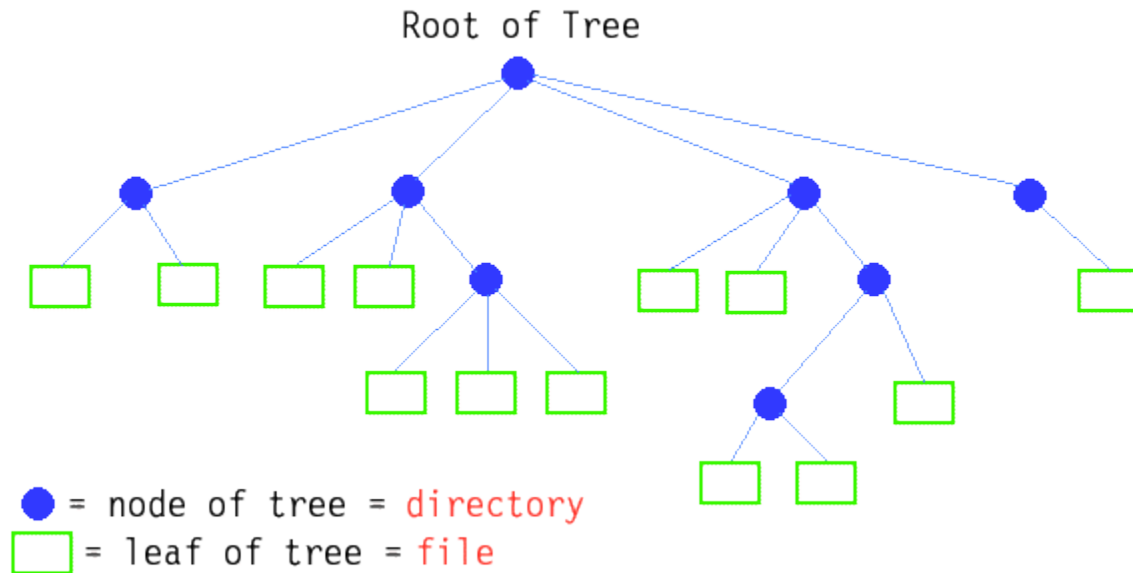


The File System

- ◆ The UNIX file system is organized as a **hierarchical tree**. (Usually drawn as an upside-down tree - you need to be standing on your head!)



- ◆ A **directory** is a listing of files and other directories. (On Windows and Mac OS, it is referred to as a *folder*.)
- ◆ Directories help keep the masses of files organized in a meaningful way.
- ◆ The **root** of the tree is a directory known as / (and is called **root or the root directory**).
- ◆ A **file** contains information or is a **special file**.

Files

- ◆ All I/O in UNIX is done through files.

Everything on a UNIX system is either a file, a process (a running program) or a user (logged in to the system).

Even then, you can think of a logged in user as a collection of one or more processes, and several open files.

- ◆ Ordinary files are used to store data, code or text.
- ◆ A program can get inputs from, and send outputs to an ordinary file.
- ◆ A special file provides a means of getting at an I/O device (line printer, USART, console, floppy disk, CD-ROM, etc.)
- ◆ A program can get inputs from, and send outputs to a special file.
- ◆ The generalized I/O scheme of UNIX is very powerful.
- ◆ An application program can put its outputs into an ordinary file.
- ◆ Later it can easily be modified to put its outputs onto a console, or to ship them out a serial line to a remote site.
- ◆ To see the contents of a directory, use the *list* command:

```
ls /  
bin  
dev  
etc  
lib  
lost+found  
mnt  
sys  
tmp  
usr  
var  
vmunix
```

What is in the Standard Directories

<i>bin</i>	The binary images of most commands are kept under this directory.
<i>dev</i>	A special file representing each I/O device exists under this directory.
<i>etc</i>	System configuration files and startup scripts.
<i>lib</i>	Shared and static libraries.
<i>lost+found</i>	If the superuser finds a strange file hanging around, he or she will probably put it here. Directory is initially empty. The file system check program (<i>fsck</i>) will also put strange files that it finds here. One of these directories exists on each disk partition or volume.
<i>mnt</i>	If you have a external storage device like a floppy disk, CD-ROM, DVD-ROM or USB flash drive that contains its own tree structure, you can “graft” the floppy or CD-ROM tree onto the main tree at node <i>/mnt</i> , or one of its subdirectories.
<i>sbin</i>	Superuser commands.
<i>tmp</i>	For temporary files; initially empty. This directory is shared by all users as a “scratch pad”.
<i>usr</i>	Additional commands, libraries and applications. On some systems, home directories for each valid user exist here.
<i>var</i>	System log files and some temporary files are stored here.
<i>vmunix</i>	Data file; this is the binary executable for the UNIX kernel, which is loaded when the system starts up. Modern systems store the kernel in different directories; not so much in the root directory any more.

Examples of Top-Level Directories

Solaris 9

```
$ uname -a
```

```
SunOS turing 5.9 Generic_122300-12 sun4u sparc SUNW, Sun-Blade-100
```

```
$ ls -F /
```

```
InstallShield/ devices/ kernel/ opt/  
TJ_D.B/ etc/ lib@ tmp/  
admin/ expert/ local/ platform/ usr/  
bin@ lost+found/ proc/ var/  
cdrom/ floppy/ mnt/ reoc/ vol/  
dev/ home/ net//sbin/ xfn/
```

```
$ ls -l /kernel/*unix
```

```
-rwxr-xr-x 1 root sys 1838292 Jul 30 18:16 /kernel/genunix
```

Mac OS X 10.4 (Tiger)

```
$ uname -a
```

```
Darwin hadblisueby.local 8.10.1 Darwin Kernel Version 8.10.1: Wed May 23 16:33:00 PDT 2007; root:xnu-792.22.5~/RELEASE/8.10.1/8.10.1
```

```
$ ls -F /
```

```
Applications/ dev/  
Desktop.D.B etc@  
Desktop.D.F incoming/  
Developer/ mach@  
Library/ mach.sym  
Network/ mach_kernel  
System/ mnt/  
User Guides And Information@ opt/  
Users/ private/  
Volumes/ sbin/  
autocount/ tmp@  
bin/ usr/  
cores/ var@
```

```
$ ls -l /*kernel
```

```
-rw-r--r-- 1 root wheel 8570180 Jun 29 20:58 /mach_kernel
```

Fedora Core 4 Linux

```
$ uname -a
```

```
Linux stallman 2.6.11-1.1369_FC4 #1 Thu Jun 22 22:55:56 EDT 2005 i686 i686 i386 GNU/Linux
```

```
$ ls -F /
```

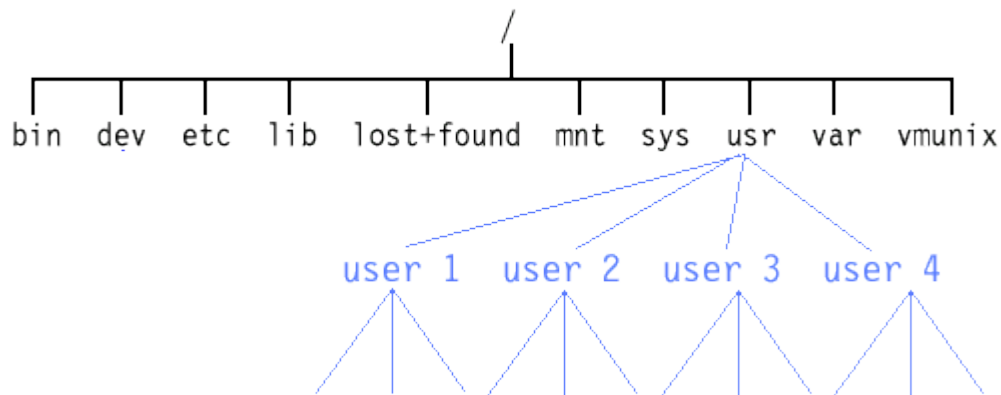
```
bin/ dev/ home/ lost+found/ misc/ net/ proc/ sbin/ srv/ tmp/ usr/  
boot/ etc/ lib/ media/ mnt/ opt/ reoc/ selinux/ sys/ var/
```

```
$ ls -l /boot/vm*
```

```
-rw-r--r-- 1 root root 1639539 Jun 22 2005 /boot/vmlinuz-2.6.11-1.1369_FC4
```


The Home Directory

- ◆ Each user is assigned a **home directory** when made a valid user on the system.
- ◆ The command `ls` returns a listing of all of the entries in your home directory.
- ◆ When you log in, you are automatically placed in your home directory.



- ◆ You can place yourself on different nodes for convenience.

```
cd proja or cd /usr/user1/proja
ls
code
data
```

Print Working Directory

- ◆ If you forget your current directory:

```
pwd
/usr/user1/proja
```

- ◆ To return to your home directory:

```
cd
pwd
/usr/user1
```

- ◆ To find out more about the *ls* command, look in the reference manual.

man ls

NAME

ls - list contents of directory

SYNTAX

ls [-lstdurcifg] name...

DESCRIPTION

For each directory argument, ls lists the contents of the directory; for each file argument ls repeats its name and any other information requested.

...

- ◆ To display a file on the console, terminal or window that is longer than 24 lines:

more longfile

- ◆ *More* is a **filter** that displays 24 lines of a file and then stops each time you hit the space bar.
- ◆ Some systems have *less* ("*less is more*") installed -- this is free software from the GNU Project and Free Software Foundation.
- ◆ Another (outdated, since it is difficult to do with today's faster hardware) way is to use:

Ctrl-S - Freezes output which is scrolling by on the screen

Ctrl-Q - Resumes output

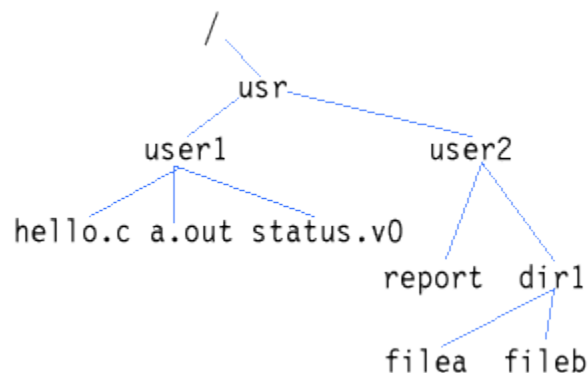
- ◆ If you make an error in typing a command, hit the **Backspace** key.
- ◆ If you want to start over, press **Ctrl-C** to get a fresh prompt.

Directories

- ◆ Each directory contains two special filenames, "." and ".."
- ◆ These files are normally not listed. In order to see them, use `ls -a`
- ◆ File "." is a shorthand name for the directory itself; file ".." is a shorthand name for the parent directory.
- ◆ Also, file "~" is a shorthand name for your home directory.

Common Use of ".", "..", and "~"

- ◆ To reference a file called *file1* in the current directory: `./file1`
- ◆ To reference a file called *file2* residing in the parent of the current directory: `../file2`
- ◆ To get a listing of the parent directory: `ls ..`
- ◆ To reference a file called *file3* residing in your home directory: `~/file3`
- ◆ To reference a file called *file4* residing in *user2*'s home directory: `~user2/file4`



- ◆ In the above tree, if you are *user 2*, then your home directory is `/usr/user2`. If that is your current working directory, and you enter `ls ../user1` the following output will be displayed:

```
ls ../user1
a.out
hello.c
status.v0
```

Three Ways to Share Files

1. Get a copy of the file, and put it under one of your directories. Now there is a two copies of that file on the disk. For example,

```
cp ../user2/status.v0 .
ls -l ../user2/status.v0 status.v0
-rw-r--r-- 1 user2 users 230 Dec 1 12:48 status.v0
-rw-r--r-- 1 user1 users 230 Dec 1 12:48 status.v0
```

2. Build a **(hard) link** from one of your directories to the file you wish to share. Now there is only one copy of the file, but it is listed in two directories:

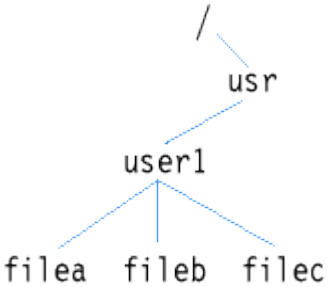
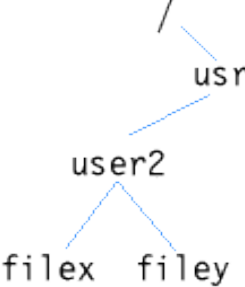
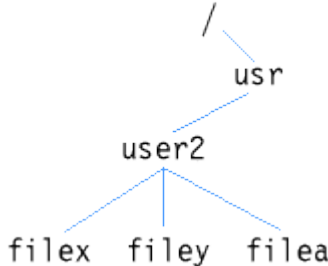
```
ln ../user2/status.v0 .
ls -l ../user2/status.v0 status.v0
-rw-r--r-- 2 user2 users 230 Dec 1 12:48 status.v0
-rw-r--r-- 2 user2 users 230 Dec 1 12:48 status.v0
```

3. Build a **symbolic (soft) link** from one of your directories to the file you wish to share. The file you link to will remain unchanged, while new file is created in your directory containing **the path** to the linked file:

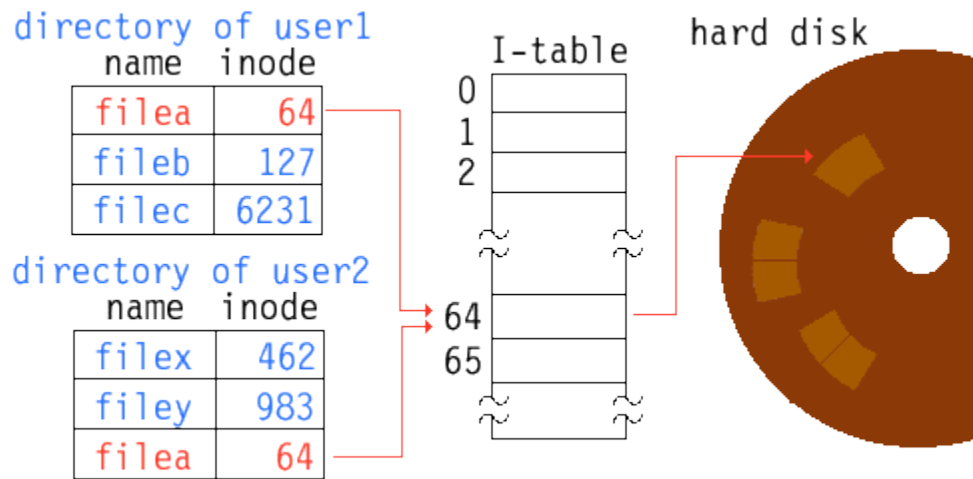
```
ln -s ../user2/status.v0 .
ls -l ../user2/status.v0 status.v0
-rw-r--r-- 1 user2 users 230 Dec 1 12:48 status.v0
lrwxrwxrwx 1 user1 users 18 Dec 1 12:48 status.v0->../user2/status.v0
```

- ◆ A directory entry consists of a name and an **inode number**.
- ◆ Each file and directory on the disk is identified by a unique **inode number**.
- ◆ To see the inode number, enter the command `ls -li`

Linking

<p>Original owner's file tree:</p>  <pre>graph TD; root[" / "] --> usr[" usr "]; usr --> user1[" user1 "]; user1 --> filea[" filea "]; user1 --> fileb[" fileb "]; user1 --> filec[" filec "];</pre>	<p>Your file tree:</p>  <pre>graph TD; root[" / "] --> usr[" usr "]; usr --> user2[" user2 "]; user2 --> filex[" filex "]; user2 --> filey[" filey "];</pre>
<p>After executing the link command:</p> <p><code>ln /usr/user1/filea .</code></p> <p>The original owner's file tree is unchanged.</p>	<p>Your file tree:</p>  <pre>graph TD; root[" / "] --> usr[" usr "]; usr --> user2[" user2 "]; user2 --> filex[" filex "]; user2 --> filey[" filey "]; user2 --> filea[" filea "];</pre> <p>There is only 1 copy of <i>filea</i> on the disk; you now have a link to it.</p>

Directories After Linking



Facts About Links

- ◆ A file cannot be removed from disk until the number of (hard) links is zero (the `rm` command subtracts 1 from the link count.)
- ◆ You cannot hard link to a directory; you can only create a symbolic (soft) link.
- ◆ If you change a file that you are linked (hard or symbolic) to, it is changed for everyone else that is linked to that file.
- ◆ You can only (hard) link to files on the **same filesystem**. For example, you cannot link to a file that is on a mountable floppy, another hard disk or hard disk partition, or CD-ROM; you can only create a symbolic link.
- ◆ If you link to someone else's file, that user is still considered to be the owner of the file.

Wildcards

- ◆ The following special constructs will allow more than one to be described with one pathname:

*	Matches any string of zero or more characters.
?	Matches any single character.
<i>[string]</i>	Matches any one character in <i>string</i> .

Wildcard Examples

Pathname	Matches
TOPDIR/*	TOPDIR/DIR1, TOPDIR/FILE1, TOPDIR/PROG1, TOPDIR/DIR2
TOPDIR/*2	TOPDIR/DIR2
TOPDIR/[DF]*	TOPDIR/DIR1, TOPDIR/FILE1, TOPDIR/DIR2
TOPDIR/DIR2/FILE?	TOPDIR/DIR2/FILE1, TOPDIR/DIR2/FILE2

Common File Commands

Note: Commands are lower case; each command may have options.

Command	Operation
cp	Copy files
mv	Rename (move) files
rm	Remove files
cat	<i>"Concatenate"</i> files - often used to list a file on the terminal
mkdir	Make a new directory
rmdir	Remove directory
ls	List directory contents

Data File Operations

<code>cp txt1 txt2</code>	Copy file <i>txt1</i> to file <i>txt2</i>
<code>cp /usr/class/fx .</code>	Copy file <i>fx</i> in directory <i>/usr/class</i> to the current directory; name of the new file will also be <i>fx</i>
<code>mv oldname newname</code>	Rename file <i>oldname</i> to <i>newname</i>
<code>mv myfile /usr/class</code>	Rename file <i>myfile</i> to a new directory. The file is now found in <i>/usr/class</i>
<code>rm olddata</code>	Delete the file <i>olddata</i>
<code>rm -i *.c</code>	Delete all files whose names end in ".c"; ask for confirmation before deleting each file

Looking At Files

<i>cat TXT1</i>	Print the contents of file <i>TXT1</i> on the screen
<i>cat *.c</i>	Print the contents of all files whose names end in ".c".
<i>ls /usr/lib</i>	List names of files in directory <i>/usr/lib</i>
<i>ls -l</i>	List names and access information for all files in the current working directory.

CTEC1430/2009W Enterprise Computing I - UNIX File System
This page intentionally left blank.