

CSE 265: System and Network Administration

- Disks
- Partitions
- Volumes
- Filesystems
- Files



Disk interfaces

- Relatively few
 - SCSI (pronounced “scuzzy”)
 - Common, widely supported
 - IDE a.k.a. ATA or PATA, and SATA
 - Inexpensive, simple
 - Fibre Channel
 - High bandwidth, lots of simultaneous devices
 - Supports 100MB/s and faster
 - Universal Serial Bus (USB)
 - Typically used for slow devices (e.g., CD-ROMs, portable, removable drives)

SCSI:

Small Computer Systems Interface

- Many versions

- SCSI-1 (1986) 8-bits, 5MB/s
- SCSI-2 (1990) added command queuing, DMA, more
- Fast SCSI-2 8-bits, 10MB/s
- Fast/wide SCSI-2 16-bits, 20MB/s
- Ultra SCSI 8 bits, 20MB/s
- Wide Ultra SCSI 16bits, 40MB/s
- Wide Ultra2 SCSI 16bits, 80MB/s
- Wide Ultra3 SCSI 16bits, 160MB/s
- Ultra-320, Ultra-640 SCSI

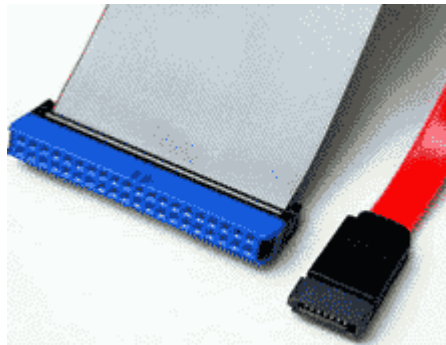


IDE a.k.a. ATA

- Integrated Drive Electronics / AT Attachment
 - Very short cable lengths (18in!)
- ATA-2 added DMA and LBA (get beyond BIOS 504MB limit)
- ATA-3 added power management, self-monitoring (16MB/s)
- Ultra-ATA added Ultra DMA/33, /66, and /133 modes (33-133MB/s)
- ATAPI interface allows non-ATA devices to connect
 - E.g., CD-ROMs

SATA

- Becoming standard equipment
 - Fast: 150-300MB/s (600MB/s now available)
 - Software compatible with parallel ATA
 - One drive per controller
 - Thin cables

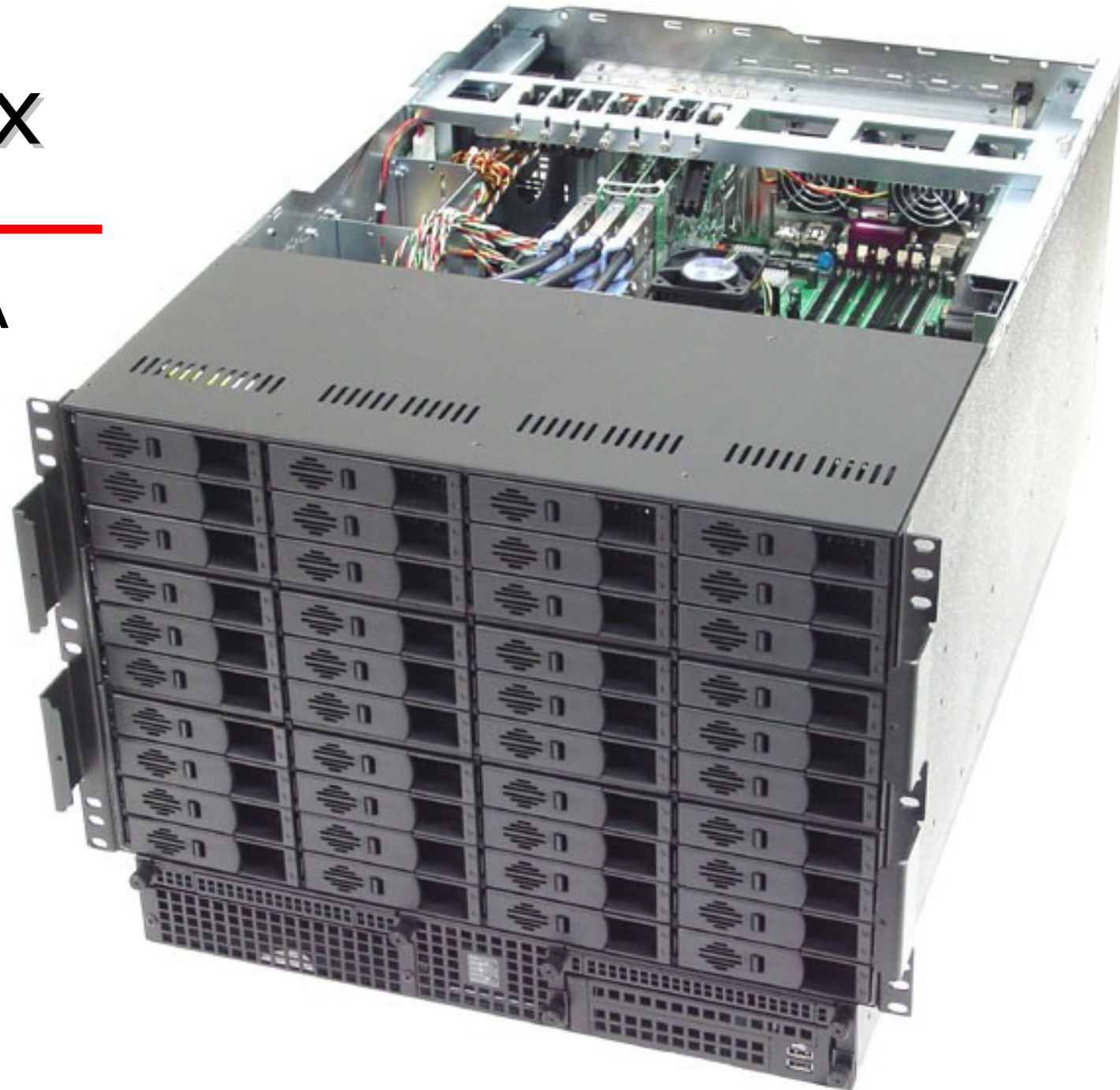


SCSI vs. IDE

- SCSI traditionally beats IDE technically, but may not be worth the price premium
- In single-user systems, IDE will provide 85%, cheaply
- For best possible performance, SCSI is better
 - e.g., in servers and multi-user systems
 - handles multiple simultaneous reqs + more devices better
 - higher-end equipment (faster, better warranty, etc.)
- SATA technology is quite good
 - Better than IDE, usually better price/performance than SCSI
- Still subject to much debate

Black box

- 40+2 SATA drives
- RAID
- Dual Xeon
- 8U tall
- Up to 80TB



Sun X4500

- 48 SATA drives
- Software RAID or ZFS
- Dual AMD
- 4U tall
- Up to 48TB



Adding a disk to Linux

STEP-BY-STEP (w/out LVM)

- Install new hardware
 - verify that hardware is recognized by BIOS or controller
- Boot, make certain device files already exist in /dev
- Use **fdisk/parted** (or similar) to partition the drive
 - Verify the system type on each partition
- Use **mke2fs** (-t ext4) on each regular partition
 - To create a filesystem
- Use **mkswap** to initialize swap partitions
- Add entries to /etc/fstab
- Mount by hand, then reboot to verify everything

Disk installation

- Connecting the disk
 - depends primarily on the interface used
- Formatting the disk
 - Need device files in /dev to access a drive
 - Formatted capacity is less than advertised
 - All drives come preformatted
 - Factory formatting often more precise
 - Avoid doing a (low-level) format!

hdparm: test/set hd params

- hdparm will do simple performance tests

```
[root@wume2 ~]# /sbin/hdparm -Tt /dev/hda
```

```
/dev/hda:
```

```
Timing cached reads: 1928 MB in 2.00 seconds = 963.26 MB/sec
```

```
Timing buffered disk reads: 122 MB in 3.03 seconds = 40.22 MB/sec
```

```
[root@wume1 ~]# /sbin/hdparm -Tt /dev/sda
```

```
/dev/sda:
```

```
Timing cached reads: 3440 MB in 2.00 seconds = 1720.77 MB/sec
```

```
Timing buffered disk reads: 162 MB in 3.03 seconds = 53.41 MB/sec
```

```
[root@night ~]# /sbin/hdparm -Tt /dev/sdd
```

```
/dev/sdd:
```

```
Timing cached reads: 10504 MB in 2.00 seconds = 5254.65 MB/sec
```

```
Timing buffered disk reads: 1196 MB in 3.00 seconds = 398.28 MB/sec
```

```
[root@morning ~]# /sbin/hdparm -Tt /dev/hda
```

```
/dev/hda:
```

```
Timing cached reads: 4092 MB in 2.00 seconds = 2047.82 MB/sec
```

```
Timing buffered disk reads: 10 MB in 3.03 seconds = 3.30 MB/sec
```

Disk partitions

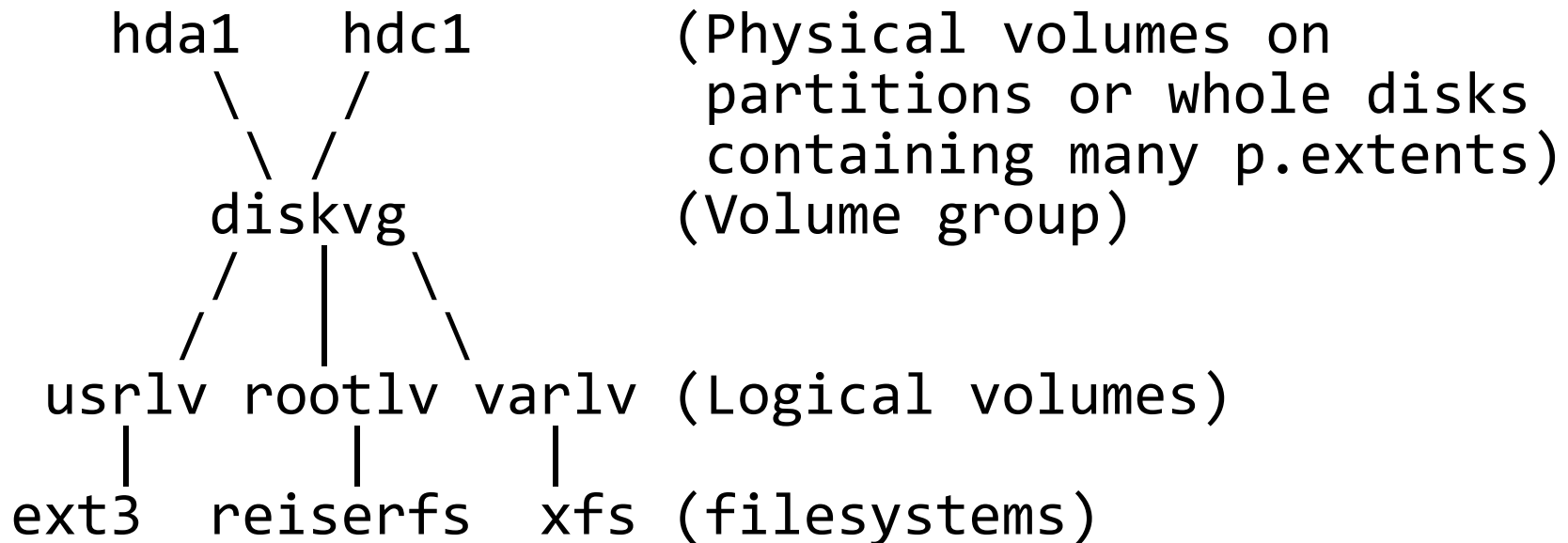
- Drive are be divided into one or more partitions that are treated independently
 - Partitions make backups easier, confine damage
- Typically have at least two or three
 - root partition (one)
 - everything needed to bring system up in single-user mode (often copied onto another disk for emergencies)
 - swap partition (at least one)
 - stores virtual memory when physical memory is insufficient
 - user partition(s)
 - home directories, data files, etc.
 - boot partition - boot loader, kernel, etc.

Logical Volumes

- Partitions are static, and sometimes you want to change them
- LVM (Linux Logical Volume Manager) lets you combine partitions and drives to present an aggregate volume as a regular block device (just like a disk or partition)
 - Use and allocate storage more efficiently
 - Move logical volumes among different physical devices
 - Grow and shrink logical volume sizes on the fly
 - Take “snapshots” of whole file systems
 - Replace on-line drives without interrupting service
- Similar systems are available for other OSes

LVM

- LVM2 was incorporated with the 2.6 kernel
- Sample organization:



Example configuration

[from fdisk]

Disk /dev/hda: 40.0 GB, 40020664320 bytes
16 heads, 63 sectors/track, 77545 cylinders
Units = cylinders of 1008 * 512 = 516096 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1	*	1	203	102280+	83	Linux
/dev/hda2		204	77545	38980368	8e	Linux LVM

Disk /dev/hdd: 15.3 GB, 15393079296 bytes
16 heads, 63 sectors/track, 29826 cylinders
Units = cylinders of 1008 * 512 = 516096 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/hdd1	*	1	29826	15032272+	83	Linux

Example configuration

```
[root@davison ~]# lvs
```

LV	VG	Attr	LSize	Origin	Snap%	Move	Log	Copy%	Convert
lv_home	vg_davison	-wi-ao	9.77g						
lv_root	vg_davison	-wi-ao	35.29g						
lv_swap	vg_davison	-wi-ao	5.44g						

```
[root@davison ~]# df -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/mapper/vg_davison-lv_root	35G	3.1G	30G	10%	/
tmpfs	1.8G	124K	1.8G	1%	/dev/shm
/dev/sda1	485M	33M	428M	7%	/boot
/dev/mapper/vg_davison-lv_home	9.7G	151M	9.0G	2%	/home

Filesystems

- Linux filesystems are created in partitions or volumes
 - ext2fs (2nd Extended File System) is old
 - ext3fs (3rd Extended File System) is common
 - Augments ext2fs to incorporate journaling
 - Journals contain filesystem updates
 - Journal log can reconstruct consistent filesystem
 - Journal speeds filesystem consistency checks
 - ext4fs (Fourth Extended File System) is modern
 - Speeds large directories
 - Compatible with ext2 and ext3
 - Other filesystems also supported
 - ReiserFS, IBM's JFS, SGI's XFS
 - Can read foreign filesystems (e.g., FAT, NTFS, ISO 9660)

ext# filesystems

- For ext2/ext3/ext4, **mke2fs** is used, which creates
 - A set of inode storage cells
 - each holds info about one file
 - A set of scattered “superblocks”
 - holds global filesystem info (multiple copies for reliability)
 - size and location of inode tables, block map and usage, etc.
 - A map of the disk blocks in the filesystem (used and free)
 - The set of data blocks

Mounting a filesystem

- Filesystem must be mounted before use
 - Must be made part of root filesystem
- Can be mounted on (top of) any directory
 - # mount /dev/sda1 /usr/local**
 - # df /usr/local**
- Use /mnt for temporary mounts
- Want to set up automatic mounting

/etc/fstab

- (Almost) every filesystem that the system knows about automatically is in /etc/fstab

```
[root@brian]# more /etc/fstab
# /etc/fstab
# Created by anaconda on Thu Jan 19 14:11:35 2012
#
# Accessible filesystems, by ref., are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and blkid(8) for more
#
/dev/mapper/vg_davison-lv_root / ext4 defaults 1 1
UUID=52bb6031-5fda-402e-bb9f-5c0fee93ca44 /boot ext4 defaults 1 2
/dev/mapper/vg_davison-lv_home /home ext4 defaults 1 2
/dev/mapper/vg_davison-lv_swap swap swap defaults 0 0
tmpfs /dev/shm tmpfs defaults 0 0
devpts /dev/pts devpts gid=5,mode=620 0 0
sysfs /sys sysfs defaults 0 0
proc /proc proc defaults 0 0
```

[u]mounting, swap

- **mount**, **umount**, **swapon** and **fsck** all read the `/etc/fstab` file
- enables
 - # **mount /mnt/cdrom**
- `fstab` entries must be in the correct order
- at startup
 - **mount -a** executed, mounts all regular partitions
 - **swapon** enables swapping on all swap partitions

fsck: check and repair filesystems

- During power failure, superblock, inodes, and data blocks may not get written to disk
- **fsck** can fix minor damage (ext3/4 systems quickly)
 - unreferenced inodes
 - inexplicably large link counts
 - unused data blocks not recorded in block maps
 - data blocks listed as free that are also used in a file
 - incorrect summary info in superblock
- More complex damage will make **fsck** ask human
 - Places unfixable files in lost+found directory
 - You should re-run **fsck** until no errors are found

The Filesystem

- A filesystem incorporates:
 - A way of naming and organizing things (namespace)
 - An API for navigating and manipulating objects
 - A security model for protecting, hiding, and sharing objects
 - An implementation to tie the model to the hardware
- Linux abstract kernel interface supports many different filesystems
 - from disk, network, memory

Pathnames

- The Linux file system is a single unified hierarchy, starting with / (the root directory)
- A pathname can be
 - absolute
 - /etc/passwd
 - relative
 - ./passwd
 - Always starts with current working directory
- No technical limitations on file naming other than length and /
 - some chars are more difficult to use (need quotes or escape)

Mounting & unmounting filesystems

- The filesystem is made of smaller filesystems
- Most filesystems occupy disk partitions
 - but can be anything that obeys the API
- Filesystems may be added or removed using the mount and umount commands
 - The mount point is a directory
 - Ex:
mount /dev/hdc1 /backup

[u]mounting filesystems

- List of filesystems is in `/etc/fstab`
 - Such filesystems are checked (**fsck -A**) and mounted (**mount -a**) at boot
- `umount` will fail if the filesystem is busy
 - busy = any open files, processes with cwd, or copies of executing programs
 - **/sbin/fuser** will show such processes
 - f – file open for reading or writing
 - c – process cwd is on filesystem
 - e – process is executing a file
 - r – process root dir is on filesystem
 - m – process has mapped file or shared lib

File tree organization

- Not really well organized
- Many files organized by function
 - difficult to upgrade
 - /etc/ contains files that are never customized, and ones that are entirely local
- There is at least one place for everything
- Admins need to learn standard places, not move or use new ones

Filesystem hierarchy

<http://www.pathname.com/fhs/>

- /bin : Essential user command binaries (for use by all users)
- /boot : Static files of the boot loader (e.g., kernel)
- /dev : Device files (terminals, disks, modems, etc.)
- /etc : Host-specific system configuration
- /home : User home directories (optional)
- /lib : Essential shared libraries and kernel modules
- /media : Filesystems on removable media
- /opt : Add-on application software packages
- /proc : Kernel and process information virtual filesystem
- /root : Home directory for the root user (optional)
- /sbin : Static system binaries for repairing, booting, & recovering OS
- /tmp : Temporary files (that disappear at reboot)
- /usr : (more next slide)
- /var : (more next slide)

/usr, /var

/usr

/usr/bin : Most commands and executables

/usr/include : Header files for C programs

/usr/lib : Libraries and support files for standard programs

/usr/local : Local software (stuff you install)

/usr/man : Manual pages

/usr/sbin : Less essential sysadmin commands

/usr/share : Content that is common to multiple systems (RO)

/usr/src : Source code for (nonlocal) software packages

/var

/var/adm : Various logs, system setup records

/var/log : System log files

/var/spool : Spooling directories for printers, mail, dns

/var/tmp : More temporary space (preserved between reboots)

File types

- Linux defines seven types of files
 - [-] - Regular files
 - [d] - Directories
 - [c] - Character device files
 - [b] - Block device files
 - [s] - Local domain sockets
 - [p] - Named pipes (FIFO)
 - [l] - Symbolic links
- **ls -ld** shows the filetype of a file

Directories

- Created with **mkdir**, deleted with **rmdir** (if empty) or **rm -r**
- Contains named references (links) to other files
- Special entries “.” and “..” refer to self and parent directories respectively
- Filenames are stored within parent directory
- More than one directory entry can refer to the same file (hard links)
 - Can be created with **ln**, removed with **rm**

Character and block device files

/dev/

- Allow programs to communicate with hardware
 - When kernel gets request that refers to device file, it is handed off to the device driver
- Character (raw) device files: drivers do i/o buffering
- Block device files: handle i/o in large chunks
- Characterized by major (which driver) and minor (which device) device numbers

```
crw-rw---- 1 root lp 6, 0 Jan 30 2003 /dev/lp0
```
- Created with **mknod** and deleted by **rm**
 - Usually managed automatically by system

Sockets & pipes

- Local domain sockets
 - Sockets provide connections between processes
 - Local/UNIX domain sockets are only accessible through the `file` system
 - Only used by processes involved in connection
 - Created with `socket`, deleted by **`rm`** or **`unlink`**
 - Used by X Windows, `syslog`, and printing system
- Named pipes
 - FIFO `file`s that allow communication between processes on same host
 - Created with **`mknod`** and deleted with **`rm`**

Symbolic links

- Commonly used to reorganize a subtree, or provide multiple points of access to a file
- “Soft links” -- record path information, but not actual file
- Created by **ln -s**, deleted with **rm**
- Can contain absolute or relative path
 - # **ln -s ../parent**
 - # **ln -s /etc/mime.types .mime.types**
- First arg is recorded, not resolved until use

File attributes

- Every file has 12 mode bits
(four octal values of 3 bits each)
- First three bits:
 - 4000 – setuid
 - 2000 – setgid
 - 1000 – sticky bit
 - On a directory, means only the owner of the file, directory, or superuser can delete or rename files
 - Keeps /tmp more private and secure

Permission bits

- Nine permission bits
 - User:owner read, write, execute
 - 400, 200, 100
 - Group read, write, execute
 - 40, 20, 10
 - Other:world read, write execute
 - 4,2,1
- Ability to delete or rename is controlled by permissions on directory

Examples

```
-rwxr-xr-x  3 root  root  63555 Mar 13  2002 /bin/gzip
crw--w----  1 root  root  4,   0 Aug  4  2003 /dev/tty0
```

- **chmod** changes permissions
- **chown** changes ownership and group
 - # **chown -R user.group /home/user**
- **umask**
 - Set shell parameters to control default permissions
 - **umask 027** gives everything to owner, forbids writes to group, and gives nothing to other users
 - Usually set in `/etc/profile` or `/etc/csh.login`