CS370 Operating Systems

Colorado State University

Yashwant K Malaiya

Fall 2016 Lecture 37

Mass Storage/ File-system

Slides based on

- Text by Silberschatz, Galvin, Gagne
- Various sources

FAQ

- Disk scheduling: are all the requests received at the same time or one at a time?
- ECC: error correction code
- What is a bad sector? How does a sector become bad?
- Boot process:
  - Bootstrap loader code in ROM
  - Full bootstrap program on disk: from boot partition on boot disk
    - Windows: boot partition identified in MBR (Master Boot Record)
- Can a local volume span multiple disks without a RAID:
  - Links or “spanned volume”
- Is RAID repair automatic or manual? either. See your RAID manual.

RAID Structure

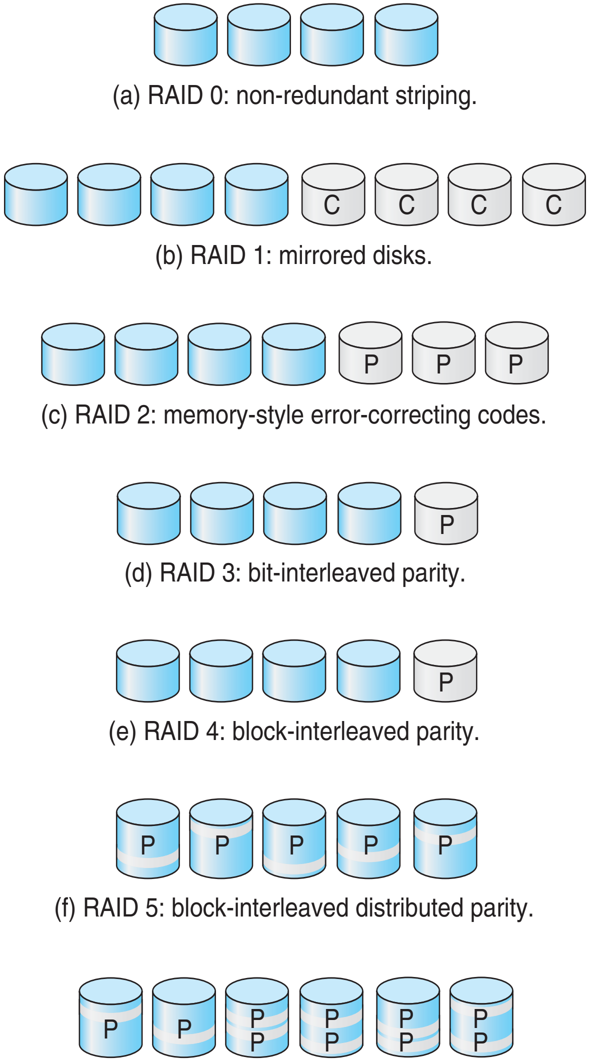
- RAID – redundant array of inexpensive disks
  - multiple disk drives provides reliability via **redundancy**
- Increases the **mean time to failure**
- **Mean time to repair –** exposure time when another failure could cause data loss
- **Mean time to data loss** based on above factors
- If mirrored disks fail independently, consider disk with 100,000 hour mean time to failure and 10 hour mean time to repair
  - Mean time to data loss is $100,000^2 / (2 * 10) = 500 * 10^6$ hours, or 57,000 years!

Inverse of probability that the two will fail within 10 hours

RAID Techniques

- **Striping** uses multiple disks in parallel by splitting data: higher performance, no redundancy (ex. RAID 0)
- **Mirroring** keeps duplicate of each disk: higher reliability (ex. RAID 1)
- **Block parity: One Disk hold** parity block for other disks. A failed disk can be rebuilt using parity. Wear leveling if interleaved (RAID 5, double parity RAID 6).
- Ideas that did not work: Bit or byte level level striping (RAID 2, 3) Bit level Coding theory (RAID 2), dedicated parity disk (RAID 4).
- Nested Combinations:
  - RAID 01: Mirror RAID 0
  - RAID 10: Multiple RAID 1, striping
  - RAID 50: Multiple RAID 5, striping
  - others

RAID Levels



(a) RAID 0: non-redundant striping.

(b) RAID 1: mirrored disks.

(c) RAID 2: memory-style error-correcting codes.

(d) RAID 3: bit-interleaved parity.

(e) RAID 4: block-interleaved parity.

(f) RAID 5: block-interleaved distributed parity.

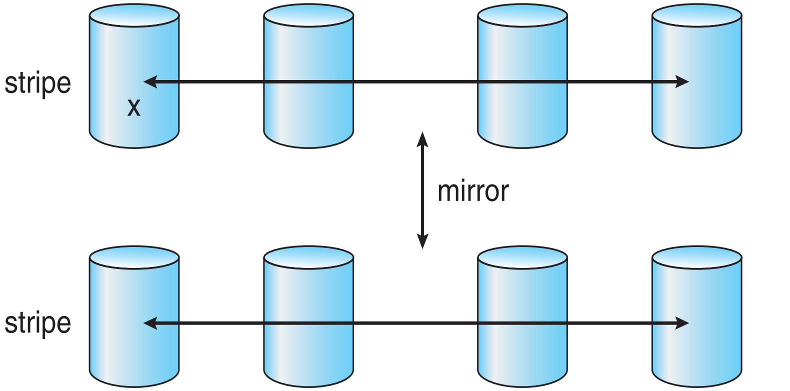
(g) RAID 6: P + Q redundancy.

Not common:
RAID 2, 3,4

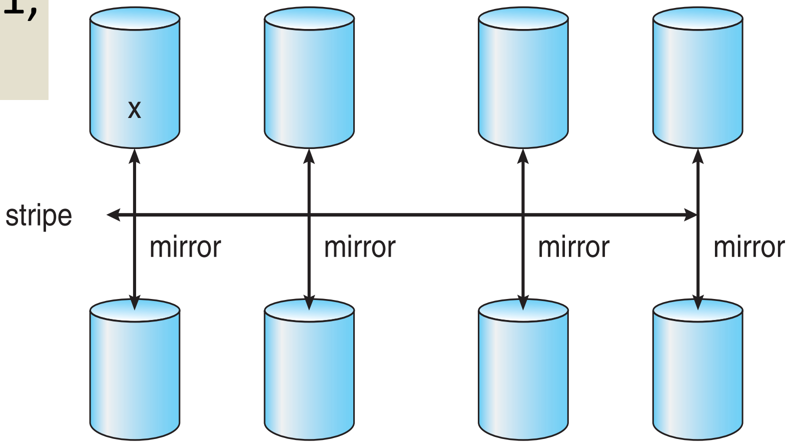
Most common
RAID 5

RAID (0 + 1) and (1 + 0)

RAID 01: Mirror RAID 0
RAID 10: Multiple RAID 1, striping



a) RAID 0 + 1 with a single disk failure.



b) RAID 1 + 0 with a single disk failure.

Colorado State University

CS370 Operating Systems

Colorado State University

Yashwant K Malaiya

Fall 2016 Lecture 37

File-system

Slides based on

- **Text by Silberschatz, Galvin, Gagne**
- **Various sources**

Chapter 11: File-System Interface

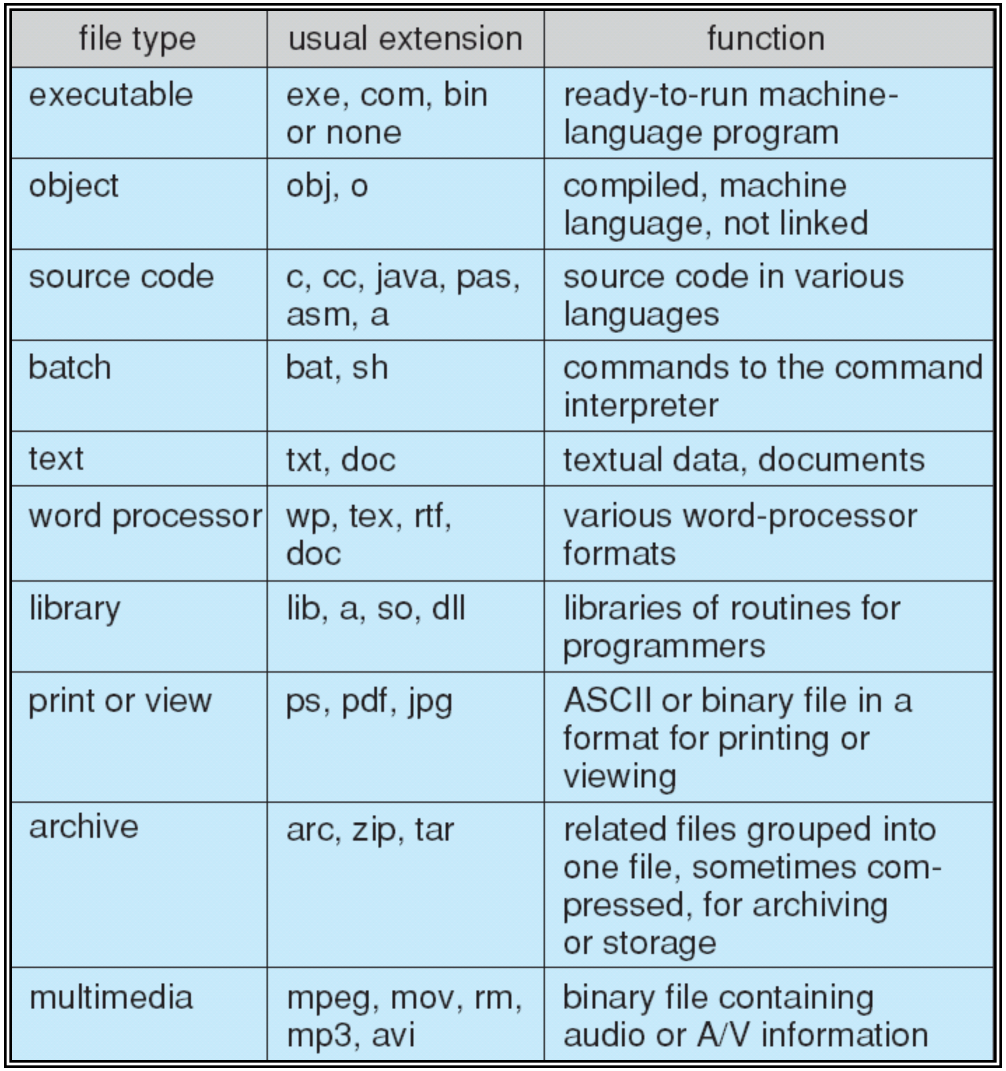
- File Concept
- Access Methods
- Disk and Directory Structure
- File-System Mounting
- File Sharing
- Protection

Outline

- File Concept, types
- Attributes, Access Methods, operations, Protection
- Directory Structure, namespace, File-System Mounting, File Sharing
- Storage abstraction: File system metadata (size, freelists), File metadata(attributes, disk block maps), datablocks
- Allocation of blocks to files: contiguous, sequential, linked list allocation, indexed
- In memory: Mount table, directory structure cache, open file table, buffers
- Unix: inodes numbers for directories and files

File types

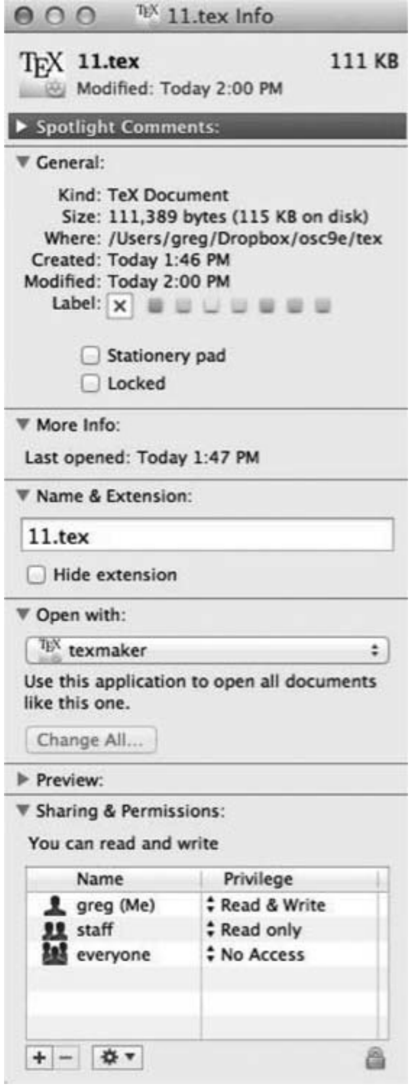
Type used by programs not OS



file type	usual extension	function
executable	exe, com, bin or none	ready-to-run machine-language program
object	obj, o	compiled, machine language, not linked
source code	c, cc, java, pas, asm, a	source code in various languages
batch	bat, sh	commands to the command interpreter
text	txt, doc	textual data, documents
word processor	wp, tex, rtf, doc	various word-processor formats
library	lib, a, so, dll	libraries of routines for programmers
print or view	ps, pdf, jpg	ASCII or binary file in a format for printing or viewing
archive	arc, zip, tar	related files grouped into one file, sometimes compressed, for archiving or storage
multimedia	mpeg, mov, rm, mp3, avi	binary file containing audio or A/V information

File Attributes

- **Name** – only information kept in human-readable form
- **Identifier** – unique tag (number) identifies file within file system
- **Type** – needed for systems that support different types
- **Location** – pointer to file location on device
- **Size** – current file size
- **Protection** – controls who can do reading, writing, executing
- **Time, date, and user identification** – data for protection, security, and usage monitoring
- Information about files are kept in the directory structure, which is maintained on the disk
- Many variations, including extended file attributes such as file checksum



11.tex Info

11.tex 111 KB
Modified: Today 2:00 PM

▶ Spotlight Comments:

▼ General:

Kind:	TeX Document
Size:	111,389 bytes (115 KB on disk)
Where:	/Users/greg/Dropbox/osc9e/tex
Created:	Today 1:46 PM
Modified:	Today 2:00 PM
Label:	☒

□ Stationery pad
□ Locked

▼ More Info:

Last opened: Today 1:47 PM

▼ Name & Extension:

11.tex

□ Hide extension

▼ Open with:

texmaker

Use this application to open all documents like this one.

Change All...

▶ Preview:

▼ Sharing & Permissions:

You can read and write

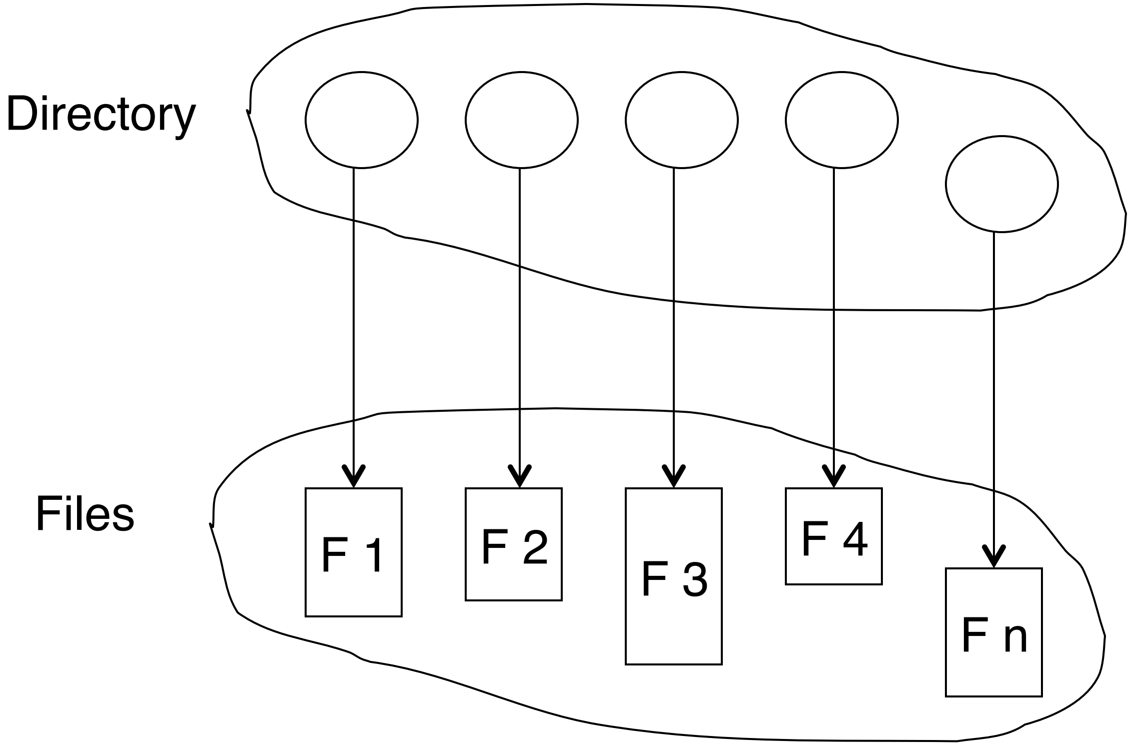
Name	Privilege
greg (Me)	Read & Write
staff	Read only
everyone	No Access

Disk Structure

- Disk can be subdivided into **partitions**
- Disks or partitions can be **RAID** protected against failure
- Partition can be **formatted** with a file system
- Entity containing file system known as a **volume**
- Each volume containing file system also tracks that file system's info in **device directory** or **volume table of contents**
- As well as **general-purpose file systems** there are many **special-purpose file systems**, frequently all within the same operating system or computer

Directory Structure

- A collection of nodes containing information about all files



Both the directory structure and the files reside on disk

Operations Performed on Directory

- Traverse the file system
- List a directory
- Search for a file
- Create/Delete/Rename a file

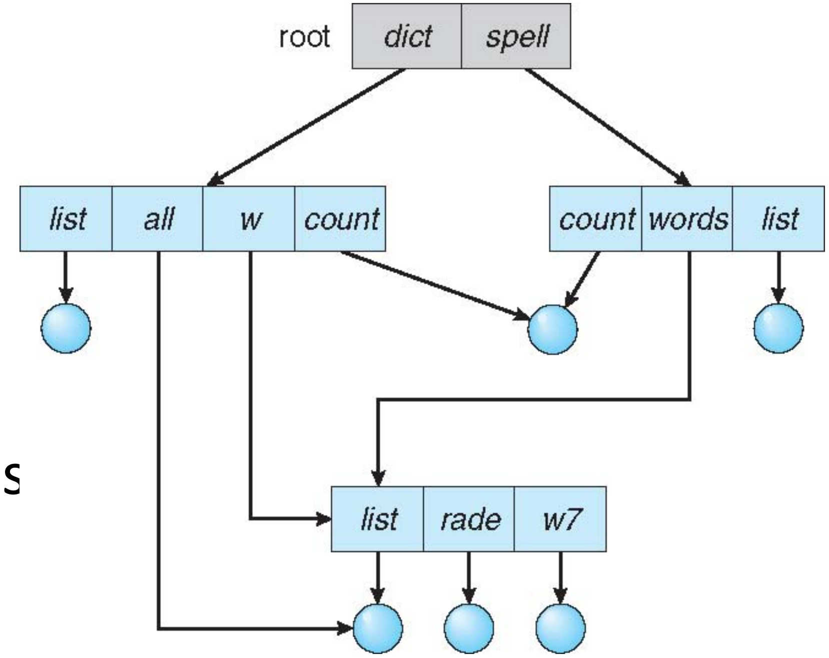
Directory Organization

The directory is organized logically to obtain

- Efficiency – locating a file quickly
- Naming – convenient to users
  - Two users can have same name for different files
  - The same file can have several different names
- Grouping – logical grouping of files by properties, (e.g., all Java programs, all games, …)

Directory Organization

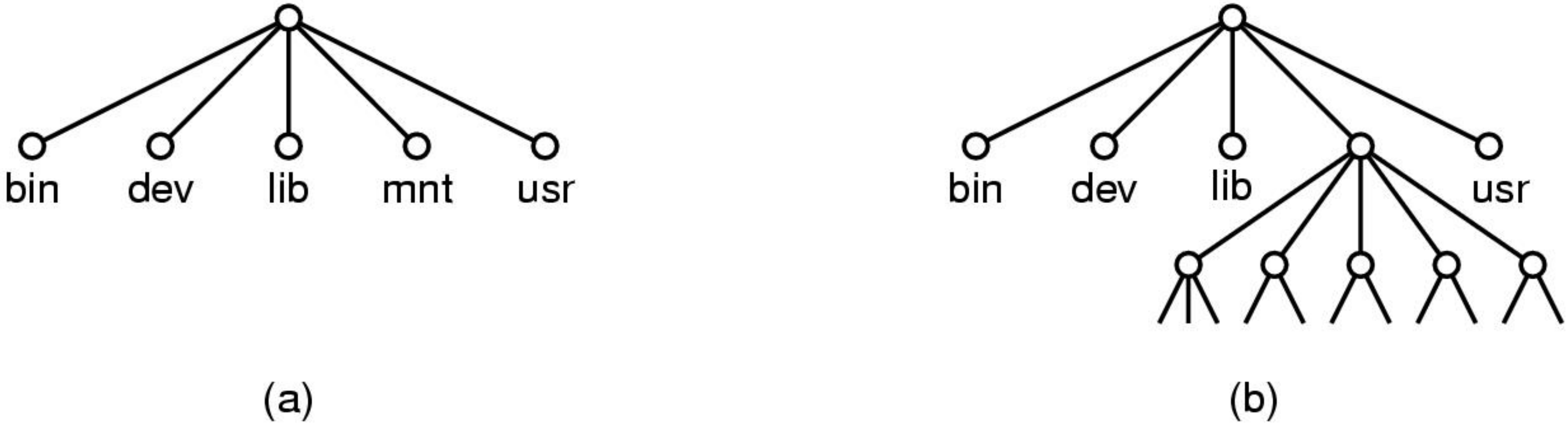
- Single level directory
- Two-level directory
- Tree-structured directories:
  - efficient grouping, searching,
  - absolute or relative path names
- Acyclic graph directories
  - Shared sub-directory, files



Colorado State University

File System Mounting

- A file system must be **mounted** before it can be accessed
- A unmounted file system is mounted at a **mount point**
- **Merges the file system**



Colorado State University

File Sharing

- Sharing of files on multi-user systems is desirable
- Sharing may be done through a **protection** scheme
- On distributed systems, files may be shared across a network
- Network File System (NFS) is a common distributed file-sharing method
- If multi-user system
  - **User IDs** identify users, allowing permissions and protections to be per-user
    Group IDs allow users to be in groups, permitting group access rights
  - Owner of a file / directory
  - Group of a file / directory

File Sharing – Remote File Systems

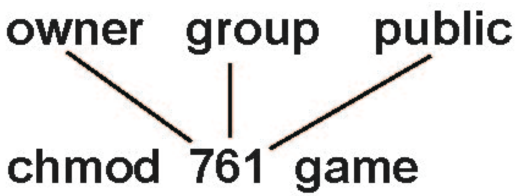
- Uses networking to allow file system access between systems
  - Manually via programs like FTP
  - Automatically, seamlessly using **distributed file systems**
  - Semi automatically via the **world wide web**
- **Client-server** model allows clients to mount remote file systems from servers
  - Server can serve multiple clients
  - Client and user-on-client identification is insecure or complicated
  - **NFS** is standard UNIX client-server file sharing protocol
  - **CIFS** is standard Windows protocol
  - Standard operating system file calls are translated into remote calls
- Distributed Information Systems (**distributed naming services**) such as LDAP, DNS, NIS, Active Directory implement unified access to information needed for remote computing

Protection: Access Lists and Groups

- Mode of access: read, write, execute
- Three classes of users on Unix / Linux

a) **owner access**	7	⇒	RWX 1 1 1
b) **group access**	6	⇒	RWX 1 1 0
c) **public access**	1	⇒	RWX 0 0 1

- Ask manager to create a group (unique name), say G, and add some users to the group.
- For a particular file (say *game*) or subdirectory, define an appropriate access.



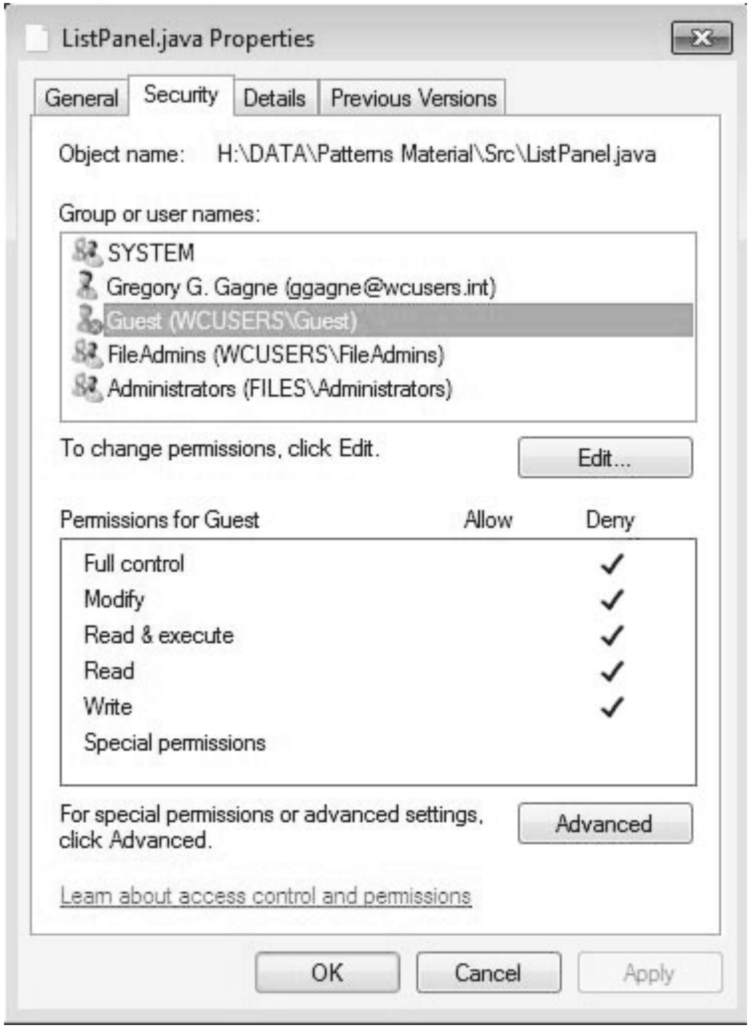
```
chmod 761 game
```

Attach a group to a file

```
chgrp      G      game
```

Quiz

Windows 7 Access-Control List Management



Object name: H:\DATA\Patterns Material\Src\ListPanel.java

Group or user names:

- SYSTEM
- Gregory G. Gagne (ggagne@wcusers.int)
- Guest (WCUSERS\Guest)
- FileAdmins (WCUSERS\FileAdmins)
- Administrators (FILES\Administrators)

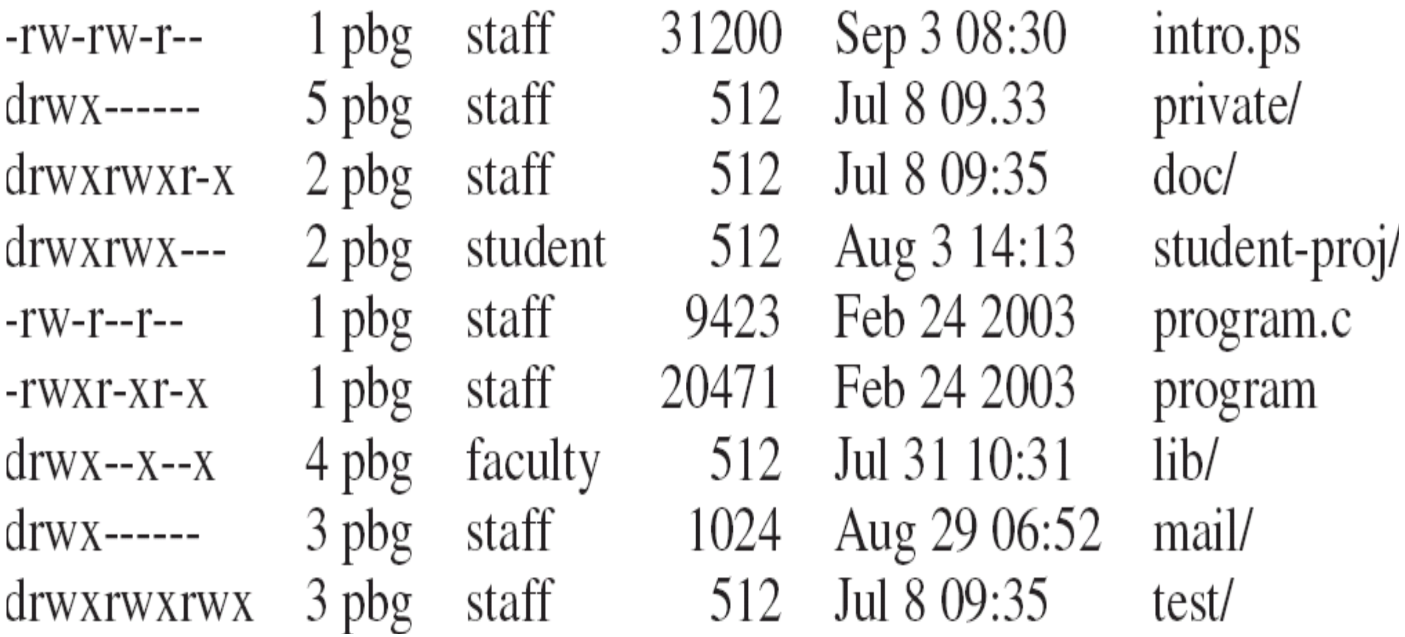
To change permissions, click Edit.

Permissions for Guest	Allow	Deny
Full control		✓
Modify		✓
Read & execute		✓
Read		✓
Write		✓
Special permissions		

For special permissions or advanced settings, click Advanced.

Learn about access control and permissions

A Sample UNIX Directory Listing



```
-rw-rw-r--   1 pbg  staff    31200  Sep 3 08:30    intro.ps
drwx------   5 pbg  staff      512  Jul 8 09.33    private/
drwxrwxr-x   2 pbg  staff      512  Jul 8 09:35    doc/
drwxrwx---   2 pbg  student    512  Aug 3 14:13    student-proj/
-rw-r--r--   1 pbg  staff     9423  Feb 24 2003    program.c
-rwxr-xr-x   1 pbg  staff    20471  Feb 24 2003    program
drwx--x--x   4 pbg  faculty    512  Jul 31 10:31   lib/
drwx------   3 pbg  staff     1024  Aug 29 06:52   mail/
drwxrwxrwx   3 pbg  staff      512  Jul 8 09:35    test/
```

Colorado State University

CS370 Operating Systems

Colorado State University

Yashwant K Malaiya

Fall 2016 Lecture 38+

File-system Implementation

Slides based on

- Text by Silberschatz, Galvin, Gagne
- Various sources

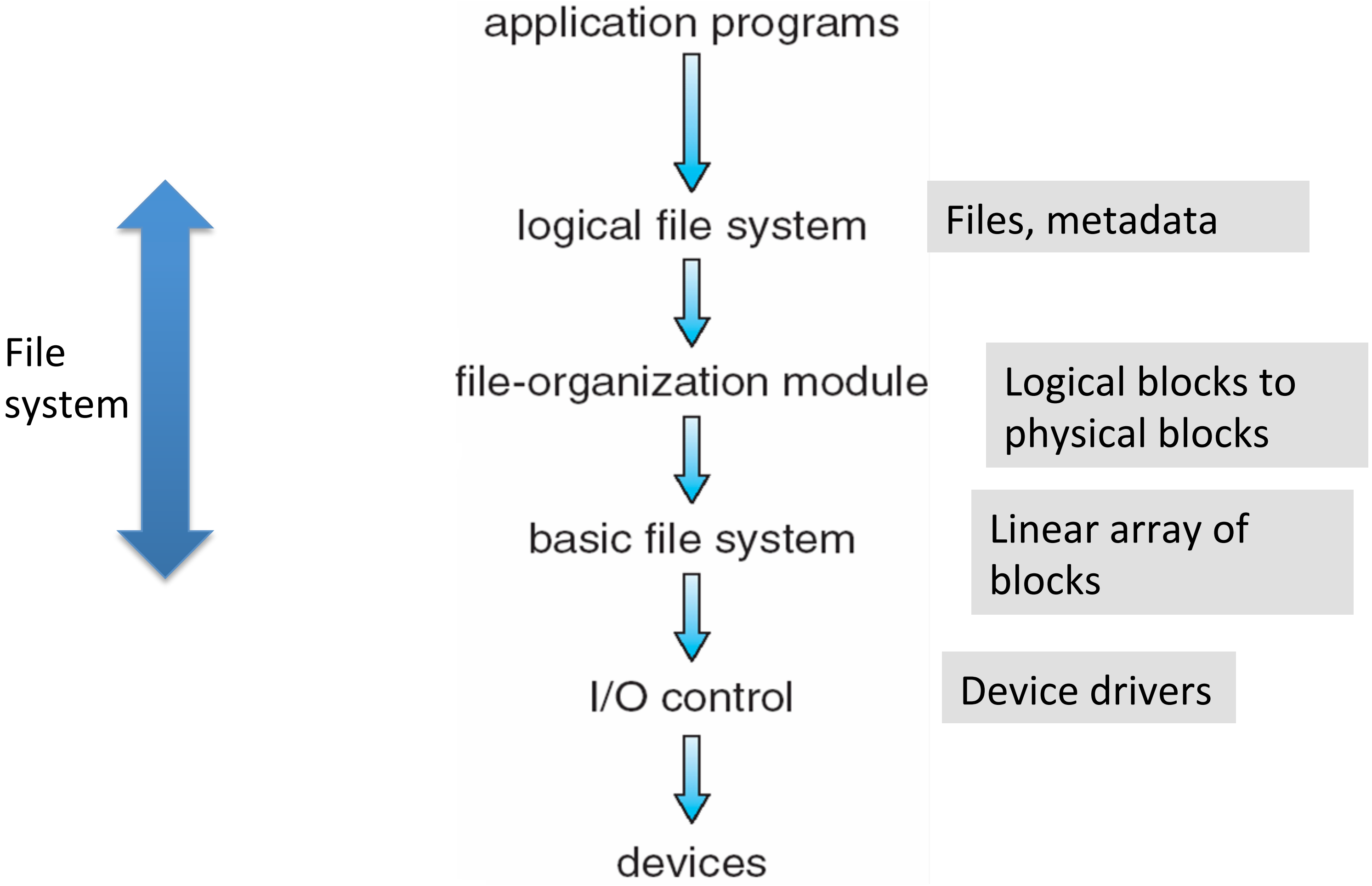
Chap 12: File System Implementation

- File-System Structure
- File-System Implementation
- Directory Implementation
- Allocation Methods
- Free-Space Management
- Efficiency and Performance
- Recovery

File-System Structure

- File structure
  - Logical storage unit
  - Collection of related information
- **File system** resides on secondary storage (disks/SSD)
  - Provides user interface to storage, mapping logical to physical
  - Provides efficient and convenient access to disk by allowing data to be stored, located retrieved easily
  - Can be on other media (flash etc), with different file system
- Disk provides in-place rewrite and random access
  - I/O transfers performed in **blocks** of **sectors** (usually 512 bytes)
- **File control block** – storage structure -information about a file (inode in Linux) inc location of data
- **Device driver** controls the physical device

Layered File System



File system

application programs

logical file system

file-organization module

basic file system

I/O control

devices

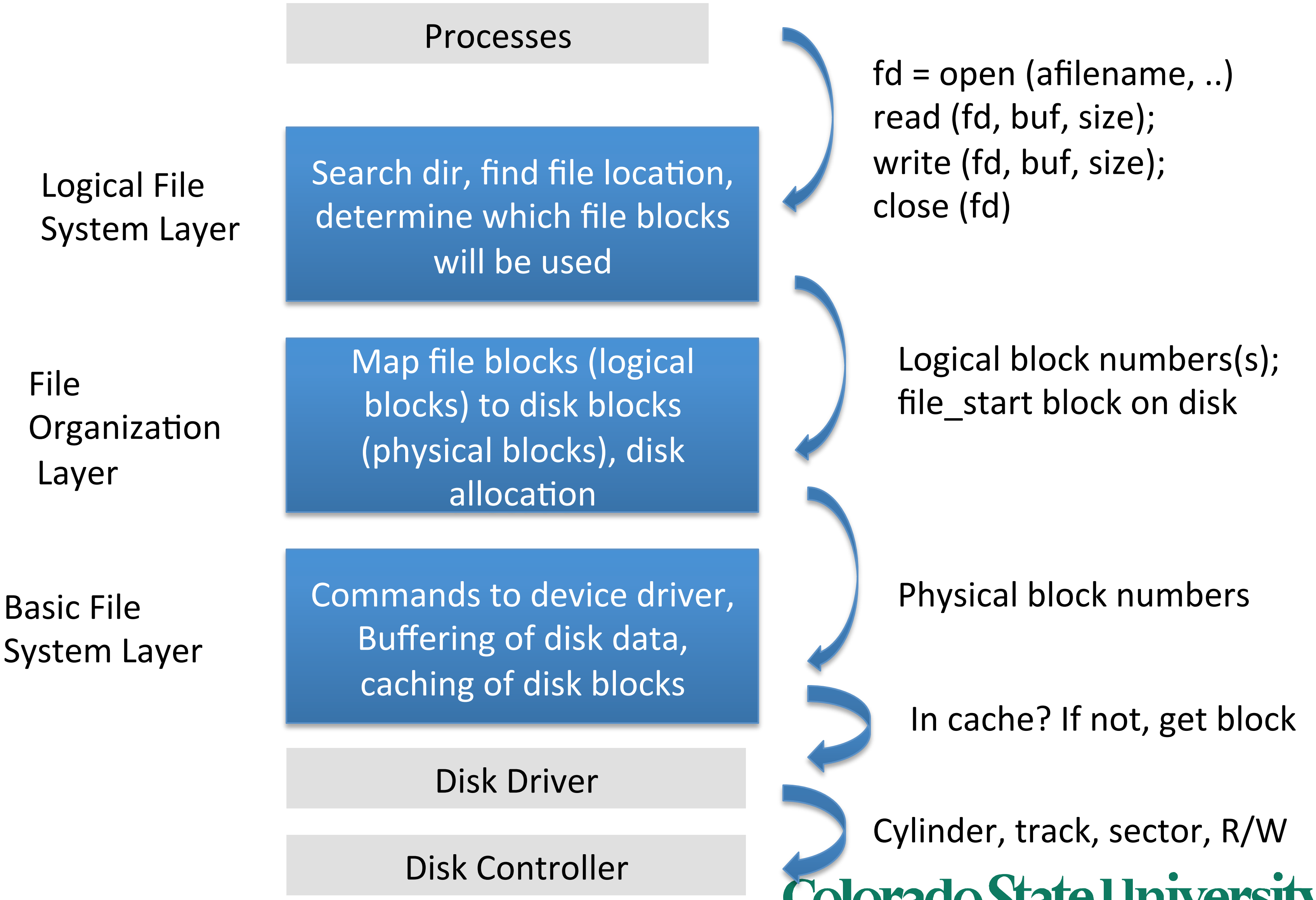
Files, metadata

Logical blocks to physical blocks

Linear array of blocks

Device drivers

Layered File System



Processes

fd = open (afilename, ..)
read (fd, buf, size);
write (fd, buf, size);
close (fd)

Logical File System Layer

Search dir, find file location, determine which file blocks will be used

Logical block numbers(s);
file_start block on disk

File Organization Layer

Map file blocks (logical blocks) to disk blocks (physical blocks), disk allocation

Physical block numbers

Basic File System Layer

Commands to device driver, Buffering of disk data, caching of disk blocks

In cache? If not, get block

Disk Driver

Cylinder, track, sector, R/W

Disk Controller

Colorado State University

File System Layers (from bottom)

- **Device drivers** manage I/O devices at the I/O control layer
  - Given commands like “read drive1, cylinder 72, track 2, sector 10, into memory location 1060” outputs low-level hardware specific commands to hardware controller
- **Basic file system** given command like “retrieve block 123” translates to device driver
  - Also manages memory buffers and caches (allocation, freeing, replacement)
    - Buffers hold *data in transit*
    - Caches hold *frequently used data*
- **File organization module** understands files, logical address, and physical blocks

  - Translates logical block # to physical block #

  - Manages free space, disk allocation

File System Layers (Cont.)

- **Logical file system** manages metadata information
  - Translates file name into file number, file handle, location by maintaining *file control blocks* (**inodes** in UNIX)
  - Directory management
  - Protection

File Systems

- Many file systems, sometimes several within an operating system
  - Each with its own format
    - Windows has FAT (1977), FAT32 (1996), NTFS (1993)
    - Linux has more than 40 types, with **extended file system** (1992) ext2 (1993), ext3 (2001), ext4 (2008);
    - plus distributed file systems
    - floppy, CD, DVD Blu-ray
  - New ones still arriving – ZFS, GoogleFS, Oracle ASM, FUSE, xFAT