System Administration Guide, Volume 1

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303-4900
U.S.A.

Part Number 805-7228–10
February 2000

Contents

About This Book

System Administration Guide, Volume I is part of a three-volume set that includes a significant part of the Solaris™ system administration information. It contains information for both SPARC™ based and IA based systems.

This book assumes that you have already installed the SunOS™ 5.8 operating system, and you have set up all networking software that you plan to use. The SunOS 5.8 operating system is part of the Solaris product family, which also includes many features, including the Solaris Common Desktop Environment (CDE). The SunOS 5.8 operating system is compliant with AT&T's System V, Release 4 operating system.

For the Solaris 8 release, new features interesting to system administrators are covered in sections called *What's New in ... ?* in the appropriate chapters.

Note - The Solaris operating environment runs on two types of hardware, or platforms—SPARC and IA. The Solaris operating environment runs on both 64–bit and 32–bit address spaces. The information in this document pertains to both platforms and address spaces unless called out in a special chapter, section, note, bullet, figure, table, example, or code example.

Who Should Use This Book

This book is intended for anyone responsible for administering one or more systems running the Solaris 8 release. To use this book, you should have 1-2 years of UNIX® system administration experience. Attending UNIX system administration training courses might be helpful.

How the System Administration Volumes Are Organized

Here is a list of the topics covered by the three volumes of the System Administration Guides.

System Administration Guide, Volume 1

- "Managing Users and Groups Topics" in *System Administration Guide, Volume 1*
- "Managing Server and Client Support Topics" in *System Administration Guide, Volume 1*
- "Shutting Down and Booting a System Topics" in *System Administration Guide, Volume 1*
- "Managing Removable Media Topics" in *System Administration Guide, Volume 1*
- "Managing Software Topics" in *System Administration Guide, Volume 1*
- "Managing Devices Topics" in *System Administration Guide, Volume 1*
- "Managing Disks Topics" in *System Administration Guide, Volume 1*
- "Managing File Systems Topics" in *System Administration Guide, Volume 1*
- "Backing Up and Restoring Data Topics" in *System Administration Guide, Volume 1*

System Administration Guide, Volume 2

- "Managing Printing Services Topics" in *System Administration Guide, Volume 2*
- "Working With Remote Systems Topics" in *System Administration Guide, Volume 2*
- "Managing Terminals and Modems Topics" in *System Administration Guide, Volume 2*
- "Managing System Security Topics" in *System Administration Guide, Volume 2*
- "Managing System Resources Topics" in *System Administration Guide, Volume 2*
- "Managing System Performance Topics" in *System Administration Guide, Volume 2*
- "Troubleshooting Solaris Software Topics" in *System Administration Guide, Volume 2*

System Administration Guide, Volume 3

- “Network Services Topics” in *System Administration Guide, Volume 3*
- “IP Address Management Topics” in *System Administration Guide, Volume 3*
- “Modem-Related Network Services Topics” in *System Administration Guide, Volume 3*
- “Accessing Remote File Systems Topics” in *System Administration Guide, Volume 3*
- “Mail Services Topics” in *System Administration Guide, Volume 3*
- “Monitoring Network Services Topics” in *System Administration Guide, Volume 3*

Ordering Sun Documents

Fatbrain.com, an Internet professional bookstore, stocks select product documentation from Sun Microsystems, Inc.

For a list of documents and how to order them, visit the Sun Documentation Center on Fatbrain.com at `http://www1.fatbrain.com/documentation/sun`.

Accessing Sun Documentation Online

The docs.sun.com℠ Web site enables you to access Sun technical documentation online. You can browse the docs.sun.com archive or search for a specific book title or subject. The URL is `http://docs.sun.com`.

What Typographic Conventions Mean

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

TABLE P–1 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. Do *not* save changes yet.

Shell Prompts in Command Examples

The following table shows the default system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

TABLE P–2 Shell Prompts

Shell	Prompt
C shell prompt	`machine_name%`
C shell superuser prompt	`machine_name#`
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.

- 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.
- The root path usually 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 paths in the examples.
- The examples in this book are for a basic SunOS 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 of the search path. Commands like `ps` or `df` are duplicated in `/usr/ucb` with different formats and options from the SunOS commands.

CHAPTER 1

Managing Users and Groups Topics

This section provides instructions for managing users and groups. This section contains these chapters.

Chapter 2	Provides overview information about setting up user accounts and groups in a network environment.
Chapter 3	Provides step-by-step instructions for setting up user accounts and groups with Admintool.

CHAPTER 2

Managing User Accounts and Groups (Overview)

This chapter provides guidelines and planning information for managing user accounts and groups. It also provides overview information about setting up user accounts and groups in a network environment. This chapter 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 concept information in this chapter.

For step-by-step instructions on managing user accounts and groups, see Chapter 3.

What's New in Managing Users and Groups?

In this Solaris release, role-based access control (RBAC) provides a flexible way to package superuser privileges for assignment to user accounts so that you don't have to give all superuser privileges to a user that needs to solve a specific problem.

See “Role-Based Access Control” in *System Administration Guide, Volume 2* for more information.

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:

Component	Description
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 include 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.

There is also a special type of user account called a *role*, which is used to give selected users special privileges. See “Role-Based Access Control” in *System Administration Guide, Volume 2* for more information.

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 might 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 might 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 might 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 2147483647, and they are required for both regular user accounts and special system accounts. The table below lists the UID numbers reserved for user accounts and system accounts.

TABLE 2–1 Reserved UID Numbers

User ID Numbers	Login Accounts	Reserved For ...
0 - 99	root, `daemon`, `bin`, `sys`, etc.	System accounts
100 - 2147483647	Regular users	General purpose accounts
60001	`nobody`	Unauthenticated users
60002	`noaccess`	Compatibility with Solaris 2.0 and compatible versions 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 might have been denied access to a printer—by being included in a printer deny list—but that attribute might 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.

Using Large User IDs and Group IDs

Previous Solaris software releases used 32-bit data types to contain the user IDs (UIDs) and group IDs (GIDs), but UIDs and GIDs were constrained to a maximum useful value of 60000. Starting with the Solaris 2.5.1 release and compatible versions, the limit on UID and GID values has been raised to the maximum value of a signed integer, or 2147483647.

UIDs and GIDs over 60000 do not have full functionality and are incompatible with many Solaris features, so avoid using UIDs or GIDs over 60000.

The table below describes interoperability issues with previous Solaris and Solaris product releases.

TABLE 2–2 Interoperability Issues for UIDs/GIDs Over 60000

Category	Product/Command	Issues/Cautions
NFS™ Interoperability	SunOS™ 4.0 NFS software and compatible versions	NFS server and client code truncates large UIDs and GIDs to 16 bits. This can create security problems if SunOS 4.0 and compatible machines are used in an environment where large UIDs and GIDs are being used. SunOS 4.0 and compatible systems require a patch.
Name Service Interoperability	NIS name service File-based name service	Users with UIDs above 60000 can log in or use the `su` command on systems running the Solaris 2.5 and compatible versions, but their UIDs and GIDs will be set to 60001 (nobody).
	NIS+ name service	Users with UIDs above 60000 are denied access on systems running Solaris 2.5 and compatible versions and the NIS+ name service.
Printed UIDs/GIDs	OpenWindows File Manager	Large UIDs and GIDs do not display correctly if the OpenWindows™ File Manager is used with the extended file listing display option.

TABLE 2–3 Large UID/GID Limitation Summary

A UID or GID Of ...	Limitations
60003 or greater	■ Users in this category logging into systems running Solaris 2.5 and compatible releases and the NIS or files name service get a UID and GID of `nobody`.
65535 or greater	■ Solaris 2.5 and compatible releases systems running the NFS version 2 software see UIDs in this category truncated to 16 bits, creating possible security problems. ■ Users in this category using the `cpio` command (using the default archive format) to copy file see an error message for each file and the UIDs and GIDs are set to `nobody` in the archive. ■ SPARC based systems: Users in this category running SunOS 4.0 and compatible applications see `EOVERFLOW` returns from some system calls, and their UIDs and GIDs are mapped to `nobody`. ■ IA based systems: Users in this category running SVR3-compatible applications will probably see `EOVERFLOW` return codes from system calls. ■ IA based systems: If users in this category attempt to create a file or directory on a mounted System V file system, the System V file system returns an `EOVERFLOW` error.
100000 or greater	■ The `ps -l` command displays a maximum five-digit UID so the printed column won't be aligned when they include a UID or GID larger than 99999.
262144 or greater	■ Users in this category using the `cpio` command (using `-H odc` format) or the `pax -x cpio` command to copy files see an error message returned for each file, and the UIDs and GIDs are set to `nobody` in the archive.
1000000 or greater	■ Users in this category using the `ar` command have their UIDs and GIDs set to `nobody` in the archive.
2097152 or greater	■ Users in this category using the `tar` command, the `cpio -H ustar` command, or the `pax -x tar` command have their UIDs and GIDs set to `nobody`.

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, numbers, or special characters. 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:

- Phrases (`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 become 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 *System Administration Guide, Volume 3*.

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.

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 56 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 moved from system to system.

Another way to customize user accounts is through role-based access control. See "Role-Based Access Control" in *System Administration Guide, Volume 2* for more information.

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. The two types of groups that a user can belong to are:

- Primary group – Specifies a group that the operating system assigns 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, might 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 Admintool software, but it doesn't matter if group 14 is his or her current primary group.

The `groups` command lists the groups that a user belongs to. A user can have only one primary group at a time. However, the user can temporarily change the user's primary group (with the `newgrp` command) to any other group in which the user 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

The table below lists the recommended tools for managing users and groups.

TABLE 2–4 Recommended Tools for Managing Users and Groups

If You Are Managing Users and Groups ...	The Recommended Tool Is ...	And You Will Need ...	To Start This Tool See ...
On remote and/or local systems in a networked, name service (NIS, NIS+) environment	AdminSuite™ 2.3's User and Group Manager (graphical user interface)	Graphics monitor running an X window environment such as CDE	*Solstice AdminSuite 2.3 Administration Guide*
On a local system	Admintool (graphical user interface)	Graphics monitor running an X window system such as CDE	Chapter 3

The Solaris commands `useradd` and `groupadd` also let you set up users and groups on a local system; however, the commands do not change name service maps or tables. The table below describes the Solaris command used to manage user accounts and groups if you are not using AdminSuite 2.3 or Admintool.

TABLE 2–5 Managing User Accounts and Groups by Using Solaris Commands

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`
Delete a User Account	NIS+	`nistbladm` `nisclient`
	NIS	`userdel` `make`
	None	`userdel`
Set Up User Account Defaults	NIS+	not available
	NIS	`useradd -D` `make`
	None	`useradd -D`
Disable a User Account	NIS+	`nistbladm`
	NIS	`passwd -r nis -l` `make`
	None	`passwd -r files -l`

TABLE 2–5 Managing User Accounts and Groups by Using Solaris Commands *(continued)*

Task	If You Use This Name Service ...	Then Use These Commands
Change a User's Password	NIS+	`passwd -r nisplus`
	NIS	`passwd -r nis`
	None	`passwd -r files`
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`
Modify Users in a Group	NIS+	`nistbladm`
	NIS	`groupmod` `make`
	None	`groupmod`
Delete a Group	NIS+	`nistbladm`

TABLE 2–5 Managing User Accounts and Groups by Using Solaris Commands *(continued)*

Task	If You Use This Name Service ...	Then Use These Commands
	NIS	`groupdel` `make`
	None	`groupdel`

What You Can Do With Admintool

Admintool is a graphical user interface that enables you to set up user accounts on a local system.

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. Use the following steps if two user accounts have duplicate user names or UID numbers:

- If two user accounts have duplicate UID numbers, use Admintool to remove one account and re-add it with a different UID number. You cannot use Admintool to modify a UID number of an existing user account.
- If two user account have duplicate user names, use Admintool to modify one of the accounts and change the user name.

If you do use Admintool to change a user name, the home directory's ownership is changed (if a home directory exists for the user).

One part of a user account that you can change is a user's group memberships. Admintool's Modify option lets you add or delete a user's secondary groups. Alternatively, you can use the Groups window 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

Delete User Accounts

When you delete a user account with Admintool, the software deletes the entries in the `passwd` and `group` files. In addition, you can delete the files in the user's home directory.

Add Customized User Initialization Files

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

You can customize the user initialization templates in the `/etc/skel` directory and then copy them to users' home directories.

Administer Passwords

You can use Admintool 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 might 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 Admintool 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.

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

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 is 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 `passwd` 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 Mechanoid:/export/home/kryten:/bin/csh
```

The table below describes the `passwd` file fields.

TABLE 2–6 Fields in the `passwd` File

Field Name	Description
username	Contains the user or login name. User names should be unique and consist of 1-8 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. 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.
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 2–13 contains a description of shell features.

Default `passwd` File

The default Solaris `passwd` file contains entries for standard daemons, processes usually started at boot time to perform some system-wide task, such as printing, network administration, and port monitoring.

```
root:x:0:1:Super-User:/:/sbin/sh
daemon:x:1:1::/:
bin:x:2:2::/usr/bin:
sys:x:3:3::/:
```

(continued)

(Continuation)

```
adm:x:4:4:Admin:/var/adm:
lp:x:71:8:Line Printer Admin:/usr/spool/lp:
uucp:x:5:5:uucp Admin:/usr/lib/uucp:
nuucp:x:9:9:uucp Admin:/var/spool/uucppublic:/usr/lib/uucp/uucico
listen:x:37:4:Network Admin:/usr/net/nls:
nobody:x:60001:60001:Nobody:/:
noaccess:x:60002:60002:No Access User:/:
nobody4:x:65534:65534:SunOS 4.x Nobody:/:
```

TABLE 2–7 Default `passwd` File Entries

User Name	User ID	Description
`root`	`0`	Superuser account
`daemon`	`1`	Umbrella system daemon associated with updating calendars, and so forth
`bin`	`2`	Administrative daemon associated with running system binaries to perform some routine system task
`sys`	`3`	Administrative daemon associated with system logging or updating files in temporary directories
`adm`	`4`	Administrative daemon associated with system logging
	`71`	Line printer daemon
`uucp`	`5`	uucp daemon
`nuucp`	`6`	uucp daemon
`listen`	`37`	Network listener daemon
`nobody`	`60001`	Anonymous user account, assigned by an NFS server when a request is received from an unauthorized root user. The nobody user account is assigned to software processes that do not need nor should have any special permissions.

TABLE 2–7 Default passwd File Entries *(continued)*

User Name	User ID	Description
noaccess	60002	Account assigned to a user or a process that needs access to a system through some application but without actually logging in.
nobody4	65534	SunOS 4.0 or 4.1 version of nobody user account.

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
```

The table below describes the shadow file fields.

TABLE 2–8 Fields in the shadow File

Field Name	Description
username	Contains the user or login name.
password	Might 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.

TABLE 2–8 Fields in the shadow File *(continued)*

Field Name	Description
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
```

The table below describes the group file fields.

TABLE 2–9 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 might be called chem. Group names can have a maximum of eight characters.
group-password	Usually contains an asterisk 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.

TABLE 2–9 Fields in the `group` File *(continued)*

Field Name	Description
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 comma-separated list of user names, representing the user's secondary group memberships. Each user can belong to a maximum of 16 secondary groups.

Default `group` file

The default Solaris `group` file contains the following system groups that support some system-wide task, such as printing, network administration, and electronic mail. Many of these having corresponding entries in the `passwd` file.

```
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:root
nobody::60001:
noaccess::60002:
nogroup::65534:
```

TABLE 2–10 Default `group` File Entries

Group Name	Group ID	Description
`root`	`0`	Superuser group
`other`	`1`	

TABLE 2–10 Default `group` File Entries *(continued)*

Group Name	Group ID	Description
`bin`	`2`	Administrative group associated with running system binaries
`sys`	`3`	Administrative group associated with system logging or temporary directories
`adm`	`4`	Administrative group associated with system logging
`uucp`	`5`	Group associated with uucp functions
`mail`	`6`	Electronic mail group
`tty`	`7`	Group associated with tty devices
	`8`	Line printer group
`nuucp`	`9`	Group associated with uucp functions
`staff`	`10`	General administrative group
`daemon`	`12`	Daemon group
`sysadmin`	`14`	Administratve group associated with Admintool and Solstice AdminSuite tools
`nobody`	`60001`	Anonymous group assigned by an NFS server when a request is received from an unauthorized root user
`noaccess`	`60002`	
`nogroup`	`65534`	

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 file 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 the table below.

TABLE 2–11 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 environment provides default user initialization files for each shell in the `/etc/skel` directory on each system, as shown in the table below.

TABLE 2–12 Default User Initialization Files

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

TABLE 2–12 Default User Initialization Files *(continued)*

You can use these files as a starting point and modify them to create a standard set of files that 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 72 for step-by-step instructions on how to create sets of user initialization files for different types of users.

When you use Admintool to create a new user account and select the create home directory option, the following files are created, depending on which login shell is selected:

Shell	Files Created
C	The `/etc/skel/local.cshrc` and the `/etc/skel/local.login` files are copied into the user's home directory and are renamed `.cshrc` and `.login`.
Bourne and Korn	The `/etc/skel/local.profile` file is copied into the user's home directory and renamed `.profile`.

If you use the `useradd` command to add a new user account and specify the `/etc/skel` directory by using the `–k` and `–m` options, all three `/etc/skel/local*` and `/etc/skel/.profile` files are copied into the user's home directory. At this point, you'll need to rename them to whatever is appropriate for the user's login shell.

Using Site Initialization Files

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 customize the user's 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

The table below 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 2–13 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

TABLE 2–13 Basic Features of Bourne, C, and Korn Shells *(continued)*

Feature	Bourne	C	Korn
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
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. The table below shows how to set environment variables in a user initialization file.

TABLE 2–14 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`

The table below describes environment and shell variables you might 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 2–15 Shell and Environment Variable Descriptions

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). No built-in behavior of the shell depends on this variable. It's only a useful variable for branching within shell scripts.
`CALENDAR`	Sets the path to the Calendar executables.

TABLE 2–15 Shell and Environment Variable Descriptions *(continued)*

Variable	Description
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 first looks 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.
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 set automatically 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.

TABLE 2–15 Shell and Environment Variable Descriptions *(continued)*

Variable	Description
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.
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.

Setting Path 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 might want to avoid, especially for superuser.
- 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
```

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.

The table below describes some of the values for the LANG and LC environment variables.

TABLE 2–16 Values for LANG and LC Variables

Value	Locale
de	German
fr	French
iso_8859_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 is 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 the table below, which shows the file and directory permissions that are created for each of the octal values of `umask`.

TABLE 2–17 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 need to change for your particular site.

Example—`.profile` File

```
PATH=$PATH:$HOME/bin:/usr/local/bin:/usr/ccs/bin:. 1
MAIL=/var/mail/$LOGNAME 2
NNTPSERVER=server1 3
MANPATH=/usr/share/man:/usr/local/man 4
PRINTER=printer1 5
umask 022 6
export PATH MAIL NNTPSERVER MANPATH PRINTER 7
```

1. Defines the user's shell search path.
2. Defines the path to the user's mail file.
3. Defines the user's Usenet news server.
4. Defines the user's search path for man pages.
5. Defines the user's default printer.
6. Sets the user's default file creation permissions.
7. Sets the listed environment variables.

Example—`.cshrc` File

```
set path=($PATH $HOME/bin /usr/local/bin /usr/ccs/bin) 1
setenv MAIL /var/mail/$LOGNAME 2
setenv NNTPSERVER server1 3
setenv PRINTER printer1 4
alias h history 5
umask 022 6
source /net/server2/site-init-files/site.login 7
```

1. Defines the user's shell search path.
2. Defines the path to the user's mail file.

3. Defines the user's Usenet news server.
4. Defines the user's default printer.
5. Creates an alias for the `history` command (the user will need to type only `h` to run the `history` command).
6. Sets the user's default file creation permissions.
7. Sets the listed environment variables.

Example—Site Initialization File

The following 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
```

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 is located on a server named `server2`. This line also assumes that the automounter is running on the user's system.

CHAPTER 3

Setting Up and Maintaining User Accounts and Groups (Tasks)

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.

For overview information about Managing User Accounts and Groups, see Chapter 2.

Becoming Superuser (root)

Most administrative tasks such as adding users require that you log in as root (UID=0) first. The root account is also known as the *superuser* account because it's used to make system changes and can override user file protection in emergency situations.

The superuser account should be used only to perform administrative tasks to prevent indiscriminate changes to the system.

You can either log into the system as superuser or use the su(1M) command to change to the superuser account.

If you are using role-based access control, you must assume a role (either superuser or some other role) to perform administrative tasks. Roles are assumed by using the su command; you cannot log in to a role directly. See "Role-Based Access Control" in *System Administration Guide, Volume 2* for more information.

▼ How to Become Superuser (root)

Become superuser by one of the following methods. Both methods require that you know the root password.

- Change to the superuser account by using the su command.

```
% su
Password: root_password
#
```

The pound sign (#) is the Bourne shell prompt for the superuser account.

- Log in as superuser on the system console.

```
hostname console: root
Password: root_password
#
```

This method is not enabled by default. You must modify the /etc/default/login file to log in as superuser on the system console. See "Securing Systems (Tasks)" in *System Administration Guide, Volume 2* for information on modifying this file.

Setting Up User Accounts Task Map

TABLE 3–1 Setting Up User Accounts Task Map

Task	Description	For Instructions, Go To
1. Customize User Initialization Files	*Optional.* Set up user initialization files (`.cshrc`, `.profile`, `.login`), so you can provide new users with consistent environments.	"How to Customize User Initialization Files" on page 72
2. Add a Group	*Optional.* To help administer users, add groups by using the Groups main window.	"How to Add a Group" on page 75
3. Add a User Account	Add a user account by using Admintool's Users main window.	"How to Add a New User Account" on page 76
4. Share the User's Home Directory	Share the user's home directory, so the directory can be remotely mounted from the user's system.	"How to Share a User's Home Directory" on page 77
5. Mount the User's Home Directory	Manually mount the user's home directory on the user's system by using the `mount` command.	"How to Mount a User's Home Directory" on page 79

User Information Data Sheet

You might find it useful to create a form like the one below to gather information about users before adding their accounts.

If you are using role-based access control, you will also need to list any roles, profiles, or authorizations intended for the user account. See "Role-Based Access Control" in *System Administration Guide, Volume 2* for more information.

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 superuser on the system where the users' home directories are created and shared.**

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

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

shared-dir	The name of a directory that is available to other systems on the network.
user-type	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-dir/skel/user-type/.cshrc
# cp /etc/skel/local.login /shared-dir/skel/user-type/.login
# cp /etc/skel/local.profile /shared-dir/skel/user-type/.profile
```

Note - If the account has profiles assigned to it, then the user has to launch a special version of the shell called a profile shell to use commands (with any security attributes) that are assigned to the profile. There are three profile shells corresponding to the types of shells: `pfsh` (Bourne shell), `pfcsh` (C shell), and `pfksh` (Korn shell).

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 56 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-dir/skel/user-type/.*
```

6. **Verify the permissions for the user initialization files are correct with the `ls -la` command.**

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 66)
# chmod 744 /export/skel/enduser/.*
```

▼ How to Start Admintool

1. **Verify that the following prerequisites are met. To use Admintool, you must:**
   - Have a bit-mapped display monitor. The Admintool software can be used only on a system with a console that has a bit-mapped screen such as a standard display monitor that comes with a Sun workstation.
   - Be running an X Window environment such as CDE.
   - Be a member of the sysadmin group (group 14).

   If you want to perform administration tasks on a system with an ASCII terminal as the console, use Solaris commands instead. See `useradd`(1M) for more information.

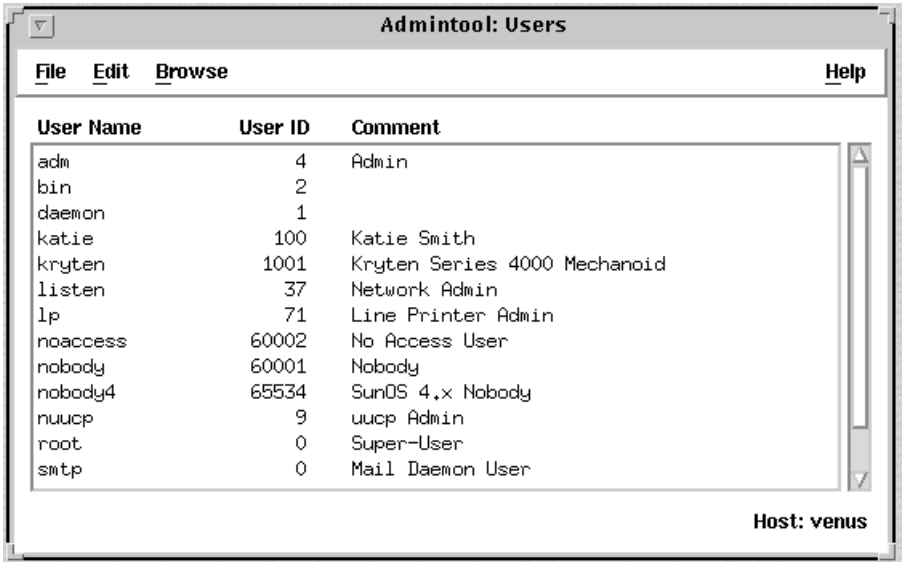
2. **Start Admintool.**

```
$ admintool &
```

   The Users main window appears.

Example—Starting Admintool

The Users main window enables you to manage user account information.



▼ How to Add a Group

1. **Start Admintool, if it's not already running.**

   See "How to Start Admintool" on page 74 for more information on starting Admintool.

2. **Choose Groups from the Browse menu.**

   The Groups window appears.

3. **Select Add from the Edit menu.**

   The Add window has several fields. If you need information to complete a field, click the Help button to see field definitions for this window.

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

5. **Type the group ID for the new group in the Group ID text box.**

   The group ID should be unique.

6. **(Optional) Type user names in the Members List text box.**

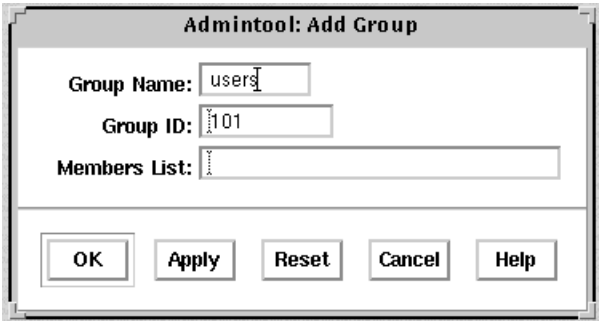
   The list of users will be added to the group. User names must be separated by commas.

7. **Click OK.**

   The list of groups displayed in the Groups window is updated to include the new group.

Example—Adding a Group

The following example adds a group named `users` that has a group ID of 101.



▼ How to Add a New User Account

1. **(Optional) Fill out the user information data sheet on "User Information Data Sheet" on page 71.**

2. **Start Admintool, if it's not already running.**

   See "How to Start Admintool" on page 74 for more information.

3. **Choose Add from the Edit menu.**

   The Add User window is displayed.

4. **Fill in the Add User window.**

   If you need information to complete a field, click the Help button to see field definitions for this window.

5. **Click OK.**

   The list of user accounts displayed in the Users main window is updated to include the new user account.

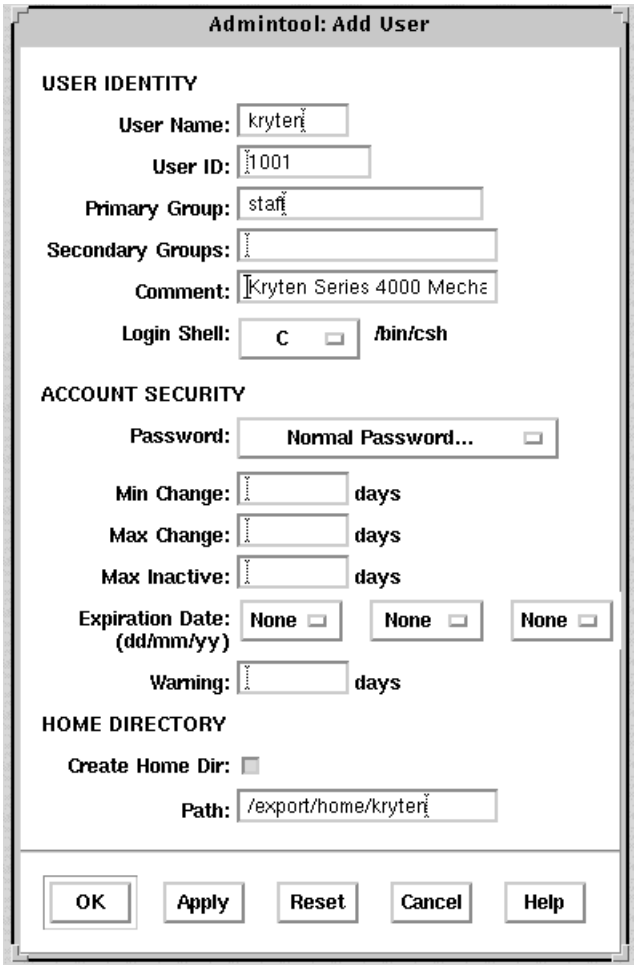
Where to Go From Here

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 77 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 Quotas (Tasks)" in *System Administration Guide, Volume 2* for information on setting disk quotas.

Example—Adding a New User Account

The following example adds the user `kryten` to the system.



▼ How to Share a User's Home Directory

1. **Become superuser on the system that contains the home directory.**

2. **Verify that the `mountd` daemon is running.**

```
# ps -ef | grep mountd
root   176     1  0   May 02 ?        0:19 /usr/lib/nfs/mountd
```

The `/usr/lib/nfs/mountd` line shows whether the `mountd` daemon is running.

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 the verification step below.
Not shared	Go to Step 6 on page 78

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

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

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.

8. **Verify that a user's home directory is shared, as follows:**

```
# share
```

Where to Go From Here

If 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 79 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
# share
-               /usr/dist               ro   ""
-               /export/home/user-name  rw   ""
```

▼ 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 77 for more information.**

2. **Log in as superuser 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
```

system-name	The name of the system where the home directory is located.
`/export/home/`*user-name*	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.
`-`	Required placeholders in the entry.
`/export/home/`*user-name*	The name of the directory where the user's home directory will be mounted.

See Chapter 36 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.

6. **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 is added.)
# mkdir -p /export/home/ripley
# mountall
# mount
/ on /dev/dsk/c0t0d0s0 read/write/setuid/intr/largefiles/onerror=panic on Fri ...
/usr on /dev/dsk/c0t0d0s6 read/write/setuid/intr/largefiles/onerror=panic on Fri ...
/proc on /proc read/write/setuid on Fri Sep 10 16:09:48 1999
/dev/fd on fd read/write/setuid on Fri Sep 10 16:09:51 1999
/etc/mnttab on mnttab read/write/setuid on Fri Sep 10 16:10:06 1999
/var/run on swap read/write/setuid on Fri Sep 10 16:10:06 1999
/tmp on swap read/write/setuid on Fri Sep 10 16:10:09 1999
/export/home/ripley on venus:/export/home/ripley /read/write/remote on ...
```

Maintaining User Accounts Task Map

TABLE 3–2 Task Map: Maintaining User Accounts

Task	Description	For Instructions, Go To
Modify a Group	Modify a group's name or the users in a group by choosing Modify from the Edit menu in the Groups window.	"How to Modify a Group" on page 81
Delete a Group	Delete a group by choosing Delete from the Edit menu in the Groups window.	"How to Delete a Group" on page 82
Modify a User Account	*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. *Change a User's Password* If you want to change a user's password, use the Password menu in the Modify window. *Change Password Aging* If you want to force users to change their passwords periodically, change the Password Aging fields in the Modify window (Account Security category).	"How to Disable a User Account" on page 84 "How to Change a User's Password" on page 85 "How to Change Password Aging for a User Account" on page 86
Delete a User Account	Delete a user account by choosing Delete from in the Edit menu in the Users window.	"How to Delete a User Account" on page 88

▼ How to Modify a Group

1. **Start Admintool, if it is not already running. Select Groups from the Browse menu.**

   See "How to Start Admintool" on page 74 for more information.

2. **Select the group entry you want to modify from the Groups window.**

3. **Choose Modify from the Edit menu.**

   The Modify Group window contains the selected group entry.

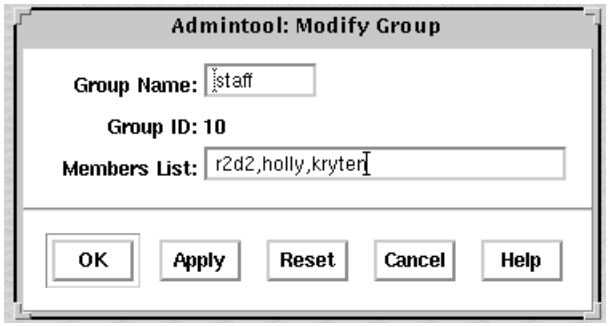
4. **Modify either the group's name or the users in the group.**

User names must be separated by commas. If you need information to complete a field, click the Help button to see field definitions for this window.

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

Example—Modifying a Group

The following example adds the users `r2d2`, `holly`, and `kryten` to the `staff` group.



▼ How to Delete a Group

1. **Start Admintool, if it's not already running. Select Groups from the Browse menu.**
   See "How to Start Admintool" on page 74 for more information.

2. **Select the group entry you want to delete from Groups window.**

3. **Choose Delete from the Edit menu.**
   A window asks you to confirm the deletion.

4. **Click OK.**
   The group entry is deleted from the Groups window.

▼ How to Modify a User Account

1. **Start Admintool, if it's not already running. Select Users from the Browse menu.**
   See "How to Start Admintool" on page 74 for more information.

2. **Select the user account entry to modify from the Users window.**

3. **Choose Modify from the Edit menu.**

   The Modify window contains the selected user account entry.

4. **Modify the user account.**

   If you need information to complete a field, click 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:

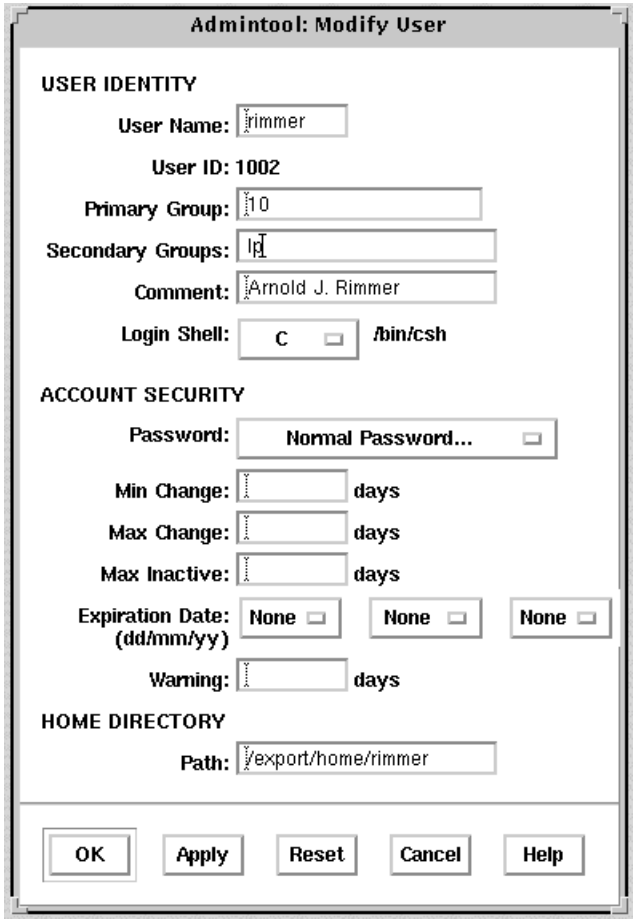
   - "How to Disable a User Account" on page 84
   - "How to Change a User's Password" on page 85
   - "How to Change Password Aging for a User Account" on page 86

5. **Click OK.**

6. **To verify that the modifications were made, double-click the modified user account entry in the Users window, then click Cancel to close the window without making any modifications.**

Example—Modifying a User Account

The following example adds the secondary group membership `lp` 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 Admintool, if it's not already running. Select Users from the Browse menu, if necessary.**

   See "How to Start Admintool" on page 74 for more information.

2. **Select the user account entry to be disabled.**

3. **Choose Modify from the Edit menu.**

   The Modify Users window contains the selected user account entry.

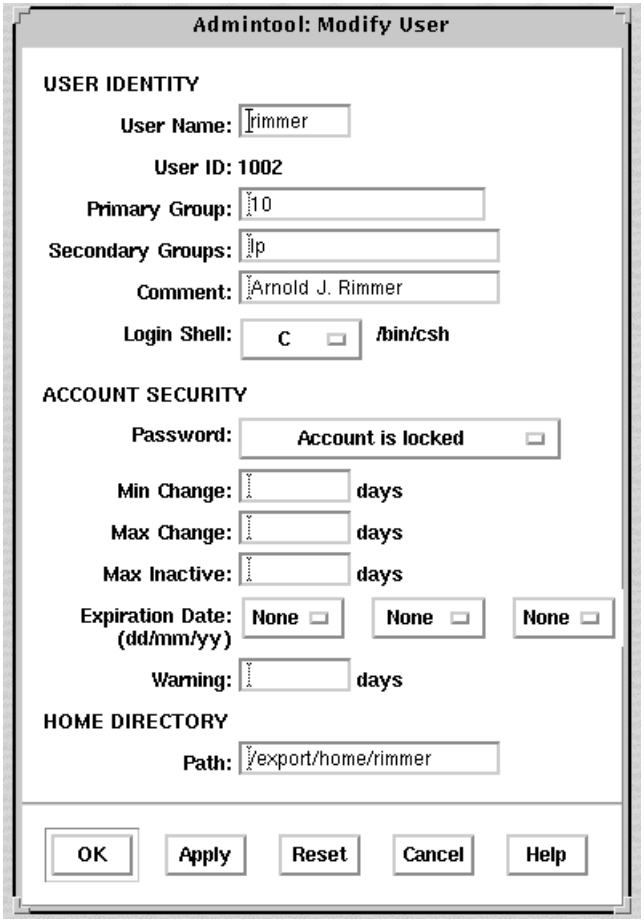
4. **Choose Account Is Locked from the Password menu.**
   This selects the locked password status, which disables the user account.

5. **Click OK.**

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

Example—Disabling a User Account

The following example disables the `rimmer` user account.



▼ How to Change a User's Password

1. **Start Admintool, if it's not already running. Select Users from the Browse menu.**

See “How to Start Admintool” on page 74 for more information.

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

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

4. **Choose Normal Password from the Password menu.**

5. **Click OK.**

Example—Changing a User’s Password

This is the pop-up window used to change user’s passwords which is available from the Add User or Modify User windows.

▼ How to Change Password Aging for a User Account

1. **Start Admintool, if it is not already running. Select Users from the Browse menu.**
   See “How to Start Admintool” on page 74 for more information.

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

3. **Choose Modify from the Edit menu.**
   The Modify window contains the selected user account entry.

4. **Change the following fields that affect password aging:**

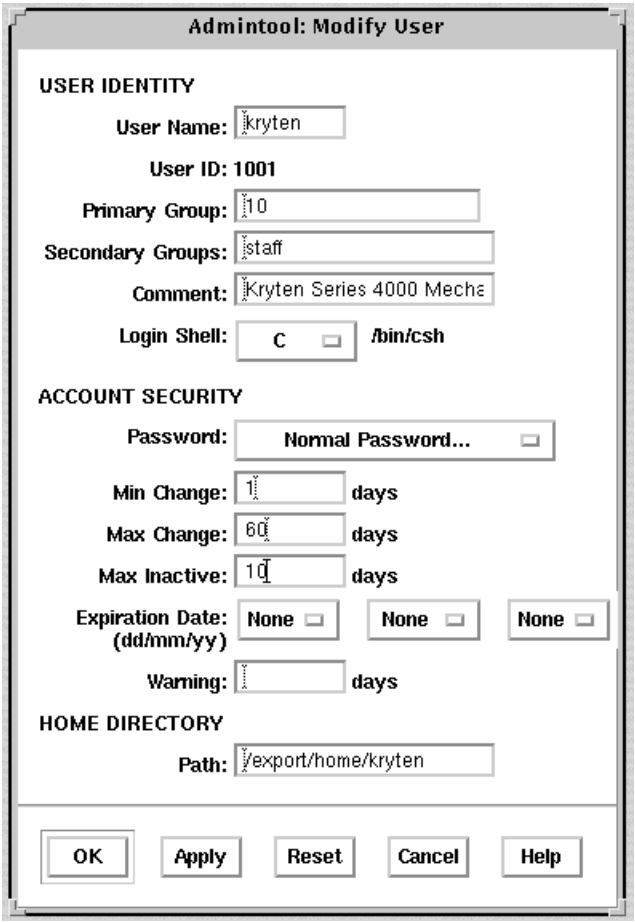
- Min Change
- Max Change
- Max Inactive
- Expiration Date
- Warning

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

5. **Click OK.**

Example—Changing Password Aging for a User Account

In the following example, the user must keep a new password for at least one day (`Min Change`) , and must change the password every 60 days (`Max Change`). The user must change the password if the account is inactive for more than 10 days (`Max Inactive`).



▼ How to Delete a User Account

1. **Start Admintool, if it's not already running. Select Users from the Browse menu, if necessary.**

   See "How to Start Admintool" on page 74 for more information.

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

3. **Choose Delete from the Edit menu.**

   The Delete window is displayed to confirm the removal of the user account.

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

5. **Click OK when you are ready to delete the user account. The user account entry is deleted from the Users main window.**

Example—Deleting a User Account

The account for user `kryten` and the `/export/home/kryten` directory is removed.

Solaris User Registration

Solaris User Registration is a tool for getting information about new Solaris releases, upgrade offers, and promotions. This graphical user interface (GUI) starts automatically when you first log into your desktop. The GUI lets you register now, later, or never. The registration process also provides Sun with the user's Solaris version, survey type, platform, hardware, and locale.

Accessing Solaris Solve℠

Completing the Solaris User Registration process provides access to Solaris Solve, an exclusive web site that offers valuable Solaris product information and solutions—all in one convenient location. It provides a quick and easy method for getting the most recent information on what's happening around the latest Solaris release. Solaris Solve also provides a preview to additional Sun contract and service opportunities.

Basically, the steps for completing Solaris User Registration and accessing Solaris Solve are:

1. Fill in the electronic Solaris User Registration profile.
2. Submit the profile by email or print the profile to fax or mail.
3. Create your login ID and password to access the Solaris Solve site.

   Even if you do not access the Solaris Solve site immediately, we recommend that you create your Solaris Solve login ID and password during the Solaris User

Registration process. A Solaris Solve login ID and password should contain six to eight alphanumeric characters without spaces and colons.

4. Access the Solaris Solve site.

Note - Solaris User Registration is not invoked if the system administrator or user is logged in as superuser.

If you choose to register, a copy of the completed form is stored in `$HOME/.solregis/uprops`. If you choose to never register and change your mind later, you can start User Registration by:

- Typing `/usr/dt/bin/solregis` at any command line prompt
- Clicking the Registration icon in the Application Manager's desktop tools folder (Common Desktop Environment desktop only)

See `solregis`(1) for more information.

Troubleshooting Solaris User Registration Problems

This section provides troubleshooting tips for solving Solaris User Registration problems.

The following table describes problems that might occur when you try to register, and actions required to resolve these conflicts.

TABLE 3–3 Registration Problem Descriptions and Suggested Resolutions

Problem Description	How to Resolve the Problem
The registration form failed to initialize: Web page window displays and requests that user see system administrator to resolve problem preventing registration setup.	Check for missing registration files.
The form could not be emailed: Dialog box displays and requests that user see system administrator to resolve problem.	Check to see if email is configured correctly. Also check if CDE is on user's system since it must be present to email completed registration form. Alternatively, users can print the form and fax or mail it.
The form could not be printed: Dialog box displays and requests that user see system administrator to resolve problem.	Check to see if the printer is configured correctly. Alternatively, the user can email form.

TABLE 3–3 Registration Problem Descriptions and Suggested Resolutions *(continued)*

Problem Description	How to Resolve the Problem
The form could not be saved: Dialog box displays and verifies that registration succeeded; however, the registration information cannot be recalled when updating registration in the future.	Check the user's home directory. Required action depends on the system's configuration.
You forgot your Solaris Solve login ID and password.	Send a mail message describing the problem to `SolarisSolve@sun.com` or see "How to Restart Solaris User Registration" on page 91
You want to restart the registration process.	"How to Restart Solaris User Registration" on page 91

▼ How to Restart Solaris User Registration

Use the following procedure to restart the Solaris User Registration process.

1. **Change to the** `$HOME/.solregis` **directory.**

```
% cd $HOME/.solregis
```

2. **Remove the** `uprops` **file.**

```
% rm uprops
```

3. **Restart the registration process.**

```
% /usr/dt/bin/solregis &
```

▼ How to Disable User Registration

The table below shows how to disable User Registration before and after installing Solaris software. Before disabling Solaris User Registration, system administrators should register for their organization.

TABLE 3–4 Ways to Disable User Registration

To Disable User Registration ...	You Can ...	For More Information See ...
Before Solaris software is installed	■ Deselect the `SUNWsregu` package (interactive installation) ■ Modify a custom JumpStart profile to not install the `SUNWsregu` package ■ Create and run a finish script that creates a file named `solregis` in the `/etc/default` directory on one or more systems with the following line in it: `DISABLE=1`	*Solaris 8 Advanced Installation Guide* `solregis`(1)
After Solaris software is installed	■ Use the `pkgrm` command to remove the `SUNWsregu` package ■ Add the `solregis` file in the `/etc/default` directory (custom JumpStart installation only)	Chapter 21 *Solaris 8 Advanced Installation Guide* `solregis`(1)

CHAPTER 4

Managing Server and Client Support Topics

This section provides a conceptual overview for managing server and client support in the Solaris environment. This section contains this chapter.

Chapter 5	Provides a high-level overview about managing server and client support on a network. This chapter describes the different system types for which you can add support, and guidelines for choosing a system type to use.

CHAPTER 5

Managing Server and Client Support (Overview)

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.

Note - AutoClient™ and diskless systems are not supported in the Solaris 8 release. See *Solstice AdminSuite 2.3 Administration Guide* for information on managing existing client systems.

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.

  Other clients might 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 `/export/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 other clients mount the file systems remotely, relying on servers to provide these services. The table below lists these and other differences for each system type.

TABLE 5–1 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* –	high	high
Standalone System	root (/) /usr /export/home	Yes	– *none* –	low	high
JavaStation™	– *none* –	No	/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:

- 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 `/export/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 it is not connected to a network.

JavaStation Client

The JavaStation™ is a client designed for zero administration. This client optimizes Java™; the JavaStation client takes full advantage of the network to deliver everything from Java applications and services to complete, integrated system and network management. The JavaStation has no local administration; booting, administration, and data storage are handled by servers.

CHAPTER 6

Shutting Down and Booting a System Topics

This section provides instructions for shutting down and booting systems running the Solaris release. This section contains these chapters.

Chapter 7	Provides an overview and guidelines for shutting down and booting a system.
Chapter 8	Provides information about run levels and boot files.
Chapter 9	Provides step-by-step instructions for shutting down a system.
Chapter 10	Provides step-by-step instructions for booting a SPARC based system.
Chapter 11	Provides step-by-step instructions for booting an IA based system.
Chapter 12	Provides a high-level overview of the boot process, including a description of the platform-specific hardware used to boot SPARC based and IA based systems.

CHAPTER 7

Shutting Down and Booting a System (Overview)

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

This is a list of overview information in this chapter.

What's New in Shutting Down and Booting a System?

This section describes new features related to shutting down and booting a system in this Solaris release.

Booting a System Over the Network With DHCP

Dynamic Host Configuration Protocol (DHCP) functionality has been added to boot a system over the network in this Solaris release. The previous network boot technology based on `RARP/bootparams` is still available.

A DHCP server must have been installed and configured in your network before you can use DHCP booting. For information on setting up a DHCP server, see "Configuring DHCP Service" in *System Administration Guide, Volume 3*.

For information on booting a SPARC based system over the network, see "SPARC: How to Boot a System Over the Network" on page 142. For information on booting an IA based system over the network, see "IA: How to Boot a System Over the Network" on page 156.

IA: Booting From CD-ROM Without the Solaris Boot Diskette

You can boot the Solaris 8 (Intel Platform Edition) directly from the locally attached CD-ROM without the Solaris boot diskette on IA based systems that support this feature.

The Solaris boot diskette is still available for the systems that do not support this feature.

See *Solaris 8 (Intel Platform Edition) Installation Guide* for information on booting IA based systems with or without the boot diskette.

Where to Find Shutting Down and Booting Tasks

Use these references to find step-by-step instructions for shutting down and booting a system.

For Information On ...	See ...
Shutting down a SPARC based or IA based system	Chapter 9
Booting a SPARC based system	Chapter 10

For Information On ...	See ...
Booting an IA based system	Chapter 11
Managing a SPARC based system with the power management software	*Using Power Management*

Shutting Down and Booting Terminology

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

- Run levels and init states – A *run level* is a letter or digit representing a system state in which a particular set of system services are available. The system is always running in one of a set of well-defined run levels. Run levels are also referred to as *init states* because the `init` process is used to perform transitions between run levels. System administrators use the `init`(1M) command to initiate a run-level transition. This book refers to init states as run levels.
- Boot types – A *boot type* describes how a system is booted. 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 and 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 superuser 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 log in. 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 – Stops the operating system and reboots to the state defined by the `initdefault` entry in the `/etc/inittab` file.
  - 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 8.

Guidelines for Booting a System

Keep the following in mind when booting a system:

- After a system is shut down, it is booted by using the `boot` command at the PROM level on a SPARC based system or by using the `boot` command at the Primary Boot Subsystem Menu on an Intel system.
- 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 based and IA based systems use different hardware components for booting. These differences are described in Chapter 12.

Performing a Reconfiguration Boot

Perform a reconfiguration boot when adding new hardware to the system. See the table below to determine which reconfiguration procedure to use.

TABLE 7–1 Reconfiguration Procedures

If You Are Reconfiguring The System To ...	See ...
Add a secondary disk	Chapter 30 or Chapter 31
Add some other peripheral device	"How to Add a Peripheral Device" on page 290

When to Shut Down a System

The following table provides a list of system administration tasks and the type of shut down needed to initiate the task.

TABLE 7–2 Shutting Down a System

If You Are ...	Change To This Run Level ...	See ...
Turning off system power due to anticipated power outage	Run level 0, where it is safe to turn off power	Chapter 9
Changing kernel parameters in the `/etc/system` file	Run level 6 (reboot the system)	Chapter 9
Performing file system maintenance, such as backing up or restoring system data	Run level S (single-user mode)	Chapter 9
Repairing a system configuration file such as `/etc/system`	See "When to Boot a System" on page 106	N/A
Adding or removing hardware from the system	Reconfiguration boot (plus turning off power when adding or removing hardware)	Chapter 24
Repairing an important system file which is causing system boot failure	See "When to Boot a System" on page 106	N/A

TABLE 7–2 Shutting Down a System *(continued)*

If You Are ...	Change To This Run Level ...	See ...
Booting the kernel debugger (kadb) to track down a system problem	Run level 0, if possible	Chapter 9
Recovering from a hung system and you want to force a crash dump	See "When to Boot a System" on page 106	N/A

See Chapter 9 for examples of shutting down a server or standalone system.

When to Boot a System

The table below provides a list of system administration tasks and the corresponding boot type used to complete the task.

TABLE 7–3 Booting a System

If You Are Rebooting the System After ...	Use This Boot Type ...	See SPARC Procedure ...	See IA Procedure ...
Turning off system power due to anticipated power outage	Turn system power back on	Chapter 9	Chapter 9
Changing kernel parameters in the /etc/system file	Reboot the system to run level 3 (multiuser mode with NFS resources shared)	"SPARC: How to Boot a System to Run Level 3 (Multiuser State)" on page 138	"IA: How to Boot a System to Run Level 3 (Multiuser State)" on page 151
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	"SPARC: How to Boot a System to Run Level S (Single-User State)" on page 139	"IA: How to Boot a System to Run Level S (Single-User State)" on page 152

TABLE 7–3 Booting a System *(continued)*

If You Are Rebooting the System After ...	Use This Boot Type ...	See SPARC Procedure ...	See IA Procedure ...
Repairing a system configuration file such as `/etc/system`	Interactive boot	"SPARC: How to Boot a System Interactively" on page 140	"IA: How to Boot a System Interactively" on page 154
Adding or removing hardware from the system	Reconfiguration boot (plus turning on system power after adding or removing hardware)	"SPARC: How to Connect a Secondary Disk and Boot" on page 365	Chapter 31
Booting the kernel debugger (`kadb`) to track down a system problem	Booting `kabd`	"SPARC: How to Boot the System With the Kernel Debugger (`kadb`)" on page 147	"IA: How to Boot a System with the Kernel Debugger (`kadb`)" on page 160
Repairing an important system file which is causing system boot failure	Recovery boot	"IA: How to Boot a System for Recovery Purposes" on page 157	"IA: How to Boot a System for Recovery Purposes" on page 157
Recovering from a hung system and you want to force a crash dump	Recovery boot	See example on "IA: How to Force a Crash Dump and Reboot the System" on page 161	See example on "IA: How to Force a Crash Dump and Reboot the System" on page 161

See Chapter 10 or Chapter 11 for examples of booting a system.

CHAPTER 8

Run Levels and Boot Files (Tasks)

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.

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 environment has eight run levels, which are described in the following table. The default run level is specified in the `/etc/inittab` file as run level 3.

TABLE 8–1 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.
s or S	Single-user state	Single-user	To run as a single user with all file systems mounted and accessible.
1	Administrative state	Single-user	To access all available file systems with user logins allowed.
2	Multiuser state	Multiuser	For normal operations. Multiple users can access the system and the entire file system. All daemons are running except for the NFS server daemons.
3	Multiuser state 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 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).

▼ 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  Sep  1 14:45     3      0  S
$
```

`run level 3`	Identifies the current run level.
`Sep 1 14:45`	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.

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*

The following table describes the fields in an `inittab` entry.

TABLE 8–2 Fields in the `inittab` File

Field	Description
id	A unique identifier for the entry.
rstate	A list of run levels to which this entry applies.
action	How the process specified in the process field is to be run. Possible values include: `initdefault`, `sysinit`, `boot`, `bootwait`, `wait`, and `respawn`.
process	The 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 ap::sysinit:/sbin/soconfig -f /etc/sock2path
3 fs::sysinit:/sbin/rcS sysinit   >/dev/msglog 2<>/dev/msglog </dev/console
4 is:3:initdefault:
5 p3:s1234:powerfail:/usr/sbin/shutdown -y -i5 -g0 >/dev/msglog 2<>/dev/...
6 sS:s:wait:/sbin/rcS              >/dev/msglog 2<>/dev/msglog </dev/
console
7 s0:0:wait:/sbin/rc0              >/dev/msglog 2<>/dev/msglog </dev/
console
8 s1:1:respawn:/sbin/rc1           >/dev/msglog 2<>/dev/msglog </dev/
console
9 s2:23:wait:/sbin/rc2             >/dev/msglog 2<>/dev/msglog </dev/
console
10 s3:3:wait:/sbin/rc3             >/dev/msglog 2<>/dev/msglog </dev/
console
11 s5:5:wait:/sbin/rc5             >/dev/msglog 2<>/dev/msglog </dev/
console
12 s6:6:wait:/sbin/rc6             >/dev/msglog 2<>/dev/msglog </dev/
console
13 fw:0:wait:/sbin/uadmin 2 0      >/dev/msglog 2<>/dev/msglog </dev/
console
14 of:5:wait:/sbin/uadmin 2 6      >/dev/msglog 2<>/dev/msglog </dev/
console
15 rb:6:wait:/sbin/uadmin 2 1      >/dev/msglog 2<>/dev/msglog </dev/
console
16 sc:234:respawn:/usr/lib/saf/sac -t 300
17 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
```

(continued)

1. Initializes STREAMS modules
2. Configures socket transport providers
3. Initializes file systems
4. Defines default run level
5. Describes a power fail shutdown
6. Defines single-user mode
7. Defines run level 0
8. Defines run level 1
9. Defines run level 2
10. Defines run level 3
11. Defines run level 5
12. Defines run level 6
13. Defines an unused level, firmware
14. Defines an unused level, off
15. Defines an unused level, reboot
16. Initializes Service Access Controller
17. Initializes console

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.

   The following table describes the key words used for run level 3's action field.

TABLE 8–3 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.

The following table describes the processes (or commands) executed at run level 3.

TABLE 8–4 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/rcS`	Mounts and checks root (`/`), `/usr`, `/var`, and `/var/adm` file systems.
`/sbin/rc2`	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 based system is `sun` The *terminal_type* on an IA based system is `AT386`

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
K07dmi              S70uucp             S75cron             S91afbinit
K07snmpdx           S71ldap.client      S75flashprom        S91ifbinit
K28nfs.server       S71rpc              S75savecore         S92volmgt
README              S71sysid.sys        S76nscd             S93cacheos.finish
S01MOUNTFSYS        S72autoinstall      S80PRESERVE         S94ncalogd
S05RMTMPFILES       S72inetsvc          S80lp               S95IIim
S20sysetup          S72slpd             S80spc              S95amiserv
S21perf             S73cachefs.daemon   S85power            S95ocfserv
S30sysid.net        S73nfs.client       S88sendmail         S99audit
S40llc2             S74autofs           S88utmpd            S99dtlogin
S47asppp            S74syslog           S89bdconfig
S69inet             S74xntpd            S90wbem
```

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

[KS][0-9][0-9]*

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*n*.d directories.

The actions of each run control script are summarized in Table 8–5.

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 superuser.**

2. **Turn off functionality.**

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

3. **Restart functionality.**

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

4. **Use the `pgrep` command to verify whether the service has been stopped or started.**

```
# pgrep -f 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
# pgrep -f nfs
#
```

Restart the NFS services by typing:

```
# /etc/init.d/nfs.server start
# pgrep -f nfs
141
143
245
```

```
247
# pgrep -f nfs -d, | xargs ps -fp
daemon   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
```

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`*n*`.d` directory you want the service to start and stop.

See the `README` file in each `/etc/rc`*n*`.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 superuser.**

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

```
# cp filename /etc/init.d
# chmod 0744 /etc/init.d/filename
# chown root:sys /etc/init.d/filename
```

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

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

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

```
# ls /etc/init.d/ /etc/rc2.d/ /etc/rcn.d/
```

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
# ls /etc/init.d /etc/rc2.d /etc/rc0.d
```

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 superuser.

2. Rename the script by adding an underscore (_) to the beginning of the new file.

```
# cd /etc/rcn.d
# mv filename _filename
```

3. Verify the script has been renamed.

```
# ls
# _filename
```

Example—Disabling a Run Control Script

The following example changes the `S100datainit` script name but saves the original script.

```
# cd /etc/rc2.d
# mv S100datainit _S100datainit
```

Run Control Script Summaries

TABLE 8–5 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 8–6 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 8–7 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 ■ 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`, `xntpd`, and `utmpd` ■ Mounts all NFS entries ■ Starts `nscd` (name service cache daemon) ■ Starts `automount`, `cron`, LP print service, `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 8–8 The `/sbin/rc3` Script

Script Name	Description
`/sbin/rc3`	Runs the `/etc/rc3.d` scripts to perform the following tasks: ■ Cleans up `sharetab` ■ Starts `nfsd` ■ Starts `mountd` ■ If the system is a boot server, starts `rarpd`, `rpc.bootparamd`, and `rpld` ■ Starts `snmpdx` (Solstice Enterprise Agents™ process).

TABLE 8–9 The /sbin/rc5 and /sbin/rc6 Scripts

Script Name	Description
/sbin/rc5 and /sbin/rc6	Runs the /etc/rc0.d/K* scripts to perform the following tasks: ■ Kills all active processes ■ Unmounts the file systems

TABLE 8–10 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

CHAPTER 9

Shutting Down a System (Tasks)

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 8.

When to Shut Down the System

Solaris 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 7 for a complete list of system administration tasks requiring a system shutdown.

For information on using your system's power management features, see *Using Power Management*.

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 10 or Chapter 11 for instructions on system recovery techniques.

The following table describes the various shutdown commands and provides recommendations for using them.

TABLE 9–1 Shutdown 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`	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.

TABLE 9–1 Shutdown Commands *(continued)*

Command	Description	This Command Is ...
`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

System devices to power down 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, a warning followed by a final shutdown message is broadcast to all users currently logged onto the system and all systems that are mounting resources from the affected system.

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

Use the who(1) 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 "How to Determine a System's Run Level" on page 110.

▼ 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 [1]     console [2]    May  7 07:30
kryten        pts/0          May  7 07:35 [3](starbug) [4]
lister        pts/1          May  7 07:40 [3](bluemidget)
```

1. Identifies the user name of the logged-in user.
2. Identifies the terminal line of the logged-in user.
3. Identifies the date and time the user logged in.
4. (Optional) Identifies the host name if a user is logged in from a remote system.

▼ How to Shut Down a Server

1. **Become superuser.**

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

```
# who
```

A list of all logged-in users is displayed. You might 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 **(1M) command.**

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

–i*init-state*	Brings 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 superuser 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.

7. 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 Based System Prompt Should Be ...	The IA Based 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:`

SPARC: Example—Bringing a System to Run Level S (Server)

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

```
# who
root       console      Jul 14 13:53
# shutdown -g180 -y

Shutdown started.    Wed Jul 14 13:55:55 MDT 1999

Broadcast Message from root (console) on earth Wed Jul 14 13:55:56...
The system earth will be shut down in 3 minutes
.
.
.
Broadcast Message from root (console) on earth Wed Jul 14 13:58:28...
The system earth will be shut down in 30 seconds
.
.
.
INIT: New run level: S
The system is coming down for administration.  Please wait.
Unmounting remote filesystems: /vol nfs done.
Jul 14 13:59:15 earth /usr/sbin/vold[376]: problem unmounting /vol;
Print services stopped.
Jul 14 13:59:16 earth syslogd: going down on signal 15
Killing user processes: done.

INIT: SINGLE USER MODE

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

SPARC: Example—Bringing a System to Run Level 0 (Server)

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

```
# who
root       console      Jul 14 14:01
rimmer     pts/0        Jul 14 14:03    (starbug)
pmorph     pts/1        Jul 14 14:04    (bluemidget)
# shutdown -i0 -g300 -y
Shutdown started.    Wed Jul 14 14:05:03 MDT 1999

Broadcast Message from root (console) on earth Wed Jul 14 14:05:03...
The system earth will be shut down in 5 minutes
.
.
.
```

(continued)

(Continuation)

```
Changing to init state 0 - please wait
#
INIT: New run level: 0
The system is coming down.  Please wait.
System services are now being stopped.
.
.
.
The system is down.
syncing file systems... done
Program terminated
Type  help  for more information
ok
```

See “How to Turn Off Power to All Devices” on page 132 if you are bringing the system to run level 0 to turn off power to all devices.

SPARC: Example—Rebooting a System to Run Level 3 (Server)

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

```
# who
root       console      Jul 14 14:14
rimmer     pts/0        Jul 14 14:15    (starbug)
pmorph     pts/1        Jul 14 14:15    (bluemidget)
# shutdown -i6 -g120 -y
Shutdown started.    Wed Jul 14 14:16:08 MDT 1999

Broadcast Message from root (console) on earth Wed Jul 14 14:16:08...
The system earth will be shut down in 2 minutes
.
.
.
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... done
rebooting...
.
.
.
```

(continued)

(Continuation)

```
earth 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 10 or Chapter 11 for instructions on bringing a system back to a multiuser state.

▼ How to Shut Down a Standalone System

1. **Become superuser.**

2. **Shut down the system by using the `init`(1M) command.**

```
# init run-level
```

run-level — Identifies the new run level.

3. **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 Based System Prompt Should Be ...	The IA Based 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 resources shared)	*hostname* `console login:`	*hostname* `console login:`

IA: Example—Bringing a System to Run Level 0 (Standalone)

In the following example, the `init` command is used to bring an IA based standalone 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 132 if you are bringing the system to run level 0 to turn off power to all devices.

SPARC: Example—Bringing a System to Run Level S (Standalone)

In the following example, the `init` is used to bring a SPARC based standalone system to run level S (single-user state).

```
# init s
#
INIT: New run level: S
The system is coming down for administration.  Please wait.
Unmounting remote filesystems: /vol nfs done.
Print services stopped.
syslogd: going down on signal 15
Killing user processes: done.
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 10 or Chapter 11 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 126.	See “How to Shut Down a Standalone System” on page 130.

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 is brought to run level 3 after the CPU is turned on.

CHAPTER 10

SPARC: Booting a System (Tasks)

This chapter describes procedures for using the OpenBoot™ PROM monitor and procedures for booting a SPARC based 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 12. For information on troubleshooting booting problems, see "What to Do If Rebooting Fails" in *System Administration Guide, Volume 2*.

For step-by-step instructions on booting an IA based system, see Chapter 11.

SPARC: Using the Boot PROM

System administrators typically use the PROM level to boot a system. Occasionally, however, you might need to change the way the system works, such as resetting 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 to add a new drive to the system either permanently or temporarily, change the network boot strategy, or if you want to temporarily boot a standalone system 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 based 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
Sun Ultra 5/10 UPA/PCI (UltraSPARC-IIi 333MHz), No Keyboard
OpenBoot 3.15, 128 MB memory installed, Serial #nnnnnnnn.
Ethernet address 8:0:20:a5:d1:3b, Host ID: 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

1. **Become superuser.**

2. **Halt the system by using the** `init`**(1M) command.**

```
# init 0
```

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 device[n]
```

`boot-device`	Identifies the parameter for setting the device from which to boot.
device[*n*]	Identifies the `boot-device` value such as a disk or the network. The *n* can be specified as the *disk number*.

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

5. **Verify the default boot device 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 new `boot-device` setting is written to the PROM.

SPARC: Examples—Changing the Default Boot Device

In this example, the default boot device is set to disk.

```
# init 0
#
INIT: New run level: 0
.
.
.
The system is down.
syncing file systems... done
Program terminated
ok setenv boot-device disk
boot-device =         disk
ok printenv boot-device
boot-device           disk                    disk
ok reset
Sun Ultra 5/10 UPA/PCI (UltraSPARC-IIi 333MHz), No Keyboard
OpenBoot 3.15, 128 MB memory installed, Serial #nnnnnnnn.
Ethernet address 8:0:20:a5:d3:4b, Host ID: nnnnnnnn.

Boot device: disk  File and args:
SunOS Release 5.8 Version 64-bit
.
.
.
pluto console login:
```

In this example the default boot device is set to the network.

```
# init 0
#
INIT: New run level: 0
.
.
.
The system is down.
syncing file systems... done
Program terminated
ok setenv boot-device net
boot-device =         net
ok printenv boot-device
boot-device           net                    disk
ok reset
Sun Ultra 5/10 UPA/PCI (UltraSPARC-IIi 333MHz), No Keyboard
OpenBoot 3.15, 128 MB memory installed, Serial #nnnnnnnn.
Ethernet address 8:0:20:a3:d54:4b, Host ID: nnnnnnnn.



Boot device: net  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.

SPARC: Booting a System

The table below describes the boot scenarios covered in this chapter.

TABLE 10–1 Boot Type Descriptions

Booting the System ...	Is Usually Done ...	See ...
To run level 3 (multiuser state with NFS resources 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.	"SPARC: How to Boot a System to Run Level 3 (Multiuser State)" on page 138
To run level S (single-user state)	After performing some system maintenance task such as backing up a file system. At this level, only local file systems are mounted and users cannot log into the system.	"SPARC: How to Boot a System to Run Level S (Single-User State)" on page 139
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.	"SPARC: How to Boot a System Interactively" on page 140
Over the network	To boot a system over the network. This procedure assumes the necessary setup has been completed on the boot server.	"SPARC: How to Boot a System Over the Network" on page 142

TABLE 10–1 Boot Type Descriptions *(continued)*

Booting the System ...	Is Usually Done ...	See ...
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.	"SPARC: How to Boot a System for Recovery Purposes" on page 143
Using `kadb`	To troubleshoot system problems by running the kernel debugger.	"SPARC: How to Stop the System for Recovery Purposes" on page 145

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 8 for a description of run levels.

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

1. **Boot to run level 3 by using the `boot`(1M) command.**

```
ok boot
```

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

2. **Verify the system boots to run level 3.**

   The login prompt is displayed when the boot process has finished successfully.

```
hostname console login:
```

SPARC: 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
SPARCstation 10 (1 X 390Z50)
ROM Rev. 2.14, 32 MB memory installed, Serial #number.
Ethernet address number, Host ID: number.

Rebooting with command:
Boot device: /iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,8...
SunOS Release 5.8 Version Generic 32-bit
Copyright (c) 1983-2000 by Sun Microsystems, Inc.
configuring IPv4 interfaces: le0.
Hostname: earth
The system is coming up.  Please wait.
checking ufs filesystems
/dev/rdsk/c0t3d0s7: is clean.
NIS domainname is Solar.COM
starting rpc services: rpcbind keyserv ypbind done.
Setting netmask of le0 to 255.255.255.0
Setting default IPv4 interface for multicast: add net 224.0/
4: gateway earth
syslog service starting.
Print services started.
volume management starting.
The system is ready.

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

```
# who -r
.       run-level 3  Jun 10 15:27     3      0
```

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

SPARC: 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 5.8 Version Generic 32-bit
Copyright (c) 1983-2000 by Sun Microsystems, Inc.
configuring IPv4 interfaces: le0.
Hostname: earth

INIT: SINGLE USER MODE

Type control-d to proceed with normal startup,
(or give root password for system maintenance): xxx
Sun Microsystems Inc.   SunOS 5.8  generic August 1999
# who -r
   .       run-level S  Jul 14 11:37     S      0  ?
(Perform some maintenance task)
# Press <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 the following table.**

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 the name of another kernel to use for booting. Or, press Return to use the default kernel (`/platform/‘uname -m‘/kernel/unix`).
`Name of system file [/etc/system]:`	Provide the name of an alternate system file and press Return. Type `/dev/null` if your `/etc/system` file has been damaged. 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 network booting.
`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.

3. **If you are not prompted to answer the questions in the table above, verify that you entered the** `boot -a` **command correctly.**

SPARC: Example—Booting a System Interactively

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

```
ok boot -a
.
.
.
Rebooting with command: boot -a
Boot device: /pci@1f,0/pci@1,1/ide@3/disk@0,0:a  File and args: -a
Enter filename [kernel/sparcv9/unix]: Return
Enter default directory for modules [/platform/SUNW,Ultra-5_10/kernel
```

(continued)

```
/platform/sun4u/kernel /kernel /usr/kernel]: Return
Name of system file [etc/system]: Return
SunOS Release 5.8 Version Generic 64-bit
Copyright (c) 1983-2000 by Sun Microsystems, Inc.
root filesystem type [ufs]: Return
Enter physical name of root device
[/pci@1f,0/pci@1,1/ide@3/disk@0,0:a]: Return
configuring IPv4 interfaces: hme0.
Hostname: starbug
The system is coming up.  Please wait.
checking ufs filesystems
.
.
.
The system is ready.
earth console login:
```

▼ SPARC: How to Boot a System Over the Network

Any system can boot over the network if there is a boot server available. You might want to boot a standalone system over the network temporarily if it cannot boot from the local disk. See "SPARC: How to Change the Default Boot Device" on page 134 for information on changing or resetting the default boot device.

There are two network configuration boot strategies to choose from on sun4u systems: RARP (Reverse Address Resolution Protocol and ONC+ RPC Bootparams Protocol) or DHCP (Dynamic Host Configuration Protocol). The default network boot strategy is set to RARP. You can use either one depending on whether a RARP or DHCP boot server is available in your network.

Note - Sun Ultra systems must have PROM version 3.25 or later to use the DHCP network boot strategy.

If both methods are available, you can specify which service to use in the `boot` command temporarily, or save the network boot strategy across system reboots at the PROM level, by setting up an NVRAM alias. The following `nvalias` command example sets up a network device alias for booting DHCP by default on a Sun Ultra 10 system.

```
ok nvalias net  /pci@1f,4000/network@1,1:dhcp
```

This alias means that when you type `boot net`, the system will boot using DHCP.

Caution - You should not use the `nvalias` command to modify the `NVRAMRC` file unless you are very familiar with the syntax of this command and the `nvunalias` command. See the *OpenBoot 3.x Command Reference Manual* for information on using these commands.

1. **Determine the method for booting over the network and select one of the following.**

   There must be a RARP or DHCP boot server already set up in your network for either of these methods to boot successfully.

   a. **Boot the system over the network by using the DHCP method.**

   ```
   ok boot net[:dhcp]
   ```

   If you have changed the PROM setting to boot DHCP by default, like in the `nvalias` example above, you only have to specify `boot net` to boot using the DHCP method.

   b. **Boot the system over the network by using the RARP method.**

   ```
   ok boot net[:rarp]
   ```

   Since RARP is the default network boot strategy, you only have to specify `boot net:rarp` if you have changed the PROM value to boot DHCP.

▼ 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 cause the boot process to fail.

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

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

If You Are Booting From ...	Then ...
Solaris installation CD	1. Insert the Solaris 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
```

9. **Verify the system boots to run level 3.**

   The login prompt is displayed when the boot process has finished successfully.

```
hostname console login:
```

SPARC: Example—Booting a System for Recovery Purposes

The following example shows how to repair an important system file (in this case, /etc/passwd) after booting from a local CD-ROM.

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

▼ SPARC: How to Stop the System for Recovery Purposes

1. **Type the abort key sequence for your system.**

   The monitor displays the ok PROM prompt.

```
ok
```

   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.

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`**(1M) command to start the boot process.**

5. **Verify the system is booted to the specified run level.**

```
# who -r
  .       run-level 3  May  2 07:39     3      0  S
```

SPARC: Example—Stopping the System for Recovery Purposes

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

SPARC: Forcing a Crash Dump and Rebooting the System

Saving crash dumps of the operating system is sometimes necessary for troubleshooting purposes. The `savecore` feature and how it is set up is described in "Managing System Crash Information" in *System Administration Guide, Volume 2*. This section only describes how to reboot the system when 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
```

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

3. **Verify the system boots to run level 3.**

The login prompt is displayed when the boot process has finished successfully.

```
hostname console login:
```

SPARC: Example—Forcing a Crash Dump and Rebooting the System

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

▼ SPARC: How to Boot the System With 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 by using the kernel debugger.**

```
ok boot kadb
```

5. **Identify** `kadb` **booting messages to verify that the system has booted using the kernel debugger.**

```
Rebooting with command: kadb
Boot device: /iommu/sbus/espdma@4,800000/esp@4,8800000/sd@3,0
.
.
.
```

SPARC: Example—Booting the System With the Kernel Debugger (kadb)

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

CHAPTER 11

IA: Booting a System (Tasks)

This chapter describes the procedures for booting an IA based system.

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

For overview information about the boot process, see Chapter 12.

For step-by-step instructions on booting a SPARC based system, see Chapter 10.

IA: Booting a System

The following table describes the boot types covered in this chapter.

TABLE 11–1 Boot Type Descriptions

Booting the System ...	Is Usually Done ...	See ...
To run the Solaris Device Configuration Assistant	After changing the hardware configuration of the system. This utility enables you to boot the Solaris system from a different boot device, configure new or misconfigured hardware, or perform other device- or boot-related tasks.	"IA: How to Boot the Solaris Device Configuration Assistant" on page 151
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.	"IA: How to Boot a System to Run Level 3 (Multiuser State)" on page 151
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.	"IA: How to Boot a System to Run Level S (Single-User State)" on page 152
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.	"IA: How to Boot a System Interactively" on page 154
Over the network	To boot a system over the network. This procedure assumes the necessary setup has been completed on the boot server.	"IA: How to Boot a System Over the Network" on page 156
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.	"IA: How to Boot a System for Recovery Purposes" on page 157
To run the Solaris kernel debugger (`kadb`)	To troubleshooting system problems.	"IA: How to Boot a System with the Kernel Debugger (`kadb`)" on page 160
To force a crash dump	To troubleshoot system problems and saving core dumps of the operating system.	"IA: How to Force a Crash Dump and Reboot the System" on page 161

TABLE 11–1 Boot Type Descriptions *(continued)*

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.

IA: Booting the Solaris Device Configuration Assistant

The Solaris Device Configuration Assistant is a program that enables you to perform various hardware configuration and booting tasks. Two ways to access the Solaris Device Configuration Assistant are from the:

- Solaris Boot Diskette
- Solaris Installation CD

In the following sections you might be requested to insert the Solaris Device Configuration Assistant Boot Diskette to boot the Configuration Assistant. If your system's BIOS supports booting from the CD, you may, instead, insert the Solaris installation CD to boot the Configuration Assistant.

▼ IA: How to Boot the Solaris Device Configuration Assistant

1. **Insert the Solaris Device Configuration Boot Diskette or the Solaris Installation CD in the appropriate drive.**

2. **Press any key to reboot the system if the system displays the** `Type any key to reboot` **prompt. You can also use the reset button at this prompt. If the system is shut down, turn the system on with the power (on/off) switch.**

3. **The first menu of the Configuration Assistant is displayed after a few minutes.**

▼ IA: 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. You can also use the reset button at this prompt. If the system is shut down, turn the system on with the power (on/off) switch.**

The Current Boot Parameters menu is displayed after a few minutes.

2. **Type** `b` **to boot the system to run level 3. Press Enter.**

   If you do not make a selection within five seconds, the system is automatically booted to run level 3.

3. **Verify the system boots to run level 3.**

   The login prompt is displayed when the boot process has finished successfully.

```
hostname console login:
```

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

```
Type any key to reboot
           .
           .
           .
                     <<< Current Boot Parameters >>>
Boot path: /pci@0,0/pci-ide@7,1/ide@0/cmdk@0,0:a
Boot args:
Type    b [file-name] [boot-flags] <ENTER>      to boot with options
or      i <ENTER>                               to enter boot interpreter
or      <ENTER>                                 to boot with defaults

                  <<< timeout in 5 seconds >>>

Select (b)oot or (i)nterpreter: b
              .
              .
              .
venus console login:
```

▼ IA: 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. You can also use the reset button at this prompt. If the system is shut down, turn the system on with the power (on/off) switch.**

   The Current Boot Parameters menu is displayed after a few minutes.

2. **Type** `b -s` **to boot the system to run level S. Press Enter.**

If you do not make a selection within five seconds, the system is automatically booted to run level 3.

3. **Type the superuser password, if prompted.**

4. **Verify the system is at run level S by using the** `who -r` **command.**

```
# who -r
   .       run-level S  Jul 19 14:37     S      0  3
```

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.**

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

```
Type any key to reboot
            .
            .
            .

                    <<< Current Boot Parameters >>>
Boot path: /pci@0,0/pci-ide@7,1/ide@0/cmdk@0,0:a
Boot args:
Type    b [file-name] [boot-flags] <ENTER>      to boot with options
or      i <ENTER>                               to enter boot interpreter
or      <ENTER>                                 to boot with defaults


                    <<< timeout in 5 seconds >>>

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
Entering System Maintenance Mode
              .
              .
              .
# who -r
```

(continued)

(Continuation)

```
   .       run-level S  Jul 19 14:37     S      0  3
(Perform some maintenance task)
# Press <Control-d>
```

▼ IA: How to Boot a System Interactively

1. **Press any key to reboot the system if the system displays the** `Type any key to reboot` **prompt. You can also use the reset button at this prompt. If the system is shut down, turn the system on with the power (on/off) switch.**

   The Primary Boot Subsystem menu is displayed after a few minutes.

2. **Select the Solaris partition (if not marked as active) from the list and press Enter. If you do not make a selection within 30 seconds, the active boot partition is selected automatically.**

   The Current Boot Parameters menu is displayed after a few minutes.

3. **Type** `b -a` **to boot the system interactively. Press Enter.**

   If you do not make a selection within five seconds, the system is automatically booted to run level 3.

4. **Answer the system prompts as described in the following table.**

If the System Displays ...	Do the Following ...
`Enter default directory for modules: [/platform/i86pc/kernel /kernel /usr/kernel]:`	Provide an alternate path for the `modules` directory and press Enter, or press Enter to use the default `modules` directory path.
`Name of system file [etc/system]:`	Provide the name of an alternate system file and press Enter, or press Enter to use the default `/etc/system` file. Type `/dev/null` if your `/etc/system` file has been damaged.

If the System Displays ...	Do the Following ...
`root filesystem type [ufs]:`	Press Enter to use the default root file system type: UFS for local disk booting, or NFS for network booting.
`Enter physical name of root device` *[physical_device_name]:*	Provide an alternate device name and press Enter, or press Enter to use the default physical name of the root device `bootpath`.

IA: Example—Booting a System Interactively

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

```
Type any key to reboot
            .
            .
            .

                    <<< Current Boot Parameters >>>
Boot path: /pci@0,0/pci-ide@7,1/ide@0/cmdk@0,0:a
Boot args:
Type    b [file-name] [boot-flags] <ENTER>      to boot with options
or      i  <ENTER>                              to enter boot interpreter
or      <ENTER>                                 to boot with defaults


                    <<< timeout in 5 seconds >>>>
Select (b)oot or (i)nterpreter: b -a
Enter default directory for modules [/platform/i86pc/kernel /kernel /usr/
kernel]: Enter
Name of system file [etc/system]:Enter
SunOS Release 5.8 Version Generic 32-bit
Copyright (c) 1983-2000 by Sun Microsystems, Inc.
root filesystem type [ufs]: Enter
Enter physical name of root device
[/pci@0,0/pci-ide@7,1/ide@0/cmdk@0,0:a]: Enter
configuring IPv4 interfaces: dnet0.
Hostname: venus
(fsck messages)
The system is coming up.  Please wait
(More messages)
venus console login:
```

▼ IA: How to Boot a System Over the Network

Any system can boot over the network if there is a boot server available. You might want to boot a standalone system over the network temporarily if it cannot boot from the local disk.

The new menu, Set Network Configuration Strategy, on the Configuration Assistant's Boot Tasks Menu, enables you to select the appropriate boot strategy.

1. **Determine whether you want to boot over the network using the RARP/ bootparams method or the DHCP method.**

   There are two network configuration strategies to choose from, RARP (Reverse Address Resolution Protocol) or DHCP (Dynamic Host Configuration Protocol). The default network boot strategy is set to RARP. You can use either one depending on whether a RARP or DHCP boot server is available in your network.

2. **Insert the Configuration Assistant Boot Diskette or the Installation CD you wish to boot from.**

3. **Press any key to reboot the system if the system displays the** `Type any key to reboot` **prompt. You can also use the reset button at this prompt. If the system is shut down, turn the system on with the power (on/off) switch.**

4. **Press F2_Continue at the Solaris Device Configuration Assistant screen to scan for devices.**

   Device identification is performed and a screen that displays the identified devices appears.

5. **Press F2_Continue at the Identified Devices screen to load drivers.**

   Bootable drivers are loaded.

6. **Press F4_Boot Tasks from the Boot Solaris menu.**

7. **Select** `Set Network Configuration Strategy` **and press F2_Continue.**

8. **Select either** `RARP` **or** `DHCP` **and press F2_Continue.**

   A screen that confirms your new network configuration strategy appears.

   Your network configuration strategy selection is saved as the default network boot method the next time this diskette is used for booting.

9. **Press F3_Back to return to the Boot Solaris menu.**

10. **Select** `NET` **as the boot device from the Boot Solaris menu. Then press F2_Continue to boot the network device.**

    The Solaris boot option screen is displayed.

▼ IA: How to Boot a System for Recovery Purposes

Follow these steps to boot the system to repair a critical system resource. The example shows you how to boot from a Solaris Installation CD or the network, mount the root (/) file system on the disk, and repair the `/etc/passwd` file.

Substitute the device name of the file system to be repaired for the *devicename* variable in the procedures below. If you need help identifying a system's device names, refer to Chapter 26.

Follow the instructions below to boot from the Solaris installation CD or the network.

1. **Boot from the Solaris installation CD (or the network) to single-user mode.**
   a. **Insert the Configuration Assistant Boot Diskette or the Installation CD you wish to boot from.**

   b. **Press any key to reboot the system if the system displays the** `Type any key to reboot` **prompt. You can also use the reset button at this prompt. If the system is shut down, turn the system on with the power (on/off) switch.**

   c. **Press the F2 key (F2_Continue) at the Solaris Device Configuration Assistant screen.**
      Device identification is performed and a screen that displays the identified devices appears.

   d. **Press the F2 key (F2_Continue) at the Identified Devices screen.**
      Bootable drivers are loaded.

   e. **Press the F2 key (F2_Continue) at the Solaris Device Configuration Assistant screen.**
      Device identification is performed and a screen that displays the identified devices appears.

   f. **Press the F2 key (F2_Continue) at the Identified Devices screen.**
      Bootable drivers are loaded.

   g. **Select the CD-ROM drive or network device from the Boot Solaris menu. Then press the F2 key (F2_Continue).**
      The Current Boot Parameters menu is displayed.

   h. **Type** `b -s` **at the prompt. Press Enter.**
      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. **Make the necessary change to the `passwd` file using an editor.**

```
# vi passwd
```

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

```
# cd /
```

6. **Unmount the `/a` directory.**

```
# umount /a
```

7. **Reboot the system.**

```
# init 6
```

8. **Verify the system boots to run level 3.**

   The login prompt is displayed when the boot process has finished successfully.

```
hostname console login:
```

IA: Example—Booting a System for Recovery Purposes

```
Type any key to reboot

SunOS Secondary Boot version 3.00


Solaris Intel Platform Edition Booting System

Running Configuration Assistant...
Autobooting from Boot path: /pci@0,0/pci-ide@7,1/ide@0/cmdk@0,0:a
```

(continued)

(Continuation)

```
If the system hardware has changed, or to boot from a different
device, interrupt the autoboot process by pressing ESC.

Press ESCape to interrupt autoboot in 5 seconds.
      .
      .
      .
Boot Solaris

Select one of the identified devices to boot the Solaris kernel and
choose Continue.

To perform optional features, such as modifying the autoboot and property
settings, choose Boot Tasks.

An asterisk (*) indicates the current default boot device.

> To make a selection use the arrow keys, and press Enter to mark it [X].

[ ]  NET : DEC 21142/21143 Fast Ethernet
on Board PCI at Dev 3
[ ]  DISK: (*) Target 0, QUANTUM  FIREBALL1280A
on Bus Mastering IDE controller on Board PCI at Dev 7, Func 1
[ ]  DISK: Target 1:ST5660A
on Bus Mastering IDE controller on Board PCI at Dev 7, Func 1
[ ]  DISK: Target 0:Maxtor 9 0680D4
on Bus Mastering IDE controller on Board PCI at Dev 7, Func 1
[ ]  CD  : Target 1:TOSHIBA  CD-ROM XM-5602B  1546
on Bus Mastering IDE controller on Board PCI at Dev 7, Func 1

F2_Continue   F3_Back   F4_Boot Tasks   F6_Help
      .
      .
      .
              <<< Current Boot Parameters >>>
Boot path: /pci@0,0/pci-ide@7,1/ide@0/cmdk@0,0:a
Boot args: kernel/unix -r

Select the type of installation you want to perform:

1 Solaris Interactive
2 Custom JumpStart
3 Solaris Web Start

Enter the number of your choice followed by <ENTER> the key.

If you enter anything else, or if you wait for 30 seconds,
an interactive installation will be started.

Select type of installation:  b -s
      .
      .
```

(continued)

```
      .
# mount /dev/dsk/c0t0d0s0 /a
      .
      .
      .
# cd /a/etc
# vi passwd
(Remove invalid entry)
# cd /
# umount /a
# init 6
```

▼ IA: How to Stop the System for Recovery Purposes

If possible, stop the system by using one of the following commands:

- If the system is running, become superuser and type `init 0` to stop the system. Press any key to reboot the system after the `Type any key to reboot` prompt appears.
- If the system is running, become superuser and type `init 6` to reboot the system.

If the system doesn't respond to any input from the mouse or keyboard, press the reset key, if it exists, to reboot the system. Or you can use the power (on/off) switch to reboot the system.

▼ IA: How to Boot a System with the Kernel Debugger (kadb)

1. **Press any key to reboot the system if the system displays the `Type any key to reboot` prompt. You can also use the reset button at this prompt. If the system is shut down, turn the system on with the power (on/off) switch.**

2. **Type `b kadb` to boot the kernel debugger. Press Enter.**

   If you do not make a selection within five seconds, the system is automatically booted to run level 3.

3. **Verify the system boots to run level 3.**

   The login prompt is displayed when the boot process has finished successfully.

```
hostname console login:
```

4. **Verify that you can access the kernel debugger by pressing F1-a.**

   The `kadb[0]:` prompt is displayed when you enter the kernel debugger.

IA: Example—Booting a System with the Kernel Debugger (kadb)

```
Type any key to reboot
        .
        .
        .
            <<< Current Boot Parameters >>>
Boot path: /pci@0,0/pci-ide@7,1/ide@0/cmdk@0,0:a
Boot args:
Type    b [file-name] [boot-flags] <ENTER>     to boot with options
or      i <ENTER>                              to enter boot interpreter
or      <ENTER>                                to boot with defaults

            <<< timeout in 5 seconds >>>

Select (b)oot or (i)nterpreter: b kadb
        .
        .
        .
naboo console login: (Enter login and password)
(Press F1-a to verify you can access the kernel debugger)
```

IA: Forcing a Crash Dump and Rebooting the System

Saving core dumps of the operating system is sometimes necessary for troubleshooting purposes. The `savecore` feature and how it is set up is described in "Managing System Crash Information" in *System Administration Guide, Volume 2*. This section only describes how to reboot the system when the `savecore` feature is enabled.

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

The system must be booted with the kernel debugger option, `kadb`, to get to the `kadb[0]:` prompt and to enable forcing the crash dump.

Note - You must be in text mode to enter the kernel debugger (kadb), so exit any window system (CDE or Open Windows) first.

1. **Press F1–a.**

```
kadb[0]:
```

   The kadb[0]: prompt is displayed.

2. **Type the following commands at the** kadb[0]: **prompt.**

```
Press <F1-a>
kadb[0]: vfs_syncall/W ffffffff
kadb[0]: 0>eip
kadb[0]: :c
kadb[0]: :c
kadb[0]: :c
```

   After the first :c is typed, the system panics, so you need to type :c again. The system panics again, so type :c a third time to force the crash dump and reboot the system.

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

3. **Verify that the system has rebooted by logging in at the console login prompt.**

CHAPTER 12

The Boot Process (Reference)

This chapter describes the hardware used for booting on SPARC based and IA based 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 10 or Chapter 11.

SPARC: The Boot PROM

Each SPARC based 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.

SPARC platform only - Some older systems might require PROM upgrades before they will work with the Solaris system software. Contact your local service provider for more information.

SPARC: The Boot Process

The following table describes the boot process.

TABLE 12–1 Description of the 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, `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, `ufsboot`, and loads it into memory.
	4. After the `ufsboot` program is loaded, the `ufsboot` program 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.
	6. The kernel creates a user process and starts the `/sbin/init` process, which starts other processes by reading the `/etc/inittab` file.
`init`	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.

IA: 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.

IA: Boot Subsystems

At three times during the Solaris boot process, you can make the following choices about a booting system:

- Primary Boot Subsystem (Partition Boot Menu) - This first menu appears if multiple operating environments exist on the disk. The menu enables you to boot any of the operating environments installed. By default, the operating environment designed as *active* is booted.

  Note that if you choose to boot a non-Solaris operating environment, the next two menus cannot be reached.

- Interrupt the Autoboot Process - If the autoboot process is interrupted, you can access the Configuration Assistant.

  The Configuration Assistant enables you to boot the Solaris system from a different boot device, configure new or misconfigured hardware, or perform other device- or boot-related tasks.

- Current Boot Parameters Menu - Two forms of this menu exist, one for a normal Solaris boot and one for a Solaris installation boot:

  - The normal Current Boot Parameters menu enables you to boot the Solaris system with options, or enter the boot interpreter.
  - The install Current Boot Parameters menu enables you to select the type of installation to be performed, or customize the boot.

The following table summarizes the purpose of the primary IA boot interfaces. See the sections that follow for a detailed description and example of each boot subsystem.

TABLE 12–2 Boot Subsystems

Boot Subsystem	Purpose
Primary Boot Subsystem	This menu appears if the disk you are booting from contains multiple operating environments, including the Solaris operating environment.
Secondary Boot Subsystem	This menu appears each time you boot the Solaris release. The Solaris release is booted automatically unless you choose to run the Solaris Device Configuration Assistant by interrupting the autoboot process.
Solaris Device Configuration Assistant/Boot Diskette	There are two ways to access the Solaris Device Configuration Assistant menus: 1. Use the Solaris Device Configuration Assistant Boot Diskette or the Solaris Installation CD (on systems that can boot from the CD-ROM drive) to boot the system. 2. Interrupt the autoboot process when booting Solaris from an installed disk.
Current Boot Parameters Menu	This menu appears when you boot the Solaris release from the disk, CD-ROM, or the network. The menu presents a list of boot options.

During the boot process, the boot subsystem menus allow you to customize boot choices. If the system receives no response during the time-out periods, it continues to boot automatically using default selections. You can stop the boot process when each boot subsystem menu is displayed, or you can let it continue automatically.

The following section provides examples of each subsystem screen.

IA: Booting Solaris

During the device identification phase, the Configuration Assistant:

- Scans for devices installed on the system
- Displays the identified devices
- Enables you to perform optional tasks such as selecting a keyboard type and editing devices and their resources

During the Boot phase, the Configuration Assistant:

- Displays a list of devices from which to boot. A device marked with an asterisk (*) is the default boot device.
- Enables you to perform optional tasks, such as editing autoboot and property settings, and choosing the network configuration strategy.

Examples of device identification during each phase are provided below. Device output varies based on your system configuration.

IA: Menus Displayed During the Device Identification Phase

Several menus are displayed as the Configuration Assistant attempts to identify devices on the system.

IA: Configuration Assistant Screen

This screen appears each time you boot the Configuration Assistant and access the menus. The Configuration Assistant runs every time the system is booted, although the autoboot process bypasses the menus.

```
Solaris Device Configuration Assistant

The Solaris(TM) (Intel Platform Edition) Device Configuration Assistant
scans to identify system hardware, lists identified devices, and can
boot the Solaris software from a specified device. This program must be
used to install the Solaris operating environment, add a driver,
or change the hardware on the system.

> To perform a full scan to identify all system hardware, choose Continue.

> To diagnose possible full scan failures, choose Specific Scan.

> To add new or updated device drivers, choose Add Driver.

About navigation...
    - The mouse cannot be used.
    - If the keyboard does not have function keys or they do not respond,
      press ESC. The legend at the bottom of the screen will change to
      show the ESC keys to use for navigation.
    - The F2 key performs the default action.

F2_Continue  F3_Specific Scan  F4_Add Driver  F6_Help
```

IA: Bus Enumeration Screen

The Bus Enumeration screen appears briefly while the Configuration Assistant gathers hardware configuration data for devices that can be detected automatically.

```
Bus Enumeration

Determining bus types and gathering hardware configuration data ...

Please wait ...
```

IA: Scanning Devices Screen

The Scanning Devices screen appears while the Configuration Assistant manually scans for devices that can only be detected with special drivers.

```
Scanning Devices

The system is being scanned to identify system hardware.

If the scanning stalls, press the system's reset button. When the
system reboots, choose Specific Scan or Help.


Scanning: Floppy disk controller

######################
|      |        |         |         |         |
0     20       40        60        80       100

Please wait ...
```

IA: Identified Devices Screen

The Identified Devices screen displays which devices have been identified on the system. From here, you can continue to the Boot Solaris menu or perform optional tasks, such as set a keyboard configuration, view and edit devices, set up a serial console, and save and delete configurations.

```
Identified Devices

The following devices have been identified on this system. To identify
devices not on this list or to modify device characteristics, such as
keyboard configuration, choose Device Tasks. Platform types may be
included in this list.

ISA: Floppy disk controller
 ISA: Motherboard
 ISA: PnP bios: 16550-compatible serial controller
 ISA: PnP bios: 16550-compatible serial controller
 ISA: PnP bios: Mouse controller
 ISA: PnP bios: Parallel port
 ISA: System keyboard (US-English)
 PCI: Bus Mastering IDE controller
```

(continued)

```
 PCI: Universal Serial Bus
 PCI: VGA compatible display adapter

F2_Continue   F3_Back   F4_Device Tasks   F6_Help
```

IA: Menus Displayed During the Boot Phase

During this phase, you can determine the way in which the system is booted.

IA: Boot Solaris Menu

The Boot Solaris menu allows you to select the device from which to boot the Solaris release. You can also perform optional tasks, such as view and edit autoboot and property settings. Once a boot device is selected and you choose Continue, the Solaris kernel will begin to boot.

```
Boot Solaris
Select one of the identified devices to boot the Solaris kernel and
choose Continue.

To perform optional features, such as modifying the autoboot and property
settings, choose Boot Tasks.

An asterisk (*) indicates the current default boot device.

> To make a selection use the arrow keys, and press Enter to mark it [X].

[X]  DISK: (*) Target 0:QUANTUM  FIREBALL1280A
on Bus Mastering IDE controller on Board PCI at Dev 7, Func 1
[ ]  DISK: Target 1:ST5660A
on Bus Mastering IDE controller on Board PCI at Dev 7, Func 1
[ ]  DISK: Target 0:Maxtor 9 0680D4
 on Bus Mastering IDE controller on Board PCI at Dev 7, Func 1
[ ]  CD  : Target 1:TOSHIBA  CD-ROM XM-5602B  1546
 on Bus Mastering IDE controller on Board PCI at Dev 7, Func 1

F2_Continue   F3_Back   F4_Boot Tasks   F6_Help
```

IA: Current Boot Parameters Menu

This menu appears each time you boot Solaris from the local disk. Let the five-second timeout elapse if you want to boot the default Solaris kernel. If you want to boot with different options, select an appropriate option before the timeout period elapses.

```
                    <<< Current Boot Parameters >>>
Boot path: /pci@0,0/pci-ide@7,1/ide@0/cmdk@0,0:a
Boot args:
Type    b [file-name] [boot-flags] <ENTER>       to boot with options
or      i <ENTER>                                to enter boot interpreter
or      <ENTER>                                  to boot with defaults


                    <<< timeout in 5 seconds >>>

Select (b)oot or (i)nterpreter:
```

IA: The Boot Process

The following table describes the boot process.

TABLE 12–3 Description of the 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. This first disk sector on the boot device contains the master boot record `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, `pboot`, loads and executes `pboot`. 4. `pboot` loads `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. You have a 30–second interval to select an alternate partition from which to boot. This step only occurs if there is more than one bootable partition present on the system.

TABLE 12–3 Description of the Boot Process *(continued)*

Boot Phase	Description
	6. `bootblk` finds and executes the secondary boot program, `boot.bin` or `ufsboot`, in the root file system. You have a 5–second interval to interrupt the autoboot to start the Configuration Assistant. 7. The secondary boot program, `boot.bin` or `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. You have a 5–second interval to specify a boot option or start the boot interpreter.
Kernel Initialization	8. The kernel initializes itself and begins loading modules, using the secondary boot program (`boot.bin` or `ufsboot`) to read the files. When the kernel has loaded enough modules to mount the root file system, it unmaps the secondary boot 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.

CHAPTER 13

Managing Removable Media Topics

This section provides instructions for using removable media in the Solaris environment. This section contains these chapters.

Chapter 14	Provides general information about using CDs and diskettes, including a comparison of automatic and manual mounting.
Chapter 15	Provides step-by-step instructions for using CDs from the command line, plus instructions for starting and stopping Volume Management.
Chapter 16	Provides step-by-step instructions for formatting and using diskettes from the command line.
Chapter 17	Provides step-by-step instructions for formatting and using PCMCIA memory cards from the command line.
Chapter 18	Provides a high-level description of how the Solaris environment creates special mount points and symbolic links to provide easy access to diskettes and CDs.

CHAPTER 14

Guidelines for Using CDs and Diskettes (Overview)

This chapter provides general guidelines for using diskettes and CDs in the Solaris environment.

This is a list of overview information in this chapter.

Where to Find Managing Removable Media Tasks

Use these references to find step-by-step instructions for managing removable media.

- Chapter 15
- Chapter 16
- Chapter 17

For information on using removable media with File Manager in the Common Desktop Environment, see *Solaris Common Desktop Environment: User's Guide.*

Removable Media 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 Table 14–1.)
- It enables you to access diskettes and CDs without having to become superuser.
- It allows you to give other systems on the network automatic access to any diskettes and CDs you insert into your system (see Chapter 15 and Chapter 16).

Comparison of Automatic and Manual Mounting

The table below compares the steps involved in manual mounting (without Volume Management) and automatic mounting (with Volume Management).

TABLE 14–1 Comparison of Manual and Automatic Mounting

Steps	Manual Mounting	Automatic Mounting
1	Insert media.	Insert media.
2	Become superuser.	For diskettes, use the `volcheck` command.
3	Determine the location of the media device.	*Volume Management automatically performs many of the tasks previously required to manually mount and work with CDs and diskettes.*
4	Create a mount point.	
5	Make sure you are not in the mount point directory.	
6	Mount the device using the proper `mount` options.	

TABLE 14–1 Comparison of Manual and Automatic Mounting *(continued)*

Steps	Manual Mounting	Automatic Mounting
7	Exit the superuser account.	
8	Work with files on media.	Work with files on media.
9	Become superuser.	
10	Unmount the media device.	
11	Eject media.	Eject media.
12	Exit the superuser account.	

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 superuser access. To make diskettes and CDs easier to work with, they are mounted in easy-to-remember locations.

TABLE 14–2 How to Access Data on Diskettes and CDs

To Access ...	Insert ...	And Find the Files In ...
Files on a diskette	The diskette and enter `volcheck`	`/vol/dev/aliases/floppy0`
Raw data on a diskette	The diskette and enter `volcheck`	`/vol/dev/aliases/floppy0`
Files on a CD	The CD and wait for a few seconds	`/cdrom/cdrom0`

If your system has more than one diskette or CD-ROM drive, see the table below for their access points.

TABLE 14–3 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`

CHAPTER 15

Using CDs From the Command Line (Tasks)

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.

Using CDs Task Map

TABLE 15–1 Using CDs Task Map

Task	Description	For Instructions, Go To ...
1. Load the CD	Insert the CD into the CD-ROM drive.	"How to Load a CD" on page 181
2. 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" on page 181
3. 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" on page 181
4. 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" on page 182
5. Eject the CD	When you finish, eject the CD from the CD-ROM drive.	"How to Eject a CD" on page 183

Using CD Names

When working with CDs, you can identify them by name or with a designator from the table below. For brevity, task descriptions use `cdrom0`, but you can replace this with either the CD name or a different designator.

TABLE 15–2 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 “How to Examine the Contents of a CD” on page 181.

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 18, 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 `ls -L` command to view the contents of `/cdrom` directory.

```
$ ls -L [-l] /cdrom/cdrom0
```

`–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
total 166
drwxr-xr-x   4 root     root        2048 Jul 21 05:18 MU
drwxr-xr-x   4 root     root        2048 Jul 21 05:18 Solaris_7_MU3
-rwxr-xr-x   1 root     root       30952 Jul 21 05:18 backout_mu
-rwxr-xr-x   1 root     root       49604 Jul 21 05:18 install_mu
```

▼ How to Copy Information From a CD

You can access a CD’s files and directories just like any other file system. The only significant restrictions 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 “How to Load a CD” on page 181.

2. **Copy the files or directories.**

To Copy ...	Use ...
A file	`cp`
A directory	`cp -r`

Example—Copying Information From a CD

The following example uses `cp` to copy a single file from the `/cdrom/solstice_sysmgt_2_3` directory into the system’s working directory (denoted by the “`.`”).

```
$ cp /cdrom/solstice_sysmgt_2_3/README .
$ ls -l
-r--r--r--   1 pmorph   users       4618 May  9 08:09 README
```

Note that when a file or directory is copied from a CD into your file system, you become its owner, but it retains the permissions it had on the CD:

```
-r--r--r--
```

To overwrite it, you’ll need to change the permissions with the `chmod` command. See “Securing Files (Tasks)” in *System Administration Guide, Volume 2* for more information on using the `chmod` command.

▼ How to Find Out If a CD Is Still in Use

1. **Become superuser.**

2. **Identify the processes accessing the CD.**

The `fuser`(1M) command lists the processes that are currently accessing the CD that you specify.

```
# fuser -u [-k] /cdrom/cdrom0
```

–u	Displays the user of the CD.
–k	Kills the process accessing the CD.

The `fuser` command might not always identify all the killed processes. To be sure, run it again with the –u option.

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 superuser), 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
```

▼ 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 shell hidden behind a desktop tool might be accessing it), use the `fuser` command, as described in “How to Find Out If a CD Is Still in Use” on page 182.

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 shared its CD-ROM according to the instructions in “How to Make Local CDs Available to Other Systems” on page 185.

1. **Select an existing directory to serve as the mount point or create one.**

```
$ mkdir directory
```

directory	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.**

```
$ showmount -e system-name
export list for system-name:
/cdrom/sol_8_sparc (everyone)
```

3. **As superuser, mount the CD.**

```
# mount -F nfs -o ro system-name:/cdrom/cd-name local-mount-point
```

system-name	The name of the system whose CD you will mount.
cd-name	The name of the CD you want to mount.
local-mount-point	The local directory onto which you will mount the remote CD.

4. **Log out as superuser.**

5. **Verify that the CD is mounted by using 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 `sol_8_sparc` from the remote system `mars` onto the `/cdrom` directory of the local system.

```
$ showmount -e starbug
export list for starbug:
/cdrom/sol_8_sparc (everyone)
$ su
Password: password
# mount -F nfs -o ro starbug:/cdrom/sol_8_sparc /cdrom
# exit
$ ls /cdrom
cdrom0     sol_8_sparc
```

▼ 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 in those drives available to other systems. (This does not apply to musical CDs.) 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 184.

1. **Become superuser.**

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 on page 186
`nfsd` is *not* running	Continue with Step 4 on page 186

4. Create a dummy directory for `nfsd` to share.

```
# mkdir / dummy-dir
```

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.

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 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 the `/etc/rmmount.conf` file.

```
# chmod 644 /etc/rmmount.conf
```

10. Add the following lines to the `/etc/rmmount.conf` file.

```
# 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 the `/etc/rmmount.conf` file.

```
# 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 184.

13. Verify that the CD is indeed available to other systems by using 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"
-    /sol_7_sparc  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.

```
# ps -ef | grep nfsd
    root 10127  9986  0 08:25:01 pts/2    0:00 grep nfsd
    root 10118     1  0 08:24:39 ?        0:00 /usr/lib/nfs/nfsd -a
# mkdir /dummy
# vi /etc/dfs/dfstab
(Add the following line:)
share -F nfs -o ro  /dummy
# eject cdrom0
# chmod 644 /etc/rmmount.conf
# vi /etc/rmmount
(Add the following line to the File System Sharing section:)
share cdrom*
# chmod 444 /etc/rmmount.conf
(Load a CD.)
# share
-               /dummy   ro   ""
-               /cdrom/sol_7_sparc/s5   ro   ""
-               /cdrom/sol_7_sparc/s4   ro   ""
-               /cdrom/sol_7_sparc/s3   ro   ""
-               /cdrom/sol_7_sparc/s2   ro   ""
-               /cdrom/sol_7_sparc/s1   ro   ""
-               /cdrom/sol_7_sparc/s0   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 superuser.**

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
```

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

This example shows an `/etc/rmmount.conf` file modified to support the Workman software.

```
# @(#)rmmount.conf 1.3     96/05/10 SMI
#
# Removable Media Mounter configuration file.
#

# File system identification
ident hsfs ident_hsfs.so cdrom
ident ufs ident_ufs.so cdrom floppy rmscsi pcmem
ident pcfs ident_pcfs.so floppy rmscsi pcmem

# Actions
action cdrom action_workman.so /usr/dist/exe/workman
action cdrom action_filemgr.so
action floppy action_filemgr.so
action rmscsi action_filemgr.so

# File System Sharing
share cdrom*
share floppy*
```

▼ How to Prepare a System for a New CD-ROM Drive

Preparing the system involves creating the `/reconfigure` file and rebooting the system so that Volume Management recognizes the new CD-ROM drive.

1. **Become superuser.**

2. **Create a file called** `/reconfigure`**.**

```
# touch /reconfigure
```

3. **Reboot the system.**

```
# init 6
```

Configuring Volume Management

Occasionally, you might want to manage diskettes or CDs without the help of Volume Management. This section describes how to stop and restart Volume Management.

▼ 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 182.

2. **Become superuser.**

3. **Enter the `volmgt stop` command.**

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

▼ How to Restart Volume Management

1. **Become superuser.**

2. **Enter the `volmgt start` command.**

```
# /etc/init.d/volmgt start
volume management starting.
```

CHAPTER 16

Formatting and Using Diskettes From the Command Line (Tasks)

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.

Formatting Diskettes Task Map

TABLE 16–1 Formatting Diskettes Task Map

Task	Description	For Instructions, Go To ...
1. Load Unformatted Diskette	Insert the diskette into the drive and enter the `volcheck` command.	"How to Load a Diskette" on page 205
2. Format the Diskette	Format the diskette for UFS. Format the diskette for DOS.	"How to Format a UFS Diskette" on page 197 "How to Format a DOS Diskette" on page 201
3. Add a UFS File System	*UFS Only. Optional.* To use the diskette for files, add a UFS file system. To use for characters, skip this step.	"How to Place a UFS File System on a Diskette" on page 200
4. 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" on page 210

Using Diskette Names

When working with diskettes, you can identify them by name or with a designator from Table 16–2. For brevity, task descriptions use `floppy0`, but you can replace this with either the diskette name or a different designator.

TABLE 16–2 How to Identify Diskettes

Diskette	Alternate Name
First diskette drive	`floppy0`
Second diskette drive	`floppy1`
Third diskette drive	`floppy2`

Note - Diskettes that are not named (that is, they 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 ...
SPARC based systems	UFS
	MS-DOS or NEC-DOS (PCFS)
IA based systems	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 based platform cannot be used for UFS on an IA platform, nor can a diskette formatted on an IA platform be used on a SPARC based platform. This is because the SPARC and IA UFS formats are different. SPARC uses little-endian bit coding, IA 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 “Formatting Diskettes Task Map” on page 193.

On a Solaris system (either SPARC or IA), 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

Diskette Size	Diskette Density	Capacity
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 the table below.

TABLE 16–3 Density Options

To Format A Diskette With This Density ...	In A Drive With This Default Density ...	Specify This `fdformat` Density Option ...
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`

TABLE 16–3 Density Options *(continued)*

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 195, a UFS diskette formatted on a SPARC based platform can only be used on another SPARC based platform, and a UFS diskette formatted on an IA platform can only be used on an IA based 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. 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]
```

–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.
	Lists all the options to the `fdformat` command, but does not format the diskette..
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, upper or lower case.
–z	Lists all the options to the `fdformat` command, but does not format the diskette.

Note - If you try to format a 720 Kbyte (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 Vs appears beneath the dots. When the series stops, the formatting is complete.

The diskette is now ready for raw character operations such as `tar` and `cpio`.

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 ]
.......................................................
vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
```

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 ]
.......................................................
vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
```

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 ]
.......................................................
vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
```

▼ How to 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 IA platforms and SPARC platforms, a UFS diskette formatted on a SPARC platform can only be used on another SPARC platform, and a UFS diskette formatted on an IA platform can only be used on an IA platform.

1. **Format the diskette for a UFS file system.**

   Use "How to Format a UFS Diskette" on page 197.

2. **Create a SunOS file system on the diskette.**

```
$ /usr/sbin/newfs -v /vol/dev/aliases/floppy0
```

`–v`	Prints status messages.
`/vol/dev/aliases/floppy0`	Indicates the location of the floppy.

   The `newfs`(1M) command displays a message asking you to confirm the creation of the file system.

3. **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.

   The diskette is now ready to be used on a SPARC platform. However, before Volume Management recognizes it, you must run the `volrmmount` command, as described in the following steps.

4. **Invoke the `volrmmount` command using the `–i` option to notify Volume Management that the diskette is inserted.**

```
$ volrmmount -i floppy0
```

5. **Verify that the UFS file system is on the diskette by using the `ls` command on the `/floppy` directory.**

If the `floppy0` subdirectory appears, the diskette has a UFS file system and has been mounted properly.

```
$ ls /floppy
floppy0
```

Example—Placing a UFS File System on a Diskette

```
$ volcheck -v
media was found
$ /usr/sbin/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 2880 18 2 8192 1024 16 10 5 2048
t 0 -1 8 15
/vol/dev/aliases/floppy0:  2880 sectors in 80 cylinders of 2 tracks,
18 sectors
        1.4MB 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,
$ volrmmount -i floppy0
$ ls /floppy
floppy0
```

▼ How to Format a DOS Diskette

You can format a DOS diskette on a SPARC or IA platform. 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 diskette.

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. 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]
```

–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.
	Lists all the options to the `fdformat` command, but does not format the diskette.
convenience-options	
–e	Ejects the diskette when done formatting.
–f	Does not ask for confirmation before formatting.
–b *label*	Name for the diskette. *label* must be eight characters or less, upper or lower case.
–z	Lists all the options to the `fdformat` command, but does not format the diskette.

Note - If you try to format a 720 Kbyte (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 Vs appears beneath the dots. When the series stops, the formatting is complete and the diskette is ready for use on a DOS system.

6. Run the `volrmmount`(1) command using the –`i` option to notify Volume Management that the diskette is inserted.

```
$ volrmmount -i floppy0
```

Volume Management mounts the diskette under `/floppy/floppy0`.

Example—Formatting a DOS Diskette

The following example formats a 1.44 Mbyte MS-DOS diskette and 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 ]
....................................................
vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
```

Using Diskettes Task Map

TABLE 16–4 Using Diskettes Task Map

Task	Description	For Instructions, Go To
1. Load the Diskette	Insert the diskette into its drive and enter the `volcheck` command.	"How to Load a Diskette" on page 205
2. 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" on page 207
3. Exchange Files	*Optional.* Copy files or directories between the diskette and your file system.	"How to Copy or Move Information From a Diskette" on page 207 "How to Copy or Move Information to a Diskette" on page 208
4. Is Diskette Still in Use?	*Optional.* Before ejecting the diskette, find out if the diskette is still in use.	"How to Find Out If a Diskette Is Still in Use" on page 209
5. Eject the Diskette	When you finish, eject the diskette from its drive.	"How to Eject a Diskette" on page 210

▼ 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 on page 205. If you need to format the diskette, go to "How to Format a UFS Diskette" on page 197 or "How to Format a DOS Diskette" on page 201.

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 "How to Format a UFS Diskette" on page 197 (for UFS) and "How to Format a DOS Diskette" on page 201 (for DOS).

`no media was found`	Volume Management did not detect a diskette. Make sure the diskette is inserted properly and run `volcheck`(1) again. If unsuccessful, check the diskette; it could be damaged. You can also try to mount the diskette manually.

4. Verify that the diskette was mounted by listing its contents.

```
$ ls /floppy
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 `ls -L` command because some directories under `/floppy` are symbolic links.

```
$ ls -L [-l] floppy0
```

–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
```

▼ How to Copy or Move Information From a Diskette

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 significant restrictions 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 that file on the diskette. Or, if you copy a file into your file system, you'll be the owner, but that file won't have write permissions (because it never had them on the diskette); you'll have to change the permissions yourself.

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 "How to Load a Diskette" on page 205. The diskette might also need to be

formatted. See “How to Format a UFS Diskette” on page 197 or “How to Format a DOS Diskette” on page 201.

2. **Copy the files or directories.**

To Copy ...	Use ...
A file	`cp`
A directory	`cp -r`

3. **Verify the copy or move operation by using 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 "How to Load a Diskette" on page 205. The diskette might also need to be formatted. See "How to Format a UFS Diskette" on page 197 or "How to Format a DOS Diskette" on page 201.

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`

4. Verify a move or copy operation by using 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 readme2.doc /floppy/floppy1
$ cp -r /home/smith/morefiles /floppy/floppy0
```

▼ How to Find Out If a Diskette Is Still in Use

1. Become superuser.

2. **Invoke the `fuser` command.**

   The `fuser` command lists the processes that are currently accessing the CD that you specify.

```
# fuser -u [-k] floppy0
```

–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 superuser), or you can use the `fuser` command with the –k option, which kills all the processes accessing that file system. The `fuser` command might not always identify all the killed processes. To be sure, run it again with the –u option.

```
# fuser -u -k /floppy/floppy0
/floppy/floppy0: 6400c(root)Killed  6399c(smith)Killed
```

▼ 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 might be accessing it), use the `fuser` command, as described in “How to Find Out If a Diskette Is Still in Use” on page 209.

2. **Eject the diskette.**

```
# eject floppy0
```

On a SPARC based system the floppy is physically ejected from its drive, but on an IA based 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 shared its diskette drive according to the instructions in "How to Make Local Diskettes Available to Other Systems" on page 212.

1. **Select an existing directory to serve as the mount point, or create one.**

```
$ mkdir directory
```

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 superuser, mount the diskette.**

```
# mount -F nfs system-name:/floppy/diskette-name local-mount-point
```

system-name	The name of the system whose diskette you will mount.
diskette-name	The name of the diskette you want to mount.
local-mount-point	The local directory onto which you will mount the remote diskette.

4. **Log out as superuser.**

5. **Verify that the diskette is mounted by using 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 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 211.

1. **Become superuser.**

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 on page 214
`nfsd` is *not* running	Continue with Step 4 on page 213

4. Create a dummy directory for `nfsd` to share.

```
# mkdir /dummy-dir
```

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 211.

13. Verify that the diskette is available to other systems by using the `share`**(1M) 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.

```
# ps -ef | grep nfsd
    root 10127  9986  0 08:25:01 pts/2    0:00 grep nfsd
    root 10118     1  0 08:24:39 ?        0:00 /usr/lib/nfs/nfsd -a
# mkdir /dummy
# vi /etc/dfs/dfstab
(Add the following line:)
share -F nfs -o ro  /dummy
# eject floppy0
# chmod 644 /etc/rmmount.conf
# vi /etc/rmmount
(Add the following line to the File System Sharing section.)
share floppy*
# chmod 444 /etc/rmmount.conf
(Load a diskette.)
# volcheck -v
media was found
# share
-               /dummy   ro   ""
-               /floppy/myfiles   rw   ""
```

CHAPTER 17

Using PCMCIA Memory Cards From the Command Line (Tasks)

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.

Formatting PCMCIA Memory Cards Task Map

TABLE 17–1 Formatting PCMCIA Memory Cards Task Map

Task	Description	For Instructions, Go To ...
1. 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" on page 227
2. Format the PCMCIA Memory Card	Format the PCMCIA memory card for UFS.	"How to Format a UFS PCMCIA Memory Card" on page 219
	Format the PCMCIA memory card for DOS.	"How to Format a DOS PCMCIA Memory Card" on page 224
3. 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" on page 222
4. Eject the PCMCIA Memory Card	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" on page 232

Using PCMCIA Memory Cards Names

When working with PCMCIA memory cards, you can identify them by name or with a designator from the table below. For brevity, task descriptions use `pcmem0`, but you can replace this with either the PCMCIA memory card's name or a different designator.

TABLE 17–2 How to Identify PCMCIA Memory Cards

PCMCIA Card	Alternate Name
First PCMCIA drive	`pcmem0`
Second PCMCIA drive	`pcmem1`
Third PCMCIA drive	`pcmem2`

TABLE 17–2 How to Identify PCMCIA Memory Cards *(continued)*

Note - PCATA drives that are not named (that is, they have no “label”) are assigned the default name of `noname`.

Hardware Considerations

A Solaris platform can format PCMCIA memory cards for use on both Solaris and DOS platforms. However, the hardware platform imposes some limitations. They are summarized in the table below.

Solaris On This Platform ...	Can Format PCMCIA Memory Cards For ...
SPARC based systems	UFS
	MS-DOS or NEC-DOS (PCFS)
IA based systems	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 platform cannot be used for UFS on an IA platform. Likewise, PCMCIA memory cards formatted on an IA platform cannot be used on a SPARC platform. This is because the SPARC and IA 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 file system you are using. See “Formatting PCMCIA Memory Cards Task Map” on page 217.

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.

▼ How to Format a UFS PCMCIA Memory Card

As mentioned in the introduction, a UFS PCMCIA memory card formatted on a SPARC based platform can be used only on another SPARC based platform, and a

UFS PCMCIA memory card formatted on an IA platform can be used only on an IA platform running the Solaris *Intel Platform Edition*.

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. 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]
```

–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, upper or lower case.
–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 Vs appears beneath the dots. When the series stops, the formatting is complete.

The PCMCIA memory card is now ready for raw character operations such as `tar` and `cpio`.

Examples—Formatting a UFS PCMCIA Memory Card

Following are examples of UFS formatting.

```
$ fdformat -v -U
Formatting in /vol/dev/aliases/unformatted
Press return to start formatting pcmem0. [ Return ]
.........................................................
vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
```

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 ]
.........................................................
vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
```

▼ 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 IA platforms and SPARC based platforms, a UFS PCMCIA memory card formatted on a SPARC based platform can only be used on another SPARC based platform, and a UFS PCMCIA memory card formatted on an IA platform can only be used on a IA platform.

1. **Format the PCMCIA memory card for a UFS file system.**

   Use the procedure “How to Format a UFS PCMCIA Memory Card” on page 219.

2. **Use the `newfs`(1M) command and the full pathname to the Volume Management directory to create a UFS file system on the PCMCIA memory card.**

```
$ /usr/sbin/newfs -v /vol/dev/aliases/pcmem0
```

`–v`	Prints status messages.
`/vol/dev/aliases/pcmem0`	Indicates the location of the memory card.

   The `newfs`(1M) command displays a message asking you to confirm the creation of the file system.

3. **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 2848 8 2 8192 1024 16 \
10 60 2048 t 0 -1 8 -1
/vol/dev/aliases/pcmem0:  2848 sectors in 128 cylinders of \
2 tracks, 8 sectors
1.0MB in 8 cyl groups (16 c/g, 0.12MB/g, 64 i/g)
super-block backups (for fsck -F ufs -o b=#) at:
```

(continued)

(Continuation)

```
32, 304, 544, 816, 1056, 1328, 1568, 1840
```

The PCMCIA memory card is now ready to be used on a SPARC based platform. However, before Volume Management recognizes the memory card, you must use the `volrmmount`(1) command as described in the following step.

4. Use the `volrmmount` command with the `–i` option to notify Volume Management that the memory card is inserted.

```
$ volrmmount -i pcmem0
```

The PCMCIA memory card should now be mounted under `/pcmem/pcmem0`.

5. Verify the UFS file system is on the PCMCIA card by using the `ls` command on the `/pcmem` directory.

If the `pcmem0` subdirectory appears, the PCMCIA memory card has a UFS file system and has been mounted properly.

```
$ ls /pcmem
pcmem0
```

Example—Placing a UFS File System on a PCMCIA Memory Card

```
$ volcheck -v
media was found
$ /usr/sbin/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 ...

$ volrmmount -i pcmem0

media was found
```

▼ How to Format a DOS PCMCIA Memory Card

You can format a DOS PCMCIA memory card on a SPARC or IA based platform. 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. 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]
```

`–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 for MS-DOS.
`–t nec –M`	Formats at 1.2 Mbytes for NEC-DOS.
	Lists all the options to the `fdformat` command, but does not format the PCMCIA memory card.
convenience-options	

`–e`	Ejects the PCMCIA memory card when done formatting.
`–f`	Does not ask for confirmation before formatting.
`–b` *label*	Name for the PCMCIA memory card. *label* must be eight characters or less, upper or lower case.
`–z`	Lists all the options to the `fdformat` command, but does not format the PCMCIA memory card.

Note - If you try to format a 720 Kbyte (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 Vs 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. **Use the `volrmmount` command with the `-i` option to notify Volume Management that the memory card is inserted.**

```
$ volrmmount -i pcmem0
```

Volume Management mounts the PCMCIA memory card under `/pcmem/pcmem0`.

Using PCMCIA Memory Cards Task Map

TABLE 17–3 Using PCMCIA Memory Cards Task Map

Task	Description	For Instructions, Go To
1. 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" on page 227
2. Examine 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 Examine the Contents of a PCMCIA Memory Card" on page 229
3. Exchange Files	*Optional.* Copy files or directories between the PCMCIA memory card and your file system.	"How to Copy or Move Information From a PCMCIA Memory Card" on page 229 "How to Copy or Move Information to a PCMCIA Memory Card" on page 230

TABLE 17–3 Using PCMCIA Memory Cards Task Map *(continued)*

Task	Description	For Instructions, Go To
4. Is PCMCIA Memory Card Still in Use?	*Optional.* Before ejecting the PCMCIA memory card, find out if the PCMCIA memory card is still in use.	"How to Find Out If a PCMCIA Memory Card Is Still In Use" on page 232
5. Eject the PCMCIA Memory Card	When you finish, eject the PCMCIA memory card.	"How to Eject a PCMCIA Memory Card" on page 232

▼ 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 "Using PCMCIA Memory Cards Task Map" on page 226. If you need to format the PCMCIA memory card, go to "How to Format a UFS PCMCIA Memory Card" on page 219 or "How to Format a DOS PCMCIA Memory Card" on page 224.

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 "How to Format a UFS PCMCIA Memory Card" on page 219 (for UFS) and "How to Format a DOS PCMCIA Memory Card" on page 224 (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.

4. 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 Examine the Contents of a PCMCIA Memory Card

Use the `ls -L` command because some directories under `/pcmem` are symbolic links:

```
$ ls -L [-l] pcmem0
```

`–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
```

▼ How to Copy or Move Information From a PCMCIA Memory Card

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 significant restrictions 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 that file on the PCMCIA memory card. Or, if you copy a file into your file system, you'll be the owner, but that file won't have write permissions (because it never had them on the PCMCIA memory card); you'll have to change the permissions yourself.

1. Make sure the PCMCIA memory card is formatted and mounted.

```
$ ls /pcmem
pcmem0  PCMCIAmemorycard-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 227. The PCMCIA memory card might also need to be formatted. See "How to Format a UFS PCMCIA Memory Card" on page 219 or "How to Format a DOS PCMCIA Memory Card" on page 224.

2. **Copy the files or directories.**

To Copy ...	Use ...
A file	`cp`
A directory	`cp -r`

3. **Verify the copy or move operation by using 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   PCMCIAmemory-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 227. The PCMCIA memory card might also need to be formatted. See "How to Format a UFS PCMCIA Memory Card" on page 219 or "How to Format a DOS PCMCIA Memory Card" on page 224.

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`

4. Verify the move or copy operation by using 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 readme2.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 superuser.**

2. **Invoke the `fuser`(1M) command.**

   The `fuser` command lists the processes that are currently accessing the CD that you specify.

```
# fuser -u [-k] pcmem0
```

–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 superuser), 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 might 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 might 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 232.

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 235.

1. **Select an existing directory to serve as the mount point or create one.**

```
$ mkdir directory
```

directory	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 superuser, mount the PCMCIA memory card.**

```
# mount -F nfs system-name:/pcmem/PCMCIAmemory-card-name local-mount-point
```

system-name	The name of the system whose PCMCIA memory card you want to mount.
PCMCIAmemory-card-name	The name of the PCMCIA memory card you want to mount.
local-mount-point	The local directory onto which you will mount the remote PCMCIA memory card.

4. **Log out as superuser.**

5. **Verify that the PCMCIA memory card is indeed mounted by using 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 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 233.

1. **Become superuser.**

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 on page 236
`nfsd` is *not* running	Continue with Step 4 on page 235

4. **Create a dummy directory for `nfsd` to share.**

```
# mkdir /dummy-dir
```

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 233.

13. **Verify that the PCMCIA memory card is indeed available to other systems by using 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.

```
# ps -ef | grep nfsd
    root 10127  9986  0 08:25:01 pts/2    0:00 grep nfsd
    root 10118     1  0 08:24:39 ?        0:00 /usr/lib/nfs/nfsd -a
# mkdir /dummy
# vi /etc/dfs/dfstab
(Add the following line:)
share -F nfs -o ro  /dummy
# eject pcmem0
# chmod 644 /etc/rmmount.conf
# vi /etc/rmmount
(Add the following line to the File System Sharing section:)
share floppy*
# chmod 444 /etc/rmmount.conf
(Load a PCMCIA memory card.)
# volcheck -v
media was found
# share
-               /dummy   ro   ""
```

(continued)

(Continuation)

```
-               /pcmem/myfiles   rw   ""
```

CHAPTER 18

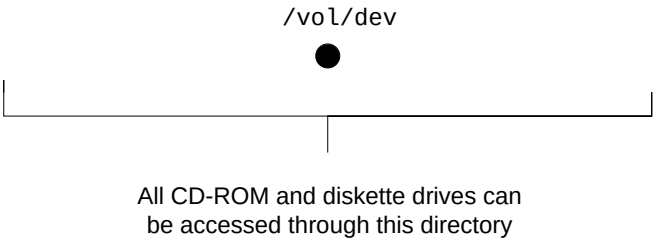
How Volume Management Works (Reference)

This chapter describes the mount points and symbolic links that Volume Management creates to accommodate removable media.

This is a list of reference information in this chapter.

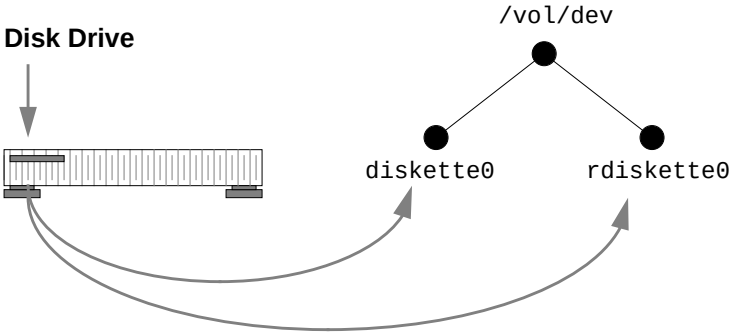
Volume Management Mounts All Removable Media

Volume Management provides access to all CD-ROM and diskette drives under `/vol/dev`:



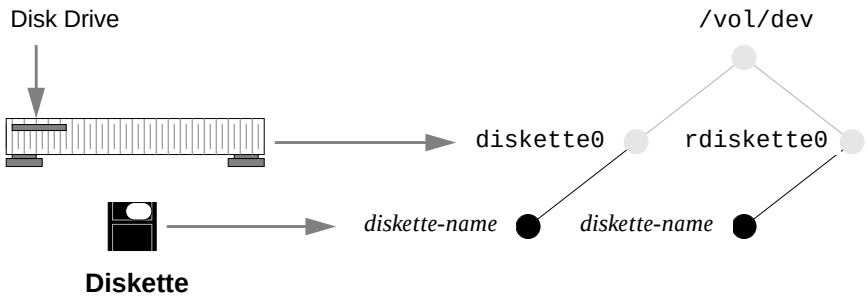
Volume Management Provides Access to Diskettes

Volume Management provides access to a system's diskette drive through subdirectories of `/vol/dev`; namely, `diskette0` and `rdiskette0`.



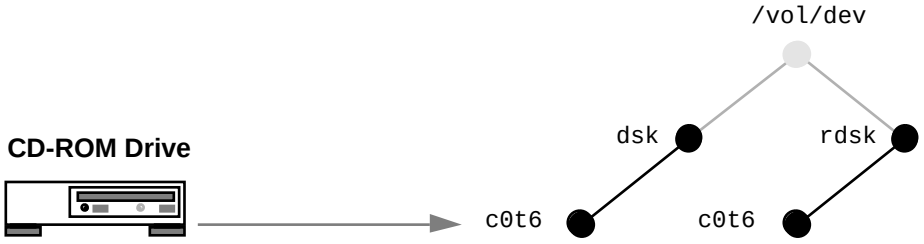
If a system has a second diskette drive, Volume Management creates a second pair of directories named `diskette1` and `rdiskette1`. For a third diskette drive, it would create `diskette2` and `rdiskette2`; and so on for additional drives.

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. (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.)



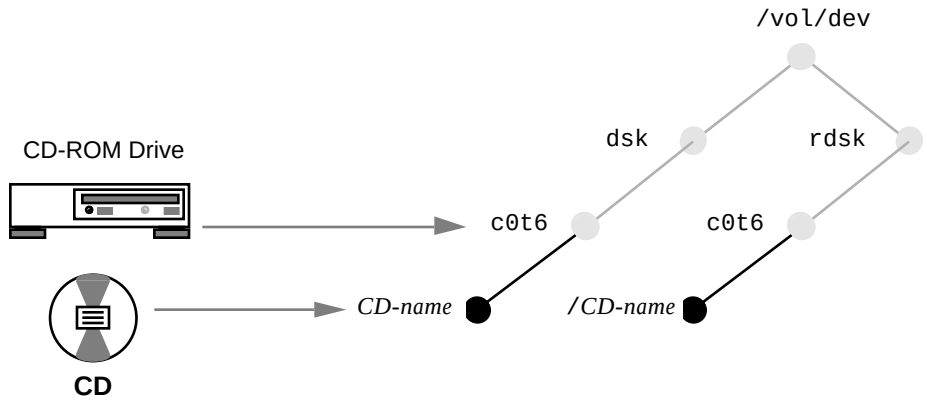
Volume Management Provides Access to CDs

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.

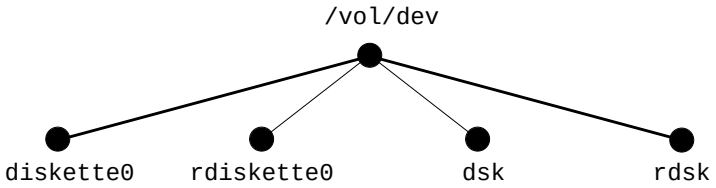


In the illustration above, the additional directory is named `c0t6`. That simply reflects a particular system's device naming conventions. The directory name on your system could be different, though it would have the same format.

The CDs themselves, however, follow a convention similar to diskettes, in that they are mounted beneath the directory belonging to their device:



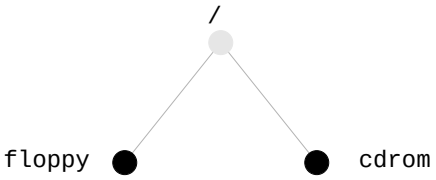
As a result of this arrangement, a system with one diskette drive and one CD-ROM drive would have the following `/vol/dev` file system:



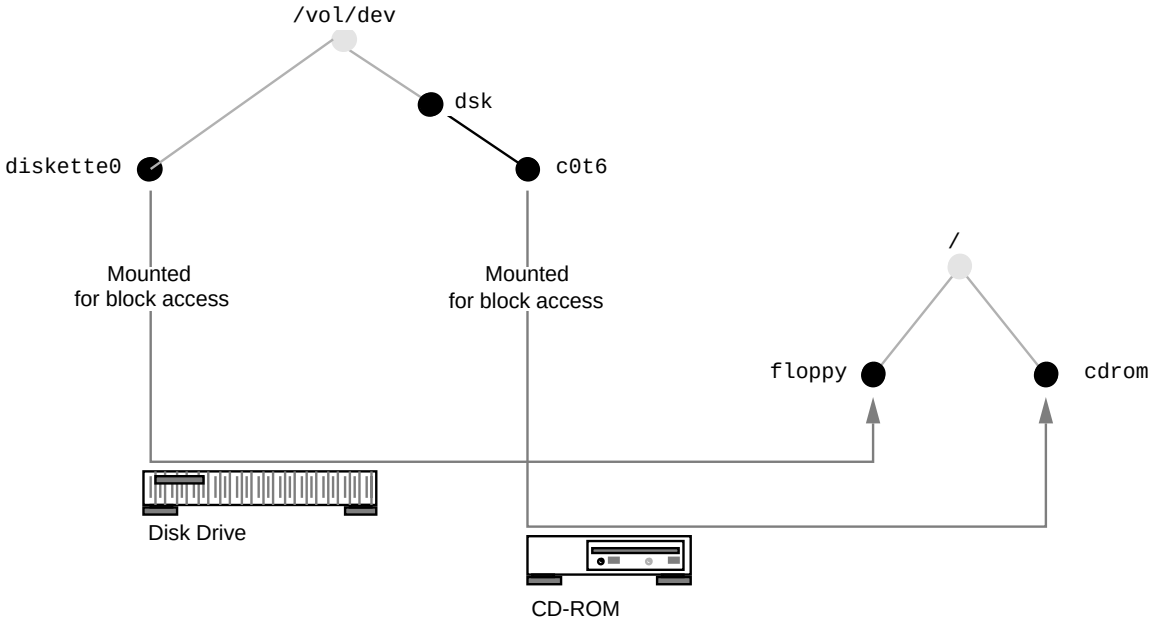
(Actually, `/vol/dev` includes an additional subdirectory named `aliases`, but that is described later in this section.)

Volume Management Supplies Convenient Mount Points for Easier Access

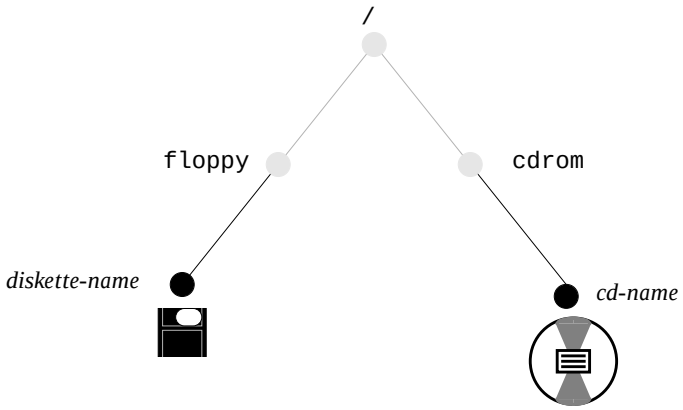
To make access more convenient, Volume Management uses two special mount points, `/floppy` and `/cdrom`.



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 “How to Format a UFS Diskette” on page 197 or “How to Format a DOS Diskette” on page 201.

If a system has multiple drives, they are mounted onto parallel directories such as `/floppy/floppy0`, `/floppy/floppy1`, `/cdrom/cdrom0`, etc.

Volume Management 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`*

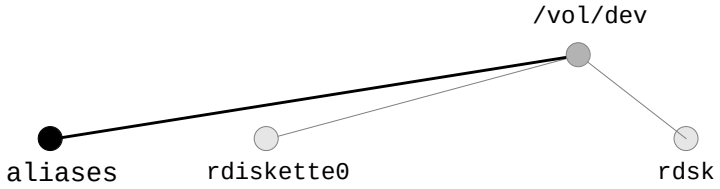
`/cdrom/cdrom0 --> /cdrom/`*`cd-name`* `--> /vol/dev/dsk/c0t6d0/`*`cd-name`*

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 14–3.

Symbolic Links for Raw Device Access

To make raw device access more convenient, Volume Management creates the `aliases` directory, under `/vol/dev`:



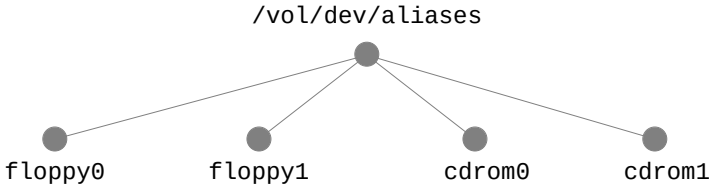
Beneath the `aliases` directory, Volume Management creates a set of symbolic links similar to those used for block access. In other words, for character access, these directories are equivalent:

`/vol/dev/aliases/floppy0 --> /vol/dev/rdiskette0/`*`diskette-name`*

`/vol/dev/aliases/cdrom0 --> /vol/dev/rdsk/c0t6d0/`*`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:



Volume Management 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 platform cannot be recognized by an IA platform. Likewise, an IA UFS CD cannot be mounted by Volume Management on a SPARC platform. The same limitation applies to diskettes. (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).

Therefore, Volume Management cannot recognize and mount IA UFS media on a SPARC platform—or SPARC UFS media on an IA platform.

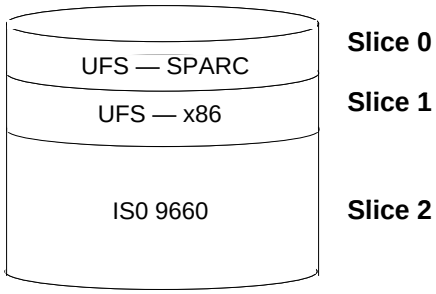
Most CDs are formatted according to the ISO 9660 standard (High Sierra File System—HSFS), which imposes no limitations on Volume Management, so incompatibility is seldom a problem with CDs.

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 197).

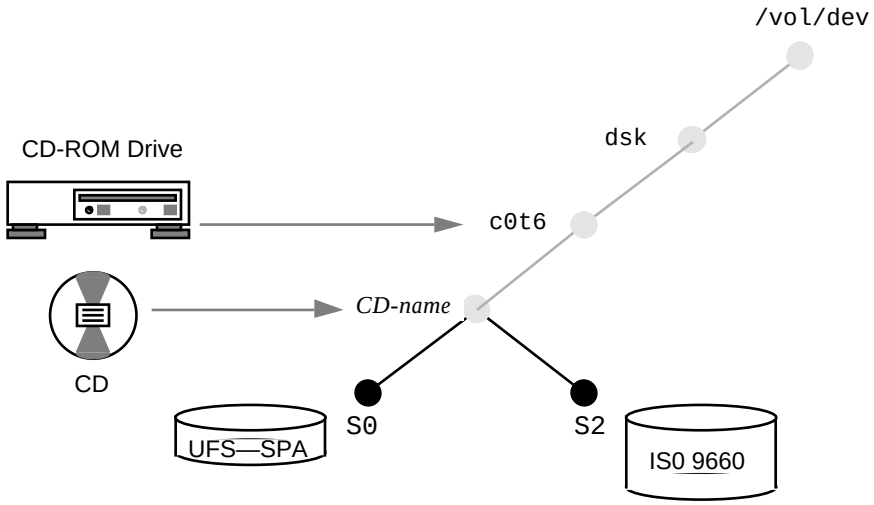
What About Mixed Formats?

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
```

CHAPTER 19

Managing Software Topics

This section provides instructions for managing Solaris software packages and patches. This section contains these chapters.

Chapter 20	Provides overview information about adding and removing software products in the Solaris operating environment.
Chapter 21	Provides step-by-step instructions for installing and removing software packages on different client types.
Chapter 22	Provides overview information and step-by-step instructions about adding and removing patches in the Solaris operating environment.

CHAPTER 20

Software Administration (Overview)

Software administration involves adding 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.

Where to Find Software Administration Tasks

Use this table to find step-by-step instructions for administering software.

For Information On ...	See ...
Installing Solaris software	*Solaris 8 (SPARC Platform Edition) Installation Guide* or *Solaris 8 (Intel Platform Edition) Installation Guide*
Adding or removing Solaris software packages after installation	Chapter 21
Adding or removing Solaris patches after installation	Chapter 22
Troubleshooting software administration problems	"Troubleshooting Software Administration Problems" in *System Administration Guide, Volume 2*

What's New in Software Management?

This Solaris release includes a new software management tool, Solaris Product Registry. Once the Solaris 8 release is installed, Product Registry provides a list of all software installed using Solaris Web Start 3.0 or the Solaris package management commands. To view the Product Registry, type `/usr/bin/prodreg` at the command line.

See "Adding and Removing Software With the Product Registry" in *Solaris 8 (SPARC Platform Edition) Installation Guide* or "Adding and Removing Software After Installing Solaris 8" in *Solaris 8 (Intel Platform Edition) Installation Guide* for more information.

Overview of Software Packages

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 (ABI), which is a supplement to the System V Interface Definition. The Solaris operating environment

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 three tools for adding and removing software from a system after the Solaris release is installed on a system:

This Tool Enables You To Add, Remove, And Display Package Information...	**And You Can ...**
Package commands (`pkgadd`, `pkgrm`, `pkginfo`)	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 these commands, it's probably easiest for you to continue using them.
Admintool	View the online help that provides general information on using this graphical interface tool. 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.
Solaris Product Registry	Launch an installer to add products.

The table below describes the advantages of using Admintool rather than the `pkgadd` and `pkgrm` commands to manage software.

TABLE 20–1 Admintool Software Management Capabilities

Software Management Tasks	**Performed With Admintool?**
Add and remove packages on standalone or server systems	Yes
Easily view all installed software	Yes

TABLE 20–1 Admintool Software Management Capabilities *(continued)*

Software Management Tasks	Performed With Admintool?
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

In previous Solaris releases, Software Manager (accessed with the `swmtool` command) was the graphical tool for adding and removing software. If you use the `swmtool` command on a Solaris 2.5 or compatible system, it will start Admintool.

What Happens When You Add or Remove a Package

The `pkgadd` and `pkgrm` commands or Admintool are used to add and remove software. Admintool is a graphical front-end to the `pkgadd` and `pkgrm` commands.

When you add a package, the `pkgadd` command uncompresses and copies files from the installation media to a local system's disk. When you remove a package, the `pkgrm` command deletes all files associated with that package, unless those files are also shared with other packages.

Package files are delivered in package format and 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. The `pkgadd` and `pkgrm` commands 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`, and `SUNWab2m`. Third-party packages usually begin with a prefix that corresponds to the company's stock symbol.
- What software is already installed – You can use the product registry, Admintool, or the `pkginfo` command to determine the software already installed on a system.
- How servers and clients share software – Clients might 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. (The section below 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 client systems can be more difficult—especially when the software resides partially on the server and partially on the client. (For example, a piece of software might 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.)

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 the `rm` command 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 the `rm` command 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 `removef`(1M) 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 base directory, and you can manipulate the base directory by setting the `basedir` keyword in a special file called an administration file. See "Avoiding User Interaction When Adding Packages" on page 254 and `admin`(4) 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.

Avoiding User Interaction When Adding Packages

Using an Administration File

When you use the `pkgadd -a` command, the `pkgadd` command 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.

This is an example of an 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
```

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 the `pkgrm` command.

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. See `admin`(4) for more information.

Using a Response File

A response file contains your answers to specific questions asked by an *interactive package.* An interactive package includes a `request` script that asks you questions prior to package installation, such as whether or not optional pieces of the package should be installed.

If you know that the package you want to install is an interactive package, prior to installation, and you want to store your answers to prevent user interaction during future installations of this package, you can use the `pkgask` command to save your response. See `pkgask`(1M) for more information on this command.

Once you have stored your responses to the questions asked by the `request` script, you can use the `pkgadd -r` command to install the package without user interaction.

CHAPTER 21

Software Administration (Tasks)

This chapter describes how to install, remove, and administer software packages with Solaris commands and the Admintool graphical interface.

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

Commands for Handling Software Packages

The table below shows commands to use for adding, removing, and checking the installation of software packages after the Solaris release is installed.

TABLE 21–1 Commands for Adding and Removing Packages

Command	Description
prodreg.1	Installs a software package with an installer
`pkgadd`(1M)	Installs a software package
`pkgrm`(1M)	Removes a software package
`pkgchk`(1M)	Checks the installation of a software package
`pkginfo`(1)	Lists software package information
`pkgparam`(1)	Displays software package parameter values

Known Problem With Adding and Removing Packages

There is a known problem with adding or removing some packages developed before the Solaris 2.5 release. If adding or removing a 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
```

Adding Packages

▼ How to Add Packages to a Standalone System

1. **Log in as superuser.**

2. **Remove any already installed packages with the same names as the ones you are adding.**

This ensures that the system keeps a proper record of software that has been added and removed. There might be times when you want to maintain multiple versions of the same application on the system. For strategies on how to do this, see “Guidelines for Removing Packages” on page 253, and for task information, see “How to Remove a Package” on page 266.

3. Add a software package to the system.

```
# pkgadd -a admin-file -d device-name pkgid ...
```

–a *admin-file*	(Optional) Specifies an administration file `pkgadd` should consult during the installation. (For details about using an administration file, see “Using an Administration File” on page 254 in the previous chapter.)
–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, the `pkgadd` command 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, the `pkgadd` command installs 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.

4. Verify that the package has been installed successfully, using 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 a Mounted CD

The following example shows a command to install the `SUNWaudio` package from a mounted Solaris 8 CD. The example also shows use of the `pkgchk` command to verify that the package files were installed properly.

```
# pkgadd -d /cdrom/sol_8_sparc/s0/Solaris_8/Product SUNWaudio
 .
 .
 .
Installation of <SUNWaudio> was successful.
# pkgchk -v SUNWaudio
/usr
/usr/bin
/usr/bin/audioconvert
/usr/bin/audioplay
/usr/bin/audiorecord
```

Example—Installing Software From a Remote Package Server

If the packages you want to install are available from a remote system, you can manually mount the directory containing the packages (in package format) 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 “Avoiding User Interaction When Adding Packages” on page 254. 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 the pkgadd command. The pkgadd command, by default, looks in the /var/spool/pkg directory for any packages specified on the command line. Note that copying packages to a spool directory is not the same as installing the packages on a system.

▼ How to Add a Package to a Spool Directory

1. **Log in as superuser to the server or standalone system.**

2. **Remove any already spooled packages with the same names as the ones you are adding.**

   For information on removing spooled packages, see “How to Remove a Spooled Package” on page 267.

3. **Add a software package to a spool directory.**

   ```
   # pkgadd -d device-name -s spooldir pkgid . . .
   ```

–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.
–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` copies all available packages.

4. Verify that the package has been copied successfully to the spool directory, using the `pkginfo` command.

```
$ pkginfo -d spooldir| grep pkgid
```

If *pkgid* is copied correctly, 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 `SUNWab2m` packages from a mounted SPARC Solaris 8 CD to the default spool directory (`/var/spool/pkg`).

```
# pkgadd -d /cdrom/sol_8_sparc/s0/Solaris_8/Product -s /var/spool/pkg SUNWaudio
Transferring <SUNWaudio> package instance
```

Example—Setting Up a Spool Directory From a Remote Package Server

If packages you want to copy are available from a remote system, you can manually mount the directory containing the packages (in package format) and copy them 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.
```

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 `pkgchk`(1M) 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 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 or server. The output shows the primary category, package name, and a description of the package.

```
$ pkginfo
system      SUNWaccr       System Accounting, (Root)
system      SUNWaccu       System Accounting, (Usr)
system      SUNWadmap      System administration applications
system      SUNWadmc       System administration core libraries
.
.
.
```

▼ How to Check the Integrity of an Installed Package

1. Log in to a system as superuser.

2. Check the status of an installed package with the pkgchk **command.**

```
# pkgchk [ -a -c -v ] pkgid ...
# pkgchk -dspooldir pkgid ...
```

–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 Display Detailed Information About a Package

List information about installed packages with the pkginfo -l command.

```
$ pkginfo -l pkgid ...
```

–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.sun4u
   VERSION:  11.8.0,REV=1999.09.18.11.52
   BASEDIR:  /
    VENDOR:  Sun Microsystems, Inc.
      DESC:  core software for a specific hardware platform group
    PSTAMP:  humbolt19990821191439
  INSTDATE:  Sep 18 1999 11:53
   HOTLINE:  Please contact your local service provider
    STATUS:  completely installed
     FILES:     95 installed pathnames
                31 shared pathnames
                35 directories
                49 executables
             11307 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` command, which will corrupt the system's record-keeping of installed packages.

▼ How to Remove a Package

1. **Log in to the system as superuser.**

2. **Remove an installed package.**

```
# pkgrm pkgid ...
```

pkgid	(Optional) Is the name of one or more packages (separated by spaces). If omitted, `pkgrm` removes all available packages.

▼ How to Remove a Spooled Package

1. **Log in as superuser.**

2. **Remove a package from a spool directory with the** `pkgrm -s` **command.**

```
# pkgrm -s spooldir pkgid ...
```

`–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` removes all available packages.

Adding and Removing Packages Using Admintool

The Solaris operating environment includes 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 Add Packages With Admintool

1. **Log in to the installed system and become superuser.**

   At the shell prompt, type:

```
$ su
```

Unless you are a member of the UNIX sysadmin group (group 14), you must become superuser on your system to add or remove software packages with Admintool.

2. **Load a CD into the CD-ROM drive.**

   Volume Manager will automatically mount the CD.

3. **Start Admintool.**

```
# admintool &
```

   The Users window is displayed.

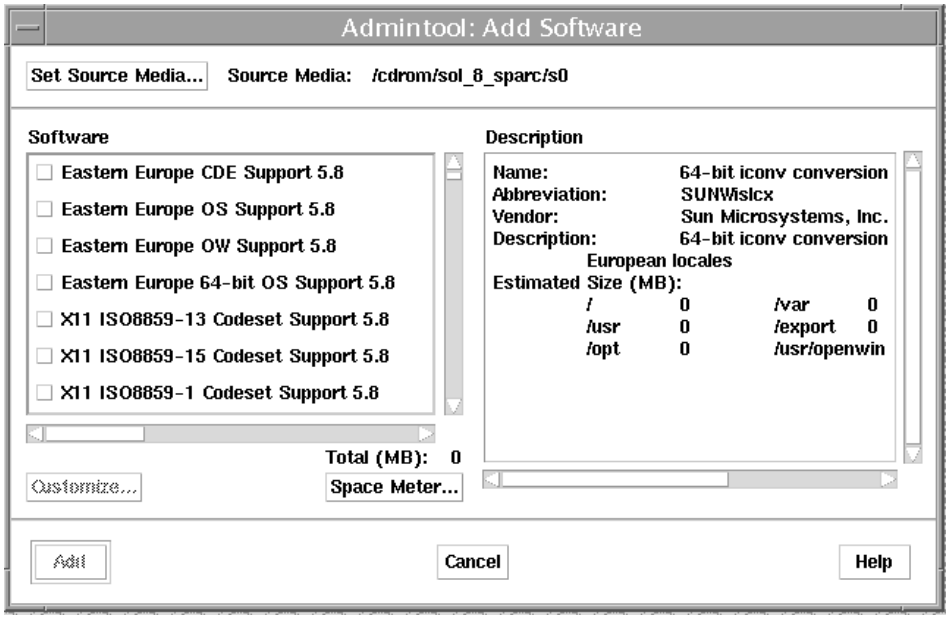
4. **Choose Software from the Browse menu.**

   The Software window is displayed.

5. **Choose Add from the Edit menu.**

   The Set Source Media window might appear. If so, specify the path to the installation media and click OK. The default path is a mounted SPARC Solaris CD.

   The Add Software window is displayed.



6. **Select the software you want to install on the local system.**

   In the Software portion of the window, click the check boxes corresponding to the software you want to install.

7. **Click Add.**

   A Command Tool window appears for each package being installed, displaying the installation output.

   The Software window refreshes to display the packages just added.

▼ How to Remove Packages With Admintool

1. **Log in to the installed system and become superuser.**

   At the shell prompt, type:

```
$ su
```

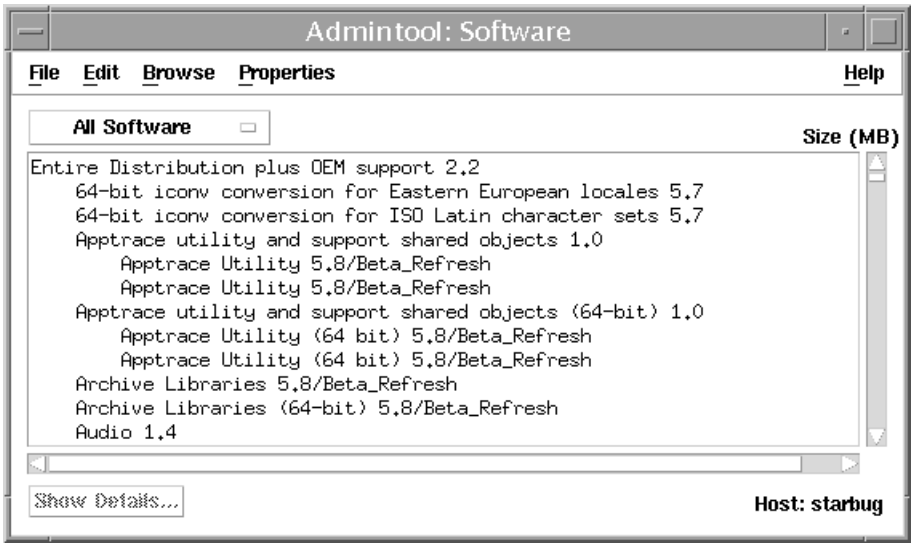
   Unless you are a member of the UNIX sysadmin group (group 14), you must become superuser on your system to add or remove software packages with Admintool.

2. **Start Admintool.**

```
# admintool &
```

3. **Choose Software from the Browse menu.**

   The Software window is displayed.



4. **Select the software you want to remove from the local system.**

5. **Choose Delete from the Edit menu.**

   A warning pop-up window is displayed to confirm whether you really want to delete the software.

6. **Click Delete to confirm that you want to remove the software.**

   For each package that is being deleted, a Command Tool window is displayed that asks for confirmation, again, before deleting the software. Type y, n, or q. If you choose to delete the software, the output from the removal process is displayed.

CHAPTER 22

Patch Administration (Overview)

For the purpose of this discussion, patch administration involves installing or removing Solaris patches from a running Solaris system. It might 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 20.)

Tools For Managing Patches

There are two utilities for managing patches:

- `patchadd` - use to install directory-format patches to a Solaris system.
- `patchrm` - use to remove patches installed on a Solaris system. This command 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 `patchadd`(1M) and `patchrm`(1M). Each patch also contains a `README` file that contains specific information about the patch.

Before installing patches, you might want to know more about patches that have previously been installed. The table below describes commands that provide useful information about patches already installed on a system.

TABLE 22–1 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`.
`patchadd -R` *client_root_path* `–p`	Shows all patches applied to a client, from the server's console.
`patchadd –p`	Shows all patches applied to a system.

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-ROM (See the table below).

TABLE 22–2 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 273 and “Patch Access Via `ftp`” on page 274. These patches are updated nightly. You also receive a patch CD-ROM 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 273 and “Patch Access Via `ftp`” on page 274.

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-ROM 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 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://metalab.unc.edu/pub/sun-info/sun-patches/`

Scroll to the bottom of this list to display the Solaris patch reports at this site.

Patch Access Via `ftp`

To access patches via `ftp`, you can use the `ftp` command 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 the connection is complete, 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 106925–02 is a SunOS 5.7 patch for the glm device driver.

What Happens When You Install a Patch

When you install a patch, the `patchadd` command calls the `pkgadd` command to install the patch packages from the patch directory to a local system’s disk. More specifically, `patchadd`:

- Determines the Solaris version number of the managing host and the target host
- Updates the patch package’s `pkginfo` file with information about patches obsoleted by the patch being installed, other patches required by this patch, and patches incompatible with this patch

During the patch installation, `patchadd` keeps a log of the patch installation in `/var/sadm/patch/`*patch-number*`/log` for the Solaris 2.4 release and compatible versions.

The `patchadd` command will not install a patch under the following conditions:

- The package is not fully installed on the host
- The patch packages architecture differs from the system's architecture
- The patch packages version does not match the installed package's version
- 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)
- The patch being installed requires another patch that is not installed

What Happens When You Remove a Patch

When you back out a patch, the `patchrm` command restores all files modified by that patch, unless:

- The patch was installed with `patchadd -d` (which instructs `patchadd` not to save copies of files being updated or replaced)
- The patch has been obsoleted by a later patch
- The patch is required by another patch

The `patchrm` command calls `pkgadd` to restore packages that were saved from the initial patch installation.

During the patch removal process, `patchrm` keeps a log of the back out process in `/tmp/backoutlog.`*process_id*. This log file is removed if the patch backs out successfully.

CHAPTER 23

Managing Devices Topics

This section provides instructions for managing devices in the Solaris environment. This section contains these chapters.

Chapter 24	Provides a high-level overview of device configuration.
Chapter 25	Provides step-by-step instructions for configuring devices.
Chapter 26	Provides an overview of device naming conventions and instructions for accessing devices.

CHAPTER 24

Device Management (Overview)

The chapter provides overview information about managing peripheral devices in the Solaris environment.

This is a list of overview information in this chapter.

For information about accessing devices, see Chapter 26.

Device management in the Solaris 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.

What's New in Device Management?

This section provides information about new Solaris 8 features related to device management.

SCSI and PCI Hot-Plugging

The `cfgadm` command is updated in this release to provide SCSI and PCI hot-plugging for supported SCSI and PCI controllers on SPARC based and IA based systems.

Hot plugging is the ability to physically add, remove, or replace system components while the system is running. *Dynamic reconfiguration*, available on certain SPARC servers, allows a service provider to remove and replace hot-pluggable system I/O boards in a running system, eliminating the time lost in rebooting. Also, if a replacement board is not immediately available, the system administrator can use dynamic reconfiguration to shut down a failing board while allowing the system to continue operation.

See your SPARC hardware manufacturer's documentation for information about whether dynamic reconfiguration is supported on your system. See the *Solaris 8 (Intel Platform Edition) Hardware Compatibility List* to verify whether your PCI controller supports hot-plugging.

Chapter 25 describes how to use the `cfgadm` command to hot-plug SCSI or PCI controllers.

Improved Device Configuration (`devfsadm`)

The `devfsadm` command manages the special device files in the `/dev` and `/devices` directories. By default, `devfsadm` attempts to load every driver in the system and attach to all possible device instances. Then it creates the device files in the `/devices` directory and the logical links in the `/dev` directory. In addition to managing the `/dev` and `/devices` directories, `devfsadm` also maintains the `path_to_inst`(4) instance database.

In previous Solaris releases, device configuration was handled by `drvconfig`, which managed the physical device entries in the `/devices` directory, and five link generators, `devlinks`, `disks`, `tapes`, `ports`, and `audlinks`, which managed the logical device entries in the `/dev` directory.

These utilities were not aware of hot-pluggable devices nor were they flexible enough for devices with multiple instances. For compatibility purposes, `drvconfig` and the other link generators are symbolic links to the `devfsadm` utility. See Chapter 25 for information about hot-pluggable devices.

Both reconfiguration boot processing and updating the `/dev` and `/devices` directories in response to dynamic reconfiguration events is handled by `devfsadmd`, the daemon version of the `devfsadm` command. This daemon is started from the `/etc/rc*` scripts when a system is booted.

Since `devfsadmd` automatically detects device configuration changes generated by any reconfiguration event, there is no need to run this command interactively.

See `devfsadm`(1M) for more information.

Where to Find Device Management Tasks

The following table 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 24–1 Where to Find Instructions for Adding a Device

For Information On ...	See the Following
Adding a disk	Chapter 30 or Chapter 31
Adding a CD-ROM or tape device	"How to Add a Peripheral Device" on page 290
Adding a modem	"Managing Terminals and Modems (Overview)" in *System Administration Guide, Volume 2*
Adding a printer	"Print Management (Overview)" in *System Administration Guide, Volume 2*

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 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 the following table.

TABLE 24–2 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 `system`(4) 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 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 might be incompatible with Solaris utilities, and might require custom maintenance and administrative utilities.

Contact your device manufacturer for more information.

Displaying Device Configuration Information

Three commands are used to display system and device configuration information:

`prtconf`(1M)	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`(1M)	Displays device configuration information including system hardware, pseudo devices, loadable modules, and selected kernel parameters.
`dmesg`(1M)	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 312 for information on the device names used to identify devices on the system.

`driver not attached` Message

The following driver-related message might 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 based system.

```
# prtconf
System Configuration:  Sun Microsystems  sun4u
Memory size: 128 Megabytes
System Peripherals (Software Nodes):
SUNW,Ultra-5_10
    packages (driver not attached)
        terminal-emulator (driver not attached)
        deblocker (driver not attached)
        obp-tftp (driver not attached)
        disk-label (driver not attached)
        SUNW,builtin-drivers (driver not attached)
        sun-keyboard (driver not attached)
        ufs-file-system (driver not attached)
    chosen (driver not attached)
    openprom (driver not attached)
        client-services (driver not attached)
    options, instance #0
    aliases (driver not attached)
    memory (driver not attached)
    virtual-memory (driver not attached)
    pci, instance #0
        pci, instance #0
            ebus, instance #0
                auxio (driver not attached)
                power, instance #0
                SUNW,pll (driver not attached)
                se, instance #0
                su, instance #0
                su, instance #1
                ecpp (driver not attached)
                fdthree, instance #0
.
.
.
```

The following `sysdef` output is displayed from an IA based system.

```
# sysdef
* Hostid
*
  29f10b4d
*
* i86pc Configuration
*
*
* Devices
*
+boot (driver not attached)
memory (driver not attached)
aliases (driver not attached)
chosen (driver not attached)
i86pc-memory (driver not attached)
i86pc-mmu (driver not attached)
openprom (driver not attached)
options, instance #0
packages (driver not attached)
delayed-writes (driver not attached)
itu-props (driver not attached)
isa, instance #0
 motherboard (driver not attached)
 pnpADP,1542, instance #0
 asy, instance #0
 asy, instance #1
 lp, instance #0 (driver not attached)
 fdc, instance #0
  fd, instance #0
  fd, instance #1 (driver not attached)
 kd (driver not attached)
 kdmouse (driver not attached)
.
.
.
```

▼ 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 based system.

```
# dmesg
date starbug genunix: [ID 540533 kern.notice] SunOS Release
5.8 Generic 64-bit
date starbug genunix: [ID 223299 kern.notice] Copyright
(c) 1983-2000 by S
un Microsystems, Inc.
date starbug genunix: [ID 678236 kern.info] Ethernet address
= 8:0:20:a6:d
4:5b
date starbug genunix: [ID 897550 kern.info] Using default
device instance
data
date starbug unix: [ID 389951 kern.info] mem = 131072K
(0x8000000)
date starbug unix: [ID 930857 kern.info] avail mem = 121724928
date starbug rootnex: [ID 466748 kern.info] root nexus
= Sun Ultra 5/10 UP
A/PCI (UltraSPARC-IIi 333MHz)
.
.
.
#
```

The following dmesg output is displayed from an IA based system.

```
# dmesg
date naboo genunix: [ID 540533 kern.notice] SunOS Release
5.8 Version Generic 32-bit
date naboo genunix: [ID 223299 kern.notice] Copyright (c)
1983-2000 by Sun Microsystems, Inc.
date naboo genunix: [ID 897550 kern.info] Using default
device instance data
date naboo unix: [ID 168242 kern.info] mem = 32380K (0x1f9f000)
date naboo unix: [ID 930857 kern.info] avail mem = 19390464
date naboo rootnex: [ID 466748 kern.info] root nexus =
i86pc
date naboo rootnex: [ID 349649 kern.info] pci0 at root:
space 0 offset 0
date naboo genunix: [ID 936769 kern.info] pci0 is /pci@0,0
date naboo genunix: [ID 678236 kern.info] Ethernet address
= 00:a0:24:89:b0:72
date naboo gld: [ID 944156 kern.info] elx0: 3COM EtherLink
III:
type "ether" mac address 00:a0:24:89:b0:72
date naboo pci: [ID 370704 kern.info] PCI-device: pci10b7,5950@c,
elx0
date naboo genunix: [ID 936769 kern.info] elx0 is /pci@0,0/pci10b7,5950@c
.
.
.
```

CHAPTER 25

Configuring Devices

The chapter provides instructions for configuration devices in the Solaris environment.

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

For information about accessing devices, see Chapter 26.

Adding, removing, or replacing devices in the Solaris environment can be done while the system is still running, if the system components support hot plugging, or the system must be rebooted to reconfigure devices if the system components do not support hot plugging.

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 might have to add a third-party device driver to support the new device.

▼ How to Add a Peripheral Device

1. **Become superuser.**

2. **Follow steps 2 and 3 of "How to Add a Device Driver" on page 291 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
```

`-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 based Platforms ...	On Intel based Platforms ...
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 often will 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.

9. **Verify that the peripheral device has been added by attempting to access the device. See Chapter 26 for information on accessing the device.**

▼ How to Add a Device Driver

This procedure assumes that the device has already been added to the system. If not, see “Adding a Peripheral Device to a System” on page 290.

1. **Become superuser.**

2. **Place the tape, diskette, or CD-ROM into the drive.**

3. **Install the driver.**

```
# pkgadd -d device package-name
```

–d *device*	Identifies the device path name.
package-name	Identifies the package name that contains the device driver.

4. **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
#
```

Dynamic Reconfiguration and Hot-Plugging

Hot-plugging is the ability to physically add, remove, or replace system components while the system is running. *Dynamic reconfiguration* refers to the ability to hot-plug system components and also the general ability to move system resources—both hardware and software—around in the system or disable them in some way without physically removing them from the system.

In this Solaris release, you can hot-plug SCSI devices on SPARC and IA based platforms and PCI adapter cards on IA based systems with the `cfgadm` command. Features of the `cfgadm` command include:

- Displaying system component status
- Testing system components
- Changing component configurations
- Displaying configuration help messages

The benefit of using the `cfgadm` command to reconfigure systems components is that you can add, remove, or replace components while the system is running. An added benefit is that the `cfgadm` command guides you through the steps needed to add, remove, or replace system components. See `cfgadm`(1M) and "SCSI Hot Plugging With the `cfgadm` Command" on page 296 for step-by-step instructions on hot-plugging SCSI components. See "IA: PCI Hot-Plugging With the `cfgadm` Command" on page 306 for step-by-step instructions on hot-plugging PCI adapter cards on IA based systems.

Note - Not all SCSI and PCI controllers support hot-plugging with the `cfgadm` command. For a list of PCI hardware that supports hot-plugging, please refer to the *Solaris 8 (Intel Platform Edition) Hardware Compatibility List*.

As part of Sun's high availability strategy, this feature is expected to be used in conjunction with additional layered products, such as alternate pathing or fail-over software, which provide fault tolerance in the event of a device failure.

Without any high availability software, you can replace a failed device by manually stopping the appropriate applications, unmounting non-critical file systems, and then proceeding with the add or remove operations.

Attachment Points

The `cfgadm` displays information about *attachment points*, which are locations in the system where dynamic reconfiguration operations can occur.

An attachment point consists of:

- An *occupant*, which represents a hardware resource that may be configured into the system, and
- A *receptacle*, which is the location that accepts the occupant.

Attachment points are represented by logical and physical attachment point IDs (ap_ids). The physical ap_id is the physical pathname of the attachment point. The logical ap_id is a user-friendly alternative for the physical ap_id. Refer to `cfgadm`(1M) for more information on ap_ids.

The logical ap_id for a SCSI Host Bus Adapter (HBA), or SCSI controller, is usually represented by the controller number, such as `c0`.

In cases where no controller number has been assigned to a SCSI HBA, then an internally-generated unique identifier is provided. An example of a unique identifier for a SCSI controller is:

```
fas1:scsi
```

The logical ap_id for a SCSI device usually looks like this:

HBA-logical-apid::device-identifier

In the example below, `c0` is the logical ap_id for the SCSI HBA:

```
c0::dsk/c0t3d0
```

The device identifier is typically derived from the logical device name for the device in the `/dev` directory. For example, a tape device with logical device name, `/dev/rmt/1`, has the following logical ap_id:

```
c0::rmt/1
```

If a logical ap_id of a SCSI device cannot be derived from the logical name in the `/dev` directory, then an internally-generated unique identifier is provided. An example of an identifier for the tape device listed above is:

```
c0::st4
```

Refer to `cfgadm_scsi`(1M) for more information on SCSI ap_ids.

The `cfgadm` command represents all resources and dynamic reconfiguration operations in terms of a common set of states (such as configured, unconfigured) and set of operations (connect, configure, unconfigure, and so on). Refer to `cfgadm`(1M) for more information on these generic states and operations.

The receptacle and occupant states for the SCSI HBA attachment points are:

Receptacle State	Description	Occupant State	Description
empty	N/A to SCSI HBA	configured	One or more devices configured on the bus
disconnected	Bus quiesced	unconfigured	No devices configured
connected	Bus active		

Receptacle and occupant state mappings for SCSI device attachment points are:

Receptacle State	Description	Occupant State	Description
empty	N/A to SCSI devices	configured	Device is configured
disconnected	Bus quiesced	unconfigured	Device is not configured
connected	Bus active		

The condition of SCSI attachment points are unknown unless there is special hardware to indicate otherwise. See the instructions below on displaying SCSI component configuration information.

IA: Detaching PCI Adapter Cards

A PCI adapter card hosting non-vital system resources can be removed if the device driver supports hot-plugging. A PCI adapter card is not detachable if it is a vital system resource. For a PCI adapter card to be detachable:

- The device driver must support hot-plugging.
- Critical resources must be accessible through an alternate pathway.

For example, if a system has only one ethernet card installed in it, the ethernet card cannot be detached without losing network connection. This replacement requires additional layered software support to keep the network connection active.

IA: Attaching PCI Adapter Cards

A PCI adapter card can be added to the system as long as:

- There are slots available.

- The device driver supports hot-plugging for this adapter card.

See "IA: PCI Hot-Plugging With the `cfgadm` Command" on page 306 for step-by-step instructions on adding or removing a PCI adapter card.

SCSI Hot Plugging With the `cfgadm` Command

The following section describes various SCSI hot plugging tasks with the `cfgadm` command.

The procedures in this section use specific devices as examples to illustrate how to use the `cfgadm` command to hot plug SCSI components. The device information that you supply, and is displayed with the `cfgadm` command, depends on your system configuration.

▼ How to Display Configuration Information for all Devices

SCSI controllers `c0` and `c1` and the devices attached to them provide examples of the type of device configuration information that can be displayed with the `cfgadm` command.

Note - If the SCSI device is not supported by the `cfgadm` command, it does not display in the `cfgadm` command output.

---

1. **Become superuser.**

2. **Display information about attachment points on the system.**

```
# cfgadm -l
Ap_Id                Type         Receptacle   Occupant     Condition
c0                   scsi-bus     connected    configured   unknown
c1                   scsi-bus     connected    configured   unknown
```

   In this example, `c0` and `c1` represent two SCSI controllers.

3. **Display information about a system's SCSI controllers and their attached devices.**

```
# cfgadm -al
Ap_Id                Type         Receptacle   Occupant     Condition
c0                   scsi-bus     connected    configured   unknown
c0::dsk/c0t0d0       disk         connected    configured   unknown
c0::rmt/0            tape         connected    configured   unknown
c1                   scsi-bus     connected    configured   unknown
c1::dsk/c1t3d0       disk         connected    configured   unknown
c1::dsk/c1t4d0       unavailable  connected    unconfigured unknown
```

Note - The `cfgadm -l` commands displays info about SCSI HBAs but not SCSI devices. Use the `cfgadm -al` command to display information about SCSI devices such as disk and tapes.

In the following examples, only SCSI attachment points are listed. The attachment points displayed on your system will depend on your system configuration.

▼ How to Unconfigure a SCSI Controller

SCSI controller `c1` provides an example of unconfiguring a SCSI controller.

1. **Become superuser.**

2. **Unconfigure a SCSI controller.**

```
# cfgadm -c unconfigure c1
```

3. **Verify the SCSI controller is unconfigured.**

```
# cfgadm -al
Ap_Id                Type         Receptacle   Occupant     Condition
c0                   scsi-bus     connected    configured   unknown
c0::dsk/c0t0d0       disk         connected    configured   unknown
c0::rmt/0            tape         connected    configured   unknown
c1                   scsi-bus     connected    unconfigured unknown
```

Notice that the `Occupant` column specifies `unconfigured`, indicating that the SCSI bus has no configured occupants.

▼ How to Configure a SCSI Controller

SCSI controller `c1` provides an example of configuring a SCSI controller.

1. **Become superuser.**

2. **Configure a SCSI controller.**

```
# cfgadm -c configure c1
```

3. **Verify the SCSI controller is configured.**

```
# cfgadm -al
Ap_Id                Type         Receptacle   Occupant     Condition
c0                   scsi-bus     connected    configured   unknown
c0::dsk/c0t0d0       disk         connected    configured   unknown
c0::rmt/0            tape         connected    configured   unknown
c1                   scsi-bus     connected    configured   unknown
c1::dsk/c1t3d0       disk         connected    configured   unknown
c1::dsk/c1t4d0       unavailable  connected    unconfigured unknown
```

The previous unconfigure procedure removed all devices on the SCSI bus. Now all the devices are configured back into the system.

▼ How to Configure a SCSI Device

SCSI disk `c1t4d0` provides an example of configuring a SCSI device.

1. **Become superuser.**

2. **Identify the device to be configured.**

```
 cfgadm -al
Ap_Id                Type          Receptacle   Occupant     Condition
c0                   scsi-bus      connected    configured   unknown
c0::dsk/c0t0d0       disk          connected    configured   unknown
c0::rmt/0            tape          connected    configured   unknown
c1                   scsi-bus      connected    configured   unknown
c1::dsk/c1t3d0       disk          connected    configured   unknown
c1::dsk/c1t4d0       unavailable   connected    unconfigured unknown
```

3. **Configure a specific SCSI device.**

```
# cfgadm -c configure c1::dsk/c1t4d0
```

4. Verify the SCSI device is configured.

```
# cfgadm -al
Ap_Id                Type         Receptacle   Occupant     Condition
c0                   scsi-bus     connected    configured   unknown
c0::dsk/c0t0d0       disk         connected    configured   unknown
c0::rmt/0            tape         connected    configured   unknown
c1                   scsi-bus     connected    configured   unknown
c1::dsk/c1t3d0       disk         connected    configured   unknown
c1::dsk/c1t4d0       disk         connected    configured   unknown
```

▼ How to Disconnect a SCSI Controller

Disconnecting a SCSI device must be done with caution, particularly when dealing with controllers for disks containing critical file systems such as root (/), `usr`, `var`, and the `swap` partition. The dynamic reconfiguration software cannot detect all cases where a system hang may result. Use this command with caution.

SCSI controller `c1` provides an example of disconnecting a SCSI device.

1. Become superuser.

2. Verify the device is connected before disconnecting it.

```
# cfgadm -al
Ap_Id                Type         Receptacle   Occupant     Condition
c0                   scsi-bus     connected    configured   unknown
c0::dsk/c0t0d0       disk         connected    configured   unknown
c0::rmt/0            tape         connected    configured   unknown
c1                   scsi-bus     connected    configured   unknown
c1::dsk/c1t3d0       disk         connected    configured   unknown
c1::dsk/c1t4d0       disk         connected    configured   unknown
```

3. Disconnect a SCSI controller.

```
# cfgadm -c disconnect c1
WARNING: Disconnecting critical partitions may cause system hang.
Continue (yes/no)? y
```

Caution - This command suspends all I/O activity on the SCSI bus until the `cfgadm -c connect` command is used. The `cfgadm` command does some basic checking to prevent critical partitions from being disconnected, but it cannot detect all cases. Inappropriate use of this command may result in a system hang and could require a system reboot.

4. Verify the SCSI bus is disconnected.

```
# cfgadm -al
Ap_Id                Type          Receptacle   Occupant     Condition
c0                   scsi-bus      connected    configured   unknown
c0::dsk/c0t0d0       disk          connected    configured   unknown
c0::rmt/0            tape          connected    configured   unknown
c1                   unavailable   disconnected configured   unknown
c1::dsk/c1t10d0      unavailable   disconnected configured   unknown
c1::dsk/c1t4d0       unavailable   disconnected configured   unknown
```

The controller and all the devices attached to it are disconnected from the system.

▼ How to Connect a SCSI Controller

SCSI controller `c1` provides an example of connecting a SCSI controller.

1. Become superuser.

2. Verify the device is disconnected before connecting it.

```
# cfgadm -al
Ap_Id                Type          Receptacle   Occupant     Condition
c0                   scsi-bus      connected    configured   unknown
c0::dsk/c0t0d0       disk          connected    configured   unknown
c0::rmt/0            tape          connected    configured   unknown
c1                   unavailable   disconnected configured   unknown
c1::dsk/c1t10d0      unavailable   disconnected configured   unknown
c1::dsk/c1t4d0       unavailable   disconnected configured   unknown
```

3. Connect a SCSI controller.

```
# cfgadm -c connect c1
```

4. Verify the SCSI controller is connected.

```
# cfgadm -al
Ap_Id                Type          Receptacle   Occupant     Condition
c0                   scsi-bus      connected    configured   unknown
c0::dsk/c0t0d0       disk          connected    configured   unknown
c0::rmt/0            tape          connected    configured   unknown
c1                   scsi-bus      connected    configured   unknown
c1::dsk/c1t3d0       disk          connected    configured   unknown
c1::dsk/c1t4d0       disk          connected    configured   unknown
```

▼ SPARC: How to Add a SCSI Device to a SCSI Bus

SCSI controller `c1` provides an example of how to add a SCSI device to a SCSI bus.

Note - When adding devices, the ap_id of the SCSI HBA (controller) to which the device is attached is specified, not the ap_id of the device itself.

---

1. Become superuser.

2. Identify the current SCSI configuration.

```
# cfgadm -al
Ap_Id                Type          Receptacle   Occupant     Condition
c0                   scsi-bus      connected    configured   unknown
c0::dsk/c0t0d0       disk          connected    configured   unknown
c0::rmt/0            tape          connected    configured   unknown
c1                   scsi-bus      connected    configured   unknown
c1::dsk/c1t3d0       disk          connected    configured   unknown
```

3. Add a SCSI device to a SCSI bus.

```
# cfgadm -x insert_device c1
Adding device to SCSI HBA: /devices/sbus@1f,0/SUNW,fas@1,8800000
This operation will suspend activity on SCSI bus: c1
Continue (yes/no)? y
SCSI bus quiesced successfully.
It is now safe to proceed with hotplug operation.
Enter y if operation is complete or n to abort (yes/no)? y
```

a. Type `y` at the `Continue (yes/no)?` prompt to proceed.

I/O activity on the SCSI bus will be suspended while the hot-plug operation is in progress.

b. Connect the device and then power it on.

c. Type `y` at the `Enter y if operation is complete or n to abort (yes/no)?` prompt after the new device has been inserted.

4. Verify the device has been added.

```
# cfgadm -al
Ap_Id                Type          Receptacle   Occupant     Condition
c0                   scsi-bus      connected    configured   unknown
c0::dsk/c0t0d0       disk          connected    configured   unknown
c0::rmt/0            tape          connected    configured   unknown
c1                   scsi-bus      connected    configured   unknown
c1::dsk/c1t3d0       disk          connected    configured   unknown
c1::dsk/c1t4d0       disk          connected    configured   unknown
```

A new disk has been added to controller `c1`.

▼ SPARC: How to Replace an Identical Device on a SCSI Controller

SCSI disk `c1t4d0` provides an example of replacing an identical device on a SCSI controller.

1. **Become superuser.**

2. **Identify the current SCSI configuration.**

```
# cfgadm -al
Ap_Id                Type          Receptacle   Occupant     Condition
c0                   scsi-bus      connected    configured   unknown
c0::dsk/c0t0d0       disk          connected    configured   unknown
c0::rmt/0            tape          connected    configured   unknown
c1                   scsi-bus      connected    configured   unknown
c1::dsk/c1t3d0       disk          connected    configured   unknown
c1::dsk/c1t4d0       disk          connected    configured   unknown
```

3. **Replace a device on the SCSI bus with another device of the same type.**

```
# cfgadm -x replace_device c1::dsk/c1t4d0
Replacing SCSI device: /devices/sbus@1f,0/SUNW,fas@1,8800000/sd@4,0
This operation will suspend activity on SCSI bus: c1
Continue (yes/no)? y
SCSI bus quiesced successfully.
It is now safe to proceed with hotplug operation.
Enter y if operation is complete or n to abort (yes/no)? y
```

   a. **Type `y` at the `Continue (yes/no)?` prompt to proceed.**
      I/O activity on the SCSI bus will be suspended while the hot-plug operation is in progress.

   b. **Power off the device to be removed and remove it. Add the replacement device, which should be of the same type and at the same address (target and lun) as the device to be removed. Then power it on.**

   c. **Type `y` at the `Enter y if operation is complete or n to abort (yes/no)?` prompt after the device has been replaced.**

4. **Verify the device has been replaced.**

```
# cfgadm -al
Ap_Id                Type          Receptacle   Occupant     Condition
c0                   scsi-bus      connected    configured   unknown
c0::dsk/c0t0d0       disk          connected    configured   unknown
c0::rmt/0            tape          connected    configured   unknown
c1                   scsi-bus      connected    configured   unknown
c1::dsk/c1t3d0       disk          connected    configured   unknown
c1::dsk/c1t4d0       disk          connected    configured   unknown
```

▼ SPARC: How to Remove a SCSI Device

SCSI disk `c1t4d0` provides an example of removing a device on a SCSI controller.

1. Become superuser.

2. Identify the current SCSI configuration.

```
# cfgadm -al
Ap_Id                Type          Receptacle   Occupant     Condition
c0                   scsi-bus      connected    configured   unknown
c0::dsk/c0t0d0       disk          connected    configured   unknown
c0::rmt/0            tape          connected    configured   unknown
c1                   scsi-bus      connected    configured   unknown
c1::dsk/c1t3d0       disk          connected    configured   unknown
c1::dsk/c1t4d0       disk          connected    configured   unknown
```

3. Remove a SCSI device from the system.

```
# cfgadm -x remove_device c1::dsk/c1t4d0
Removing SCSI device: /devices/sbus@1f,0/SUNW,fas@1,8800000/sd@4,0
This operation will suspend activity on SCSI bus: c1
Continue (yes/no)? y
SCSI bus quiesced successfully.
It is now safe to proceed with hotplug operation.
Enter y if operation is complete or n to abort (yes/no)? y
```

a. Type `y` at the `Continue (yes/no)?` prompt to proceed.

I/O activity on the SCSI bus will be suspended while the hot-plug operation is in progress.

b. Power off the device to be removed and remove it.

c. **Type `y` at the** `Enter y if operation is complete or n to abort (yes/no)?` **prompt after the device has been removed.**

4. Verify the device has been removed from the system.

```
# cfgadm -al
Ap_Id                Type          Receptacle   Occupant     Condition
c0                   scsi-bus      connected    configured   unknown
c0::dsk/c0t0d0       disk          connected    configured   unknown
c0::rmt/0            tape          connected    configured   unknown
c1                   scsi-bus      connected    configured   unknown
c1::dsk/c1t3d0       disk          connected    configured   unknown
```

SPARC: Troubleshooting SCSI Configuration Problems

Error Message

```
cfgadm: Component system is busy, try again: failed to offline:
     device path
          Resource              Information
     ------------------  -------------------------
     /dev/dsk/c1t0d0s0   mounted filesystem "/file-system"
```

Cause

You attempted to remove or replace a device with a mounted file system.

Solution

Unmount the file system listed in the error message and try the `cfgadm` operation again.

IA: PCI Hot-Plugging With the `cfgadm` Command

The following section describes different hot-plugging operations and then provides step-by-step instructions for hot-plugging PCI adapter cards on IA based systems.

In the following examples, only PCI attachment points are listed, for brevity. The attachment points displayed on your system will depend on your system configuration.

▼ IA: How to Display PCI Slot Configuration Information

The `cfgadm`(1M) command displays the status of PCI hot-pluggable devices and slots on a system.

1. **Become superuser.**

2. **Display PCI slot configuration information.**

```
# cfgadm
Ap_Id                Type         Receptacle   Occupant     Condition
pci1:hpc0_slot0      unknown      empty        unconfigured unknown
pci1:hpc0_slot1      unknown      empty        unconfigured unknown
pci1:hpc0_slot2      unknown      empty        unconfigured unknown
pci1:hpc0_slot3      ethernet/hp  connected    configured   ok
pci1:hpc0_slot4      unknown      empty        unconfigured unknown
# cfgadm -s "cols=ap_id:type:info" pci
Ap_Id                Type         Information
pci1:hpc0_slot0      unknown      Slot 7
pci1:hpc0_slot1      unknown      Slot 8
pci1:hpc0_slot2      unknown      Slot 9
pci1:hpc0_slot3      ethernet/hp  Slot 10
pci1:hpc0_slot4      unknown      Slot 11
```

The logical `ap_id`, `pci1:hpc0_slot0`, is the logical `ap_id` for that particular hot-pluggable slot, `Slot 7`, (physical identification of this slot). The component `hpc0` indicates the hot-pluggable adapter card for this slot and `pci1` indicates the PCI bus instance. The `Type` field indicates the type of PCI adapter card present in the slot.

▼ IA: How to Remove a PCI Adapter Card

1. **Become superuser.**

2. **Determine which slot the adapter card is in.**

```
# cfgadm
Ap_Id                Type         Receptacle   Occupant     Condition
pci1:hpc0_slot0      unknown      empty        unconfigured unknown
pci1:hpc0_slot1      unknown      empty        unconfigured unknown
pci1:hpc0_slot2      unknown      empty        unconfigured unknown
pci1:hpc0_slot3      ethernet/hp  connected    configured   ok
pci1:hpc0_slot4      unknown      empty        unconfigured unknown
```

3. **Stop the application that has the device open.**

   For example, if this is an ethernet card, use `ifconfig(1M)` to bring down the interface and unplumb the interface.

4. **Unconfigure the device.**

```
# cfgadm -c unconfigure pci1:hpc0_slot3
```

5. **Confirm the device has been unconfigured.**

```
# cfgadm
Ap_Id                Type         Receptacle   Occupant     Condition
pci1:hpc0_slot0      unknown      empty        unconfigured unknown
pci1:hpc0_slot1      unknown      empty        unconfigured unknown
pci1:hpc0_slot2      unknown      empty        unconfigured unknown
pci1:hpc0_slot3      ethernet/hp  connected    unconfigured unknown
pci1:hpc0_slot4      unknown      empty        unconfigured unknown
```

6. **Disconnect the power to the slot.**

```
# cfgadm -c disconnect pci1:hpc0_slot3
```

7. **Confirm the device has been disconnected.**

```
# cfgadm
Ap_Id                Type         Receptacle   Occupant     Condition
pci1:hpc0_slot0      unknown      empty        unconfigured unknown
pci1:hpc0_slot1      unknown      empty        unconfigured unknown
pci1:hpc0_slot2      unknown      empty        unconfigured unknown
pci1:hpc0_slot3      ethernet/hp  disconnected unconfigured unknown
pci1:hpc0_slot4      unknown      empty        unconfigured unknown
```

8. Open the slot latches and remove the board.

▼ IA: How to Add a PCI Adapter Card

1. Become superuser.

2. Identify the hot-pluggable slot and open latches.

3. Insert the adapter card into a hot-pluggable slot.

4. Determine which slot the adapter card is in once it is inserted and the latches are closed.

```
# cfgadm
Ap_Id                Type         Receptacle   Occupant     Condition
pci1:hpc0_slot0      unknown      empty        unconfigured unknown
pci1:hpc0_slot1      unknown      empty        unconfigured unknown
pci1:hpc0_slot2      unknown      empty        unconfigured unknown
pci1:hpc0_slot3      ethernet/hp  disconnected unconfigured unknown
pci1:hpc0_slot4      unknown      empty        unconfigured unknown
```

5. Connect the power to the slot.

```
# cfgadm -c connect pci1:hpc0_slot3
```

6. Confirm the slot is connected.

```
# cfgadm
Ap_Id                Type         Receptacle   Occupant     Condition
pci1:hpc0_slot0      unknown      empty        unconfigured unknown
pci1:hpc0_slot1      unknown      empty        unconfigured unknown
pci1:hpc0_slot2      unknown      empty        unconfigured unknown
pci1:hpc0_slot3      ethernet/hp  connected    unconfigured unknown
pci1:hpc0_slot4      unknown      empty        unconfigured unknown
```

7. Configure the PCI hot-pluggable adapter card.

```
# cfgadm -c configure pci1:hpc0_slot3
```

8. Verify the configuration of the adapter card in the slot.

```
# cfgadm
Ap_Id                Type         Receptacle   Occupant     Condition
pci1:hpc0_slot0      unknown      empty        unconfigured unknown
pci1:hpc0_slot1      unknown      empty        unconfigured unknown
pci1:hpc0_slot2      unknown      empty        unconfigured unknown
pci1:hpc0_slot3      ethernet/hp  connected    configured   unknown
pci1:hpc0_slot4      unknown      empty        unconfigured unknown
```

9. Configure any supporting software if this is a new device.

For example, if this is an ethernet card, use the `ifconfig(1m)` command to set up the interface.

IA: Troubleshooting PCI Configuration Problems

Error Message

```
cfgadm: Configuration operation invalid: invalid transition
```

Cause

An invalid transition was attempted.

Solution

Check whether the `cfgadm -c` command was issued appropriately. Use `cfgadm` to check the current receptacle and occupant state and make sure the `ap_id` is correct.

Error Message

```
cfgadm: Attachment point not found
```

Cause

Specified attachment point was not found.

Solution

Check whether the attachment point is correct. Use `cfgadm` to display a list of available attachment points. Also check the physical path to see if the attachment point is still there.

Note - In addition to the `cfgadm` command, several other commands are helpful during hot-plug operations. The `prtconf(1M)` command displays whether or not Solaris recognizes the hardware. After inserting hardware, use the `prtconf` command to verify that the hardware is recognized. After a configure operation, use the `prtconf -D` command to verify the driver is attached to the newly installed hardware device.

CHAPTER 26

Accessing Devices (Overview)

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 24.

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 the *OpenBoot 3.x Command Reference Manual* for more information on device hierarchy.

Device Naming Conventions

Devices are referenced in three ways in the Solaris environment.

- Physical device name – Represents the full device pathname in the device information hierarchy. Physical device names are displayed by 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 by 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 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.

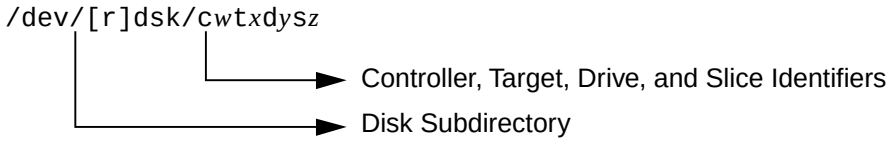
Logical Disk Device Names

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 man page for that command.

The following table 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`(1M)	Block	`df /dev/dsk/c0t3d0s6`
`fsck`(1M)	Raw	`fsck -p /dev/rdsk/c0t0d0s0`
`mount`(1M)	Block	`mount /dev/dsk/c1t0d0s7 /export/home`

TABLE 26–1 Device Interface Type Required by Some Frequently Used Commands *(continued)*

Command	Interface Type	Example of Use
`newfs`(1M)	Raw	`newfs /dev/rdsk/c0t0d1s1`
`prtvtoc`(1M)	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. The following table describes the different types of direct or bus-oriented controllers on different platforms.

TABLE 26–2 Controller Types

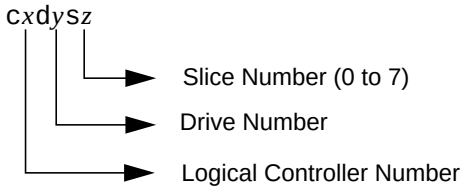
Direct controllers	Bus-Oriented Controllers
Xylogics (SPARC)	SCSI (SPARC / IA)
IDE (IA)	IPI (SPARC)

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 based system, follow the naming convention shown in the figure below.

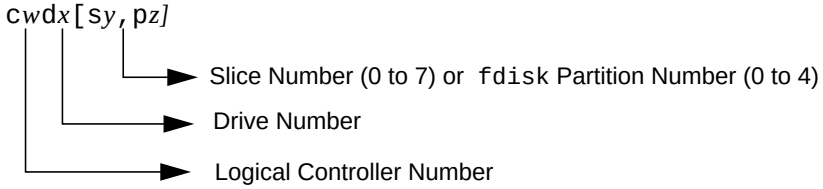


To indicate the whole disk, specify slice 2 (2).

If you have only one controller on your system, *x* will always be 0.

IA: Disks With Direct Controllers

To specify a slice on a disk with an IDE controller on an IA based system, follow the naming convention shown in the figure below.

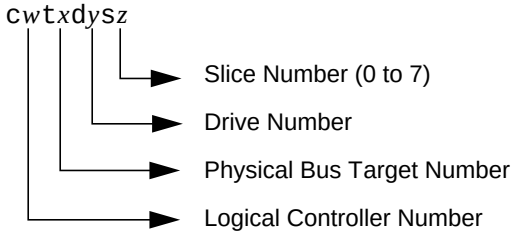


To indicate the entire Solaris `fdisk` partition, specify slice 2 (s2).

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 based system, follow the naming convention shown in the following figure.



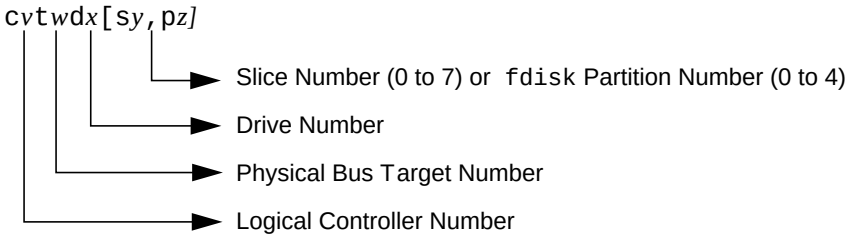
If you have only one controller on your system, *w* will always be 0.

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).

IA: Disks With SCSI Controllers

To specify a slice on a disk with a SCSI controller on an IA based system, follow the naming convention shown in the following figure.



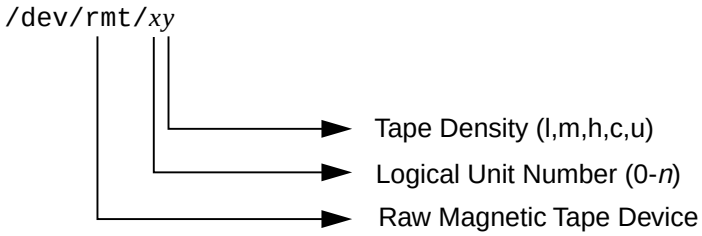
If you have only one controller on your system, *v* will always be 0.

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 might 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 47.

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 14 for information on accessing your CD-ROM device.

CHAPTER 27

Managing Disks Topics

This section provides instructions for managing disks in the Solaris environment. This section contains these chapters.

Chapter 28	Provides an overview of Solaris disk slices and an introduction to the `format` utility.
Chapter 29	Provides step-by-step instructions for formatting a disk, examining disk labels, and repairing a defective disk sector.
Chapter 30	Provides step-by-step instructions for adding a disk to a SPARC based system.
Chapter 31	Provides step-by-step instructions for adding a disk to an IA based system.
Chapter 32	Provides a description of the `format` utility's menu and commands. This chapter also includes information about the `format.dat` file, rules for providing input to `format` commands, and instructions on using the help facility.

CHAPTER 28

Disk Management (Overview)

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 30 or Chapter 31.

What's New in Disk Management?

This section describes features introduced in the Solaris 8 release.

IA: Support for Large Disks

This Solaris release fully uses disks larger than 8 Gbytes because of improved BIOS interfaces. Previously, the following limitations impacted IA based systems running the Solaris Intel Platform Edition:

- On IDE disks, only the first 8 Gbytes could be used by the system.
- On either SCSI or IDE disks, only the first 8 Gbytes could only be used for the root (/) slice.

Both of these restrictions have been removed on systems with improved BIOSes.

See *Solaris 8 (Intel Platform Edition) Installation Guide* for more information.

Where to Find Disk Management Tasks

Use these references to find step-by-step instructions for managing disks.

- Chapter 30
- Chapter 31

Introduction

Managing disks in the Solaris 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 might 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 Chapter 34 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 IA platforms. The table below summarizes the differences:

TABLE 28–1 Slice Differences on Platforms

SPARC Platforms	IA Platforms
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 based systems, Solaris defines eight disk slices and assigns to each a conventional use. These slices are numbered 0 through 7. The table below summarizes the contents of the eight Solaris slices on a SPARC based system.

TABLE 28–2 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 the `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.

IA: Disk Slices

On IA based 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 a Solaris `fdisk` partition as shown in the following table.

TABLE 28–3 IA: 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 the `format` utility 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.

TABLE 28–3 IA: Customary Disk Slices *(continued)*

Slice	File System	Usually Found on Client or Server Systems?	Purpose
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.

Using Raw Data Slices

The SunOS operating system stores the disk label in block 0, cylinder 0 of each disk. This means that using third-party database applications that create raw data slices must not start at block 0, cylinder 0, or the disk label will be overwritten and the data on the disk will be inaccessible.

Do not use the following areas of the disk for raw data slices, which are sometimes created by third-party database applications:

1. Block 0, cylinder 0, where the disk label is stored.
2. Avoid cylinder 0 entirely for improved performance.
3. Slice 2, which represents the entire disk.

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.

The system configurations are:

- Servers
- Standalone systems

Each system configuration requires the use of different slices. The table below lists these requirements.

TABLE 28–4 System Configurations and Slice Requirements

Slice	Servers	Standalone Systems
0	root	root
1	swap	swap
2	—	—
3	`/export`	—
4	`/export/swap`	—
5	`/opt`	`/opt`
6	`/usr`	`/usr`
7	`/export/home`	`/home`

See “Overview of System Types” on page 96 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 it uses 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

The table below shows the features and associated benefits that the `format` utility provides.

TABLE 28–5 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 32.

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 might 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 30 and Chapter 31.

See the section below for guidelines on using the `format` utility.

Guidelines for Using the `format` Utility

TABLE 28–6 The `format` Utility Guidelines

Use `format` 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 might 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 342
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 30 or Chapter 31 or if the system must be reinstalled, *Solaris 8 Advanced Installation Guide*
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 30 or Chapter 31

TABLE 28–6 The `format` Utility Guidelines *(continued)*

Use `format` To ...	**Considerations ...**	**Where to Go ...**
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 30 or Chapter 31
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 might 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 might need to remap a bad sector and restore any lost data.	Chapter 32

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 341 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 342 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 might 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. The table below describes partition table terminology.

TABLE 28–7 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.

TABLE 28–7 Partition Table Terminology *(continued)*

Partition Term	Value	Description
	`wu rm`	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 tags are assigned by convention and require no maintenance.

See "How to Display Disk Slice Information" on page 344 or "How to Examine a Disk Label" on page 348 for more information on displaying the partition table.

Examples—Partition Tables

The following partition table example is displayed from a 1.05-Gbyte disk using the `format` utility:

```
Total disk cylinders available: 2036 + 2 (reserved cylinders)

Part      Tag    Flag     Cylinders        Size            Blocks
  0       root    wm       0 -  300      148.15MB    (301/0/0)   303408
  1       swap    wu     301 -  524      110.25MB    (224/0/0)   225792
  2     backup    wm       0 - 2035     1002.09MB    (2036/0/0) 2052288
  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     525 - 2035      743.70MB    (1511/0/0) 1523088
  7 unassigned    wm       0               0         (0/0/0)          0
```

The partition table contains the following information:

Column Name	Description
`Part`	Partition (or slice number). See Table 28–7 for a description of this column.
`Tag`	Partition tag. See Table 28–7 for a description of this column.
`Flags`	Partition flag. See Table 28–7 for a description of this column.

Column Name	Description
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/c0t1d0s0
* /dev/rdsk/c0t1d0s0 partition map
*
* Dimensions:
*     512 bytes/sector
*      72 sectors/track
*      14 tracks/cylinder
*    1008 sectors/cylinder
*    2038 cylinders
*    2036 accessible cylinders
*
* Flags:
*   1: unmountable
*  10: read-only
*
*                          First     Sector    Last
* Partition  Tag  Flags    Sector     Count    Sector  Mount Directory
       0      2    00          0    303408    303407   /
       1      3    01     303408    225792    529199
       2      5    00          0   2052288   2052287
       6      4    00     529200   1523088   2052287   /usr
```

The disk label includes the following information:

Dimensions – This section describes the physical dimensions of the disk drive.

Flags – This section describes the flags listed in the partition table section. See Table 28–7 for a description of partition flags.

Partition (or Slice) Table – This section contains the following information:

Column Name	Description
Partion	Partition (or slice number). See Table 28–7 for a description of this column.
Tag	Partition tag. See Table 28–7 for a description of this column.

Column Name	Description
`Flags`	Partition flag. See Table 28–7 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 367 or “IA: How to Create Disk Slices and Label a Disk” on page 386 for information on using the free hog slice.

CHAPTER 29

Administering Disks (Tasks)

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 release.

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

For overview information about disk management, see Chapter 28.

Administering Disks Task Map

TABLE 29–1 Administering Disks Task Map

Task	Description	For Instructions, Go To
1. 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" on page 339
2. Format the Disk	Determine whether a disk is already formatted by using the `format` utility. In most cases, disks are already formatted. Use the `format` utility if you need to format a disk.	"How to Determine if a Disk is Formatted" on page 341 "How to Format a Disk" on page 342
3. Display Slice Information	Display slice information by using the `format` utility.	"How to Display Disk Slice Information" on page 344
4. Label the Disk	Create the disk label by using the `format` utility.	"How to Label a Disk" on page 346
5. Examine the Disk Label	Examine the disk label by using the `prtvtoc` command.	"How to Examine a Disk Label" on page 348
6. Create a `format.dat` Entry	Create a `format.dat` entry to support a third-party disk.	"How to Create a `format.dat` Entry" on page 353
7. 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" on page 357
8. If Necessary, Fix a Defective Disk Sector	Fix a defective disk sector by using the `format` utility.	"How to Repair a Defective Sector" on page 359

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 32 for information on using the `format` utility.

▼ How to Identify the Disks on a System

1. **Become superuser.**

2. **Identify the disks that are recognized on the system with the `format` utility.**

```
# format
```

The format utility displays a list of disks that it recognizes under `AVAILABLE DISK SELECTIONS`.

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 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
     /iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd@1,0
  1. c0t3d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
     /iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd@3,0
Specify disk (enter its number):
```

The `format` output associates a disk's physical and local device name to the disk's marketing name which appears in angle brackets <>. This is an easy way to identify which local device names represent the disks connected to your system. See Chapter 26 for a description of local 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>
```

(Continuation)

```
     /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 based system.

```
# format
AVAILABLE DISK SELECTIONS:
  0. c0t3d0 <SUN2.1G cyl 2733 alt 2 hd 19 sec 80>
     /iommu@0,10000000/sbus@0,10001000/espdma@5,8400000/esp@5,8800000/
sd@3,0
Specify disk (enter its number):
```

The `format` output identifies that disk 0 (target 3) is connected to the first SCSI host adapter (`espdma@...`), which is connected to the first SBus device (`sbus@0...`). The output also associates both the physical and logical device name to the disk's marketing name, `SUN02.1G`.

The following example identifies the disks on an IA based system.

```
# format
AVAILABLE DISK SELECTIONS:
  0. c0d0 <DEFAULT cyl 615 alt 2 hd 64 sec 63>
     /pci@0,0/pci-ide@7,1/ata@0/cmdk@0,0
  1. c0d1 <DEFAULT cyl 522 alt 2 hd 32 sec 63>
     /pci@0,0/pci-ide@7,1/ata@0/cmdk@1,0
  2. c1d0 <DEFAULT cyl 817 alt 2 hd 256 sec 63>
     /pci@0,0/pci-ide@7,1/ata@1/cmdk@0,0
Specify disk (enter its number):
```

The `format` output identifies that disk 0 is connected to the first PCI host adapter (`pci-ide@7...`), which is connected to the ATA device (`ata...`). The `format` output on an IA based 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 30 or Chapter 31.
Is a third-party disk	Go to "Creating a `format.dat` Entry" on page 353.
Label was corrupted by a system problem, such as a power failure	Go to "How to Label a Disk" on page 346.
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 superuser.**

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
```

4. **Verify that the disk you chose is formatted by identifying the following message.**

```
[disk formatted]
```

Example—Determining if a Disk Is Formatted

The following example shows that disk `c0t3d0` is formatted.

```
# format
AVAILABLE DISK SELECTIONS:
  0. c0t1d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
     /iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd@1,0
  1. c0t3d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
     /iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd@3,0
Specify disk (enter its number): 0
selecting c0t1d0
[disk formatted]
```

▼ How to Format a Disk

1. **Become superuser.**

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 might 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 23 minutes (estimated). Continue? yes
```

5. Verify that the disk format is successful by identifying the following messages.

```
Beginning format. The current time Tue ABC xx xx:xx:xx xxxx

Formatting...
done

Verifying media...
        pass 0 - pattern = 0xc6dec6de
   2035/12/18

        pass 1 - pattern = 0x6db6db6d
   2035/12/18

Total of 0 defective blocks repaired.
```

Example—Formatting a Disk

The following example formats the disk `c0t3d0`.

```
# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
  0. c0t1d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
     /iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd@1,0
  1. c0t3d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
     /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> format
Ready to format. Formatting cannot be interrupted
and takes 23 minutes (estimated). Continue? yes
Beginning format. The current time is Wed Jul 14 10:03:34 1999
Formatting ...
done
Verifying media...
        pass 0 - pattern = 0xc6dec6de
   2035/12/18
```

(continued)

```
       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 367 or "IA: How to Create Disk Slices and Label a Disk" on page 386 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 superuser.**
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
#
```

7. **Verify displayed slice information by identifying specific slice tags and slices.**

   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/dsk/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: 2036 + 2 (reserved cylinders)

Part      Tag    Flag     Cylinders        Size            Blocks
  0       root    wm       0 -  300      148.15MB    (301/0/0)   303408
  1       swap    wu     301 -  524      110.25MB    (224/0/0)   225792
  2     backup    wm       0 - 2035     1002.09MB    (2036/0/0) 2052288
  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     525 - 2035      743.70MB    (1511/0/0) 1523088
  7 unassigned    wm       0               0         (0/0/0)          0
partition> q
format> q
#
```

See Chapter 28 for a detailed description of the slice information displayed in these examples.

The following example displays the slice information on disk `/dev/dsk/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: 817 + 2 (reserved cylinders)

Part      Tag    Flag     Cylinders       Size            Blocks
  0 unassigned    wm       3 - 816        6.26GB    (814/0/0) 13128192
  1 unassigned    wm       0              0         (0/0/0)          0
  2     backup    wm       0 - 816        6.28GB    (817/0/0) 13176576
  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 unassigned    wm       0              0         (0/0/0)          0
  7 unassigned    wm       0              0         (0/0/0)          0
  8       boot    wu       0 -   0        7.88MB    (1/0/0)      16128
  9 alternates    wu       1 -   2       15.75MB    (2/0/0)      32256
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 might 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 superuser.**

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 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** `y` **at the** `Label it now?` **prompt.**

```
Disk not labeled. Label it now? y
```

The disk is now labeled. Go to step 10 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)[12]: 12
```

8. **Label the disk. If the disk is not labeled, the following message is displayed.**

```
Disk not labeled. Label it now? y
```

Otherwise you are prompted with this message:

```
Ready to label disk, continue? y
```

9. **Use the** `verify` **command from the** `format` **main menu to verify the disk label.**

```
format> verify
```

10. Exit the `format` utility by typing `q` at the `format>` prompt.

```
partition> q
format> q
#
```

Example—Labeling a Disk

The following example automatically configures and labels a 1.05-Gbyte disk.

```
# format
 c1t0d0: configured with capacity of 1002.09MB

AVAILABLE DISK SELECTIONS:
   0. c0t3d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
     /iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd@1,0
   1. c1t0d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
     /iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd@1,0
Specify disk (enter its number): 1
Disk not labeled.  Label it now?  yes
format> verify
#
```

▼ How to Examine a Disk Label

Examine disk label information by using the `prtvtoc`(1M) command. See Chapter 28 for a detailed description of the disk label and the information displayed by the `prtvtoc` command.

1. **Become superuser.**

2. **Display the disk label information by using the `prtvtoc` command.**

```
# prtvtoc /dev/rdsk/device-name
```

device-name — Raw disk device you want to examine.

Example—Examining a Disk Label

The following example shows the disk label information for disk `/dev/rdsk/c0t1d0s0`.

```
# prtvtoc /dev/rdsk/c0t1d0s0
* /dev/rdsk/c0t1d0s0 partition map
*
* Dimensions:
*     512 bytes/sector
*      72 sectors/track
*      14 tracks/cylinder
*    1008 sectors/cylinder
*    2038 cylinders
*    2036 accessible cylinders
*
* Flags:
*   1: unmountable
*  10: read-only
*
*                           First     Sector    Last
* Partition  Tag  Flags    Sector     Count    Sector  Mount Directory
       0      2    00          0    303408    303407   /
       1      3    01     303408    225792    529199
       2      5    00          0   2052288   2052287
       6      4    00     529200   1523088   2052287   /usr
#
```

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 to be recreated 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 10 or Chapter 11 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:

```
cwtxdy: configured with capacity of abcMB
```

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  = <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
pcyl        = 2038
ncyl        = 2036
acyl        =    2
nhead       =   14
nsect       =   72
Part      Tag    Flag     Cylinders        Size            Blocks
  0       root    wm       0 -  300      148.15MB    (301/0/0)   303408
  1       swap    wu     301 -  524      110.25MB    (224/0/0)   225792
  2     backup    wm       0 - 2035     1002.09MB    (2036/0/0) 2052288
  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
```

(continued)

(Continuation)

```
  6        usr    wm     525 - 2035      743.70MB    (1511/0/0) 1523088
  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
```

The format utility displays 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)[12]: 12
```

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 for information about using the `fsck` command.

Adding a Third-Party Disk

The Solaris environment supports many third-party disks. However, you might 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 adding system disks or secondary disks in Chapter 30 or Chapter 31.

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 with the Solaris 2.3 release, the `format` utility automatically configures the SCSI-2 drivers if the drives are powered on during a reconfiguration boot. See "How to Automatically Configure a SCSI Drive" on page 355 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 32 for more information on adding an entry to the `/etc/format.dat` file.

▼ How to Create a `format.dat` Entry

1. **Become superuser.**

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 32.**

   Use the disk's hardware product documentation to gather the required information.

Automatically Configuring SCSI Disk Drives

In Solaris 2.3 release and compatible versions, 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.

The following steps are involved in configuring a SCSI drive using autoconfiguration:

- Shutting down the system
- Attaching the SCSI disk drive to the system
- Turning on the disk drive
- Performing a reconfiguration boot
- Using 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 Drive" on page 355 for step-by-step instructions on configuring a SCSI disk drive automatically.

Here are the default slice rules that `format` uses to create the partition table.

TABLE 29–2 SCSI Disk Slice Rules

Disk Size	Root File System	Swap Slice
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
More than 2.0 Gbytes	128 Mbytes	128 Mbytes

In all cases, slice 6 (for the `/usr` file system) 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 32 for more information about using SCSI automatic configuration.

▼ How to Automatically Configure a SCSI Drive

1. **Become superuser.**

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
```

`–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 superuser, invoke the `format` utility, and select the disk to be configured automatically.**

```
# format
Searching for disks...done
c1t0d0: configured with capacity of 1002.09MB
AVAILABLE DISK SELECTIONS:
  0. c0t1d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
     /iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd@1,0
  1. c0t3d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
     /iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd@3,0
Specify disk (enter its number): 1
```

10. **Reply `yes` to the prompt to label the disk.**

    Replying `y` will cause the disk label to be generated and written to the disk by the autoconfiguration feature.

```
Disk not labeled. Label it now? y
```

11. **Verify the disk label with the `verify` command.**

```
format> verify
```

12. **Exit the `format` utility.**

```
format> q
```

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 might become aware of defective sectors when you:

- Run surface analysis on a disk.

  See "The `analyze` Menu" on page 396 for more information on the analysis functionality of the `format` utility.

  The defective area reported while your system is running might 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 357 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 'name':
 ASC = 0x11 (unrecovered read error), ASCQ = 0x0, FRU = 0x0
```

  The above console message indicates that block 179 might 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 superuser.**

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 by typing `setup` at the `analyze>` prompt. Use the parameters shown here:

```
analyze> setup
Analyze entire disk [yes]? n
Enter starting block number [0, 0/0/0]: 12330
Enter ending block number [2052287, 2035/13/71]: 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 [126, 0/1/54]: 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
   2035/12/1825/7/24
        pass 1
Block 12354  (18/4/18), Corrected media error (hard data ecc)
   25/7/24
^C
```

(continued)

```
Total of 1 defective blocks repaired.
```

▼ How to Repair a Defective Sector

1. **Become superuser.**

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 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
     /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.**

```
Enter absolute block number of defect: 12354
   Ready to repair defect, continue? y
   Repairing block 12354 (18/4/18)...ok.
format>
```

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 357 for more information.

Tips and Tricks for Managing Disks

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. c0t1d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
     /iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd@1,0
  1. c0t3d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
     /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 8f 00 00 12 53 45 41 47 41 54 45 20     ........NAME....
53 54 31 31 32 30 30 4e 20 53 55 4e 31 2e 30 35     ST11200N SUN1.05
38 33 35 38 30 30 30 33 30 32 30 39 00 00 00 00     835800030209....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00     ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00     ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00     ................
00 43 6f 70 79 72 69 67 68 74 20 28 63 29 20 31     .Copyright (c) 1
39 39 32 20 53 65 61 67 61 74 65 20 41 6c 6c 20     992 NAME    All
72 69 67 68 74 73 20 72 65 73 65 72 76 65 64 20     rights reserved
30 30 30                                            000
Vendor:   name
Product:  ST11200N SUN1.05
Revision: 8358
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/cwtxdysz | fmthard -s - /dev/rdsk/cwt${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.
#
```

CHAPTER 30

SPARC: Adding a Disk (Tasks)

This chapter provides the procedures for adding a disk to a SPARC based system.

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

For overview information about disk management, see Chapter 28. For step-by-step instructions on adding a disk to an IA based system, see Chapter 31.

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 systems 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 Task Map

TABLE 30–1 SPARC: Adding a System or Secondary Disk Task Map

Task	Description	For Instructions, Go To
1. Connect the Disk and Boot	*System Disk* Connect the new disk and boot from a local or remote Solaris 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" on page 365 "SPARC: How to Connect a Secondary Disk and Boot" on page 365
2. 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" on page 367
3. 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" on page 371
4. 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.	Chapter 44
5. 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" on page 372

▼ 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 CD.**

If You Are Booting From ...	Then ...
A Solaris 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 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 367.

▼ SPARC: How to Connect a Secondary Disk and Boot

1. **Become superuser.**

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 353 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
```

`-i0`	Brings the system down to init state 0 (zero), the power-down state.
`-g`*n*	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 367.

▼ SPARC: How to Create Disk Slices and Label a Disk

1. **Become superuser.**

2. **Start the `format`(1M) 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
```

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 335 for more information about the free hog slice.

8. **Create a new partition table by answering `y` when prompted to continue.**

```
Do you wish to continue creating a new partition table based on
above table[yes]? y
```

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) and 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 `y` when asked.

```
Okay to make this the current partition table[yes]? y
```

If you don't want the current partition table and you want to change it, answer `no` and go to Step 6 on page 367.

11. Name the partition table.

```
Enter table name (remember quotes): "partition-name"
```

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> q
```

14. Verify the disk label using the `verify` command.

```
format> verify
```

15. Quit the `format` menu.

```
format> q
```

SPARC: Example—Creating Disk Slices and Labeling a System Disk

The following example uses the `format` utility to divide a 1-Gbyte disk into three slices: one for the root (/) file system, one for the swap area, and one for the `/usr` file system.

```
# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
   0. c0t1d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
      /iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd@1,0
   1. c0t3d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
      /iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd@3,0
Specify disk (enter its number): 0
selecting c0t1d0
[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)          0
  1       swap    wu       0               0         (0/0/0)          0
  2     backup    wu       0 - 2035     1002.09MB    (2036/0/0) 2052288
  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       0               0         (0/0/0)          0
  7 unassigned    wm       0               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]? 6
Enter size of partition ‘0’ [0b, 0c, 0.00mb]: 200mb
Enter size of partition ‘1’ [0b, 0c, 0.00mb]: 200mb
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 -  406      200.32MB    (407/0/0)   410256
  1       swap    wu     407 -  813      200.32MB    (407/0/0)   410256
  2     backup    wu       0 - 2035     1002.09MB    (2036/0/0) 2052288
```

(continued)

(Continuation)

```
  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     814 - 2035      601.45MB    (1222/0/0) 1231776
  7 unassigned    wm       0               0         (0/0/0)          0

Okay to make this the current partition table[yes]? yes
Enter table name (remember quotes): "disk0"
Ready to label disk, continue? yes
partition> quit
format> verify
format> quit
```

SPARC: Example—Creating Disk Slices and Labeling a Secondary Disk

The following example uses the `format` utility to divide a 1-Gbyte disk into one slice for the `/export/home` file system.

```
# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
   0. c0t1d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
      /iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd@1,0
   1. c0t3d0 <SUN1.05 cyl 2036 alt 2 hd 14 sec 72>
      /iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd@3,0
Specify disk (enter its number): 0
selecting c0t1d0
[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)          0
  1       swap    wu       0               0         (0/0/0)          0
  2     backup    wu       0 - 2035     1002.09MB    (2036/0/0) 2052288
  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       0               0         (0/0/0)          0
  7 unassigned    wm       0               0         (0/0/0)          0
Do you wish to continue creating a new partition
table based on above table[yes]? y
```

(continued)

```
Free Hog partition[6]? 7
Enter size of partition '0' [0b, 0c, 0.00mb, 0.00gb]:
Enter size of partition '1' [0b, 0c, 0.00mb, 0.00gb]:
Enter size of partition '3' [0b, 0c, 0.00mb, 0.00gb]:
Enter size of partition '4' [0b, 0c, 0.00mb, 0.00gb]:
Enter size of partition '5' [0b, 0c, 0.00mb, 0.00gb]:
Enter size of partition '6' [0b, 0c, 0.00mb, 0.00gb]:
 Part      Tag    Flag     Cylinders        Size            Blocks
  0       root    wm       0               0         (0/0/0)          0
  1       swap    wu       0               0         (0/0/0)          0
  2     backup    wu       0 - 2035     1002.09MB    (2036/0/0) 2052288
  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       0               0         (0/0/0)          0
  7 unassigned    wm       0 - 2035     1002.09MB    (2036/0/0) 2052288
Okay to make this the current partition table[yes]? yes
Enter table name (remember quotes): "home"
Ready to label disk, continue? y
partition> q
format> verify
format> q
#
```

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 371.

▼ SPARC: How to Create File Systems

1. **Become superuser.**

2. **Create a file system for each slice with the** `newfs`**(1M) command.**

```
# newfs /dev/rdsk/cwtxdysz
```

`/dev/rdsk/c`*wtxdysx* — Raw device for the file system to be created.

See Chapter 35 for more information about the `newfs` command.

3. **Verify the new file system by mounting it on an unused mount point.**

```
# mount /dev/dsk/cwtxdysz /mnt
# ls
lost+found
```

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 44. After the root (/) and /usr file systems are restored, install the boot block. Go to "SPARC: How to Install a Boot Block on a System Disk" on page 372.
Secondary Disk	You might need to restore file systems on the new disk. Go to Chapter 44. If you are not restoring file systems on the new disk, you are finished adding a secondary disk. See Chapter 36 for information on making the file systems available to users.

▼ SPARC: How to Install a Boot Block on a System Disk

1. **Become superuser.**

2. **Install a boot block on a system disk using the** `installboot`**(1M) command.**

```
# installboot /usr/platform/‘uname -i‘/lib/fs/ufs/bootblk /dev/rdsk/cwtxdys0
```

`/usr/platform/‘uname -i‘/lib/fs /ufs/bootblk`	Boot block code.
`/dev/rdsk/`*cwtxdy*`s0`	Raw device of the root (/) file system.

3. **Verify the boot blocks are installed by rebooting the system to run level 3.**

```
# 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
```

CHAPTER 31

IA: Adding a Disk (Tasks)

This chapter provides the procedures for adding a disk on an IA based system. This is a list of the step-by-step instructions in this chapter.

For overview information about disk management, see Chapter 28. For step-by-step instructions on adding a disk to a SPARC based system, see Chapter 30.

IA: 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 systems 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.

IA: Adding a System or Secondary Disk Task Map

TABLE 31–1 IA: Adding a System or Secondary Disk Task Map

Task	Description	For Instructions, Go To
1. Connect the Disk and Boot	*System Disk* Connect the new disk and boot from a local or remote Solaris CD. *Secondary Disk* Connect the new disk and perform a reconfiguration boot, so the system will recognize the disk.	"IA: How to Connect a System Disk and Boot" on page 377 "IA: How to Connect a Secondary Disk and Boot" on page 378
2. Create Slices and Label the Disk	Create disk slices and label the disk if it has not already been done by the disk manufacturer.	"IA: How to Create a Solaris `fdisk` Partition" on page 379 and "IA: How to Create Disk Slices and Label a Disk" on page 386
3. 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.	"IA: How to Create File Systems" on page 388
4. 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.	Chapter 44
5. Install Boot Block	*System Disk Only.* Install the boot block on the root (`/`) file system, so the system can boot.	"IA: How to Install a Boot Block on a System Disk" on page 389

IA: 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 might 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).

---

x86 platform only - Solaris slices are sometimes called partitions. This user guide uses the term slice, but some Solaris documentation and programs might refer to a *slice* as a *partition*. To avoid confusion, Solaris documentation tries to distinguish between `fdisk` partitions (which are supported only on Solaris™ (Intel Platform Edition) and the divisions within the Solaris `fdisk` partition, which might be called slices or partitions.

▼ IA: 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 *Solaris 8 (Intel Platform Edition) 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 CD.**

   If you are booting from the network, skip step a.

   a. **Insert the Solaris 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 Boot Solaris screen is displayed after a few minutes.

d. Select the CD-ROM drive or net(work) as the boot device from the Boot Solaris screen.

The Current Boot Parameters screen 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.

IA: Where to Go From Here

After you boot the system, you can create slices and a disk label on the disk. Go to "IA: How to Create Disk Slices and Label a Disk" on page 386.

▼ IA: How to Connect a Secondary Disk and Boot

1. **Become superuser.**

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
```

`–i0`	Brings the system down to init state 0 (zero), the power-down state.
`–g`*n*	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 *Solaris 8 (Intel Platform Edition) Device Configuration Guide* for 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.

IA: Where to Go From Here

After you boot the system, you can create slices and a disk label on the disk. Go to "IA: How to Create Disk Slices and Label a Disk" on page 386.

▼ IA: How to Create a Solaris `fdisk` Partition

1. **Make sure you have read "IA: Guidelines for Creating an `fdisk` Partition" on page 376.**

2. **Become superuser.**

3. **Start the `format`(1M) 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
```

disk-number	Is the number of the disk on which to create a Solaris `fdisk` partition.

5. Go into the `fdisk` menu.

```
format> fdisk
```

The `fdisk` menu displayed is dependent upon whether the disk has existing `fdisk` partitions. Determine the next step using the following table.

If You Want To ...	Go To ...	See ...
Create a Solaris `fdisk` partition to span the entire disk.	Step 6	"IA: Example—Creating a Solaris `fdisk` Partition That Spans the Entire Drive" on page 383
Create a Solaris `fdisk` partition and preserve existing non-Solaris `fdisk` partition(s).	Step 7	"IA: Example—Creating a Solaris `fdisk` Partition and Preserving an Existing `fdisk` Partition" on page 384
Create a Solaris `fdisk` partition and additional non-Solaris `fdisk` partition(s).	Step 7	"IA: Example—Creating a Solaris `fdisk` Partition and an Additional `fdisk` Partition" on page 385

6. Create and activate a Solaris `fdisk` partition spanning the entire disk by specifying `y` at the prompt. Then go to step 14.

```
The recommended default partitioning for your disk is:

  a 100% ''SOLARIS System'' partition.
```

(continued)

(Continuation)

```
To select this, please type ‘‘y’’.  To partition your disk
differently, type ‘‘n’’ and the ‘‘fdisk’’ program will
let you select other partitions. y
```

7. Specify `n` at the prompt if you do not want the Solaris `fdisk` partition to span the entire disk.

```
To select this, please type "y".  To partition your disk
differently, type "n" and the "fdisk" program will let you
select other partitions. n
Total disk size is 2694 cylinders
            Cylinder size is 765 (512 byte) blocks
                                              Cylinders
    Partition   Status    Type      Start   End   Length    %
    =========   ======    ========  =====   ===   ======   ===
THERE ARE NO PARTITIONS CURRENTLY DEFINED 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:
```

8. Select option 1, `Create a partition`, to create an `fdisk` partition.

```
            Total disk size is 2694 cylinders
            Cylinder size is 765 (512 byte) blocks
                                              Cylinders
    Partition   Status    Type      Start   End   Length    %
    =========   ======    ========  =====   ===   ======   ===
THERE ARE NO PARTITIONS CURRENTLY DEFINED 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)
```

(continued)

(Continuation)

```
Enter Selection: 1
```

9. 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
```

10. Identify the percentage of disk to be reserved for the Solaris `fdisk` partition. Keep in mind the size of any existing `fdisk` partitions when calculating this percentage.

```
Indicate the percentage of the disk you want this partition
to use (or enter "c" to specify in cylinders). nn
```

11. Activate the Solaris `fdisk` partition by typing `y` at the prompt.

```
Do you want this to become the Active partition? If so, it will be
activated each time you reset your computer or when you turn it on
again. Please type "y" or "n". y
```

The `Enter Selection:` prompt is displayed after the `fdisk` partition is activated.

12. Select option 1, `Create a partition`, to create another `fdisk` partition.

See steps 9-11 for instructions on creating an `fdisk` partition.

13. Update the disk configuration and exit the `fdisk` menu from the selection menu.

```
Selection: 4
```

14. Relabel the disk using the `label` **command.**

```
WARNING: Solaris fdisk partition changed - Please relabel the disk
format> label
Ready to label disk, continue? yes
format>
```

15. Quit the `format` **menu.**

```
format> quit
```

IA: Where to Go From Here

After you create a Solaris `fdisk` partition on the disk, you can create slices on the disk. Go to "IA: How to Create Disk Slices and Label a Disk" on page 386.

IA: Example—Creating a Solaris `fdisk` Partition That Spans the Entire Drive

The following example uses the `format`'s utility's `fdisk` option to create a Solaris `fdisk` partition that spans the entire drive.

```
# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
       0. c0d0 <DEFAULT cyl 2466 alt 2 hd 16 sec 63>
          /pci@0,0/pci-ide@7,1/ide@0/cmdk@0,0
       1. c0d1 <DEFAULT cyl 522 alt 2 hd 32 sec 63>
          /pci@0,0/pci-ide@7,1/ide@0/cmdk@1,0
       2. c1d0 <DEFAULT cyl 13102 alt 2 hd 16 sec 63>
          /pci@0,0/pci-ide@7,1/ide@1/cmdk@0,0
Specify disk (enter its number): 0
selecting c0d0
Controller working list found
[disk formatted]
format> fdisk
The recommended default partitioning for your disk is:

  a 100% "SOLARIS System" partition.
```

(continued)

(Continuation)

```
To select this, please type "y".  To partition your disk
differently, type "n" and the "fdisk" program will let you
select other partitions. y

WARNING: Solaris fdisk partition changed - Please relabel the disk
format> label
Ready to label disk, continue? yes
format> quit
```

IA: Example—Creating a Solaris `fdisk` Partition and Preserving an Existing `fdisk` Partition

The following example describes how to create a Solaris `fdisk` partition on a disk that has an existing `DOS-BIG` `fdisk` partition.

```
format> fdisk
             Total disk size is 2694 cylinders
             Cylinder size is 765 (512 byte) blocks
                                          Cylinders
     Partition   Status    Type     Start  End   Length    %
     =========   ======   ========  =====   ===   ====== ===
        1                 DOS-BIG       1   538     538   20
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: 1
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
Indicate the percentage of the disk you want this partition
to use (or enter "c" to specify in cylinders). 80
Do you want this to become the Active partition? If so, it will be
activated each time you reset your computer or when you turn it on
again. Please type "y" or "n". y
Partition 2 is now the Active partition Total disk size is 2694
cylinders
             Cylinder size is 765 (512 byte) blocks
                                          Cylinders
    Partition   Status    Type      Start   End   Length    %
    =========   ======    ========  =====   ===   ======   ===
        1                 DOS-BIG       1   538     538     20
        2       Active    SOLARIS     539  2693    2155     80
SELECT ONE OF THE FOLLOWING:
     1.   Create a partition
     2.   Change Active (Boot from) partition
```

(continued)

(Continuation)

```
     3.   Delete a partition
     4.   Exit (Update disk configuration and exit)
     5.   Cancel (Exit without updating disk configuration)
Enter Selection: Selection: 4
WARNING: Solaris fdisk partition changed - Please relabel the disk
format> label
Ready to label disk, continue? yes
format> q
```

IA: Example—Creating a Solaris `fdisk` Partition and an Additional `fdisk` Partition

This following example describes how to create a Solaris `fdisk` partition and a `DOSBIG` `fdisk` partition.

```
format> fdisk
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. n
             Total disk size is 2694 cylinders
             Cylinder size is 765 (512 byte) blocks
                                       Cylinders
    Partition   Status    Type      Start   End   Length    %
    =========   ======    ========  =====   ===   ======   ===
THERE ARE NO PARTITIONS CURRENTLY DEFINED 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: 1
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) ?8
Indicate the percentage of the disk you want this partition
to use (or enter "c" to specify in cylinders). 20
Do you want this to become the Active partition? If so, it will be
activated each time you reset your computer or when you turn it on
again. Please type "y" or "n". n
             Total disk size is 2694 cylinders
             Cylinder size is 765 (512 byte) blocks
                                       Cylinders
    Partition   Status    Type      Start   End   Length    %
    =========   ======    ========  =====   ===   ======   ===
        1                 DOS-BIG       1   538     538     20
SELECT ONE OF THE FOLLOWING:
```

(continued)

(Continuation)

```
     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: 1
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
Indicate the percentage of the disk you want this partition
to use (or enter "c" to specify in cylinders). 80
Do you want this to become the Active partition? If so, it will be
activated each time you reset your computer or when you turn it on
again. Please type "y" or "n". y
Partition 2 is now the Active partition Total disk size is 2694
cylinders
            Cylinder size is 765 (512 byte) blocks
                                       Cylinders
    Partition   Status    Type      Start   End   Length    %
    =========   ======    ========  =====   ===   ======   ===
        1                 DOS-BIG       1   538      538     20
        2       Active    SOLARIS     539  2693     2155     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: 4
format> q
```

▼ IA: How to Create Disk Slices and Label a Disk

1. **Become superuser.**

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
```

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 335 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) and 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 on page 387.

11. Name the partition table.

```
Enter table name (remember quotes): "partition-name"
```

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. Verify the new disk label with verify **command.**

```
format> verify
```

15. Quit the format **menu.**

```
format> quit
```

IA: Where to Go From Here

After you create disk slices and label the disk, you can create file systems on the disk. Go to "IA: How to Create File Systems" on page 388.

▼ IA: How to Create File Systems

1. Become superuser.

2. Create a file system for each slice with the newfs**(1M) command.**

```
# newfs /dev/rdsk/cwtxdysz
```

`/dev/rdsk/cwtxdysz`	Raw device for the file system to be created.

See Chapter 36 for more information about the `newfs` command.

3. Verify the new file system by mounting it on an unused mount point.

```
# mount /dev/dsk/cwtxdysz /mnt
# ls /mnt
lost+found
```

IA: 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 44. After the root (`/`) and `/usr` file systems are restored, install the boot block. Go to "IA: How to Install a Boot Block on a System Disk" on page 389.
Secondary Disk	You might need to restore file systems on the new disk. Go to Chapter 44. If you are not restoring file systems on the new disk, you are finished adding a secondary disk. See Chapter 36 for information on making the file systems available to users.

▼ IA: How to Install a Boot Block on a System Disk

1. Become superuser.

2. Install the boot block.

```
# installboot /usr/platform/‘uname -i‘/lib/fs/ufs/pboot /usr/platform/‘uname -i‘
/lib/fs/ufs/bootblk /dev/rdsk/cwtxdys2
```

`/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/`*cwtxdy*`s2`	Is the raw device name that represents the whole disk.

3. Verify the boot blocks are installed by rebooting the system to run level 3.

```
# init 6
```

IA: Example—Installing a Boot Block on a System Disk

```
# installboot /usr/platform/i86pc/lib/fs/ufs/pboot
 /usr/platform/i86pc/lib/fs/ufs/bootblk /dev/rdsk/c0t6d0s2
```

CHAPTER 32

The format Utility (Reference)

This chapter describes the format utility's menu and commands.

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

See Chapter 28 for a conceptual overview of when to use the format utility.

Requirements or Restrictions for Using the format Utility

You must be superuser to use the format utility. If you are not superuser, 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 `format`'s `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
        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>
```

The table below describes the `format` main menu items.

TABLE 32–1 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 394 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: 1. Information found in the `format.dat` file 2. Information from the automatic configuration process 3. Information you enter at the prompt 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 396 for more information.
`defect`	Menu	Retrieve and print defect lists. See "The `defect` Menu" on page 398 for more information.
`backup`	Command	Search for backup labels.

TABLE 32–1 The `format` Main Menu Item Descriptions *(continued)*

Item	Command or Menu?	Allows You To ...
`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.
`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>
```

The table below describes the `partition` menu items.

TABLE 32–2 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.
`print`	View the current slice table.
`label`	Write the slice map and label to the current disk.
`quit`	Exit the partition menu.

IA: The `fdisk` Menu

The `fdisk` menu appears on IA based 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)
```

(continued)

(Continuation)

```
Enter Selection:
```

The table below describes the `fdisk` menu items.

TABLE 32–3 IA: 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 environment 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 stage 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)
```

(continued)

```
    print    - display data buffer
    setup    - set analysis parameters
    config   - show analysis parameters
    quit
analyze>
```

The table below describes the `analyze` menu items.

TABLE 32–4 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.
`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.

TABLE 32–4 The `analyze` Menu Item Descriptions *(continued)*

The Command ...	Allows You To ...
`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>
```

The table below describes the `defect` menu items.

TABLE 32–5 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 file, `/etc/format.dat`, contains:

- Disk types
- Default slice tables

The `format.dat` file shipped with the Solaris operating environment 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 environment
- A disk with a slice table that is different from the Solaris operating environment 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 of the `format.dat` File

The `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 of the `format.dat` File

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 in the `format.dat` File

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 the table below.

TABLE 32–6 `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 searches 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 environment 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 recreated slices on any of the disks on your system. Add as many slice definitions to the data file as you need.

Disk Type (`format.dat`)

`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 environment 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. The table below describes the identifiers that must also be assigned values in all `disk_type` definitions.

TABLE 32–7 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.

TABLE 32–7 Required `disk_type` Identifiers *(continued)*

Identifier	Description
`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 might be necessary depending on the controller. The table below describes the assignments required for SCSI controllers.

TABLE 32–8 `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 = "SUN1.3G" \
        : ctlr = SCSI : fmt_time = 4 \
        : trks_zone = 17 : asect = 6 : atrks = 17 \
        : ncyl = 1965 : acyl = 2 : pcyl = 3500 : nhead = 17 : nsect = 80 \
        : rpm = 5400 : bpt = 44823

disk_type = "SUN2.1G" \
        : ctlr = SCSI : fmt_time = 4 \
        : ncyl = 2733 : acyl = 2 : pcyl = 3500 : nhead = 19 : nsect = 80 \
        : rpm = 5400 : bpt = 44823

disk_type = "SUN2.9G" \
        : ctlr = SCSI : fmt_time = 4 \
        : ncyl = 2734 : acyl = 2 : pcyl = 3500 : nhead = 21 : nsect = 99 \
        : rpm = 5400
```

Partition or Slice Tables (`format.dat`)

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. The table below describes the identifiers that must be assigned values in all slice tables.

TABLE 32–9 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 = "SUN1.3G" \
        : disk = "SUN1.3G" : ctlr = SCSI \
        : 0 = 0, 34000 : 1 = 25, 133280 : 2 = 0, 2672400 : 6 = 123, 2505120

partition = "SUN2.1G" \
        : disk = "SUN2.1G" : ctlr = SCSI \
```

(continued)

(Continuation)

```
        : 0 = 0, 62320 : 1 = 41, 197600 : 2 = 0, 4154160 : 6 = 171, 3894240

partition = "SUN2.9G" \
        : disk = "SUN2.9G" : ctlr = SCSI \
        : 0 = 0, 195426 : 1 = 94, 390852 : 2 = 0, 5683986 : 6 = 282, 5097708
```

Specifing the Location of a `format` Data File

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 environment and should always be present.

Rules for Input to `format` Commands

When using the `format` utility, you need to provide various kinds of information. This section describes the rules for this information. See “Using `format` Help” on page 406 for information on using `format`’s help facility when inputting data.

Inputting Numbers to `format` Commands

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
```

Specifying Block Numbers to `format` Commands

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 by the cylinder/head/sector designation. In this method, 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: 0xabcd
Enter defective block number: 334/$/2
Enter defective block number: 892//$
```

The `format` utility 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.

Specifying `format` 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 you enter 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
```

Specifying Disk Names to `format` Commands

There are certain times in `format` when you must name something. In these cases, you are free to specify any string you want 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.

Using `format` 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 `format` 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.

CHAPTER 33

Managing File Systems Topics

This section provides instructions for managing file systems in the Solaris operating environment. This section contains these chapters.

Chapter 34	Provides a high-level overview of file system concepts, including descriptions of the types of file systems, commonly used administration commands, and the basics of mounting and unmounting file systems.
Chapter 35	Provides step-by-step procedures to create a UFS file system, create and preserve a temporary file system (TMPFS), and create a loopback file system (LOFS).
Chapter 36	Provides step-by-step procedures to determine what file systems are mounted, how to mount files listed in the `/etc/vfstab` file, and how to mount UFS, NFS, and PCFS (DOS) file systems.
Chapter 37	Provides overview information and step-by-step instructions for using the Cache File System (CacheFS™).
Chapter 38	Provides step-by-step procedures for configuring additional swap space, monitoring swap resources, creating swap files and making them available, and removing extra swap space.

Chapter 39	Provides information on how the file system state is recorded, what is checked by the `fsck` program, how to modify automatic boot checking, and how to use the `fsck` program.
Chapter 40	Provides file system reference information, including default directories for the root (`/`) and `/usr` file systems, default directories contained within the `/kernel` directory, and specifics for the `mkfs` and `newfs` commands.

CHAPTER 34

Managing File Systems (Overview)

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

What's New in File Systems?

This section describes new file system features.

The /var/run File System

A new TMPFS-mounted file system, `/var/run`, is the repository for temporary system files that are not needed across system reboots in this Solaris release and future releases. The `/tmp` directory continues to be repository for non-system temporary files.

Because `/var/run` is mounted as a memory-based file system rather than a disk-based file system, updates to this directory do not cause unnecessary disk traffic that would interfere with systems running power management software.

The `/var/run` directory requires no administration. You may notice that it is not unmounted with the `umount -a` or the `umountall` command.

For security reasons, `/var/run` is owned by root.

Mount Table Changes (`/etc/mnttab`)

In previous Solaris releases, `/etc/mnttab` was a text-based file that stored information about mounted file systems. The downside of being a file was that it could get out of sync with the actual state of mounted file systems.

Now the `/etc/mnttab` file is a MNTFS file system that provides read-only information directly from the kernel about mounted file systems for the local system.

Note the following `mnttab` behavior changes:

- Programs or scripts attempting to write to `/etc/mnttab` will fail.
- The `mount -m` option for faking `mnttab` entries no longer works.

No administration is required for the `/etc/mnttab` mount table.

See `mnttab`(4) for more information.

Using the Universal Disk Format (UDF) File System

The UDF file system, the industry-standard format for storing information on the optical media technology called *DVD* (Digital Versatile Disc or Digital Video Disc), is included in this Solaris release.

The UDF file system is provided as dynamically loadable, 32–bit and 64–bit modules, with system administration utilities for creating, mounting, and checking the file system on both SPARC and IA platforms. The Solaris UDF file system works with supported ATAPI and SCSI DVD drives, CD-ROM devices, and disk and diskette drives. In addition, the Solaris UDF file system is fully compliant with the UDF 1.50 specification.

The UDF file system support is provided in the following new packages:

- `SUNWudfr` — 32–bit kernel component
- `SUNWudfrx` — 64–bit kernel component

- SUNWudf — /usr component

UDF Features and Benefits

In this Solaris release, the UDF file system provides the following features:

- Ability to access the industry standard CD-ROM and DVD-ROM media when they contain a UDF file system.
- Flexibility in exchanging information across platforms and operating systems.
- A mechanism for implementing new applications rich in broadcast-quality video, high-quality sound along with the richness in interactivity using the DVD video specification based on UDF format.

The following features are not included in this UDF file system release:

- Support for write-once media, CD-RW, and DVD-RAM, with either the sequential disk-at-once and incremental recording.
- UFS components such as quotas, ACLs, transaction logging, file system locking, and file system threads, which are not part of the UDF 1.50 specification.

Hardware and Software Requirements

The UDF file system requires the following:

- The Solaris 7 11/99 or the Solaris 8 release
- Supported SPARC or Intel platforms
- Supported CD-ROM or DVD-ROM device

UDF Compatibility Issues

This first Solaris UDF file system implementation provides:

- Support for industry-standard read-write UDF version 1.50.
- Fully internationalized file system utilities.

▼ How to Connect a DVD-ROM Device

1. **Become superuser.**
2. **Create the** `/reconfigure` **file.**

```
# touch /reconfigure
```

3. **Shut down the system and turn off power.**

```
# init 0
```

4. **Connect the DVD-ROM device.**

5. **Turn on power to the system.**

▼ How to Access Files on a DVD-ROM Device

1. **Verify the DVD-ROM device is automatically mounted.**

```
$ ls /cdrom
```

Note - If the system has both a CD-ROM and DVD-ROM device, the CD-ROM might be named `/cdrom/cdrom0` and the DVD-ROM might be named `/cdrom/cdrom1`. If the system only has a DVD-ROM device, then try using `/cdrom/cdrom0`.

---

2. **Display content with `ls` command.**

```
$ ls /cdrom/cdrom1
Copyright  filea  fileb
```

Automatic display with the CDE file manager is not implemented yet. All other CDE file manager functions—drag and drop for copying and imagetool features—are available.

▼ How to Display UDF File System Parameters

Display UDF file system parameters by using the `mkfs` command.

1. **Become superuser.**

2. **Display UDF file system parameters.**

```
# mkfs -F udfs -m /dev/rdsk/device-name
```

▼ How to Create a UDF File System

Create a UDF file system by using the `mkfs` command.

1. **Become superuser.**
2. **Create a UDF file system.**

```
# mkfs -F udfs /dev/rdsk/device-name
```

3. **Verify the UDF file system is created by mounting it. See "How to Mount a UDF File System" on page 414 for more information.**

See `mkfs_udfs`(1M) for more information.

▼ How to Identify the UDF File System Type

Identify the UDF file system type by using the `fstyp` command.

1. **Become superuser.**
2. **Determine whether a file system is a UDF file system.**

```
# fstyp -v /rdev/dsk/device-name
```

▼ How to Check a UDF File System

Check the integrity of a UDF file system by using the `fsck` command.

1. **Become superuser.**
2. **Check a UDF file system.**

```
# fsck -F udfs /dev/rdsk/device-name
```

See `fsck_udfs`(1M) for more information.

▼ How to Mount a UDF File System

Mount a UDF file system.

1. **Become superuser.**
2. **Mount a UDF file system.**

```
# mount -F udfs /dev/dsk/device-name /mount-point
```

3. **Verify the UDF file system is mounted.**

```
# ls /mount-point
```

See `mount_udfs`(1M) for more information.

▼ How to Unmount a UDF File System

Unmount a UDF file system.

1. **Become superuser.**
2. **Unmount a UDF file system.**

```
# umount /dev/dsk/device-name
```

▼ How to Label a Device with a UDF File System and Volume Name

Create a file system and volume name for a UDF file system.

1. **Become superuser.**
2. **Create a file system and volume name for the UDF file system.**

```
# labelit -F UDFS /dev/rdsk/device-name fsname volume
```

See `labelit_udfs`(1M) for more information.

Overview of File Systems

A file system is a structure of directories used to organize and store files. The term *file system* is used to describe:

- A particular type of file system: disk-based, network-based, or virtual
- The entire file tree from the root directory downward
- The data structure of a disk slice or other media storage device
- 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 operating environment 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; and makes it easier to add new file systems.

Administering file systems is one of your most important system administration tasks. Read this chapter for file system background and planning information. Refer to other chapters in the *System Administration Guide* for instructions about the following tasks:

For This Task ...	**See ...**
Creating new file systems	Chapter 35 and Chapter 37
Making local and remote files available to users	Chapter 36
Connecting and configuring new disk devices	Chapter 28
Designing and implementing a backup schedule and restoring files and file systems as needed	Chapter 42
Checking for and correcting file system damage	Chapter 39

Types of File Systems

The Solaris operating environment 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 430.

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:

Disk-Based File System	Format Description
UFS	UNIX file system (based on the BSD Fast File system that was provided in the 4.3 Tahoe release). UFS is the default disk-based file system for the Solaris operating environment. Before you can create a UFS file system on a disk, the disk must be formatted and divided into slices. See Chapter 28 for complete information on formatting disks and dividing disks into slices.
HSFS	High Sierra, Rock Ridge, 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-ROMs, 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 features 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.
UDF	The UDF file system, the industry-standard format for storing information on the optical media technology called DVD (Digital Versatile Disc or Digital Video Disc).

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
- UDF with DVD

These associations are not, however, restrictive. For example, CD-ROMs and diskettes can have UFS file systems created on them.

Network-Based File Systems

Network-based file systems can be accessed over the network. Typically, network-based file systems reside on one system, typically a server, and are accessed by other systems across the network. NFS™ is the only available network-based or distributed computing file system.

With NFS, you can administer distributed *resources* (files or directories) by exporting them from a server and mounting them on individual clients. See "The NFS Environment" on page 428 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, the Cache File System (CacheFS) uses a file system on the disk to contain the cache, and some virtual file systems, such as the Temporary File System (TMPFS), 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 37 for detailed information on setting up and administering CacheFS File Systems.

The Temporary File System

The Temporary File System (TMPFS) uses local memory for file system reads and writes, which is typically much faster than a UFS file system. Using TMPFS 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, and the operating system generates a lot of disk or

network activity while manipulating these files. Using TMPFS to hold these temporary files can significantly speed up their creation, manipulation, and deletion.

Files in TMPFS file systems 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 operating environment. You can copy or move files into or out of the `/tmp` directory, just as you would in a UFS file system.

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 might not execute because there is not enough swap space.

See Chapter 35 for information about creating TMPFS file systems. See Chapter 38 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 root (`/`) on `/tmp/newroot`, which will make the entire file system hierarchy look like it is duplicated under `/tmp/newroot`, including any file systems mounted from NFS servers. All files will be accessible either with a path name starting from root (`/`), or with a path name starting from `/tmp/newroot`.

See Chapter 35 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.

Caution - Do not delete the files in the `/proc` directory. Deleting processes from the `/proc` directory will not 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.

Virtual File System	Description
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 both a generic and a file system–specific component. You should use the generic commands whenever possible, which call the file system–specific component. The table below lists the generic file system administrative commands, which are located in the `/usr/sbin` directory.

TABLE 34–1 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

TABLE 34–1 Generic File System Administrative Commands *(continued)*

Command	Description
`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 local and remote file systems
`mountall`(1M)	Mounts all file systems specified in the virtual file system table (`/etc/vfstab`)
`ncheck`(1M)	Generates a list of path names with their i-numbers
`umount`(1M)	Unmounts local and remote file systems
`umountall`(1M)	Unmounts all file systems specified in a virtual file system table (`/etc/vfstab`)
`volcopy`(1M)	Makes an image copy of a file system

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 the –F option, if supplied.
2. By matching a special device with an entry in /etc/vfstab file (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 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 a UFS file system, type **man mount_ufs**.

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 Solaris installation process enables you to install a default set of directories and uses a set of conventions to group similar types of files together. The table below provides a summary of the default Solaris file systems, and shows the type of each file system.

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 for more detailed information on the default directories for the root (/) and /usr file systems.

TABLE 34–2 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 might 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 might be a UFS file system on a local disk slice.
/tmp	TMPFS	Temporary files, cleared each time the system is booted or the /tmp file system is unmounted.
/proc	PROCFS	A list of active processes, by number.

Swap Space

The Solaris operating environment uses some disk slices for temporary storage rather than for file systems. These slices are called *swap* slices, or *swap space*. Swap space is used as virtual memory storage areas when the system does not have enough physical memory to handle current processes.

Since many applications rely on swap space, it is important to know how to plan for, monitor, and add more swap space when needed. For an overview about swap space and instructions for adding swap space, see Chapter 38.

The UFS File System

UFS is the default disk-based file system in Solaris operating environment. Most of the time, when you administer a disk-based file system, you will be administering UFS file systems. UFS provides the following features:

UFS Feature	Description
State flags	Show the state of the file system: clean, stable, active, logging, or unknown. These flags eliminate unnecessary file system checks. If the file system is "clean," "stable," or "logging," 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). The Solaris operating environment 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 Sun, provides this capability.
Large files	By default, a UFS file system can have regular files larger than 2 Gbytes (gigabytes). You must explicitly use the `nolargefiles` mount option to enforce a 2 Gbyte maximum file size limit. This limit was removed in the Solaris 2.6 release.

See Chapter 40 for detailed information about the UFS file system.

Parts of a UFS File System

When you create a UFS file system, the disk slice is divided into *cylinder groups*, which are made up of one or more consecutive disk cylinders. The cylinder groups are then further divided into addressable 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. See "The Structure of UFS File System Cylinder Groups" on page 538 for more detailed information about each type of block.

If you want to customize a file system using arguments with the `newfs` command or the `mkfs` command, see Chapter 40 for information about altering these parameters.

UFS Logging

UFS logging is the process of storing transactions (changes that make up a complete UFS operation) in a log before the transactions are applied to the UFS file system. Once a transaction is stored, the transaction can be applied to the file system later.

At reboot, the system discards incomplete transactions, but applies the transactions for completed operations. The file system remains consistent because only completed transactions are ever applied. This is true even when a system crashes, which normally interrupts system calls and introduces inconsistencies into a UFS file system.

UFS logging provides two advantages. It prevents file systems from becoming inconsistent, therefore eliminating the need to run `fsck`. And, because `fsck` can be bypassed, UFS logging reduces the time required to reboot a system if it crashes, or after an unclean halt (see "What `fsck` Checks and Tries to Repair" on page 510 for details on unclean halts). UFS logging can significantly reduce the boot time on systems that have large file systems, which usually take a long time to read and verify with `fsck`.

The log created by UFS logging is continually flushed as it fills up. The log is totally flushed when the file system is unmounted or as a result of the `lockfs -f` command.

UFS logging is not enabled by default. To enable UFS logging, you must specify the `–o logging` option with the `mount` command in the `/etc/vfstab` file or when mounting the file system. The log is allocated from free blocks on the file system, and it is sized approximately 1 Mbyte per 1 Gbyte of file system, up to a maximum of 64 Mbytes. Logging can be enabled on any UFS, including the root (`/`) file system. Also, the `fsdb` command has been updated with new debugging commands to support UFS logging.

Planning UFS File Systems

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 `/export/home` and swap space 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 should have three file systems: `/`, `/usr`, and swap space. On other disks, create one or, at most, two file systems; one being additional swap space, preferably.

Fewer, roomier file systems cause less file fragmentation than many small, over-crowded file systems. 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 35 for information on default file system parameters as well as procedures for creating new UFS file systems.

Mounting and Unmounting File Systems

Before you can access the files on a file system, you need to mount the file system. Mounting a file system attaches that file system to a directory (*mount point*) and makes it 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.

When you mount a file system, any files or directories in the underlying mount point directory are unavailable as long as the file system is mounted. These files are not permanently affected by the mounting process, and they become available again when the file system is unmounted. However, mount directories are typically empty, because you usually do not want to obscure existing files.

For example, the figure below shows a local file system, starting with a root (/) file system and subdirectories `sbin`, `etc`, and `opt`.

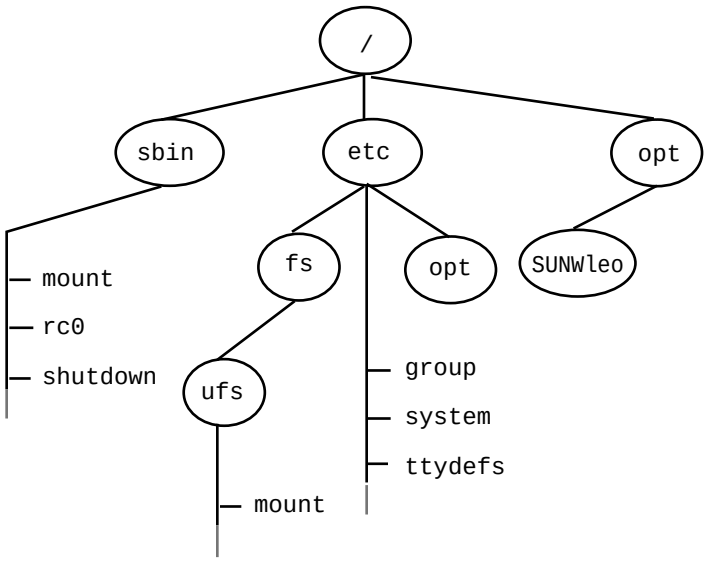


Figure 34–1 Sample root (/) File System

Now, say you wanted to access a local file system from the `/opt` file system that contains a set of unbundled products.

First, you must create a directory to use as a mount point for the file system you want to mount, for example, `/opt/unbundled`. Once the mount point is created, you can mount the file system (by using the `mount` command), which makes all of the files and directories in `/opt/unbundled` available, as shown in the figure below. See Chapter 36 for detailed instructions on how to perform these tasks.

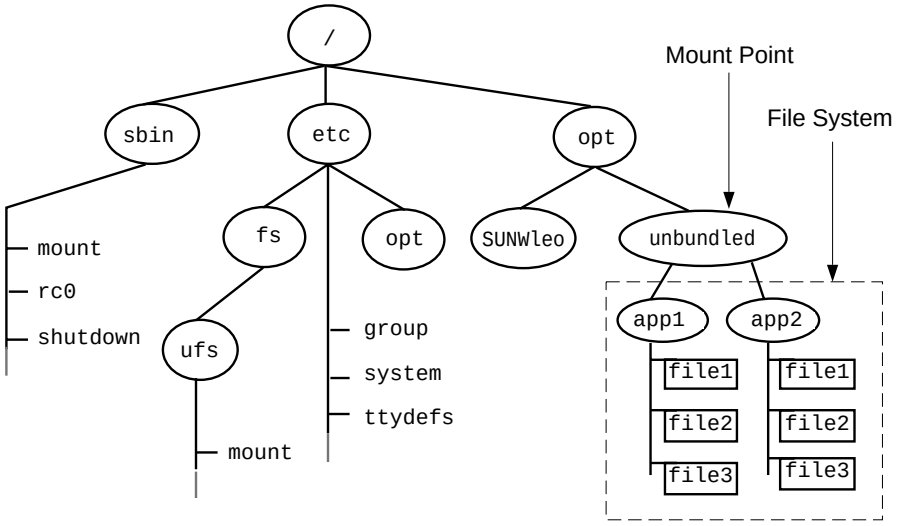


Figure 34–2 Mounting a File System

The Mounted File System Table

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 this file with the `cat` or `more` commands, but you cannot edit it. Here is an example of an `/etc/mnttab` file:

```
$ more /etc/mnttab
/dev/dsk/c0t0d0s0  /  ufs rw,intr,largefiles,onerror=panic,suid,dev=2200000 938557523
/proc   /proc   proc    dev=3180000     938557522
fd      /dev/fd fd      rw,suid,dev=3240000     938557524
mnttab  /etc/mnttab     mntfs   dev=3340000     938557526
swap    /var/run        tmpfs   dev=1   938557526
swap    /tmp    tmpfs   dev=2   938557529
/dev/dsk/c0t0d0s7 /export/
home ufs rw,intr,largefiles,onerror=panic,suid,dev=2200007 938557529
$
```

The Virtual File System Table

It would be a very time-consuming and error-prone task to manually mount file systems every time you wanted to access them. To fix this, the virtual file system table (the `/etc/vstab` file) was created to maintain a list of file systems and how to mount them. The `/etc/vfstab` file provides two important features: you can specify file systems to automatically mount when the system boots, and you can mount file systems by using only the mount point name, because the `/etc/vfstab` file contains the mapping between the mount point and the actual device slice name.

A default `/etc/vfstab` file is created when you install a system depending on the selections you make when installing system software; however, you can edit the `/etc/vfstab` file on a system whenever you want. To add an entry, the main information you need to specify is the device where the file system resides, the name of the mount point, the type of the file system, whether you want it to mount automatically when the system boots (by using the `mountall` command), and any mount options.

The following is an example of an `/etc/vfstab` file. Comment lines begin with #. This example shows an `/etc/vfstab` file for a system with two disks (`c0t0d0` and `c0t3d0`).

```
$ 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 /test      ufs     2       yes     -
```

(continued)

```
$
```

In the above example, the last entry specifies that a UFS file system on the `/dev/dsk/c0t3d0s7` slice will be automatically mounted on the `/test` mount point when the system boots. Note that, for root (`/`) and `/usr`, the `mount at boot` field value is specified as `no`, because these file systems are mounted by the kernel as part of the boot sequence before the `mountall` command is run.

See Chapter 36 for descriptions of each of the `/etc/vfstab` fields and information on how to edit and use the file.

The NFS Environment

NFS is a distributed file system service that can be used to share *resources* (files or directories) from one system, typically a server, with other systems across the network. For example, you might want to share third-party applications or source files with users on other systems.

NFS makes the actual physical location of the resource irrelevant to the user. 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 files are virtually indistinguishable from local ones.

A system becomes an NFS server if it has resources to share over the network. A server keeps a list of currently shared resources and their access restrictions (such as read / write or read-only).

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 and rebooting the system

See Chapter 36 for information on how to share resources. See *System Administration Guide, Volume 3* for a complete description of NFS.

AutoFS

You can mount NFS file system resources by using a client-side service called automounting (or AutoFS), which enables a system to automatically mount and unmount NFS resources whenever you access them. The resource remains mounted as long as you remain in the directory and are using a file. If the resource is not accessed for a certain period of time, it is automatically unmounted.

AutoFS provides the following features:

- NFS resources don't need to be mounted when the system boots, which saves booting time.
- Users don't need to know the root password to mount and unmount NFS resources.
- Network traffic might be reduced, since NFS resources are only mounted when they are in use.

The AutoFS service is initialized by `automount`, which is run automatically when a system is booted. The automount daemon, `automountd`, runs continuously and is responsible for the mounting and unmounting of the NFS file systems on an as-needed basis. By default, the Solaris operating environment automounts `/home`.

AutoFS works with file systems specified in the name service. This information can be maintained in NIS, NIS+, or local `/etc` files. With AutoFS, you can specify multiple servers to provide the same file system. 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.

See *System Administration Guide, Volume 3* for complete information on how to set up and administer AutoFS.

The Cache File System (CacheFS)

If you want to improve the performance and scalability of an NFS or CD-ROM file system, you should use the Cache File System (CacheFS). 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). You can also combine CacheFS with the AutoFS service to help boost performance and scalability.

See Chapter 37 for detailed information about CacheFS.

Deciding How to Mount File Systems

The table below provides guidelines on mounting file systems based on how you use them.

TABLE 34–3 Determining How to Mount File Systems

If You Need to Mount ...	Then You Should Use ...
Local or remote file systems infrequently	The `mount` command entered manually from the command line.
Local file systems frequently	The `/etc/vfstab` file, which will mount the file system automatically when the system is booted in multi-user state.
Remote file systems frequently, such as home directories	■ The `/etc/vfstab` file, which will automatically mount the file system when the system is booted in multi-user state. ■ AutoFS, which will automatically mount or unmount the file system when you change into (mount) or out of (unmount) the directory. To enhance performance, you can also cache the remote file systems by using CacheFS.

You can mount a CD-ROM containing a file system by simply inserting it into the drive (Volume Management will automatically mount it). You can mount a diskette containing a file system by inserting it into the drive and running the `volcheck` command. See Chapter 14 for more information.

Determining a File System's Type

You can determine a file system's type by using the following:

- The `FS type` field in the virtual file system table (`/etc/vfstab` file)
- The `/etc/default/fs` file for local file systems
- The `/etc/dfs/fstypes` file for NFS file systems

▼ How to Determine a File System's Type

This procedure works whether the file system is mounted or not.

Determine a file system's type by using the `grep` command.

```
$ grep mount-point fs-table
```

mount-point	Specifies the mount point name of the file system for which you want to know the type. For example, the `/var` directory.
fs-table	Specifies the absolute path to the file system table in which to search for the file system's type. If the file system is mounted, *fs-table* should be `/etc/mnttab`. If it isn't mounted, *fs-table* should be `/etc/vfstab`.

Information for the mount point is displayed.

Note - If you have the raw device name of a disk slice, you can use the `fstyp`(1M) command to determine a file system's type (if the disk slice contains a file system).

Example—Determining a File System's Type

The following example uses the `/etc/vfstab` to determine the type of the `/export` file system.

```
$ grep /export /etc/vfstab
/dev/dsk/c0t3d0s6   /dev/rdsk/c0t3d0s6  /export ufs   2       yes    -
$
```

The following example uses the `/etc/mnttab` file to determine the file system type of the currently mounted diskette (mounted by volume management).

```
$ grep /floppy /etc/mnttab
/vol/dev/diskette0/unnamed_floppy   /floppy/
unnamed_floppy  pcfs rw,nohidden,
nofoldcase,dev=16c0009      89103376
$
```

CHAPTER 35

Creating File Systems (Tasks)

This chapter describes how to create UFS, TMPFS, and LOFS file systems. For UFS file systems, this chapter shows you how to create a file system on a hard disk using the `newfs` command. Because TMPFS and LOFS are virtual file systems, you actually "access" them by mounting them.

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

Note - For instructions on how to create UFS and DOS file systems on removable media, see Chapter 14.

Creating a UFS File System

Before you can create a UFS file system on a disk, the disk must be formatted and divided into slices. A disk slice is a physical subset of a disk that is composed of a single range of contiguous blocks. A slice can be used either as a raw device that provides, for example, swap space, or to hold a disk-based file system. See Chapter 28 for complete information on formatting disks and dividing disks into slices.

Logical volume management products, like Solstice DiskSuite, create more sophisticated *meta devices*, that expand beyond single slice or single disk boundaries. See *Solstice DiskSuite 4.2.1 User's Guide* for more information about meta devices.

Note - Solaris device names use the term slice (and the letter `s` in the device name) to refer to the slice number. Slices are also called “partitions.”

You need to create UFS file systems only occasionally, because the Solaris operating environment automatically creates them 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 of a file system

The `newfs` command is the standard way to create UFS file systems. The `newfs`(1M) command is a convenient front-end to the `mkfs`(1M) command, which actually creates the new file system. The `newfs` command reads parameter defaults, such as tracks per cylinder and sectors per track, 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. The table below shows the default parameters used by the `newfs` command.

TABLE 35–1 Default Parameters Used by the `newfs` Command

Parameter	Default Value
Block size	8 Kbytes
Fragment size	1 Kbyte
Minimum free space	((64 Mbytes / partition size) * 100), rounded down to the nearest integer and limited to between 1% and 10%, inclusively
Rotational delay	Zero
Optimization type	Time
Number of inodes	1 for each 2 Kbytes of data space

▼ How to Create a UFS File System

1. **Make sure you have met the following prerequisites:**
   - The disk must be formatted and divided into slices before you can create UFS file systems on it. See Chapter 28 for complete information on formatting disks and dividing disks into slices.
   - You need to know the device name of the slice that will contain the file system. See Chapter 29 for information on finding disks and disk slice numbers.
   - If you are re-creating an existing UFS file system, unmount it.
   - You must be superuser.

2. **Create the UFS file system.**

```
# newfs [-N] [-b size] [-i bytes] /dev/rdsk/device-name
```

–N	Displays what parameters `newfs` would pass to `mkfs` without actually creating the file system. This is a good way to test the `newfs` command.
–b *size*	Specifies the file system block size, either 4096 or 8192 bytes per block. The default is 8192.
–i *bytes*	Specifies the number of bytes per inode. The default varies depending on the disk size. See `newfs`(1M) for more information.
device-name	Specifies the disk device name on which to create the new file system.

The system asks for confirmation.

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. This might cause the system to panic.

3. **To verify the creation of the UFS file system, check the new file system with the `fsck`(1M) command.**

```
# fsck /dev/rdsk/device-name
```

device-name	Specifies the name of the disk device containing the new file system.

The `fsck` command checks the consistency of the new file system, reports problems it finds, and prompts you before repairing the problems. See Chapter 39 for more information on `fsck`.

Example—Creating a UFS File System

The following example creates a UFS file system on `/dev/rdsk/c0t1d0s7`.

```
# newfs /dev/rdsk/c0t1d0s7
/dev/rdsk/c0t1d0s7:  725760 sectors in 720 cylinders of 14 tracks, 72 sectors
        354.4MB in 45 cyl groups (16 c/g, 7.88MB/g, 3776 i/g)
super-block backups (for fsck -F ufs -o b=#) at:
 32, 16240, 32448, 48656, 64864, 81072, 97280, 113488, 129696, 145904, 162112,
 178320, 194528, 210736, 226944, 243152, 258080, 274288, 290496, 306704,
 322912, 339120, 355328, 371536, 387744, 403952, 420160, 436368, 452576,
 468784, 484992, 501200, 516128, 532336, 548544, 564752, 580960, 597168,
 613376, 629584, 645792, 662000, 678208, 694416, 710624,
#
```

Where to Go From Here

To mount the file system and make it available, go to Chapter 36.

Creating a Temporary File System (TMPFS)

The Temporary File System (TMPFS) uses local memory for file system reads and writes, which is typically much faster than a UFS 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. Files in TMPFS file systems do not survive across reboots or unmounts.

If you create multiple TMPFS file systems, be aware that they all use the same system resources. Files created under one TMPFS file system use up the space available for any other TMPFS, unless you limit TMPFS sizes using the `–o size` option of the `mount` command.

See the `tmpfs`(7FS) man page for more information.

▼ How to Create a TMPFS File System

1. **Become superuser.**

2. **If necessary, create the directory where you want to mount the TMPFS file system and set permissions and ownership as necessary.**

3. **Create a TMPFS file system.**

   To set up the system to automatically create a TMPFS file system when it boots, see "Example—Creating a TMPFS File System at Boot Time" on page 437.

```
# mount -F tmpfs [-o size=number ] number swap mount-point
```

–o size=*number*	Specifies the size of the TMPFS file system in Mbytes.
mount-point	The directory on which the TMPFS file system is mounted.

4. **Look at the output from the** `mount` **command to verify that the TMPFS file system has been created.**

```
# mount -v
```

Example—Creating a TMPFS File System

The following example creates a new directory, `/export/reports`, and mounts a TMPFS file system at that point, limiting it to 50 Mbytes.

```
# mkdir /export/reports
# chmod 777 /export/reports
# mount -F tmpfs -o size=50 swap /export/reports
```

Example—Creating a TMPFS File System at Boot Time

You can set up the system to automatically create a TMPFS file system when it boots by adding an entry to the `/etc/vfstab` file. The following example shows an entry in the `/etc/vfstab` file that will create a TMPFS file system on `/export/test` when the system boots. Since the `size=`*number* option is not specified, the size of the TMPFS file system on `/export/test` is limited only by the available system resources.

```
swap - /export/test  tmpfs   -  yes  -
```

For more information the `/etc/vfstab` file, see “The `/etc/vfstab` Field Descriptions” on page 446.

Creating a Loopback File System (LOFS)

A LOFS file system is a virtual file system that provides an alternate path to an existing file system. When other file systems are mounted onto a LOFS file system, the original file system does not change.

See the `lofs`(7FS) man page for more information.

Caution - Be careful when creating LOFS file systems. Because these are virtual file systems, the potential for confusing both users and applications is enormous.

▼ How to Create a LOFS File System

1. **Become superuser.**

2. **Create the directory where you want to mount the LOFS file system and give it the appropriate permissions and ownership.**

3. **Create a LOFS file system.**

   To set up the system to automatically create a TMPFS file system when it boots, see “Example—Creating a LOFS File System at Boot Time” on page 439.

   ```
   # mount -F lofs loopback-directory  mount-point
   ```

loopback-directory	Specifies the file system to be mounted on the loopback mount point.
mount-point	Specifies the directory on which to mount the LOFS file system.

4. **Look at the output from the `mount` command to verify that the LOFS file system has been created.**

```
# mount -v
```

Example—Creating a LOFS File System

The following example illustrates how to mount and test new software as a loopback file system without actually having to install it.

```
# mkdir /tmp/newroot
# mount -F lofs /new/dist /tmp/newroot/usr/local
# chroot /tmp/newroot command
```

Example—Creating a LOFS File System at Boot Time

You can set up the system to automatically create a LOFS file system when it boots by adding an entry to the end of the `/etc/vfstab` file. The following example shows an entry in the `/etc/vfstab` file that will create a LOFS file system for the root (`/`) file system on `/tmp/newroot`.

```
/ - /tmp/newroot  lofs   -  yes  -
```

Caution - Make sure the loopback entries are the last entries in the `/etc/vfstab` file. Otherwise, if the `/etc/vfstab` entry for a loopback file system precedes the file systems to be included in it, the loopback file system cannot be created.

For more information the `/etc/vfstab` file, see "The `/etc/vfstab` Field Descriptions" on page 446.

CHAPTER 36

Mounting and Unmounting File Systems (Tasks)

This chapter describes how to mount and unmount file systems. 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 to the system so you can use it. You make a file system available by mounting it, which attaches the file system to the system directory tree at the specified mount point. The root (`/`) file system is always mounted. Any other file system can be connected or disconnected from the root (`/`) file system.

The table below provides guidelines on mounting file systems based on how you use them.

TABLE 36–1 Determining How to Mount File Systems

If You Need to Mount ...	Then You Should Use ...
Local or remote file systems infrequently	The `mount` command entered manually from the command line.
Local file systems frequently	The `/etc/vfstab` file, which will mount the file system automatically when the system is booted in multi-user state.
Remote file systems frequently, such as home directories	■ The `/etc/vfstab` file, which will automatically mount the file system when the system is booted in multi-user state. ■ AutoFS, which will automatically mount or unmount the file system when you change into (mount) or out of (unmount) the directory. To enhance performance, you can also cache the remote file systems by using CacheFS.

You can mount a CD-ROM containing a file system by simply inserting it into the drive (Volume Management will automatically mount it). You can mount a diskette containing a file system by inserting it into the drive and running the `volcheck`(1) command. See Chapter 14 for more information.

Commands Used to Mount and Unmount File Systems

The table below lists the commands in the `/usr/sbin` directory that you use to mount and unmount file systems.

TABLE 36–2 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 known inconsistencies. If you receive an error message from the `mount` or `mountall` command, you might need to check the file system. See Chapter 39 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 accessing a file or directory in the file system, if a program has a file open in that file system, or if the file system is shared.

Commonly Used Mount Options

The table below describes the commonly used 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`.

For a complete list of mount options for each file system type, refer to the specific mount command man pages (for example, `mount_ufs`(1M)).

TABLE 36–3 Commonly Used –o Mount Options

Option	File System	Description
`bg | fg`	NFS	If the first attempt fails, retries in the background (`bg`) or in the foreground (`fg`). This option is safe for non-critical `vfstab` entries. The default is `fg`.
`hard | soft`	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 delivered to a process that is hung while waiting for a response on a hard-mounted file system. The default is `intr` (interrupts allowed).
`largefiles | nolargefiles`	UFS	Enables you to create files larger than 2 Gbytes. The `largefiles` option means that a file system mounted with this option *might* contain files larger than 2 Gbytes, but it is not a requirement. The default is `largefiles`. If the `nolargefiles` option is specified, the file system could not be mounted on a system running Solaris 2.6 or compatible versions.
`logging | nologging`	UFS	Enables logging for the file system. UFS logging is the process of storing transactions (changes that make up a complete UFS operation) into a log before the transactions are applied to the UFS file system. Logging helps prevent UFS file systems from becoming inconsistent, which means `fsck` can be bypassed. Bypassing `fsck` reduces the time to reboot a system if it crashes, or after a system is shutdown uncleanly. The log is allocated from free blocks on the file system, and is sized approximately 1 Mbyte per 1 Gbyte of file system, up to a maximum of 64 Mbytes. The default is `nologging`.
`noatime`	UFS	Suppresses access time updates on files, except when they coincide with updates to the `ctime` or `mtime`. See `stat`(2). This option reduces disk activity on file systems where access times are unimportant (for example, a Usenet news spool). The default is normal access time (`atime`) recording.

TABLE 36–3 Commonly Used –o Mount Options *(continued)*

Option	File System	Description
`remount`	All	Changes the mount options associated with an already-mounted file system. This option can generally be used with any option except `ro`, but what can be changed with this option is dependent on the file system type.
`retry=`*n*	NFS	Retries the mount operation when it fails. *n* is the number of times to retry.
`ro | rw`	CacheFS, NFS, PCFS, UFS, S5FS	Specifies read / write or read-only. If you do not specify this option, the default is read / write. The default option for HSFS is `ro`.
`suid | nosuid`	CacheFS, HSFS, NFS, S5FS, UFS	Allows or disallows `setuid` execution. The default is to allow `setuid` execution.

▼ How to Determine Which File Systems Are Mounted

You can determine which file systems are mounted by using the `mount` command.

```
$ mount [ -v ]
```

–v Displays the list of mounted file systems in verbose mode.

Example—Determining Which File Systems Are Mounted

```
$ mount
/ on /dev/dsk/c0t0d0s0 read/write/setuid/intr/largefiles/onerror=panic on ...
/usr on /dev/dsk/c0t0d0s6 read/write/setuid/intr/largefiles/onerror=panic on ...
/proc on /proc read/write/setuid on Fri Sep 10 16:09:48 1999
/dev/fd on fd read/write/setuid on Fri Sep 10 16:09:51 1999
/etc/mnttab on mnttab read/write/setuid on Fri Sep 10 16:10:06 1999
/var/run on swap read/write/setuid on Fri Sep 10 16:10:06 1999
/tmp on swap read/write/setuid on Fri Sep 10 16:10:09 1999
/export/home on /dev/dsk/c0t0d0s7 read/write/setuid/intr/largefiles/onerror=panic ...
$
```

Mounting File Systems (/etc/vfstab File)

The /etc/vfstab Field Descriptions

An entry in the /etc/vfstab file has seven fields, which are described in the table below.

TABLE 36–4 /etc/vfstab Field Descriptions

Field Name	Description
device to mount	■ The block device name for a local UFS file system (for example, /dev/dsk/c0t0d0s0). ■ The resource name for a remote file system (for example, myserver:/export/home). For more information about NFS, see *System Administration Guide, Volume 3*. ■ The block device name of the slice on which to swap (for example, /dev/dsk/c0t3d0s1). ■ The /proc directory for the proc file system type.
device to fsck	The raw (character) device name that corresponds to the UFS 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	Identifies where to mount the file system (for example, /usr).
FS type	The type of file system identified by the device to mount field.

TABLE 36–4 `/etc/vfstab` Field Descriptions *(continued)*

Field Name	Description
`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 zero, UFS file systems are not checked but non-UFS file systems are checked. When the field contains a value greater than zero, the file system is always checked. All file systems with a value of `1` in this field, are checked one at a time in the order they appear in the `vfstab` file. 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. Otherwise, the value of the pass number does not have any effect. The `fsck pass` field does not explicitly specify the order in which file systems are checked, other than as described above.
`mount at boot`	Set to `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. The root (`/`), `/usr` and `/var` file systems are not mounted from the `vfstab` file initially. This field should always be set to `no` for these file systems and for virtual file systems such as `/proc` and `/dev/fd`.
`mount options`	A list of comma-separated options (with no spaces) that are used in mounting the file system. Use a dash (`-`) to indicate no options. See Table 36–3 for a list of commonly used mount 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 (`-`), otherwise the system might not boot successfully. Similarly, white space should not be used in a field value.

▼ How to Add an Entry to the `/etc/vfstab` File

1. **Become superuser.**

   Also, 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.

2. **Edit the `/etc/vfstab` file and add an entry.**

Note - Since the root (/) file system is mounted read-only by the kernel during the boot process, only the `remount` option (and options that can be used in conjunction with `remount`) affect the root (/) entry in the `/etc/vfstab` file.

See Table 36–4 for detailed information about the `/etc/vfstab` field entries. Make sure that 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 to the `/etc/vfstab` File

The following 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.

```
#device           device             mount    FS       fsck   mount    mount
#to mount         to fsck            point    type     pass   at boot  options
#
/dev/dsk/c0t3d0s7 /dev/rdsk/c0t3d0s7 /files1  ufs      2      yes       -
```

The following 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` because it's an NFS file system. 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.

```
#device           device             mount    FS       fsck   mount    mount
#to mount         to fsck            point    type     pass   at boot  options
pluto:/export/man   -                /usr/man nfs      -      yes       ro,soft
```

The following 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. LOFS file systems must always be mounted after the file systems used to make up the LOFS file system.

```
#device           device             mount    FS       fsck   mount    mount
#to mount         to fsck            point    type     pass   at boot  options
#
/                   -                /tmp/
newroot lofs -      yes       -
```

▼ How to Mount a File System (/etc/vfstab File)

1. **Become superuser.**

   Also, 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.

2. **Mount a file system listed in the** `/etc/vfstab` **file.**

```
# mount mount-point
```

mount-point	Specifies 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 (/etc/vfstab File)

The following example mounts the `/usr/dist` file system listed in the `/etc/vfstab` file.

```
# mount /usr/dist
```

▼ How to Mount All File Systems (/etc/vfstab File)

1. **Become superuser.**

   Also, 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.

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.

`-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 specified type 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 All File Systems (`/etc/vfstab` File)

The following example shows the messages displayed if file systems are already mounted when you use the `mountall` command.

```
# mountall
/dev/rdsk/c0t0d0s7 already mounted
mount: /tmp already mounted
mount: /dev/dsk/c0t0d0s7 is already mounted, /export/home is busy,
        or the allowable number of mount points has been exceeded
```

The following example mounts all the local systems listed in the `/etc/vfstab` file.

```
# mountall -l
# mount
/ on /dev/dsk/c0t0d0s0 read/write/setuid/intr/largefiles/onerror=panic on ...
/usr on /dev/dsk/c0t0d0s6 read/write/setuid/intr/largefiles/onerror=panic on ...
/proc on /proc read/write/setuid on Fri Sep 10 16:09:48 1999
/dev/fd on fd read/write/setuid on Fri Sep 10 16:09:51 1999
/etc/mnttab on mnttab read/write/setuid on Fri Sep 10 16:10:06 1999
/var/run on swap read/write/setuid on Fri Sep 10 16:10:06 1999
/tmp on swap read/write/setuid on Fri Sep 10 16:10:09 1999
/export/home on /dev/dsk/c0t0d0s7 read/write/setuid/intr/largefiles/onerror=panic on ...
```

The following example mounts all the remote file systems listed in the `/etc/vfstab` file.

```
# mountall -r
# mount
/ on /dev/dsk/c0t0d0s0 read/write/setuid/intr/largefiles/
onerror=panic on ...
/usr on /dev/dsk/c0t0d0s6 read/write/setuid/intr/largefiles/
onerror=panic on ...
/proc on /proc read/write/setuid on Fri Sep 10 16:09:48 1999
/dev/fd on fd read/write/setuid on Fri Sep 10 16:09:51 1999
/etc/mnttab on mnttab read/write/setuid on Fri Sep 10 16:10:06 1999
/var/run on swap read/write/setuid on Fri Sep 10 16:10:06 1999
/tmp on swap read/write/setuid on Fri Sep 10 16:10:09 1999
/export/home on /dev/dsk/c0t0d0s7 read/write/setuid/intr/largefiles/
onerror=panic on ...
/usr/dist on mars:/usr/dist remote/read/write/
setuid on Tue Sep 14 15:32:18 1999
```

Mounting File Systems (`mount` Command)

▼ How to Mount a UFS File System

1. **Become superuser.**

   Also, 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.

2. **Mount the UFS file system by using the `mount` command.**

```
# mount [-o mount-options] /dev/dsk/device-name mount-point
```

–o *mount-options*	Specifies mount options that you can use to mount a UFS file system. See Table 36–3 or `mount_ufs`(1M) for a list of options.
`/dev/dsk/`*device-name*	Specifies the disk device name for the slice holding the file system (for example, `/dev/dsk/c0t3d0s7`). See "How to Display Disk Slice Information" on page 344 to get slice information for a disk.
mount-point	Specifies the directory on which to mount the file system.

Example—Mounting a UFS File System

The following example mounts `/dev/dsk/c0t3d0s7` on the `/files1` directory.

```
# mount /dev/dsk/c0t3d0s7 /files1
```

Example—Mounting a UFS File System With Logging Enabled

UFS logging eliminates file system inconsistency, which can significantly reduce the time of system reboots. The following example mounts `/dev/dsk/c0t3d0s7` on the `/files1` directory with logging enabled.

```
# mount -o logging /dev/dsk/c0t3d0s7 /files1
```

▼ How to Remount a UFS File System Without Large Files

When you mount a file system, the `largefiles` option is selected by default, which enables you to create files larger than 2 Gbytes. Once a file system contains large files, you cannot remount the file system with the `nolargefiles` option or mount it on a system running Solaris 2.6 or compatible versions, until you remove any large files and run `fsck` to reset the state to `nolargefiles`.

This procedure assumes that the file system is in the `/etc/vfstab` file.

1. **Become superuser.**

2. **Make sure there are no large files in the file system.**

```
# cd mount-point
# find . -xdev -size +20000000 -exec ls -l {} \;
```

mount-point	Specifies the mount point of the file system you want to check for large files.

If large files exist within this file system, they must be removed or moved to another file system.

3. Unmount the file system.

```
# umount mount-point
```

4. Reset the file system state.

```
# fsck mount-point
```

5. Remount the file system with the `nolargefiles` **option.**

```
# mount -o nolargefiles mount-point
```

Example—Mounting a File System Without Large Files

The following example checks the `/datab` file system and remounts it with the `nolargefiles` option.

```
# cd /datab
# find . -xdev -size +20000000 -exec ls -l {} \;
# umount /datab
# fsck /datab
# mount -o nolargefiles /datab
```

▼ How to Mount an NFS File System

1. **Become superuser.**

   Also, 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.

2. **Make sure the resource (file or directory) is available from a server.**

   To mount an NFS file system, the resource must be made available on the server by using the `share` command. See *System Administration Guide, Volume 3* for information on how to share resources.

3. **Mount the NFS file system by using the `mount` command.**

```
# mount -F nfs [-o mount-options] server:/directory mount-point
```

–o *mount-options*	Specifies `mount` options that you can use to mount an NFS file system. See Table 36–3 for the list of commonly used `mount` options or `mount_nfs`(1M) for a complete list of options.
server:/directory	Specifies the server's host name that contains the shared resource, and the path to the file or directory to mount.
mount-point	Specifies the directory on which to mount the file system.

Example—Mounting an NFS File System

The following example mounts the `/export/packages` directory on `/mnt` from the server `pluto`.

```
# mount -F nfs pluto:/export/packages /mnt
```

▼ IA: How to Mount a System V (S5FS) File System

1. **Become superuser.**

   Also, 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.

2. **Mount the S5FS file system by using the `mount` command.**

```
# mount -F s5fs [-o mount-options] /dev/dsk/device_name mount-point
```

–o *mount-options*	Specifies mount options that you can use to mount a S5FS file system. See Table 36–3 for the list of commonly used mount options or `mount_s5fs`(1M) for a complete list of options.
`/dev/dsk/`*device-name*	Specifies the disk device name of the slice holding the file system (for example, `/dev/dsk/c0t3d0s7`). See "How to Display Disk Slice Information" on page 344 to get slice information for a disk.
mount-point	Specifies the directory on which to mount the file system.

IA: Example—Mounting an S5FS File System

The following example mounts `/dev/dsk/c0t3d0s7` on the `/files1` directory.

```
# mount -F s5fs /dev/dsk/c0t3d0s7 /files1
```

▼ IA: 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. **Become superuser.**

   Also, 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.

2. **Mount the PCFS file system by using the `mount` command.**

```
# mount -F pcfs [-o rw | ro] /dev/dsk/device-name:logical-drive  mount-point
```

`–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*	Specifies 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	Specifies the directory on which to mount the file system.

Note that the *device-name* and *logical-drive* must be separated by a colon.

IA: Examples—Mounting a PCFS (DOS) File System From a Hard Disk

The following example mounts the logical drive in the Primary DOS slice on the `/pcfs/c` directory.

```
# mount -F pcfs /dev/dsk/c0t0d0p0:c /pcfs/c
```

The following example mounts the first logical drive in the Extended DOS slice read-only on `/mnt`.

```
# mount -F pcfs -o ro /dev/dsk/c0t0d0p0:2 /mnt
```

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 need to check and repair it using the `fsck` command. See Chapter 39 for more information about the `fsck` command.

It is a good idea to unmount a file system before doing a complete backup. See Chapter 43 for more information about doing backups.

Note - File systems are automatically unmounted as part of the system shutdown procedure.

Prerequisites

The prerequisites to unmounting file systems are:

- You must be superuser.
- A file system must be available for unmounting. You cannot unmount a file system that is busy. A file system is considered busy if a user is accessing a directory in the file system, if a program has a file open in that file system, or if it is being shared. 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 Accessing a File System" on page 457 for more details.

    Notify users if you need to unmount a file system they are using.
  - Unsharing the file system

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 Determine Which File Systems Are Mounted" on page 445.

▼ How to Stop All Processes Accessing a File System

1. **Become superuser.**

2. **List all the processes that are accessing the file system, so you know which processes you are going to stop.**

```
# fuser -c [ -u ] mount-point
```

–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	The name of the file system for which you want to stop processes.

3. Stop all processes accessing 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.

4. Verify that there are no processes accessing the file system.

```
# fuser -c mount-point
```

Example—Stopping All Processes Accessing a File System

The following example stops process `4006c` that is using the `/export/home` file system.

```
# 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 `/`, `/usr`, or `/var`):

Note - The root (/), /usr, and /var file systems are special cases. The root (/) file system can be unmounted only during a shutdown, since the system needs the root (/) file system to function.

1. **Make sure you have met the prerequisites listed on "Prerequisites" on page 457.**

2. **Unmount the file system.**

```
# umount mount-point
```

mount-point	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

The following example unmounts a local home file system.

```
# umount /export/home
```

The following example unmounts the file system on slice 7.

```
# umount /dev/dsk/c0t0d0s7
```

▼ How to Unmount All File Systems (/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 "Prerequisites" on page 457.**

2. **Unmount all the file systems listed in the /etc/vfstab file.**

```
# umountall
```

All systems that are unmounted, except those that are busy.

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 Accessing a File System" on page 457.**

4. **Repeat Step 2 as needed until all file systems are unmounted.**

CHAPTER 37

The Cache File System (Tasks)

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).

The following 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. The figure below 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, which can be either NFS or HSFS (High Sierra File System). When the user attempts to access files that are part of the back file system, those files are placed in the cache. To the user, the initial request to access a file might seem slow, but subsequent uses of the same file will be faster.

Note - You can mount only file systems that are shared. See `share`(1M) for information on sharing file systems.

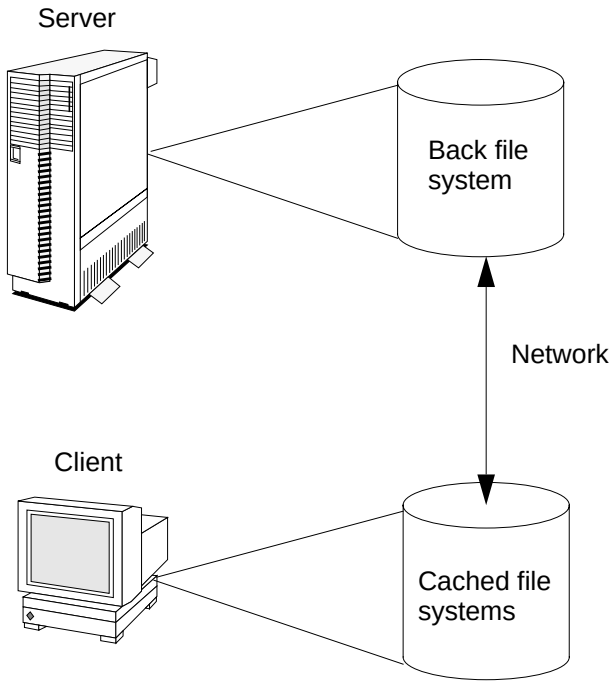


Figure 37–1 How CacheFS Works

Setting Up a Cached File System Task Map

TABLE 37–1 Setting Up a Cached File System Task Map

Task	Description	For Instructions, Go To
1. Create a Cache	Use the `cfsadmin` command to create a cache.	"How to Create a Cache" on page 464
2. 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`" on page 465

TABLE 37–1 Setting Up a Cached File System Task Map *(continued)*

Task	Description	For Instructions, Go To
	Cache a file system by editing the `/etc/vfstab` file.	"How to Mount a File System in a Cache by Editing the `/etc/vfstab` File" on page 468
	Cache a file system by using AutoFS.	"How to Mount a File System in a Cache With AutoFS" on page 469

Creating a Cache

The following procedure describes how to create a cache directory.

▼ How to Create a Cache

1. **Become superuser.**
2. **Create a cache using the `cfsadmin -c` command.**

```
# cfsadmin -c cache-directory
```

cache-directory	Indicates the name of the directory where the cache resides. For more information, see `cfsadmin`(1M).

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.

```
# mkdir /local
# cfsadmin -c /local/mycache
```

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.

The table below describes three ways to mount cached file systems:

To Mount A Cached File System By ...	You Need To Do This ...
Using the `mount`(1M) command	Every time the system reboots in order to access the same file system.
Editing the `/etc/vfstab` file	Only once. The `/etc/vfstab` file remains unchanged after the system reboots.
Using AutoFS	Only once. AutoFS maps remain unchanged after the system reboots.

Choose the method of mounting file systems that best suits your environment.

Note - Caching of the root (`/`) and `/usr` file systems is not supported in CacheFS. To cache the root (`/`) and `/usr` file systems, you must purchase the Solstice AutoClient product. For more information about the AutoClient product, see the *Solstice AutoClient 2.1 Administration Guide.*

▼ How to Specify a File System to Be Mounted in a Cache With `mount`

1. **Become superuser.**

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` 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` command.

```
# mount -F cachefs -o backfstype=fstype,cachedir=cache-directory[, options]
back-filesystem  mount-point
```

fstype	Indicates the file system type of the back file system (can be either `NFS` or `HSFS`).
cache-directory	Indicates 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 464.
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	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	Indicates the directory where the file system is mounted.

4. Verify that the cache you created was actually mounted by using 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
```

(continued)

```
      garbage collection:      0
```

The mount point is the cached file system you created. For more information about the `cachefsstat` command, see “CacheFS Statistics” on page 489.

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
```

Examples—Specifying a File System to be Mounted in a Cache With `mount`

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` option 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` command.

```
# 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 496 for more information.

```
# mount -F cachefs -o backfstype=nfs,cachedir=/local/mycache,demandconst merlin:/docs /docs
```

▼ How to Mount a File System in a Cache by Editing the `/etc/vfstab` File

1. **Become superuser.**

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/devicename /dev/rdsk/devicename
/mount-point cachefs  2    yes  -
```

   This line represents the new entry.

3. **Mount the cached file system using the `mount` command, as follows:**

```
# mount /mount-point
```

   or reboot.

Example—Mounting a File System in a Cache by Editing the `/etc/vfstab` File

The following example shows the `/etc/vfstab` entry for the cache file system.

```
#device           device             mount      FS    fsck  mount   mount
#to mount         to fsck            point      type  pass  at boot options
#
/dev/dsk/c0t1d0s0 /dev/rdsk/c0t1d0s0 /usr/local cachefs  2   yes     -
```

The `/usr/local` directory is mounted in the cache directory.

```
# mount /usr/local
```

▼ How to Mount a File System in a Cache With AutoFS

You can mount a file system in a cache with AutoFS 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) for details on automount maps. Also see *System Administration Guide, Volume 3*.

1. **Become superuser.**

2. **Using an editor, add the following line to the** `auto_direct` **map:**

```
/mount-point -fstype=cachefs,cachedir=/directory,backfstype=nfs
server:/file-system
```

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.**

5. **Verify that the entry was made correctly by changing to the file system you mounted in the cache, and then list the contents, as follows:**

```
# cd filesystem
# ls filesystem
```

For more information about AutoFS and how to edit the maps, refer to the AutoFS chapter of the *System Administration Guide, Volume 3*.

Example—Mounting a File System in a Cache With AutoFS

The following `auto_master` entry automatically mounts the cache file system in the `/docs` directory.

```
/docs  -fstype=cachefs,cachedir=/local/mycache,backfstype=nfs
merlin:/docs
```

Maintaining a Cached File System Task Map

TABLE 37–2 Maintaining a Cached File System Task Map

Task	Description	For Instructions, Go To
1. Modify the Cache	Modify the cache behavior.	"How to Modify File Systems in a Cache" on page 471
2. Display Cache Information	Display information about cached file systems by using the `cfsadmin` command.	"How to Display Information About Cached File Systems" on page 472
3. Perform Consistency Checking	Perform consistency checking on demand by using the `cfsadmin` command.	"How to Specify Consistency Checking on Demand" on page 473

TABLE 37–2 Maintaining a Cached File System Task Map *(continued)*

Task	Description	For Instructions, Go To
4. Delete a Cache	Delete cached file systems by using the `umount` command and the `cfsadmin` command.	"How to Delete a Cached File System" on page 473
5. 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" on page 475

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 Chapter 36. When you modify a file system in the cache, you need to delete the cache and then recreate it. You might also need to reboot your machine in single user mode, depending on how your file systems are shared and accessed.

The following example shows some of the steps involved in this procedure.

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 might have other commands you prefer to use to accomplish some of the tasks shown in this example.

```
# shutdown -g30 -y
.
.
.
Type Cntrl-d to proceed with normal startup,
(or give root password for system maintenance):
# enter password:
.
.
.
Here is where you might 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
# init 6
.
.
.
console login:
password:
# mount -F cachefs -o backfstype=nfs,cachedir=/local/cache1, demandconst merlin:/docs /docs
#
```

If you did not successfully remount the file system in the cache, the system displays an error message similar to the following:

```
cachefsstat: /doc: not a cachefs mount point
```

▼ How to Display Information About Cached File Systems

1. **Become superuser.**

2. **Display information about all file systems cached under a specified cache.**

   ```
   # cfsadmin -l cache-directory
   ```

   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 name of 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 superuser.**

2. **Mount the file system in the cache specifying the** `demandconst` **option of the** `mount` **command, as follows:**

```
# mount -F cachefs -o backfstype=nfs,cachedir=/directory,demandconst
server:/file-system  /mount-point
```

3. **To initiate consistency checking on a specific cached file system, use the** `cfsadmin -s` **command as follows:**

```
# cfsadmin -s /mount-point
```

   For more information about consistency checking, see “Consistency Checking of Cached File Systems With the Back File System” on page 496.

▼ How to Delete a Cached File System

1. **Become superuser.**

2. **Unmount the cached file system.**

```
# umount mount-point
```

mount-point specifies the cached file system that you want to delete.

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
cache-ID
#
```

4. **Delete a cached file system from a specified cache.**

```
# cfsadmin -d cache-id cache-directory
```

cache-id	Indicates the name of the cached file system, which is the last line of the `cfsadmin -l` output. See "How to Display Information About Cached File Systems" on page 472 for more information. You can delete all the cached file systems in a particular cache by specifying `all` for *cache-id*.
cache-directory	Specifies the directory where the cache resides.

5. **Verify that the file system has been deleted.**

The cache ID of the file system you just deleted should be missing from the output of the following command. Refer to `cfsadmin`(1M) for more information about the fields specified in the command output.

```
# cfsadmin -l cache-directory
cfsadmin: list cache FS information
   maxblocks     90%
   minblocks      0%
   threshblocks  85%
   maxfiles      90%
   minfiles       0%
   threshfiles   85%
   maxfilesize    3MB
#
```

Examples—Deleting a Cached File System

The following example unmounts a cached file system and deletes the cached file system from the cache.

```
# umount /docs
# 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
```

▼ How to Check the Integrity of Cached File Systems

Use the `fsck` command to check the integrity of cached file systems. The CacheFS version of `fsck` automatically corrects problems without requiring user interaction. You should not need to run `fsck` manually for cached file systems; `fsck` 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 `fsck_cachefs`(1M) for more information.

1. **Become superuser.**

2. **Check the cached file systems under a specified cache.**

```
# fsck -F cachefs [-m -o noclean] cache-directory
```

`–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	Indicates 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
#
```

Managing Your Cache File Systems With `cachefspack`

For general use, CacheFS operates automatically, without requiring any action from the user. Files are cached on a most recently used basis. With the *packing* feature, you can take a more active role in managing your cache by ensuring that certain files or directories are always updated in the cache.

Packing enables you to specify files and directories to be loaded in the cache. It ensures that current copies of these files are available in the cache.

The *packing list* contains the names of specific files and directories. It can also contain other packing lists. This saves you having to specify individual files and directories in case you have many items to pack in your cache.

The `cachefspack` command provides you with added control of your CacheFS file systems, employing the packing functionality.

▼ How to Pack Files in the Cache

Pack files in the cache using the `cachefspack` command.

```
$ cachefspack -p filename
```

–p	Specifies that you want the file or files packed. This is also the default.
filename	Specifies the name of the file or directory you want packed in the cache. When you specify a directory, all of its subdirectories are also packed. For more information, see `cachefspack`(1M).

Examples—Packing Files in the Cache

The following example shows the file `projects` specified to be packed in the cache.

```
$ cachefspack -p projects
```

The following example shows several files specified to be packed in the cache.

```
$ cachefspack -p projects updates master_plan
```

The following example shows a directory specified to be packed in the cache.

```
$ cachefspack -p /usr/openwin/bin
```

Packing Lists

One of the features of the `cachefspack` command is the ability to pack packing lists. This saves the time of having to specify each individual file that you want packed in the cache.

A packing list contains files or directories to be packed in the cache. If a directory is in the packing list, all of its subdirectories and files will also be packed.

▼ How to Create a Packing List

To create a packing list, open a file by using `vi` or the editor of your choice. The packing list file format uses the same format as the `filesync` command. See `filesync`(1) for more information.

Example—Creating a Packing List

The following example shows the contents of a packing list file.

```
BASE /home/ignatz
LIST plans
```

(Continuation)

```
LIST docs
IGNORE *.ps
```

- The path identified with the BASE statement is the directory where you have items you wish to pack.
- The two LIST statements identify specific files within that directory to pack.
- The IGNORE statement identifies the file type of .ps, which you do not wish to pack.

▼ How to Pack Files in the Cache as Specified in a Packing List

To pack files using the packing list, use the cachefspack -f command, as follows:

```
$ cachefspack -f packing-list
```

This means you want the software to read the packing list and pack files based on the information specified in the packing list.

–f	Specifies that you want to use a packing list.
packing-list	Specifies the name of the packing list.

Example—Packing Files in the Cache as Specified in a Packing List

This examples uses the list.pkg file as the packing list for the cachefspack command.

```
$ cachefspack -f list.pkg
```

▼ How to Specify Files in the Packing List to be Treated as Regular Expressions

To specify that one or more files in the packing list should be treated as regular expressions (not as literal file names), use the –r option with the –f option of the cachefspack command. The –r option cannot be used alone.

```
$ cachefspack -rf packing_list
```

where *packing_list* contains a `LIST` command defined as follows:

```
LIST *.doc
```

`–r`	Specifies that you want the file or files defined in the `LIST` command treated as regular expressions, and not as literal file names.
`–f`	Specifies that you want the packing list packed in the cache.
packing_list	Indicates the name of the packing list that contains the `LIST` command with the file or files you want treated as regular expressions.

Example—Specifying Files in the Packing List to be Treated as Regular Expressions

The following example shows the packing list `list.pkg` specified to be packed in the cache. `list.pkg` contains a `LIST` command that defines a regular expression.

```
$ cachefspack -rf list.pkg
```

The software will pack the file `list.pkg` into the cache and treat the file names defined in the `LIST` command as regular expressions, and not as literal file names.

▼ How to Pack Files From a Shared Directory

1. **To pack files from a shared directory, and to ensure that you pack only those files that you own, define the `LIST` command within the packing list file as follows:**

   ```
   LIST !find . -user your_user_name -print
   ```

2. **Pack the packing list in the cache using the `cachefspack -sf` command.**

   ```
   $ cachefspack -sf packing_list
   ```

`–s`	Adjusts the output of the find command to be suitable for the packing list.
`–f`	Specifies a packing list to read.
filename	Specifies the name of the packing list to read.

Note - The `–s` option must be used with the `–f` option. The `–s` option cannot be used alone.

Example—Packing Files From a Shared Directory

The following example shows how to define a `LIST` command in the packing list to pack only the files from the base directory that you own:

```
LIST !find . -user jones -print
```

The following example shows how you would then specify packing the packing list.

```
$ cachefspack -sf /projects/proj_1
```

Unpacking Files

You might need to remove, or unpack, a file from the cache. Perhaps you have some files or directories that are a higher priority than others, so you need to unpack the less critical files. For example, you finished up a project and have archived the files associated with that project. You are now working on a new project, and therefore, a new set of files.

▼ How to Unpack Files or Packing Lists From the Cache

Unpack files or packing lists from the cache using the `–u` or `–U` option of the `cachefspack` command.

```
$ cachefspack -u filename | -U cache-directory
```

–u	Specifies that you want the file or files unpacked. You must specify a filename with this option.
filename	Specifies the name of the file or packing list you want unpacked in the cache. For more information about the `cachefspack` command, see the man page.
–U	Specifies that you want to unpack all files in the cache.

Examples—Unpacking Files or Packing Lists From the Cache

The following example shows the file `/usr/openwin/bin/xlogo` specified to be unpacked from the cache.

```
$ cachefspack -u /usr/openwin/bin/xlogo
```

The following example shows several files specified to be unpacked from the cache.

```
$ cd /usr/openwin/bin
$ cachefspack -u xlogo xview xcolor
```

You can also unpack a packing list, which is a file that contains the path to a directory of files, as follows:

```
$ cachefspack -uf list.pkg
```

The following example uses the `-U` option to specify all files in a cache directory to be unpacked.

```
$ cachefspack -U /local/mycache
```

You cannot unpack a cache that does not have at least one file system mounted. With the `-U` option, if you specify a cache that does not contain mounted file systems, you will see output similar to the following:

```
$ cachefspack -U /local/mycache
cachefspack: Could not unpack cache /local/mycache, no mounted
filesystems in the cache.
```

Displaying Packed Files Information

You might want to view information about the files that you've specified to be packed, and what their packing status is.

▼ How to Display Packed Files Information

To display packed files information, use `cachefspack -i` command.

```
$ cachefspack -i[v] cached-filename-or-directory
```

`–i`	Specifies you want to view information about your packed files.
`–v`	The verbose option.
cached-filename-or-directory	Specifies the name of the file or directory for which to display information.

Example—Displaying Packed Files Information

The following example shows that a file called `doc_file` is successfully packed.

```
$ cachefspack -i doc_file
cachefspack: file doc_file marked packed YES, packed YES
```

The following example shows a directory called `/usr/openwin`, which contains a subdirectory `bin`. The subdirectory `bin` has three files: `xterm`, `textedit`, and `resize`. Although the files `xterm` and `resize` are specified to be packed, they are not. The file `textedit` is successfully packed.

```
$ cd /usr/openwin
$ cachefspack -i bin
.
.
.
cachefspack: file /bin/xterm marked packed YES, packed NO
cachefspack: file /bin/textedit marked packed YES,
packed YES
cachefspack: file /bin/resize marked packed YES,
packed NO
.
```

(continued)

```
.
.
```

If you use the –iv options in combination, you will get additional information as to whether or not the file or directory specified has been flushed from the cache. For example:

```
$ cd /usr/openwin
$ cachefspack -iv bin
.
.
.
cachefspack: file /bin/xterm marked packed YES, packed NO,
nocache YES
cachefspack: file /bin/textedit marked packed YES,
packed YES, nocache NO
cachefspack: file /bin/resize marked packed YES,
packed NO
nocache NO
.
.
.
```

The last line of the example above shows that the directory contents have not been flushed from the cache.

Viewing Help on the cachefspack Command

You can print out a brief help summary of all the cachefspack options and what they mean by using the -h option as follows:

```
$ cachefspack -h
Must select 1 and only 1 of the following 5 options
-d Display selected filenames
-i Display selected filenames packing status
-p Pack selected filenames
-u Unpack selected filenames
-U Unpack all files in directory 'dir'

-f Specify input file containing rules
-h Print usage information
```

```
-r Interpret strings in LIST rules as regular expressions
-s Strip ’./’ from the beginning of a pattern name
-v Verbose option
files - a list of filenames to be packed/unpacked
```

cachefspack Errors

You might see the following error messages when you use the `cachefspack` command.

```
cachefspack: pathname - can’t open directory: permission denied
```

Cause

You might not have the correct permissions to access the file or directory.

Action

Set the correct permissions.

```
cachefspack: pathname - can’t open directory: no such file
or
directory
```

Cause

You might not have the correct file or directory.

Action

Check for a possible typo.

```
cachefspack: pathname - can’t open directory: stale NFS file handle
```

Cause

The file or directory might have been moved or deleted from the server at the time you attempted to access it.

Action

Verify that the file or directory on the server is still accessible.

```
cachefspack: pathname - can't open directory: interrupted
system
call
```

Cause

You might have pressed Control-c inadvertently while issuing the command.

Action

Reissue the command.

```
cachefspack: pathname - can't open directory: I/O error
```

Cause

A hardware problem.

Action

Check your hardware connections.

```
cachefspack: error opening dir
```

Cause

You might not have the correct file or directory. The path identified after the BASE command in the file format could be a file and not a directory. The path specified must be a directory.

Action

Check for a possible typo. Check the path identified after the BASE command in your file format. Make sure it is a directory, and not a file.

```
cachefspack: unable to get shared objects
```

Cause

The executable might be corrupt or it's a format that is not recognizable.

Action

No corrective action can be taken.

```
cachefspack: filename - can't pack file: permission denied
```

Cause

You might not have the correct permissions to access the file or directory.

Action

Set the correct permissions.

```
cachefspack: filename - can't pack file: no such file or directory
```

Cause

You might not have the correct file or directory.

Action

Check for a possible typo.

```
cachefspack: filename- can't pack file: stale NFS file handle
```

Cause

The file or directory might have been moved or deleted from the server at the time you attempted to access it.

Action

Verify that the file or directory on the server is still accessible.

```
cachefspack: filename- can't pack file: interrupted system call
```

Cause

You might have pressed Control-c inadvertently while issuing the command.

Action

Reissue the command.

```
cachefspack: filename- can't pack file: I/O error
```

Cause

A hardware problem.

Action

Check your hardware connections.

```
cachefspack: filename- can't pack file: no space left on device.
```

Cause

You are out of disk space. The cache is at maximum capacity.

Action

You need to increase disk space. Increase the size of the cache.

```
cachefspack: filename - can't unpack file: permission denied
```

Cause

You might not have the correct permissions to access the file or directory.

Action

Set the correct permissions.

```
cachefspack: filename - can't unpack file: no such file or directory
```

Cause

You might not have the correct file or directory.

Action

Check for a possible typo.

```
cachefspack: filename- can't unpack file: stale NFS file handle
```

Cause

The file or directory might have been moved or deleted from the server at the time you attempted to access it.

Action

Verify that the file or directory on the server is still accessible.

```
cachefspack: filename - can't unpack file: interrupted system call
```

Cause

You might have pressed Control-c inadvertently while issuing the command.

Action

Reissue the command.

```
cachefspack: filename- can't unpack file I/O error
```

Cause

A hardware problem.

Action

Check your hardware connections.

```
cachefspack: only one 'd', 'i', 'p', or 'u' option allowed
```

Cause

You entered more than one of the above options in a command session.

Action

Select one option for the command session.

```
cachefspack: can't find environment variable.
```

Cause

You forgot to set a corresponding environment variable to match the $ in your configuration file.

Action

Define the environment variable in the proper location.

```
cachefspack: skipping LIST command - no active base
```

Cause

A LIST command is present in your configuration file that has no corresponding BASE command.

Action

Define the BASE command.

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`(1M)	Specifies the location of the log file. This command also displays where the statistics are currently being logged, and enables you to halt logging.
`cachefswssize`(1M)	Interprets the log file to give a recommended cache size.
`cachefsstat`(1M)	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.

Note - The CacheFS statistics commands can be issued from any directory. You must be superuser to issue the `cachefswssize` 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 492.

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 Task Map

The table below shows the steps involved in setting up CacheFS statistics.

TABLE 37–3 Setting Up CacheFS Statistics Task Map

Task	Description	For Instructions, Go To
1. Set Up Logging	Set up logging on a cached file system using the `cachefslog` command.	"How to Set Up the Logging Process" on page 491
2. Locate the Log File	Locate the log file with the `cachefslog` command.	"How to Locate the Log File" on page 491
3. Stop the Logging Process	Stop the logging process with the `cachefslog` command.	"How to Stop the Logging Process" on page 492
4. View the Cache Size	View the cache size using the `cachefswssize` command.	"How to View the Working Set (Cache) Size" on page 492
5. View the Cache Statistics	View the statistics using the `cachefsstat` command.	"How to View Cache Statistics" on page 494

CacheFS Logging

This section describes how to set up and view CacheFS logging.

▼ How to Set Up the Logging Process

1. **Set up the logging process with the** `cachefslog` **command.**

```
$ cachefslog -f log-file-path mount-point
```

`–f`	Sets up the logging process.
log-file-path	Specifies 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.

2. **Verify that you set up the log file correctly by using the** `cachefslog` **command, as follows:**

```
$ cachefslog mount-point
```

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
```

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
```

mount-point	Specifies the cached file system for which you want to view the statistics.

Examples—Locating the Log File

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 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 halts logging on `/home/stuff`.

```
$ cachefslog -h /home/stuff
not logged: /home/stuff
```

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 might 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 superuser.**

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.

```
# 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 might want to view certain information about a specific cached file system. The following table explains the terminology displayed in the statistics output.

TABLE 37–4 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
```

mount-point — Specifies 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)
```

(continued)

```
 consistency checks: 700 (650 pass, 50 fail)
           modifies: 321
garbage collection:  0
```

The Cache Structure and Behavior

Each cache has a set of parameters that determines how it behaves and its structure. The parameters are set to default values which are listed in Table 37–5. The default values specify that the entire front file system is used for caching, which is the recommended method of caching file systems.

TABLE 37–5 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 might 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 it is, 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.

CHAPTER 38

Configuring Additional Swap Space (Tasks)

This is a list of the overview conceptual information and step-by-step instructions in this chapter.

About Swap Space

It is important for administrators to understand the features of the SunOS 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 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 virtual memory system maps physical copies of files on disk to virtual addresses in memory. Physical memory pages which contain 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 there is no identity assigned to the disk space backing the memory.

The Solaris environment uses 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 might 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 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, real directory.

  Using your compiler's `TMPDIR` variable only controls whether the compiler is using `/tmp` or not. It has no 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 0

messages.1:Sep 21 20:52:11 mars genunix: [ID 470503 kern.warning]
WARNING: Sorry, no swap space to grow stack for pid 100295 (myprog)
```

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 message 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 the table below.

TABLE 38–1 Default Swap Space Allocations

If Your System Has *n* Mbytes of Physical Memory ...	**Then the Default Swap Space Allocated Is ...**
16-63	32 Mbytes
64-127	64 Mbytes
128-511	128 Mbytes
greater than 512	256 Mbytes

Additional swap space can also be added to the system by creating a swap file. See "Adding More Swap Space" on page 503 for information about creating a swap file.

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.

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.

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" on page 498 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
```

(continued)

(Continuation)

```
/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 the table below.

TABLE 38–2 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.

TABLE 38–2 Output of the `swap -s` Command *(continued)*

Keyword	Description
`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 easiest 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.

Alternative ways to add more swap space are to repartition an existing disk or add another disk. See Chapter 28 for information on how to repartition a disk.

Creating a Swap File

The following general steps are involved in creating a swap file:

- Creating a swap file using the `mkfile` command.
- Activating the swap file with the `swap` command.
- Adding an entry for the swap file in the `/etc/vfstab` file so that it's 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 filled 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.

The table below shows the options to the `mkfile` command.

TABLE 38–3 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 superuser.**

   You can create a swap file without root permissions, but it is a good idea for root to be the owner of the swap file to avoid accidental overwriting.

2. **Create the swap file.**

```
# mkfile nnn[k|b|m] filename
```

   The swap file of the size *nnn* (in Kbytes, bytes, or Mbytes) and name you specify is created.

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, the system is rebooted, or the swap file is removed. Keep in mind that you can't unmount a file system while some process or program is swapping to the swap file.

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, like this:**

```
/path/filename   -      -      swap    -     no     -
```

5. **Verify that the swap file is added.**

```
$ /usr/sbin/swap -l
```

Example—Creating a Swap File and Making It Available

The following examples shows how to create a 24 Mbyte swap file called `/files/swapfiles`.

```
# mkdir /files
# mkfile 24m /files/swapfile
# swap -a /files/swapfile
# vi /etc/vfstab
(An entry is added for the swap file):
/files/swapfile   -      -       swap     -     no     -
# 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 superuser.**
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 36 for information on mounting a file system.

Example—Removing Extra Swap Space

The following examples shows how to delete the `/files/swapfile` swap file.

```
# swap -d /files/swapfile
# (Remove the deleted swap entry from the /etc/vfstab file)
# rm /files/swapfile
# swap -l
swapfile             dev  swaplo  blocks   free
/dev/dsk/c0t2d0s1   32,17       8  205624  192720
```

CHAPTER 39

Checking File System Integrity

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

See "Troubleshooting File System Problems" in *System Administration Guide, Volume 2* for information about `fsck` error messages.

See Chapter 40 for background information on the UFS file system structures referred to in this chapter.

File System Integrity

The UFS file system relies on an internal set of tables to keep track of inodes 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, and the `fsck` options. 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` error messages are covered in "Troubleshooting File System Problems" in *System Administration Guide, Volume 2*.

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.

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 `/sbin/rcS` 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 the table below.

TABLE 39–1 State Flag Values

State Flag Value	Description
FSACTIVE	When a file system is mounted and then modified, the state flag is set to FSACTIVE. The file system might 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 might 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.
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.
FSLOG	If the file system was mounted with UFS logging, the state flag is set to FSLOG. Any file system with an FSLOG 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 might 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, FSSTABLE, or FSLOG.

The table below shows how the state flag is modified by the fsck command, based on its initial state.

TABLE 39–2 How the State Flag is Modified by `fsck`

Inital State: Before `fsck`	**State After** `fsck`		
	No Errors	**All Errors Corrected**	**Uncorrected Errors**
unknown	FSSTABLE	FSSTABLE	unknown
FSACTIVE	FSSTABLE	FSSTABLE	FSACTIVE
FSSTABLE	FSSTABLE	FSSTABLE	FSACTIVE
FSCLEAN	FSCLEAN	FSSTABLE	FSACTIVE
FSBAD	FSSTABLE	FSSTABLE	FSBAD
FSLOG	FSLOG	FSLOG	FSLOG

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 Might Occur

Every working day hundreds of files might 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 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 might 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 might 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 might result from an *unclean shutdown*, if a system is shut down improperly, or when a mounted file system is taken offline improperly. To prevent unclean shutdowns, 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 that `fsck` applies to these UFS file system components: superblock, cylinder group blocks, inodes, indirect blocks, and data blocks.

Superblock Checks

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 Checks

The file system size must be larger than the number of blocks used by the superblock and 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 Block Checks

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 Inode Checks

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 might be one of eight types:

- Regular

- Directory
- Block special
- Character special
- FIFO (named-pipe)
- Symbolic link
- Shadow (used for ACLs)
- Socket

Inodes might 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 Checks

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` might 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 Block Checks

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 might 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 Number Checks

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 might be caused by an indirect block not being written to the file system. The `fsck` program prompts you to clear the inode.

Inode Size Checks

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 listed above 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 Checks

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 a new directory entry 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 Checks

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 "`..`" Entry Checks

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.

The `fsck` Summary Message

When you run fsck interactively and it completes successfully, the following message is displayed:

```
# fsck /dev/rdsk/c0t0d0s7
** /dev/rdsk/c0t0d0s7
** Last Mounted on /export/home
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
** Phase 5 - Check Cyl groups
2 files, 9 used, 2833540 free (20 frags, 354190 blocks, 0.0% fragmentation)
#
```

The last line of `fsck` output describes the following information about the file system:

# files	Number of inodes in use
# used	Number of fragments in use
# free	Number of unused fragments
# frags	Number of unused non-block fragments
# blocks	Number of unused full blocks
% fragmentation	Percentage of fragmentation, where: free fragments x 100 / total fragments in the file system

Modifying File System Checking at Boot Time

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 (/), /usr, and /var 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 36 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:/usr/dist -        /usr/dist       nfs     no      yes     -
$
```

The table below describes the function of the `fsck pass` field.

TABLE 39–3 The `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`	The file system specific `fsck` command is called.	When the value is `0` 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 (`–o p` option), `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 File System Checking at Boot Time

1. **Become superuser.**
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 might 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 might be displayed in the console window or the system might crash.

Before using `fsck`, you might want to refer to "Syntax and Options for the `fsck` Command" on page 524 and "Troubleshooting File System Problems" in *System Administration Guide, Volume 2* for more information.

▼ How to See If a File System Needs Checking

1. **Become superuser.**
2. **Unmount the file system if it is mounted.**

```
# umount /mount-point
```

3. **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

The following example shows that the file system needs checking.

```
# fsck -m /dev/rdsk/c0t0d0s6
** /dev/rdsk/c0t0d0s6
ufs fsck: sanity check: /dev/rdsk/c0t0d0s6 needs checking
```

▼ How to Check File Systems Interactively

1. **Become superuser.**

2. **Unmount the local file systems 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 "Troubleshooting File System Problems" in *System Administration Guide, Volume 2* for information about how to respond to the error message prompts to interactively check one or more UFS file systems.

Caution - Running `fsck` on a mounted file system might cause a system to crash if `fsck` makes any changes, unless stated otherwise, such as running fsck in single-user mode to repair a file system.

4. **If you corrected any errors, type `fsck` and press Return.**

`fsck` might 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 524.

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 might 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

The following example checks `/dev/rdsk/c0t0d0s6` and corrects the incorrect block count.

```
# 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 shutdown. 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 shutdown. 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 superuser.**

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 following example preens the `/usr` file system.

```
# 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 superuser.**

2. **Change to a directory outside the damaged file system.**

3. **Unmount the file system.**

```
# umount mount-point
```

Caution - Be sure to use the `newfs -N` 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, unless the file system was created with special parameters. See "Deciding on Custom File System Parameters" on page 542 for information on creating a customized 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

The following example restores the superblock copy `5264` 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 might 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 might help you fix the problem. For example, the messages might point to a bad directory. If you delete the directory, you might 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 might, ultimately, need to re-create the file system and restore its contents from backup media. See Chapter 44 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 might 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` command tries to work around bad blocks on the disk. If the disk is too severely damaged, however, the problems might persist, even after reformatting. See `format`(1M) for information about using the `format` command. See Chapter 30 or Chapter 31 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` or `fsck -n`)
- 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]
```

The table below describes the options and arguments to the generic `fsck` command.

TABLE 39–4 The `fsck` Command Options and Arguments

Option Type	Option	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 `/sbin/rcS` to determine whether a file system needs to be checked.

TABLE 39–4 The `fsck` Command Options and Arguments *(continued)*

Option Type	Option	Description
	–y or –Y or –n or –N	Runs the command automatically answering yes or no to all prompts.
	c	Converts an old pre-SunOS 4.1 file system with statically allocated tables to new 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 convert 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` 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.
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.

TABLE 39–4 The `fsck` Command Options and Arguments *(continued)*

Option Type	Option	Description
	`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.

CHAPTER 40

UFS File System Reference

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

Default Directories for root (/) and /usr File Systems

The /kernel directory contains only platform-independent objects, including a platform-independent kernel, genunix. See Table 40–3 for a description of /platform and /usr/platform, the platform-dependent directories.

The table below describes all the default directories contained in the root (/) file system.

TABLE 40–1 Default Directories in the root (/) File System

Directory	Description
/	Root of the overall file system name space
/dev	Primary location for special files

TABLE 40–1 Default Directories in the root (/) File System *(continued)*

Directory	Description
/dev/cfg	Symbolic links to physical ap_ids
/dev/cua	Device files for uucp
/dev/dsk	Block disk devices
/dev/fbs	Frame buffer device files
/dev/md	Logical volume management meta-disk devices
/dev/fd	File descriptors
/dev/pts	pty slave devices
/dev/rdsk	Raw disk devices
/dev/rmt	Raw tape devices
/dev/sad	Entry points for the STREAMS Administrative Driver
/dev/sound	Audio device and audio device control files
/dev/swap	Default swap device
/dev/term	Serial devices
/etc	Host-specific system administrative configuration files and databases
/etc/acct	Accounting configuration information
/etc/cron.d	Configuration information for cron
/etc/default	Defaults information for various programs
/etc/dmi	Solstice Enterprise Agents™ configuration files
/etc/dfs	Configuration information for shared file systems

TABLE 40–1 Default Directories in the root (/) File System *(continued)*

Directory	Description
/etc/dhcp	Dynamic Host Configuration Protocol (DHCP) configuration files
/etc/fn	Federated Naming Service and x.500 support files
/etc/fs	Binaries organized by file system types for operations required before /usr is mounted
/etc/gss	Generic Security Service (GSS) Application Program Interface configuration files
/etc/inet	Configuration files for Internet services
/etc/init.d	Scripts for changing between run levels
/etc/lib	Dynamic linking libraries needed when /usr is not available
/etc/llc2	Logical link control (llc2) driver configuration files
/etc/lp	Configuration information for the printer subsystem
/etc/mail	Mail subsystem configuration information
/etc/net	Configuration information for TI (transport- independent) network services
/etc/nfs	NFS server logging configuration file
/etc/openwin	OpenWindows™ configuration files
/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

TABLE 40–1 Default Directories in the root (/) File System *(continued)*

Directory	Description
/etc/rcS.d	Scripts for bringing the system up in single user mode
/etc/rpcsec	This directory may contain a NIS+ authentication configuration file
/etc/saf	Service access facility files (including FIFOs)
/etc/security	Basic Security Module (BSM) configuration files
/etc/skel	Default profile scripts for new user accounts
/etc/tm	Trademark files; contents displayed at boot time
/etc/uucp	uucp configuration information
/export	Default directory for users' home directories, client file systems, or other shared file systems
/home	Default directory or mount point for a user's home directory on a standalone system. When AutoFS is running, you cannot create any new entries in this directory.
/kernel	Directory 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 40–3 for the /platform and /usr/platform directory structure.
/mnt	Convenient, temporary mount point for file systems
/opt	Default directory or mount point for add-on application packages
/sbin	Essential executables used in the booting process and in manual system failure recovery
/stand	Standalone programs
/tmp	Temporary files; cleared during boot sequence

TABLE 40–1 Default Directories in the root (/) File System *(continued)*

Directory	Description
/usr	Mount point for the /usr file system. See Table 40–2 for more information.
/var	Directory for varying files, which usually includes temporary, logging, or status files
/var/adm	System logging and accounting files
/var/audit	Basic Security Module (BSM) audit files
/var/crash	Default depository for kernel crash dumps
/var/cron	cron's log file
/var/dmi	Solstice Enterprise Agents™ (SEA) Desktop Management Interface (DMI) run time components
/var/dt	dtlogin configuration files
/var/ftp	FTP server directory
/var/inet	IPv6 router state files
/var/log	System log files
/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/nfs	NFS server log files
/var/ntp	Network Time Protocol (NTP) server state directory
/var/opt	Root of a subtree for varying files associated with software packages

TABLE 40–1 Default Directories in the root (/) File System *(continued)*

Directory	Description
/var/preserve	Backup files for vi and ex
/var/run	Temporary system files that are not needed across system reboots. This is a TMPFS-mounted directory.
/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/statmon	Network status monitor files
/var/tmp	Directory for temporary files; not cleared during boot sequence
/var/uucp	uucp log and status files
/var/yp	NIS databases (for backwards compatibility with NIS and unnecessary after full transition to NIS+)

The table below describes the default directories in the /usr file system.

TABLE 40–2 Default Directories in the `/usr` File System

Directory	Description
`4lib`	SunOS 4.1 binary compatibility package libraries
`5bin`	Symbolic link to the `/usr/bin` directory
`X`	Symbolic link to the `/usr/openwin` directory
`adm`	Symbolic link to the `/var/adm` directory
`aset`	Directory for Automated Security Enhancement Tools (ASET) programs and files
`bin`	Location for standard system commands
`ccs`	C compilation programs and libraries
`demo`	Demo programs and data
`dict`	Symbolic link to the `/usr/share/lib/dict` directory, which contains the dictionary file used by the UNIX spell program
`dt`	Directory or mount point for CDE software
`games`	An empty directory, which is a remnant of the SunOS 4.0/4.1 software
`include`	Header files (for C programs, etc.)
`java*`	Directories containing Java™ programs and libraries
`kernel`	Additional kernel modules
`kvm`	Implementation architecture-specific binaries and libraries
`lib`	Various program libraries, architecture-dependent databases, and binaries not invoked directly by the user
`local`	Commands local to a site
`mail`	Symbolic link to the `/var/mail` directory

TABLE 40–2 Default Directories in the /usr File System *(continued)*

Directory	Description
man	Symbolic link to the /usr/share/man directory
net	Directory for network listener services
news	Symbolic link to the /var/news directory
oasys	Files pertaining to the Form and Menu Language Interpreter (FMLI) execution environment
old	Programs that are being phased out
openwin	Directory or mount point for OpenWindows software
perl5	Perl 5 programs and documentation
platform	See Table 40–3 for more information
preserve	Symbolic link to the /var/preserve directory
proc	Directory for the proc tools
pub	Files for online man page and character processing
sadm	Various files and directories related to system administration
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/src	Source code for kernel, libraries, and utilities
snadm	Programs and libraries related to system and network administration

TABLE 40–2 Default Directories in the /usr File System *(continued)*

Directory	Description
spool	Symbolic link to the /var/spool directory
src	Symbolic link to the share/src directory
tmp	Symbolic link to the var/tmp directory
ucb	Berkeley compatibility package binaries
ucbinclude	Berkeley compatibility package header files
ucblib	Berkeley compatibility package libraries
vmsys	Directory for Framed Access Command Environment (FACE) programs
xpg4	Directory for POSIX-compliant utilities

The Platform-Dependent Directories

The table below describes the platform-dependent objects in the /platform and /usr/platform directories.

TABLE 40–3 The /platform and /usr/platform Directories

Directory	Description
/platform	Contains a series of directories, one per supported platform that need to reside in the root (/) file system.
/platform/*/kernel	Contains platform-dependent kernel components, including the file unix, the core kernel that is platform dependent. See kernel(1M).

TABLE 40–3 The `/platform` and `/usr/platform` Directories *(continued)*

Directory	Description
`/usr/platform`	Contains platform-dependent objects that do not need to reside in the root (`/`) file system. It contains objects which replace the contents of `/usr/kvm`, which has been removed.
`/usr/platform/*/lib`	Contains platform-dependent objects similar to those found in the `/usr/lib` directory.
`/platform/*/sbin`	Contains platform-dependent objects similar to those found in the `/usr/sbin` directory.

The Structure of UFS File System Cylinder Groups

When you create a UFS file system, the disk slice is divided into *cylinder groups*, which is made up of one or more consecutive disk cylinders. The cylinder groups are then further divided into addressable 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 these four types of blocks:

This Block Type ...	Stores ...
Boot block	Information used when booting the system
Superblock	Detailed information about the file system
Inode	All information about a file
Storage or data block	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. The figure below shows this chaining of address blocks starting from the inode.

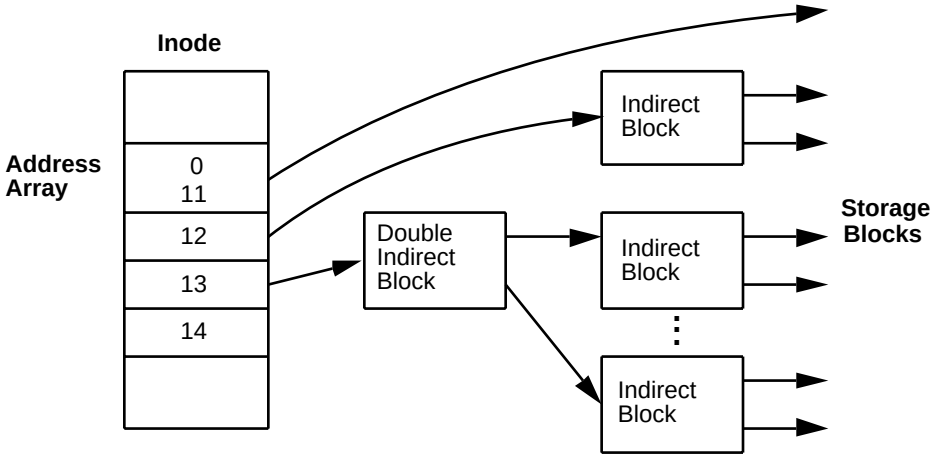


Figure 40–1 The File System Address Chain in a UFS System

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, The figure below shows a series of cylinder groups in a generic UFS file system.

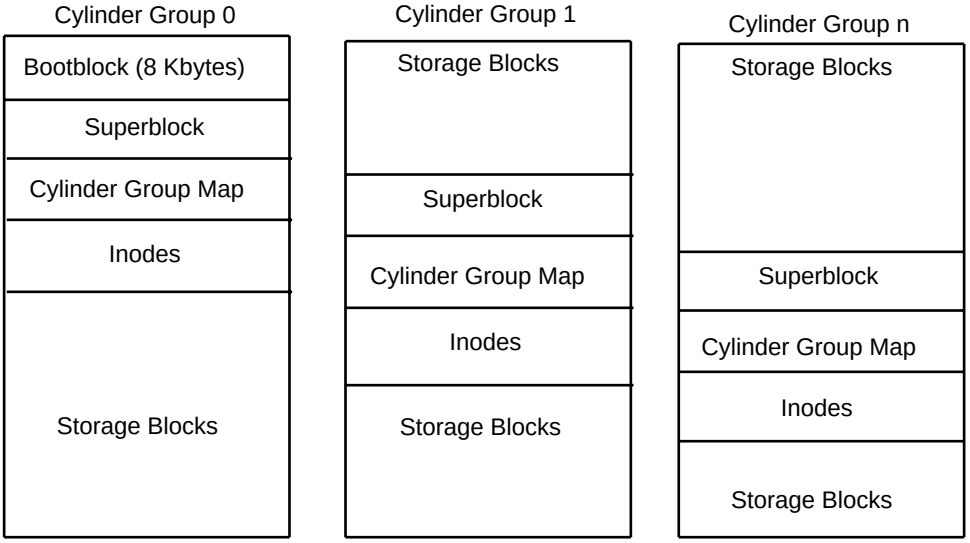


Figure 40–2 A Typical UFS File System

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 files

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.

SPARC platform only - You can only specify 8192–byte block size on the sun4u platform.

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 might equal the full block size, in which case the fragment is not a fragment at all. This configuration might 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 ((64 Mbytes/partition size) * 100), rounded down to the nearest integer and limited between 1% and 10%, inclusively. 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 is zero, as delay-based calculations are not effective when combined with modern on-disk caches.

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.
- **Time –** When you select time optimization, disk blocks are allocated as quickly as possible, with less emphasis on their placement. When there is enough free space, it is relatively easy to allocate disk blocks effectively, without resulting in too much fragmentation. The default is *time*.

  You can change the value of the optimization type parameter for an existing file system using the `tunefs` command.

Number of Files

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 recreating the file system.

The default number of bytes per inode is 2048 bytes (2 Kbytes) if the file system is less than one Gbyte. If the file system is larger than one Gbyte, the following formula is used:

File System Size	Number of Bytes Per Inode
Less than or equal to 1 Gbyte	2048
Less than 2 Gbytes	4096
Less than 3 Gbytes	6144
3 Gbytes or greater	8192

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] raw_device
```

The table below describes the options and arguments to the `newfs` command.

TABLE 40–4 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.
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.
`–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).
`–c` *cgsize*	The number of disk cylinders per cylinder group. The default value is calculated by dividing the number of sectors in the file system by the number of sectors in a gigabyte, and then multiplying the result by 32. The default value ranges from 16 to 256.
`–m` *free*	The minimum percentage of free disk space to allow. The default is ((64 Mbytes / partition size) * 100), rounded down to the nearest integer and limited between 1% and 10%, inclusively.

TABLE 40–4 The `newfs` Command Options and Arguments *(continued)*

Option	Description
`–r` *rpm*	The speed of the disk, in revolutions per minute. This setting is driver- or device-specific. If the drive can report how fast it spins, `mkfs` uses this value. If not, 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 many inodes to create. See the section above for the default values.
`–o` *opt*	Optimization type to use for allocating disk blocks to files: `space` or `time`. The default is `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 zero.
`–n` *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* x *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).
raw_device	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 or PCFS 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.

The generic `mkfs` command is located in `/usr/sbin`. See `mkfs`(1M) for a description of the arguments and options.

UFS Direct Input/Output (I/O)

Direct I/O is intended to boost bulk I/O operations. Bulk I/O operations use large buffer sizes to transfer large files (larger than 256 Kbytes).

An example of a bulk I/O operation is downloading satellite data, which writes large amounts of data to a file. Direct I/O data is read or written into memory without using the overhead of the operating system's page caching mechanism.

There is a potential penalty on direct I/O startup. If a file requested for I/O is already mapped by another application, the pages will have to be flushed out of memory before the direct I/O operation can begin.

See `directio`(3C) for more information.

Direct I/O can also be enabled on a file system by using the `forcedirectio` option to the `mount` command. Enabling direct I/O is a performance benefit only when a file system is transferring large amounts of sequential data.

When a file system is mounted with this option, data is transferred directly between a user's address space and the disk. When forced direct I/O is not enabled for a file system, data transferred between a user's address space and the disk is first buffered in the kernel address space.

The default behavior is no forced direct I/O on a UFS file system. See `mount_ufs`(1M) for more information.

▼ How to Enable Forced Direct I/O on a UFS File System

1. **Become superuser.**

2. **Mount a file system with the** `forcedirectio` **mount option.**

```
# mount -F ufs -o forcedirectio /dev/dsk/c0t3d0s7 /datab
```

3. **Verify the mounted file system has forced direct I/O enabled.**

```
# mount
                    .
                    .
                    .
/export/home on /dev/dsk/c0t3d0s7 read/write/setuid/forcedirectio ...
```

CHAPTER 41

Backing Up and Restoring Data Topics

This section provides instructions for backing up and restoring data in the Solaris environment. This section contains these chapters.

Chapter 42	Provides guidelines and planning information on backing up and restoring data using the `ufsdump` and `ufsrestore` commands.
Chapter 43	Provides step-by-step instructions for backing up individual files and complete file systems from local or remote devices.
Chapter 44	Provides step-by-step instructions for restoring individual files and complete file systems.
Chapter 45	Describes how `ufsdump` works, and the syntax and options for the `ufsdump` and `ufsrestore` commands.
Chapter 46	Provides step-by-step instructions for copying file systems to disk, for using the `dd`, `cpio`, and `tar` commands with different backup media, and copying files with a different header format.
Chapter 47	Provides step-by-step instructions for how to add a tape drive, how to determine the type of tape drive, backup device names, and working with tape drives and magnetic tape cartridges.

CHAPTER 42

Backing Up and Restoring File Systems (Overview)

This chapter provides guidelines and planning information on backing up and restoring complete file systems using the `ufsdump` and `ufsrestore` commands.

Here is a list of concept information in this chapter.

Where to Find Backup and Restore Tasks

Use Chapter 43 and Chapter 44 to find step-by-step instructions for backing up and restoring file systems (using the `ufsdump` and `ufsrestore` commands).

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, other commands are available for copying files and file systems for sharing or transporting files. The table below provides pointers to all commands that copy individual files and/or file systems to media.

TABLE 42–1 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`(1M) command	Chapter 43 or Chapter 45
Back up complete file systems for all systems on a network from a server	Solstice Backup™ software	*Solstice Backup 5.1 Administration Guide*
Back up and restore a NIS+ master server	`nisbackup`(1M) and `nisrestore`(1M) commands	*Solaris Naming Administration Guide*
Copy, list, and retrieve files on tape Copy, list, and retrieve files on diskette	`tar`(1), `cpio`(1), or `pax`(1) command `tar`(1) command	Chapter 46
Copy master disk to a clone disk	`dd`(1M) command	Chapter 46
Restore complete file systems or individual files from removable media to a working directory	`ufsrestore`(1M) command	Chapter 44

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, earthquakes)
- Problems when reinstalling or upgrading a system

Choosing a Tape Device

The table below shows typical tape devices used for storing file systems during the backup process. Capacity depends on the type of drive and the data being written to the tape. For more detailed information on tape devices, see Chapter 47.

TABLE 42–2 Typical Media for Backing Up File Systems

Media	Capacity
1/2-inch reel tape	140 Mbytes (6250 bpi)
2.5-Gbyte 1/4 inch cartridge (QIC) tape	2.5 Gbytes
DDS3 4-mm cartridge tape (DAT)	12 - 24 Gbytes
14-Gbyte 8-mm cartridge tape	14 Gbytes
DLT™ 7000 1/2-inch cartridge tape	35 - 70 Gbytes

Planning Which File Systems to Back Up

You should back up all file systems that are critical to users, including file systems that change frequently. The tables below provide general guidelines on the file systems to back up for standalone systems and servers.

TABLE 42–3 File Systems to Back Up for Standalone Systems

Consider Backing Up These File Systems ...	Because ...	And At This Interval ...
root (/) – partition 0	The root (/) file system contains the kernel and might contain the /var directory in which frequently modified files such as mail and accounting are kept.	At regular intervals.
/usr – partition 6, /opt	Installing new software and adding new commands typically affects the /usr and /opt file systems. /opt is either part of root (/) or is its own file system.	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 might have created these file systems.	As your site requires.

TABLE 42–4 File Systems to Back Up for Servers

Consider Backing Up These File Systems ...	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's 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 backup 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 might want to back up root (/) daily (or /var, if it is a separate file system). /export - the root (/) directory of 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 delivers mail to client systems; then you should back up /export more frequently. /usr and /opt - contents are fairly static and need to be backed up 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.

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. The table below lists the tasks you can perform with them. For information on how these commands work and their syntax, see Chapter 45.

TABLE 42–5 Tasks You Can Perform With the `ufsdump` and `ufsrestore` Commands

With This Command ...	You Can ...	Comments
`ufsdump`	Back up complete or partial 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 interface.
	Back up incremental file system changes	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 or a file.
	Automate backups	Use the `crontab` utility to run 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 of when each file system was backed up	Use the `–u` option.

TABLE 42–5 Tasks You Can Perform With the `ufsdump` and `ufsrestore` Commands *(continued)*

With This Command ...	You Can ...	Comments
	Verify the 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. The table below lists the differences between these types of backup procedures.

TABLE 42–6 Differences Between Full and Incremental Backups

Backup Type	Copies	Advantages	Disadvantages
Full	A complete file system or directory	Everything is in one place	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. You might have to search multiple tapes.
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 contains a file can take time. You might have to go back to last full dump.

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 are accidentally deleted

How Often Should You Do Backups?

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 most sites, so incremental backups are used most often. In this case, you should back up your site enough to restore files from the last four weeks. This requires at least four sets of tapes—one for each week, which you would reuse each month. In addition, you should archive the monthly backups for at least a year, and then keep yearly backups 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.

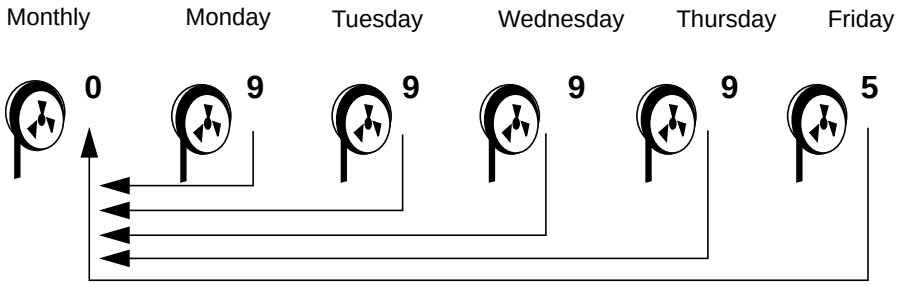


Figure 42–1 Incremental Backup: Daily Cumulative

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 (2) 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 numbers themselves have no defined meaning; you attribute meaning by ordering them in a high/low sequence.

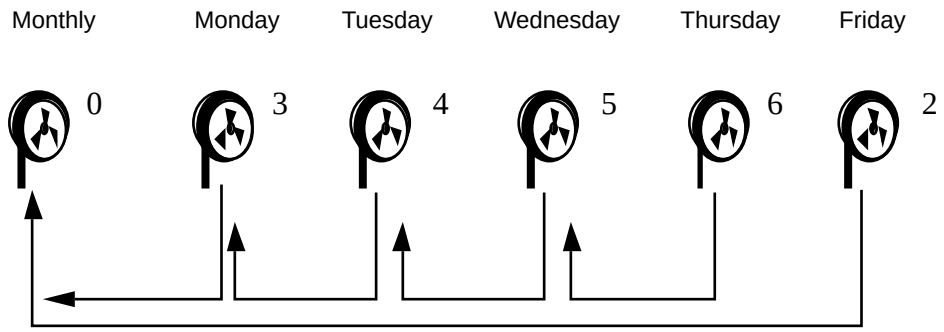


Figure 42–2 Incremental Backup: Daily Discrete

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

The table below shows the most commonly used incremental backup schedule; it is recommended for most situations. With this schedule:

- All files that have changed since the lower-level backup at the end of the previous week are saved each day.
- For each weekday level 9 backup, the previous level 0 or level 5 is the closest backup at a lower level. Therefore, each weekday tape contains all the files changed since the end of the previous week (or the initial level 0 for the first week).
- For each Friday level 5 backup, 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 42–7 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

The table below shows how the contents of the tapes can change across two weeks using the previous schedule. Each letter represents a different file.

TABLE 42–8 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

Tape Requirements

With this schedule, you 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 the following tapes: the level 0, the most recent Friday tape, and the most recent daily tape since the last Friday tape (if any).

Example—Daily Cumulative, Weekly Incremental Backups

The table below 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 42–9 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	4
Week 3		9	9	9	9	5
Week 4		9	9	9	9	6

The table below shows how the contents of the tapes can change across two weeks using the previous schedule. Each letter represents a different file.

TABLE 42–10 Contents of Tapes for Daily Cumulative / Weekly Incremental Backup 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 Requirements

With this schedule, you 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 need the following tapes: the level 0, all the Friday tapes, and the most recent daily tape since the last Friday tape (if any).

Example—Daily Incremental, Weekly Cumulative Backups

The table below 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 42–11 Daily Incremental / Weekly Cumulative Backup Schedule

	Floating	Mon	Tues	Wed	Thurs	Fri
1st of Month	0					
Week 1		3	4	5	6	2
Week 2		3	4	5	6	2

TABLE 42–11 Daily Incremental / Weekly Cumulative Backup Schedule *(continued)*

	Floating	Mon	Tues	Wed	Thurs	Fri
Week 3		3	4	5	6	2
Week 4		3	4	5	6	2

The table below shows how the contents of the tapes can change across two weeks using the previous schedule. Each letter represents a different file.

TABLE 42–12 Contents of Tapes for Daily / Weekly Cumulative Backup Schedule

	Mon	Tues	Wed	Thurs	Fri
Week 1	a b	c d	e f g	hi	a b c d e f g h i
Week 2	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 Requirements

With this schedule you need at least nine tapes (if you want to reuse daily tapes—not recommended), or 21 tapes (if you save weekly tapes for a month): one for the level 0, four for the Fridays, and four or 16 daily tapes.

If you need to restore the complete file system, you need the following tapes: the level 0, the most recent Friday tape, and all the daily tapes since the last Friday tape (if any).

Example—Backup Schedule for a Server

The table below 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 42–13 Schedule of Backups for a Server Example

Directory	Date	Level	Tape Name
root (/)	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
root (/)	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
root (/)	3rd Sunday	0	*n* tapes

TABLE 42–13 Schedule of Backups for a Server Example *(continued)*

Directory	Date	Level	Tape Name
/usr	3rd Sunday	0	"
/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	5	R
root (/)	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 root (/), /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 root (/), /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 Suggestions

The table below provides other suggestions for scheduling backups.

TABLE 42–14 Other Suggestions 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 lower level. However, with this schedule you should use a different tape 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 0 backups on the same day.
Need to minimize tapes	Increase the level of incremental backups done across the week.	This means only changes from day to day are saved on each daily tape.
	Increase the level of backups done at the end of the week.	This means only changes from week to week (rather than the entire month) are saved on the weekly tapes.
	Put each day's and week's incremental backups onto the same tape.	This is done by using the no rewind option in the `ufsdump` command.

CHAPTER 43

Backing Up Files and File Systems (Tasks)

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 45.

Preparing to Do Backups

Preparing to back up file systems begins with planning, which is described in Chapter 42 and includes choosing:

- A tape drive
- The file systems to back up
- The type of backup (full or incremental)
- A backup schedule

This section describes other tasks you might 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 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
#
#/dev/dsk/c1d0s2  /dev/rdsk/c1d0s2   /usr         ufs    1      yes     -
fd                -                  /dev/fd      fd     -      no      -
/proc             -                  /proc        proc   -      no      -
/dev/dsk/c0t0d0s1 -                  -            swap   -      no      -
/dev/dsk/c0t0d0s0 /dev/rdsk/c0t0d0s0 /            ufs    1      no      -
/dev/dsk/c0t0d0s6 /dev/rdsk/c0t0d0s6 /usr         ufs    1      no      -
/dev/dsk/c0t0d0s7 /dev/rdsk/c0t0d0s7 /export/home ufs    2      yes     -
mars:/share/kit   -                  /kit         nfs    -      yes     -
mars:/db/doc      -                  /db/doc      nfs    -      yes     -
```

▼ How to Determine the Number of Tapes for a Full Backup

1. **Become superuser.**

2. **Estimate the size of the backup in bytes by using the** `usfdump S` **command.**

```
# ufsdump S filesystem
```

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 42–2 for a list of tape capacities.

Example—Determining Number of Tapes

In this example, the file system of 489,472 bytes will easily 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 occurring means that 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 46 for information about using the `dd` command.
- If you are doing remote backups across the network, the system with the tape drive must have entries in its `/.rhosts` file for each client that will be using the drive. Also, the system initiating the backup must be included in the `/.rhosts` file on each system 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 the SunOS 4.1.1 release or compatible versions; use `/dev/rmt/0` for a system running the Solaris 8 release or compatible versions.

Note - Use the `nisbackup` command to back up a NIS+ master server running the Solaris 2.5 release or compatible versions. See *Solaris Naming Administration Guide* for information on using this command.

▼ How to Do Backups to 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 superuser.**

2. **Bring the system to run level S (single-user mode).**

```
# shutdown -g30 -y
```

3. **[Optional] Check the file system for consistency with the `fsck` command.**

   Running the `fsck -m` command 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.

```
# fsck -m /dev/rdsk/ device-name
```

4. **If you need to back up file systems to a remote tape drive:**

   a. **On the system to which the tape drive is attached (the tape server), add the following entry to its `/.rhosts` file.**

```
host root
```

host	Specifies the name of the system on which you will run `ufsdump` to perform the backup.

   b. **On the tape server, verify that the host added to the `/.rhosts` file is accessible through the name service.**

5. **Identify the device name of the tape drive.**

   The default tape drive is `/dev/rmt/0`.

6. **Insert a tape that is not write protected into the tape drive.**

7. **Back up file systems using the `ufsdump` command.**

   Use the following table to select the most common options and arguments for the `ufsdump` command. See Chapter 45 for other options and arguments.

To ...	Use This Option or Argument ...	For Example ...	See ...
Do a full backup	0 option	`ufsdump 0ucf /dev/rmt/0 /`	"Example—Full Backup, root (/)" on page 576
Do an incremental backup	1-9 option	`ufsdump 9ucf /dev/rmt/0 /`	"Example—Incremental Backup, root (/) " on page 576
Back up individual files	Specify a file or directory	`ufsdump ucf /dev/rmt/0 /export/home/kryten`	
Record dumps to `/etc/dumpdates` file	`–u` option	`ufsdump 9ucf /dev/rmt/0 /export/home`	"Example—Incremental Backup, root (/) " on page 576
Specify a cartridge tape	`–c` option	`ufsdump 9ucf /dev/rmt/0 /export/home`	"Example—Incremental Backup, root (/) " on page 576
Specify the tape drive	`–f` *dump-file*	`ufsdump 9ucf /dev/rmt/0 /export/home`	"Example—Incremental Backup, root (/) " on page 576
Back up local file systems to a remote system's tape device	*remote-system*:*dump-file*	`ufsdump 0ucf pluto:/dev/rmt/0 /export/home`	"Example—Full Backup to Remote System (Solaris 8 Data to Solaris 8 System)" on page 578

8. **If prompted, remove the tape and replace with the next volume.**

9. **Label each tape with the volume number, level, date, system name, disk slice, and file system.**

10. **Bring the system back to run level 3 by pressing Control-d.**

11. **Verify the backup was successful by using the `ufsrestore` command to display the tape contents.**

    This command is described in Chapter 44.

Example—Full Backup, root (/)

The following example shows a full backup of the root (/) file system to a QIC–150 tape (/dev/rmt/0).

```
# shutdown -g30 -y
# ufsdump 0ucf /dev/rmt/0 /
  DUMP: Writing 63 Kilobyte records
  DUMP: Date of this level 0 dump: Tue Jul 13 10:46:09 1999
  DUMP: Date of last level 0 dump: the epoch
  DUMP: Dumping /dev/rdsk/c0t0d0s0 (starbug:/) to /dev/rmt/0.
  DUMP: Mapping (Pass I) [regular files]
  DUMP: Mapping (Pass II) [directories]
  DUMP: Estimated 71058 blocks (34.70MB).
  DUMP: Dumping (Pass III) [directories]
  DUMP: Dumping (Pass IV) [regular files]
  DUMP: Tape rewinding
  DUMP: 70936 blocks (34.64MB) on 1 volume at 64 KB/sec
  DUMP: DUMP IS DONE
  DUMP: Level 0 dump on Tue Jul 13 10:46:09 1999
  # ufsrestore tf /dev/rmt/0
         2      .
         3      ./lost+found
      5696      ./usr
     11392      ./var
     17088      ./export
     22784      ./export/home
     28480      ./opt
      5697      ./etc
     11393      ./etc/default
     11394      ./etc/default/sys-suspend
     11429      ./etc/default/cron
     11430      ./etc/default/devfsadm
     11431      ./etc/default/dhcpagent
     11432      ./etc/default/fs
     11433      ./etc/default/inetinit
     11434      ./etc/default/kbd
     11435      ./etc/default/nfslogd
     11436      ./etc/default/passwd
                .
                .
                .
  # (Press Control-d to bring system to run level 3)
```

Example—Incremental Backup, root (/)

The following example shows an incremental backup of the root (/) file system to a 4-mm DAT tape (/dev/rmt/0).

```
# ufsdump 9ucf /dev/rmt/0 /
  DUMP: Writing 63 Kilobyte records
  DUMP: Date of this level 9 dump: Tue Jul 13 10:58:12 1999
  DUMP: Date of last level 0 dump: Tue Jul 13 10:46:09 1999
  DUMP: Dumping /dev/rdsk/c0t0d0s0 (starbug:/) to /dev/rmt/0.
  DUMP: Mapping (Pass I) [regular files]
```

(Continuation)

```
  DUMP: Mapping (Pass II) [directories]
  DUMP: Mapping (Pass II) [directories]
  DUMP: Mapping (Pass II) [directories]
  DUMP: Mapping (Pass II) [directories]
  DUMP: Estimated 200 blocks (100KB).
  DUMP: Dumping (Pass III) [directories]
  DUMP: Dumping (Pass IV) [regular files]
  DUMP: Tape rewinding
  DUMP: 124 blocks (62KB) on 1 volume at 8 KB/sec
  DUMP: DUMP IS DONE
  DUMP: Level 9 dump on Tue Jul 13 10:58:12 1999
# ufsrestore tf /dev/rmt/0
         2      .
         3      ./lost+found
      5696      ./usr
     11392      ./var
     17088      ./export
     22784      ./export/home
     28480      ./opt
      5697      ./etc
     11393      ./etc/default
     11394      ./etc/default/sys-suspend
     11429      ./etc/default/cron
     11430      ./etc/default/devfsadm
     11431      ./etc/default/dhcpagent
     11432      ./etc/default/fs
     11433      ./etc/default/inetinit
     11434      ./etc/default/kbd
     11435      ./etc/default/nfslogd
     11436      ./etc/default/passwd
     11437      ./etc/default/tar
                .
                .
                .
```

Example—Full Backup, Individual Home Directory

The following example shows a full backup of the `/export/home/kryten` directory to a 4-mm DAT tape.

```
# ufsdump 0ucf /dev/rmt/0 /export/home/kryten
  DUMP: Writing 63 Kilobyte records
  DUMP: Date of this level 0 dump: Tue Jul 13 11:30:45 1999
  DUMP: Date of last level 0 dump: the epoch
  DUMP: Dumping /dev/rdsk/c0t3d0s7 (pluto:/export/home) to /dev/rmt/0.
  DUMP: Mapping (Pass I) [regular files]
  DUMP: Mapping (Pass II) [directories]
  DUMP: Estimated 232 blocks (116KB).
  DUMP: Dumping (Pass III) [directories]
  DUMP: Dumping (Pass IV) [regular files]
```

(continued)

(Continuation)

```
  DUMP: Tape rewinding
  DUMP: 124 blocks (62KB) on 1 volume at 8 KB/sec
  DUMP: DUMP IS DONE
# 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 8 Data to Solaris 8 System)

The following example shows a full backup of a local `/export/home` file system on a Solaris 8 system to a tape device on a remote Solaris 8 system called `starbug`.

```
# ufsdump 0ucf starbug:/dev/rmt/0 /export/home
DUMP: Writing 63 Kilobyte records
DUMP: Date of this level 0 dump: Tue Jul 13 13:14:40 1999
DUMP: Date of last level 0 dump: the epoch
DUMP: Dumping /dev/rdsk/c0t3d0s7 (mars:/export/home) to starbug:/dev/rmt/0
DUMP: Mapping (Pass I) [regular files]
DUMP: Mapping (Pass II) [directories]
DUMP: Estimated 476 blocks (238KB).
DUMP: Dumping (Pass III) [directories]
DUMP: Dumping (Pass IV) [regular files]
DUMP: Tape rewinding
DUMP: 376 blocks (188KB) on 1 volume at 21 KB/sec
DUMP: DUMP IS DONE
DUMP: Level 0 dump on Tue Jul 13 13:14:40 1999
 # ufsrestore tf starbug:/dev/rmt/0
         2      .
         3      ./lost+found
      3776      ./kryten
      3777      ./kryten/.cshrc
      3778      ./kryten/.login
      3779      ./kryten/b
      3780      ./kryten/memos
      7552      ./kryten/letters
      7553      ./kryten/letters/b
      7554      ./kryten/letters/letter1
```

(continued)

```
      7555      ./kryten/letters/letter2
      7556      ./kryten/letters/letter3
     11328      ./kryten/reports
     11329      ./kryten/reports/reportA
     11330      ./kryten/reports/reportB
     11331      ./kryten/reports/reportC
                .
                .
                .
#
```

Example—Full Backup to Remote System (Solaris 8 Data to SunOS 4.1.4 System)

The following example shows a full backup of a local `/export/home` file system on a Solaris 8 system to a tape device on a remote SunOS 4.1.4 system (`mars`).

Note - Notice the SunOS 4.*x*-style device name (`/dev/rst0`) used with the `ufsdump` command.

```
# ufsdump 0ucf mars:/dev/rst0 /export/home
  DUMP: Writing 63 Kilobyte records
  DUMP: Date of this level 0 dump: Thu Jul 15 09:13:01 1999
  DUMP: Date of last level 0 dump: the epoch
  DUMP: Dumping /dev/rdsk/c0t0d0s7 (starbug:/export/home) to mars:/dev/
rst0.
  DUMP: Mapping (Pass I) [regular files]
  DUMP: Mapping (Pass II) [directories]
  DUMP: Estimated 5690 blocks (2.78MB).
  DUMP: Dumping (Pass III) [directories]
  DUMP: Dumping (Pass IV) [regular files]
  DUMP: Tape rewinding
  DUMP: 5542 blocks (2.71MB) on 1 volume at 77 KB/sec
  DUMP: DUMP IS DONE
  DUMP: Level 0 dump on Thu Jul 15 09:13:01 1999
# 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
```

(continued)

(Continuation)

```
        32      ./kryten/reports/reportB
        33      ./kryten/reports/reportC
                .
                .
                .
#
```

Example—Full Backup to Remote System (SunOS 4.1.4 Data to Solaris 8)

The following example shows a full backup of a local root (/) file system on a Sun 4.1.4 system (`mars`) to a remote tape device on a Solaris 8 system called `starbug`.

Note - When you back up data on a system running SunOS 4.1.4 or a compatible version, you must use the `dump` command—not the `ufsdump` command.

```
# dump 0ucf starbug:/dev/rmt/0 /
  DUMP: Date of this level 0 dump: Wed Jul  7 06:19:33 1999
  DUMP: Date of last level 0 dump: the epoch
  DUMP: Dumping /dev/rsd0a (/) to /dev/rmt/0 on host starbug
  DUMP: mapping (Pass I) [regular files]
  DUMP: mapping (Pass II) [directories]
  DUMP: estimated 123706 blocks (60.40MB) on 1.41 tape(s).
  DUMP: dumping (Pass III) [directories]
  DUMP: dumping (Pass IV) [regular files]
  DUMP: level 0 dump on Wed Jul  7 06:19:33 1999
  DUMP: Tape rewinding
  DUMP: 123680 blocks (60.39MB) on 2 volumes
  DUMP: DUMP IS DONE
# restore tf starbug:/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.4.1.4
      7553      ./etc
      7554      ./etc/sendmail.cf
      7555      ./etc/aliases
      7556      ./etc/aliases.dir
      7557      ./etc/aliases.pag
      7558      ./etc/holidays
      7559      ./etc/dumpdates
                .
```

(continued)

(Continuation)

```
                    .
                    .
#
```

CHAPTER 44

Restoring Files and File Systems (Tasks)

This chapter describes the procedures for restoring file systems.

Here 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 46 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 to disk, relative to the current working directory, from backups created using the `ufsdump` command. 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 as superuser, 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:

- The tapes (or diskettes) you need
- The raw device name on which you want to restore the file system
- 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 572 for more information.

Determining the Type of Tape Drive You Need

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 might 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 47 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 after a disk head crash. You might need to replace the hardware before you can restore the software. See Chapter 30 or Chapter 31 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 consistently backed up file systems, you can restore them to their state from the time of the last incremental backup.

Restoring Individual Files and Directories

When you back up files and directories, you save them relative to the file system in which they belong. When you restore files and directories, `ufsrestore` recreates the file hierarchy in the current working directory. For example, files backed up from the `/export/doc/books` directory (where `/export` is the file system), would be saved relative to `/export`. In other words, the `book1` file in the `docs` directory would be saved as `./doc/books/book1` on the tape. Later on, if you restored the `./doc/books/book1` file to the `/var/tmp` directory, the file would be restored to `/var/tmp/doc/books/book1`.

When restoring individual files and directories, it is a good idea to restore them to a temporary location, such as the `/var/tmp` directory. 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.

Note - Do not restore files in the `/tmp` directory even temporarily. The `/tmp` directory is usually mounted as a TMPFS file system and TMPFS does not support UFS file system attributes such as ACLs.

Restoring Files and File Systems

Things you need to know:

- The tapes that 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 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.**

   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, unless the user requests otherwise.

3. **If you have online archive files, use the** `ufsrestore` **command to identify correct media.**

```
# ufsrestore ta archive-name ./path/filename ./path/filename
```

`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` prints out the inode number and file name. If unsuccessful, `ufsrestore` prints 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 /dev/rmt/`***n*** **option to position the tape at the dump you want to use.**

```
# ufsrestore tfs /dev/rmt/n tape_number
```

Example—Determining Which Tapes to Use

If you use `ufsdump` to dump the `/usr` file system, the table of contents lists only the files and directories under `/usr`. The following example checks if `/usr/bin/pwd` is in the online archive.

```
# ufsrestore ta archive-name ./bin/pwd
```

The following example checks if `/usr/bin/pwd` is on the backup tape.

```
# ufsrestore tf /dev/rmt/n ./bin/pwd
```

▼ How to Restore Files Interactively

1. **Become superuser.**

2. **Write-protect the tape.**

3. **Insert the volume 1 tape into the tape drive.**

4. **Change to a directory that will be used to restore the files temporarily.**

```
# cd /var/tmp
```

   To avoid conflicts with other users, you might want to create and change to a subdirectory, such as `/var/tmp/restore`, in which to restore the files.

   If you are restoring a hierarchy, you should restore the files in a temporary directory on the same file system where the files will reside, so you can use the `mv` command to move the entire hierarchy where it belongs after it is restored.

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. **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
```

    You must wait while `ufsrestore` performs its final cleanup.

11. **Quit the `ufsrestore` program.**

```
ufsrestore> quit
```

    You then see the shell prompt.

12. **Verify the restored files.**
    a. **List the restored files and directories.**

```
# ls -l
```

       A list of files and directories is appears.

    b. **Check the list to be sure all the files and directories you specified in the list have been restored.**

    c. **Move the files to the proper directories.**

Example—Restoring Files Interactively

The following example extracts the files `/etc/passwd` and `/etc/shadow` from the backup tape.

```
# cd /var/tmp
# ufsrestore if /dev/rmt/0
ufsrestore> ls
.:
 .cpr_config   etc/          lost+found/   sbin/         usr/
 TT_DB/        export/       mnt/          sccs/         var/
 b/            home/         net/          share/        vol/
 bin           kernel/       opt/          shared/       ws/
 dev/          lib           platform/     src/          xfn/
 devices/      license/      proc/         tmp/
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 Non-Interactively

1. **Become superuser.**

2. **Write-protect the tape for safety.**

3. **Insert the volume 1 tape into the tape drive.**

4. **Change to a directory for restoring files temporarily.**

   ```
   # cd /var/tmp
   ```

   To avoid conflicts with other users, you might want to create and change to a subdirectory, such as `/var/tmp/restore`, in which to restore the files.

If you are restoring a hierarchy, you should restore the files in a temporary directory on the same file system where the files will reside, so you can use the `mv` command to move the entire hierarchy where it belongs after it is restored.

5. **Use the `ufsrestore` command to restore the file.**

```
# ufsrestore xvf /dev/rmt/n filename ...
```

`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 ...	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
```

8. **Verify the restored files.**
   a. **List the restored files and directories.**

```
# ls -l
```

   A list of files and directories is displayed.

   b. **Check the list to be sure all the files and directories you specified in the list have been restored.**

   c. **Move the files to the proper directories.**

Example—Restoring Specific Files Non-Interactively

The following example restores the `passwd` and `shadow` files 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 Jul 14 08:42:42 1999
Dumped from: the epoch
Level 0 dump of a partial file system on starbug:/etc
Label: none
Extract directories from tape
Initialize symbol table.
Make node ./etc
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
extract file ./etc/shadow
Add links
Set directory mode, owner, and times.
set owner/mode for '.'? [yn] n
Directories already exist, set modes anyway? [yn] n
# cd etc
# mv passwd /etc
# mv shadow /etc
# ls -l /etc
```

▼ How to Restore Files Using a Remote Tape Drive

You can restore files from a remote tape drive by adding *remote-host*: to the front of the tape device name, when using the `ufsrestore` command.

```
ufsrestore xf [user@]remote-host:/dev/rmt/n filename
```

Example—Restoring Files Using a Remote Drive

The following example restores files using a remote tape drive `/dev/rmt/0` on the system `venus`.

```
# ufsrestore xf venus:/dev/rmt/0 filename
```

▼ How to Restore a Complete File System

Note - You cannot use this procedure to restore root (/) or /usr. See "How to Restore the root (/) and /usr File Systems" on page 595 for instructions on restoring these file systems.

1. **Become superuser.**

2. **If necessary, unmount the file system.**

```
# umount /dev/rdsk/device-name
```

3. **Create the new file system with the** newfs**(1M) 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 destroy the wrong file system.

4. **Confirm that the new file system should be created.**

```
newfs: construct a new file system /dev/rdsk/cwtxdysz:(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 into 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 would be prompted to load each tape in numeric order.

10. Remove the tape and load the next level tape in the drive.

Always restore tapes starting with 0 and continuing until you reach the highest level.

11. Repeat Step 7 on page 592 through Step 10 on page 593 for each level of dump, from the lowest to the highest level.

12. Verify the file system is restored.

```
# ls
```

13. Remove the `restoresymtable` file.

```
# rm restoresymtable
```

The `restoresymtable` file created by `ufsrestore` is removed.

14. Change to another directory.

```
# cd /
```

15. Unmount the newly restored file system.

```
# umount /mnt
```

16. Remove the last tape and insert a new tape that is not write-protected in the tape drive.

17. Use the `ufsdump` command to make a level 0 backup of 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 (the restored file system will appear to have changed since the previous backup).

18. Mount the restored file system.

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

The restored file system is mounted and available for use.

19. Verify the restored and mounted file system is available.

```
# ls mount-point
```

Example—Restoring a Complete File System

The following example restores the `/export/home` file system.

```
# umount /export/home
# newfs /dev/rdsk/c0t3d0s7
newfs: construct a new file system /dev/rdsk/c0t3d0s7: (y/n)? y
/dev/rdsk/c0t3d0s7:
410400 sectors in 270 cylinders of 19 tracks, 80 sectors
  200.4MB in 17 cyl groups (16 c/g, 11.88MB/g, 5696 i/g)
super-block backups (for fsck -F ufs -o b=#) at:
 32, 24432, 48832, 73232, 97632, 122032, 146432, 170832, 195232, 219632,
 244032, 268432, 292832, 317232, 341632, 366032, 390432,
# mount /dev/dsk/c0t3d0s7 /mnt
# cd /mnt
# ufsrestore rvf /dev/rmt/0
Verify volume and initialize maps
Media block size is 126
Dump   date: Wed Jul 14 08:49:33 1999
Dumped from: the epoch
Level 0 dump of /export/home on earth:/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/reports
Extract new leaves.
Check pointing the restore
extract file ./kryten/.cshrc
extract file ./kryten/.login
extract file ./kryten/b
extract file ./kryten/memos
extract file ./kryten/letters/b
extract file ./kryten/letters/letter1
extract file ./kryten/letters/letter2
extract file ./kryten/letters/letter3
extract file ./kryten/reports/reportA
extract file ./kryten/reports/reportB
extract file ./kryten/reports/reportC
```

(continued)

```
Add links
Set directory mode, owner, and times.
Check the symbol table.
Check pointing the restore
# ls
# 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 30 or Chapter 31.

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. **Write-protect the tapes.**

5. **Use the ufsrestore command to restore the root file system.**

   ```
   # ufsrestore rvf /dev/rmt/n
   ```

   The level 0 tape is restored.

6. **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.

7. **Continue to use the** `ufsrestore` **command as needed.**

```
# ufsrestore rvf /dev/rmt/n
```

The next level tape is restored.

8. **Repeat Step 6 on page 595 and Step 7 on page 596 for each additional tape.**

9. **Verify the file system is restored.**

```
# ls
```

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. **Create the boot blocks on the root partition by using the** `installboot`**(1M) command.**

```
# installboot  /usr/platform/'uname-i'/lib/fs/ufs/bootblk /dev/rdsk/devicename
```

See "SPARC: Example—Restoring the root (/) File System" on page 597 for an example of using the `installboot` command on a SPARC based system or "IA: Example—Restoring the root (/) File System" on page 597 for an example of using the `installboot` command on an IA based system.

15. **Insert a new tape in the tape drive.**

16. **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.

17. **Repeat steps 5 through 18 for the `/usr` file system, if necessary.**

18. **Reboot the system.**

```
# init 6
```

The system is rebooted.

SPARC: Example—Restoring the root (/) File System

```
# mount /dev/dsk/c0t3d0s0 /mnt
# cd /mnt
# tapes
# ufsrestore rvf /dev/rmt/0
# ls
# rm restoresymtable
# cd /
# umount /mnt
# fsck /dev/rdsk/c0t3d0s0
# installboot /usr/platform/sun4m/lib/fs/ufs/bootblk /dev/rdsk/c0t3d0s0
# ufsdump 0uf /dev/rmt/0 /dev/rdsk/c0t3d0s0
# init 6
```

IA: Example—Restoring the root (/) File System

```
# mount /dev/dsk/c0t3d0s0 /mnt
# cd /mnt
# tapes
# ufsrestore rvf /dev/rmt/0
# ls
# rm restoresymtable
# cd /
# umount /mnt
# fsck /dev/rdsk/c0t3d0s0
# installboot /usr/platform/‘uname -i‘/lib/fs/ufs/pboot /usr/platform/‘uname -i‘/lib/fs/
```

(continued)

(Continuation)

```
ufs/bootblk /dev/rdsk/c0t3d0s0
# ufsdump 0uf /dev/rmt/0 /dev/rdsk/c0t3d0s0
# init 6
```

CHAPTER 45

The ufsdump and ufsrestore Commands (Reference)

This chapter contains reference information on the ufsdump and ufsrestore commands.

Here is a list of 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 With `ufsdump`

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.

Role of the `/etc/dumpdates` File

The `ufsdump` command, when used with the `–u` option, maintains and updates the `/etc/dumpdates` file. 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/c0t0d0s0              9 Tue Jul 13 10:58:12 1999
/dev/rdsk/c0t0d0s0              0 Tue Jul 13 10:46:09 1999
/dev/rdsk/c0t0d0s1              0 Tue Jul 13 13:41:04 1999
```

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 modified 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.

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 in order to restore the entire file system.

Note - The `/etc/dumpdates` file 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 might 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 that might 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 640 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 on the remote system. If you specify the device as *user@host*:*device*, `ufsdump` tries to access the device on the remote system as the specified user. In this case, the specified user must be included in the `/.rhosts` file on 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 device name (for example, `/dev/rst0`). If the system is running Solaris software, use the SunOS 5.8 convention (for example, `/dev/rmt/0`).

Note - You must specify remote devices explicitly with the *dump-file* argument. In previous SunOS releases, the `rdump` command directed the output to the remote device defined by the dumphost alias. `ufsdump` does not have an `rufsdump` counterpart.

Using Standard Output With `ufsdump`

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 640 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 or files (rather than a whole file system), a level 0 backup is done. Incremental backups do not apply.

End-of-Media Detection

The `ufsdump` command 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. 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.

Limitations of the `ufsdump` Command

The table below lists tasks you cannot perform with the `ufsdump` command.

TABLE 45–1 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 might 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 `ufsdump` 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

The table below describes the options for the `ufsdump` command.

TABLE 45–2 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 at a time.
`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.

TABLE 45–2 Options for the `ufsdump` Command *(continued)*

Option	Description
`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).
`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 `/etc/dumpdates` file. 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 only on an unmounted file system, because any activity in the file system causes it to report discrepancies.

TABLE 45–2 Options for the `ufsdump` Command *(continued)*

Option	Description
`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 `/.rhosts` files on clients and servers if doing centralized backups.

  For general information on security, see "Managing System Security (Overview)" in *System Administration Guide, Volume 2*.

Options and Arguments for the `ufsrestore` Command

`ufsrestore` Command Syntax

The syntax of the `ufsrestore` command is:

```
ufsrestore [options][arguments][filename ...]
```

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	Specifies files to be restored as arguments to the `x` or `t` options, and must always come last.

`ufsrestore` Options and Arguments

You must use one (and only one) of the `ufsrestore` options shown in the table below.

TABLE 45–3 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-like commands to browse the contents of the media and select individual files or directories to restore. See “Commands for Interactive Restore” on page 610 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 (for example, `restoresymtable`) is also included. To completely restore a file system, use this option to restore the full (level 0) dump and each subsequent incremental dump. Although intended for a new file system (one just created with the `newfs` command), 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.

TABLE 45–3 One Required Option for the `ufsrestore` Command *(continued)*

Option	Description
`x` [*filename*…]	Extract. Selectively restores the files you specify by the *filename* argument. *filename* can be a list of files and directories. All files under a specified directory are restored unless you also use the `h` option. If you omit *filename* 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*…]	Table of contents. Checks the files specified in the *filename* 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* 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*, 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.

Additional `ufsrestore` options are described in the table below.

TABLE 45–4 Additional Options for the `ufsrestore` Command

Option	Description
`a` *archive-file* [*filename*…]	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 are prompted to mount the appropriate volume before extracting the file(s).
`b` *factor*	Blocking factor. Number of 512-byte blocks read from tape at a time. By default, `ufsrestore` tries to figure out the block size that was used in writing the tape.
`d`	Debug. Turn on debugging messages.

TABLE 45–4 Additional Options for the `ufsrestore` Command *(continued)*

Option	Description
`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 (first volume). 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 continue. This option tells the command to assume a yes response.

Commands for Interactive Restore

TABLE 45–5 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. 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 the latest revisions of 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. The files and directories are deleted only from the extract list you are building. They are not deleted from the media or the file system.
`extract`	Extracts the files in the list and restores them relative to the current working directory on the disk. Specify `1` when asked for a volume number for a single-volume backup. If you are doing a multitape or multidiskette restore and restoring a small number of files, start with the last tape or diskette instead.
`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.
`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.

TABLE 45–5 Commands for Interactive Restore *(continued)*

Option	Description
`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 displays information on each file as it is extracted.
`what`	Displays the backup header from the tape or diskette.

CHAPTER 46

Copying UFS Files and File Systems (Tasks)

This chapter describes how to copy UFS files and file systems to disk, tape, and diskettes using various backup commands.

Here is a list of the step-by-step instructions in this chapter:

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 45. 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`.

The table below describes when to use the various backup commands.

TABLE 46–1 When to Use Various Backup Commands

If You Want To ...	Then Use ...	Reference
Back up file systems to tape	`ufsdump`(1M)	"How to Do Backups to Tape" on page 574
Restore file systems from tape	`ufsrestore`(1M)	"How to Restore a Complete File System" on page 592
Transport files to other systems	`pax`(1), `tar`(1), or `cpio`(1)	"Copying Files and File Systems to Tape" on page 621
Copy files or file systems between disks	`dd`(1M)	"How to Clone a Disk (`dd`)" on page 616
Copy files to diskette	`tar`(1)	"How to Copy Files to a Single Formatted Diskette (`tar`)" on page 634

The table below describe various backup and restore commands.

TABLE 46–2 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`	No	Yes	Logical
`pax`	Yes	Yes	Logical
`dd`	Yes	No	Physical
`ufsdump/ufsrestore`	Yes	Yes	Logical

The following sections describe the advantages and disadvantages of each method and provide examples of how to use the commands.

Copying File Systems Between Disks

Two commands are used to copy file systems between disks:

- `volcopy`
- `dd`

The next section describes how to use the `dd` command to copy file systems between disks.

Making a Literal File System Copy

The `dd` command makes a literal (block-level) 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 without first specifying an appropriate block size.

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 default block size is 512 bytes.

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
```

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 superuser.**

3. **Create the /reconfigure file on the system so the system will recognize the clone disk to be added when it reboots.**

   ```
   # touch /reconfigure
   ```

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/rdsk/device-name of=/dev/rdsk/device-name bs=blocksize
```

`if=/dev/rdsk/`*device-name*	Represents the overlap slice of the master disk device, usually slice 2.
`of=/dev/rdsk/`*device-name*	Represents the overlap slice of the clone disk device, usually slice 2.
`bs=`*blocksize*	Block size, such as 128 Kbytes or 256 Kbytes. A large block size value decreases the time it takes to copy.

8. **Check the new file system.**

```
# fsck /dev/rdsk/device-name
```

9. **Mount the clone disk's root (/) file system.**

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

10. **Edit the clone disk's** `/etc/vfstab` **to reference the correct device names.**
    For example, changing all instances of `c0t3d0` with `c0t1d0`.

11. **Unmount the clone disk's root (/) file system.**

```
# umount /mnt
```

12. **Shut down the system.**

```
# init 0
```

13. **Boot from the clone disk to single-user mode.**

```
# boot diskn -s
```

Note - The `installboot` command is not needed for the clone disk because the boot blocks are copied as part of the overlap slice.

14. Unconfigure the clone disk.

```
# sys-unconfig
```

The system is shut down after it is unconfigured.

15. Boot from the clone disk again and provide its system information, such as host name, time zone, and so forth.

```
# boot diskn
```

16. Log in as superuser to verify the system information after the system is booted.

```
hostname console login:
```

Example—Cloning a Disk (dd)

```
# init 0
ok boot
# dd if=/dev/rdsk/c0t0d0s2 of=/dev/rdsk/c0t2d0s2 bs=128k
# fsck /dev/rdsk/c0t2d0s2
# mount /dev/dsk/c0t2d0s2 /mnt
# cd /mnt/etc
# vi vfstab
(Modify entries for the new disk)
# cd /
# umount /mnt
# init 0
# boot disk2 -s
# sys-unconfig
# boot disk2
```

Copying Directories Between File Systems (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 copies a list of files 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.

With cpio, you frequently use commands like ls and find to list and select the files you want to copy, piping the output to the cpio command.

▼ How to Copy Directories Between File Systems (cpio)

1. **Become superuser.**

2. **Change to the appropriate directory.**

```
# cd filesystem1
```

3. **Copy the directory tree from *filesystem1* to *filesystem2* by using a combination of the find and cpio commands.**

```
# find . -print -depth | cpio -pdm filesystem2
```

.	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.

You might also specify the –u option. This option forces an unconditional copy. Otherwise older files do not replace newer files. This might be useful if you want an exact copy of a directory, and some of the files being copied might already exist in the target directory.

4. Verify the copy was successful by displaying the destination directory contents.

```
# cd filesystem2
# ls
```

5. If appropriate, remove the source directory.

```
# rm -rf filesystem1
```

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. Because all three commands use the raw device, you do not need to format or make a file system on tapes before you use them.

TABLE 46–3 Advantages and Disadvantages of `cpio`, `pax`, and `tar` Commands

Command	Function	Advantages	Disadvantages
`pax`	Copy files, special files, or file systems that require multiple tape volumes or when you want to copy files to and from POSIX-compliant systems	■ Better portability than the `tar` or `cpio` commands for POSIX-compliant systems ■ Multi-vendor support	See disadvantages for `tar` command, except that `pax` can create multi-tape volumes
`tar`	Copy files and directory subtrees to a single tape	■ Available on most UNIX operating systems ■ Public domain versions are readily available	■ Is not aware of file system boundaries ■ Full pathname length cannot exceed 255 characters ■ Does not copy empty directories or special files such as device files ■ Cannot be used to create multi-tape volumes
`cpio`	Copy files, special files, or file systems that require multiple tape volumes or when you want to copy files from SunOS 5.8 systems to SunOS 4.0/4.1 systems	■ Packs data onto tape more efficiently than `tar` ■ Skips over any bad spots in a tape when restoring. ■ Provides options for writing files with different header formats (`tar`, `ustar`, `crc`, `odc`, `bar`) for portability between different system types ■ Creates multi-tape volumes	

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 639 for more information about tape drives and device names.

Copying Files to Tape (`tar` Command)

Things you should know before copying files to tape with the `tar` command:

- Copying files to a tape using the –`c` option to `tar` destroys any files already on the tape at or beyond the current tape position.
- You can use filename substitution wildcards (`?` and `*`) as part of the file names you specify when copying files. For example, to copy all documents with a `.doc` suffix, type `*.doc` as the filename argument.
- You cannot use filename substitution wildcards for extracting files from a `tar` archive.

▼ 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 ...
```

`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 and directories 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.**

5. **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 624 for more information on listing files on a tar tape.**

```
$ tar tvf /dev/rmt/n
```

Example—Copying Files to a Tape (`tar`)

The following example copies three files to the 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 59 tape blocks
a reports/reportB 61 tape blocks
a reports/reportC 63 tape blocks
$ tar tvf /dev/rmt/n
```

▼ 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
```

`t`	Lists the table of contents for the files on the tape.
`v`	Used with the `t` option, and provides detailed information about the files on the tape.
`f /dev/rmt/`*n*	Indicates the tape device.
filename ...	Indicates the files and directories you want to retrieve.

Example—Listing the Files on a Tape (`tar`)

The following example lists the files on the tape in drive 0.

```
$ tar tvf /dev/rmt/0
drwx--x--x   0/1        0 Jul 14 09:24 1999 reports/
-rw------t   0/1    30000 Jul 14 09:23 1999 reports/reportA
-rw------t   0/1    31000 Jul 14 09:24 1999 reports/reportB
-rw------t   0/1    32000 Jul 14 09:24 1999 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 ...]
```

`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.
filename	Specifies a file to retrieve.

4. **Verify the files are copied by listing the contents of the current directory.**

```
$ ls -l
```

Example—Retrieving the Files on a Tape (tar)

The following example retrieves all the files from the tape in drive 0.

```
$ cd /var/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
$ ls -l
```

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 624 for instructions.

See `tar`(1) for more information.

Copying Files to a Tape With `pax`

This section describes how to copy files with the `pax` command.

▼ 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 filename ...
```

`-w`	Enables the write mode.
`-f /dev/rmt/0`	Identifies the tape drive.
filename ...	Indicates the files and directories you want to copy.

4. **Verify the files are copied to tape.**

```
$ pax -f /dev/rmt/0
```

5. **Remove the tape from the drive and write the names of the files on the tape label.**

Example—Copying Files to a Tape (pax)

```
$ pax -w -f /dev/rmt/0 .
$ pax -f /dev/rmt/0
filea fileb filec
```

See pax(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
```

ls	Provides the cpio command with a list of file names.
cpio -oc	Specifies that cpio should operate in copy-out mode (–o) and write header information in ASCII character format (–c). This ensures portability to other vendor's systems.
> /dev/rmt/*n*	Specifies the output file.

   All 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 shown.

3. **Verify the files are copied to tape by using the following** cpio **command.**

```
$ cpio -civt < /dev/rmt/0
```

4. **Remove the tape from the drive and write the names of the files on the tape label.**

Example—Copying All Files in a Directory to a Tape (cpio)

The following example copies all of the files in the directory /export/home/kryten to the tape in tape drive 0.

```
$ cd /export/home/kryten
$ ls | cpio -oc > /dev/rmt/0
92 blocks
$ cpio -civt < /dev/rmt/0
-rw------t    1 kryten     users        400 Jul 14 09:28 1999, b
drwx--x--x    2 kryten     users          0 Jul 14 09:26 1999, letters
-rw------t    1 kryten     users      10000 Jul 14 09:26 1999, letter1
-rw------t    1 kryten     users      10100 Jul 14 09:26 1999, letter2
-rw------t    1 kryten     users      11100 Jul 14 09:27 1999, letter3
-rw------t    1 kryten     users      12300 Jul 14 09:27 1999, letter4
drwx--x--x    2 kryten     users          0 Jul 14 09:27 1999, memos
-rw------t    1 kryten     users        400 Jul 14 09:28 1999, memosmemoU
-rw------t    1 kryten     users        500 Jul 14 09:28 1999, memosmemoW
-rw------t    1 kryten     users        100 Jul 14 09:27 1999, memosmemoX
-rw------t    1 kryten     users        200 Jul 14 09:28 1999, memosmemoY
-rw------t    1 kryten     users        150 Jul 14 09:28 1999, memosmemoZ
drwx--x--x    2 kryten     users          0 Jul 14 09:24 1999, reports
92 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 an archive tape into the tape drive.**

2. **List the files on the tape using the `cpio` command.**

```
$ cpio -civt < /dev/rmt/n
```

`–c`	Specifies that `cpio` should read files in ASCII character format.
`–i`	Specifies that `cpio` should operate in copy-in mode (even though it's only listing files at this point).
`–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 input file of an existing `cpio` archive.

Example—Listing the Files on a Tape (cpio)

The following example lists the files on the tape in drive 0.

```
$ cpio -civt < /dev/rmt/0
drwx--x--x    2 kryten    users        0 Jul 14 09:34 1999, answers
-rw------t    1 kryten    users      800 Jul 14 09:36 1999, b
drwx--x--x    2 kryten    users        0 Jul 14 09:32 1999, sc.directives
-rw------t    1 kryten    users   200000 Jul 14 09:35 1999, direct241
drwx--x--x    2 kryten    users        0 Jul 14 09:32 1999, tests
-rw------t    1 kryten    users      800 Jul 14 09:36 1999, test13times
396 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 recreate the file on your system.

Caution - Using absolute path names can be dangerous because you might overwrite existing files on your system.

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 -icvd < /dev/rmt/n
```

–i	Reads in the contents of the tape.
–c	Specifies that cpio should read files in ASCII character format.
–v	Displays the files being retrieved in a format similar to the output from the ls command.
–d	Create directories as needed.
< /dev/rmt/*n*	Specifies the output file.

4. **Verify the files are copied by listing the contents of the current directory.**

```
$ ls -l
```

Example—Retrieving All Files From a Tape (cpio)

The following example retrieves all files from the tape in drive 0.

```
$ cd /var/tmp
cpio -icvd < /dev/rmt/0
answers
sc.directives
tests
8 blocks
$ ls -l
```

▼ 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
```

`–i`	Reads in the contents of the tape.
`–c`	Specifies that `cpio` should read headers in ASCII character format.
`–v`	Displays the files as they are retrieved in a format similar to the output from the `ls` 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 input file.

4. **Verify the files are copied by listing the contents of the current directory.**

```
$ ls -l
```

Example—Retrieving Specified Files From a Tape (cpio)

The following example retrieves all files with the suffix `chapter` from the tape in drive 0.

```
$ cd /home/smith/Book
$ cpio -icv "*chapter" < /dev/rmt/0
Boot.chapter
Directory.chapter
Install.chapter
Intro.chapter
31 blocks
$ ls -l
```

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:**
   - The local hostname (and optionally the username of the user doing the copy) must appear in the remote system's `/etc/hosts.equiv` file, or the user doing the copy must have his or her home directory accessible on the remote machine, and have the local machine name in `$HOME/.rhosts`. See `hosts.equiv`(4) for more information.
   - An entry for the remote system must be in the local system's `/etc/inet/hosts` file or in the name service `hosts` file.

2. **To test whether or not you have the appropriate permission to execute a remote command, try the following:**

```
$ rsh remotehost echo test
```

   If "test" is echoed back to you, you have permission to execute remote commands. If "Permission denied" is echoed, check your setup as described in step 1 above.

3. **To copy files to a remote tape drive, use the `tar` and `dd` commands.**

```
$ tar cf - files | rsh remotehost dd of=/dev/rmt/n obs=blocksize
```

`tar cf`	Creates a tape archive and specifies the tape device.
`-` (Hyphen)	Represents a place holder for the tape device.
files	Identifies files to be copied.
`|` *rsh remotehost*	Pipes the `tar` command's output to a remote shell to copy the files.
`dd of=`*/dev/rmt/n*	Represents the output device.
`obs=`*blocksize*	Represents the blocking factor.

4. **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 obs=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
6+9 records in
0+1 records out
```

▼ How to Extract Files From a Remote Tape Drive

1. **Change to a temporary directory.**

```
$ cd /var/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 -
```

`rsh` *remotehost*	Indicates a remote shell that is started to extract the files from the tape device using the `dd` command.
`dd if=/dev/rmt/`*n*	Indicates the input device.
`| tar xvBpf -`	Pipes the output of the `dd` command to the `tar` command used to restored the files.

3. Verify that the files have been extracted.

```
$ ls -l /var/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
$ ls -l /var/tmp
```

Copying Files and File Systems to Diskette

Before you can copy files or file systems to diskette, you must format the diskette. See Chapter 16 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

- Copying files to a formatted diskette using the `–c` option of `tar` destroys any files already on the diskette.
- A diskette that already contains a `tar` image is not mountable.

▼ 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.**

```
$ volcheck
```

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 ...
```

The file names you specify are copied to the diskette, overwriting any existing files on the diskette.

6. **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`)” on page 635 for more information on listing files.**

```
$ tar tvf /vol/dev/rdiskette0/unlabeled
```

7. **Remove the diskette from the drive.**

8. **Write the names of the files on the diskette label.**

Example—Copying Files to a Single Formatted Diskette (tar)

The following example copies two files 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
$ tar tvf /vol/dev/rdiskette0/unlabeled
```

▼ How to List the Files on a Diskette (tar)

1. **Insert a diskette into the drive.**

2. **Run** volcheck **to make the diskette available.**

```
$ volcheck
```

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)

The following example lists the files on a diskette.

```
$ tar tvf /vol/dev/rdiskette0/unlabeled
rw-rw-rw-6693/10  44032 Jun  9 15:45 evaluation.doc
rw-rw-rw-6693/10  43008 Jun  9 15:55 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.**

```
$ volcheck
```

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. **Verify the files have been retrieved by listing the contents of the current directory.**

```
$ ls -l
```

6. **Remove the diskette from the drive.**

Examples—Retrieving Files From a Diskette (tar)

The following example retrieves all the files from a diskette.

```
$  /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
$ ls -l
```

The following example retrieves an individual file from a diskette.

```
$ tar xvf /vol/dev/rdiskette0/unlabeled evalutation.doc
x evaluation.doc, 44032 bytes, 86 tape blocks
$ ls -l
```

The file names you specify are extracted from the diskette and placed in the current working 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 627 for information on how to use `cpio`.

Copying Files With a Different Header Format

Archives created with the SunOS 5.8 `cpio` command might 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` values 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 4.0/4.1 `bar` command, use the `–H bar` option to `cpio`.

Note - You can use only 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. **Insert the diskette into the drive.**

3. **Run `volcheck` to make the diskette available.**

```
$ volcheck
```

4. **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
```

CHAPTER 47

Managing Tape Drives (Tasks)

This chapter describes how to manage tape drives.

Here is a list of the step-by-step instructions in this chapter:

Choosing Which Media to Use

You typically back up Solaris 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.

The table below shows typical media used for backing up file systems and shows the storage capacity for each. Capacity depends on the type of drive and the data being written to the tape.

TABLE 47–1 Media Storage Capacities

Media	Capacity
1/2-inch reel tape	140 Mbytes (6250 bpi)
2.5-Gbyte 1/4 inch cartridge (QIC) tape	2.5 Gbytes
DDS3 4-mm cartridge tape (DAT)	12 - 24 Gbytes
14-Gbyte 8-mm cartridge tape	14 Gbytes
DLT™ 7000 1/2-inch cartridge tape	35 - 70 Gbytes

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. The table below shows this naming scheme.

TABLE 47–2 Basic Device Names for Backup Devices

Device Type	Name
Tape	`/dev/rmt/`*n*
Diskette	`/vol/dev/rdiskette0/unlabeled`

In general, you specify a tape drive device as shown in the figure below.

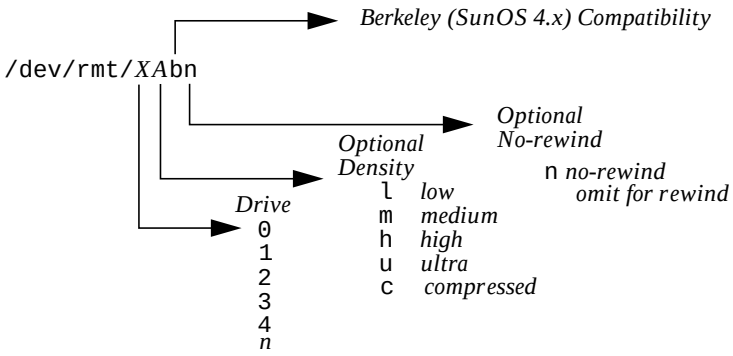


Figure 47–1 Tape Drive Device Names

If you don't specify the density, a tape drive typicallys write 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. To determine the different densities that are supported for a drive, look at the `/dev/rmt` subdirectory, which includes the set of tape device files that support different output densities for each tape.

Also, a SCSI controller can have a maximum of seven SCSI tape drives.

Specifying the Default Density for a Tape Drive

Normally, you specify a tape drive by its logical unit number, which can run from 0 to *n*. The table below describes how to specify tape device names using default density settings.

TABLE 47–3 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. The table below describes how to specify different densities for a tape drive.

TABLE 47–4 Specifying Different Densities for a Tape Drive

To Specify The ...	Use ...
First drive, low density, rewinding	`/dev/rmt/0l`
First drive, low density, nonrewinding	`/dev/rmt/0ln`
Second drive, medium density, rewinding	`/dev/rmt/1m`
Second drive, nonrewinding, medium density	`/dev/rmt/1mn`

The unit and density characters are shown in Figure 47–1.

Displaying Tape Drive Status

You can use the `status` option with the `mt` command to get status information about tape drives. The `mt` command reports information about any tape drives described in the `/kernel/drv/st.conf` file.

▼ How to Display Tape Drive Status

1. **Load a tape into the drive you want information about.**

2. **Display tape drive status 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— Displaying Tape Drive Status

The following 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
```

The following example shows a quick way to poll a system and locate all of its tape drives.

```
$ 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
$
```

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

The following example retensions the tape in drive `/dev/rmt/1`.

```
$ 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

The following example rewinds the tape in drive `/dev/rmt/1`.

```
$ 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 42, 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

B

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

Y