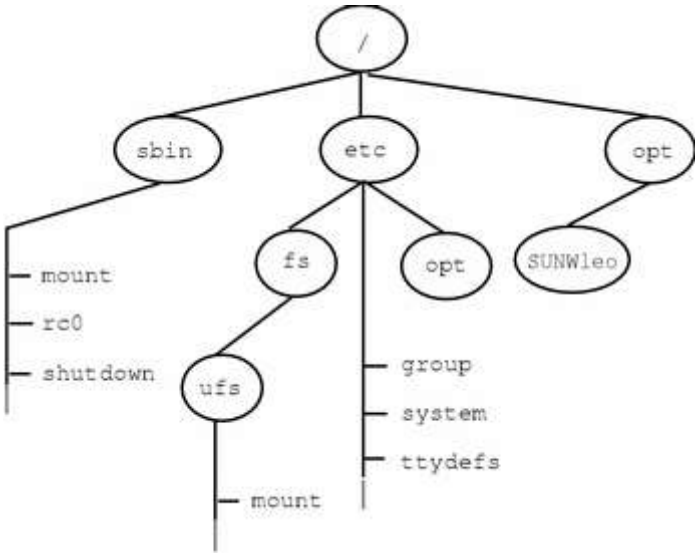
Mounting and Unmounting File Systems

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`.

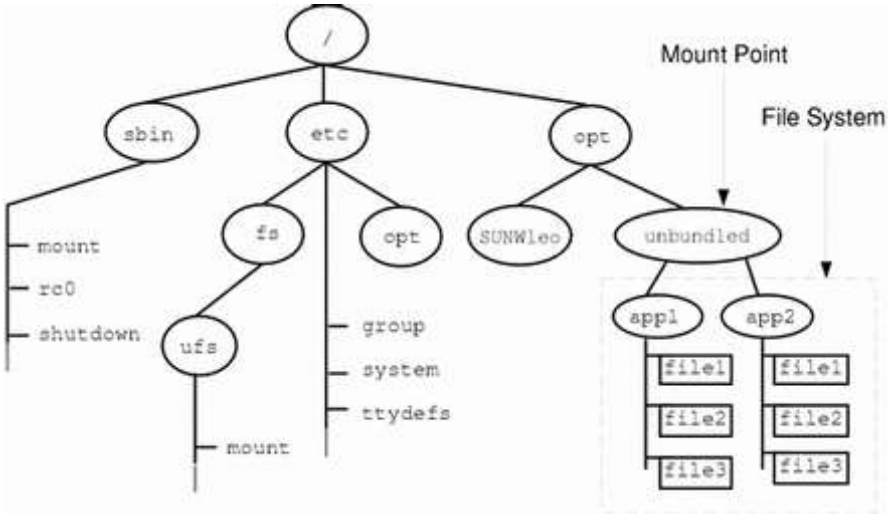
Figure 34-1 Sample root (/) File System



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, Mounting and Unmounting File Systems (Tasks) 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     -
$
```

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, Mounting and Unmounting File Systems (Tasks) 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, Mounting and Unmounting File Systems (Tasks) 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, The Cache File System (Tasks) 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.

If You Need to Mount ...	Then You Should Use ...
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, Guidelines for Using CDs and Diskettes (Overview) for more information.

- *Previous*: The UFS File System
- *Next*: Determining a File System's Type