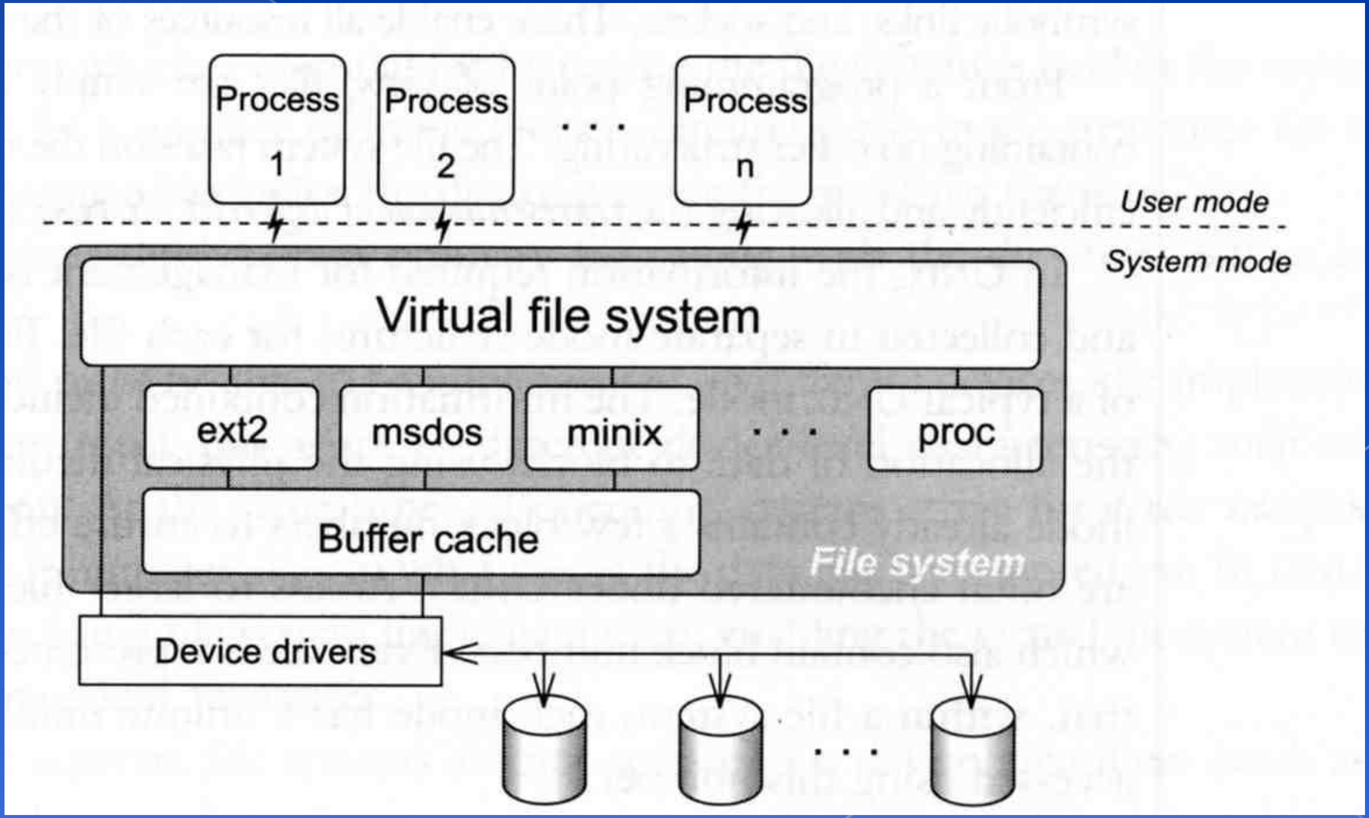
Chapter 5
The Filesystem

Outline

- File System Architecture
- Pathname
- File Tree
- Mounting
- File Types
- inode and file
- Link
- File Access Mode
- Changing File Owner
- FreeBSD bonus flags

File System Architecture (1)

- Application ⇔ Kernel ⇔ Hardware
  - Applications call system-calls to request service
  - Kernel invokes corresponding drivers to fulfill this service



File System Architecture (2)

- The basic purpose of filesystem
  - Represent and organize the system’s storage
  - Four main components:
    - Namespace
      - A way of naming things and arranging them in a hierarchy
    - API
      - A set of system calls for navigating and manipulating nodes
    - Security model
      - A scheme for protecting, hiding and sharing things
    - Implementation
      - Code that ties the logical model to an actual disk

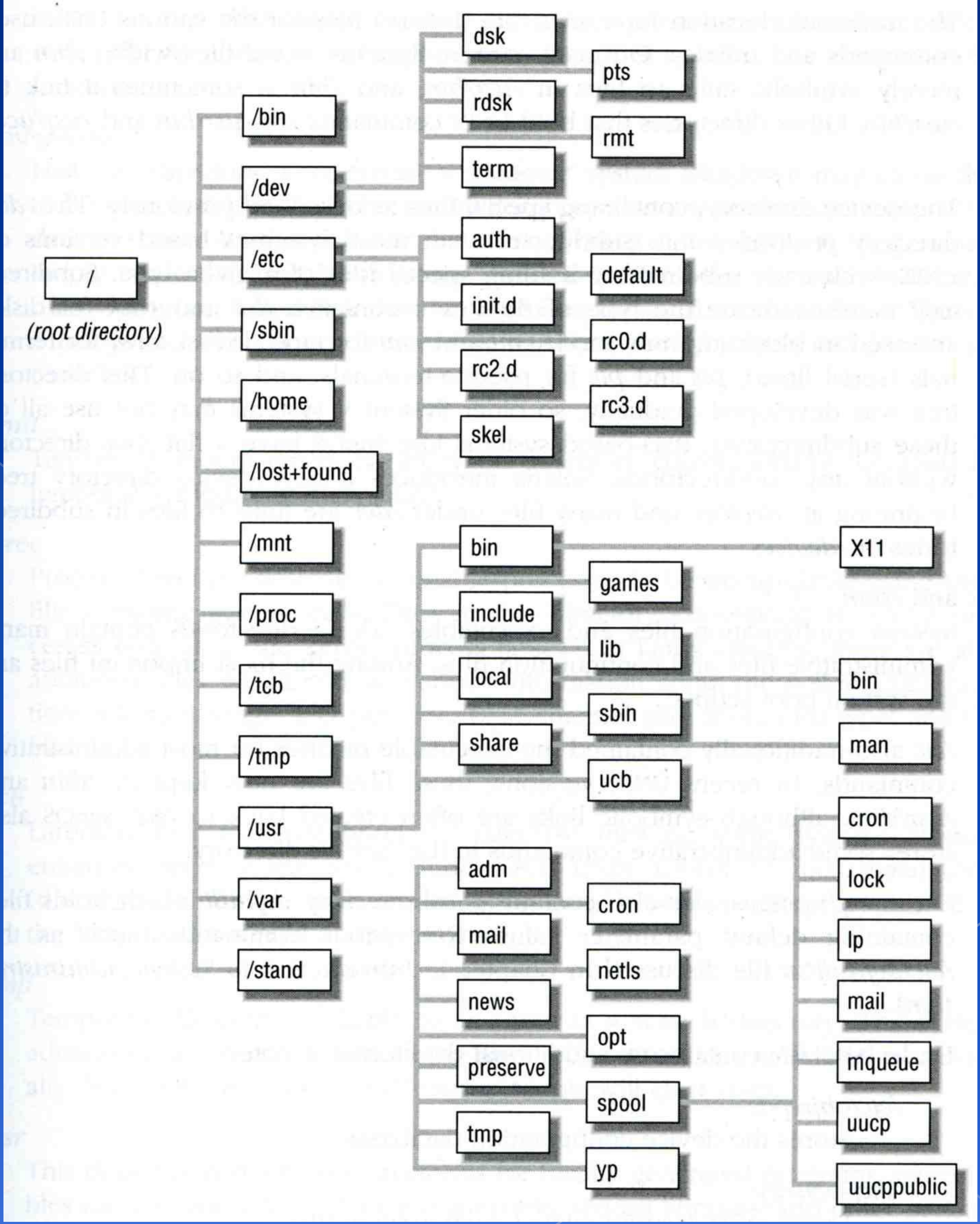
File System Architecture (3)

- Objects in the filesystem:
  - What you can find in a filesystem:
    - Files and directories
    - Hardware device files
    - Processes information
    - Interprocess communication channel
    - Shared memory segments
  - We can use common filesystem interface to access such “object”
    - open、read、write、close、seek、ioctl…

pathname

- Two kinds of path
  - Absolute path ➔ start from /
    - Such as /u/gcp/94/9455648/killme/haha.c
  - Relative path ➔ start from your current directory
    - Such as ../test/hehe.c
- Constrains of pathname
  - Single component: ≦ 255 characters
  - Single absolute path: ≦ 1023 characters

File Tree


/
(root directory)
/bin
/dev
/etc
/sbin
/home
/lost+found
/mnt
/proc
/tcb
/tmp
/usr
/var
/stand
dsk
pts
rdsk
rmt
term
auth
default
init.d
rc0.d
rc2.d
rc3.d
skel
bin
X11
games
include
lib
local
bin
sbin
share
man
ucb
cron
lock
adm
cron
mail
lp
netls
mail
news
opt
mqueue
preserve
spool
uucp
tmp
yp
uucppublic

Layout of File Systems (1)

pathname	Contents
/	The root directory of the file system
/bin & /sbin	User utilities & system programs fundamental to both single-user and multi-user environments
/usr	User utilities and applications
/usr/bin & /usr/sbin	Local executable
/lib	Shared and archive libraries
/libexec	Critical system utilities needed for binaries in /bin and /sbin
/mnt	Empty directory commonly used by system administrators as a temporary mount point
/tmp	Temporary files that are not guaranteed to persist across sys- tem reboots, also, there is /var/tmp
/usr/lib	Support libraries for standard UNIX programs
/usr/libexec	System daemons & system utilities (executed by other programs)
/usr/include	Libraries Header files
/usr/local	local executables, libraries, etc

Layout of File Systems (2)

pathname	Contents
/usr/src	BSD, third-party, and/or local source files
/usr/obj	architecture-specific target tree produced by building the /usr/src tree
/etc	system configuration files and scripts
/usr/local/etc	/etc of /usr/local, mimics /etc
/dev	Device entries for disks, terminals, modems, etc
/proc	Images of all running process
/var	Multi-purpose log, temporary, transient, and spool files
/var/db	Database files
/var/db/pkg & /var/db/ports	Ports Collection management files. ports(7)
/var/log	Various system log files
/var/mail	user mailbox files
/var/spool	Spooling directories for printers, mails, etc

hier(7)

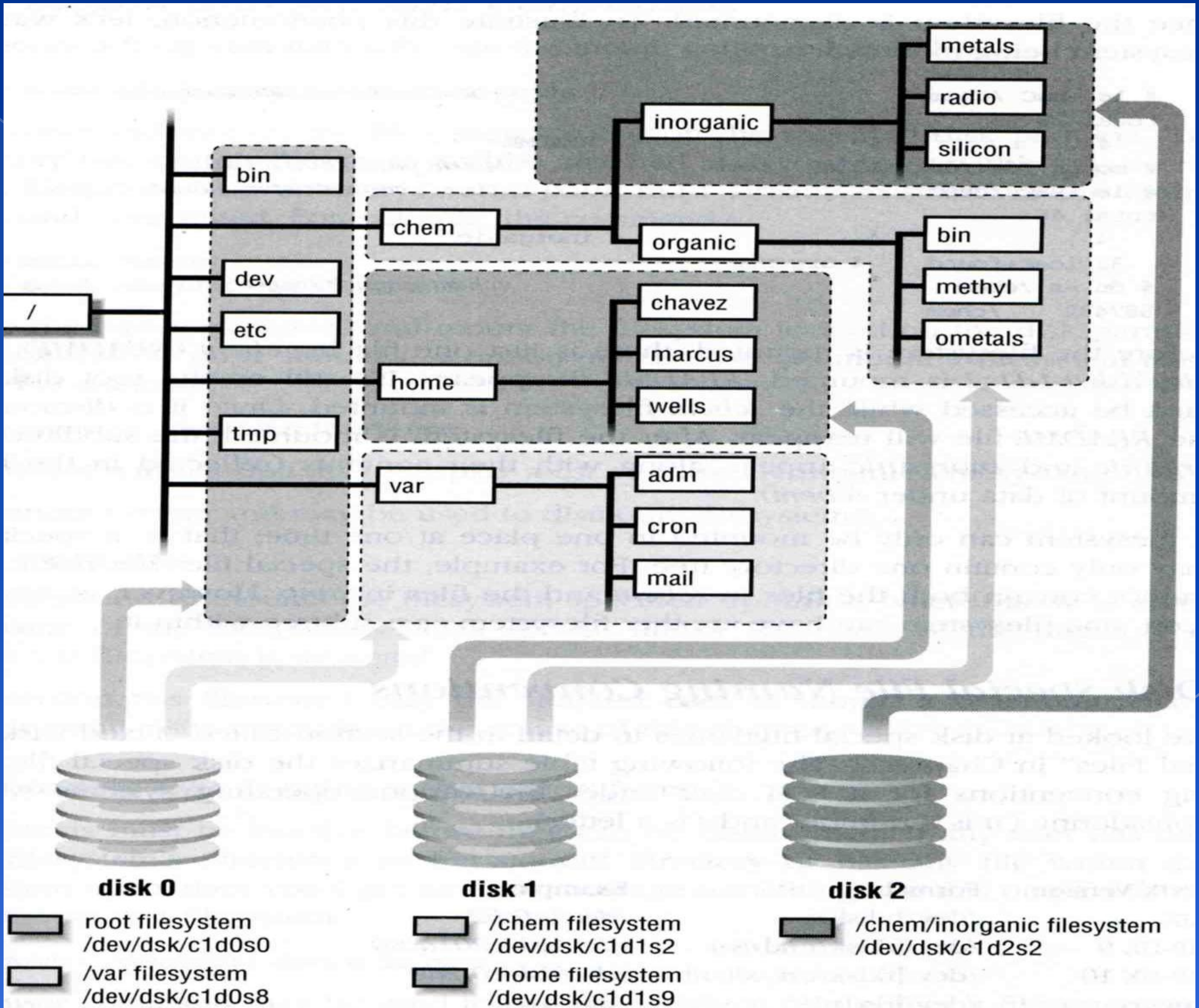
Mounting file system (1)

- The filesystem in composed of chunks
  - Most are disk partitions
  - Network file servers
  - Memory disk emulators
  - Kernel components
  - …, etc.
- "mount" command
  - Map the mount point of the existing file tree to the root of the newly attached filesystem
  - $ mount /dev/ad2s1e /home2
  - The previous contents of the mount point become inaccessible

mount(8)

Mounting file system (2)

- Example



Mounting file system (3)

- Filesystem table – fstab
  - Automatically mounted at boot time
  - /etc/fstab
    - Filesystem in this file will be checked and mounted automatically at boot time

Ex. bsd1's /etc/fstab

```
# Device        Mountpoint    FStype  Options        Dump    Pass#
/dev/ad0s1b     none          swap    sw             0       0
/dev/ad0s1a     /             ufs     rw             1       1
/dev/ad0s1e     /backup       ufs     rw             2       2
/dev/ad0s1d     /home         ufs     rw,noatime,nosuid  2       2
/dev/acd0       /cdrom        cd9660  ro,noauto      0       0
csduty:/bsdhome /bsdhome      nfs     rw,noauto      0       0
```

fstab(5)

Mounting file system (4)

- Unmounting File Stsyem
  - "umount" command
    - $ umount node | device
      - Ex: umount /home, umount /dev/ad0s1e
  - Busy filesystem
    - Someone's current directory is there or there is opened file
    - Use "umount –f"
    - We can use "lsof" or "fstat" like utilities to figure out who makes it busy

Mounting file system (5)

- lsof, fuser and fstat commands
  - lsof (sysutils/lsof) - list open files

```
knight:~ -lwhsu- lsof /home/lwhsu
COMMAND      PID  USER   FD    TYPE DEVICE SIZE/OFF     NODE NAME
ssh         1848 lwhsu  cwd    VDIR   0,89     7168 16109568 /home/lwhsu
tcsh        3826 lwhsu  cwd    VDIR   0,89     7168 16109568 /home/lwhsu
lsof        4398 lwhsu  cwd    VDIR   0,89     7168 16109568 /home/lwhsu
```

  - fuser (sysutils/fuser) - list IDs of all processes that have one or more files open

```
knight:~ -lwhsu- fuser /home/lwhsu
/home/lwhsu: 33686c 11196c  5189c 50352c 69153c
```

  - fstat (FreeBSD) - identify active files

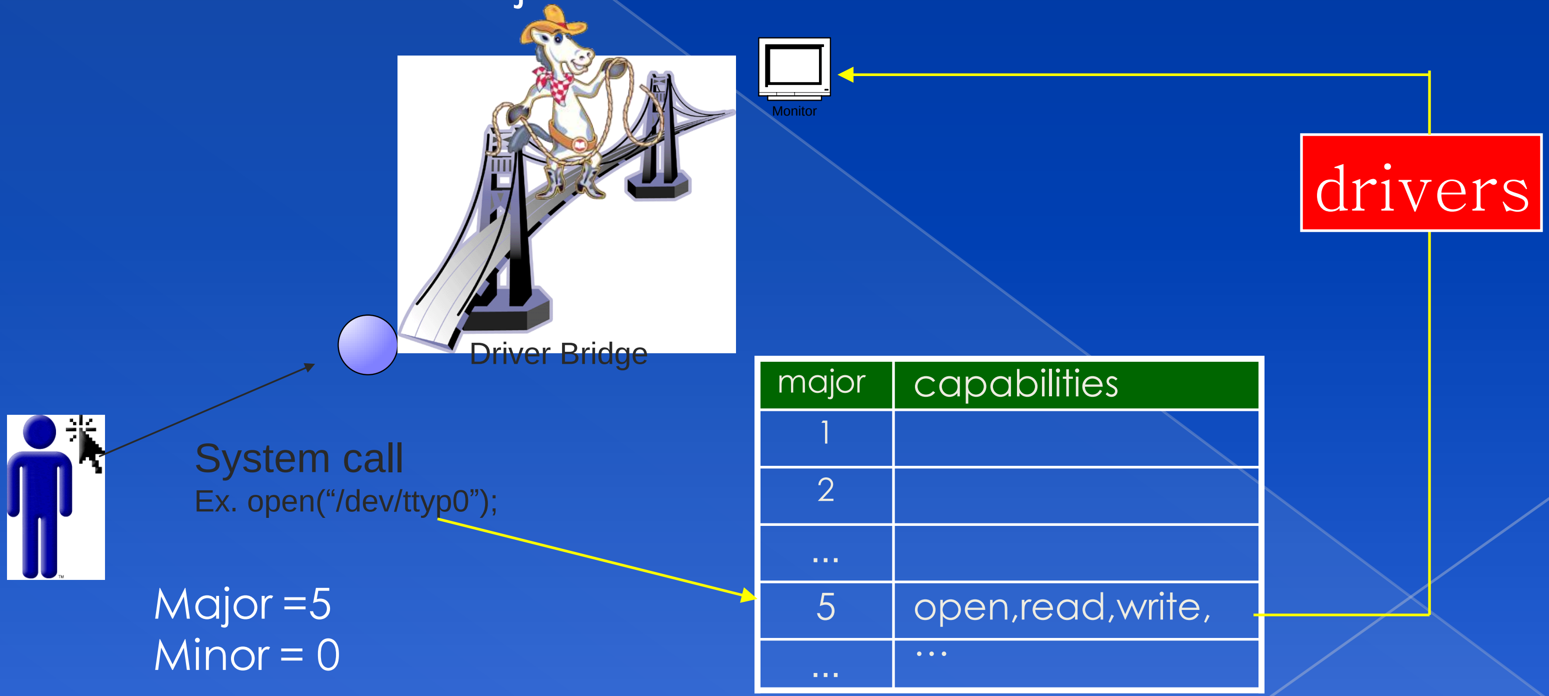
```
knight:~ -lwhsu- fstat /home/lwhsu
USER     CMD          PID   FD MOUNT      INUM MODE         SZ|DV R/W NAME
lwhsu    fstat      98620   wd /home   16109568 drwxr-xr-x    7168  r  /home/lwhsu
lwhsu    tcsh       72861   wd /home   16109568 drwxr-xr-x    7168  r  /home/lwhsu
lwhsu    ssh        16600   wd /home   16109568 drwxr-xr-x    7168  r  /home/lwhsu
```

File Types (1)

- File types
  - Regular files
  - Directories
    - Include “.” and “..”
  - Character and Block device files
  - UNIX domain sockets
  - Named pipes
  - Symbolic links

File Types (2)

- character and block device files
  - Use "mknod" to build special file
    - $ mknod name [b | c] major minor [owner:group]
      - The same major number will use the same driver



Monitor

drivers

Driver Bridge

System call

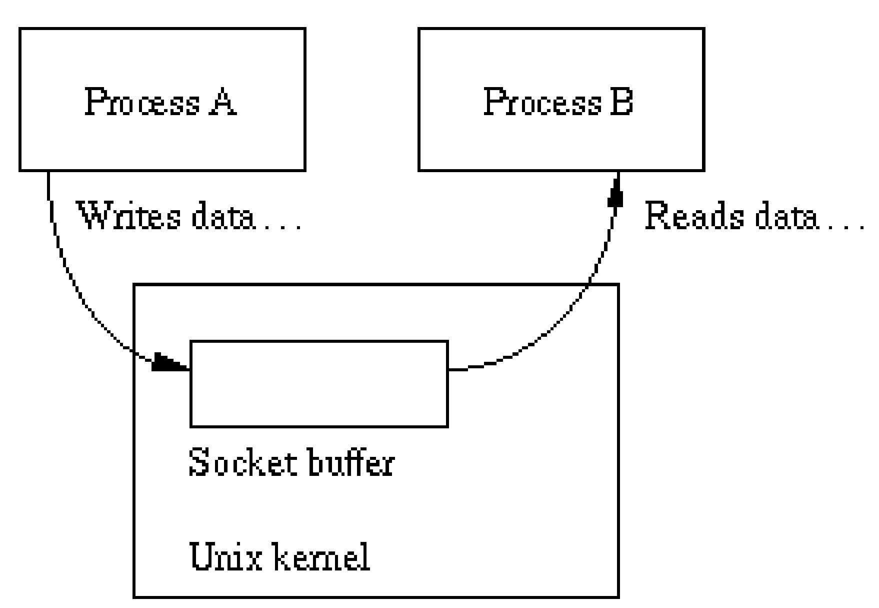
Ex. open("/dev/ttyp0");

Major =5
Minor = 0

major	capabilities
1	
2	
…	
5	open,read,write,
…	…

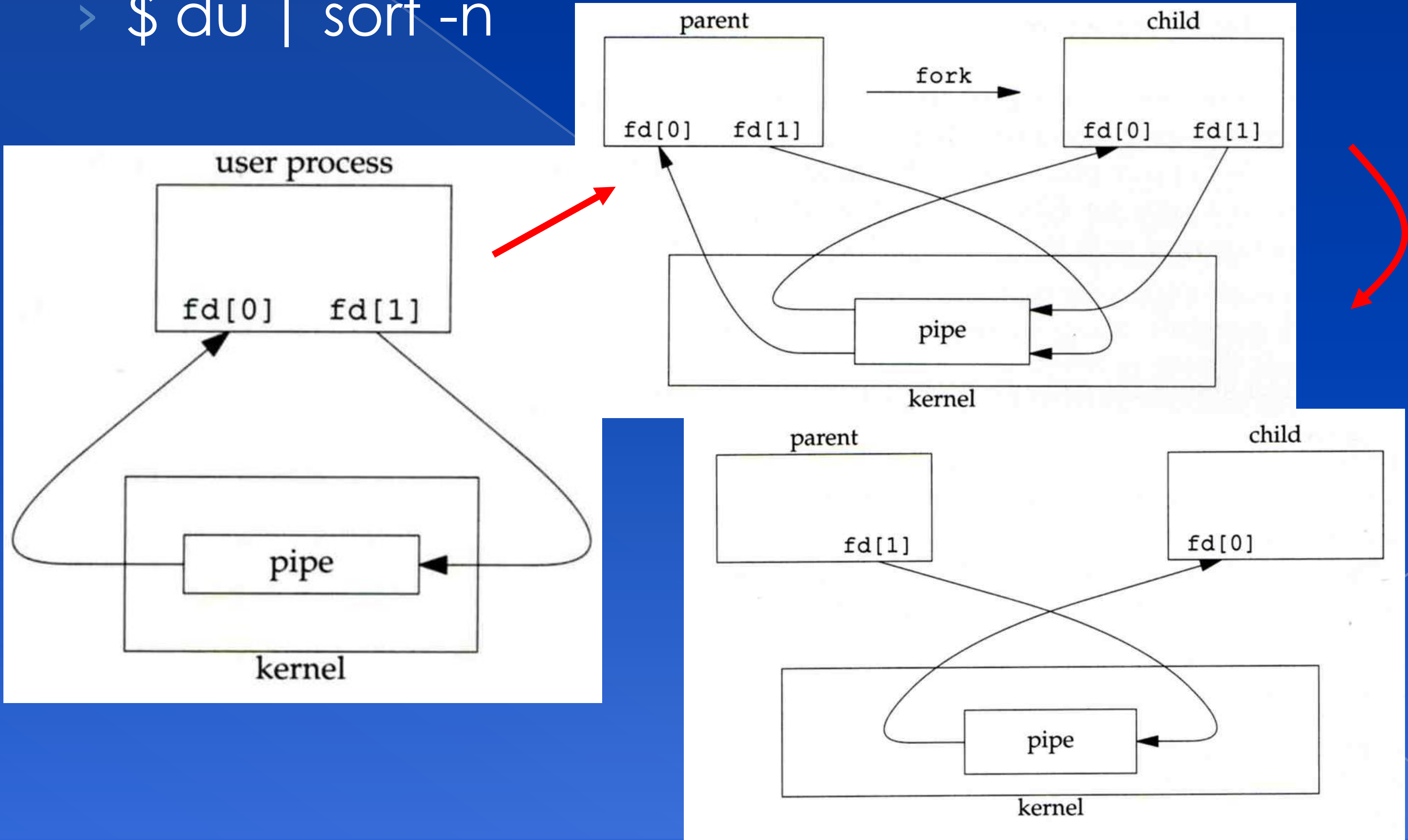
File Types (3)

- UNIX domain socket
  - Created by socket()
  - Local to a particular host
  - Be referenced through a filesystem object rather than a network port



File Types (5)

- Pipe
  - $ du | sort -n



File Types (4)

- Named Pipe
  - Let two processes do “FIFO” communication
  - $ mkfifo [-m mode] fifo_name ...

```
$ mkfifo pipe
$ du >> pipe
(another process)
$ sort -n pipe
```

mkfifo(2)

File Types (6)

- Symbolic Link
  - A file which points to another pathname
  - $ ln -s source_file target_file
  - Like “short-cut” in Windows

File Types (7)

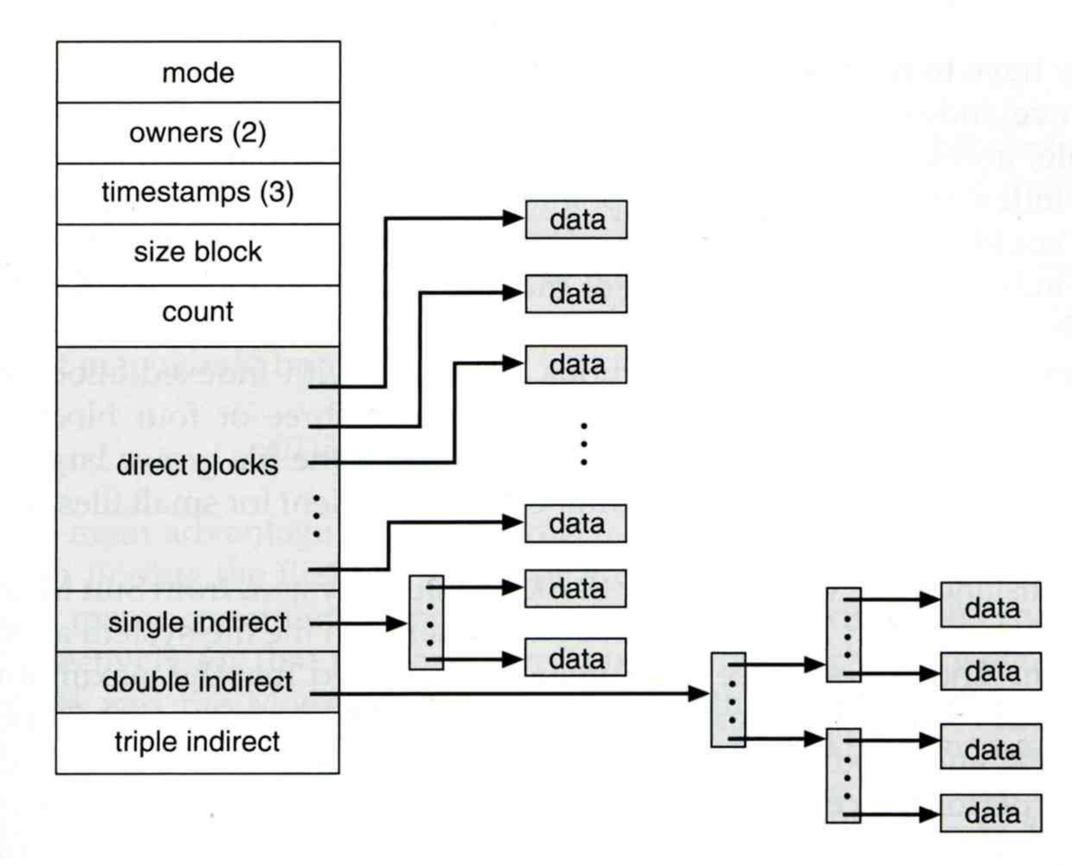
- File type encoding used by ls

File type	Symbol	Created by	Removed by
Regular file	-	editors, cp, etc	rm
Directory	d	mkdir	rmdir, rm -r
Character device file	c	mknod	rm
Block device file	b	mknod	rm
UNIX domain socket	s	socket(2)	rm
Named pipe	p	mknod	rm
Symbolic link	l	ln -s	rm

ls(1), “The Long Format” section

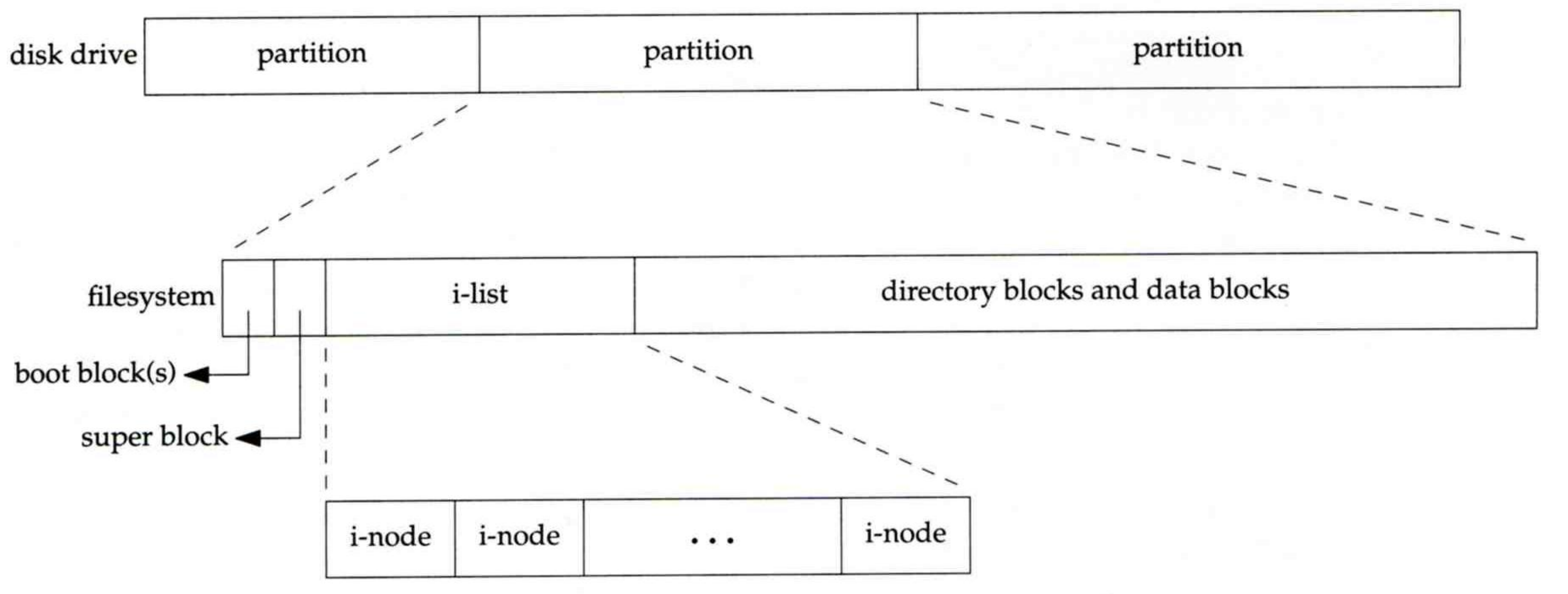
inode and file (1)

- inode
  - A structure that records information of a file
    - ls -i



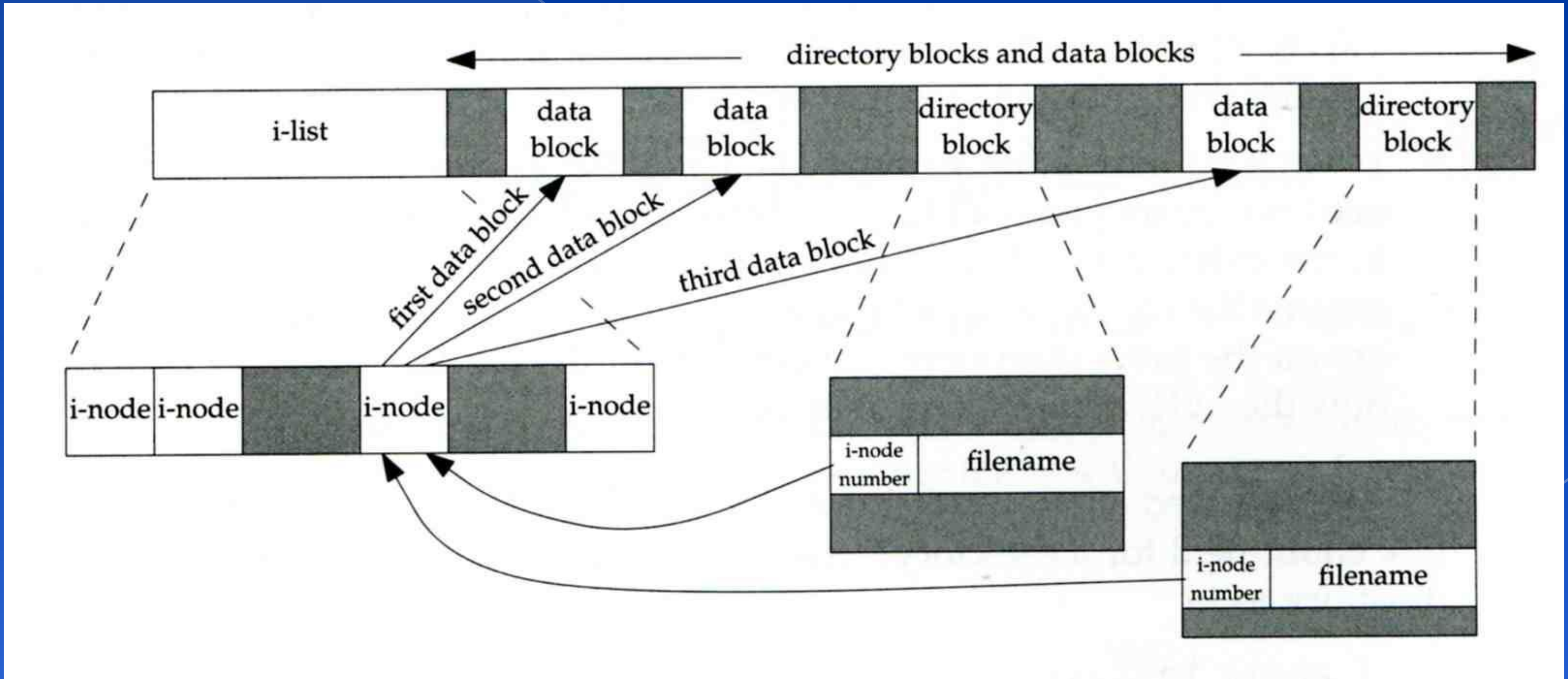
inode and file (2)

- Filesystem
  - Boot blocks
  - Super block
  - Inode list
  - Data block



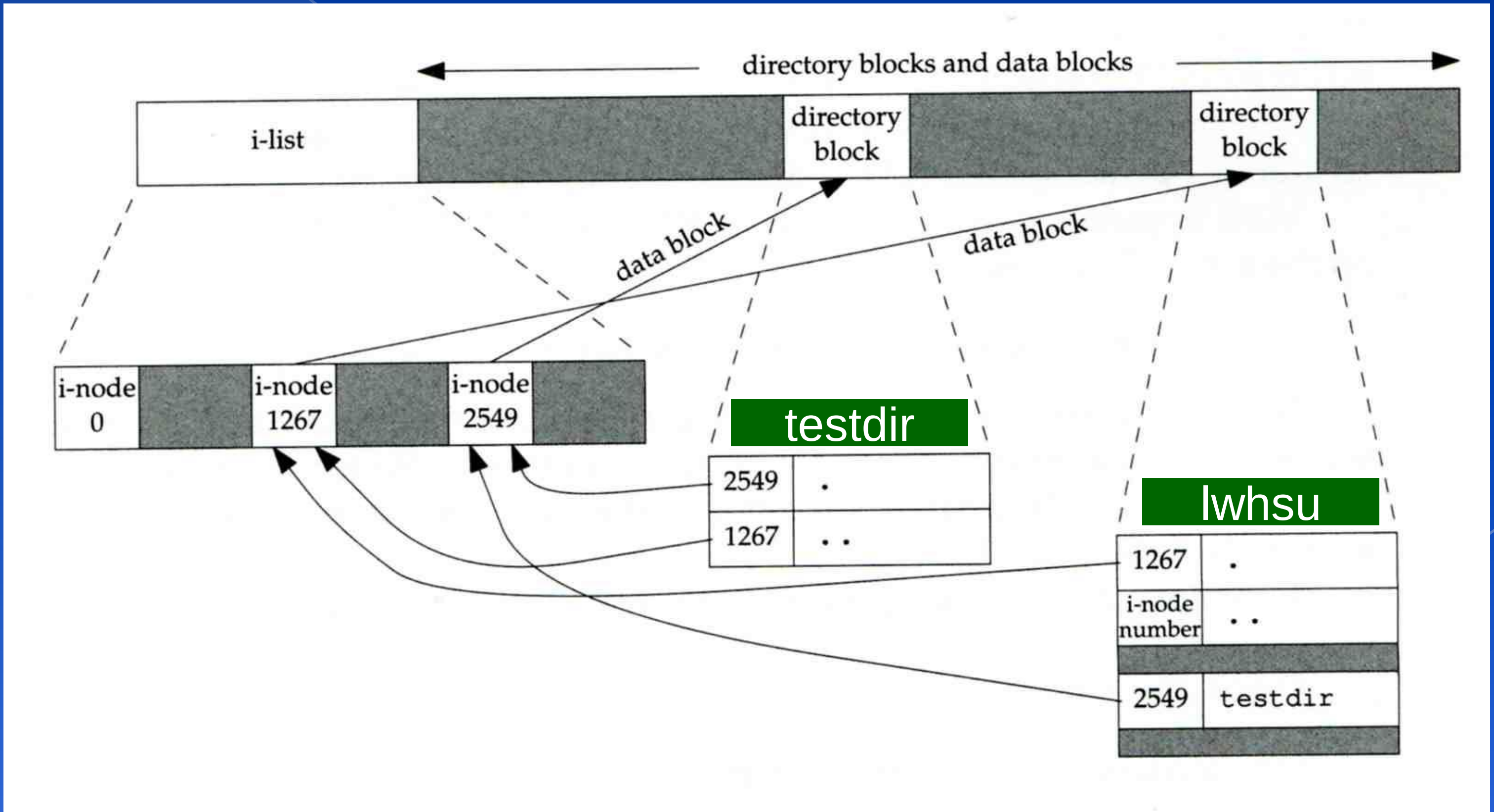
inode and file (3)

- More detail of inode and data block



inode and file (4)

- .
- ..
- testdir

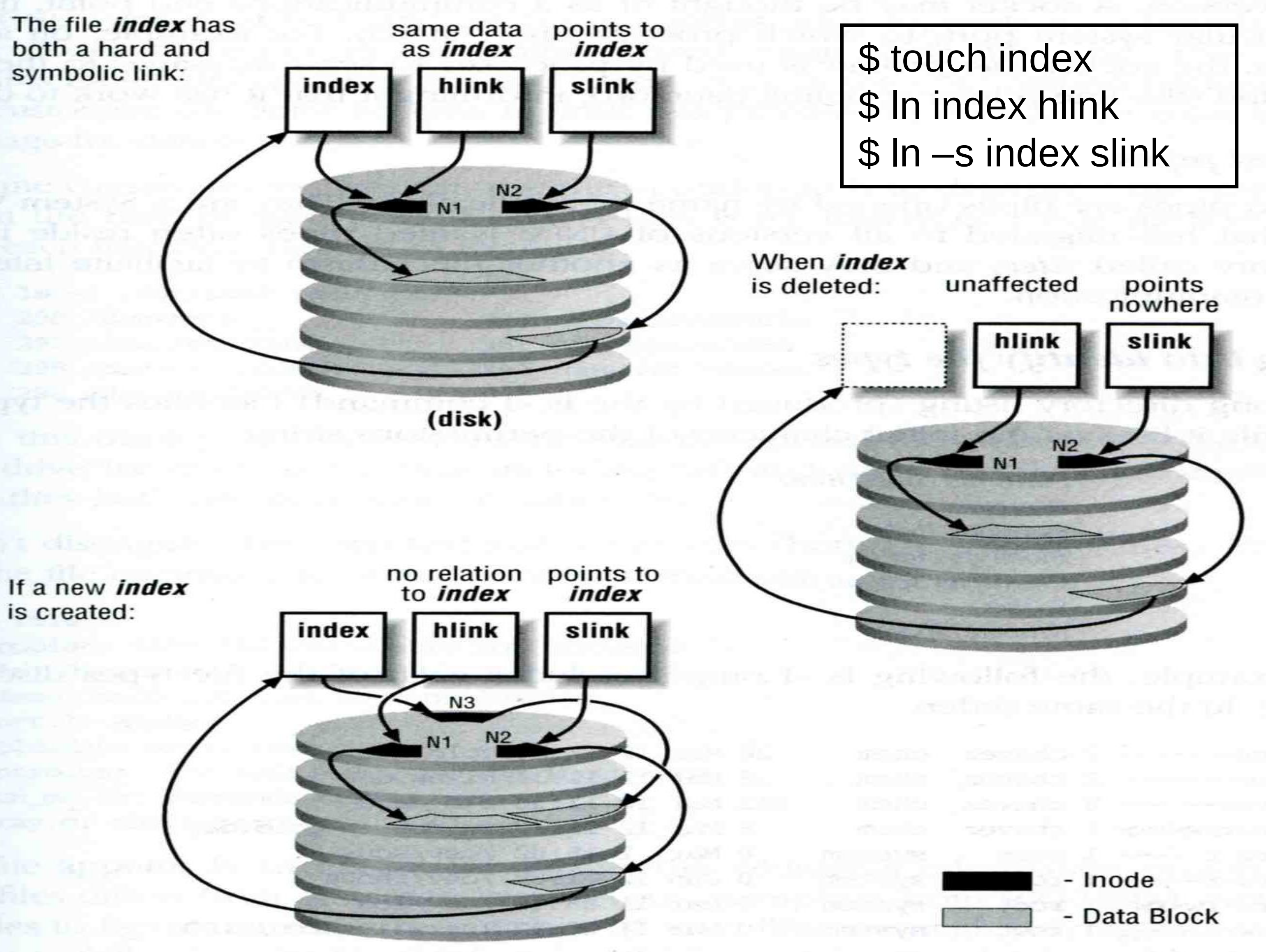


/home/lwhsu/adir

Hard Link V.S. Symbolic Link (1)

- Link
  - Hard link
    - associate two or more filenames with the same inode
    - $ ln source_file target_file
  - Soft (symbolic) link
    - A file which points to another pathname
    - $ ln -s source_file target_file

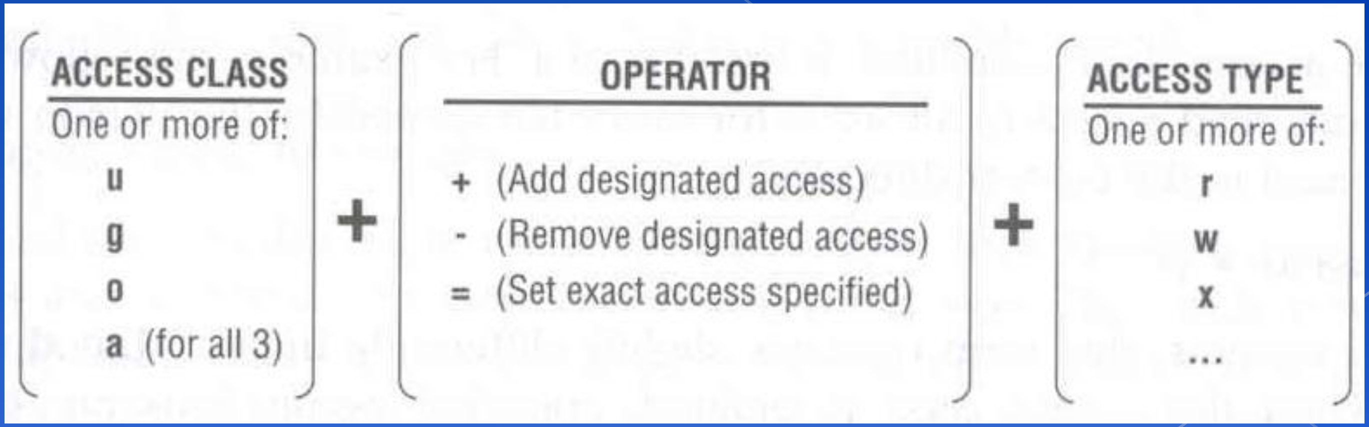
Hard Link V.S. Symbolic Link (2)



```
$ touch index
$ ln index hlink
$ ln –s index slink
```

File Access Mode (1)

- rwx r-x r-x
  - User, group, other privileges
- chmod command
  - `$ chmod access-string file ...`
    - `$ chmod u+x test.sh`
    - `$ chmod go-w .tcshrc`
    - `$ chmod u+w,r-w hehe haha`
    - `$ chmod –R 755 public_html/`



ACCESS CLASS
One or more of:
u
g
o
a (for all 3)

+

OPERATOR
+ (Add designated access)
- (Remove designated access)
= (Set exact access specified)

+

ACCESS TYPE
One or more of:
r
w
x
...

chmod(1), “MODES” section

File Access Mode (2)

- setuid, setgid, sticky bit
  - setuid, setgid on file
    - The effective uid/gid of resulting process will be set to the UID/GID of the file
    - setuid
      - passwd, chsh, crontab
    - setgid
      - top, fstat, write
  - setgid on directory
    - Cause newly created files within the directory to be the same group as directory
  - sticky on directory
    - Do not allow to delete or rename a file unless you are
      - The owner of the file
      - The owner of the directory
      - root

File Access Mode (3)

- Decimal argument of chmod
  - setuid: 4000
  - setgid: 2000
  - stiky : 1000

Mode	Attribute	Mode	Attribute
755	- rwx r-x r-x	644	- rw- r-- r--
4755	- rws r-x r-x	600	- rw- --- ---
2755	- rwx r-s r-x	400	- r-- r-- r--
2775	d rwx rws r-x	1777	d rwx rwx rwt
755	d rwx r-x r-x	4555	- r-s r-x r-x
750	d rwx r-x ---	711	- rwx --x --x
700	d rwx --- ---	711	d rwx --x --x

File Access Mode (4)

- Assign default permissions: umask
  - Shell built-in command
  - Inference the default permissions given to the files newly created.
  - The newly created file permission:
    - Use full permission bit (file: 666, dir: 777) xor umask value.
  - Example:

umask	New File	New Dir
022	- rw- r-- r--	d rwx r-x r-x
033	- rw- r-- r--	d rwx r-- r--
066	- rw- --- ---	d rwx --x --x
000	- rw- rw- rw-	d rwx rwx rwx
477	- r-- --- ---	d r-x --- ---
777	- --- --- ---	d --- --- ---

Changing File Owner

- Changing File Owner/Group
  - Commands:
    - chown -- change user owner
    - chgrp -- change group owner
- Change the file ownership and group ownership
  - $ chown –R lwhsu /home/lwhsu
  - $ chgrp –R gcs /home/lwhsu
  - $ chown –R lwhsu:gcs /home/lwhsu
  - $ chown –R :gcs /home/lwhsu

FreeBSD bonus flags

- chflags command

schg	system immutable flag	(root only)
sunlnk	system undeletable flag	(root only)
sappnd	system append-only flag	(root only)
uappend	user append-only flag	(root, user)
uunlnk	user undeletable flag	(root, user)
…		

```
knight:~/killme -lwhsu- touch file
knight:~/killme -lwhsu- ls -lo
-rw-r--r--  1 lwhsu  user   - 0 Oct  3 18:23 file

knight:~/killme -lwhsu- chflags uunlnk file
knight:~/killme -lwhsu- ls -lo
-rw-r--r--  1 lwhsu  user  uunlnk 0 Oct  3 18:23 file

knight:~/killme -lwhsu- rm -f file
rm: file: Operation not permitted

knight:~/killme -lwhsu- sudo rm -f file
rm: file: Operation not permitted

knight:~/killme -lwhsu- chflags nouunlnk file
knight:~/killme -lwhsu- rm -f file
knight:~/killme -lwhsu- ls –lo
knight:~/killme -lwhsu-
```

chflags(1)