough space in the `lost+found` directory, `fsck` increases its size.

Understanding How the File System State Is Recorded

The `fsck` command uses a *state flag*, which is stored in the superblock, to record the condition of the file system. This flag is used by the `fsck` command to determine whether or not a file system needs to be checked for consistency. The flag is used by the `/etc/bcheckrc` script during booting and by the `fsck` command when run from a command line using the `-m` option. If you ignore the result from the `-m` option to `fsck`, all file systems can be checked regardless of the setting of the state flag.

The possible state flag values are described in Table 39-1.

Table 39-1 State Flag Values

State Flag Value	Description
FSCLEAN	If the file system was unmounted properly, the state flag is set to FSCLEAN. Any file system with an FSCLEAN state flag is not checked when the system is booted.
FSSTABLE	The file system is (or was) mounted but has not changed since the last checkpoint (sync or fsflush) which normally occurs every 30 seconds. For example, the kernel periodically checks if a file system is idle and, if so, flushes the information in the superblock back to the disk and marks it FSSTABLE. If the system crashes, the file system structure is stable, but users may lose a small amount of data. File systems that are marked FSSTABLE can skip the checking before mounting. The mount(2) system call will not mount a file system for read / write if the file system state is not FSCLEAN or FSSTABLE.
FSACTIVE	When a file system is mounted and then modified, the state flag is set to FSACTIVE. The file system may contain inconsistencies. A file system will be marked as FSACTIVE before any modified metadata is written to the disk. When a file system is unmounted gracefully, the state flag is set to FSCLEAN. A file system with the FSACTIVE flag must be checked by fsck because it may be inconsistent.
FSBAD	If the root (/) file system is mounted when its state is not FSCLEAN or FSSTABLE, the state flag is set to FSBAD. The kernel will not change this file system state to FSCLEAN or FSSTABLE. If a root (/) file system is flagged FSBAD as part of the boot process, it will be mounted read-only. You can run fsck on the raw root device. Then remount the root (/) file system as read / write.

Table 39-2 shows when the state flag is modified.

Table 39-2 When the State Flag is Modified

Before `fsck`	After `fsck`		
Initial State	**No Errors**	**New State All Errors Corrected**	**Uncorrected Errors**
unknown	stable	stable	unknown
active	stable	stable	active
stable	stable	stable	active
clean	clean	stable	active
bad	stable	stable	bad

What `fsck` Checks and Tries to Repair

This section describes what happens in the normal operation of a file system, what can go wrong, what problems `fsck` (the checking and repair utility) looks for, and how it corrects the inconsistencies it finds.

Why Inconsistencies May Occur

Every working day hundreds of files may be created, modified, and removed. Each time a file is modified, the operating system performs a series of file system updates. These updates, when written to the disk reliably, yield a consistent file system.

When a user program does an operation to change the file system, such as a write, the data to be written is first copied into an internal in-core buffer in the kernel. Normally, the disk update is handled asynchronously; the user process is allowed to proceed even though the data write may not happen until long after the write system call has returned. Thus at any given time, the file system, as it resides on the disk, lags behind the state of the file system represented by the in-core information.

The disk information is updated to reflect the in-core information when the buffer is required for another use or when the kernel automatically runs the `fsflush` daemon (at 30-second intervals). If the system is halted without writing out the in-core information, the file system on the disk will be in an inconsistent state.

A file system can develop inconsistencies in several ways. The most common causes are operator error and hardware failures.

Problems may result from an *unclean halt*, if a system is shut down improperly, or when a mounted file system is taken offline improperly. To prevent unclean halts, the current state of the file systems must be written to disk (that is, “synchronized”) before halting the CPU, physically taking a disk pack out of a drive, or taking a disk offline.

Inconsistencies can also result from defective hardware. Blocks can become damaged on a disk drive at any time, or a disk controller can stop functioning correctly.

The UFS Components That Are Checked for Consistency

This section describes the kinds of consistency checks the `fsck` applies to these UFS file system components: superblock, cylinder group blocks, inodes, indirect blocks, and data blocks.

Superblock

The superblock stores summary information, which is the most commonly corrupted item in a UFS file system. Each change to the file system inodes or data blocks also modifies the superblock. If the CPU is halted and the last command is not a `sync` command, the superblock will almost certainly be corrupted.

The superblock is checked for inconsistencies in:

- File system size
- Number of inodes
- Free-block count
- Free-inode count

File System and Inode List Size

The file system size must be larger than the number of blocks used by the superblock and the number of blocks used by the list of inodes. The number of inodes must be less than the maximum number allowed for the file system. The file system size and layout information are the most critical pieces of information for `fsck`. Although there is no way to actually check these sizes, because they are statically determined when the file system is created, `fsck` can check that the sizes are within reasonable bounds. All other file system checks require that these sizes be correct. If `fsck` detects corruption in the static parameters of the primary superblock, it requests the operator to specify the location of an alternate superblock.

Free Blocks

Free blocks are stored in the cylinder group block maps. `fsck` checks that all the blocks marked as free are not claimed by any files. When all the blocks have been accounted for, `fsck` checks to see if the number of free blocks plus the number of blocks claimed by the inodes equal the total number of blocks in the file system. If anything is wrong with the block allocation maps, `fsck` rebuilds them, leaving out blocks already allocated.

The summary information in the superblock contains a count of the total number of free blocks within the file system. The `fsck` program compares this count to the number of free blocks it finds within the file system. If the counts do not agree, `fsck` replaces the count in the superblock with the actual free-block count.

Free Inodes

The summary information in the superblock contains a count of the free inodes within the file system. The `fsck` program compares this count to the number of free inodes it finds within the file system. If the counts do not agree, `fsck` replaces the count in the superblock with the actual free inode count.

Inodes

The list of inodes is checked sequentially starting with inode 2 (inode 0 and inode 1 are reserved). Each inode is checked for inconsistencies in:

- Format and type
- Link count
- Duplicate block
- Bad block numbers
- Inode size

Format and Type of Inodes

Each inode contains a mode word, which describes the type and state of the inode. Inodes may be one of six types:

- Regular
- Directory
- Block special
- Character special
- FIFO (named-pipe)
- Symbolic link

Inodes may be in one of three states:

- Allocated
- Unallocated
- Partially allocated

When the file system is created, a fixed number of inodes are set aside, but they are not allocated until they are needed. An allocated inode is one that points to a file. An unallocated inode does not point to a file and, therefore, should be empty. The partially allocated state means that the inode is incorrectly formatted. An inode can get into this state if, for example, bad data is written into the inode list because of a hardware failure. The only corrective action `fsck` can take is to clear the inode.

Link Count

Each inode contains a count of the number of directory entries linked to it. The `fsck` program verifies the link count of each inode by examining the entire directory structure, starting from the root directory, and calculating an actual link count for each inode.

Discrepancies between the link count stored in the inode and the actual link count as determined by `fsck` may be of three types:

- The stored count is *not* 0 and the actual count is 0.

  This condition can occur if no directory entry exists for the inode. In this case, `fsck` puts the disconnected file in the `lost+found` directory.

- The stored count is *not* 0 and the actual count is *not* 0, but the counts are *unequal*.

This condition can occur if a directory entry has been added or removed but the inode has not been updated. In this case, `fsck` replaces the stored link count with the actual link count.

- The stored count is 0 and the actual count is not 0.

  In this case `fsck` changes the link count of the inode to the actual count.

Duplicate Blocks

Each inode contains a list, or pointers to lists (indirect blocks), of all the blocks claimed by the inode. Because indirect blocks are owned by an inode, inconsistencies in indirect blocks directly affect the inode that owns the indirect block.

The `fsck` program compares each block number claimed by an inode to a list of allocated blocks. If another inode already claims a block number, the block number is put on a list of duplicate blocks. Otherwise, the list of allocated blocks is updated to include the block number.

If there are any duplicate blocks, `fsck` makes a second pass of the inode list to find the other inode that claims each duplicate block. (A large number of duplicate blocks in an inode may be caused by an indirect block not being written to the file system.) It is not possible to determine with certainty which inode is in error. The `fsck` program prompts you to choose which inode should be kept and which should be cleared.

Bad Block Numbers

The `fsck` program checks each block number claimed by an inode to see that its value is higher than that of the first data block and lower than that of the last data block in the file system. If the block number is outside this range, it is considered a bad block number.

Bad block numbers in an inode may be caused by an indirect block not being written to the file system. The `fsck` program prompts you to clear the inode.

Inode Size

Each inode contains a count of the number of data blocks that it references. The number of actual data blocks is the sum of the allocated data blocks and the indirect blocks. `fsck` computes the number of data blocks and compares that block count against the number of blocks the inode claims. If an inode contains an incorrect count, `fsck` prompts you to fix it.

Each inode contains a 64-bit size field. This field shows the number of characters (data bytes) in the file associated with the inode. A rough check of the consistency of the size field of an inode is done by using the number of characters shown in the size field to calculate how many blocks should be associated with the inode, and then comparing that to the actual number of blocks claimed by the inode.

Indirect Blocks

Indirect blocks are owned by an inode. Therefore, inconsistencies in an indirect block affect the inode that owns it. Inconsistencies that can be checked are:

- Blocks already claimed by another inode
- Block numbers outside the range of the file system

The consistency checks are also performed for indirect blocks.

Data Blocks

An inode can directly or indirectly reference three kinds of data blocks. All referenced blocks must be of the same kind. The three types of data blocks are:

- Plain data blocks
- Symbolic-link data blocks
- Directory data blocks

Plain data blocks contain the information stored in a file. Symbolic-link data blocks contain the path name stored in a symbolic link. Directory data blocks contain directory entries. `fsck` can check the validity only of directory data blocks.

Directories are distinguished from regular files by an entry in the `mode` field of the inode. Data blocks associated with a directory contain the directory entries. Directory data blocks are checked for inconsistencies involving:

- Directory inode numbers pointing to unallocated inodes
- Directory inode numbers greater than the number of inodes in the file system
- Incorrect directory inode numbers for "." and ".." directories
- Directories disconnected from the file system

Directory Unallocated

If the inode number in a directory data block points to an unallocated inode, `fsck` removes the directory entry. This condition can occur if the data blocks containing the directory entries are modified and written out but the inode does not get written out. This condition can occur if the CPU is halted without warning.

Bad Inode Number

If a directory entry inode number points beyond the end of the inode list, `fsck` removes the directory entry. This condition can occur when bad data is written into a directory data block.

Incorrect "." and ".." Entries

The directory inode number entry for "`.`" must be the first entry in the directory data block. It must reference itself; that is, its value must be equal to the inode number for the directory data block.

The directory inode number entry for "`..`" must be the second entry in the directory data block. Its value must be equal to the inode number of the parent directory (or the inode number of itself if the directory is the root directory).

If the directory inode numbers for "`.`" and "`..`" are incorrect, `fsck` replaces them with the correct values. If there are multiple hard links to a directory, the first one found is considered the real parent to which "`..`" should point. In this case, `fsck` recommends you have it delete the other names.

Disconnected Directories

The fsck program checks the general connectivity of the file system. If a directory is found that is not linked to the file system, fsck links the directory to the lost+found directory of the file system. (This condition can occur when inodes are written to the file system but the corresponding directory data blocks are not.)

Regular Data Blocks

Data blocks associated with a regular file hold the contents of the file. fsck does not attempt to check the validity of the contents of a regular file's data blocks.

Error Messages

Normally, fsck is run non-interactively to *preen* the file systems after an abrupt system halt in which the latest file system changes were not written to disk. Preening automatically fixes any basic file system inconsistencies and does not try to repair more serious errors. While preening a file system, fsck fixes the inconsistencies it expects from such an abrupt halt. For more serious conditions, the command reports the error and terminates.

When you run fsck interactively, fsck reports each inconsistency found and fixes innocuous errors. However, for more serious errors, the command reports the inconsistency and prompts you to choose a response. When you run fsck using the -y or -n options, your response is predefined as yes or no to the default response suggested by fsck for each error condition.

Some corrective actions will result in some loss of data. The amount and severity of data loss may be determined from the fsck diagnostic output.

fsck is a multipass file system check program. Each pass invokes a different phase of the fsck program with different sets of messages. After initialization, fsck performs successive passes over each file system, checking blocks and sizes, path names, connectivity, reference counts, and the map of free blocks (possibly rebuilding it). It also performs some cleanup.

The phases (passes) performed by the UFS version of fsck are:

- Initialization
- Phase 1 – Check blocks and sizes

- Phase 2 – Check path names
- Phase 3 – Check connectivity
- Phase 4 – Check reference counts
- Phase 5 – Check cylinder groups

The next sections describe the error conditions that may be detected in each phase, the messages and prompts that result, and possible responses you can make.

Messages that may appear in more than one phase are described in "General fsck Error Messages" on page 644. Otherwise, messages are organized alphabetically by the phases in which they occur.

Many of the messages include the abbreviations shown in Table 39-3:

Table 39-3 Error Message Abbreviations

Abbreviation	**Meaning**
`BLK`	Block number
`DUP`	Duplicate block number
`DIR`	Directory name
`CG`	Cylinder group
`MTIME`	Time file was last modified
`UNREF`	Unreferenced

Many of the messages also include variable fields, such as inode numbers, which are represented in this book by an italicized term, such as *inode-number*. For example, this screen message:

```
INCORRECT BLOCK COUNT I=2529
```

is shown as:

`INCORRECT BLOCK COUNT I=`*inode-number*

General `fsck` *Error Messages*

The error messages in this section may be displayed in any phase after initialization. Although they offer the option to continue, it is generally best to regard them as fatal. They reflect a serious system failure and should be handled immediately. When confronted with such a message, terminate the program by entering no. If you cannot determine what caused the problem, contact your local service provider or another qualified person.

```
CANNOT SEEK: BLK block-number (CONTINUE)
```

A request to move to a specified block number block-number in the file system failed. This message indicates a serious problem, probably a hardware failure.

If you want to continue the file system check, do a second run of `fsck` to recheck the file system. If the block was part of the virtual memory buffer cache, `fsck` will terminate with a fatal I/O error message.

```
CANNOT READ: BLK block-number (CONTINUE)
```

A request to read a specified block number in the file system failed. The message indicates a serious problem, probably a hardware failure. If you want to continue the file system check, `fsck` will retry the read and display a list of sector numbers that could not be read.

If `fsck` tries to write back one of the blocks on which the read failed it displays this message:

```
WRITING ZERO'ED BLOCK sector-numbers TO DISK
```

If the disk is experiencing hardware problems, the problem will persist. Run `fsck` again to recheck the file system. If the block was part of the virtual memory buffer cache, `fsck` terminates and displays this error message:

```
Fatal I/O error
```

```
CANNOT WRITE: BLK block-number (CONTINUE)
```

A request to write a specified block number *block-number* in the file system failed. The disk may be write-protected. Check the write-protect lock on the drive. If that is not the problem, contact your local service provider or another qualified person. If you continue the file system check, the write operation will be retried. Sectors that could not be written are shown with this message:

```
THE FOLLOWING SECTORS COULD NOT BE WRITTEN: sector-numbers
```

where *sector-numbers* indicates the sectors that could not be written. If the disk has hardware problems, the problem will persist. This error condition prevents a complete check of the file system. Run `fsck` a second time to recheck this file system. If the block was part of the virtual memory buffer cache, `fsck` terminates and displays this error message:

```
Fatal I/O error
```

Initialization Phase `fsck` *Messages*

In the initialization phase, command-line syntax is checked. Before the file system check can be performed, `fsck` sets up tables and opens files.

The messages in this section relate to error conditions resulting from command-line options, memory requests, the opening of files, the status of files, file system size checks, and the creation of the scratch file. All such initialization errors terminate `fsck` when it is preening the file system.

```
bad inode number inode-number to ginode
```

Reason Error Occurred

An internal error occurred because of a nonexistent inode *inode-number*. `fsck` exits.

How to Solve the Problem

Contact your local service provider or another qualified person.

```
cannot alloc size-of-block map bytes for blockmap
cannot alloc size-of-free map bytes for freemap
cannot alloc size-of-state map bytes for statemap
cannot alloc size-of-lncntp bytes for lncntp
```

Reason Error Occurred

Request for memory for its internal tables failed. `fsck` terminates. This message indicates a serious system failure that should be handled immediately. This condition may occur if other processes are using a very large amount of system resources.

How to Solve the Problem

Killing other processes may solve the problem. If not, contact your local service provider or another qualified person.

```
Can’t open checklist file: filename
```

Reason Error Occurred

The file system checklist file *filename* (usually `/etc/vfstab`) cannot be opened for reading. `fsck` terminates.

How to Solve the Problem

Check if the file exists and if its access modes permit read access.

```
Can’t open filename
```

Reason Error Occurred

`fsck` cannot open file system *filename*. When running interactively, `fsck` ignores this file system and continues checking the next file system given.

How to Solve the Problem

Check to see if read and write access to the raw device file for the file system is permitted.

```
Can’t stat root
```

Reason Error Occurred

`fsck` request for statistics about the root directory failed. `fsck` terminates.

How to Solve the Problem

This message indicates a serious system failure. Contact your local service provider or another qualified person.

```
Can't stat filename
Can't make sense out of name filename
```

Reason Error Occurred

fsck request for statistics about the file system *filename* failed. When running interactively, fsck ignores this file system and continues checking the next file system given.

How to Solve the Problem

Check if the file system exists and check its access modes.

```
filename: (NO WRITE)
```

Reason Error Occurred

Either the -n option was specified or fsck could not open the file system *filename* for writing. When fsck is running in no-write mode, all diagnostic messages are displayed, but fsck does not attempt to fix anything.

How to Solve the Problem

If -n was not specified, check the type of the file specified. It may be the name of a regular file.

```
IMPOSSIBLE MINFREE=percent IN SUPERBLOCK (SET TO DEFAULT)
```

Reason Error Occurred

The superblock minimum space percentage is greater than 99 percent or less than 0 percent.

How to Solve the Problem

To set the minfree parameter to the default 10 percent, type y at the default prompt.
To ignore the error condition, type n at the default prompt.

```
INTERNAL INCONSISTENCY: message
```

Reason Error Occurred	How to Solve the Problem

`fsck` has had an internal error, whose message is *message*.	If one of the following messages are displayed, contact your local service provider or another qualified person:
	
	`MAGIC NUMBER WRONG`
	`NCG OUT OF RANGE`
	`CPG OUT OF RANGE`
	`NCYL DOES NOT JIBE WITH NCG*CPG`
	`SIZE PREPOSTEROUSLY LARGE`
	`TRASHED VALUES IN SUPER BLOCK`

This message may be followed by an error in the following example:

```
filename: BAD SUPER BLOCK: block-number
USE AN ALTERNATE SUPER-BLOCK TO SUPPLY NEEDED INFORMATION;
e.g., fsck[-f ufs] -o b=# [special ...]
where # is the alternate superblock.  See fsck_ufs(1M)
```

Reason Error Occurred	How to Solve the Problem
The superblock has been corrupted.	Use an alternative superblock to supply needed information. Specifying block 32 is a good first choice. You can locate an alternative copy of the superblock by running the `newfs -N` command on the slice. Be sure to specify the `-N` option; otherwise, `newfs` overwrites the existing file system.

```
UNDEFINED OPTIMIZATION IN SUPERBLOCK (SET TO DEFAULT)
```

Reason Error Occurred	How to Solve the Problem
The superblock optimization parameter is neither OPT_TIME nor OPT_SPACE.	To minimize the time to perform operations on the file system, type y at the SET TO DEFAULT prompt. To ignore this error condition, type n.

Phase 1: Check Blocks and Sizes Messages

This phase checks the inode list. It reports error conditions encountered while:

- Checking inode types
- Setting up the zero-link-count table
- Examining inode block numbers for bad or duplicate blocks
- Checking inode size
- Checking inode format

All errors in this phase except INCORRECT BLOCK COUNT, PARTIALLY TRUNCATED INODE, PARTIALLY ALLOCATED INODE, and UNKNOWN FILE TYPE terminate fsck when it is preening a file system.

The other possible error messages displayed in this phase are referenced below.

- *BAD STATE state-number TO BLKERR*
- *block-number DUP I=inode-number*
- *EXCESSIVE BAD BLOCKS I=inode-number (CONTINUE)*
- *EXCESSIVE DUP BLKS I=inode-number (CONTINUE)*
- *INCORRECT BLOCK COUNT I=inode-number (number-of-BAD-DUP-or-missing-blocks should be number-of-blocks-in-filesystem) (CORRECT)*
- *LINK COUNT TABLE OVERFLOW (CONTINUE)*
- *PARTIALLY ALLOCATED INODE I=inode-number (CLEAR)*
- *PARTIALLY TRUNCATED INODE I=inode-number (SALVAGE)*
- *UNKNOWN FILE TYPE I=inode-number (CLEAR)*

These messages (in alphabetical order) may occur in phase 1:

block-number `BAD I=`*inode-number*

Reason Error Occurred	How to Solve the Problem
Inode *inode-number* contains a block number *block-number* with a number lower than the number of the first data block in the file system or greater than the number of the last block in the file system. This error condition may generate the `EXCESSIVE BAD BLKS` error message in phase 1 if inode *inode-number* has too many block numbers outside the file system range. This error condition generates the `BAD/DUP` error message in phases 2 and 4.	N/A

`BAD MODE: MAKE IT A FILE?`

Reason Error Occurred	How to Solve the Problem
The status of a given inode is set to all 1s, indicating file system damage. This message does not indicate physical disk damage, unless it is displayed repeatedly after `fsck -y` has been run.	Type `y` to reinitialize the inode to a reasonable value.

`BAD STATE` *state-number* `TO BLKERR`

Reason Error Occurred	How to Solve the Problem
An internal error has scrambled the `fsck` state map so that it shows the impossible value *state-number*. `fsck` exits immediately.	Contact your local service provider or another qualified person.

block-number `DUP I=`*inode-number*

Reason Error Occurred	How to Solve the Problem
Inode *inode-number* contains a block number *block-number*, which is already claimed by the same or another inode. This error condition may generate the `EXCESSIVE DUP BLKS` error message in phase 1 if inode *inode-number* has too many block numbers claimed by the same or another inode. This error condition invokes phase 1B and generates the `BAD/DUP` error messages in phases 2 and 4.	N/A

`DUP TABLE OVERFLOW (CONTINUE)`

Reason Error Occurred	How to Solve the Problem
There is no more room in an internal table in `fsck` containing duplicate block numbers. If the `-o p` option is specified, the program terminates.	To continue the program, type `y` at the `CONTINUE` prompt. When this error occurs, a complete check of the file system is not possible. If another duplicate block is found, this error condition repeats. Increase the amount of virtual memory available (by killing some processes, increasing swap space) and run `fsck` again to recheck the file system. To terminate the program, type `n`.

EXCESSIVE BAD BLOCKS I=*inode-number* (CONTINUE)

Reason Error Occurred

Too many (usually more than 10) blocks have a number lower than the number of the first data block in the file system or greater than the number of the last block in the file system associated with inode *inode-number*. If the -o p (preen) option is specified, the program terminates.

How to Solve the Problem

To continue the program, type y at the CONTINUE prompt. When this error occurs, a complete check of the file system is not possible. You should run fsck again to recheck the file system.
To terminate the program, type n.

EXCESSIVE DUP BLKS I=*inode-number* (CONTINUE)

Reason Error Occurred

Too many (usually more than 10) blocks are claimed by the same or another inode or by a free-list. If the -o p option is specified, the program terminates.

How to Solve the Problem

To continue the program, type y at the CONTINUE prompt. When this error occurs, a complete check of the file system is not possible. You should run fsck again to recheck the file system.
To terminate the program, type n.

INCORRECT BLOCK COUNT I=*inode-number* (*number-of-BAD-DUP-or-missing-blocks* should be *number-of-blocks-in-filesystem*) (CORRECT)

Reason Error Occurred

The block count for inode *inode-number* is *number-of-BAD-DUP-or-missing-blocks*, but should be *number-of-blocks-in-filesystem*. When preening, fsck corrects the count.

How to Fix the Problem

To replace the block count of inode *inode-number* by *number-of-blocks-in-filesystem*, type y at the CORRECT prompt.
To terminate the program, type n.

```
LINK COUNT TABLE OVERFLOW (CONTINUE)
```

Reason Error Occurred

There is no more room in an internal table for `fsck` containing allocated inodes with a link count of zero. If the `-o p` (preen) option is specified, the program exits and `fsck` has to be completed manually.

How to Fix the Problem

To continue the program, type `y` at the `CONTINUE` prompt. If another allocated inode with a zero-link count is found, this error condition repeats. When this error occurs, a complete check of the file system is not possible. You should run `fsck` again to recheck the file system. Increase the virtual memory available by killing some processes or increasing swap space, then run `fsck` again. To terminate the program, type `n`.

`PARTIALLY ALLOCATED INODE I=`*inode-number* `(CLEAR)`

Reason Error Occurred

Inode *inode-number* is neither allocated nor unallocated. If the `-o p` (preen) option is specified, the inode is cleared.

How to Solve the Problem

To deallocate the inode *inode-number* by zeroing out its contents, type `y`. This may generate the `UNALLOCATED` error condition in phase 2 for each directory entry pointing to this inode.
To ignore the error condition, type `n`. A no response is appropriate only if you intend to take other measures to fix the problem.

`PARTIALLY TRUNCATED INODE I=`*inode-number* `(SALVAGE)`

Reason Error Occurred

`fsck` has found inode *inode-number* whose size is shorter than the number of blocks allocated to it. This condition occurs only if the system crashes while truncating a file. When preening the file system, `fsck` completes the truncation to the specified size.

How to Solve the Problem

To complete the truncation to the size specified in the inode, type `y` at the `SALVAGE` prompt.
To ignore this error condition, type `n`.

`UNKNOWN FILE TYPE I=`*inode-number* `(CLEAR)`

Reason Error Occurred

The mode word of the inode *inode-number* shows that the inode is not a pipe, special character inode, special block inode, regular inode, symbolic link, FIFO file, or directory inode. If the `-o p` option is specified, the inode is cleared.

How to Solve the Problem

To deallocate the inode *inode-number* by zeroing its contents, which results in the `UNALLOCATED` error condition in phase 2 for each directory entry pointing to this inode, type `y` at the `CLEAR` prompt.
To ignore this error condition, type `n`.

Phase 1B: Rescan for More DUPS Messages

When a duplicate block is found in the file system, this message is displayed:

```
block-number DUP I=inode-number
```

Reason Error Occurred	How to Solve the Problem
Inode *inode-number* contains a block number *block-number* that is already claimed by the same or another inode. This error condition generates the BAD/DUP error message in phase 2. Inodes that have overlapping blocks may be determined by examining this error condition and the DUP error condition in phase 1.	When a duplicate block is found, the file system is rescanned to find the inode that previously claimed that block.

Phase 2: Check Path Names Messages

This phase removes directory entries pointing to bad inodes found in phases 1 and 1B. It reports error conditions resulting from:

- Incorrect root inode mode and status
- Directory inode pointers out of range
- Directory entries pointing to bad inodes
- Directory integrity checks

When the file system is being preened (`-o p` option), all errors in this phase terminate `fsck`, except those related to directories not being a multiple of the block size, duplicate and bad blocks, inodes out of range, and extraneous hard links.

Other possible error messages displayed in this phase are referenced below.

- *BAD INODE state-number TO DESCEND*
- *BAD INODE NUMBER FOR '.' I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename (FIX)*
- *BAD INODE NUMBER FOR '..' I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename (FIX)*
- *BAD RETURN STATE state-number FROM DESCEND*
- *BAD STATE state-number FOR ROOT INODE*
- *BAD STATE state-number FOR INODE=inode-number*

- *DIRECTORY TOO SHORT I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename (FIX)*
- *DIRECTORY filename: LENGT file-size NOT MULTIPLE OF block-number (ADJUST)*
- *DIRECTORY CORRUPTED I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename (SALVAGE)*
- *DUP/BAD I=inode-number OWNER=O MODE=M SIZE=file-size MTIME=modification-time TYPE=filename (REMOVE)*
- *DUPS/BAD IN ROOT INODE (REALLOCATE)*
- *EXTRA '.' ENTRY I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename (FIX)*
- *EXTRA '..' ENTRY I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename (FIX)*
- *hard-link-number IS AN EXTRANEOUS HARD LINK TO A DIRECTORY filename (REMOVE)*
- *inode-number OUT OF RANGE I=inode-number NAME=filename (REMOVE)*
- *MISSING '.' I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename (FIX)*
- *MISSING '.' I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename CANNOT FIX, FIRST ENTRY IN DIRECTORY CONTAINS filename*
- *MISSING '.' I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename CANNOT FIX, INSUFFICIENT SPACE TO ADD '.'*
- *MISSING '..' I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename (FIX)*
- *MISSING '..' I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename CANNOT FIX, SECOND ENTRY IN DIRECTORY CONTAINS filename*
- *MISSING '..' I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename CANNOT FIX, INSUFFICIENT SPACE TO ADD '..'*
- *NAME TOO LONG filename*
- *ROOT INODE UNALLOCATED (ALLOCATE)*
- *ROOT INODE NOT DIRECTORY (REALLOCATE)*
- *UNALLOCATED I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time type=filename (REMOVE)*
- *ZERO LENGTH DIRECTORY I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename (REMOVE)*

`BAD INODE` *state-number* `TO DESCEND`

Reason Error Occurred

An `fsck` internal error has passed an invalid state *state-number* to the routine that descends the file system directory structure. `fsck` exits.

How to Solve the Problem

If this error message is displayed, contact your local service provider or another qualified person.

`BAD INODE NUMBER FOR '.' I=`*inode-number* `OWNER=`*UID* `MODE=`*file-mode* `SIZE=`*file-size* `MTIME=`*modification-time* `DIR=`*filename* `(FIX)`

Reason Error Occurred

A directory *inode-number* has been found whose inode number for "." does not equal *inode-number*.

How to Solve the Problem

To change the inode number for "`.`" to be equal to *inode-number*, type `y` at the `FIX` prompt.
To leave the inode numbers for "." unchanged, type `n`.

`BAD INODE NUMBER FOR '..' I=`*inode-number* `OWNER=`*UID* `MODE=`*file-mode* `SIZE=`*file-size* `MTIME=`*modification-time* `DIR=`*filename* `(FIX)`

Reason Error Occurred

A directory *inode-number* has been found whose inode number for "`..`" does not equal the parent of *inode-number*.

How to Solve the Problem

To change the inode number for "`..`" to be equal to the parent of *inode-number*, type `y` at the `FIX` prompt. (Note that "`..`" in the root inode points to itself.)
To leave the inode number for "`..`" unchanged, type `n`.

`BAD RETURN STATE` *state-number* `FROM DESCEND`

Reason Error Occurred

An `fsck` internal error has returned an impossible state *state-number* from the routine that descends the file system directory structure. `fsck` exits.

How to Solve the Problem

If this message is displayed, contact your local service provider or another qualified person.

`BAD STATE` *state-number* `FOR ROOT INODE`

Reason Error Occurred

An internal error has assigned an impossible state *state-number* to the root inode. `fsck` exits.

How to Solve the Problem

If this error message is displayed, contact your local service provider or another qualified person.

`BAD STATE` *state-number* `FOR INODE=`*inode-number*

Reason Error Occurred

An internal error has assigned an impossible state *state-number* to inode *inode-number*. `fsck` exits.

How to Solve the Problem

If this error message is displayed, contact your local service provider or another qualified person.

`DIRECTORY TOO SHORT I=`*inode-number* `OWNER=`*UID* `MODE=`*file-mode* `SIZE=`*file-size* `MTIME=`*modification-time* `DIR=`*filename* `(FIX)`

Reason Error Occurred

A directory *filename* has been found whose size *file-size* is less than the minimum directory size. The owner *UID*, mode *file-mode*, size *file-size*, modify time *modification-time*, and directory name *filename* are displayed.

How to Solve the Problem

To increase the size of the directory to the minimum directory size, type `y` at the `FIX` prompt.
To ignore this directory, type `n`.

`DIRECTORY` *filename*`: LENGT` *file-size* `NOT MULTIPLE OF` *block-number* `(ADJUST)`

Reason Error Occurred

A directory *filename* has been found with size *file-size* that is not a multiple of the directory block size *block-number*.

How to Solve the Problem

To round up the length to the appropriate block size, type `y`. When preening the file system (`-o p` option), `fsck` only displays a warning and adjusts the directory.
To ignore this condition, type `n`.

```
DIRECTORY CORRUPTED I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time
DIR=filename (SALVAGE)
```

Reason Error Occurred	How to Solve the Problem
A directory with an inconsistent internal state has been found.	To throw away all entries up to the next directory boundary (usually a 512-byte boundary), type `y` at the `SALVAGE` prompt. This drastic action can throw away up to 42 entries. Take this action only after other recovery efforts have failed. To skip to the next directory boundary and resume reading, but not modify the directory, type `n`.

```
DUP/BAD I=inode-number OWNER=O MODE=M SIZE=file-size MTIME=modification-time TYPE=filename (REMOVE)
```

Reason Error Occurred	How to Solve the Problem
Phase 1 or phase 1B found duplicate blocks or bad blocks associated with directory or file entry *filename*, inode *inode-number*. The owner *UID*, mode *file-mode*, size *file-size*, modification time *modification-time*, and directory or file name *filename* are displayed. If the `-p` (preen) option is specified, the duplicate/bad blocks are removed.	To remove the directory or file entry *filename*, type `y` at the `REMOVE` prompt. To ignore this error condition, type `n`.

```
DUPS/BAD IN ROOT INODE (REALLOCATE)
```

Reason Error Occurred

Phase 1 or phase 1B has found duplicate blocks or bad blocks in the root inode (usually inode number 2) of the file system.

How to Solve the Problem

To clear the existing contents of the root inode and reallocate it, type `y` at the `REALLOCATE` prompt. The files and directories usually found in the root will be recovered in phase 3 and put into the `lost+found` directory. If the attempt to allocate the root fails, `fsck` will exit with: `CANNOT ALLOCATE ROOT INODE`.
To get the `CONTINUE` prompt, type `n`. To respond to the `CONTINUE` prompt, type:
`y` – To ignore the `DUPS/BAD` error condition in the root inode and continue running the file system check. If the root inode is not correct, this may generate many other error messages.
`n` – To terminate the program.

EXTRA '.' ENTRY I=*inode-number* OWNER=*UID* MODE=*file-mode* SIZE=*file-size* MTIME=*modification-time* DIR=*filename* (FIX)

Reason Error Occurred

A directory *inode-number* has been found that has more than one entry for "`.`".

How to Solve the Problem

To remove the extra entry for "`.`" type `y` at the `FIX` prompt.
To leave the directory unchanged, type `n`.

```
EXTRA '..' ENTRY I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename
(FIX)
```

Reason Error Occurred

A directory *inode-number* has been found that has more than one entry for "`..`" (the parent directory).

How to Solve the Problem

To remove the extra entry for '`..`' (the parent directory), type `y` at the `FIX` prompt. To leave the directory unchanged, type `n`.

```
hard-link-number IS AN EXTRANEOUS HARD LINK TO A DIRECTORY filename (REMOVE)
```

Reason Error Occurred

`fsck` has found an extraneous hard link *hard-link-number* to a directory *filename*. When preening (`-o p` option), `fsck` ignores the extraneous hard links.

How to Solve the Problem

To delete the extraneous entry *hard-link-number* type `y` at the `REMOVE` prompt. To ignore the error condition, type `n`.

```
inode-number OUT OF RANGE I=inode-number NAME=filename (REMOVE)
```

Reason Error Occurred

A directory entry *filename* has an inode number *inode-number* that is greater than the end of the inode list. If the `-p` (preen) option is specified, the inode will be removed automatically.

How to Solve the Problem

To delete the directory entry *filename* type `y` at the `REMOVE` prompt. To ignore the error condition, type `n`.

```
MISSING '.' I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename
(FIX)
```

Reason Error Occurred

A directory *inode-number* has been found whose first entry (the entry for “.”) is unallocated.

How to Solve the Problem

To build an entry for “.” with inode number equal to *inode-number*, type `y` at the `FIX` prompt. To leave the directory unchanged, type `n`.

```
MISSING '.' I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename
CANNOT FIX, FIRST ENTRY IN DIRECTORY CONTAINS filename
```

Reason Error Occurred

A directory *inode-number* has been found whose first entry is *filename*. `fsck` cannot resolve this problem.

How to Solve the Problem

Mount the file system and move entry *filename* elsewhere. Unmount the file system and run `fsck` again.

```
MISSING '.' I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename
CANNOT FIX, INSUFFICIENT SPACE TO ADD '.'
```

Reason Error Occurred

A directory *inode-number* has been found whose first entry is not “.”. `fsck` cannot resolve the problem.

How to Solve the Problem

If this error message is displayed, contact your local service provider or another qualified person.

```
MISSING '..' I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename
(FIX)
```

Reason Error Occurred

A directory *inode-number* has been found whose second entry is unallocated.

How to Solve the Problem

To build an entry for "`..`" with inode number equal to the parent of *inode-number*, type `y` at the `FIX` prompt. (Note that "`..`" in the root inode points to itself.) To leave the directory unchanged, type `n`.

```
MISSING '..' I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename
CANNOT FIX, SECOND ENTRY IN DIRECTORY CONTAINS filename
```

Reason Error Occurred

A directory *inode-number* has been found whose second entry is *filename*. `fsck` cannot resolve this problem.

How to Solve the Problem

Mount the file system and move entry *filename* elsewhere. Then unmount the file system and run `fsck` again.

```
MISSING '..' I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time DIR=filename
CANNOT FIX, INSUFFICIENT SPACE TO ADD '..'
```

Reason Error Occurred

A directory *inode-number* has been found whose second entry is not "`..`" (the parent directory). `fsck` cannot resolve this problem.

How to Solve the Problem

Mount the file system and move the second entry in the directory elsewhere. Then unmount the file system and run `fsck` again.

`NAME TOO LONG` *filename*

Reason Error Occurred	How to Solve the Problem
An excessively long path name has been found, which usually indicates loops in the file system name space. This error can occur if a privileged user has made circular links to directories.	Remove the circular links.

`ROOT INODE UNALLOCATED (ALLOCATE)`

Reason Error Occurred	How to Solve the Problem
The root inode (usually inode number 2) has no allocate-mode bits.	To allocate inode 2 as the root inode, type `y` at the `ALLOCATE` prompt. The files and directories usually found in the root will be recovered in phase 3 and put into the `lost+found` directory. If the attempt to allocate the root fails, `fsck` displays this message and exits: `CANNOT ALLOCATE ROOT INODE`. To terminate the program, type `n`.

```
ROOT INODE NOT DIRECTORY (REALLOCATE)
```

Reason Error Occurred

The root inode (usually inode number 2) of the file system is not a directory inode.

How to Solve the Problem

To clear the existing contents of the root inode and reallocate it, type `y` at the `REALLOCATE` prompt. The files and directories usually found in the root will be recovered in phase 3 and put into the `lost+found` directory. If the attempt to allocate the root fails, `fsck` displays this message and exits:
`CANNOT ALLOCATE ROOT INODE`
To have `fsck` prompt with `FIX`, type `n`.

`UNALLOCATED I=`*inode-number* `OWNER=`*UID* `MODE=`*file-mode* `SIZE=`*file-size* `MTIME=`*modification-time* `type=`*filename*
`(REMOVE)`

Reason Error Occurred

A directory or file entry *filename* points to an unallocated inode *inode-number*. The owner *UID*, mode *file-mode*, size *file-size*, modify time *modification-time*, and file name *filename* are displayed.

How to Solve the Problem

To delete the directory entry *filename*, type `y` at the `REMOVE` prompt. To ignore the error condition, type `n`.

ZERO LENGTH DIRECTORY I=*inode-number* OWNER=*UID* MODE=*file-mode* SIZE=*file-size* MTIME=*modification-time* DIR=*filename* (REMOVE)

Reason Error Occurred

A directory entry *filename* has a size *file-size* that is zero. The owner *UID*, mode *file-mode*, size *file-size*, modify time *modification-time*, and directory name *filename* are displayed.

How to Solve the Problem

To remove the directory entry *filename*, type y at the REMOVE prompt. This results in the BAD/DUP error message in phase 4. To ignore the error condition, type n.

Phase 3: Check Connectivity Messages

This phase checks the directories examined in phase 2 and reports error conditions resulting from:

- Unreferenced directories
- Missing or full lost+found directories

Other possible error messages displayed in this phase are referenced below.

- *BAD INODE state-number TO DESCEND*
- *DIR I=inode-number1 CONNECTED. PARENT WAS I=inode-number2*
- *DIRECTORY filename LENGTH file-size NOT MULTIPLE OF block-number (ADJUST)*
- *lost+found IS NOT A DIRECTORY (REALLOCATE)*
- *NO lost+found DIRECTORY (CREATE)*
- *NO SPACE LEFT IN /lost+found (EXPAND)*
- *UNREF DIR I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time (RECONNECT)*

`BAD INODE` *state-number* `TO DESCEND`

Reason Error Occurred	How to Solve the Problem
An internal error has caused an impossible state *state-number* to be passed to the routine that descends the file system directory structure. `fsck` exits.	If this occurs, contact your local service provider or another qualified person.

`DIR I=`*inode-number1* `CONNECTED. PARENT WAS I=`*inode-number2*

Reason Error Occurred	How to Solve the Problem
This is an advisory message indicating a directory inode *inode-number1* was successfully connected to the `lost+found` directory. The parent inode *inode-number2* of the directory inode *inode-number1* is replaced by the inode number of the `lost+found` directory.	N/A

`DIRECTORY` *filename* `LENGTH` *file-size* `NOT MULTIPLE OF` *block-number* `(ADJUST)`

Reason Error Occurred	How to Solve the Problem
A directory *filename* has been found with size *file-size* that is not a multiple of the directory block size B. (This condition can recur in phase 3 if it is not adjusted in phase 2.)	To round up the length to the appropriate block size, type `y` at the `ADJUST` prompt. When preening, `fsck` displays a warning and adjusts the directory. To ignore this error condition, type `n`.

```
lost+found IS NOT A DIRECTORY (REALLOCATE)
```

Reason Error Occurred

The entry for lost+found is not a directory.

How to Solve the Problem

To allocate a directory inode and change the lost+found directory to reference it, type y at the REALLOCATE prompt. The previous inode reference by the lost+found directory is not cleared and it will either be reclaimed as an unreferenced inode or have its link count adjusted later in this phase. Inability to create a lost+found directory displays the message:
SORRY. CANNOT CREATE lost+found DIRECTORY
and aborts the attempt to link up the lost inode, which generates the UNREF error message in phase 4.
To abort the attempt to link up the lost inode, which generates the UNREF error message in phase 4, type n.

```
NO lost+found DIRECTORY (CREATE)
```

Reason Error Occurred

There is no lost+found directory in the root directory of the file system. When preening, fsck tries to create a lost+found directory.

How to Solve the Problem

To create a lost+found directory in the root of the file system, type y at the CREATE prompt. This may lead to the message NO SPACE LEFT IN / (EXPAND). If the lost+found directory cannot be created, fsck displays the message:
SORRY. CANNOT CREATE lost+found DIRECTORY
and aborts the attempt to link up the lost inode. This in turn generates the UNREF error message later in phase 4.
To abort the attempt to link up the lost inode, type n.

```
NO SPACE LEFT IN /lost+found (EXPAND)
```

Reason Error Occurred	How to Solve the Problem
Another entry cannot be added to the `lost+found` directory in the root directory of the file system because no space is available. When preening, `fsck` expands the `lost+found` directory.	To expand the `lost+found` directory to make room for the new entry, type y at the `EXPAND` prompt. If the attempted expansion fails, `fsck` displays: `SORRY. NO SPACE IN lost+found DIRECTORY` and aborts the request to link a file to the `lost+found` directory. This error generates the `UNREF` error message later in phase 4. Delete any unnecessary entries in the `lost+found` directory. This error terminates `fsck` when preening is in effect. To abort the attempt to link up the lost inode, type `n`..

`UNREF DIR I=`*inode-number* `OWNER=`*UID* `MODE=`*file-mode* `SIZE=`*file-size* `MTIME=`*modification-time* `(RECONNECT)`

Reason Error Occurred	How to Solve the Problem
The directory inode *inode-number* was not connected to a directory entry when the file system was traversed. The owner *UID*, mode *file-mode*, size *file-size*, and modification time *modification-time* of directory inode *inode-number* are displayed. When preening, `fsck` reconnects the non-empty directory inode if the directory size is non-zero. Otherwise, `fsck` clears the directory inode.	To reconnect the directory inode *inode-number* into the `lost+found` directory, type `y` at the `RECONNECT` prompt. If the directory is successfully reconnected, a `CONNECTED` message is displayed. Otherwise, one of the `lost+found` error messages is displayed. To ignore this error condition, type n. This error causes the UNREF error condition in phase 4.

Phase 4: Check Reference Counts Messages

This phase checks the link count information obtained in phases 2 and 3. It reports error conditions resulting from:

- Unreferenced files
- A missing or full `lost+found` directory

- Incorrect link counts for files, directories, symbolic links, or special files
- Unreferenced files, symbolic links, and directories
- Bad or duplicate blocks in files and directories
- Incorrect total free-inode counts

All errors in this phase (except running out of space in the `lost+found` directory) are correctable when the file system is being preened.

Other possible error messages displayed in this phase are referenced below.

- *BAD/DUP type I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time (CLEAR)*
- *(CLEAR)*
- *LINK COUNT type I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time COUNT link-count SHOULD BE corrected-link-count (ADJUST)*
- *lost+found IS NOT A DIRECTORY (REALLOCATE)*
- *NO lost+found DIRECTORY (CREATE)*
- *NO SPACE LEFT IN /lost+found (EXPAND)*
- *UNREF FILE I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time (RECONNECT)*
- *UNREF type I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time (CLEAR)*
- *ZERO LENGTH DIRECTORY I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time (CLEAR)*

`BAD/DUP` *type* `I=`*inode-number* `OWNER=`*UID* `MODE=`*file-mode* `SIZE=`*file-size* `MTIME=`*modification-time* `(CLEAR)`

Reason Error Occurred

Phase 1 or phase 1B found duplicate blocks or bad blocks associated with file or directory inode *inode-number*. The owner *UID*, mode *file-mode*, size *file-size*, and modification time *modification-time* of inode *inode-number* are displayed.

How to Solve the Problem

To deallocate inode *inode-number* by zeroing its contents, type `y` at the `CLEAR` prompt.
To ignore this error condition, type `n`.

```
(CLEAR)
```

Reason Error Occurred

The inode mentioned in the `UNREF` error message immediately preceding cannot be reconnected. This message does not display if the file system is being preened because lack of space to reconnect files terminates `fsck`.

How to Solve the Problem

To deallocate the inode by zeroing out its contents, type `y` at the `CLEAR` prompt.
To ignore the preceding error condition, type `n`.

`LINK COUNT` *type* `I=`*inode-number* `OWNER=`*UID* `MODE=`*file-mode* `SIZE=`*file-size* `MTIME=`*modification-time* `COUNT` *link-count* `SHOULD BE` *corrected-link-count* `(ADJUST)`

Reason Error Occurred

The link count for directory or file inode *inode-number* is *link-count* but should be *corrected-link-count*. The owner *UID*, mode *file-mode*, size *file-size*, and modification time *modification-time* of inode *inode-number* are displayed. If the `-o p` option is specified, the link count is adjusted unless the number of references is increasing. This condition does not occur unless there is a hardware failure. When the number of references is increasing during preening, `fsck` displays this message and exits: `LINK COUNT INCREASING`

How to Solve the Problem

To replace the link count of directory or file inode *inode-number* with *corrected-link-count*, type `y` at the `ADJUST` prompt.
To ignore this error condition, type `n`.

```
lost+found IS NOT A DIRECTORY (REALLOCATE)
```

Reason Error Occurred

The entry for `lost+found` is not a directory.

How to Solve the Problem

To allocate a directory inode and change the `lost+found` directory to reference it, type `y` at the `REALLOCATE` prompt. The previous inode reference by the `lost+found` directory is not cleared. It will either be reclaimed as an unreferenced inode or have its link count adjusted later in this phase. Inability to create a `lost+found` directory displays this message: `SORRY. CANNOT CREATE lost+found DIRECTORY` and aborts the attempt to link up the lost inode. This error generates the `UNREF` error message later in phase 4. To abort the attempt to link up the lost inode, type `n`.

```
NO lost+found DIRECTORY (CREATE)
```

Reason Error Occurred

There is no `lost+found` directory in the root directory of the file system. When preening, `fsck` tries to create a `lost+found` directory.

How to Solve the Problem

To create a `lost+found` directory in the root of the file system, type `y` at the `CREATE` prompt. If the `lost+found` directory cannot be created, `fsck` displays the message: `SORRY. CANNOT CREATE lost+found DIRECTORY` and aborts the attempt to link up the lost inode. This error in turn generates the `UNREF` error message later in phase 4.

To abort the attempt to link up the lost inode, type `n`.

```
NO SPACE LEFT IN / lost+found (EXPAND)
```

Reason Error Occurred

There is no space to add another entry to the `lost+found` directory in the root directory of the file system. When preening, `fsck` expands the `lost+found` directory.

How to Solve the Problem

To expand the `lost+found` directory to make room for the new entry, type y at the EXPAND prompt. If the attempted expansion fails, `fsck` displays the message: `SORRY. NO SPACE IN lost+found DIRECTORY` and aborts the request to link a file to the `lost+found` directory. This error generates the `UNREF` error message later in phase 4. Delete any unnecessary entries in the `lost+found` directory. This error terminates `fsck` when preening (`-o p` option) is in effect.
To abort the attempt to link up the lost inode, type `n`.

`UNREF FILE I=`*inode-number* `OWNER=`*UID* `MODE=`*file-mode* `SIZE=`*file-size* `MTIME=`*modification-time* `(RECONNECT)`

Reason Error Occurred

File inode *inode-number,* was not connected to a directory entry when the file system was traversed. The owner *UID*, mode *file-mode*, size *file-size*, and modification time *modification-time* of inode *inode-number* are displayed. When `fsck` is preening, the file is cleared if either its size or its link count is zero; otherwise, it is reconnected.

How to Solve the Problem

To reconnect inode *inode-number* to the file system in the `lost+found` directory, type `y`. This error may generate the `lost+found` error message in phase 4 if there are problems connecting inode *inode-number* to the `lost+found` directory.
To ignore this error condition, type `n`. This error always invokes the `CLEAR` error condition in phase 4.

```
UNREF type I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time (CLEAR)
```

Reason Error Occurred

Inode *inode-number* (whose *type* is directory or file) was not connected to a directory entry when the file system was traversed. The owner *UID*, mode *file-mode*, size *file-size*, and modification time *modification-time* of inode *inode-number* are displayed. When `fsck` is preening, the file is cleared if either its size or its link count is zero; otherwise, it is reconnected.

How to Solve the Problem

To deallocate inode *inode-number* by zeroing its contents, type `y` at the `CLEAR` prompt.
To ignore this error condition, type `n`.

```
ZERO LENGTH DIRECTORY I=inode-number OWNER=UID MODE=file-mode SIZE=file-size MTIME=modification-time
(CLEAR)
```

Reason Error Occurred

A directory entry *filename* has a size *file-size* that is zero. The owner *UID*, mode *file-mode*, size *file-size*, modification time *modification-time*, and directory name *filename* are displayed.

How to Solve the Problem

To deallocate the directory inode *inode-number* by zeroing out its contents, type `y`.
To ignore the error condition, type `n`.

Phase 5: Check Cylinder Groups Messages

This phase checks the free-block and used-inode maps. It reports error conditions resulting from:

- Allocated inodes missing from used-inode maps
- Free blocks missing from free-block maps
- Free inodes in the used-inode maps
- Incorrect total free-block count
- Incorrect total used inode count

The possible error messages displayed in this phase are referenced below.

- *BLK(S) MISSING IN BIT MAPS (SALVAGE)*
- *CG character-for-command-option: BAD MAGIC NUMBER*
- *FREE BLK COUNT(S) WRONG IN SUPERBLK (SALVAGE)*
- *SUMMARY INFORMATION BAD (SALVAGE)*
- *number-of files, number-of-files used, number-of-files free (number-of frags, number-of blocks, percent fragmentation)*
- ****** FILE SYSTEM WAS MODIFIED ******
- *filename FILE SYSTEM STATE SET TO OKAY*
- *filename FILE SYSTEM STATE NOT SET TO OKAY*

```
BLK(S) MISSING IN BIT MAPS (SALVAGE)
```

Reason Error Occurred	How to Solve the Problem
A cylinder group block map is missing some free blocks. During preening, `fsck` reconstructs the maps.	To reconstruct the free-block map, type `y` at the `SALVAGE` prompt. To ignore this error condition, type `n`.

`CG` *character-for-command-option*`: BAD MAGIC NUMBER`

Reason Error Occurred	How to Solve the Problem
The magic number of cylinder group *character-for-command-option* is wrong. This error usually indicates that the cylinder group maps have been destroyed. When running interactively, the cylinder group is marked as needing reconstruction. `fsck` terminates if the file system is being preened.	If this occurs, contact your local service provider or another qualified person.

```
FREE BLK COUNT(S) WRONG IN SUPERBLK (SALVAGE)
```

Reason Error Occurred

The actual count of free blocks does not match the count of free blocks in the superblock of the file system. If the `-o p` option was specified, the free-block count in the superblock is fixed automatically.

How to Solve the Problem

To reconstruct the superblock free-block information, type `y` at the `SALVAGE` prompt.
To ignore this error condition, type `n`.

```
SUMMARY INFORMATION BAD (SALVAGE)
```

Reason Error Occurred

The summary information is incorrect. When preening, `fsck` recomputes the summary information.

How to Solve the Problem

To reconstruct the summary information, type `y` at the `SALVAGE` prompt.
To ignore this error condition, type `n`.

Cleanup Phase Messages

Once a file system has been checked, a few cleanup functions are performed. The cleanup phase displays the following status messages.

number-of `files,` *number-of-files* `used,` *number-of-files* `free (`*number-of* `frags,` *number-of* `blocks,` *percent* `fragmentation)`

This message indicates that the file system checked contains *number-of* files using *number-of* fragment-sized blocks, and that there are *number-of* fragment-sized blocks free in the file system. The numbers in parentheses break the free count down into *number-of* free fragments, *number-of* free full-sized blocks, and the *percent* fragmentation.

```
***** FILE SYSTEM WAS MODIFIED *****
```

This message indicates that the file system was modified by `fsck`. If this file system is mounted or is the current root (`/`) file system, reboot. If the file system is mounted, you may need to unmount it and run `fsck` again; otherwise, the work done by `fsck` may be undone by the in-core copies of tables.

filename `FILE SYSTEM STATE SET TO OKAY`

This message indicates that file system *filename* was marked as stable. `fsck` with the `-m` option uses this information to determine that the file system does not need checking.

filename `FILE SYSTEM STATE NOT SET TO OKAY`

This message indicates that file system *filename* was *not* marked as stable. `fsck` with the `-m` option uses this information to determine that the file system needs checking.

Modifying Automatic Boot Checking

During boot up, a preliminary check on each file system to be mounted from a hard disk is run using the boot script `/sbin/rcS`, which checks the root (`/`) and `/usr` file systems. The other `rc` shell scripts then use the `fsck` command to check each additional file system sequentially. They do not check file systems in parallel. File systems are checked sequentially during booting even if the `fsck pass` numbers are greater than one.

The `/etc/vfstab` File

When you run the commands for checking and mounting file systems without specifying a file system directly, the commands step through the file system table (`/etc/vfstab`) using the information specified in the various fields. The `fsck pass` field specifies information for file system checking. The `mount at boot` field specifies information for mounting the file system at boot time.

When you create new file systems, add entries to `/etc/vfstab` indicating whether they are to be checked and mounted at boot time. See Chapter 34, "Mounting and Unmounting File Systems" for more information about adding entries to the `/etc/vfstab` file.

Information in the `/etc/vfstab` file is specific for the slices and file systems for each system. Here is an example of an `/etc/vfstab` file:

```
$ more /etc/vfstab
#device         device          mount           FS      fsck    mount    mount
#to mount       to fsck         point           type    pass    at boot  options#
#/dev/dsk/c1d0s2 /dev/rdsk/c1d0s2 /usr          ufs     1       yes      -
/proc           -               /proc           proc    -       no       -
fd              -               /dev/fd         fd      -       no       -
swap            -               /tmp            tmpfs   -       yes      -

/dev/dsk/c0t0d0s0 /dev/rdsk/c0t0d0s0 /          ufs     1       no       -
/dev/dsk/c0t0d0s1 -                  -          swap    -       no       -
/dev/dsk/c0t0d0s6 /dev/rdsk/c0t0d0s6 /usr       ufs     2       no       -
/dev/dsk/c0t0d0s7 /dev/rdsk/c0t0d0s7 /opt       ufs     3       yes      -
pluto:/export/svr4/man -    /usr/man        nfs     no      yes      -
$
```

Table 39-4 describes the function of the `fsck` pass field.

Table 39-4 `fsck pass field`

If the `fsck` pass field is Set To ...	**Then ...**	**Comments**
- (hyphen)	The generic `fsck` command will not check the file system regardless of the state of the file system.	Use a hyphen for read-only file systems, remote file systems, or pseudo file systems, such as `/proc`, to which checking does not apply.
`0` or greater	The file system specific `fsck` command is called.	When the value is greater for UFS file systems, the file system is not checked.
`1` or greater and `fsck -o p` is used	The file system specific `fsck` automatically checks UFS file systems in parallel.	The value can be any number greater than 1.

In `preen` mode, `fsck` allows only one active file system check per disk, starting a new check only after the previous one is completed. `fsck` automatically uses the major and minor numbers of the devices on which the file systems reside to determine how to check file systems on different disks at the same time.

When the `fsck pass` number is 1, file systems are checked sequentially, in the order they appear in the `/etc/vfstab` file. Usually, the root (`/`) file system has the `fsck pass` set to 1.

Note – `fsck` does *not* use the `fsck pass` number to determine the sequence of file system checking.

▼ How to Modify Automatic Checking Done During Booting

1. **Become root.**
2. **Edit** `/etc/vfstab` **entries in the** `fsck pass` **field, and save the changes.**
   The next time the system is booted, the new values are used.

Interactively Checking and Repairing a UFS File System

You may need to interactively check file systems:

- When they cannot be mounted
- When they develop problems while in use

When an in-use file system develops inconsistencies, error messages may be displayed in the console window or the system may crash.

Before using `fsck`, you may want to refer to “Syntax and Options for the fsck Command” on page 689 and “Error Messages” on page 643 for more information.

▼ How to See If a File System Needs Checking

1. **Become root.**
2. **Check the file system.**

```
# fsck -m /dev/rdsk/device-name
```

In this command, the state flag in the superblock of the file system you specify is checked to see whether the file system is clean or requires checking.

If you omit the device argument, all the UFS file systems listed in `/etc/vfstab` with a `fsck pass` value greater than 0 are checked.

Example—Seeing If a File System Needs Checking

In this example, the first file system needs checking; the second file system does not:

```
# fsck -m /dev/rdsk/c0t0d0s6
** /dev/rdsk/c0t0d0s6
ufs fsck: sanity check: /dev/rdsk/c0t0d0s6 needs checking
# fsck -m /dev/rdsk/c0t0d0s7
** /dev/rdsk/c0t0d0s7
ufs fsck: sanity check: /dev/rdsk/c0t0d0s7 okay
```

▼ How to Check File Systems Interactively

1. **Become root.**
2. **Unmount the local file system except root (/) and `/usr`.**

```
# umountall -l
```

3. **Check the file system.**

```
# fsck
```

All file systems in the `/etc/vfstab` file with entries in the `fsck pass` field greater than zero are checked. You can also specify the mount point directory or `/dev/rdsk/`*device-name* as arguments to `fsck`. Any inconsistency messages are displayed. See “Error Messages” on page 643 for information about how to respond to the error message prompts to interactively check one or more UFS file systems.

4. **If you corrected any errors, type `fsck` and press Return.**
   `fsck` may not be able to fix all errors in one execution. If you see the message `FILE SYSTEM STATE NOT SET TO OKAY`, run the command again. If that does not work, see "How to Fix a UFS File System fsck Cannot Repair" on page 688.

5. **Rename and move any files put in the `lost+found` directory.**
   Individual files put in the `lost+found` directory by `fsck` are renamed with their inode numbers. If possible, rename the files and move them where they belong. You may be able to use the `grep` command to match phrases with individual files and the `file` command to identify file types. When whole directories are dumped into `lost+found`, it is easier to figure out where they belong and move them back.

Example—Checking File Systems Interactively

In this example, `/dev/rdsk/c0t0d0s6` is checked and the incorrect block count is corrected:

```
# fsck /dev/rdsk/c0t0d0s6
checkfilesys: /dev/rdsk/c0t0d0s6
** Phase 1 - Check Block and Sizes
INCORRECT BLOCK COUNT I=2529 (6 should be 2)
CORRECT? y

** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
** Phase 5 - Cylinder Groups
929 files, 8928 used, 2851 free (75 frags, 347 blocks, 0.6%
fragmentation)
/dev/rdsk/c0t0d0s6 FILE SYSTEM STATE SET TO OKAY

***** FILE SYSTEM WAS MODIFIED *****
```

Preening UFS File Systems

The `preen` option to `fsck` (`fsck -o p`) checks UFS file systems and automatically fixes the simple problems that normally result from an unexpected system halt. It exits immediately if it encounters a problem that requires operator intervention. The `preen` option also permits parallel checking of file systems.

You can run `fsck` with the `-o p` option to preen the file systems after an unclean halt. In this mode, `fsck` does not look at the clean flag and does a full check. These actions are a subset of the actions that `fsck` takes when it runs interactively.

▼ How to Preen a File System

1. **Become root.**

2. **Unmount the file system.**

```
# umount mount-point
```

3. **Check a UFS file system with the preen option.**

```
# fsck -o p /dev/rdsk/device-name
```

You can preen individual file systems by using *mount-point* or `/dev/rdsk/`*device-name* as arguments to `fsck`.

Example—Preening a File System

The `/usr` file system is preened in this example.

```
# fsck -o p /usr
```

Restoring a Bad Superblock

When the superblock of a file system becomes damaged, you must restore it. `fsck` tells you when a superblock is bad. Fortunately, redundant copies of the superblock are stored within a file system. You can use `fsck -o b` to replace the superblock with one of the copies.

▼ How to Restore a Bad Superblock

1. **Become root.**
2. **Change to a directory outside the damaged file system.**
3. **Unmount the file system.**

```
# umount mount-point
```

Caution – Be sure to use the `-N` option with `newfs` in the next step. If you omit the `-N` option, you will create a new, empty file system.

4. **Display the superblock values with the `newfs -N` command.**

```
# newfs -N /dev/rdsk/device-name
```

The output of this command displays the block numbers that were used for the superblock copies when `newfs` created the file system.

5. **Provide an alternative superblock with the `fsck` command.**

```
# fsck -F ufs -o b=block-number /dev/rdsk/device-name
```

`fsck` uses the alternative superblock you specify to restore the primary superblock. You can always try `32` as an alternative block, or use any of the alternative blocks shown by `newfs -N`.

Example—Restoring a Bad Superblock

In this example, superblock copy 5264 is restored for the /files7 file system:

```
# cd /
# umount /files7
# newfs -N /dev/rdsk/c0t3d0s7
/dev/rdsk/c0t3d0s7: 163944 sectors in 506 cylinders of 9 tracks, 36 sectors
 83.9MB in 32 cyl groups (16 c/g, 2.65MB/g, 1216 i/g)
super-block backups (for fsck -b #) at:
 32, 5264, 10496, 15728, 20960, 26192, 31424, 36656, 41888,
 47120, 52352, 57584, 62816, 68048, 73280, 78512, 82976, 88208,
 93440, 98672, 103904, 109136, 114368, 119600, 124832, 130064, 135296,
 140528, 145760, 150992, 156224, 161456,
# fsck -F ufs -o b=5264 /dev/rdsk/c0t3d0s7
Alternate superblock location: 5264.
** /dev/rdsk/c0t3d0s7
** Last Mounted on
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
** Phase 5 - Check Cyl groups
36 files, 867 used, 75712 free (16 frags, 9462 blocks, 0.0% fragmentation)
/dev/rdsk/c0t3d0s7 FILE SYSTEM STATE SET TO OKAY

***** FILE SYSTEM WAS MODIFIED *****
#
```

If the superblock in the root (/) file system becomes damaged and you cannot boot the system, reinstall /kernel/unix and rebuild the root (/) file system with newfs. Because a superblock is created by the newfs command, you do not need to restore it.

How to Fix a UFS File System fsck Cannot Repair

Sometimes you need to run fsck a few times to fix a file system because problems corrected on one pass may uncover other problems not found in earlier passes. fsck does not keep running until it comes up clean, so you must rerun it manually.

Pay attention to the information displayed by `fsck`. It may help you fix the problem. For example, the messages may point to a bad directory. If you delete the directory, you may find that `fsck` runs cleanly.

If `fsck` still cannot repair the file system, you can try to use the `fsdb`, `ff`, `clri`, and `ncheck` commands to figure out and fix what is wrong. See `fsdb(1M)`, `ff(1M)`, `clri(1M)`, and `ncheck(1M)` for information about how to use these commands. You may, ultimately, need to re-create the file system and restore its contents from backup media. See Chapter 43, “Restoring Files and File Systems” for information about restoring complete file systems.

If you cannot fully repair a file system but you can mount it read-only, try using `cp`, `tar`, or `cpio` to retrieve all or part of the data from the file system.

If hardware disk errors are causing the problem, you may need to reformat and divide the disk into slices again before re-creating and restoring file systems. Hardware errors usually display the same error again and again across different commands. The `format(1M)` command tries to work around bad blocks on the disk. If the disk is too severely damaged, however, the problems may persist, even after reformatting. See `format(1M)` for information about using the `format` command. See Chapter 29, “SPARC: Adding a Disk” or Chapter 30, “x86: Adding a Disk” for information about installing a new disk.

Syntax and Options for the `fsck` *Command*

The `fsck` command checks and repairs inconsistencies in file systems. It has four options:

- Checks only whether a file system can be mounted (`fsck -m`)
- Interactively asks for confirmation before making repairs (`fsck`)
- Assumes yes or no response for all repairs (`fsck -y`)
- Noninteractively preens the file system, fixing all expected (innocuous) inconsistencies, but exiting when a serious problem is encountered (`fsck -o p`)

Generic `fsck` Command Syntax, Options, and Arguments

The `fsck` command has two components: a generic component and a component specific to each type of file system. The generic commands apply to most types of file systems, while the specific commands apply to only one type of file system. You should always use the generic command, which calls the file system-specific command, as needed.

Usually, you must be superuser to run `fsck`. You can run the `fsck` command without being superuser; but to make repairs, you should unmount the file system and you must have read permission for the raw device file for the slice (a potential security hole).

The generic `fsck` command goes through `/etc/vfstab` to see what file systems to check. It runs the appropriate file system-specific `fsck` command on each file system listed, except those excluded by an `fsck pass` number of `-` or `0` (UFS only).

The generic `fsck` command has the following syntax:

```
/usr/sbin/fsck [-F type] [-V] [-m] [special]

/usr/sbin/fsck [-F type] [-V] -[y|Y]|[n|N] [-o specific-options][special]
```

Table 39-5 describes the options and arguments to the generic `fsck` command.

Table 39-5 The `fsck` Command Options and Arguments *(1 of 3)*

Option Type	Option - Generic	Description
Generic	`-F`	Specifies the file system type *(type)*. If *type* is not specified on the command line, it is obtained from `/etc/vfstab` by matching an entry in that file with the *special* device name specified. If no entry is found, the default local file system type specified in `/etc/default/fs` is used.
	`-V`	Echoes the completed command line (verbose). The echoed line includes additional information derived from `/etc/vfstab`. This option can be used to verify and validate the command line. It does not execute the command.
	`-m`	Performs a preliminary check only. It returns a code indicating the state of the file system: `0` for "clean" and `32` for "dirty." This option is used by the startup script `/etc/bcheckrc` to determine whether a file system needs to be checked.
	`-y` or `-Y` or `-n` or `-N`	Runs the command automatically answering yes or no to all prompts.

Table 39-5 The `fsck` Command Options and Arguments *(2 of 3)*

Option Type	Option - Generic	Description
	`c`	Converts an old format file system with statically allocated tables to new format dynamically allocated tables. Static allocation imposes a hard maximum on table size, while dynamic allocation means space for tables can be added as needed after the initial allocation. If the file system is in the new format, convert it to the old format, unless the table allocation exceeds the fixed maximum allowed in the old format. `fsck` lists the direction of the conversion. In interactive mode, `fsck` prompts for confirmation before doing the conversion. When you use the `-o p` option, the conversion is attempted without asking for confirmation. This option is useful when you want to covert a number of file systems at once. You can determine whether a file system is in the old or new format by running the `fstyp(1M)` command, and looking at the first line displayed.
	`w`	Checks only file systems that permit write access.
	special	Specifies the mount point or raw device name of one or more file systems. An entry for the mount point must exist in `/etc/vfstab`. If you omit the *special* argument, entries in `/etc/vfstab` with a specified `fsck` device and a `fsck pass` number greater than zero are checked. If preening (`-o p`) is in effect and more than one entry has an `fsck pass` number greater than 1, file systems on different disks are checked in parallel.

Table 39-5 The `fsck` Command Options and Arguments *(3 of 3)*

Option Type	Option - Generic	Description
Specific		This is a comma-separated list of options that follow the `-o` option. Describes the options that are passed to the UFS-specific `fsck` command for interpretation.
	`p`	Preens. Runs the command automatically in silent mode, correcting what it can, but exiting when it encounters a problem that requires intervention. This option also enables parallel checking of UFS file systems.
	`b=`*blocknumber*	Uses the alternative (redundant) superblock, located at the specified location. This option can be used to repair a bad superblock. You can display a list of alternative superblocks by using the `newfs -N` command.

File System Reference 40

This is a list of the reference information in this chapter.

Default Directories for Root (/) and /usr File Systems

Table 40-1 describes all the directories contained in the default root (/) and /usr file systems.

Table 40-1 Default Directories for root (/) and /usr File Systems *(1 of 6)*

Directory	Description
Directories in the root (/) file system:	
/	Root of the overall file system name space
/dev	Primary location for special files
/dev/dsk	Block disk devices
/dev/pts	pty slave devices
/dev/rdsk	Raw disk devices

Table 40-1 Default Directories for root (/) and /usr File Systems *(2 of 6)*

Directory	Description
/dev/rmt	Raw tape devices
/dev/sad	Entry points for the STREAMS Administrative Driver
/dev/term	Terminal devices
/etc	Host-specific system administrative configuration files and databases
/etc/acct	Accounting system configuration information
/etc/cron.d	Configuration information for cron
/etc/default	Defaults information for various programs
/etc/dfs	Configuration information for exported file systems
/etc/fs	Binaries organized by file system types for operations required before /usr is mounted.
/etc/inet	Configuration files for Internet services
/etc/init.d	Scripts for changing between run levels
/etc/lp	Configuration information for the printer subsystem
/etc/mail	Mail subsystem configuration
/etc/net	Configuration information for TI (transport-independent) network services
/etc/opt	Configuration information for optional packages
/etc/rc0.d	Scripts for entering/leaving run level 0
/etc/rc1.d	Scripts for entering/leaving run level 1
/etc/rc2.d	Scripts for entering/leaving run level 2
/etc/rc3.d	Scripts for entering/leaving run level 3
/etc/rcS.d	Scripts for bringing the system up in single user mode
/etc/saf	Service access facility files (including FIFOs)
/etc/skel	Default profile scripts for new user accounts
/etc/sm	Status monitor information
/etc/sm.bak	Backup copy of status monitor information
/etc/tm	Trademark files; contents displayed at boot time

Table 40-1 Default Directories for root (/) and /usr File Systems *(3 of 6)*

Directory	Description
/etc/uucp	uucp configuration information
/export	Default root of the exported file system tree
/home	Default root of a subtree for user directories
/kernel	Subtree of platform-independent loadable kernel modules required as part of the boot process. It includes the generic part of the core kernel that is platform independent, /kernel/genunix. See Table 40-2 for the /kernel directory structure.
/mnt	Convenient, temporary mount point for file systems
/opt	Root of a subtree for add-on application packages
/opt/SUNWspro	Mount/installation point for unbundled language products
/sbin	Essential executables used in the booting process and in manual system failure recovery
/stand	Standalone programs
/tmp	Temporary files; cleared during boot sequence
/usr	Mount point for /usr file system
/var	Root of a subtree of varying files
/var/adm	System logging and accounting files
/var/crash	Default depository for kernel crash dumps
/var/cron	cron's log file
/var/lp	Line printer subsystem logging information
/var/mail	Directory where users' mail is kept
/var/news	Community service messages (*note*: not the same as USENET-style news)
/var/nis	NIS+ databases
/var/opt	Root of a subtree for varying files associated with software packages
/var/preserve	Backup files for vi and ex

Table 40-1 Default Directories for root (/) and /usr File Systems *(4 of 6)*

Directory	Description
/var/sadm	Databases maintained by the software package management utilities
/var/saf	saf (service access facility) logging and accounting files
/var/spool	Directories for spooled temporary files
/var/spool/cron	cron and at spool files
/var/spool/locks	Spooling lock files
/var/spool/lp	Line printer spool files
/var/spool/mqueue	Mail queued for delivery
/var/spool/pkg	Spooled packages
/var/spool/uucp	Queued uucp jobs
/var/spool/uucppublic	Files deposited by uucp
/var/tmp	Directory for temporary files; not cleared during boot sequence
/var/uucp	c log and status files
/var/yp	NIS databases (for backwards compatibility with NIS and unnecessary after full transition to NIS+)
Directories in the /usr file system	
bin	Location for standard system commands
demo	Demo programs and data
games	An empty directory, which is a remnant of the SunOS 4.x software
include	Header files (for C programs, etc.)
kernel	Additional modules
kvm	Implementation architecture-specific binaries and libraries

Table 40-1 Default Directories for root (/) and /usr File Systems *(5 of 6)*

Directory	Description
lib	Various program libraries, architecture-dependent databases, and binaries not invoked directly by the user
lib/acct	Accounting scripts and binaries
lib/class	Scheduling class-specific directories containing executables for priocntl and dispadmin commands
lib/font	troff font description files
lib/fs	File system type-dependent modules; not invoked directly by the user
lib/iconv	Conversion tables for iconv(1)
lib/libp	Profiled libraries
lib/locale	Internationalization localization databases
lib/lp	Line printer subsystem databases and back-end executables
lib/mail	Auxiliary programs for the mail subsystem
lib/netsvc	Internet network services
lib/nfs	Auxiliary programs and daemons related to NFS
lib/pics	PIC archives needed to build the run-time linker
lib/refer	Auxiliary refer-related programs
lib/sa	Scripts and commands for the system activity report package
lib/saf	Auxiliary programs and daemons related to the service access facility
lib/uucp	Auxiliary uucp-related programs and daemons
lib/zoneinfo	Time zone information
local	Commands local to a site
old	Programs that are being phased out
openwin	Mount/installation point for OpenWindows software
sadm	Various files and directories related to system administration; see specifics below

Table 40-1 Default Directories for root (/) and /usr File Systems *(6 of 6)*

Directory	Description
sadm/bin	"valtools" binaries for use by FMLI scripts
sadm/install	Executables and scripts for pkg management
sbin	Executables for system administration
sbin/static	Statically linked version of selected programs from /usr/bin and /usr/sbin
share	Architecture-independent sharable files
share/lib	Architecture-independent databases
share/lib/keytables	Keyboard layout description tables
share/lib/mailx	mailx-related help files
share/lib/nterm	nroff terminal tables
share/lib/pub	Various data files
share/lib/spell	Auxiliary spell-related databases and scripts
share/lib/tabset	Tab setting escape sequences
share/lib/terminfo	terminfo-style terminal description files
share/lib/tmac	[nt]roff macro packages
share/src	Source code for kernel, libraries, and utilities
ucb	Berkeley compatibility package binaries
ucbinclude	Berkeley compatibility package header files
ucblib	Berkeley compatibility package libraries

The /kernel Directory

The /kernel directory contains platform-dependent information that needs to reside in the root (/) file system. These directories are described in Table 40-2.

Table 40-2 The /kernel Directory

Directory	Description
/platform	Subtree of platform-specific objects which need to reside on the root filesystem. It contains a series of directories, one per supported platform. The semantics of the series of directories is equivalent to root (/).
/platform/*/kernel	Platform-dependent objects with semantics equivalent to /kernel. It includes the file unix, the core kernel that is platform dependent. See kernel(1M).
/platform/*/lib	Platform-dependent objects with semantics equivalent to /lib.
/platform/*/sbin	Platform-dependent objects with semantics equivalent to /sbin.

The Structure of UFS File System Cylinder Groups

When you create a UFS file system, the disk slice is divided into cylinder groups, which are then divided into blocks to control and organize the structure of the files within the *cylinder group*. A cylinder group has one or more consecutive disk cylinders. Each type of block has a specific function in the file system. A UFS file system has these four types of blocks:

- Boot block – Used to store information used when booting the system
- Superblock – Used to store much of the information about the file system
- Inode – Used to store all information about a file except its name
- Storage or data block – Used to store data for each file

This section provides additional information about the organization and function of these blocks.

The Boot Block

The boot block stores the procedures used in booting the system. If a file system is not to be used for booting, the boot block is left blank. The boot block appears only in the first cylinder group (cylinder group 0) and is the first 8 Kbytes in a slice.

The Superblock

The superblock stores much of the information about the file system. A few of the more important things it contains are:

- Size and status of the file system
- Label (file system name and volume name)
- Size of the file system logical block
- Date and time of the last update
- Cylinder group size
- Number of data blocks in a cylinder group
- Summary data block
- File system state: clean, stable, or active
- Path name of the last mount point

The superblock is located at the beginning of the disk slice, and is replicated in each cylinder group. Because the superblock contains critical data, multiple superblocks are made when the file system is created. Each of the superblock replicas is offset by a different amount from the beginning of its cylinder group. For multiple-platter disk drives, the offsets are calculated so that a superblock appears on each platter of the drive. That way, if the first platter is lost, an alternate superblock can always be retrieved. Except for the leading blocks in the first cylinder group, the leading blocks created by the offsets are used for data storage.

A summary information block is kept with the superblock. It is not replicated, but is grouped with the first superblock, usually in cylinder group 0. The summary block records changes that take place as the file system is used, and lists the number of inodes, directories, fragments, and storage blocks within the file system.

Inodes

An inode contains all the information about a file except its name, which is kept in a directory. An inode is 128 bytes. The inode information is kept in the cylinder information block, and contains:

- The type of the file
    - Regular
    - Directory
    - Block special
    - Character special
    - Symbolic link
    - FIFO, also known as named pipe
    - Socket
- The mode of the file (the set of read-write-execute permissions)
- The number of hard links to the file
- The User ID of the owner of the file
- The Group ID to which the file belongs
- The number of bytes in the file
- An array of 15 disk-block addresses
- The date and time the file was last accessed
- The date and time the file was last modified
- The date and time the file was created

The array of 15 disk addresses (0 to 14) point to the data blocks that store the contents of the file. The first 12 are direct addresses; that is, they point directly to the first 12 logical storage blocks of the contents of the file. If the file is larger than 12 logical blocks, the 13th address points to an indirect block, which contains direct block addresses instead of file contents. The 14th address points to a double indirect block, which contains addresses of indirect blocks. The 15th address is for triple indirect addresses, if they are ever needed. Figure 40-1 shows this chaining of address blocks starting from the inode.

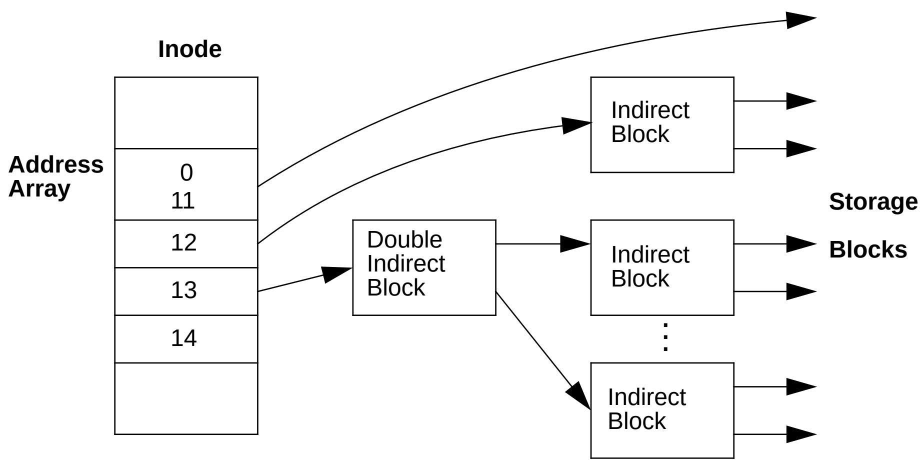


Figure 40-1 The File System Address Chain in a UFS System

Data Blocks

The rest of the space allocated to the file system is occupied by data blocks, also called storage blocks. The size of these data blocks is determined at the time a file system is created. Data blocks are allocated, by default, in two sizes: an 8-Kbyte logical block size, and a 1-Kbyte fragmentation size.

For a regular file, the data blocks contain the contents of the file. For a directory, the data blocks contain entries that give the inode number and the file name of the files in the directory.

Free Blocks

Blocks not currently being used as inodes, as indirect address blocks, or as storage blocks are marked as free in the cylinder group map. This map also keeps track of fragments to prevent fragmentation from degrading disk performance.

To give you an idea of the appearance of a typical UFS file system, Figure 40-2 shows a series of cylinder groups in a generic UFS file system.

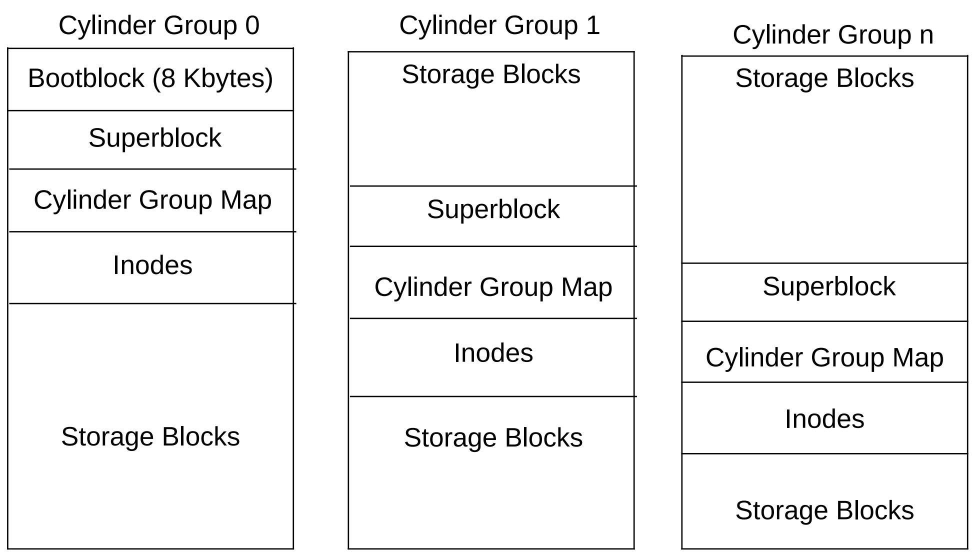


Figure 40-2 A Typical UFS File System

Deciding on Custom File System Parameters

Before you choose to alter the default file system parameters assigned by the `newfs` command, you need to understand them. This section describes each of these parameters:

- Block size
- Fragment size
- Minimum free space
- Rotational delay
- Optimization type
- Number of inodes

Logical Block Size

The logical block size is the size of the blocks that the UNIX kernel uses to read or write files. The logical block size is usually different from the physical block size (usually 512 bytes), which is the size of the smallest block that the disk controller can read or write.

You can specify the logical block size of the file system. After the file system is created, you cannot change this parameter without rebuilding the file system. You can have file systems with different logical block sizes on the same disk.

By default, the logical block size is 8192 bytes (8 Kbytes) for UFS file systems. The UFS file system supports block sizes of 4096 or 8192 bytes (4 or 8 Kbytes). 8 Kbytes is the recommended logical block size.

To choose the best logical block size for your system, consider both the performance desired and the available space. For most UFS systems, an 8-Kbyte file system provides the best performance, offering a good balance between disk performance and use of space in primary memory and on disk.

As a general rule, to increase efficiency, use a larger logical block size for file systems where most of the files are very large. Use a smaller logical block size for file systems where most of the files are very small. You can use the `quot -c` *file-system* command on a file system to display a complete report on the distribution of files by block size.

Fragment Size

As files are created or expanded, they are allocated disk space in either full logical blocks or portions of logical blocks called fragments. When disk space is needed to hold a data for a file, full blocks are allocated first, and then one or more fragments of a block are allocated for the remainder. For small files allocation begins with fragments.

The ability to allocate fragments of blocks to files, rather than just whole blocks, saves space by reducing *fragmentation* of disk space resulting from unused holes in blocks.

You define the *fragment size* when you create a UFS file system. The default fragment size is 1 Kbyte. Each block can be divided into 1, 2, 4, or 8 fragments, which results in fragment sizes from 8192 bytes to 512 bytes (for 4-Kbyte file systems only). The lower bound is actually tied to the disk sector size, typically 512 bytes.

Note – The upper bound may equal the full block size, in which case the fragment is not a fragment at all. This configuration may be optimal for file systems with very large files when you are more concerned with speed than with space.

When choosing a fragment size, look at the trade-off between time and space: a small fragment size saves space, but requires more time to allocate. As a general rule, to increase storage efficiency, use a larger fragment size for file systems where most of the files are large. Use a smaller fragment size for file systems where most of the files are small.

Minimum Free Space

The *minimum free space* is the percentage of the total disk space held in reserve when you create the file system. The default reserve is 10 percent. Free space is important because file access becomes less and less efficient as a file system gets full. As long as there is an adequate amount of free space, UFS file systems operate efficiently. When a file system becomes full, using up the available user space, only root can access the reserved free space.

Commands such as `df` report the percentage of space that is available to users, excluding the percentage allocated as the minimum free space. When the command reports that more than 100 percent of the disk space in the file system is in use, some of the reserve has been used by root.

If you impose quotas on users, the amount of space available to the users does not include the free space reserve. You can change the value of the minimum free space for an existing file system by using the `tunefs` command.

Rotational Delay (Gap)

The *rotational delay* is the expected minimum time (in milliseconds) it takes the CPU to complete a data transfer and initiate a new data transfer on the same disk cylinder. The default delay depends on the type of the disk, and is usually optimized for each disk type.

When writing a file, the UFS allocation routines try to position new blocks on the same disk cylinder as the previous block in the same file. The allocation routines also try to optimally position new blocks within tracks to minimize the disk rotation needed to access them.

To position file blocks so they are “rotationally well-behaved,” the allocation routines must know how fast the CPU can service transfers and how long it takes the disk to skip over a block. Using options to the `mkfs` command, you can indicate how fast the disk rotates and how many disk blocks (sectors) it has per track. The allocation routines use this information to figure out how many milliseconds it takes to skip a disk block. Then using the expected transfer time (rotational delay), the allocation routines can position or place blocks so that the next block is just coming under the disk head when the system is ready to read it.

Note – It is not necessary to specify the rotational delay (`-d` option to `newfs`) for some devices.

Place blocks consecutively only if your system is fast enough to read them on the same disk rotation. If the system is too slow, the disk spins past the beginning of the next block in the file and must complete a full rotation before the block can be read, which takes a lot of time. You should try to specify an appropriate value for the gap so that the head is located over the appropriate block when the next disk request occurs.

You can change the value of this parameter for an existing file system by using the `tunefs` command. The change applies only to subsequent block allocation, not to blocks already allocated.

Optimization Type

The *optimization type* is either *space* or *time*.

- **Space –** When you select space optimization, disk blocks are allocated to minimize fragmentation and disk use is optimized. Space is the default when you set the minimum free space to less than 10 percent.
- **Time –** When you select time optimization, disk blocks are allocated as quickly as possible, with less emphasis on their placement. Time is the default when you set the minimum free space to 10 percent or greater. When there is enough free space, it is relatively easy to allocate disk blocks effectively, without resulting in too much fragmentation.

You can change the value of the optimization type parameter for an existing file system using the `tunefs` command.

Number of Bytes per Inode

The number of inodes determines the number of files you can have in the file system: one inode for each file. The *number of bytes per inode* determines the total number of inodes created when the file system is made: the total size of the file system divided by the number of bytes per inode. Once the inodes are allocated, you cannot change the number without re-creating the file system.

The default number of bytes per inode is 2048 bytes (2 Kbytes), which assumes the average size of each file is 2 Kbytes or greater. Most files are larger than 2 Kbytes. If you have a file system with many symbolic links, they can lower the average file size. If your file system is going to have many small files, you can give this parameter a lower value. Note, however, that having too many inodes is much better than running out of them. If you have too few inodes, you could reach the maximum number of files on a disk slice that is practically empty.

Commands for Creating a Customized File System

This section describes the two commands you use to create a customized file system:

- newfs
- mkfs

The newfs Command Syntax, Options, and Arguments

The newfs command is a friendlier version of the mkfs command that is used to create file systems. The newfs command is located in the /usr/sbin directory.

The syntax is:

newfs [-Nv] [*mkfs_options*] *special*

Table 40-3 describes the options and arguments to the newfs command.

Table 40-3 The newfs Command Options and Arguments

Option	Description
N	Displays the file system parameters that would be used in creating the file system without actually creating it. This option does not display the parameters used to create an existing file system.
-v	Displays the parameters that are passed to the mkfs command and creates the file system, unless used with the -N option.
mkfs-options	Use the following options to set the parameters passed to the mkfs command. The options are listed below in the order they are passed to mkfs. Separate the options with spaces without any preceding keywords.
-s *size*	The size of the file system in sectors. The default is automatically determined from the disk label.
-t *ntrack*	The number of tracks per cylinder on the disk. The default is determined from the disk label.
-b *bsize*	The logical block size in bytes to use for data transfers. Specify the size of 4096 or 8192 (4 or 8 Kbytes). The default is 8192 bytes (8 Kbytes).
-f *fragsize*	The smallest amount of disk space in bytes that is allocated to a file. Specify the fragment size in powers of two in the range from 512 to 8192 bytes. The default is 1024 bytes (1 Kbyte).

Table 40-3 The `newfs` Command Options and Arguments *(Continued)*

Option	Description
`-c` *cgsize*	The number of disk cylinders per cylinder group. This number must be in the range 1 to 32. The default is 16.
`-m` *free*	The minimum percentage of free disk space to allow. The default is 10 percent.
`-r` *rpm*	The speed of the disk, in revolutions per minute. The default is 3600. This parameter is converted to revolutions per second before it is passed to `mkfs`.
`-i` *nbpi*	The number of bytes per inode to use in computing how may inodes to create. The default is 2048.
`-o` *opt*	Optimization type to use for allocating disk blocks to files: `s` for space or `t` for time.
`-a` *apc*	The number of alternate blocks per disk cylinder (SCSI devices only) to reserve for bad block placement. The default is 0.
`-d` *gap*	(Rotational delay) The expected minimum number of milliseconds it takes the CPU to complete a data transfer and initiate a new data transfer on the same disk cylinder. The default is 4.
`-d` *nrpos*	The number of different rotation positions in which to divide a cylinder group. The default is 8.
`-C` *maxcontig*	The maximum number of blocks, belonging to one file, that will be allocated contiguously before inserting a rotational delay. The default varies from drive to drive. Drives without internal (track) buffers (or drives/controllers that don't advertise the existence of an internal buffer) default to 1. Drives with buffers default to 7. This parameter is limited in the following way: *blocksize* * *maxcontig* must be <= *maxphys* *maxphys* is a read-only kernel variable that specifies the maximum block transfer size (in bytes) that the I/O subsystem is capable of satisfying. (This limit is enforced by `mount`, not by `newfs` or `mkfs`.) This parameter also controls clustering. Regardless of the value of *rotdelay,* clustering is enabled only when *maxcontig* is greater than 1. Clustering allows higher I/O rates for sequential I/O and is described in `tunefs(1M)`.
special	The special character (raw) device file name of the partition to contain the file system. This argument is required.

Examples—newfs Command Options and Arguments

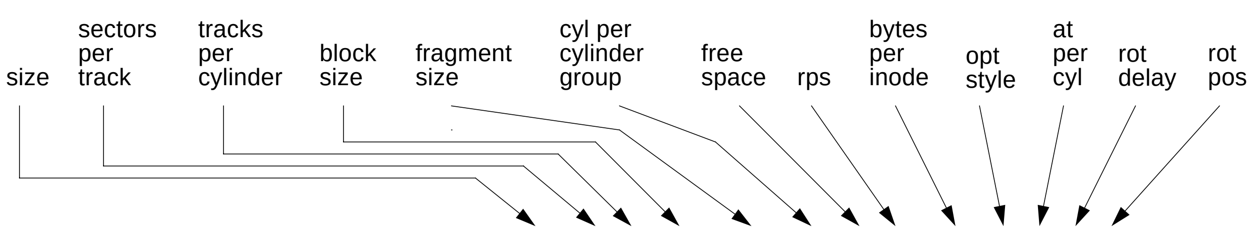
This `newfs` example uses the `-N` option to display file system information, including the backup superblocks.

```
# newfs -N /dev/rdsk/c0t0d0s0
/dev/rdsk/c0t0d0s0:     37260 sectors in 115 cylinders of 9 tracks, 36 sectors
        19.1MB in 8 cyl groups (16 c/g, 2.65MB/g, 1216 i/g)
superblock backups (for fsck -b #) at:
 32, 5264, 10496, 15728, 20960, 26192, 31424, 36656,
#
```

The Generic mkfs Command

The generic `mkfs` command calls a file system-specific `mkfs`, which then creates a file system of a specified type on a specified disk slice. Although `mkfs` can support different types of file systems, in practice you would use it to create UFS file systems. To make other types of file systems, you would have to write the software for the file system–specific versions of the `mkfs` command to use. Normally, you do not run `mkfs` directly; it is called by the `newfs` command.

This annotated example illustrates all of the arguments to the `mkfs` command.




```
# mkfs -F ufs /dev/rdsk/c0t0d0s0 37260 36 9 8192 1024 16 10 60 2048 t 0 -1 8
```

The generic `mkfs` command is located in `/usr/sbin`. See `mkfs(1M)` for a description of the arguments and options.

Part 9 — Backing Up and Restoring Data

This part provides instructions for backing up and restoring data in the Solaris environment.

Overview of Backing Up and Restoring File Systems 41

This chapter provides guidelines and planning information on backing up and restoring complete file systems using the `ufsdump` and `ufsrestore` commands.

This is a list of overview information in this chapter.

For step-by-step instructions on backing up file systems using the `ufsdump` command, see Chapter 42, "Backing Up Files and File Systems." For step-by-step instructions on restoring file systems using the `ufsrestore` command, see Chapter 43, "Restoring Files and File Systems."

41

Definition: Backing Up and Restoring File Systems

Backing up file systems means copying file systems to removable media (such as tape) to safeguard against loss, damage, or corruption. Restoring file systems means copying reasonably-current backup files from removable media to a working directory.

This chapter describes the commands for *scheduled* backup and restore operations (`ufsdump` and `ufsrestore`); however, there are other commands you can use for copying files and file systems for sharing or transporting files. Table 41-1 provides pointers to all commands that copy individual files and/or file systems to media.

Table 41-1 Commands for Copying Files and File Systems

If You Want To ...	**Then Use ...**	**And Go To ...**
Back up complete or individual file systems to a local or remote tape device	`ufsdump` command	Chapter 42, “Backing Up Files and File Systems” or Chapter 44, “Reference Information on ufsdump and ufsrestore”
Back up complete file systems for all systems on a network from a server	Networker for Solaris software	*Networker for Solaris Administrator’s Guide*
Copy, list, and retrieve files on tape Copy, list, and retrieve files on diskette	`tar`, `cpio`, or `pax` command `tar` command	Chapter 45, “Copying UFS Files and File Systems”
Copy master disk to a clone disk	`dd` command	Chapter 45, “Copying UFS Files and File Systems”
Restore complete file systems or individual files from removable media to a working directory	`ufsrestore` command	Chapter 43, “Restoring Files and File Systems”

Why You Should Back Up File Systems

Backing up files is one of the most crucial system administration functions. You should perform regularly-scheduled backups to prevent loss of data due to:

- System crashes
- Accidental deletion of files
- Hardware failures
- Natural disasters (for example, fire, hurricanes)
- Problems when reinstalling or upgrading a system

Choosing a Tape Device

Table 41-2 shows typical tape devices used for storing file systems during the backup process. For more detailed information on tape devices, see Chapter 46, “Managing Tape Drives.”

Table 41-2 Typical Media For Backing Up File Systems

Media	Capacity	Tape Length
1/2-inch reel tape	40–45 Mbytes	2300 feet
60-Mbyte 1/4-inch streaming cartridge tape	60 Mbytes	425 feet
150-Mbyte 1/4-inch streaming cartridge tape	150 Mbytes	700 feet
2.3-Gbyte 8-mm cartridge tape	2.3 Gbytes	6000 feet
5.0-Gbyte 8-mm cartridge tape	5.0 Gbytes	13000 feet
4-mm cartridge tape (DAT)[1]	—	—
3.5-inch diskette[2]	1422 blocks (1.44 Mbytes)	—

1. Capacity depends on the type of drive and the data being written to the tape.

2. You can back up file systems using diskettes, but this is time-consuming and cumbersome.

Planning Which File Systems to Back Up

You should back up any file systems that are critical to users, including file systems that change frequently. Table 41-3 and Table 41-4 provide general guidelines on the file systems to back up for standalone systems and servers.

Table 41-3 File Systems to Back Up For Standalone Systems

Consider Backing Up These File Systems[1] ...	Because ...	And At This Interval ...
root (/) – partition 0 *	The root (/) file system contains the kernel and may contain the /var directory in which frequently modified files are kept such as mail and accounting.	At regular intervals.
/usr – partition 6 *	Installing new software and adding new commands typically affects user file systems.	Occasionally.
/export/home	The /export/home file system contains directories and subdirectories of all users on the standalone system.	More often than root (/) or /usr, perhaps as often as once a day, depending on your site needs.
/export , /var, or other file systems	During installation of Solaris software, you may have created these file systems.	As your site requires.

1. Use the df command or look at /etc/vfstab file to find out on which slice a file system is located.

* Indicates file systems created by default when installing Solaris software.

Table 41-4 File Systems to Back Up For Servers

Consider Backing Up These File Systems[1] ...	Because ...	And At This Interval ...
root (/) – partition 0 * /export – partition 3 * /usr – partition 6 *	These file systems contain the kernel, major commands, and executables.	Once a day to once a month depending on your site needs. root (/) - if you frequently add and remove clients and hardware on the network, you have to change important files in root (/), including the kernel configuration file. In this case, you should do a full back up on the root (/) file system between once a week and once a month. If your site keeps users' mail in the /var/mail directory on a mail server (which client systems then mount), you may want to back up root (/) daily. /export - the root (/) directory of diskless clients is kept in the /export file system. Because the information it contains is similar to the server's root directory in slice 0, it does not change frequently. You need to back up only occasionally, unless your site sends mail to clients' root directories; then you should back up /export more frequently. /usr - contents are fairly static and only need to be backed up from once a week to once a month.
/export/home – partition 7 *	The /export/home file system contains the home directories and subdirectories of all the users on the system; its files are volatile.	Once a day to once a week.

1. Use the df command or look at /etc/vfstab file to find out on which slice a file system is located.

* Indicates file systems created by default when installing Solaris software.

Note – You do not need to back up a server's /export/swap file system.

Overview of the Backup and Restore Commands

The `ufsdump` and `ufsrestore` commands are the recommended commands for scheduled backups of complete file systems. Table 41-5 lists the tasks you can perform with them. For information on how these commands work and their syntax, see Chapter 44, "Reference Information on ufsdump and ufsrestore."

Table 41-5 Tasks You Can Perform With the `ufsdump` and `ufsrestore` Commands

With This Command...	You Can ...	Comments
`ufsdump`		
	Back up complete or individual file systems to local or remote tape drives	The tape device can be on any system in the network to which the user has access. This command works quickly because it is aware of the structure of the UFS file system type, and works directly through the raw device file.
	Back up incremental backups	This enables you to back up only those files that were changed since a previous backup.
	Back up groups of systems over the network from a single system	You can run `ufsdump` from one system on each remote system through a remote shell or remote login, and direct the output to the system on which the drive is located. Or you can pipe the output to the `dd` command.
	Automate backups	Use the `crontab` utility that calls a script that starts the `ufsdump` command.
	Restrict user access to backup tables	Use the `a` option.
	Determine the size of a backup without actually doing the backup	Use the `S` option.
	Keep a log file of each dump record	Use the `u` option.
	Verify contents of the tape against the source file system	Use the `v` option.
`ufsrestore`		
	Restore individual or complete file systems from a local or remote tape drive	

Choosing the Type of Backup

With the `ufsdump` command, you can perform full or incremental backups. Table 41-6 lists the differences between these types of backup procedures.

Table 41-6 Differences Between Full and Incremental Backups

This Type of Backup ...	Copies ...	And Has These Advantages ...	And These Disadvantages ...
Full	A complete file system or directory.	More difficult to retrieve small changes in file systems.	Requires large numbers of backup tapes that take a long time to write. Takes longer to retrieve individual files because the drive has to move sequentially to the point on the tape where the file is located.
Incremental	Only files in the specified file system that have changed since a previous backup.	Easier to retrieve small changes in file systems.	Finding which incremental tape a file is on can take time.

Guidelines For Scheduling Backups

A *backup schedule* is the schedule you establish to run the `ufsdump` command. This section provides guidelines on the factors to weigh when creating a backup schedule, guidelines on how often to back up file systems, and sample backup schedules.

What Drives a Backup Schedule

The schedule you create depends on:

- Your need to minimize the number of tapes
- Time available for doing backups
- Time available to do a full restore of a damaged file system
- Time available for retrieving individual files that get accidentally deleted

How Often Should You Backup?

If you do not need to minimize time and media spent on backups, you can do full backups every day. However, this is not realistic for many sites, so incremental backups are most often used. In this case, it is recommended that you back up your site to be able to restore files from the last four weeks. This requires at least four sets of tapes—one for each week, which you would rotate each month. In addition, you should archive the monthly backups for at least a year, and then keep a yearly backup for a number of years.

Using Dump Levels to Create Incremental Backups

The dump level you specify in the `ufsdump` command (0-9) determines which files are backed up. Specifying dump level 0 creates a full backup. Numbers 1-9 are used to schedule incremental backups, but have *no defined meanings*. Numbers 1-9 are just a range of numbers used to schedule cumulative or discrete backups. The only meaning levels 1-9 have is in relationship to each other, as a higher or lower number.

The following examples show the flexibility of the incremental dump procedure using levels 1-9.

Dump Levels For Daily, Cumulative Backups

Doing daily, cumulative incremental backups is the most commonly used backup scheme and is recommended for most situations. The following example shows a schedule using a level 9 dump each day, and a level 5 dump on Friday to restart the process.

Note – In the following example, you could have used other numbers in the 1-9 range to produce the same results. The key is having the same number each day, with any *lower* number on Friday. For example, you could have specified levels 4, 4, 4, 4, 2 or 7, 7, 7, 7, 5.

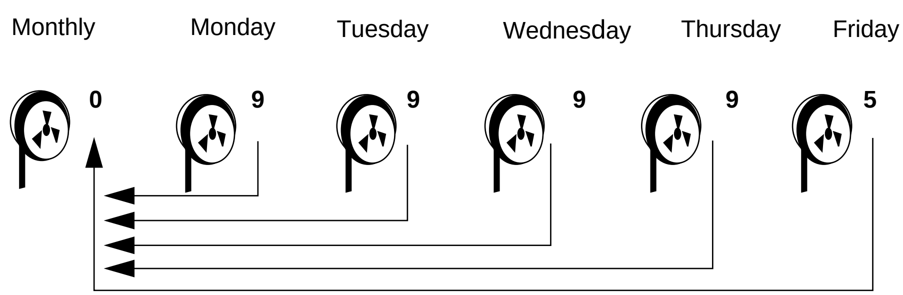


Figure 41-1 Incremental Backup: Daily Cumulative

Dump Levels For Daily, Discrete Backups

The following example shows a schedule where you capture only a day's work on different tapes. In this case, sequential dump level numbers are used during the week (3,4,5,6) with a lower number (3) on Friday.

Note – In the following example, you could have used the sequence 6,7,8,9 followed by 2, or 5,6,7,8 followed by 3. Remember, the number themselves have no defined meaning; you attribute meaning by ordering them in a high/low sequence.

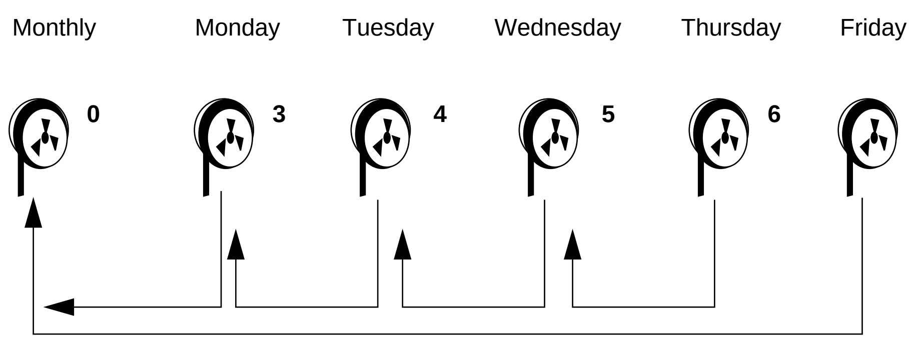


Figure 41-2 Incremental Backup: Daily Discrete

Sample Backup Schedules

This section provides sample backup schedules. All schedules assume you begin with a full backup (level 0), and that you use the u option to record each backup.

Example—Daily Cumulative, Weekly Cumulative Backups

Table 41-7 shows the most commonly used incremental backup schedule; it is recommended for most situations.

Table 41-7 Daily Cumulative/Weekly Cumulative Backup Schedule

	Floating	**Mon**	**Tues**	**Wed**	**Thurs**	**Fri**
1st of Month	0					
Week 1		9	9	9	9	5
Week 2		9	9	9	9	5
Week 3		9	9	9	9	5
Week 4		9	9	9	9	5

With this schedule:

- Each weekday tape accumulates all files changed since the end of the previous week (or the initial level 0 for the first week).
- Each Friday's tape contains all the files changed since the first level 0. For the level 9 backups, the previous level 0 or level 5 is the closest backup at a lower level.
- All files that have changed since the lower-level backup at the end of the previous week are saved each day.
- For each Friday level 5, the nearest lower-level backup is the level 0 done at the beginning of the month. Therefore, each Friday's tape contains all the files changed during the month to that point.

Table 41-8 shows how the contents of the tapes can change across two weeks.

Table 41-8 Contents of Tapes for Daily / Weekly Cumulative Schedule

	Mon	Tues	Wed	Thurs	Fri
Week 1	a b	a b c	a b c d	a b c d e	a b c d e f
Week 2	g	g h	g h i	g h i j	a b c d e f g h i j k

Example—Daily Cumulative, Weekly Incremental Backups

Table 41-9 shows a schedule where each weekday tape accumulates all files that changed since the beginning of the week (or the initial level 0 for the first week), and each Friday's tape contains all the files changed that week.

Table 41-9 Daily Cumulative / Weekly Incremental Backup Schedule

	Floating	Mon	Tues	Wed	Thurs	Fri
1st of Month	0					
Week 1		9	9	9	9	3
Week 2		9	9	9	9	3
Week 3		9	9	9	9	4
Week 4		9	9	9	9	5

Table 41-10 shows how the contents of the tapes can change across two weeks.

*Table 41-10*Contents of Tapes for Daily Cumulative / Weekly Incremental Schedule

	Mon	Tues	Wed	Thurs	Fri
Week 1	a b	a b c	a b c d	a b c d e	a b c d e f
Week 2	g	g h	g h i	g h i j	g h i j k

Tape Needs

With this schedule, you will need six tapes (if you want to reuse daily tapes), or nine tapes (if you want to use four different daily tapes): one for the level 0, four for the Fridays, and one or four daily tapes.

If you need to restore a complete file system, you will need five tapes: the level 0, all preceding Friday tapes (three), and the most recent daily tape.

Example—Daily Incremental, Weekly Cumulative Backups

Table 41-11 shows a schedule where each weekday tape contains only the files changed since the previous day, and each Friday's tape contains all files changed since the initial level 0 at the beginning of the month.

Table 41-11 Daily Incremental / Weekly Cumulative Backup Schedule

	Floating	Mon	Tues	Wed	Th	Fri
1st of Month	0					
Week2		3	4	5	6	2
Week3		3	4	5	6	2
Week4		3	4	5	6	2

Table 41-12 shows how the contents of the tapes can change across two weeks.

Table 41-12 Contents of Tapes for Daily / Weekly Cumulative Schedule

	Mon	Tues	Wed	Th	Fri
Week1	a b	c d	e f g	h	a b c d e f g h i
Week2	j k l	m	n o	p q	a b c d e f g h i j k l m n o p q r s

Tape Needs

With this schedule you will need at least nine tapes: one for the level 0, four for the Fridays, and four daily tapes, assuming you reuse daily tapes each week, which is not recommended. If you save the weekly tapes for a month, you need 21 tapes.

If you need to restore the complete file system, you need six tapes: the level 0, the most recent Friday tape, and *all* the daily tapes for that week.

Example—Backup Schedule for a Server

Table 41-13 shows an example backup strategy for a heavily-used file server on a small network where users are doing file-intensive work, such as program development or document production. It assumes that the backup period begins on a Sunday and consists of four seven-day weeks.

Table 41-13 Schedule of Backups for an Example Server

Directory	**Date**	**Level**	**Tape Name**
`/`	1st Sunday	0	*n* tapes
`/usr`	1st Sunday	0	"
`/export`	1st Sunday	0	"
`/export/home`	1st Sunday	0	"
	1st Monday	9	A
	1st Tuesday	9	B
	1st Wednesday	5	C
	1st Thursday	9	D
	1st Friday	9	E
	1st Saturday	5	F
`/`	2nd Sunday	0	*n* tapes
`/usr`	2nd Sunday	0	"
`/export`	2nd Sunday	0	"
`/export/home`	2nd Sunday	0	"
	2nd Monday	9	G
	2nd Tuesday	9	H
	2nd Wednesday	5	I
	2nd Thursday	9	J
	2nd Friday	9	K
	2nd Saturday	5	L
`/`	3rd Sunday	0	*n* tapes
`/usr`	3rd Sunday	0	"

Table 41-13 Schedule of Backups for an Example Server *(Continued)*

Directory	Date	Level	Tape Name
/export	3rd Sunday	0	"
/export/home	3rd Sunday	0	"
	3rd Monday	9	M
	3rd Tuesday	9	N
	3rd Wednesday	5	O
	3rd Thursday	9	P
	3rd Friday	9	Q
	3rd Saturday	95	R
/	4th Sunday	0	*n* tapes
/usr	4th Sunday	0	"
/export	4th Sunday	0	"
/export/home	4th Sunday	0	"
	4th Monday	9	S
	4th Tuesday	9	T
	4th Wednesday	5	U
	4th Thursday	9	V
	4th Friday	9	W
	4th Saturday	5	X

With this plan, you use 4*n* tapes (the number of tapes needed for four full backups of `/`, `/usr`, `/export`, and `/export/home`), plus 24 additional tapes for the incremental backups of `/export/home`. This plan assumes that each incremental backup uses one tape and you save the tapes for a month.

Here's how this plan works:

1. On each Sunday, do a full backup (level 0) of `/`, `/usr`, `/export`, and `/export/home`. Save the level 0 tapes for at least 3 months.
2. On the first Monday of the month, use tape A to do a level 9 backup of `/export/home`. `ufsdump` copies all files changed since the previous lower-level backup (in this case, the level 0 backup that you did on Sunday).
3. On the first Tuesday of the month, use tape B to do a level 9 backup of `/export/home`. Again, `ufsdump` copies all files changed since the last lower-level backup—Sunday's level 0 backup.
4. On the first Wednesday, use tape C to do a level 5 backup. `ufsdump` copies all files changed since Sunday.
5. Do the Thursday and Friday level 9 backups on tapes D and E. `ufsdump` copies all files changed since the last lower-level backup—Wednesday's level 5 backup.
6. On the first Saturday of the month, do a level 5 backup of `/export/home`, which copies all files changed since the previous lower-level backup—in this case, the level 0 backup you did on Sunday. Store tapes A-F until the first Monday of the next 4-week period, when you use them again.
7. Repeat steps 1–6 for the next three weeks, using tapes G-L and 4*n* tapes for the level 0 on Sunday, and so on.
8. For each 4-week period, repeat steps 1–7, using a new set of tapes for the level 0s and reusing tapes A–X for the incremental backups. The level 0 tapes could be reused after 3 months.

This plan lets you save files in their various states for a month. It requires many tapes, but ensures that you have a library of tapes to draw upon. To reduce the number of tapes, you could reuse Tapes A-F each week.

Other Backup Scheduling Recommendations

Table 41-14 provides other recommendations for scheduling backups.

Table 41-14 Other Recommendations for Scheduling Backing Up Systems

If You ...	**Then ...**	**Comments**
Need to restore different versions of files (for example, file systems used for word processing)	• Do daily incremental backups every working day. • Do *not* reuse the same tape for daily incremental backups.	This schedule saves all files modified that day, as well as those files still on disk that were modified since the last backup of a level lower. However, with this schedule you should use different tapes each day because a file changed on Tuesday, and again on Thursday, goes onto Friday's lower-level backup looking like it did Thursday night—not Tuesday night. If a user needs the Tuesday version, you cannot restore it unless you have a Tuesday backup tape (or a Wednesday backup tape). Similarly, a file that is present on Tuesday and Wednesday, but removed on Thursday, does not appear on the Friday lower-level backup.
Need to quickly restore a complete file system	Do lower-level backups more frequently.	—
Are backing up a number of file systems on the same server	Consider offsetting the schedule for different file systems.	This way you're not doing all level 0s on the same day.
Need to minimize tapes	• Increase the level of incremental backups done across the week. • Increase the level of backups done at the end of the week. • Put each day's and week's incremental backups onto the same tape.	This means only changes from day to day are saved on each daily tape. This means only changes from week to week (rather than the entire month) are saved on the weekly tapes. This is done by using the no rewind option in the `ufsdump` command.

Backing Up Files and File Systems 42

This chapter describes the procedures for backing up file systems using the `ufsdump` command.

This is a list of the step-by-step instructions in this chapter.

For detailed information on syntax, options, and arguments for the `ufsdump` command, see Chapter 44, “Reference Information on ufsdump and ufsrestore.”

42

Preparing to Do Backups

Preparing to back up file systems begins with planning, which is described in Chapter 41, “Overview of Backing Up and Restoring File Systems” and covers choosing:

- A tape drive
- Which file systems to back up
- The type of backup (full or incremental)
- A backup schedule

This section describes other tasks you may need to perform before backing up file systems including:

- Finding names of file systems to back up
- Determining the number of tapes for a full backup

▼ How to Find File System Names

1. **Display the contents of the** `/etc/vfstab` **file.**

```
$ more /etc/vfstab
```

2. **Look in the** `mount point` **column for the name of the file system.**

3. **You will use the mount point in the** `mount point` **column when you back up the file system.**

Example—Finding File System Names

```
$ more /etc/vfstab
#device           device                 mount            FS      fsck        mount   mount
#to mount         to fsck               point            type    pass        at boot options
#
/proc             -                      /proc            proc    -           no      -
swap              -                      /tmp             tmpfs   -           yes     -
/dev/dsk/c0t3d0s0 /dev/rdsk/c0t3d0s0     /                ufs     1           no      -
/dev/dsk/c0t3d0s1   -                    -                swap    -           no      -
/dev/dsk/c0t1d0s6 /dev/rdsk/c0t1d0s6     /usr             ufs     2           no      -
neptune:/export/usr/openwin  -           /usr/openwin     nfs     -           yes     -
neptune:/export/usr/man -                /usr/man         nfs     -           yes     -
```

▼ How to Determine the Number of Tapes for Full Backup

1. **Become root.**
2. **Estimate the size of the backup in bytes by using the** `usfdump S` **command.**

```
# ufsdump S filesystem
```

In this command,

`S`	Displays the estimated number of bytes needed to do the backup.

3. **Divide the estimated size by the capacity of the tape to see how many tapes you need.**
See Table 41-2 on page 719 for a list of tape capacities.

Example—Determining Number of Tapes

In this example, the file system of 489472 bytes will fit on a 150-Mbyte tape.

```
# ufsdump S /export/home
489472
```

Doing Backups

The following are general guidelines for performing backups:

- Use single-user mode or unmount the file system.
- Be aware that backing up file systems when there are directory-level operations (such as creating, removing, and renaming files) and file-level activity, means some data will not be included in the backup.
- You can run the `ufsdump` command from a single system and remotely back up groups of systems across the network through remote shell or remote login, and direct the output to the system on which the tape drive is located. (Typically, the tape drive is located on the system from which you run the `ufsdump` command, but it does not have to be.)

  Another way to back up files to a remote drive is to pipe the output from the `ufsdump` command to the `dd` command. See Chapter 45, "Copying UFS Files and File Systems," for information about using the `dd` command.
- If you are doing remote backups across the network, the system initiating the backup must have entries in its `/.rhosts` file for the hosts it will back up.
- To specify a remote drive on a system, use the naming convention that matches the OS release of the system with the remote tape drive. For example, use `/dev/rst0` for a remote drive on a system running SunOS 4.1.x; use `/dev/rmt/0m` for a system running Solaris 2.x.

▼ How to Do Backups on Cartridge Tape

The following steps provide the general steps for backing up file systems using the `ufsdump` command. The examples show specific uses of options and arguments.

1. **Become root.**
2. **Shut down the system.**

```
# init 0
```

3. **Bring the system to run level S (single-user mode).**

```
ok boot -s
```

4. **[Optional] Check the file system for consistency with the `fsck` command.** Running the `fsck` command using the `-m` option checks for consistency of file systems. For example, power failures can leave files in an inconsistent state. For more information on the `fsck` command, see Chapter 39, "Checking File System Integrity."

```
# fsck -m /dev/rdsk/device-name
```

5. **If you will be backing up file systems onto a remote tape drive:**
   a. Add the following entry to the `./rhosts` file of the system that is initiating the backup:

```
host root
```

   b. Verify that the host name added to the `/.rhosts` file above is accessible via the local `/etc/inet/hosts` file or available NIS or NIS+ name server.

6. **Insert a tape that is not write protected into the tape drive.**

7. Back up files systems using the `ufsdump` command.
Use the following table to select the most common options and arguments for the `ufsdump` command. See Chapter 44, "Reference Information on ufsdump and ufsrestore" for other options and arguments.

To ...	Use This Option or Argument ...	For Example ...
Do a full backup	0 option (dump level)	`ufsdump 0ucf /dev/rmt/`*n files-to-backup*
Do an incremental backup	1-9 option (dump level)	`ufsdump 9ucf /dev/rmt/`*n files-to-backup*
Back up individual files	*filename* argument	`ufsdump 9ucf /dev/rmt/`*n /export/home/user1/mail*
Record dumps to file	`u` option (updates `/etc/dumpdates` file)	`ufsdump 9ucf /dev/rmt/`*n files-to-backup*
Specify a cartridge tape	`c` option	`ufsdump 9ucf /dev/rmt/`*n files-to-backup*
Specify the tape drive file name	`f` *dump-file*	`ufsdump 9ucf /dev/rmt/1` *files-to-backup*
Back up local file systems to a remote host's tape device	*remote-host*: argument	`ufsdump 0ucf pluto:/dev/rmt/0` *files-to-backup*
Remotely log into a system and back up its file systems to a remote tape device	*user@remote-host*: argument	`ufsdump 0ucf sam@chico:/dev/rmt/0` *files-to-backup*

8. When prompted, remove the tape and replace with the next volume.

9. Label each tape with the volume number, level, date, system name, and file system.

10. **Bring the system back to run level 3 by pressing Control-d.**

Example—Full Backup, root (/), Cartridge Tape

In this example, a full backup of the root (/) file system is made on a 150-Mbyte cartridge tape (`/dev/rmt/0`).

```
# init 0
ok boot -s
# ufsdump 0ucf /dev/rmt/0 /
 DUMP: Date of this level 0 dump: Thu Oct 20 15:17:45 1994
 DUMP: Date of last level 0 dump: the epoch
 DUMP: Dumping /dev/rdsk/c0t3d0s0 (/) to /dev/rmt/0
 DUMP: mapping (Pass I) [regular files]
 DUMP: mapping (Pass II) [directories]
 DUMP: estimated 40044 blocks (19.55MB)
 DUMP: Writing 63 Kilobyte records
 DUMP: dumping (Pass III) [directories]
 DUMP: dumping (Pass IV) [regular files]
 DUMP: level 0 dump on Thu Oct 20 15:17:45 1994
 DUMP: Tape rewinding
 DUMP: 40022 blocks (19.54MB) on 1 volume
 DUMP: DUMP IS DONE
# (Press Control-D to bring system to run level 3)
```

Example—Full Backup, /export/home, 4-mm DAT Tape

In this example, a full backup of the `/export/home` file system is made on a 4-mm DAT tape.

```
# ufsdump 0ucf /dev/rmt/0 /export/home
 DUMP: Date of this level 0 dump: Fri Oct 21 10:36:45 1994
 DUMP: Date of last level 0 dump: the epoch
 DUMP: Dumping /dev/rdsk/c0t3d0s7 (/export/home) to /dev/rmt/0
 DUMP: mapping (Pass I) [regular files]
 DUMP: mapping (Pass II) [directories]
 DUMP: estimated 19558 blocks (9.55MB)
 DUMP: Writing 63 Kilobyte records
 DUMP: dumping (Pass III) [directories]
 DUMP: dumping (Pass IV) [regular files]
 DUMP: level 0 dump on Fri Oct 21 10:36:45 1994
 DUMP: Tape rewinding
 DUMP: 19558 blocks (9.55MB) on 1 volume
 DUMP: DUMP IS DONE
#
```

Verification—Full Backup, /export/home, 4-mm DAT Tape

Use the `ufsrestore` command to display the contents of the tape. This command is described in Chapter 43, "Restoring Files and File Systems."

```
# ufsrestore tf /dev/rmt/0
 2  .
 3  ./lost+found
 2688./export
 5376./opt
 8064./usr
 10752./var
 13440./var/sadm
 2689./var/sadm/install
 5377./var/sadm/install/admin
 5572./var/sadm/install/admin/default
 8065./var/sadm/install/logs
 2690./var/sadm/install/.lockfile
 2766./var/sadm/install/contents
 10753./var/sadm/pkg
          .
          .
          .
#
```

Example—Incremental Backup, root (/)

In this example, an incremental backup of the root (/) is made on a 4-mm DAT tape device.

```
# ufsdump 9ucf /dev/rmt/0 /
  DUMP: Date of this level 9 dump: Mon Oct 24 09:26:14 1994
  DUMP: Date of last level 0 dump: Mon Oct 24 09:05:00 1994
  DUMP: Dumping /dev/rdsk/c0t3d0s0 (/) to /dev/rmt/0
  DUMP: mapping (Pass I) [regular files]
  DUMP: mapping (Pass II) [directories]
  DUMP: mapping (Pass II) [directories]
  DUMP: mapping (Pass II) [directories]
  DUMP: estimated 9988 blocks (4.88MB)
  DUMP: Writing 63 Kilobyte records
  DUMP: dumping (Pass III) [directories]
  DUMP: dumping (Pass IV) [regular files]
  DUMP: level 9 dump on Mon Oct 24 09:26:14 1994
  DUMP: Tape rewinding
  DUMP: 9992 blocks (4.88MB) on 1 volume
  DUMP: DUMP IS DONE
#
```

Verification—Incremental Backup, root (/)

```
# ufsrestore tf /dev/rmt/0
         2   .
     10752   ./var
     13440   ./var/sadm
      2689   ./var/sadm/install
      2690   ./var/sadm/install/.lockfile
      2766   ./var/sadm/install/contents
     10753   ./var/sadm/pkg
      2734   ./var/adm
      2780   ./var/adm/sa
      2951   ./var/adm/sa/sa24
      2735   ./var/cron
      2885   ./var/cron/log
      8086   ./var/spool
      8087   ./var/spool/locks
      8088   ./var/tmp
#
```

Example—Full Backup, Individual Home Directory

In this example, a full backup of the `/export/home/kryten` directory is made on a 4-mm DAT tape.

```
# ufsdump 0ucf /dev/rmt/0 /export/home/kryten
  DUMP: Date of this level 0 dump: Mon Oct 24 11:06:10 1994
  DUMP: Date of last level 0 dump: the epoch
  DUMP: Dumping /dev/rdsk/c0t3d0s7 (/export/home) to /dev/rmt/0
  DUMP: mapping (Pass I) [regular files]
  DUMP: mapping (Pass II) [directories]
  DUMP: estimated 180 blocks (90KB)
  DUMP: Writing 63 Kilobyte records
  DUMP: dumping (Pass III) [directories]
  DUMP: dumping (Pass IV) [regular files]
  DUMP: 176 blocks (88KB) on 1 volume
  DUMP: DUMP IS DONE
#
```

Verification—Full Backup, Individual Home Directory

```
# ufsrestore tf /dev/rmt/0
         2    .
      2688    ./kryten
      5409    ./kryten/letters
      5410    ./kryten/letters/letter1
      5411    ./kryten/letters/letter2
      5412    ./kryten/letters/letter3
      2689    ./kryten/.profile
      8096    ./kryten/memos
        30    ./kryten/reports
        31    ./kryten/reports/reportA
        32    ./kryten/reports/reportB
        33    ./kryten/reports/reportC
#
```

Example—Full Backup to Remote System (Solaris 2.x Data to Solaris 2.x System)

In this example, the local /export/home file system on a Solaris 2.x system is backed up to a tape device on a remote Solaris 2.x system called pluto.

```
# ufsdump 0ucf pluto:/dev/rmt/0 /export/home
  DUMP: Date of this level 0 dump: Tue Oct 25 10:30:53 1994
  DUMP: Date of last level 0 dump: the epoch
  DUMP: Dumping /dev/rdsk/c0t3d0s7 (/export/home) to /dev/rmt/0
on host pluto
  DUMP: mapping (Pass I) [regular files]
  DUMP: mapping (Pass II) [directories]
  DUMP: estimated 19574 blocks (9.56MB)
  DUMP: Writing 63 Kilobyte records
  DUMP: dumping (Pass III) [directories]
  DUMP: dumping (Pass IV) [regular files]
  DUMP: level 0 dump on Tue Oct 25 10:30:53 1994
  DUMP: Tape rewinding
  DUMP: 19574 blocks (9.56MB) on 1 volume
  DUMP: DUMP IS DONE
#
```

Verification—Full Backup to Remote System (Solaris 2.x Data to Solaris 2.x System)

```
# ufsrestore tf pluto:/dev/rmt/0
         2.
         3./lost+found
      2688./kryten
      5409./kryten/letters
      5410./kryten/letters/letter1
      5411./kryten/letters/letter2
      5412./kryten/letters/letter3
      2689./kryten/.profile
      8096./kryten/memos
        30./kryten/reports
        31./kryten/reports/reportA
        32./kryten/reports/reportB
        33./kryten/reports/reportC
              .
              .
              .
#
```

Example—Full Backup to Remote System (Solaris 2.x Data to Sun 4.1.x System)

In this example, the local /export/home file system on a Solaris 2.x system is backed up to a tape device on a remote SunOS 4.1.x system (mars). Notice the SunOS 4.x-style device name (/dev/rst0) used with the ufsdump command. To run the ufsdump command, you would log in as root on the Solaris 2.x system.

```
# ufsdump 0ucf mars:/dev/rst0 /export/home
  DUMP: Date of this level 0 dump: Tue Oct 25 15:06:47 1994
  DUMP: Date of last level 0 dump: the epoch
 DUMP: Dumping /dev/rdsk/c0t3d0s7 (/export/home) to /dev/rst0 on
host mars
  DUMP: mapping (Pass I) [regular files]
  DUMP: mapping (Pass II) [directories]
  DUMP: estimated 19574 blocks (9.56MB)
  DUMP: Writing 63 Kilobyte records
  DUMP: dumping (Pass III) [directories]
  DUMP: dumping (Pass IV) [regular files]
  DUMP: level 0 dump on Tue Oct 25 15:06:47 1994
  DUMP: Tape rewinding
  DUMP: 19574 blocks (9.56MB) on 1 volume
  DUMP: DUMP IS DONE
#
```

Verification—Full Backup to Remote System (Solaris 2.x Data to Sun 4.1.x System)

```
# ufsrestore tf mars:/dev/rst0
         2      .
         3      ./lost+found
      2688      ./kryten
      5409      ./kryten/letters
      5410      ./kryten/letters/letter1
      5411      ./kryten/letters/letter2
      5412      ./kryten/letters/letter3
      2689      ./kryten/.profile
      8096      ./kryten/memos
        30      ./kryten/reports
        31      ./kryten/reports/reportA
        32      ./kryten/reports/reportB
        33      ./kryten/reports/reportC
                .
                .
                .
```

Example—Full Backup to Remote System (SunOS 4.1.x Data to Solaris 2.x)

In this example, the local root (/) file system on a Sun 4.1.x system (`mars`) is backed up to a remote tape device on a Solaris 2.x system called `pluto`.

Note – Notice that when you back up data to a SunOS 4.1.x system, you must use the `dump` command—not the `ufsdump` command.

```
mars# dump 0ucf pluto:/dev/rmt/0 /
  DUMP: Date of this level 0 dump: Tue Oct 25 16:05:19 1994
  DUMP: Date of last level 0 dump: the epoch
  DUMP: Dumping /dev/rsd2a (/) to /dev/rmt/0 on host pluto
  DUMP: mapping (Pass I) [regular files]
  DUMP: mapping (Pass II) [directories]
  DUMP: estimated 8686 blocks (4.24MB) on 0.10 tape(s).
  DUMP: dumping (Pass III) [directories]
  DUMP: dumping (Pass IV) [regular files]
  DUMP: level 0 dump on Tue Oct 25 16:05:19 1994
  DUMP: Tape rewinding
  DUMP: 8690 blocks (4.24MB) on 1 volume
  DUMP: DUMP IS DONE
mars#
```

Verification—Full Backup to Remote System (SunOS 4.1.x Data to Solaris 2.x)

```
mars# restore tf pluto:/dev/rmt/0
         2      .
         3      ./lost+found
      3776      ./export
      7552      ./home
     11328      ./usr
     15104      ./pcfs
      3777      ./tftpboot
      3778      ./tftpboot/tftpboot
      3794      ./tftpboot/boot.sun4c.sunos.4.1.3
      7553      ./etc
      7554      ./etc/sendmail.cf
      7555      ./etc/aliases
      7556      ./etc/aliases.dir
      7557      ./etc/aliases.pag
      7558      ./etc/holidays
      7559      ./etc/dumpdates
      7560      ./etc/fbtab
      7561      ./etc/filetype
      7562      ./etc/format.dat
      7563      ./etc/fstab
                .
                .
                .
mars#
```

42

Troubleshooting

Symptom: Root (/) File System Fills Up

You do a backup of a file system. Nothing is written to the media, but the root (`/`) file system fills up. The `ufsdump` command prompts you to install the second volume of media when the root (`/`) file system is full. `Filesystem is full` messages will be displayed in the console window.

Explanation:

If you used an invalid destination device name with the `f` option, the `ufsdump` command wrote to a file in the `/dev` directory of the root (`/`) file system, filling it up. For example, if you typed `/dev/rmt/st0` instead of `/dev/rmt/0`, the backup file `/dev/rmt/st0` was created on the disk rather than being sent to the tape drive.

Restoring Files and File Systems 43

This chapter describes the procedures for restoring file systems.

This is a list of step-by-step instructions in this chapter.

This chapter describes how to use the `ufsrestore(1M)` command to restore files and file systems that were backed up using the `ufsdump` command. See Chapter 45, "Copying UFS Files and File Systems," for information about other commands you can use to archive, restore, copy, or move files and file systems.

Preparing to Restore Files and File Systems

The `ufsrestore` command copies files from backups created using the `ufsdump` command into the current working directory. You can use `ufsrestore` to reload an entire file system hierarchy from a level 0 dump and incremental dumps that follow it or to restore one or more single files from any dump tape. If `ufsrestore` is run by root, files are restored with their original owner, last modification time, and mode (permissions).

Before you start to restore files or file systems, you need to know:

- Which tapes (or diskettes) you need
- The raw device name for the file systems you want to back up
- The type of tape drive you will use
- The device name (local or remote) for the tape drive

Determining the Disk Device Name

If you have properly labeled your backup tapes, you should be able to use the disk device name (`/dev/rdsk/`*devicename*) from the tape label. See "How to Find File System Names" on page 735 for more information.

Determining the Type of Tape Drive You Will Use

You must use a tape drive that is compatible with the backup media to restore the files. The format of the backup media determines which drive you must use to restore files. For example, if your backup media is 8-mm tape, you must use an 8-mm tape drive to restore the files.

Determining the Tape Device Name

You may have specified the tape device name (`/dev/rmt/`*n*) as part of the backup tape label information. If you are using the same drive to restore a backup tape, you can use the device name from the label. See Chapter 46, "Managing Tape Drives" for more information on media devices and device names.

Restoring Complete File Systems

Occasionally, a file system becomes so damaged that you must completely restore it. Typically, you need to restore a complete file system because of a disk head crash. You may need to replace the hardware before you can restore the software. See Chapter 29, "SPARC: Adding a Disk," or Chapter 30, "x86: Adding a Disk" for information on how to replace a disk. Fully restoring a file system such as `/export/home` can take a lot of time. If you have faithfully backed up file systems, you can restore them to their state as of the last incremental backup.

Restoring Individual Files and Directories

This section describes how to restore individual files and directories using a local tape drive. See "How to Use a Remote Drive to Restore Files" on page 766 for information on how to use a remote drive to restore files.

When you restore files in a directory other than the root directory of the file system, `ufsrestore` re-creates the file hierarchy in the current directory. For example, if you restore files to `/home1` that were backed up from `/home/doc/books` the files are restored in the directory `/home1/doc/books`.

When restoring individual files and directories, it is a good idea to restore them to a temporary directory such as `/var/tmp`. After you verify them, you can move the files to their proper locations. You can restore individual files and directories to their original locations. If you do so, be sure you are not overwriting newer files with older versions from the backup tape.

Using a Remote Drive to Restore Files

You can restore files from a remote drive by adding *remote-host:* to the front of the tape device name. Here is the syntax:

```
ufsrestore rf [user@]remote-host:/dev/rmt/unit filename
```

For example, to access `/dev/rmt/0` on the system `venus`, type:

```
# ufsrestore rf venus:/dev/rmt/0 filename
```

Troubleshooting

Make Sure the Backup and Restore Commands Match

You can only use `ufsrestore` to restore files backed up with `ufsdump`. If you back up with `tar`, restore with `tar`. If you use the `ufsrestore` command to restore a tape that was written with another command, an error message tells you that the tape is not in `ufsdump` format.

Check to Make Sure You Have the Right Current Directory

It is easy to restore files to the wrong location. Because the `ufsdump` command always copies files with full path names relative to the root of the file system, you should usually change to the root directory of the file system before running `ufsrestore`. If you change to a lower-level directory, after you restore the files you will see a complete file tree created under that directory.

Use the Old `restore` Command to Restore Multivolume Diskette Backups

You cannot use the `ufsrestore` command to restore files from a multivolume backup set of diskettes made with the `dump` command. You must restore the files on a SunOS 4.x system.

Interactive Commands

When you use the interactive command, a `ufsrestore>` prompt is displayed, as shown in this example:

```
% ufsrestore ivf /dev/rmt/0
Verify volume and initialize maps
Tape block size is 126
Dump date: Thu Aug 18 09:06:43 1994
Dumped from: Sun Aug 14 08:25:10 1994
Level 0 dump of /usr on pilgrim:/dev/dsk/c0t1d0s6
Label:none
Extract directories from tape
Initialize symbol table.
ufsrestore>
```

At the ufsrestore> prompt, you can use the commands listed on page 786 to find files, create a list of files to be restored, and restore them.

Restoring File Systems

Things you need to know:

- Which tapes have the files to be restored
- The path name of the files to be restored

▼ How to Determine Which Tapes to Use

1. **Ask the user the date when the file or file system was lost, or the approximate date the files to be recovered were last modified.**

2. **Refer to your backup plan to find the date of the last backup that would have the file or file system on it.**
   Note that you do not necessarily use the most recently backed up version of the file. To retrieve the most recent version of a file, work backward through the incremental backups from highest to lowest level and most recent to least recent.

3. **If you have online archive files, use the ufsrestore command to identify correct media.**

```
# ufsrestore ta archive-name ./path/filename ./path/filename
```

In this command,

`t`	List each file that appears on the tape.
`a`	Reads the table of contents from the online archive file instead of the tape.
archive-name	Identifies the online archive file name.
./path/filename	Identifies the file name(s) you are looking for on the online archive. If successful, `ufsrestore` will print out the inode number and file name. If unsuccessful, `ufsrestore` will print an error message.

4. **Insert the media containing the backups in the drive and use the `ufsrestore` command to verify the correct media.**

```
# ufsrestore tf device-name ./path/filename ./path/filename
```

Be sure to use the complete path for the *filename(s)*. If a file is in the backup, its name and inode number is listed. Otherwise, a message says it is not on the volume.

5. **If you have multiple dump files on the same tape, use the `s` *n* option to position the tape at the dump you want to use.**

```
# ufsrestore xfs /dev/rmt/n tape_number
```

Example—Determining Which Tapes to Use

If you use `ufsdump` to dump the `/usr` slice, the table of contents lists only the files and directories under `/usr`. To see if `/usr/bin/pwd` is in the online archive, type:

```
# ufsrestore ta archive-name ./bin/pwd
```

To see if /usr/bin/pwd is on the backup tape, type:

```
# ufsrestore tf /dev/rmt/n ./bin/pwd
```

▼ How to Restore Files Interactively

1. **Become root.**
2. **Write-protect the tape.**
3. **Put the backup tape in the tape drive.**
4. **Change to a directory that will be used to restore the files temporarily.**

```
# cd /var/tmp
```

If you want to restore the files to a different directory, substitute the directory name for /var/tmp in this step.

5. **Use the ufsrestore command to start the interactive restoration.**
Some informational messages and the ufsrestore> prompt are displayed.

```
ufsrestore> if /dev/rmt/n
```

6. **Create a list of files to be restored.**
   a. **List the contents of a directory.**

```
ufsrestore> ls directory
```

   b. **Change to a directory.**

```
ufsrestore> cd directory-name
```

c. **Create a list of files and directories you want to restore.**

```
ufsrestore> add filename filename
```

d. **[Optional] If you need to remove a directory or file name from the list of files to be restored, use the `delete` command.**

```
ufsrestore> delete filename
```

7. **Turn on `verbose` mode to display the file names as they are being restored.**

```
ufsrestore> verbose
```

8. **Use the `extract` command after the list is complete.**

```
ufsrestore> extract
```

The `ufsrestore` command asks you which volume number to use.

9. **Type the volume number and press Return. If you have only one volume, type `1` and press Return.**

```
Specify next volume #: 1
```

The files and directories in the list are extracted and restored to the current working directory.

10. **To keep the mode of the current directory unchanged, enter `n` at the `set owner/mode` prompt.**

```
set owner/mode for '.'? [yn] n
```

11. **Quit the `ufsrestore` program.**

```
ufsrestore> quit
```

The shell prompt is displayed.

Verification—Restoring Files Interactively

1. **List the restored files and directories.**

```
# ls -l
```

A list of files and directories is displayed.

2. **Check the list to be sure all the files and directories you specified in the list have been restored.**

3. **Move the files to the proper directories.**

Example—Restoring Files Interactively

In this example, the files /etc/passwd and /etc/shadow are extracted from the backup tape.

```
# cd /var/tmp
# ufsrestore if /dev/rmt/0
ufsrestore> ls
.:
 .OWdefaults         bin         lib             sbin/
 .Xauthority         dev/        lost+found/     shared/
 .desksetdefaults    devices/    misc/           tmp/
 .fm/                etc/        mnt/            ufsboot
 .mailtool-init      export/     net/            usr/
 .openwin-init       home/       nfs/            var/
 .profile            hsfsboot    opt/            ws/
 .wastebasket/       kadb        proc/
 .xsun.pluto:0       kernel/
ufsrestore> cd etc
ufsrestore> add passwd shadow
ufsrestore> verbose
verbose mode on
ufsrestore> extract
Extract requested files
You have not read any volumes yet.
Unless you know which volume your file(s) are on you should start
with the last volume and work towards the first.
Specify next volume #: 1
extract file ./etc/shadow
extract file ./etc/passwd
Add links
Set directory mode, owner, and times.
set owner/mode for ‘.’? [yn] n
ufsrestore> quit
#
```

How to Restore Specific Files

1. **Become root.**
2. **Write-protect the tape for safety.**
3. **Put the backup tape in the tape drive.**
4. **Change to a directory for restoring files temporarily.**

```
# cd /var/tmp
```

If you want to restore the files to a different directory, substitute the directory name for `/var/tmp` in this step.

5. **Use the** `ufsrestore` **command to restore the file.**

```
# ufsrestore xvf /dev/rmt/n filename
```

In this command,

`x`	Tells `ufsrestore` to copy specific files or directories in the *filename* argument.
`v`	Displays the file names as they are restored.
`f /dev/rmt/`*n*	Identifies the tape device name.
filename	Is one or more individual file or directory names separated by spaces, for example: `./export/home/user1/mail` `./export/home/user2/mail`.

6. **Type the volume number where files are located and press Return.**

```
Specify next volume #: 1
```

The file is restored to the current working directory.

7. **To keep the mode of the current directory unchanged, type** `n` **and press Return at the** `set owner/mode` **prompt.**

```
set owner/mode for '.'? [yn] n
```

Verification—Restoring Specific Files

1. **List the current directory.**
   A listing for the file is displayed.

```
# ls -l filename
```

2. **Move the file to the proper directory.**

```
# mv filename /directory/filename
```

Example—Restoring Specific Files

In this example, the `passwd` and `shadow` files are restored to the `/var/tmp` directory.

```
# cd /var/tmp
# ufsrestore xvf /dev/rmt/0 ./etc/passwd ./etc/shadow
Verify volume and initialize maps
Media block size is 126
Dump   date: Wed Oct 26 14:38:37 1994
Dumped from: the epoch
Level 0 dump of / on pluto:/dev/dsk/c0t3d0s0
Label: none
Extract directories from tape
Initialize symbol table.
Warning: ./etc: File exists
Extract requested files
You have not read any volumes yet.
Unless you know which volume your file(s) are on you should start
with the last volume and work towards the first.
Specify next volume #: 1
extract file ./etc/passwd
Add links
Set directory mode, owner, and times.
set owner/mode for '.'? [yn] n
# cd etc
# mv passwd /etc
# mv shadow /etc
# ls -l etc
```

How to Use a Remote Drive to Restore Files

You can restore files from a remote drive by adding *remote-host*: to the front of the tape device name. Here is the syntax:

```
ufsrestore rf [user@]remote-host:/dev/rmt/n  filename
```

For example, to access a remote tape drive /dev/rmt/0 on the system venus, type:

```
# ufsrestore rf venus:/dev/rmt/0 filename
```

How to Restore a Complete File System

Note – You cannot use this procedure to restore / or /usr. See "How to Restore the root (/) and /usr File Systems" on page 770 for instructions on restoring these file systems.

1. **Become root.**
2. **If necessary, unmount the file system.**

```
# umount /dev/rdsk/device-name
```

3. **Create the new file system with the newfs command.**

```
# newfs /dev/rdsk/device-name
```

You are asked if you want to construct a new file system on the raw device. Verify that the device-name is correct so you don't wipe out the wrong file system.

4. **Confirm that the new file system should be created.**

```
newfs: construct a new file system /dev/rdsk/cntndnsn: (y/n)? y
```

The new file system is created.

5. **Mount the new file system on a temporary mount point.**

```
# mount /dev/dsk/device-name /mnt
```

6. **Change to the `/mnt` directory.**

```
# cd mnt
```

You have changed to the mount-point directory.

7. **Write-protect the tapes.**

8. **Insert the first volume of the level 0 tape in the tape drive.**

9. **Use the `ufsrestore` command to restore the files on the tapes.**

```
# ufsrestore rvf /dev/rmt/n
```

The level 0 dump is restored. If the dump required multiple tapes, you will be prompted to load the next tape.

10. **Remove the tape and load the next tape in the drive.**
Always restore tapes starting with 0 and continuing until you reach the highest level.

11. **Use the `ufsrestore` command to restore the file system.**

```
# ufsrestore rvf /dev/rmt/n
```

The next level tape is restored. If the dump required multiple tapes, you will be prompted to load the next tape.

12. **Remove the** `restoresymtable` **file.**

```
# rm restoresymtable
```

The `restoresymtable` file created by `ufsrestore` is removed.

13. **Change to another directory.**

```
# cd /
```

14. **Unmount the newly restored file system.**

```
# umount /mnt
```

15. **Remove the last tape and insert a new tape that is not write-protected in the tape drive.**

16. **Use the** `ufsdump` **command to back up the newly restored file system.**

```
# ufsdump 0uf /dev/rmt/n /dev/rdsk/device-name
```

You should always do an immediate backup of a newly created file system because `ufsrestore` repositions the files and changes the inode allocation.

17. **Mount the restored file system.**

```
# mount /dev/dsk/device-name /mount-point
```

The restored file system is mounted and available for use.

Example—Restoring a Complete File System

```
# umount /export/home
# newfs /dev/rdsk/c0t3d0s7
newfs: construct a new file system /dev/rdsk/c0t3d0s7: (y/n)? y
/dev/rdsk/c0t3d0s7:41040 sectors in 57 cylinders of 9 tracks, 80
sectors
    21.0MB in 4 cyl groups (16 c/g, 5.90MB/g, 2688 i/g)
super-block backups (for fsck -F ufs -o b=#) at:
 32, 11632, 23232, 34832,
# mount /dev/dsk/c0t3d0s7 /mnt
# cd /mnt
# ufsrestore rvf /dev/rmt/0
Verify volume and initialize maps
Media block size is 126
Dump date: Fri Oct 28 11:00:30 1994
Dumped from: the epoch
Level 0 dump of /export/home on skisun:/dev/dsk/c0t3d0s7
Label: none
Begin level 0 restore
Initialize symbol table.
Extract directories from tape
Calculate extraction list.
Warning: ./lost+found: File exists
Make node ./kryten
Make node ./kryten/letters
Make node ./kryten/memos
Make node ./kryten/reports
Make node ./rimmer
Make node ./rimmer/sc.directives
Make node ./rimmer/tests
Make node ./rimmer/answers
Extract new leaves.
Check pointing the restore
# rm restoresymtable
# cd /
# umount /mnt
# ufsdump 0ucf /dev/rmt/0 /export/home
                    .
                    .
                    .
# mount /dev/dsk/c0t3d0s7 /export/home
# ls /export/home
```

▼ How to Restore the root (/) and /usr File Systems

1. **Add a new system disk to the system where the root (/) and `/usr` file systems will be restored.**
   For a detailed description about adding a system disk, refer to Chapter 29, “SPARC: Adding a Disk,” or Chapter 30, “x86: Adding a Disk.”

2. **Mount the new file system on a temporary mount point.**

```
# mount /dev/dsk/device-name /mnt
```

3. **Change to the `/mnt` directory.**

```
# cd /mnt
```

4. **Create the tape device entries.**

```
# tapes
```

5. **Write-protect the tapes.**

6. **Use the `ufsrestore` command to restore the root file system.**

```
# ufsrestore rvf /dev/rmt/n
```

   The level 0 tape is restored.

7. **Remove the tape and load the next level tape in the drive.**
   Always restore tapes starting with 0 and continuing from lowest to highest level.

8. **Continue to use the `ufsrestore` command as needed.**

```
# ufsrestore rvf /dev/rmt/n
```

   The next level tape is restored.

9. **Repeat steps 7 and 8 for each additional tape.**

10. **Remove the** `restoresymtable` **file.**

```
# rm restoresymtable
```

Removes the `restoresymtable` file that is created and used by `ufsrestore` to check point the restore.

11. **Change to the root (/) directory.**

```
# cd /
```

12. **Unmount the newly created file system.**

```
# umount /mnt
```

13. **Check the new file system.**

```
# fsck /dev/rdsk/device-name
```

The restored file system is checked for consistency.

14. **Insert a new tape in the tape drive.**

15. **Back up the new file system.**

```
# ufsdump 0uf /dev/rmt/n /dev/rdsk/device-name
```

A level 0 backup is performed. Always do an immediate backup of a newly created file system because `ufsrestore` repositions the files and changes the inode allocation.

16. **Repeat steps 5 through 18 for the** `/usr` **file system, if necessary.**

17. Reboot the system.

```
# init 6
```

The system is rebooted.

Example—Restoring the root (/) File System

```
mount /dev/dsk/c0t3d0s0 /mnt
# cd /mnt
# umount /mnt
# tapes
# ufsrestore rvf /dev/rmt/0
# rm restoresymtable
# cd /
# umount /mnt
# fsck /dev/rdsk/c0t3d0s0
# ufsdump 0uf /dev/rmt/0 /dev/rdsk/c0t3d0s0
# init 6
```

Reference Information on `ufsdump` *and* `ufsrestore` 44

This chapter contains reference information on the `ufsdump` and `ufsrestore` commands.

This is a list of reference information in this chapter.

How `ufsdump` *Works*

The `ufsdump` command makes two passes when backing up a file system. On the first pass, it scans the raw device file for the file system and builds a table of directories and files in memory. It then writes the table to the backup media. In the second pass, `ufsdump` goes through the inodes in numerical order, reading the file contents and writing the data to the media.

Determining Device Characteristics

The `ufsdump` command needs to know only an appropriate block size and how to detect the end of media.

Detecting the End of Media

`ufsdump` writes a sequence of fixed-size records. When `ufsdump` receives notification that a record was only partially written, it assumes that it has reached the physical end of the media. This method works for most devices. If a device is not able to notify `ufsdump` that only a partial record has been written, a media error occurs as `ufsdump` tries to write.

Note – DAT devices and 8mm tape devices detect end-of-media. Cartridge tape devices and 1/2 inch tape devices do not detect end-of-media.

Copying Data

The `ufsdump` command copies data only from the raw disk slice. If the file system is still active, anything in memory buffers is probably not copied. The backup done by `ufsdump` does not copy free blocks, nor does it make an image of the disk slice. If symbolic links point to files on other slices, the link itself is copied.

The Role of `/etc/dumpdates`

The `ufsdump` command, when used with the `u` option, maintains and updates a file named `/etc/dumpdates`. Each line in `/etc/dumpdates` shows the file system backed up, the level of the last backup, and the day, date, and time of the backup. Here is a typical `/etc/dumpdates` file from a file server:

```
/dev/rdsk/c0t1d0s0 0 Fri Nov 6 07:54:38 1993
/dev/rdsk/c0t1d0s5 0 Sat Oct 10 07:53:44 1993
/dev/rdsk/c0t1d0s7 0 Sat Oct 10 07:56:57 1993
/dev/rdsk/c0t1d0s6 0 Sat May 23 08:02:34 1993
/dev/rdsk/c0t1d0s0 5 Fri Nov 6 07:55:20 1993
/dev/rdsk/c0t1d0s7 5 Fri Nov 6 07:58:08 1993
/dev/rdsk/c0t1d0s6 5 Fri May 29 09:03:07 1993
/dev/rdsk/c0t1d0s5 9 Thu Nov 5 07:15:51 1993
/dev/rdsk/c0t1d0s4 9 Thu Nov 5 07:18:04 1993
/dev/rdsk/c0t1d0s6 9 Thu Jun 4 09:21:02 1993
```

When you do an incremental backup, the `ufsdump` command consults `/etc/dumpdates` to find the date of the most recent backup of the next lower level. Then it copies to the media all files that were updated since the date of that lower-level backup. After the backup is complete, a new information line, describing the backup you just completed, replaces the information line for the previous backup at that level. On the date that you do a level 0 backup, `/etc/dumpdates` contains one information line for each backed up file system at each level.

Use the `/etc/dumpdates` file to verify that backups are being done. This verification is particularly important if you are having equipment problems. If a backup cannot be completed because of equipment failure, the backup is not recorded in the `/etc/dumpdates` file.

If you need to restore an entire disk, check the `/etc/dumpdates` file for a list of the most recent dates and levels of backups so that you can determine which tapes you need to restore the entire file system.

Note – `/etc/dumpdates` is a text file that can be edited, but edit it only at your own risk. If you make changes to the file that do not match your archive tapes, you may not be able to find the tapes (or files) you need.

Backup Device (dump-file) Argument

The *dump-file* argument (to the `f` option) specifies the destination of the backup, which can be one of the following:

- Local tape drive or diskette drive
- Remote tape drive or diskette drive
- Standard output

Use this argument when the destination is not the default local tape drive `/dev/rmt/0`. If you use the `f` option, then you must specify a value for *dump-file*.

Note – The *dump-file* argument can also point to a file on a local or remote disk, which, if used by mistake, can fill up a file system.

Local Tape or Diskette Drive

Typically, *dump-file* specifies a raw device file for a tape or diskette drive. When `ufsdump` writes to an output device, it creates a single backup file which may span multiple tapes or diskettes.

You specify the tape or diskette device on your system using a device abbreviation. The first device is always 0. For example, if you have a SCSI tape controller and one QIC-24 tape drive that uses medium-density formatting, use this device name:

```
/dev/rmt/0m
```

When you specify a tape device name, you can also type the letter "`n`" at the end of the name to indicate that the tape drive should not rewind after the backup is completed. For example:

```
/dev/rmt/0mn
```

Use the "no-rewind" option if you want to put more than one file onto the tape. If you run out of space during a backup, the tape does not rewind before `ufsdump` asks for a new tape. See "Backup Device Names" on page 819 for a complete description of device naming conventions.

Remote Tape or Diskette Drive

You specify a remote tape or diskette drive using the syntax *host:device*. `ufsdump` writes to the remote device when `root` on the local system has access to the remote system. If you usually run `ufsdump` as root, the name of the local system must be included in the `/.rhosts` file of the remote system. If you specify the device as *user@host:device*, `ufsdump` tries to execute as the specified user on the remote system. In this case, the specified user must have a `.rhosts` file on the remote system to allow the user to access the remote system.

Use the naming convention for the device that matches the operating system for the system on which the device resides, not the system from which you run the `ufsdump` command. If the drive is on a system that is running a previous SunOS release (for example, 4.1.1), use the SunOS 4.1.x device name (for example, `/dev/rst0`). If the system is running Solaris software, use the SunOS 5.x convention (for example, `/dev/rmt/0m`).

Note – You must specify remote devices explicitly with the *dump-file* argument. In previous SunOS releases `rdump` directed the output to the remote device defined by the dumphost alias. `ufsdump` does not have an `rufsdump` counterpart.

Standard Output

When you specify a dash (`-`) as the *dump-file* argument, `ufsdump` writes to the standard output.

Note – The `v` option (verify) does not work when the *dump-file* argument is standard output.

You can use the `ufsdump` and `ufsrestore` commands in a pipeline to copy a file system by writing to the standard output with `ufsdump` and reading from the standard input with `ufsrestore`, as shown in this example:

```
# ufsdump 0f - /dev/rdsk/c0t0d0s7 | (cd /home; ufsrestore xf -)
```

Specifying Files to Back Up

You must always include *files-to-backup* as the last argument on the command line. This argument specifies the source or contents of the backup. It usually identifies a file system but can also identify individual files or directories.

For a file system, specify the raw device file for a disk slice. It includes the disk controller abbreviation (`c`), the target number (`t`) for SCSI devices only, a number indicating the disk number (`d`), and the slice number (`s`). For example, if you have a SCSI disk controller on your standalone system (or server) and you want to back up `/usr` located in slice 6, specify the device as follows:

`/dev/rdsk/c0t0d0s6`

You can specify the file system by its mount point directory (for example, `/home`), as long as there is an entry for it in the `/etc/vfstab` file.

See “Backup Device Names” on page 819 for a complete description of device naming conventions.

For individual files or directories, type one or more names separated by spaces.

Note – When you use `ufsdump` to back up one or more directories (rather than a whole file system), a level 0 backup is done. Incremental backups do not apply.

End-of-Media Detection

`ufsdump` automatically detects the end-of-media for most devices. Therefore, you do not usually need to use the `c`, `d`, `s`, and `t` options to perform multivolume backups.

The only time you need to use the end-of-media options is when `ufsdump` does not understand the way the device detects the end-of-media or you are going to restore the files on a system with an older version of the `restore` command. To ensure compatibility with older versions of the `restore` command, the size option can still force `ufsdump` to go to the next tape or diskette before reaching the end of the current tape or diskette.

Specifying Tape Characteristics

If you do not specify any tape characteristics, the `ufsdump` command uses a set of defaults. Table 44-1 shows some arguments to the `ufsdump` command that work well for different types of tape cartridges. You can specify tape cartridge (`c`), density (`d`), size (`s`), and number of tracks (`t`). Note that you can specify the options in any order as long as the arguments that follow match the order of the options.

Table 44-1 Arguments to `ufsdump` to Specify Tape Capacity

Tape	Arguments
Diskette	`ufsdump Ds 1422`
60-Mbyte cartridge	`ufsdump cdst 1000 425 9`
150-Mbyte cartridge	`ufsdump cdst 1000 700 18`
1/2-inch tape	`ufsdump dsb 1600 2300 126`
2.3-Gbyte 8-mm tape	`ufsdump dsb 54000 6000 126`
5.0-Gbyte 8-mm tape	`ufsdump dsb 54000  13000 126`
5.0-Gbyte 4-mm tape	`ufsdump b 96`

Limitations of the `ufsdump` *Command*

Table 44-2 lists tasks you cannot perform with the `ufsdump` command.

Table 44-2 Tasks You Cannot Perform With The `ufsdump` Command

The `ufsdump` Command Does Not ...	Comments
Automatically calculate the number of tapes or diskettes needed for backing up file systems.	You can use the dry run mode (S option) to determine the amount of space that is needed before actually backing up file systems.
Provide built-in error checking to minimize problems when backing up an active file system.	—
Enable you to back up files that are remotely mounted from a server.	Files on the server must be backed up on the server itself. Users are denied permission to run `ufsdump` on files they own that are located on a server.

Options and Arguments for the `ufsdump` Command

This section describes in detail the options and arguments for the `ufsdump` command. The syntax for the `ufsdump` command is:

```
/usr/sbin/ufsdump [options] [arguments] files-to-back-up
```

options	Is a single string of one-letter option names.
arguments	Identifies option arguments and may be multiple strings. The option letters and the arguments that go with them must be in the same order
files-to-back-up	Identifies the files to back up and these arguments must always come last.

Default Command Options

If you run the `ufsdump` command without any options, use this syntax:

```
# ufsdump files-to-back-up
```

`ufsdump` uses these options, by default:

```
ufsdump 9uf /dev/rmt/0 files-to-back-up
```

These options do a level 9 incremental backup to the default tape drive at its preferred density.

Options for the `ufsdump` *Command*

Table 44-3 describes the options for the `ufsdump` command.

Table 44-3 Options for the `ufsdump` Command

Option	Description
`0-9`	Backup level. Level 0 is for a full backup of the whole file system specified by *files-to-backup*. Levels 1–9 are for incremental backups of files that have changed since the last lower-level backup.
`a` *archive-file*	Archive file. Store (archive) a backup table of contents in a specified file on the disk. The file can be understood only by `ufsrestore`, which uses it to determine whether a file to be restored is present in a backup file, and if so, on which volume of the media it resides.
`b` *factor*	Blocking factor. The number of 512-byte blocks to write to tape per operation.
`c`	Cartridge. Back up to cartridge tape. When end-of-media detection applies, this option sets the block size to 126.
`d` *bpi*	Tape density. You need to use this option only when `ufsdump` cannot detect the end of the media.
`D`	Diskette. Back up to diskette.
`f` *dump-file*	Dump file. Write the files to the destination specified by *dump-file* instead of the default device. If the file is specified as *user@system*`:`*device*, `ufsdump` attempts to execute as the specified user on the remote system. The specified user must have a `.rhosts` file on the remote system that allows the user invoking the command on the local system to access the remote system.
`l`	Autoload. Use this option if you have an autoloading (stackloader) tape drive. When the end of a tape is reached, this option takes the drive offline and waits up to two minutes for the tape drive to be ready again. If the drive is ready within two minutes, it continues. If it is not ready after two minutes, it prompts the operator to load another tape.
`n`	Notify. When intervention is needed, send a message to all terminals of all users in the `sys` group.
`o`	Offline. When finished with a tape or diskette, take the drive offline, rewind (if tape), and if possible remove the media (for example, eject a diskette or remove 8-mm autoloaded tape).

Table 44-3 Options for the `ufsdump` Command *(Continued)*

Option	Description
`s` *size*	Size. Specify the length of tapes in feet or number of 1024-byte blocks for diskettes. You need to use this option only when `ufsdump` cannot detect the end of the media.
`S`	Estimate size of backup. Determine the amount of space that is needed to perform the backup, without actually doing it, and output a single number indicating the estimated size of the backup in bytes.
`t` *tracks*	Tracks. Specify the number of tracks for 1/4-inch cartridge tape. You need to use this option only when `ufsdump` cannot detect the end of the media.
`u`	Update the dump record. For a completed backup on a file system, add an entry to the file `/etc/dumpdates`. The entry indicates the device name for the file system's disk slice, the backup level (0–9), and the date. No record is written when you do not use the `u` option or when you back up individual files or directories. If a record already exists for a backup at the same level, it is replaced.
`v`	Verify. After each tape or diskette is written, verify the contents of the media against the source file system. If any discrepancies occur, prompt the operator to mount new media, then repeat the process. Use this option on an unmounted file system only, because any activity in the file system causes it to report discrepancies.
`w`	Warning. List the file systems appearing in `/etc/dumpdates` that have not been backed up within a day. When you use this option all other options are ignored.
`W`	Warning with highlight. Show all the file systems that appear in `/etc/dumpdates` and highlight those file systems that have not been backed up within a day. When you use this option all other options are ignored.

Note – The `/etc/vfstab` file does not contain information about how often to back up a file system.

The `ufsdump` Command and Security Issues

If you are concerned about security:

- Require root access for the `ufsdump` command.
- Ensure root access entries are removed from `/.rhost` files on clients and servers if doing centralized backups.

For more general information on security, see "Managing System Security" in *System Administration Guide, Volume II*.

Options and Arguments for the `ufsrestore` Command

Command Syntax

The syntax of the `ufsrestore` command is:

```
ufsrestore [options] [arguments] [filename(s) …]
```

options	Is a single string of one-letter option names. You must choose one and *only* one of these options: `i`, `r`, `R`, `t`, or `x`.
arguments	Follows the option string with the arguments that match the options. The option names and the arguments that go with them must be in the same order.
filename(s)	Specifies files to be restored as arguments to the `x` or `t` options, and must always come last.

Options and Arguments

You must use one (and only one) of the ufsrestore options shown in Table 44-4.

Table 44-4 One Required Option for the ufsrestore Command

Option	Description
i	Interactive. Runs ufsrestore in an interactive mode. In this mode, you can use a limited set of shell commands to browse the contents of the media and select individual files or directories to restore. See "Commands for Interactive Restore" on page 786 for a list of available commands.
r	Recursive. Restores the entire contents of the media into the current working directory (which should be the top level of the file system). Information used to restore incremental dumps on top of the full dump (e.g., restoresymtable) is also included. To completely restore a file system, use this option to restore the full (level 0) dump and then for each incremental dump. Although intended for a new file system (one just created with the newfs command), the file system may contain files and files not on the backup media are preserved.
R	Resume restoring. Prompts for the volume from which to resume restoring and restarts from a checkpoint. You rerun the ufsrestore command with this option after a full restore (r option) is interrupted.
x [*filename(s)*…]	Extract. Selectively restores the files you specify by the *filename(s)* argument. *filename(s)* can be a list of files and directories. All files under a specified directory are restored unless you also use the h option also. If you omit *filename(s)* or enter "." for the root directory, all files on all volumes of the media (or from standard input) are restored. Existing files are overwritten, and warnings are displayed.
t [*filename(s)*…]	Table of contents. Checks the files specified in the *filename(s)* argument against the media. For each file, lists the full file name and the inode number (if the file is found) or indicates the file is not on the "volume" (meaning any volume in a multivolume dump). If you do not enter the *filename(s)* argument, all files on all volumes of the media are listed (without distinguishing on which volume files are located). If you also use the h option, only the directory files specified in *filename(s)*, not their contents, are checked and listed. The table of contents is read from the first volume of the media, or, if you use the a option, from the specified archive file. This option is mutually exclusive with the x and r options.

In addition to one of the options shown in Table 44-4, you can choose from the options shown in Table 44-5.

Table 44-5 Additional Options for the `ufsrestore` Command

Option	Description
`a` *archive-file* [*filename(s)*…]	Takes the dump table of contents from the specified *archive-file* instead of from the media (first volume). You can use this option in combination with the `t`, `i`, or `x` options to check for the files in the dump without having to mount any media. If you use it with the `x` and interactive extract options, you will be prompted to mount the appropriate volume before extracting the file(s).
`b` *factor*	Blocking factor. Number of 512-byte blocks to write to tape per operation. By default, `ufsrestore` tries to figure out the block size used in writing the tape.
`d`	Debug. Turn on debugging messages.
`f` *backup-file*	Backup file. Reads the files from the source indicated by *backup-file*, instead of from the default device file `/dev/rmt/0m`. If you use the `f` option, you must specify a value for *backup-file*. When *backup-file* is of the form *system:device*, `ufsrestore` reads from the remote device. You can also use the *backup-file* argument to specify a file on a local or remote disk. If *backup-file* is '-', the files are read from standard input.
`h`	Turns off directory expansion. Only the directory file you specify is extracted or listed.
`m`	Restores specified files into the current directory on the disk regardless of where they are located in the backup hierarchy and renames them with their inode number. For example, if the current working directory is `/files`, a file in the backup named `./dready/fcs/test` with inode number 42, is restored as `/files/42`. This option is useful only when you are extracting a few files.
`s` *n*	Skips to the *n*th backup file on the media. This option is useful when you put more than one backup on a single tape.
`v`	Verbose. Displays the names and inode numbers of each file as it is restored.
`y`	Continues when errors occur reading the media and tries to skip over bad blocks instead of stopping and asking whether to abort. This option tells the command to assume a yes response.

Commands for Interactive Restore

Table 44-6 Commands for Interactive Restore

Option	Description
`ls` [*directory-name*]	Lists the contents of either the current directory or the specified directory. Directories are marked by a `/` suffix and entries in the current list to be restored (extracted) are marked by an `*` prefix. Inode numbers are shown if the verbose option is used.
`cd` *directory-name*	Changes to the specified directory in the backup hierarchy.
`add` [*filename*]	Adds the current directory or the specified file or directory to the list of files to extract (restore). If you do not use the `h` option, all files in a specified directory and its subdirectories are added to the list. Note that all the files you want to restore to a directory might not be on a single backup tape or diskette. You might need to restore from multiple backups at different levels to get all the files.
`delete` [*filename*]	Deletes the current directory or the specified file or directory from the list of files to extract (restore). If you do not use the `h` option, all files in the specified directory and its subdirectories are deleted from the list. Note that the files and directories are deleted only from the extract list you are building. They are not deleted from the media.
`extract`	Extracts the files in the list and restores them to the current working directory on the disk. Specify `1` when asked for a volume number. If you are doing a multitape or multidiskette restore and restoring a small number of files, start with the last tape or diskette.
`help`	Displays a list of commands you can use in interactive mode.
`pwd`	Displays the path name of the current working directory in the backup hierarchy.
`q`	Quits interactive mode without restoring any additional files.

Table 44-6 Commands for Interactive Restore *(Continued)*

Option	Description
setmodes	Lets you set the mode for files to be restored to match the mode of the root directory of the file system from which they were backed up. You are prompted with: set owner/mode for '.' [yn]? Type y (for yes) to set the mode (permissions, owner, times) of the current directory to match the root directory of the file system from which they were backed up. Use this mode when restoring a whole file system. Type n (for no) to leave the mode of the current directory unchanged. Use this mode when restoring part of a backup to a directory other than the one from which the files were backed up.
verbose	Turns on or off the verbose option (which can also be entered as v on the command line outside of interactive mode). When verbose is on, the interactive ls command lists inode numbers and the ufsrestore command display information on each file as it is extracted.
what	Displays the backup header on the tape or diskette.

Copying UFS Files and File Systems 45

This chapter describes how copy UFS files and file systems to disk, tape, and diskettes using various backup commands.

This is a list of the step-by-step instructions in this chapter.

45

Commands for Copying File Systems

When you need to back up and restore complete file systems, use the `ufsdump` and `ufsrestore` commands described in Chapter 44, "Reference Information on ufsdump and ufsrestore." When you want to copy or move individual files, portions of file systems, or complete file systems, you can use the procedures described in this chapter as an alternative to `ufsdump` and `ufsrestore`.

Table 45-1 describes when to use the various backup commands.

Table 45-1 When to Use Various Backup Commands

If You Want To ...	Then Use ...	Page Reference
Back up file systems to tape	`ufsdump`	page 738
Restore file systems from tape	`ufsrestore`	page 766
Transport files to other systems	`pax`, `tar`, or `cpio`	page 797
Copy files or file systems to disk	`dd`	page 792
Copy files to diskette	`tar`	page 811

Table 45-2 describe various backup and restore commands.

Table 45-2 Summary of Various Backup Commands

Command Name	Aware of File System Boundaries?	Support Multi-Volume Backups?	Physical or Logical Copy?
`volcopy`	Yes	Yes	Physical
`tar`	No	No	Logical
`cpio`	Yes	Yes	Logical
`pax`	Yes	Yes	Logical
`dd`	No	Yes	Physical
`ufsdump/ ufsrestore`	Yes	Yes	Logical

These sections describe the advantages and disadvantages of each method and provide examples of how to use the commands.

Copying File Systems to Disk

Two commands are used to copy file systems to disk:

- volcopy
- dd

The next section describes how to use the dd command to copy file systems to disk.

Making a Literal File System Copy

The dd command makes a literal (block) copy of a complete UFS file system to another file system or to a tape. By default, the dd command copies its standard input to its standard output.

Note – Do not use the dd command with variable-length tape drives.

You can specify a device name in place of the standard input or the standard output or both. In this example, contents of the diskette are copied to a file in the /tmp directory:

```
$ dd < /floppy/floppy0 > /tmp/output.file
2400+0 records in
2400+0 records out
```

The dd command reports on the number of blocks it reads and writes. The number after the + is a count of the partial blocks that were copied.

The dd command syntax is different from most other commands. Options are specified as *keyword*=*value* pairs, where *keyword* is the option you want to set and *value* is the argument for that option. For example, you can replace the standard input and output with this syntax:

dd if=*input-file* of=output-file

For example, to use the *keyword*=*value* pairs instead of the redirect symbols in the previous example, you would type:

```
$ dd if=/floppy/floppy0 of=/tmp/output.file
```

▼ How to Clone a Disk (dd)

1. **Make sure the source and destination disks have the same disk geometry.**
2. **Become root.**
3. **Create the** `/reconfigure` **file on the system with the master disk so that it will recognize the clone disk once it is rebooted.**
4. **Shut down the system.**

```
# init 0
```

5. **Attach the clone disk to the system.**
6. **Boot the system.**

```
ok boot
```

7. **Use the** `dd` **command to copy the master disk to the clone disk.**

```
# dd if=/dev/dsk/device-name of=/dev/dsk/device-name bs=blocksize
```

In this command,

`if=/dev/dsk/`*device-name*	Represents the master disk device as the input device.
`of=/dev/dsk/`*device-name*	Represents the clone disk device as the output device.
`bs=`*blocksize*	Is the block size.

8. **Mount the clone disk's root (**`/`**) file system.**

```
# mount /dev/dsk/device-name /mnt
```

9. **Edit the clone disk's** `/etc/vfstab` **to reference the correct device names.**

10. **Unmount the clone disk's root (/) file system.**

```
# umount /mnt
```

11. **Shut down the system.**

```
# init 0
```

12. **Boot from the clone disk to single-user mode.**

```
# boot diskn -s
```

13. **Unconfigure the clone disk.**

```
# sys-unconfig
```

The system is shut down after it is unconfigured.

14. **Boot from the clone disk again and provide its system information, such as host name, time zone, etc.**

```
# boot diskn
```

Example—Cloning a Disk (`dd`*)*

```
# init 0
ok boot
# dd if=/dev/dsk/c0t0d0s2 of=/dev/dsk/c0t2d0s2 bs=100k
# mount /dev/dsk/c0t2d0s2 /mnt
# cd /mnt/etc
# vi vfstab
# cd /
# umount /mnt
# init 0
# book disk2 -s
# sys-unconfig
# book disk2
```

Verification—Cloning a Disk (`dd`*)*

Once the disk is configured, log in as root to verify the system information.

Copying Directories Between File Systems using the `cpio` *Command*

You can use the `cpio` (copy in and out) command to copy individual files, groups of files, or complete file systems. This section describes how to use the `cpio` command to copy complete file systems.

The `cpio` command is an archiving program that takes a list of files and copies them into a single, large output file. It inserts headers between the individual files to facilitate recovery. You can use the `cpio` command to copy complete file systems to another slice, another system, or to a media device such as tape or diskette.

Because the `cpio` command recognizes end-of-media and prompts you to insert another volume, it is the most effective command (other than `ufsdump`) to use to create archives that require multiple tapes or diskettes.

You frequently use commands like `ls` and `find` to list and select the files you want to copy and then pipe the output to the `cpio` command.

▼ How to Copy Directories Between File Systems (`cpio`)

1. **Become root.**

2. **Change to the appropriate directory.**

```
# cd /filesystem1
```

3. **Copy the directory tree using a combination of the** `find` **and** `cpio` **commands.**

```
# find . –print –depth | cpio –pdm /filesystem2
```

In this command,

.	Starts in the current working directory.
`-print`	Prints the file names.

`-depth`	Descends the directory hierarchy and prints file names on the way back up.
`-p`	Creates a list of files.
`-d`	Creates directories as needed.
`-m`	Sets the correct modification times on directories.

The files from the directory name you specify are copied and symbolic links are preserved.

4. If appropriate, remove the source directory.

```
# rm -rf /filesystem1
```

Verification—Copying Directories Between File Systems (`cpio`*)*

Display the contents of the destination directory to verify the copy was successful.

```
# cd /filesystem2
# ls
```

Example—Copying Directories Between File Systems (`cpio`*)*

```
# cd /data1
# find . –print –depth | cpio –pdm /data2
19013 blocks
# cd /data2
# ls
# rm -rf /data1
```

See `cpio(1)`for more information.

Copying Files and File Systems to Tape

The pax, tar and cpio commands can be used to copy files and file systems to tape. The command you choose depends on how much flexibility and precision you require for the copy.

Use pax to copy files and directory subtrees to a single tape. This command provides better portability that tar or cpio.

Use tar to copy files and directory subtrees to a single tape. Note that the SunOS 5.x tar command can archive special files (block and character devices, FIFOs) but the SunOS 4.x tar command cannot extract them.

Use cpio to copy arbitrary sets of files, special files, or file systems that require multiple tape volumes or when you want to copy files from SunOS 5.x systems to SunOS 4.x systems. The cpio command packs data onto tape more efficiently than tar and skips over any bad spots in a tape when restoring. The cpio command also provides options for writing files with different header formats (tar, ustar, crc, odc, bar) for portability between systems of different types.

Because tar and cpio use the raw device, you do not need to format or make a file system on tapes before you use them. The tape drive and device name you use depend on the hardware and configuration for each system. See "Choosing Which Media to Use" on page 818 for more information about tape drives and device names.

Copying Files to Tape With pax

The pax command is used to copy files and directories to a single tape.

▼ How to Copy Files to a Tape (pax)

1. **Change to the directory that contains the files you want to copy.**
2. **Insert a write-enabled tape into the tape drive.**
3. **Copy the files to tape with the pax command.**

```
$ pax -w -f /dev/rmt/0 .
```

In this command,

w	Copies the current directory contents to tape.
`f /dev/rmt/0`	Identifies the tape drive.

4. **Remove the tape from the drive and write the names of the files on the tape label.**

Copying Files to Tape With `tar`

Things You Should Know Before Copying Files to Tape With `tar`

- Copying files to a tape using the `c` option to `tar` destroys any files already on the tape.
- You can use metacharacters (`?` and `*`) as part of the file names you specify. For example, to copy all documents with a `.doc` suffix, type `*.doc` as the file name argument.

▼ How to Copy Files to a Tape (`tar`)

1. **Change to the directory that contains the files you want to copy.**
2. **Insert a write-enabled tape into the tape drive.**
3. **Copy the files to tape with the `tar` command.**

```
$ tar cvf /dev/rmt/n filename filename filename
```

In this command,

`c`	Indicates you want to create an archive.
`v`	Displays the name of each file as it is archived.
`f /dev/rmt/`*n*	Indicates that the archive should be written to the specified device or file.
filename	Indicates the files you want to copy.

The file names you specify are copied to the tape, overwriting any existing files on the tape.

4. **Remove the tape from the drive and write the names of the files on the tape label.**

Verification—Copying Files to a Tape (`tar`)

Verify that the files copied are on the tape using the `tar` command with the `t` option, which displays the tape's contents. See "How to List the Files on a Tape (tar)" on page 800 for more information on listing files on a tar tape.

```
$ tar tvf /dev/rmt/n
```

Example—Copying Files to a Tape (`tar`)

In this example, three files are copied to a tape in tape drive 0.

```
$ cd /export/home/kryten
$ ls reports
reportA reportB reportC
$ tar cvf /dev/rmt/0 reports
a reports/ 0 tape blocks
a reports/reportA 2 tape blocks
a reports/reportB 5 tape blocks
a reports/reportC 6 tape blocks
$
```

▼ How to List the Files on a Tape (`tar`)

1. **Insert a tape into the tape drive.**
2. **Display the tape contents with the `tar` command.**

```
$ tar tvf /dev/rmt/n
```

In this command,

`t`	Lists the table of contents for the files on the tape.
`v`	Used with the `t` option provides detailed information about the files on the tape.
`f /dev/rmt/`*n*	Indicates the tape device.

Example—Listing the Files on a Tape (`tar`)

In this example, the table of contents for the tape in drive 0 contains three files.

```
$ tar tvf /dev/rmt/0
drwxr-xr-x101/100    0 Oct 24 11:05 1994 reports/
-rw-r--r--101/100  611 Oct 24 11:05 1994 reports/reportA
-rw-r--r--101/100 2322 Oct 24 11:05 1994 reports/reportB
-rw-r--r--101/100 2596 Oct 24 11:05 1994 reports/reportC
```

▼ How to Retrieve Files From a Tape (`tar`)

1. **Change to the directory where you want to put the files.**
2. **Insert the tape into the tape drive.**
3. **Retrieve files from the tape using the `tar` command.**

```
$ tar xvf /dev/rmt/n filename filename
```

In this command,

`x`	Indicates that files should be extracted from the specified archive file. All of the files on the tape in the specified drive are copied to the current directory.
`v`	Displays the name of each file as it is archived.
`f /dev/rmt/`*n*	Indicates the tape device containing the archive.

Example—Retrieving the Files on a Tape (`tar`)

In this example, all files are copied from the tape in drive 0.

```
$ cd /usr/tmp
$ tar xvf /dev/rmt/0
x reports/, 0 bytes, 0 tape blocks
x reports/reportA, 0 bytes, 0 tape blocks
x reports/reportB, 0 bytes, 0 tape blocks
x reports/reportC, 0 bytes, 0 tape blocks
x reports/reportD, 0 bytes, 0 tape blocks
```

Note – The names of the files extracted from the tape must exactly match the names of the files stored on the archive. If you have any doubts about the names or paths of the files, first list the files on the tape. See "How to List the Files on a Tape (tar)" on page 800 for instructions.

See `tar(1)` for more information.

▼ How to Copy All Files in a Directory to a Tape (`cpio`)

1. **Insert a tape that is not write-protected into the tape drive.**

2. **Copy files to a tape using the `ls` and `cpio` commands.**

```
$ ls | cpio -oc > /dev/rmt/n
```

In this command,

`ls`	Provides the `cpio` command with a list of file names.
`cpio -oc`	Specifies that `cpio` should read from standard output (`-o`) and read information in ASCII character format (`c`).
`> /dev/rmt/`*n*	Specifies the output file.

All of the files in the directory are copied to the tape in the drive you specify, overwriting any existing files on the tape. The total number of blocks copied is displayed.

3. **Remove the tape from the drive and write the names of the files on the tape label.**

Verification—Copying All Files in a Directory to a Tape (`cpio`)

```
$ cpio -civt < /tmp/cpio.file
drwxr-xr-x   2 kryten   users    0     Oct 24 11:05 1994, letters
drwxr-xr-x   2 kryten   users    0     Oct 24 11:05 1994, memos
drwxr-xr-x   2 kryten   users    0     Nov  8 14:14 1994, reports
8 blocks
$
```

Example—Copying All Files in a Directory to a Tape (`cpio`)

In this example, all of the files in the directory `/export/home/kryten` are copied to the tape in tape drive 0.

```
$ cd /export/home/kryten
$ ls | cpio -oc > /dev/rmt/0
8 blocks
$
```

▼ How to List the Files on a Tape (`cpio`)

Note – Listing the table of contents takes as long as it does to read the archive file because the `cpio` command must process the entire archive.

---

1. **Insert a tape into the tape drive.**
2. **List the files on tape using the `cpio` command.**

```
$ cpio -civt < /dev/rmt/n
```

In this command,

`c`	Specifies that `cpio` should read files in ASCII character format.
`i`	Reads in the contents of the tape.

`v`	Displays the output in a format similar to the output from the `ls -l` command.
`t`	Lists the table of contents for the files on the tape in the tape drive you specify.
`< /dev/rmt/`*n*	Specifies the output file.

Example—Listing the Files on a Tape (`cpio`)

In this example, the table of contents for the tape in drive 0 contains three files.

```
$ cpio -civt < /dev/rmt/0
drwxr-xr-x  2 rimmer   users   0   Oct 28 09:17 1994, answers
drwxr-xr-x  2 rimmer   users   0   Oct 28 09:17 1994, sc.directives
drwxr-xr-x  2 rimmer   users   0   Oct 28 09:17 1994, tests
8 blocks
```

▼ How to Retrieve All Files From a Tape (`cpio`)

If the archive was created using relative path names, the input files are built as a directory within the current directory when you retrieve the files. If, however, the archive was created with absolute path names, the same absolute paths are used to re-create the file on your system.

Caution – Using absolute path names can be dangerous because you will overwrite the original files.

1. **Change to the directory where you want to put the files.**
2. **Insert the tape into the tape drive.**

3. **Copy all files from the tape to the current directory using the `cpio` command.**

```
$ cpio -icv < /dev/rmt/n
```

In this command,

`i`	Reads in the contents of the tape.
`c`	Specifies that `cpio` should read files in ASCII character format.
`v`	Displays the output in a format similar to the output from the `ls -l` command.
`< /dev/rmt/`*n*	Specifies the output file.

Example—Retrieving All Files From a Tape (`cpio`)

In this example, all files are copied from the tape in drive 0.

```
$ cd /usr/tmp
cpio -icv < /dev/rmt/0
answers
sc.directives
tests
8 blocks
```

▼ How to Retrieve Specific Files From a Tape (cpio)

1. **Change to the directory where you want to put the files.**
2. **Insert the tape into the tape drive.**
3. **Retrieve a subset of files from a tape using the `cpio` command.**

```
$ cpio -icv "*file" < /dev/rmt/n
```

In this command,

`i`	Reads in the contents of the tape.
`c`	Specifies that `cpio` should read files in ASCII character format.
`v`	Displays the output in a format similar to the output from the `ls -l` command.
`"*`*file*`"`	Specifies that all of the files that match the pattern are copied to the current directory. You can specify multiple patterns, but each must be enclosed in double quotation marks.
`< /dev/rmt/`*n*	Specifies the output file.

Example—Retrieving Specified Files From a Tape (`cpio`*)*

In this example, all files with suffix `chapter` are copied from tape drive 0.

```
$ cd /home/smith/Book
$ cpio -icv "*chapter" < /dev/rmt/0
Boot.chapter
Directory.chapter
Install.chapter
Intro.chapter
31 blocks
```

See `cpio(1)` for more information.

How to Copy Files to a Remote Tape Drive (`tar` and `dd`)

1. **The following prerequisites must be met to use a remote tape drive:**

- A user entry must be in the remote system's `/.rhost` file
- An entry for the remote system must be in the local system's `/etc/inet/hosts` file

2. **To copy files to a remote tape drive use the `tar` and `dd` commands.**

```
$ tar cf - files | rsh remotehost dd of=/dev/rmt/n bs=blocksize
```

In this command,

`tar cf`	Creates a tape archive and specifies the tape device.
`-`	Represents a placeholder for the tape device.
files	Identifies files to be copied.
| *rsh remotehost*	Pipe `tar` command to a remote shell to copy the files.
`dd of=`*/dev/rmt/w*	Represents the output device.
`bs=`*blocksize*	Represents the blocking factor.

3. **Remove the tape from the drive and write the names of the files on the tape label.**

Example—Copying Files to a Remote Tape Drive (`tar` and `dd`)

```
# tar cvf - * | rsh mercury dd of=/dev/rmt/0 bs=126b
a answers/ 0 tape blocks
a answers/test129 1 tape blocks
a sc.directives/ 0 tape blocks
a sc.directives/sc.190089 1 tape blocks
a tests/ 0 tape blocks
a tests/test131 1 tape blocks
0+2 records in
0+2 records out
```

▼ How to Extract Files From a Remote Tape Drive

1. **Change to a temporary directory.**

```
$ cd /usr/tmp
```

2. **To extract files to a remote tape drive use the `tar` and `dd` commands.**

```
$ rsh remotehost dd if=/dev/rmt/n | tar xvBpf -
```

In this command,

`rsh` *remotehost*	Is a remote shell that is started to extract the files from the tape device using the `dd` command.
`dd if=/dev/rmt/`*n*	Represents in the input device.
`| tar xvBpf -`	Pipes the output of the `dd` command to the `tar` command used to restored the files.

Verification—Extracting Files From a Remote Tape Drive

Verify that the files have been extracted.

```
$ ls -l /usr/tmp
```

Example—Extracting Files From a Remote Tape Drive

```
# rsh mercury dd if=/dev/rmt/0 | tar xvBpf -
x answers/, 0 bytes, 0 tape blocks
x answers/test129, 48 bytes, 1 tape blocks
20+0 records in
20+0 records out
x sc.directives/, 0 bytes, 0 tape blocks
x sc.directives/sc.190089, 77 bytes, 1 tape blocks
x tests/, 0 bytes, 0 tape blocks
x tests/test131, 84 bytes, 1 tape blocks
```

Copying Files and File Systems to Diskette

Before you can copy files or file systems to diskette, you must format the diskette. See Chapter 14, "Formatting and Using Diskettes From the Command Line" for information on how to format a diskette.

Use the `tar` command to copy UFS files to a single formatted diskette.

Use the `cpio` command if you need to copy UFS files to multiple formatted diskettes. `cpio` recognizes end-of-media and prompts you to insert the next volume.

Note – Using the `cpio` command to copy UFS files to multiple formatted diskettes is not a straightforward procedure because of Volume Management.

Use double-sided high-density 3.5-inch diskettes (diskettes are marked "DS, HD").

Things You Should Know When Copying Files to Diskettes

- If the diskette contains a mounted file system, you must unmount the file system before running `tar`. You can unmount the file system using the `umount` command, or you can unmount the file system and format the diskette using the `fdformat -U` command.
- Copying files to a formatted diskette using the `c` option of `tar` destroys any files already on the diskette.
- Reformatting a diskette destroys any files or archives that were already on the diskette.

▼ How to Copy Files to a Single Formatted Diskette (`tar`)

1. **Change to the directory that contains the files you want to copy.**
2. **Insert a formatted diskette that is not write-protected into the drive.**
3. **Make the diskette available using the `volcheck` command.**
4. **Unmount any file system on the diskette and reformat it.**

```
$ fdformat -U /vol/dev/aliases/floppy0
```

5. **Copy the files to diskette using the `tar` command.**

```
$ tar cvf /vol/dev/rdiskette0/unlabeled filename filename filename
```

The file names you specify are copied to the diskette, overwriting any existing files on the diskette.

6. **Remove the diskette from the drive.**
7. **Write the names of the files on the diskette label.**

Verification—Copying Files to a Single Formatted Diskette (`tar`*)*

```
$ tar tvf /vol/dev/rdiskette0/unlabeled
```

Verify that the files copied are on the diskette using the `tar` command with the `t` option, which displays the diskette's contents. See "How to List the Files on a Diskette (tar)" for more information on listing files.

Example—Copying Files to a Single Formatted Diskette (`tar`*)*

In this example, two files are copied to a diskette:

```
$ cd /home/smith
$ ls evaluation*
evaluation.doc   evaluation.doc.backup
$ tar cvf /vol/dev/rdiskette0/unlabeled evaluation*
a evaluation.doc 86 blocks
a evaluation.doc.backup 84 blocks
```

▼ How to List the Files on a Diskette (`tar`)

1. **Insert a diskette into the drive.**
2. **Run** `volcheck` **to make the diskette available.**
3. **Use the** `tar` **command to list the files on a diskette.**

```
$ tar tvf /vol/dev/rdiskette0/unlabeled
```

Example—Listing the Files on a Diskette (`tar`*)*

In this example, the table of contents for the diskette shows two files:

```
$ tar tvf /vol/dev/rdiskette0/unlabeled
rw-rw-rw-6693/10  44032 Apr 23 14:54 1994 evaluation.doc
rw-rw-rw-6693/10  43008 Apr 23 14:47 1994 evaluation.doc.backup
$
```

See `tar(1)` for more information.

If you need a multiple-volume interchange utility, use the `cpio` command. The `tar` command is only a single-volume utility.

▼ How to Retrieve Files From a Diskette (`tar`)

1. **Change to the directory where you want to put the files.**
2. **Insert the diskette into the drive.**
3. **Run `volcheck` to make the diskette available.**
4. **Use the `tar` command to retrieve files from a diskette.**

   ```
   $ tar xvf /vol/dev/rdiskette0/unlabeled
   ```

   All of the files on the diskette are copied to the current directory.

5. **Remove the diskette from the drive.**

Examples—Retrieving Files From a Diskette (`tar`)

In this example, all files are copied from the diskette:

```
$ cd /home/smith/Evaluations
$ tar xvf /vol/dev/rdiskette0/unlabeled
x evaluation.doc, 44032 bytes, 86 tape blocks
x evaluation.doc.backup, 43008 bytes, 84 tape blocks
```

Use the `tar` command to retrieve individual files from a diskette.

```
$ tar xvf /vol/dev/rdiskette0/unlabeled filename filename filename
```

The file names you specify are extracted from the diskette and placed in the current working directory.

Verification—Retrieving Files From a Diskette (`tar`)

```
$ ls /directory
```

▼ How to Archive Files to Multiple Diskettes

If you are copying large files or file systems onto diskettes, you want to be prompted to replace a full diskette with another formatted diskette. The `cpio` command provides this capability. The `cpio` commands you use are the same as you would use to copy files to tape, except you would specify `/vol/dev/aliases/floppy0` as the device instead of the tape device name. See “How to Copy All Files in a Directory to a Tape (cpio)” on page 803 for information on how to use `cpio`.

Copying Files with a Different Header Format

Archives created with the SunOS 5.x `cpio` command may not be compatible with older SunOS releases. The `cpio` command allows you to create archives that can be read with several other formats. You specify these formats using the `-H` option and one of these arguments:

- `crc` or `CRC` – ASCII header with checksum
- `ustar` or `USTAR` – IEEE / P1003 Data Interchange
- `tar` or `TAR` – `tar` header and format
- `odc` – ASCII header with small device numbers
- `bar` – `bar` header and format

The syntax for using the header options is:

```
cpio -o -H header-option < file-list > output-archive
```

▼ How to Create an Archive for Older SunOS Releases

Use the `cpio` command to create the archive.

```
$ cpio -oH odc < file-list > /dev/rmt/n
```

The `-H` options have the same meaning for input as they do for output. If the archive was created using the `-H` option, you must use the same option when the archive is read back in or the `cpio` command will fail, as shown below.

Example—Creating an Archive for Older SunOS Releases

```
$ find . -print | cpio -oH tar > /tmp/test
113 blocks
$ cpio -iH bar < /tmp/test
cpio: Invalid header "bar" specified
USAGE:
        cpio -i[bcdfkmrstuvBSV6] [-C size] [-E file] [-H hdr]
[-I file [-M msg]] [-R id] [patterns]
        cpio -o[acvABLV] [-C size] [-H hdr] [-O file [-M msg]]
        cpio -p[adlmuvLV] [-R id] directory
```

When you create an archive using different options, always write the command syntax on the media label along with the names of the files or file system on the archive.

If you do not know which `cpio` options were used when an archive was created, all you can do is experiment with different combinations of the options to see which ones allow the archive to be read.

See `cpio(1)`for a complete list of options.

Retrieving Files Created With the `bar` *Command*

To retrieve files from diskettes that were archived using the SunOS 4.x `bar` command, use the `-H bar` option to `cpio`.

Note – You can only use the `-H bar` option with `-i` to retrieve files. You cannot create files with the `bar` header option.

▼ How to Retrieve `bar` Files From a Diskette

1. **Change to the directory where you want to put the files.**

2. **Use the `cpio` command to retrieve `bar` files from a diskette.**
   All the files on the diskette are copied to the current directory.

```
$ cpio -ivH bar < /vol/dev/rdiskette/unlabeled
```

Managing Tape Drives 46

This chapter describes how to manage tape drives.

This is a list of the step-by-step instructions in this chapter.

46

Choosing Which Media to Use

You typically back up Solaris 2.5 systems using:

- 1/2-inch reel tape
- 1/4-inch streaming cartridge tape
- 8-mm cartridge tape
- 4-mm cartridge tape (DAT)

You can perform backups using diskettes, but this is time-consuming and cumbersome.

The media you choose depends on the availability of the equipment that supports it and of the media (usually tape) that you use to store the files. Although you must do the backup from a local system, you can write the files to a remote device.

Table 46-1 shows typical media used for backing up file systems and shows the length (or storage capacity) for each.

Table 46-1 Media Storage Capacities

Media	Capacity	Tape Length
1/2-inch tape	40–45 Mbytes	2300 feet
60-Mbyte 1/4-inch cartridge	60 Mbytes	425 feet
150-Mbyte 1/4-inch cartridge	150 Mbytes	700 feet
2.3-Gbyte 8-mm	2.3 Gbytes	6000 feet
5.0-Gbyte 8-mm	5.0 Gbytes	13000 feet
3.5-inch diskette	1422 blocks (1.44 Mbytes)	

Note – Capacity for 4-mm cartridge tapes depends on the type of drive and the data being written to the tape.

Backup Device Names

You specify a tape or diskette drive to use for backup by supplying a logical device name. This name points to the subdirectory containing the "raw" device file and includes the logical unit number of the drive. Tape drive naming conventions use a logical, not a physical, device name. Table 46-2 shows this naming scheme.

Table 46-2 Basic Device Names for Backup Devices

Device Type	**Name**
Tape	`/dev/rmt/`*n*
Diskette	`/vol/dev/rdiskette0/unlabeled`

The drive writes at its "preferred" density, which usually means the highest density it supports. Most SCSI drives can automatically sense the density or format on the tape and read it accordingly.

Tape drives fall into two categories based on controller type:

- Xylogics 472 for 1/2-inch rack-mounted (top-loaded) reel-to-reel drives (see Figure 46-1)
- SCSI for 1/4-inch cartridge, 1/2-inch front-loaded reel-to-reel, and 4-mm or 8-mm helical scan drives

Within the `/dev/rmt` subdirectory is a set of tape device files that support different output densities.

In general, you specify a tape drive device as shown in Figure 46-1.

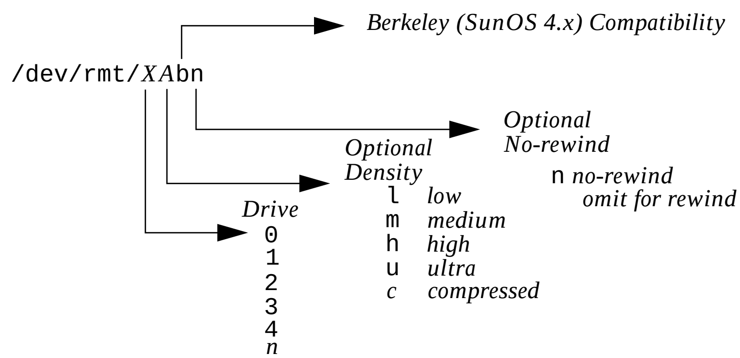


Figure 46-1 Tape Drive Device Names

You can have both SCSI and non-SCSI tape drives on the same system.

- A SCSI controller can have a maximum of seven SCSI tape drives.
- A non-SCSI controller can have a maximum of four tape drives.
- For each drive number (*X*), the density character depends on the controller and drive type as described in the following paragraphs.

Table 46-3 shows the device abbreviation for different tape controllers/units and media. Note that the first character in the device abbreviation for drive number does not have to be 0 as shown, but could be 1, 2, or 3, and so on, depending on how many tape drives are attached to the system.

Table 46-3 Device Abbreviations for Tape Controllers/Units and Media

Controller	Drive Unit	Size	Type	Format	Tracks	Device Abbreviation
Xylogics® 472	Fujitsu M2444	1/2-inch	Reel	1600 bpi	9	`/dev/rmt/0m`
		1/2-inch	Reel	6250 bpi	9	`/dev/rmt/0h`
SCSI front-loaded	HP	1/2-inch	Reel	800 bpi	9	`/dev/rmt/0m`
				6250 bpi	9	`/dev/rmt/0h`
SCSI	Sysgen®	1/4-inch	Cartridge	QIC-11	4	`/dev/rmt/0l`
				QIC-24	4	`/dev/rmt/0m`
				QIC-11	9	`/dev/rmt/0l`
				QIC-24	9	`/dev/rmt/0m`
	Emulex® MT-02	1/4-inch	Cartridge	QIC-11	4	`/dev/rmt/0l`
				QIC-24	4	`/dev/rmt/0m`
				QIC-11	9	`/dev/rmt/0l`
				QIC-24	9	`/dev/rmt/0m`
	Archive® QIC-150	1/4-inch	Cartridge	QIC-150	18	`/dev/rmt/0h`
	Wangtek™ QIC-150	1/4-inch	Cartridge	QIC-150	18	`/dev/rmt/0h`
	Desktop Backup Pack	1/4-inch	Cartridge	QIC-150	18	`/dev/rmt/0h`
	Exabyte® 8200 (2.3 GB)	8 mm	Cartridge	8 mm	Helical Scan	`/dev/rmt/0m`
	Exabyte 8500 (2.3 GB)	8 mm	Cartridge	8 mm	Helical Scan	`/dev/rmt/0l`
	Exabyte 8500 (5 GB)	8 mm	Cartridge	8 mm	Helical Scan	`/dev/rmt/0m`
	Archive® Python	4 mm	Cartridge	4 mm	Helical Scan	`/dev/rmt/0`

Rack-Mounted Non-SCSI 1/2-Inch Reel Drives

For 1/2-inch rack-mounted tape drives with either a Tapemaster or Xylogics 472 controller, substitute the density from Table 46-4 for the *A* variable in the device name (`/dev/rmt/`*XA*).

Table 46-4 Designating Density for Rack-Mounted 1/2-inch Tape Drives

Character	Density
`null`	Default “preferred” (highest) density (usually 6250 bpi uncompressed)
`l`	800 bpi
`m`	1600 bpi
`h`	6250 bpi
`u`	6250 bpi compressed

If you omit the density character, the tape is usually written at its highest density, not compressed.

SCSI 1/4-Inch Cartridge and 1/2-Inch Front-Loaded Reel Drives

For SCSI 1/4-inch cartridge and 1/2-inch front-loaded reel drives, substitute the density from Table 46-5 for the *A* variable in the device name (`/dev/rmt/`*XA*).

Table 46-5 Designating Format or Density for SCSI Tape Drives

Character	Density 1/4-Inch Cartridge	Density 1/2-Inch Front-Loaded Reel-to-Reel
`null`	Default preferred (highest) density	Default preferred (highest) density
`l`	QIC-11 format	800 bpi
`m`	QIC-24 format	1600 bpi
`h`	QIC-150	6250 bpi
`u`	Reserved	Reserved

For 1/4-inch cartridges, density is specified by the format in which the data is written: the QIC format. The QIC-11 and QIC-24 format write approximately 1000 bytes per inch on each track. The density for QIC-150 is somewhat higher. The “preferred” density for a 60-Mbyte 1/4-inch cartridge drive is QIC-24 and for a 150-Mbyte 1/4-inch cartridge drive is QIC-150.

An 18-track drive can write only QIC-150; it cannot be switched to write QIC-24 or QIC-11. Format selection is only useful for drives that can write both QIC-24 and QIC-11.

Specifying the Default Density for a Tape Drive

Normally, you specify a tape drive by its logical unit number, which may run from 0 to *n*. Table 46-6 describes how to specify tape device names using default density settings.

Table 46-6 Specifying Default Densities for a Tape Drive

To Specify The ...	Use ...
First drive, rewinding	`/dev/rmt/0`
First drive, nonrewinding	`/dev/rmt/0n`
Second drive, rewinding	`/dev/rmt/1`
Second drive, nonrewinding	`/dev/rmt/1n`

By default, the drive writes at its “preferred” density, which is usually the highest density it supports. If you do not specify a tape device, the command writes to drive number 0 at the default density the device supports.

Specifying Different Densities for a Tape Drive

To transport a tape to a system whose tape drive supports only a certain density, specify a device name that writes at the desired density. Table 46-7 describes how to specify different densities for a tape drive.

Table 46-7 Specifying Different Densities for a Tape Drive

To Specify The ...	Use ...
First drive, rewinding, low density	`/dev/rmt/0l`
First drive, nonrewinding, low density	`/dev/rmt/0ln`
Second drive, rewinding, medium density	`/dev/rmt/1m`
Second drive, nonrewinding, medium density	`/dev/rmt/1mn`

The unit and density characters are shown in Figure 46-1.

Determining Tape Drive Types

You can use the `status` option to the `mt` command to get status information about the Xylogics 472 1/2-inch tape drive and the Exabyte EXB-8200 8-mm tape drive.

The `mt` command also reports information about these 1/4-inch tape drives:

- Xylogics 472
- Exabyte EXB-8200
- Sysgen (QIC-24)
- Emulex MT-02 (QIC-24)
- Archive QIC-150
- Wangtek QIC-150

▼ How to Determine the Type of a Tape Drive

1. **Load a tape into the drive you want information about.**

2. **Display tape drive information with the `mt` command.**

```
# mt -f /dev/rmt/n status
```

3. **Repeat steps 1-2, substituting tape drive numbers 1, 2, 3, and so on to display information about all available tape drives.**

Example—Determining the Type of a Tape Drive

This example shows status for a QIC-150 tape drive (`/dev/rmt/0`) and an Exabyte tape drive (`/dev/rmt/1`).

```
$ mt -f /dev/rmt/0 status
Archive QIC-150 tape drive:
   sense key(0x0)= No Additional Sense   residual= 0   retries= 0
   file no= 0   block no= 0
$ mt -f /dev/rmt/1 status
Exabyte EXB-8200 8mm tape drive:
sense key(0x0)= NO Additional Sense residual= 0  retries= 0
file no= 0   block no= 0
```

Here is an quick way to poll a system and locate all tape drives. In this example, the tape drive is at 0.

```
$ for drive in 0 1 2 3 4 5 6 7
> do
> mt -f /dev/rmt/$drive status
> done
Archive QIC-150 tape drive:
   sense key(0x0)= No Additional Sense   residual= 0   retries= 0
   file no= 0   block no= 0
/dev/rmt/1: No such file or directory
/dev/rmt/2: No such file or directory
/dev/rmt/3: No such file or directory
/dev/rmt/4: No such file or directory
/dev/rmt/5: No such file or directory
/dev/rmt/6: No such file or directory
/dev/rmt/7: No such file or directory
$
```

▼ How to Show the Status of a Magnetic Tape Drive

To display tape drive status information, use the `mt` command.

```
$ mt -f /dev/rmt/n status
```

Status for the tape drive you specify is displayed.

Examples—Showing the Status of a Magnetic Tape Drive

In the following example, there is no tape in drive `/dev/rmt/1`.

```
$ mt -f /dev/rmt/1 status
/dev/rmt/1: no tape loaded or drive offline
$
```

In this example, the status is shown for the tape in drive `/dev/rmt/1`.

```
$ mt -f /dev/rmt/1 status
Archive QIC-150 tape drive:
   sense key(0x6)= unit attention   residual= 0   retries= 0
   file no= 0   block no= 0
$
```

Handling Magnetic Tape Cartridges

If errors occur when reading a tape, retension the tape, clean the tape drive, and then try again.

▼ How to Retension a Magnetic Tape Cartridge

Retension a magnetic tape cartridge with the `mt` command.

```
$ mt -f /dev/rmt/n retension
```

Example—How to Retension a Magnetic Tape Drive

In this example, the tape in drive `/dev/rmt/1` is retensioned.

```
$ mt -f /dev/rmt/1 retension
$
```

Note – Do not retension non-QIC tape drives.

▼ How to Rewind a Magnetic Tape Cartridge

To rewind a magnetic tape cartridge, use the `mt` command.

```
$ mt -f /dev/rmt/n rewind
```

Example—Rewinding a Magnetic Tape Cartridge

In this example, the tape in drive `/dev/rmt/1` is rewound.

```
$ mt -f /dev/rmt/1 rewind
$
```

Guidelines for Drive Maintenance and Media Handling

A backup tape that cannot be read is useless. It is a good idea to clean and check your tape drives periodically to ensure correct operation. See your hardware manuals for instructions on procedures for cleaning a tape drive. You can check your tape hardware by:

- Copying some files to the tape, reading them back, and then comparing the original with the copy.
- Or, you could use the `-v` option of the `ufsdump` command to verify the contents of the media with the source file system. The file system must be unmounted or completely idle for the `-v` option to be effective.

Be aware that hardware can fail in ways that the system does not report.

Always label your tapes after a backup. If you have planned a backup strategy similar to those suggested in Chapter 41, “Overview of Backing Up and Restoring File Systems,” you should indicate on the label “Tape A,” “Tape B,” and so forth. This label should never change. Every time you do a backup, make another tape label containing the backup date, the name of the machine and file system backed up, backup level, the tape number (1 of *n*, if it spans multiple volumes), plus any information specific to your site. Store your tapes in a dust-free safe location, away from magnetic equipment. Some sites store archived tapes in fireproof cabinets at remote locations.

You should create and maintain a log that tracks which media (tape volume) stores each job (backup) and the location of each backed-up file.

Index

C

D

E

K

L

M

N

T

U

W

X

Y

Z