

The Filesystem

- ♦ Chapter 5, pages 63 – 72
- ♦ Represent and organize the systems storage resources
- ♦ Many other objects mapped onto the filesystem
 - ♦ Processes
 - ♦ Non-storage hardware devices (ex: serial ports)
 - ♦ Interprocessor communication channels
- ♦ Complication: different portions of the filesystem handled by different kernel drivers

Four Filesystem Components

- ◆ Namespace – method of naming and organizing
- ◆ API – system calls for navigating and manipulating
- ◆ Security Model – scheme for protecting, hiding, and sharing objects
- ◆ Implementation – software tying together the logical model and the hardware

Disk Based Filesystems

- ◆ ext2
- ◆ Journaling filesystems
 - ◆ ext3 for Redhat (tune2fs to convert)
 - ◆ <http://www.redhat.com/support/wpapers/redhat/ext3/>
- ◆ Foreign filesystems support
 - ◆ Windows (FAT, FAT32, NTFS)
 - ◆ Apple (HFS)
 - ◆ CD-ROM (ISO-9660)
 - ◆ More than any other version of UNIX

Pathnames

- ◆ Pathname: list of directories that must be traversed to reach a file along with the filename
- ◆ Filesystem is a single unified hierarchy starting with / (root directory)
- ◆ Absolute (/usr/bin/foo)
- ◆ Relative (book/ch5/filesystem)

Path/Filename Restrictions

- ◆ Filesystem has no limitation on depth
- ◆ File/directory names may be no more than 255 characters in length
- ◆ Pathname limited to 4,095 characters
- ◆ Filenames cannot contain “/” or nulls
- ◆ Spaces in filenames are permitted
 - ◆ Must be quoted (less “My file.txt”)
 - ◆ Using spaces in filenames is DUMB

Filesystem Details

- ♦ A filesystem is composed of filesystems each consisting of one directory and its subdirectories
- ♦ Most filesystems are disk partitions but may be:
 - ♦ Network file servers, kernel components, memory-based disk emulators, etc.
 - ♦ Loopback (/dev/loop0) – mount a file as a filesystem
 - ♦ Install from CD ISOs without burning
 - ♦ Create an encrypted filesystem in a file
 - ♦ <http://www.anandtech.com/guides/viewfaq.html?i=137>

Mounting Filesystems

- ♦ Mount – attaches root of a filesystem to an existing directory within the filesystem (mount point)
 - ♦ Example: `mount /dev/hda2 /mnt/windows`
- ♦ `/etc/fstab` – info on filesystems customarily mounted on a particular system
 - ♦ Allows filesystem to be checked (fsck) and mounted
 - ♦ More in Ch 8, page 141

Unmounting Filesystems

- ◆ Unmount – removes mounted filesystem
 - ◆ Cannot unmount a filesystem that is busy
 - ◆ Example: `umount /mnt/windows` or
`umount /dev/hda2`

fuser

- ♦ fuser – identify processes using file or filesystem
 - ♦ fuser -mv mount point
 - ♦ fuser -v filename
 - ♦ -k flag kills all processes using the resource
- ♦ Access codes
 - ♦ f – process has a file open for reading or writing
 - ♦ c – process's current directory is on the filesystem
 - ♦ e – process is currently executing a file
 - ♦ r – process's root directory is on the filesystem
 - ♦ m – process has mapped a file or shared library

fuser Output

```
% fuser -mv /usr
```

	USER	PID	ACCESS	COMMAND
	root	444	...m	atd
/usr	root	499	...m	sshd
	root	520	...m	lpd
	...			

Filesystem Organization

- ◆ /etc – critical startup and configuration files
- ◆ /boot – kernel (vmlinuz)
- ◆ /dev – device files
- ◆ /bin, /sbin – essential user and system commands
- ◆ /usr – secondary files and commands
- ◆ /var – spools, logs, accounting info
- ◆ See page 68 for full list
- ◆ Filesystem Hierarchy Standard
<http://www.pathname.com/fhs/>

File Types

- ◆ Determine file type with ls
 - ◆ % ls -ld filename
 - ◆ drwxr-x 27 root root 4096 Jul 15 20:57 ...
- ◆ File type codes

Symbol	File Type	Created By	Removed By
-	regular file	editors, cp. etc.	rm
d	directory	mkdir	rmdir, rm -r
c	character device file	mknod	rm
b	block device file	mknod	rm
s	unix domain socket	socket(2)	rm
p	named pipe	mknod	rm
l	symbolic link	ln -s	rm

rm

- ◆ `rm` removes any type of file
- ◆ Special cases
 - ◆ A file named `-f`
 - ◆ Use more complete path: `rm ./-f`
 - ◆ Use `--` option: `rm -- -f`
 - ◆ Filenames with control characters (`foo<Ctrl-D>bar`)
 - ◆ Use pattern matching with `-i` to confirm delete
 - ◆ `rm -i foo*`
 - ◆ `rm -i *`

File Type Details

- ◆ Regular - “bag o'bytes”, Linux imposes no structure on contents
- ◆ Directories – contain named references to other files
 - ◆ “.” and “..”
 - ◆ A file's name is stored with its parent directory, not the file itself
- ◆ Hard links
 - ◆ Indistinguishable from the original file
 - ◆ Cannot delete original until all hard links deleted
 - ◆ Created by `ln oldfile newfile`

Character/Block Devices

- ◆ Device files (/dev) used to communicate with device drivers
- ◆ Character devices – device handles I/O buffering
- ◆ Block devices – I/O done in large chunks, kernel handles buffering
- ◆ Major device number – tells kernel which driver the file refers to
- ◆ Minor device number – tells driver which physical unit to address
 - ◆ Ex: /dev/lp0 – major device # 6, minor device #0

Character/Block Devices Cont.

- ♦ Minor device number can be used in other ways
- ♦ Two device files can refer to the same physical device
 - ♦ `/dev/ttyS0`
 - ♦ `/dev/cua0`
- ♦ Creating device files
 - ♦ `mknod`
 - ♦ `/dev/MAKEDEV` creates most common device files

Local Domain Sockets

- ♦ Sockets – communications connections between processes
 - ♦ Network sockets – use network ports
 - ♦ Local Domain Sockets – use a filesystem object
 - ♦ Used for printing, X Windows, and syslog
- ♦ Local sockets may only be read/written by processes involved in the connection
- ♦ Create with socket
- ♦ Remove with rm or unlink

Named Pipes

- ◆ Unnamed pipes
 - ◆ `ls -l | less`
- ◆ Named pipes – allow communication between 2 processes
 - ◆ Also known as FIFO files
 - ◆ Ex: `ls -l > pipe1` in one virtual terminal
`cat < pipe1` in another virtual terminal
 - ◆ `ls -l` is blocked until `cat` begins reading from the pipe
 - ◆ <http://www.linuxjournal.com/article.php?sid=2156>
- ◆ Created with `mknod` or `mkfifo`

Symbolic Links

- ♦ Symbolic (“soft”) link – file which contains a pathname reference to another file
 - ♦ Absolute or relative pathname
 - ♦ File being referenced may be deleted
 - ♦ Can form loops
- ♦ Created with `ln -s pathname filename`