國立陽明交通大學
NATIONAL YANG MING CHIAO TUNG UNIVERSITY

Institute of Artificial Intelligence Innovation
Department of Computer Science

Operating System

Lecture 09: File System Interface

Shuo-Han Chen 陳碩漢
shch@nycu.edu.tw

Wed. 10:10 - 12:00 EC115 +
Fri. 11:10 – 12:00 Online

Course Schedule

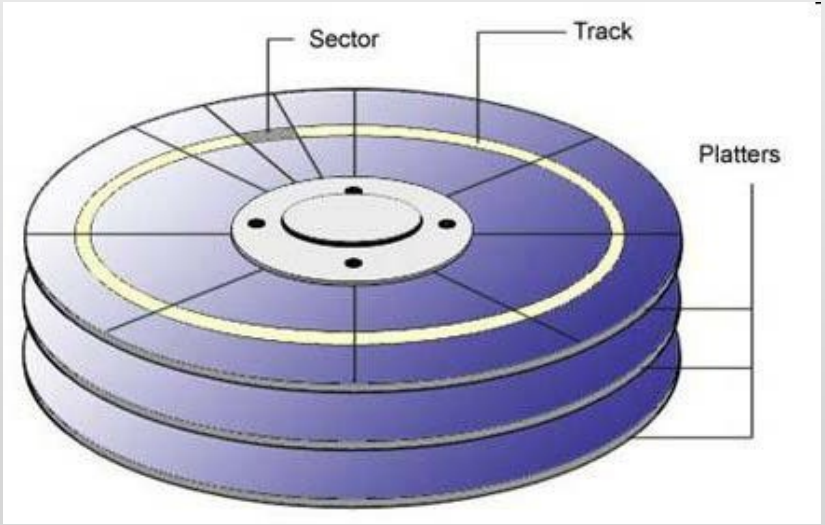
W	Date	Lecture	Online	Homework
1	Sept. 4	Lec00: Couse Overview & Historical Prospective		
2	Sept. 11	Lec01: Introduction	V	
3	Sept. 18	Lec02: OS Structure	V	HW01 Due 10/5
4	Sept. 25	Lec03: Processes Concept	X	
5	Oct. 2	Typhoon – No class	V	
6	Oct. 9	Lec07: Memory Management	V	
7	Oct. 16	Lec08: Virtual Memory Management	V	HW02 Due 11/2
8	Oct. 23	Lec04: Multithreaded Programming	V	
9	Oct. 30	Midterm Exam		
10	Nov. 6	Lec05: Process Scheduling	V	Let's take a breath
11	Nov. 13	Lec06: Process Synchronization & Deadlocks	X	HW03
12	Nov. 20	School Event – No class	V	
13	Nov. 27	Lec09: File System Interface	V	
14	Dec. 4	Lec10: File System Implementation	V	HW04
15	Dec. 11	Lec11: Mass Storage System & Lec12: IO Systems	V	
16	Dec. 18	School Final Exam		

Overview

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

File Concept

- File: a logical storage unit created by OS
  - v.s. physical storage unit in disk (sector, track)
- File attributes
  - Identifier: non-human-readable name
  - Name
  - Type
  - Location
  - Size
  - Protection
  - Last-access time, Last-updated time

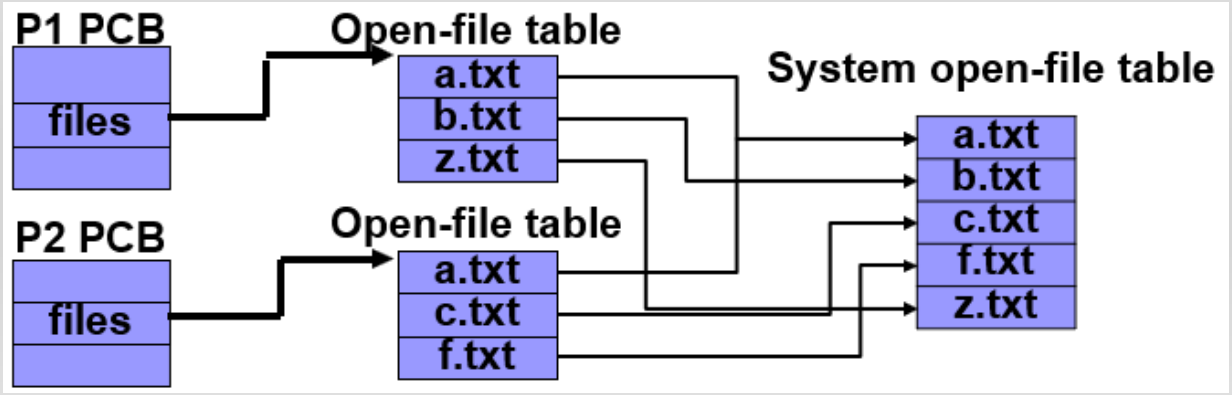


File Operations

- File operations include
  - Creating a file
  - Writing a file
  - Reading a file
  - Repositioning within a file (i.e. file seek)
  - Deleting a file
  - Truncating a file
- Process: open-file table
- OS: system-wide table

Open-File Tables

- Per-process table
  - Tracking all files opened by this process
  - Current file pointer for each opened file
  - Access rights and accounting information
- System-wide table
  - Each entry in the per-process table points to this table
  - Process-independent information such as disk location, access dates, file size
  - Open count



Open File Attributes

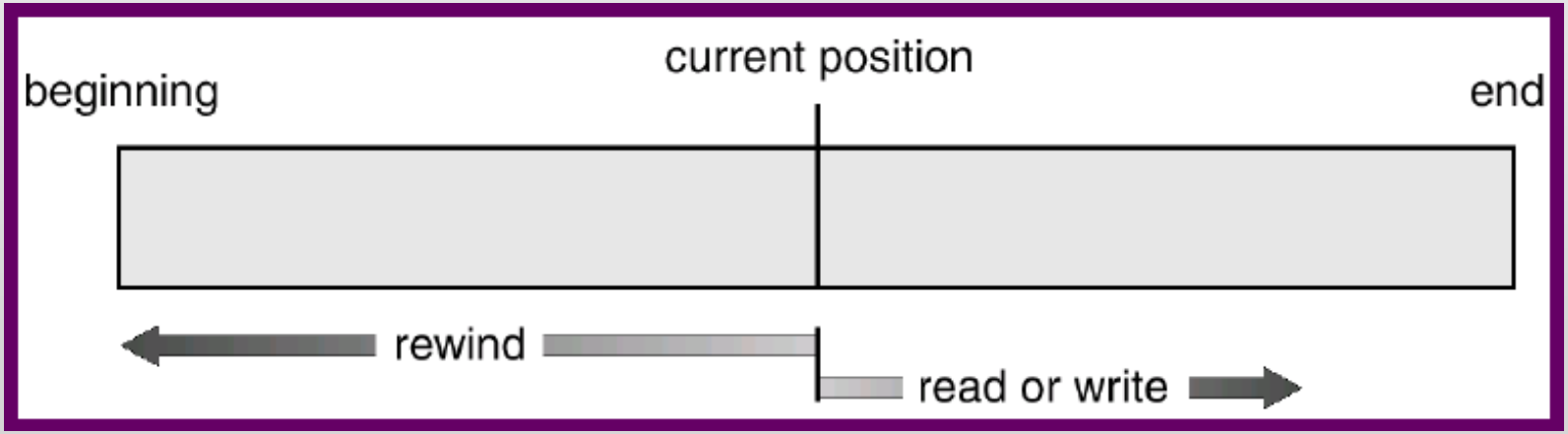
- Open-file attributes (metadata)
  - File pointer (per-process)
  - File open count (system table)
  - Disk location (system table)
  - Access rights (per-process)
- File types
  - .exe, .com, .obj, .cc, .mov, etc
  - Hint for OS to operate file in a reasonable way

file type	usual extension	function
executable	exe, com, bin or none	read 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, rrf, doc	various word-processor formats
library	lib, a, so, dll, mpeg, mov, rm	libraries of routines for programmers
print or view	arc, zip, tar	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	binary file containing audio or A/V information

Access Method

Access Methods

- Sequential access
  - Read/write next (block)
  - Reset: repositioning the file pointer to the beginning
  - Skip/rewind n records



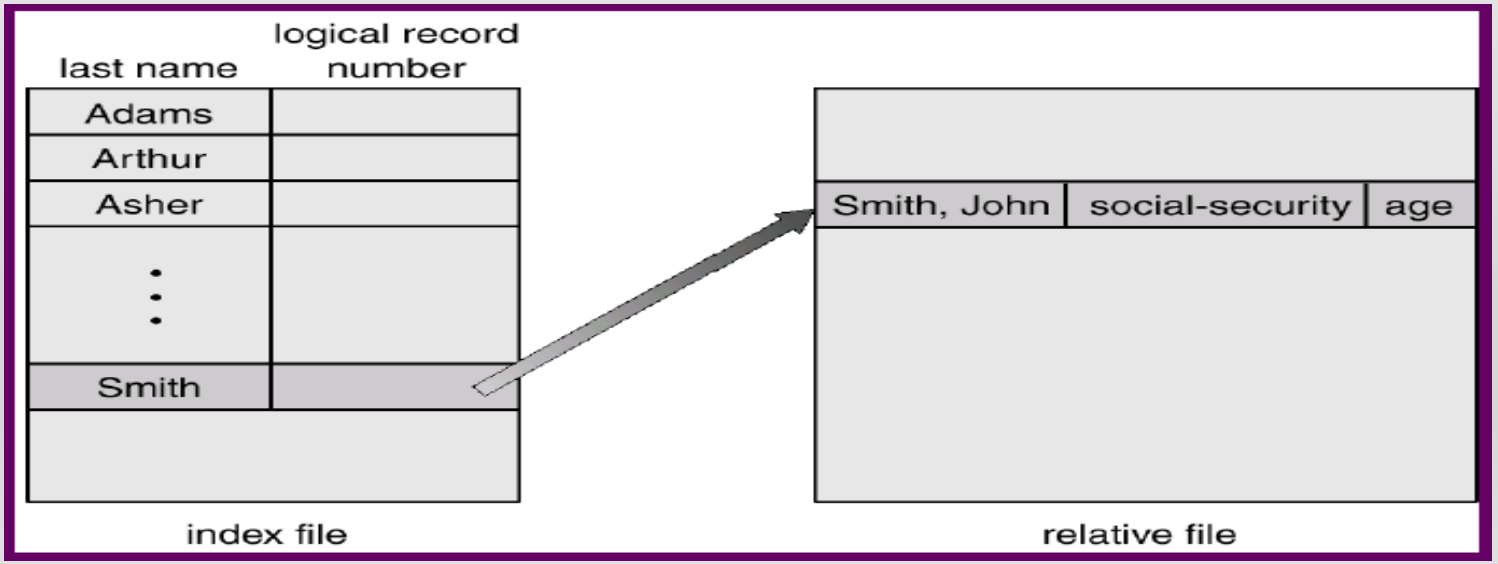
Access Methods

- Direct (relative) access
  - Access an element at an arbitrary position in a sequence
  - File operations include the block # as parameter
  - Often use random access to refer the access pattern from direct access

sequential access	implementation for direct access
reset	*cp* = 0;
read next	*read cp*; *cp* = *cp*+1;
write next	*write cp*; *cp* = *cp*+1;

Index Access Methods

- Index: contains pointers to blocks of a file
- To find a record in a file:
  - search the index file -> find the pointer
  - use the pointer to directly access the record
- With a large file -> index could become too large



Review Slides (I)

- File vs. Sector, Track
- Open-file (in-memory) attributes
  - Per-process, system-wide?
- File-access methods?
  - Sequential access
  - Direct access
  - Index access

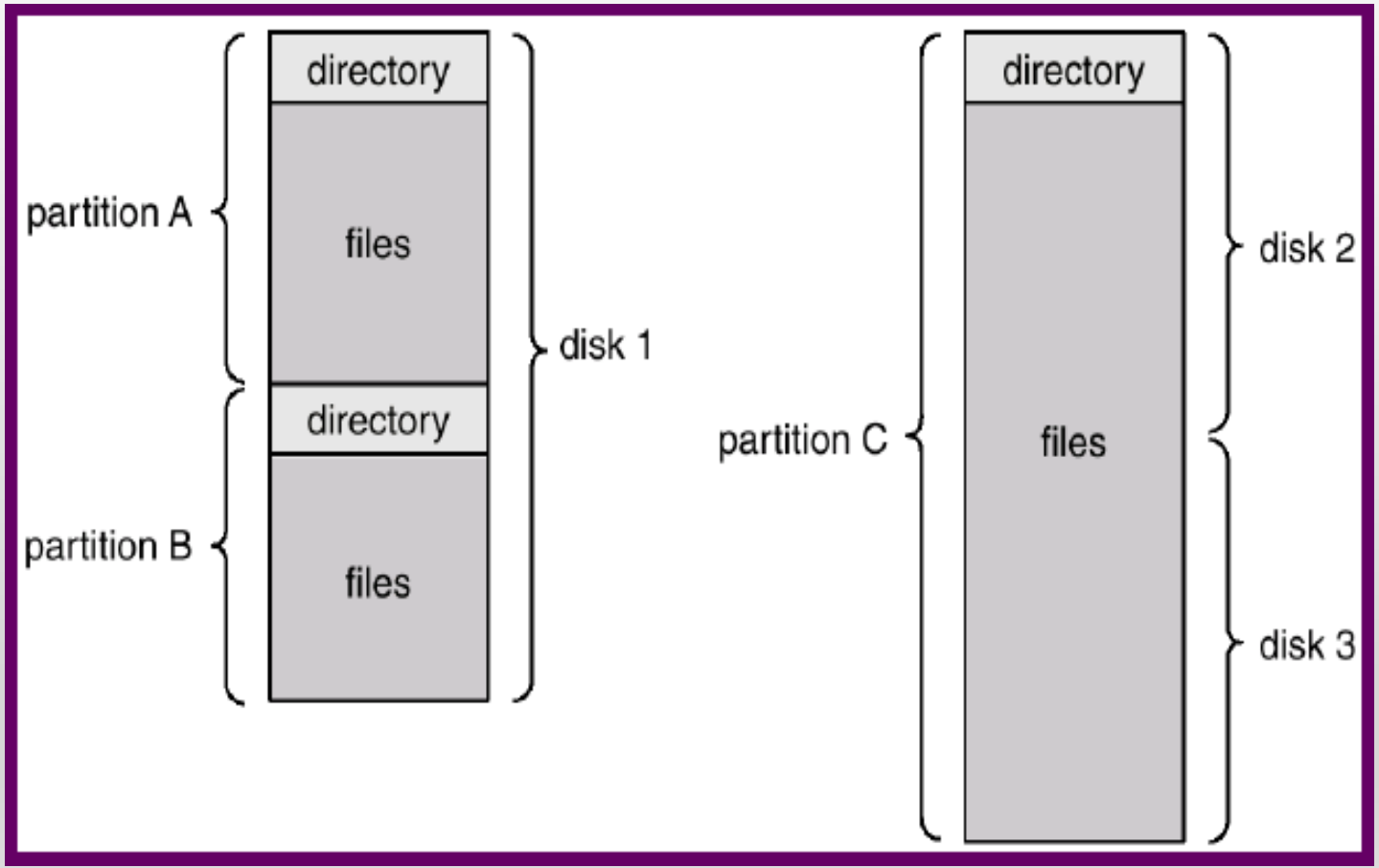
Directory Structure

Partition, Volume & Directory

- A partition (formatted or raw)
  - raw partition (no file system): UNIX swap space, database
  - Formatted partition with file system is called volume
  - a partition can be a portion of a disk or group of multiple disks (distributed file system)
  - Some storage devices (e.g.: floppy disk) does not and cannot have partition

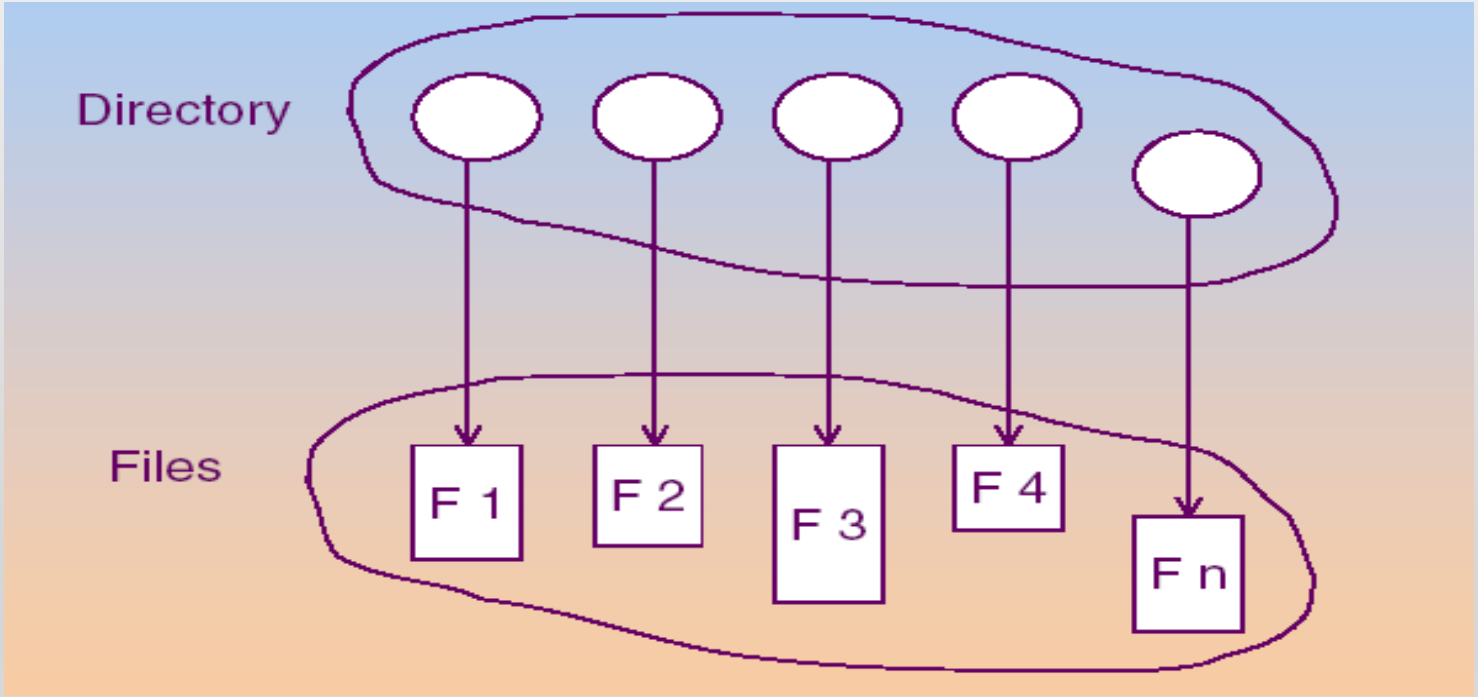
- Directories are used by file system to store the information about the files in the partition

File-System Organization



Directory vs. File

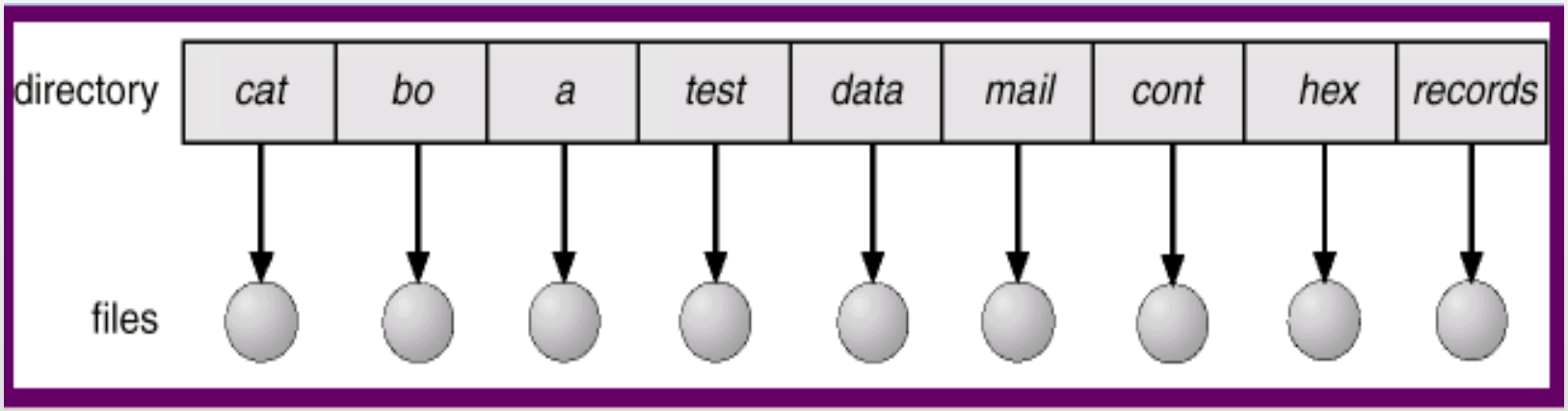
- Directory: A collection of nodes containing information about all files
  - Both the directory structure and the files reside on disk



Directory Operations

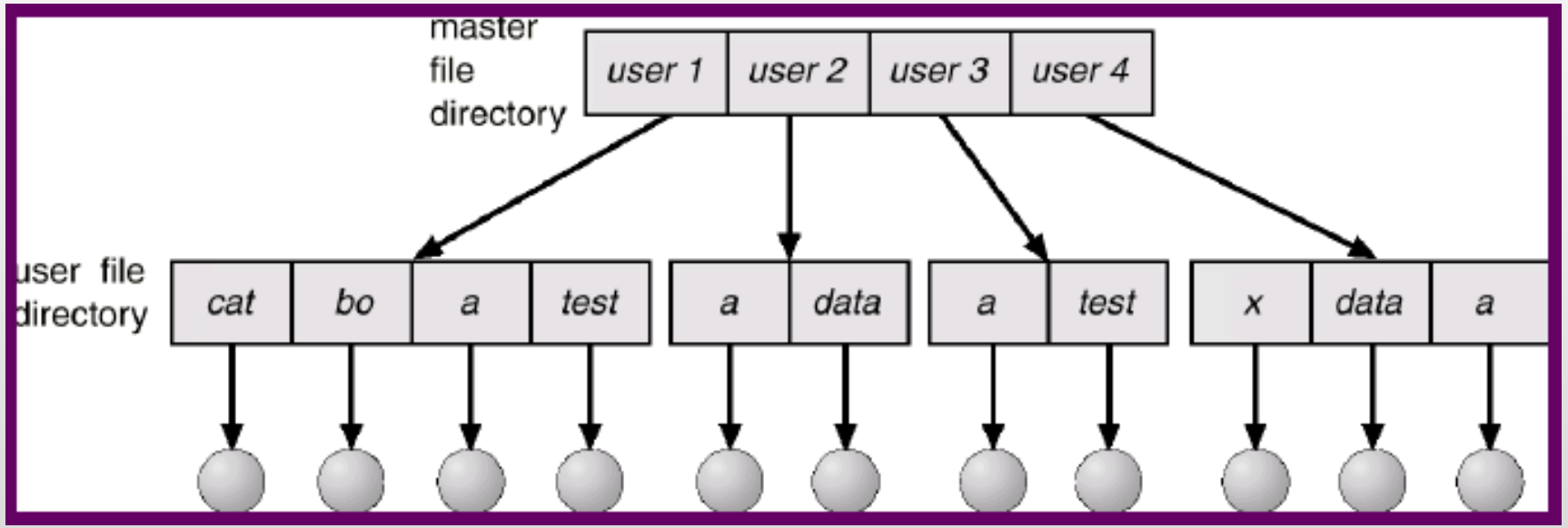
- Search for a file
- Create a file
- Delete a file
- List a directory
- Rename a file
- Traverse the file system

Single-Level Directory



- All files in one directory
  - Filename has to be unique
  - Poor efficiency in locating a file as number of files increases

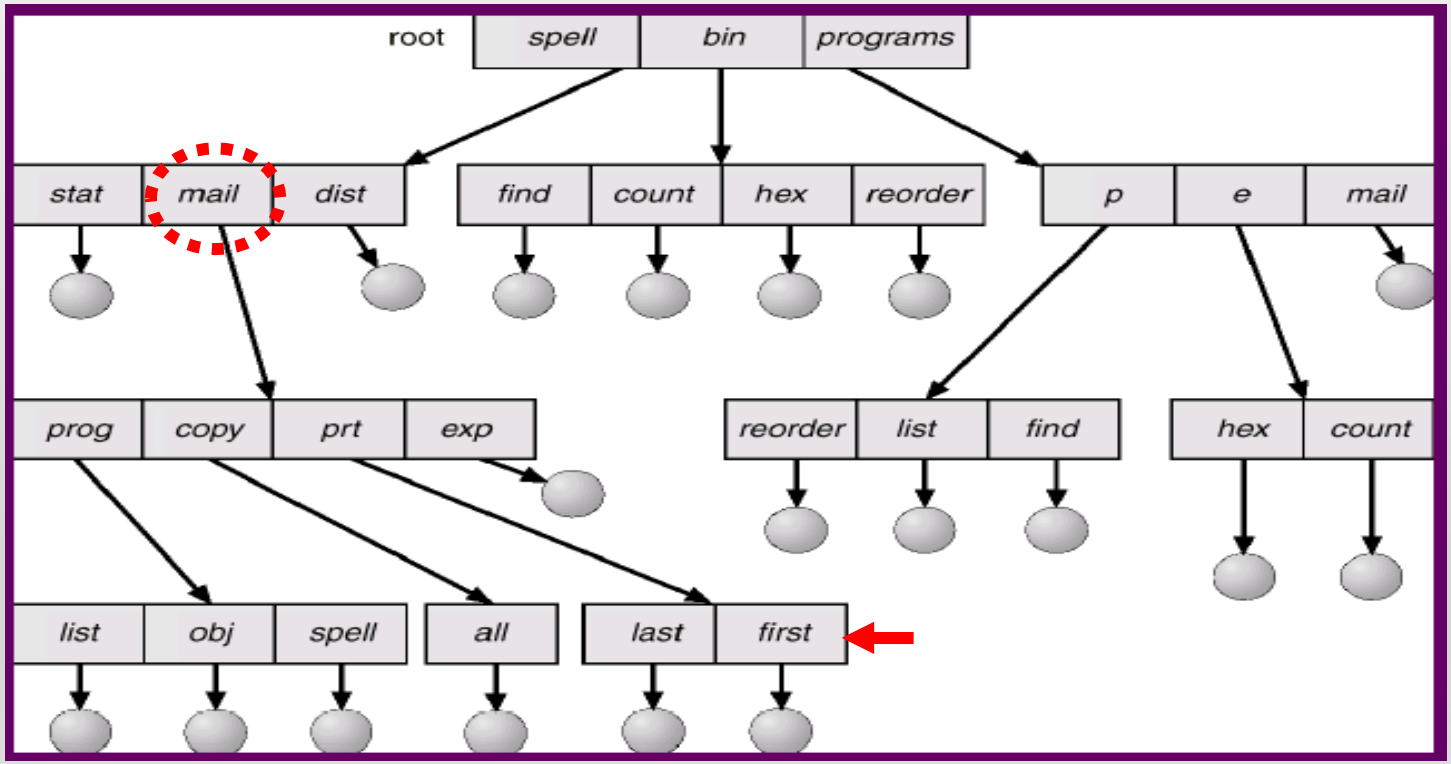
Two-Level Directory



- a separate dir for each user
- path = user name + file name
- single-level dir problems still exists per user

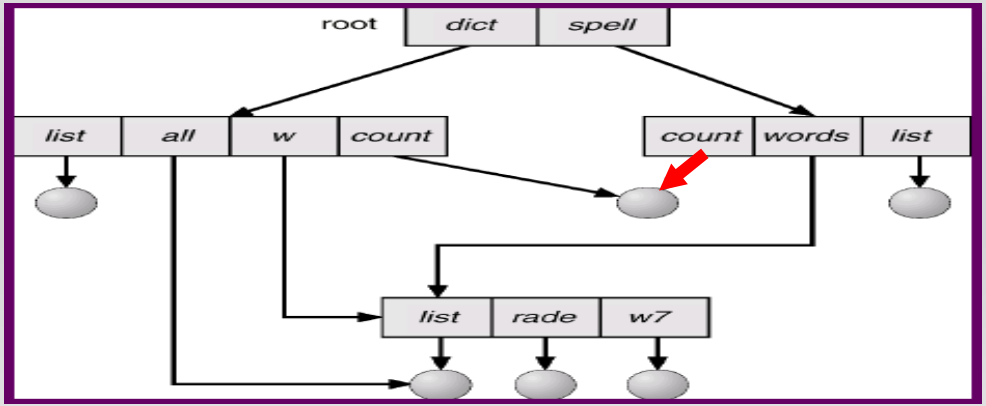
Tree-Structured Directory

- Absolute path: starting from the root
- Relative path: starting from a directory

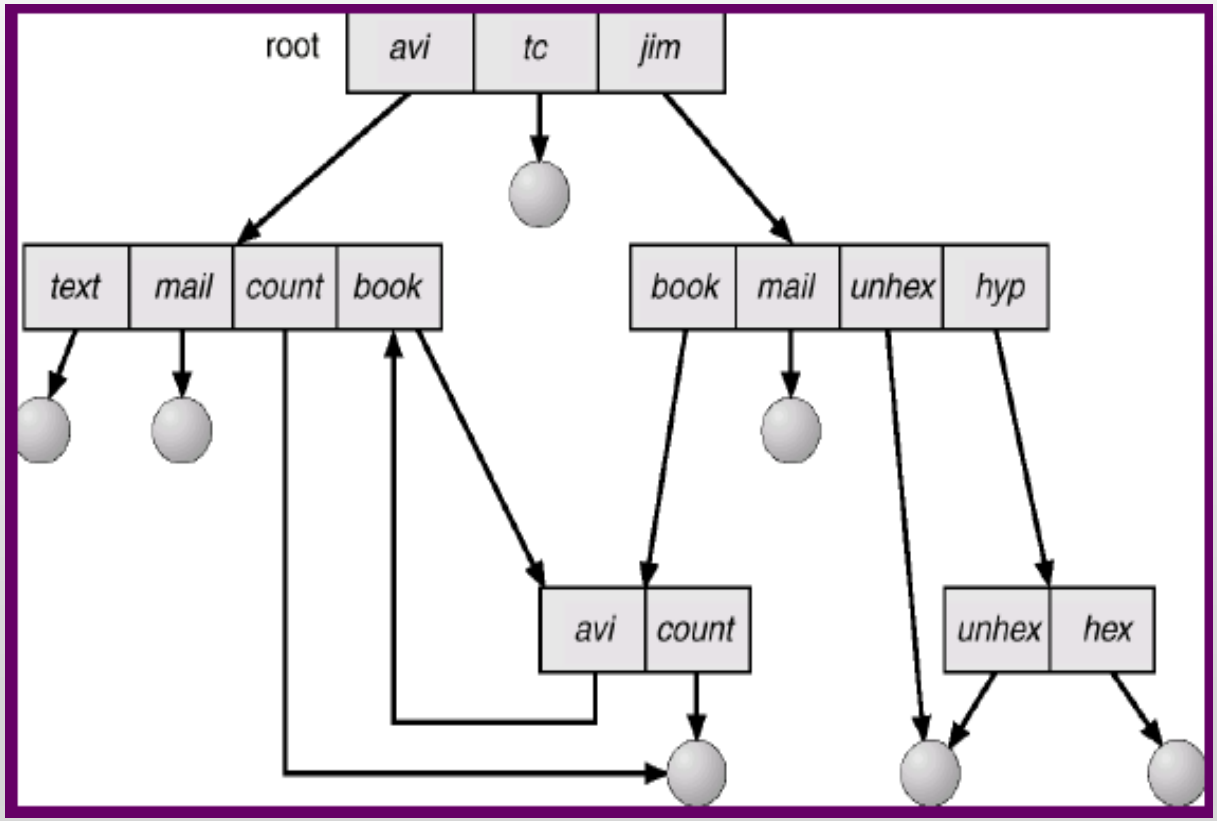


Acyclic-Graph Directory

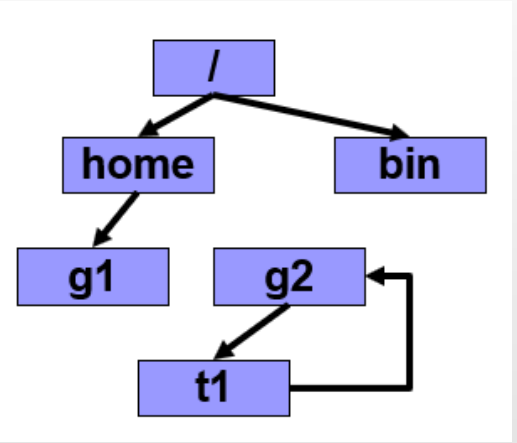
- Use links to share files or directories
  - UNIX-like: symbolic link (ln -s /spell/count /dict/count)
- A file can have multiple absolute paths
- When does a file actually get deleted?
  - deleting the link but not the file
  - deleting the file but leaves the link -> dangling pointer
- deleting the file when reference counters is 0



General-Graph Directory



General-Graph Directory



- May contain cycles
  - Reference count does not work any more
  - E.g. self-referencing file
- How can we deal with cycles?
  - Garbage collection
    - First pass traverses the entire graph and marks accessible files or directories
    - Second pass collect and free everything that is un-marked
    - Poor performance on millions of files …
  - Use cycle-detection algorithm when a link is created

Review Slides (II)

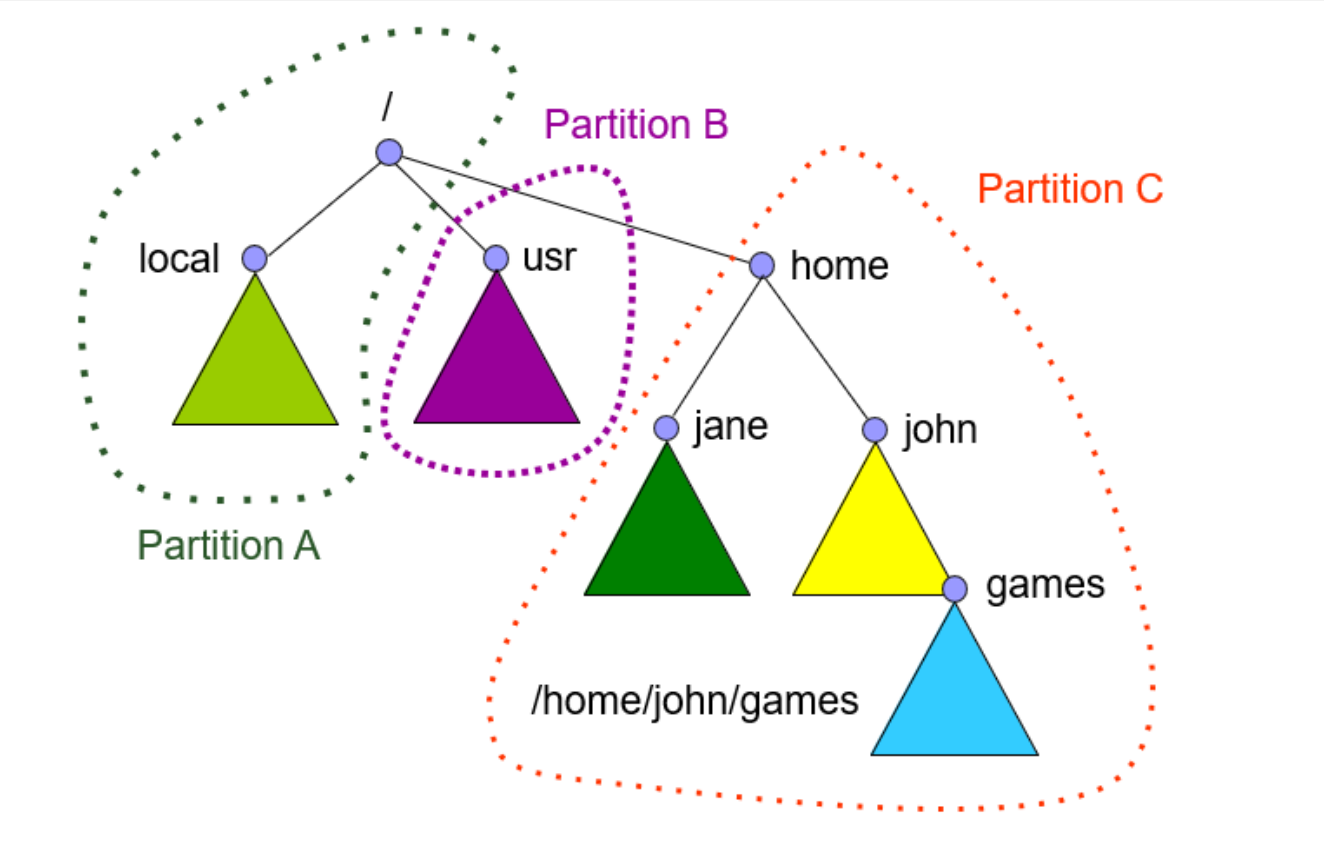
- Directory structure: pros & cons
  - One-level directory
  - Two-level directory
  - Tree-structured directory
  - Acyclic-graph directory
  - General-graph directory

File-System Mounting & File Sharing

File System Mounting

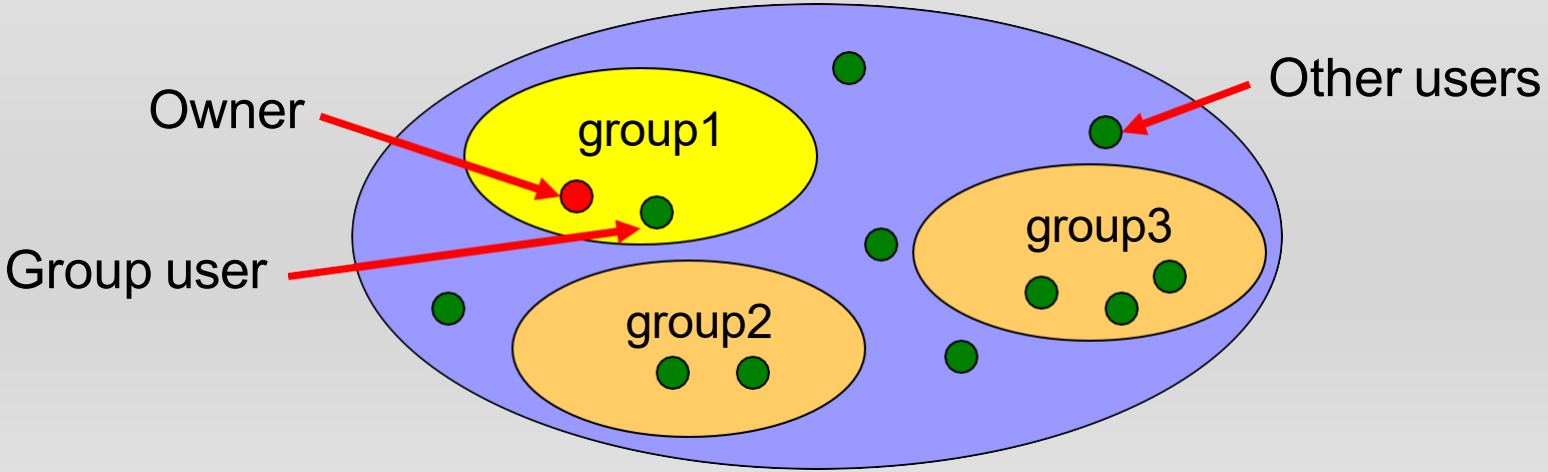
- A file system must be mounted before it can be accessed
- Mount point: the root path that a FS will be mounted to
- Mount timing:
  - boot time
  - automatically at run-time
  - manually at run-time

File System Mounting Example



File Sharing on Multiple Users

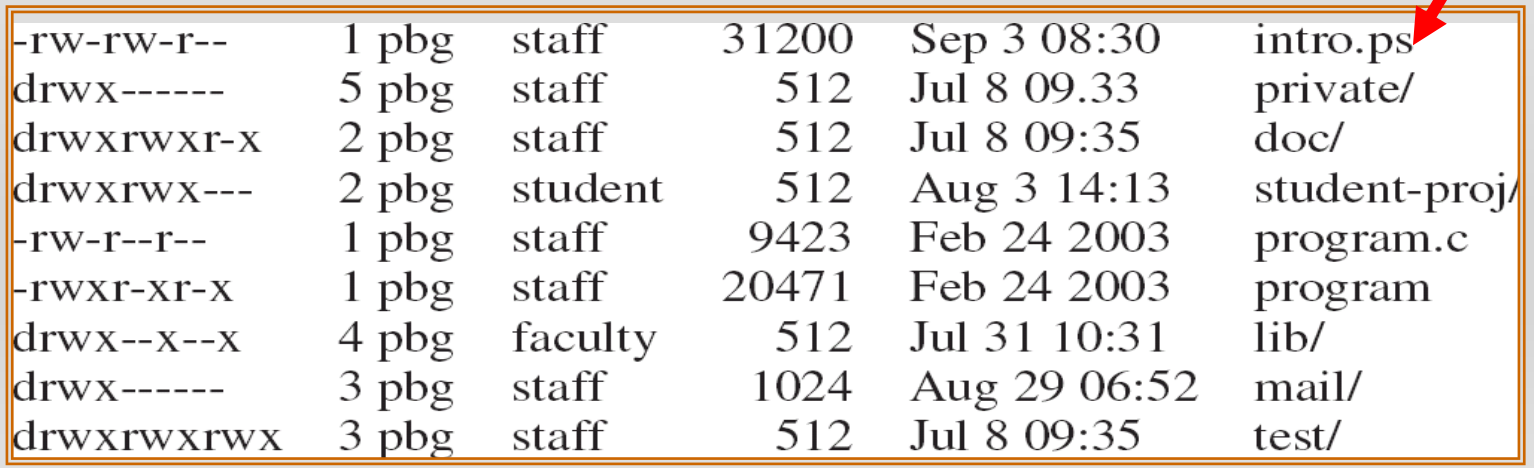
- Each user: (userID, groupID)
  - ID is associated with every ops/process/thread the user issues
- Each file has 3 sets of attributes
  - owner, group, others
- Owner attributes describe the privileges for the owner of the file
  - same for group/others attributes
  - group/others attributes are set by owner or root



Access-Control List

- We can create an access-control list (ACL) for each user
  - check requested file access against ACL
  - problem: unlimited # of users
- 3 classes of users -> 3 ACL (RWX) for each file
  - owner (e.g. 7 = RWX = 111)
  - group (e.g. 6 = RWX = 110)
  - public (others) (e.g. 4 = RWX = 100)

chmod 664 intro.ps



-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/

File Protection

- File owner/creator should be able to control
  - what can be done
  - by whom
  - -> Access control list (ACL)
- Files should be kept from
  - physical damage (reliability): i.e. RAID
  - improper access (protection): i.e. password

Review Slides (III)

- File system mounting point, timing?
- Access-control list? How does it function?

Reading Material & HW

- Chap 10
- Problems
  - 10.1: Consider a file system where a file can be deleted and its disk space reclaimed while links to that file still exist. What problems may occur if a new file is created in the same storage area or with the same absolute path name? How can these problems be avoided?
  - 10.4: Provide examples of applications that typically access files according to “sequential” and “random”.
  - 10.6: If the operating system knew that a certain application was going to access file data in a sequential manner, how could it exploit this information to improve performance?

Consistency Semantics

- When files are shared, ops from different users to the same file must be **synchronized**
- UNIX semantics
  - write is visible to all other users opening the same file
  - Open-file option: share the same file pointer
- Session semantics (AFS file system)
  - write is not visible to all other users
  - once a file is closed, changes are visible for sessions starting later -> current sessions do not see changes
- Immutable-Shared-Files semantics
  - once a file is declared shared, it cannot be modified

File Sharing on Remote File Systems

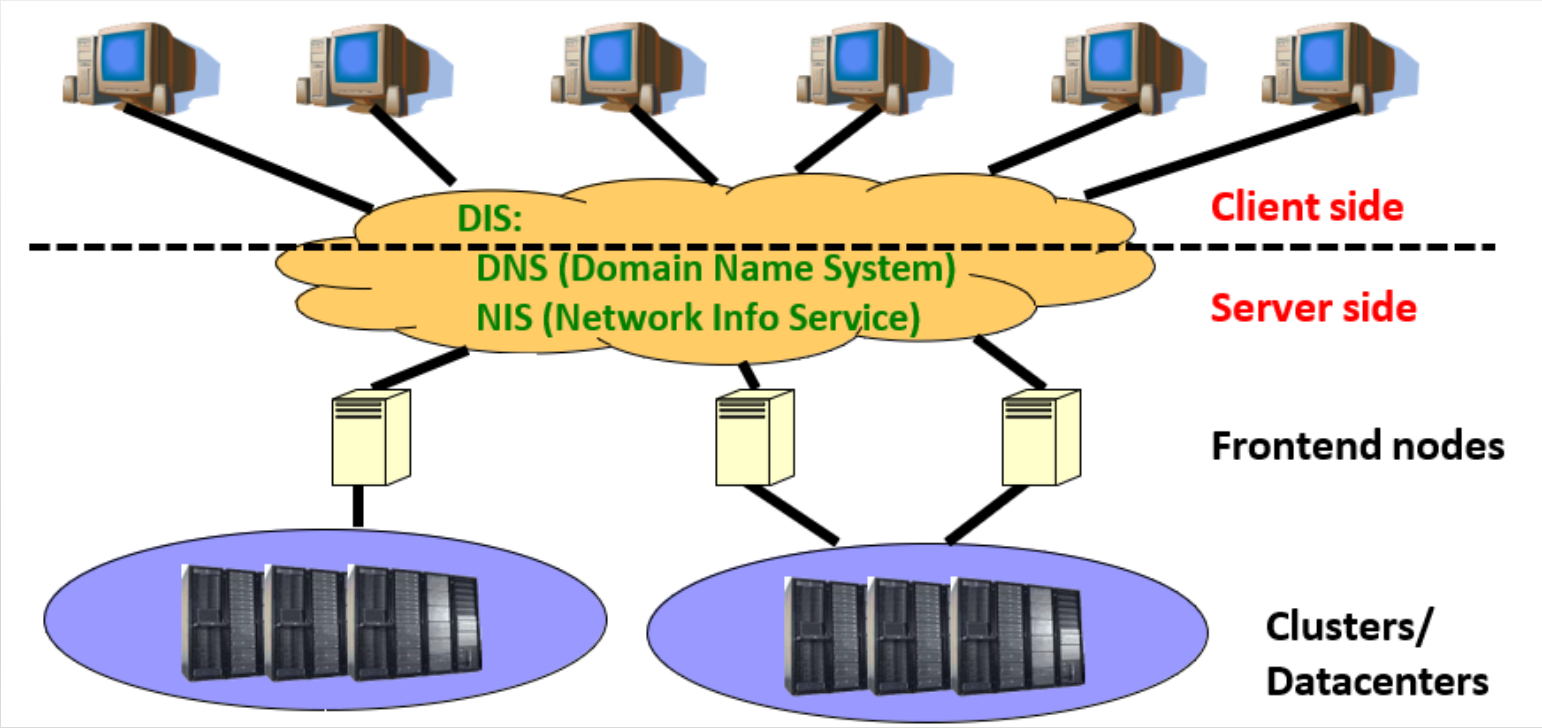
- Uses networking to allow file system access between systems
  - Manually via programs like FTP
  - Semi automatically via the world wide web
  - Automatically, seamlessly using distributed file systems

Client-server model

- Allows clients to mount remote file systems from servers
  - Sever: the machine that owns the files and serves multiple clients
  - Client: the machine that accesses remote files
  - Standard OS file calls are translated into remote calls
  - Client and user-on-client identification is insecure or complicated
- Example:
  - NFS (network file sysytem) for UNIX
  - CIFS (common interface file system) for Windows

Distributed Information Systems

- Distributed naming services
  - Provide unified access to the info for remote computing



Failure Modes

- Failures:
  - HW: disk, network cable, switch, server, etc.
  - SW: corruption or inconsistency of file, directory structure, etc.
- We need to recover:
  - Data: files, directory contents
  - Metadata: data and system management info.
- Stateful vs. Stateless communication protocol:
  - Stateless: treats each request as an independent transaction that is unrelated to any previous request (HTTP)
  - Stateful: info. maintained on both client and server is required

Q & A

Thank you for your attention