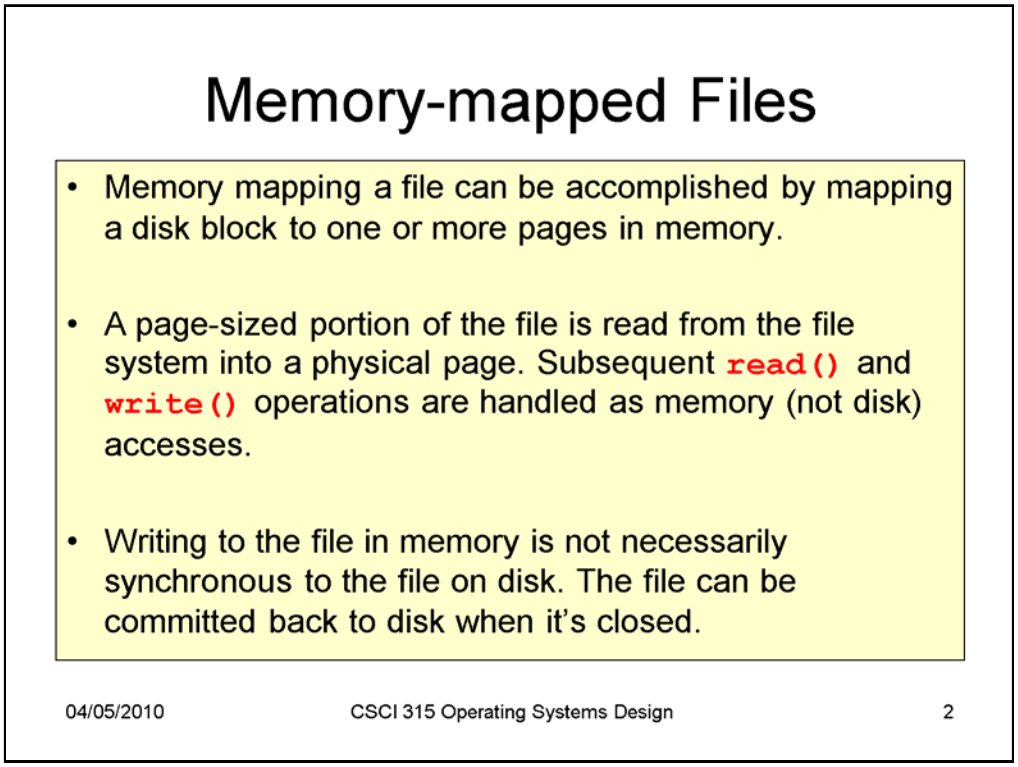
BUCKNELL UNIVERSITY
Computer Science

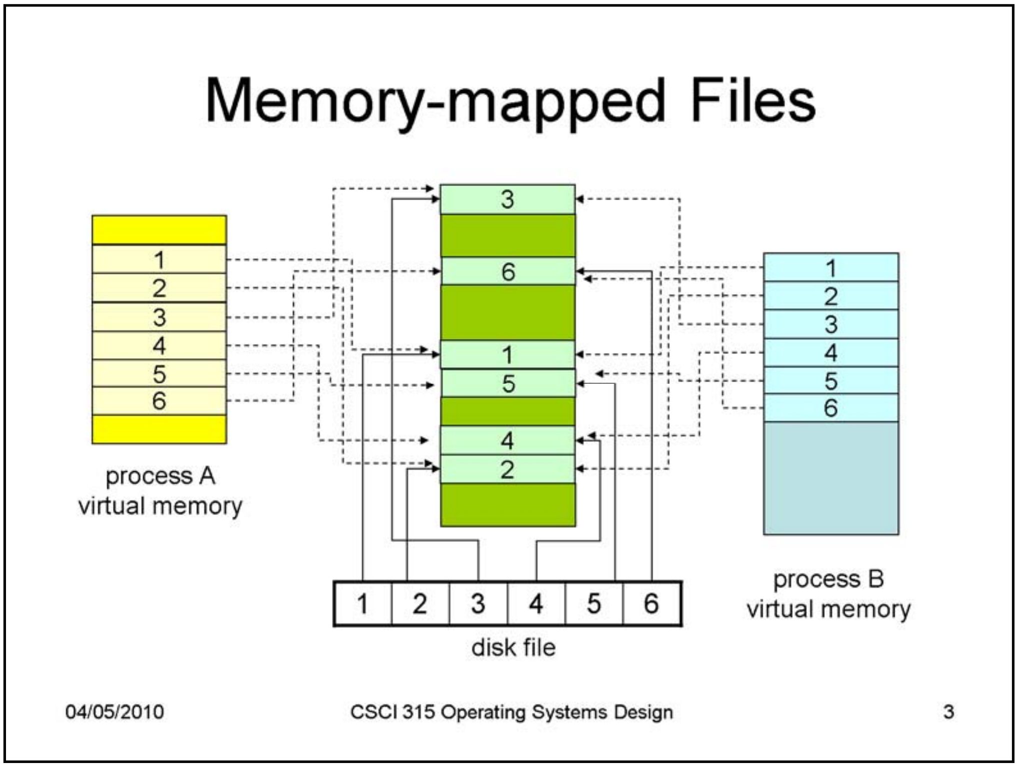
CSCI 315 Operating Systems Design

Virtual Memory Wrap-up;
File System Interface



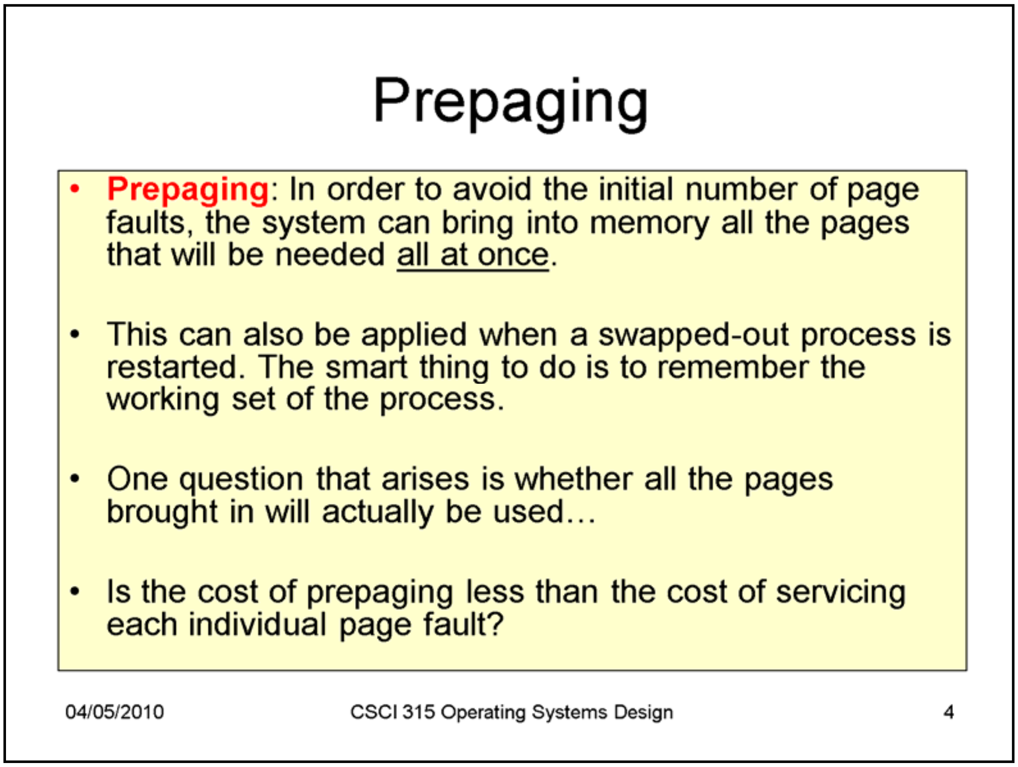
Memory-mapped Files

- Memory mapping a file can be accomplished by mapping a disk block to one or more pages in memory.
- A page-sized portion of the file is read from the file system into a physical page. Subsequent **`read()`** and **`write()`** operations are handled as memory (not disk) accesses.
- Writing to the file in memory is not necessarily synchronous to the file on disk. The file can be committed back to disk when it’s closed.



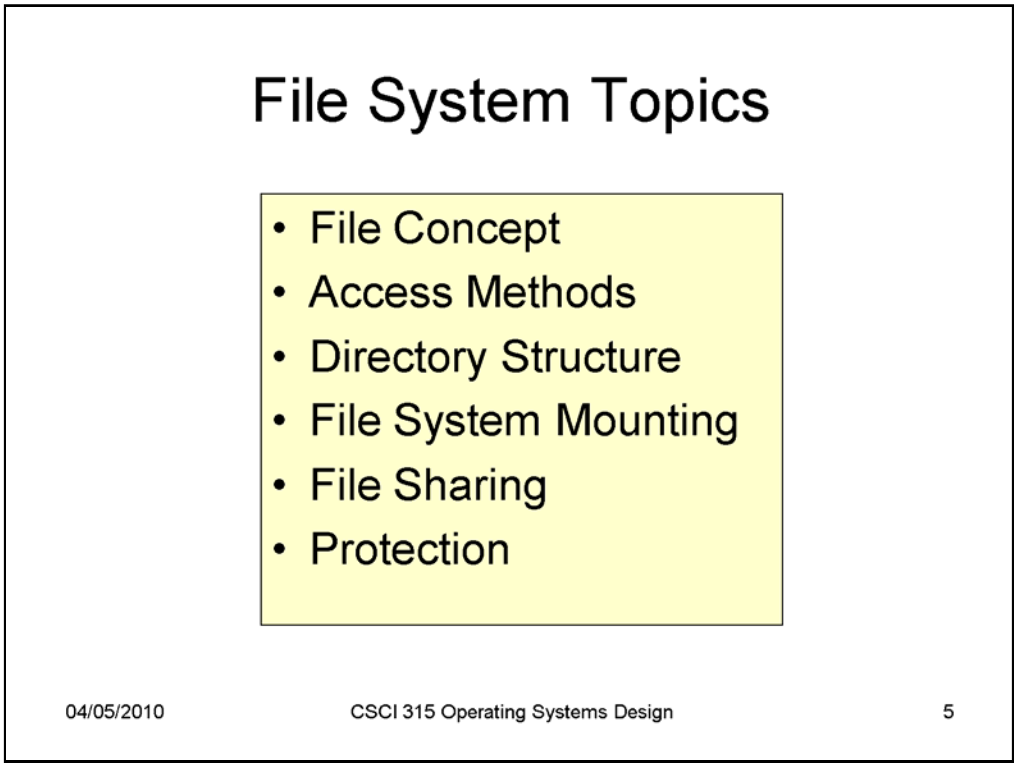
Memory-mapped Files

04/05/2010 CSCI 315 Operating Systems Design 3



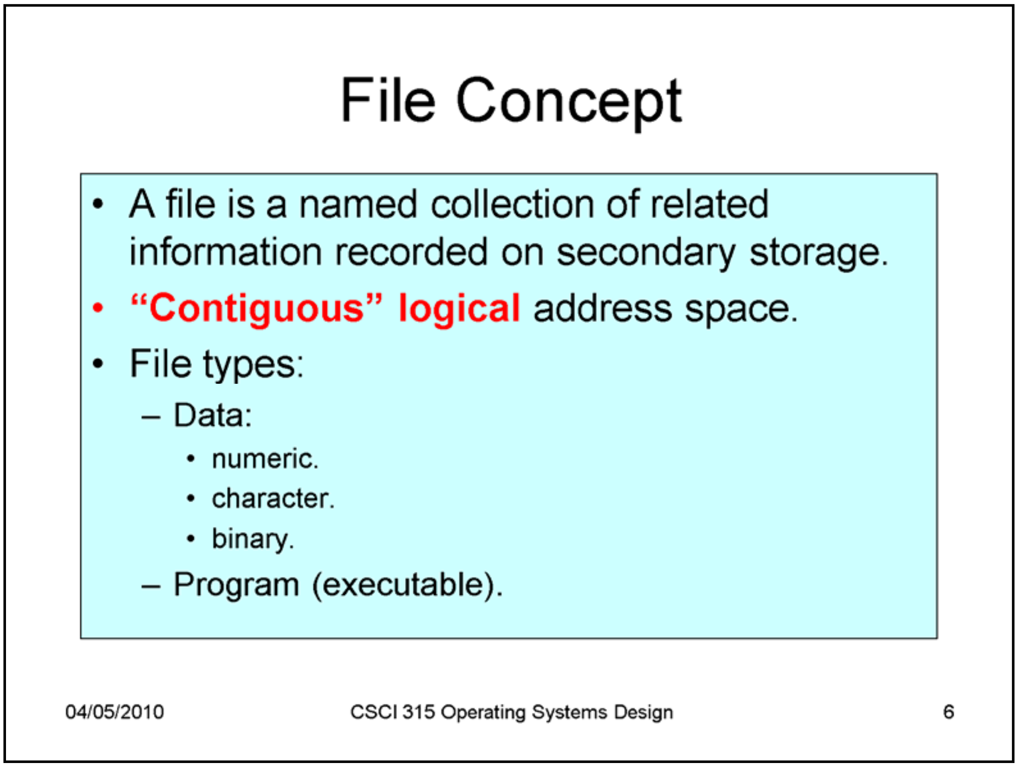
Prepaging

- **Prepaging**: In order to avoid the initial number of page faults, the system can bring into memory all the pages that will be needed all at once.
- This can also be applied when a swapped-out process is restarted. The smart thing to do is to remember the working set of the process.
- One question that arises is whether all the pages brought in will actually be used…
- Is the cost of prepaging less than the cost of servicing each individual page fault?



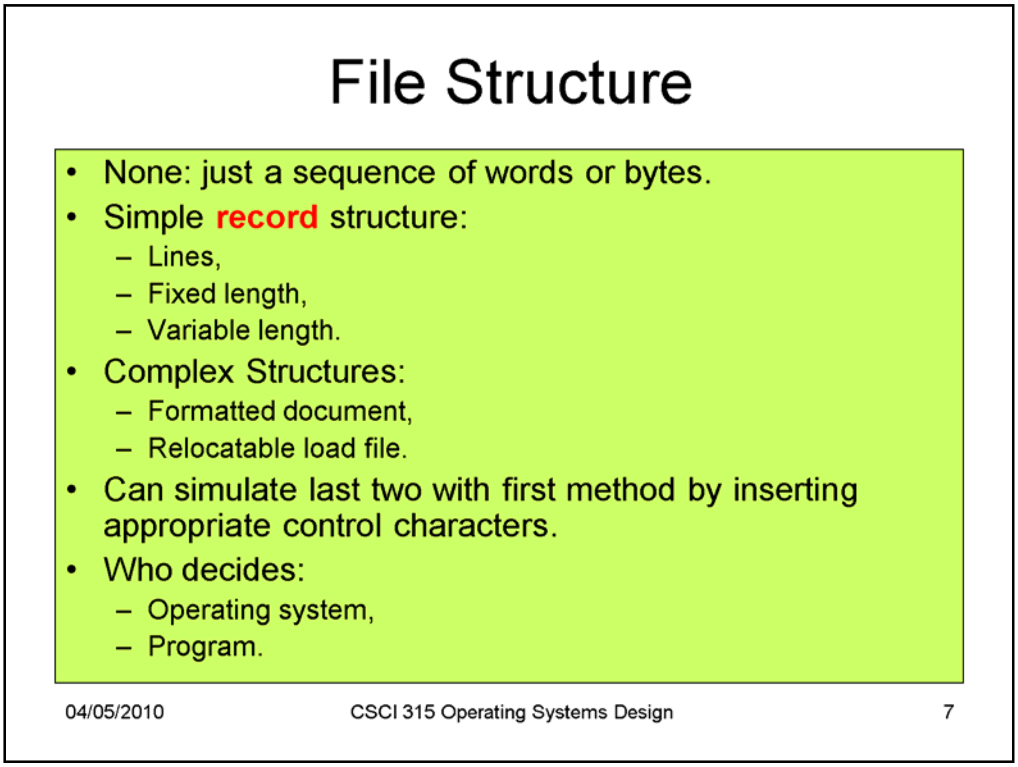
File System Topics

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



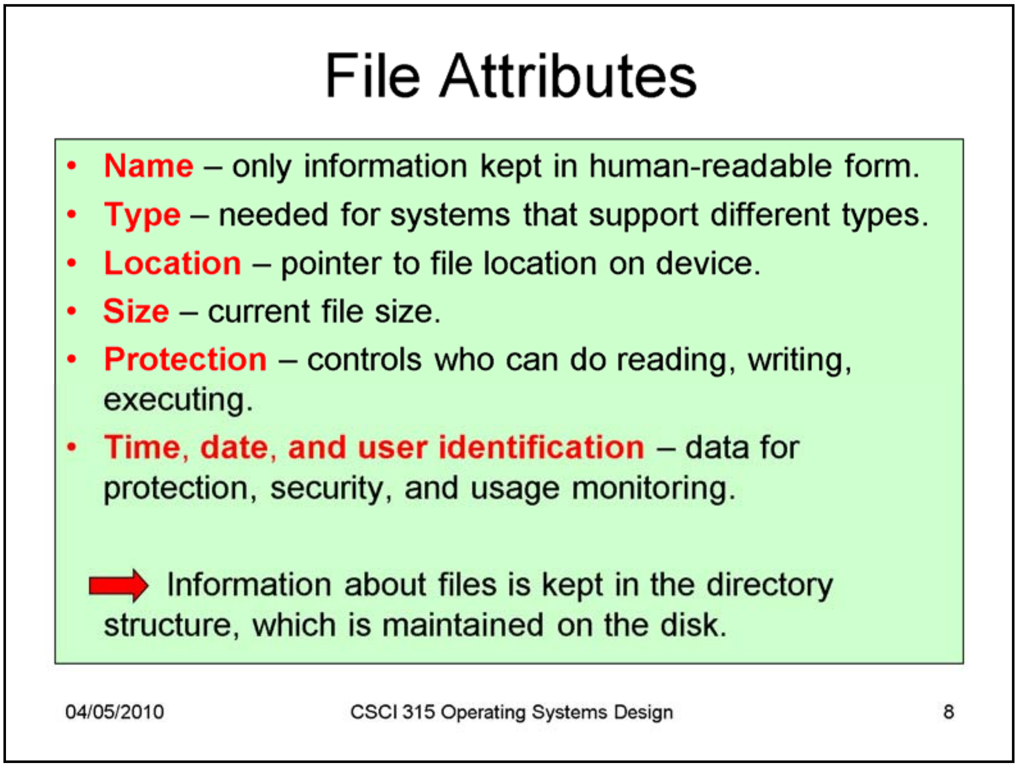
File Concept

- A file is a named collection of related information recorded on secondary storage.
- **“Contiguous” logical** address space.
- File types:
  - – Data:
    - numeric.
    - character.
    - binary.
  - – Program (executable).



File Structure

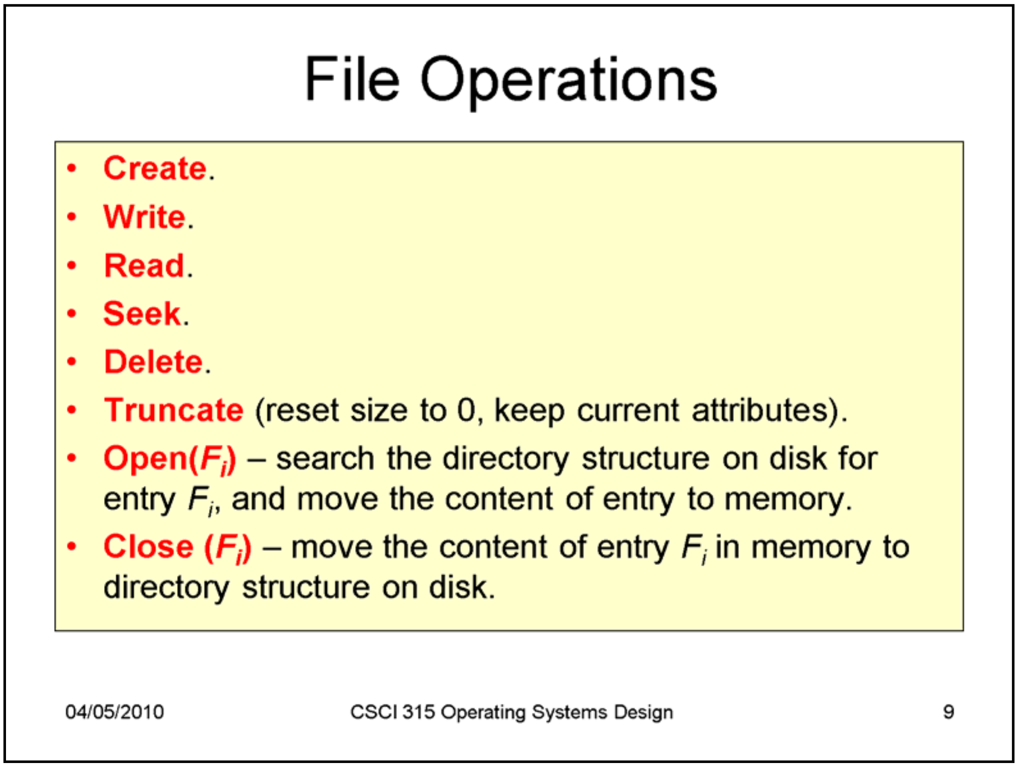
- None: just a sequence of words or bytes.
- Simple **record** structure:
  - Lines,
  - Fixed length,
  - Variable length.
- Complex Structures:
  - Formatted document,
  - Relocatable load file.
- Can simulate last two with first method by inserting appropriate control characters.
- Who decides:
  - Operating system,
  - Program.



File Attributes

- **Name** – only information kept in human-readable form.
- **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.

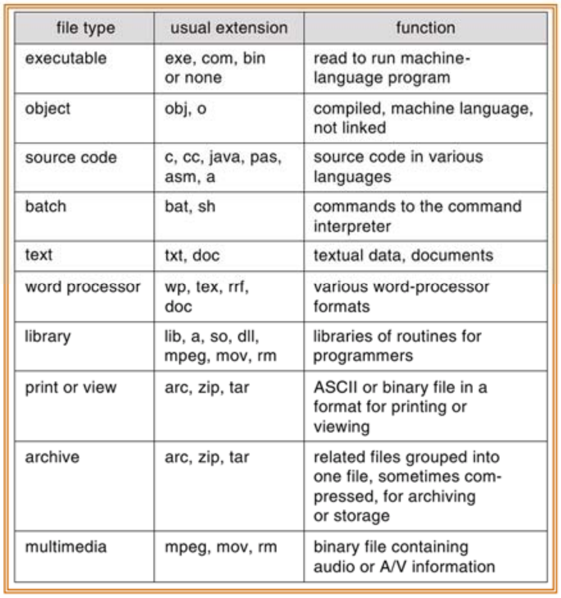
➡ Information about files is kept in the directory structure, which is maintained on the disk.



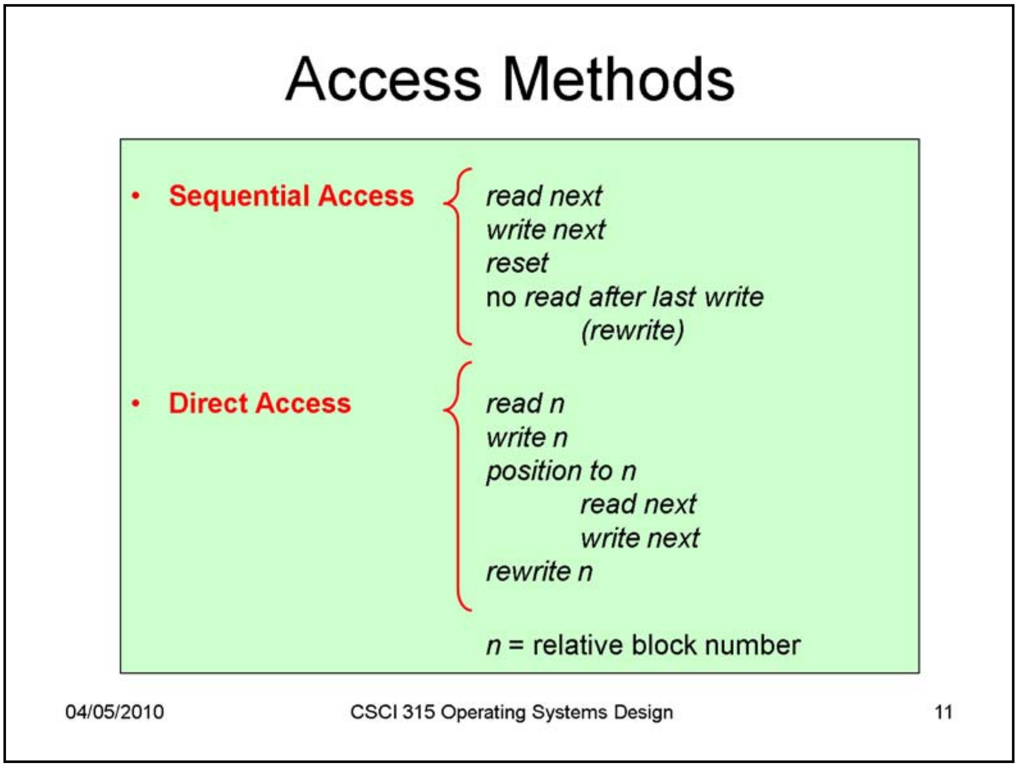
File Operations

- **Create**.
- **Write**.
- **Read**.
- **Seek**.
- **Delete**.
- **Truncate** (reset size to 0, keep current attributes).
- **Open(F_i)** – search the directory structure on disk for entry F_i, and move the content of entry to memory.
- **Close (F_i)** – move the content of entry F_i in memory to directory structure on disk.

File Types: Name and Extension



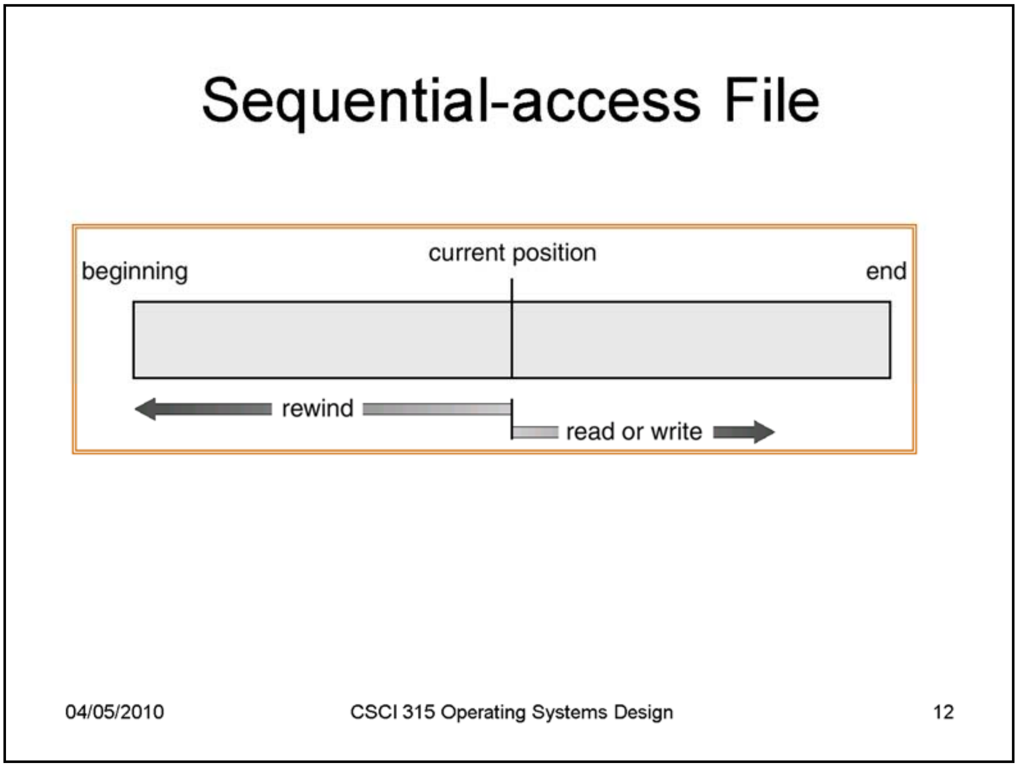
file type	usual extension	function
executable	exe, com, bin or none	read to run machine-language program
object	obj, o	compiled, machine language, not linked
source code	c, cc, java, pas, asm, a	source code in various languages
batch	bat, sh	commands to the command interpreter
text	txt, doc	textual data, 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 Methods

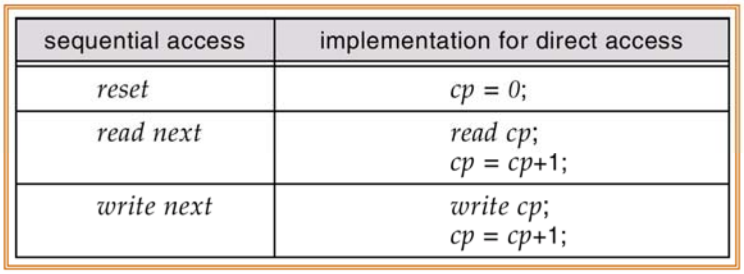
- **Sequential Access**
  - *read next*
  - *write next*
  - *reset*
  - no *read after last write*
    (rewrite)
- **Direct Access**
  - *read n*
  - *write n*
  - *position to n*
    - *read next*
    - *write next*
  - *rewrite n*

n = relative block number



Sequential-access File

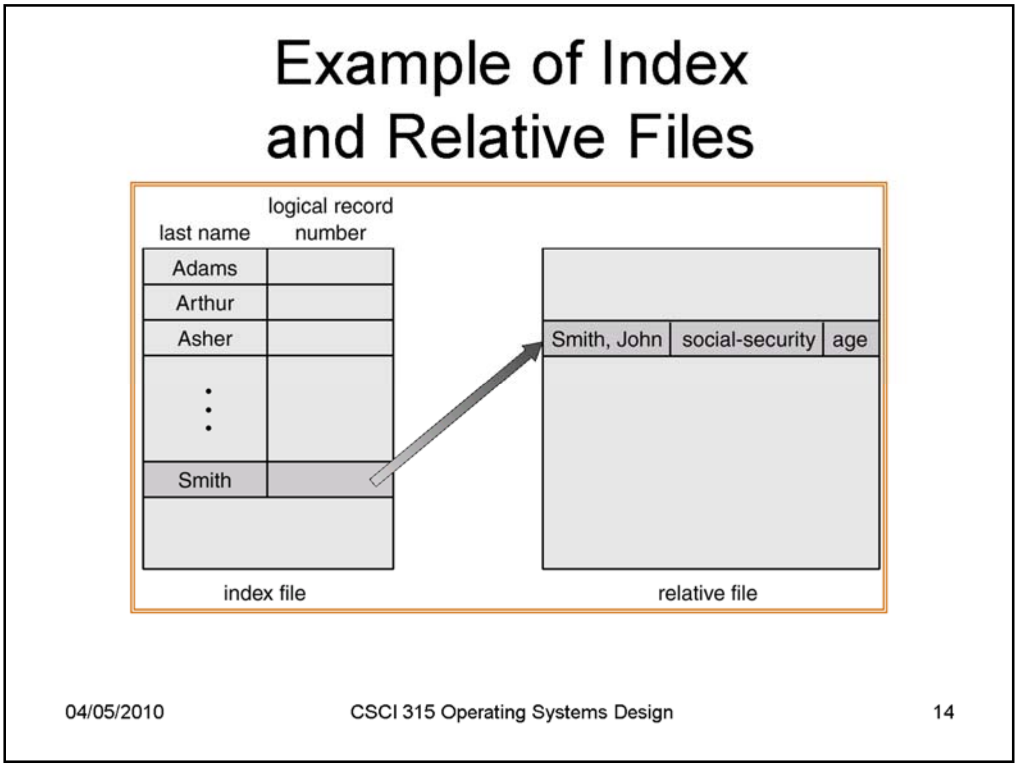
Simulation of Sequential Access on a Direct-access File



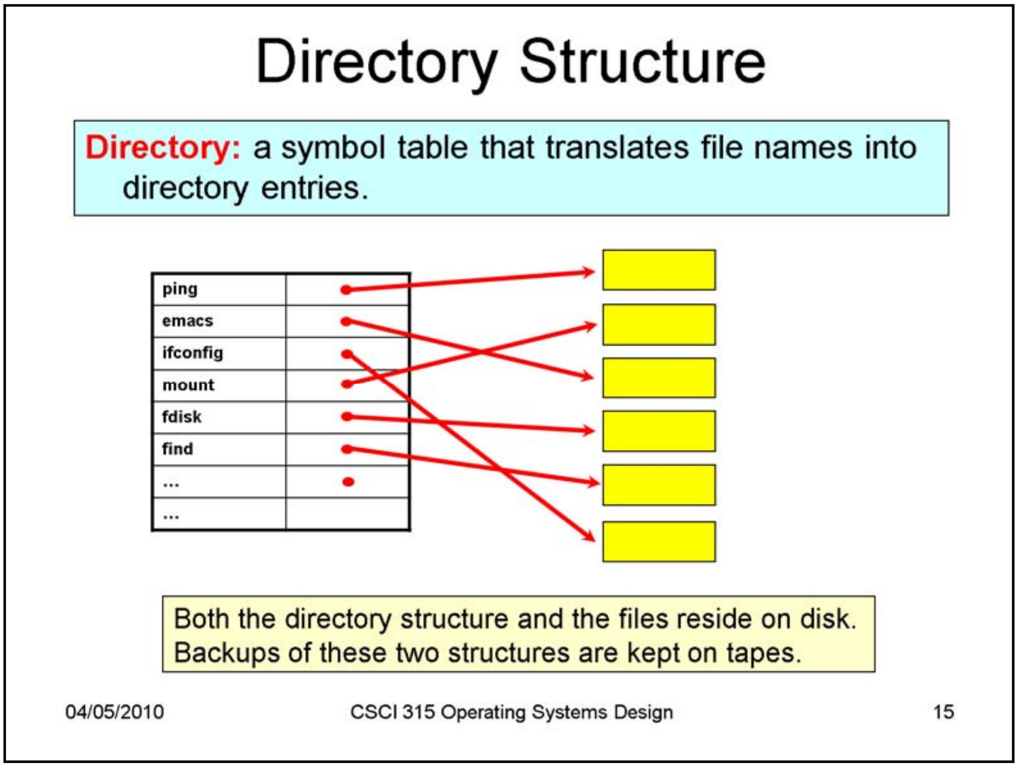
sequential access	implementation for direct access
reset	*cp = 0;*

read next	*read cp;*
	cp = cp+1;

write next	*write cp;*
	cp = cp+1;



Example of Index and Relative Files

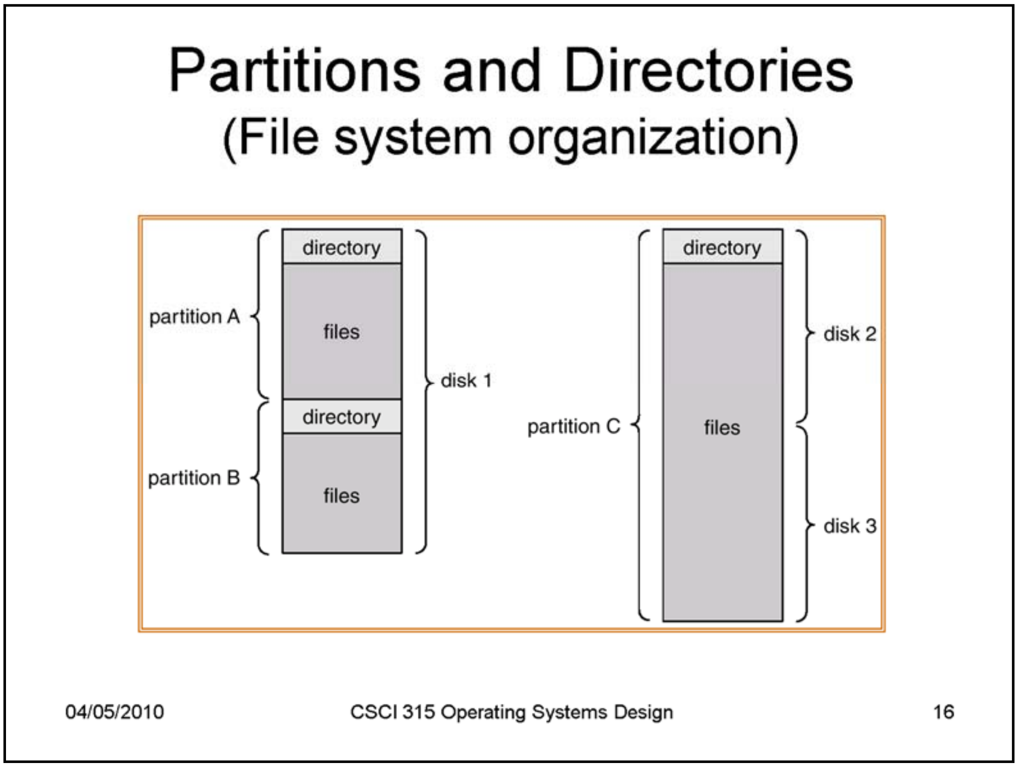


Directory Structure

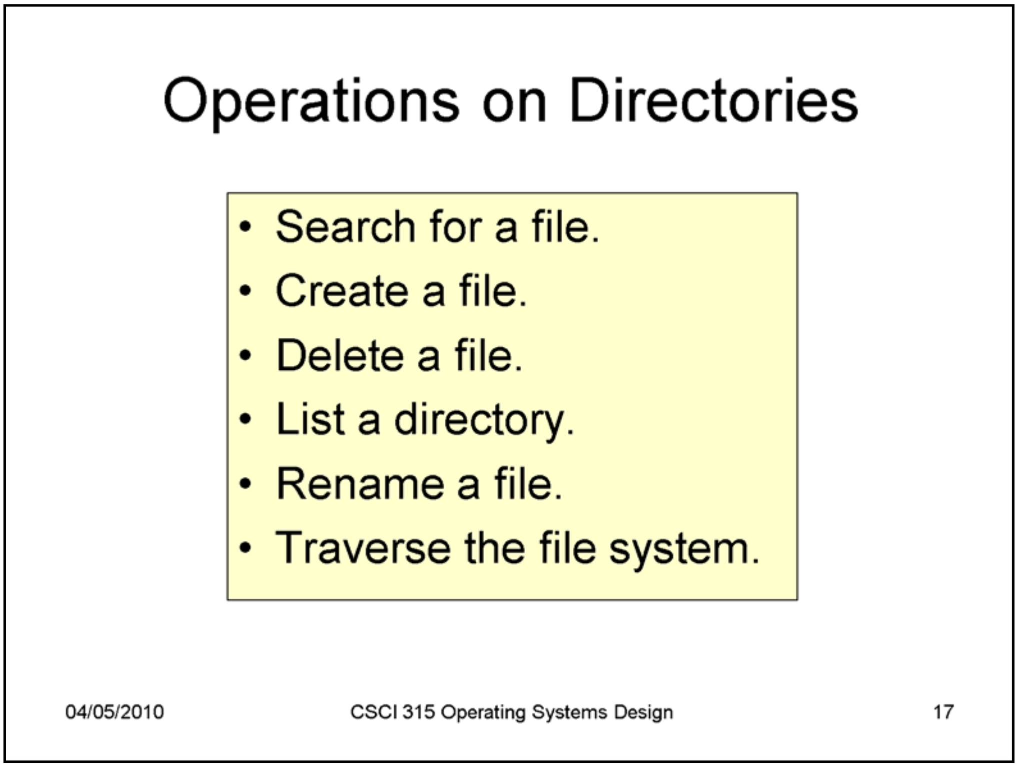
Directory: a symbol table that translates file names into directory entries.

ping	●
emacs	●
ifconfig	●
mount	●
fdisk	●
find	●
…	●
…	

Both the directory structure and the files reside on disk. Backups of these two structures are kept on tapes.

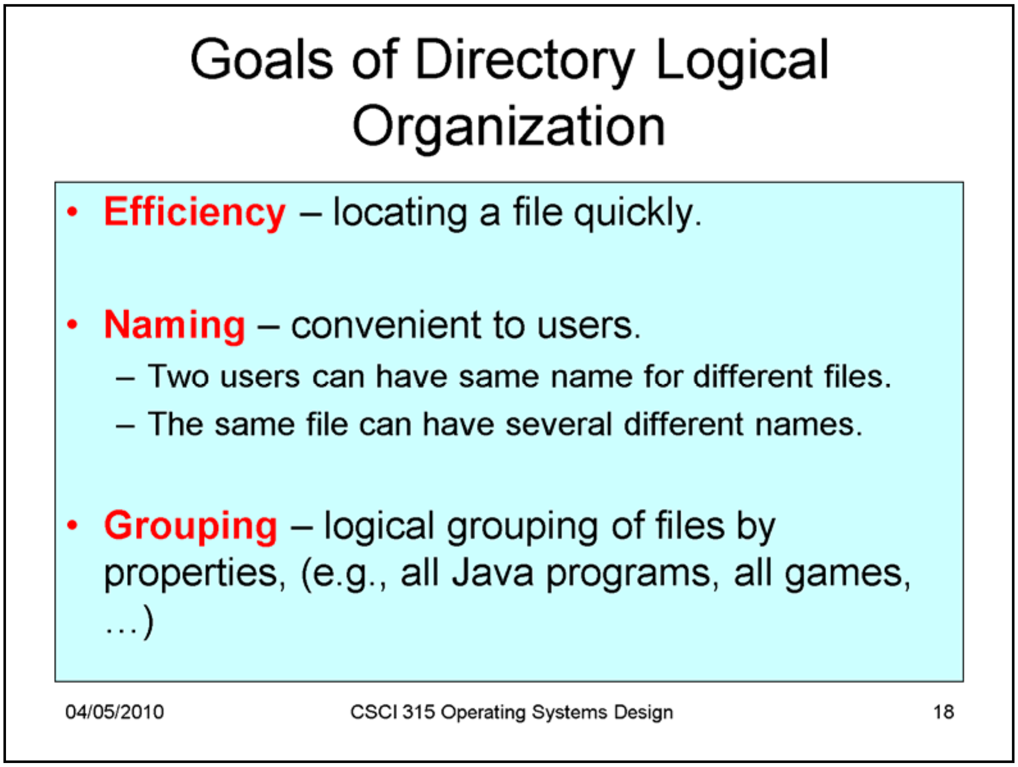


Partitions and Directories
(File system organization)



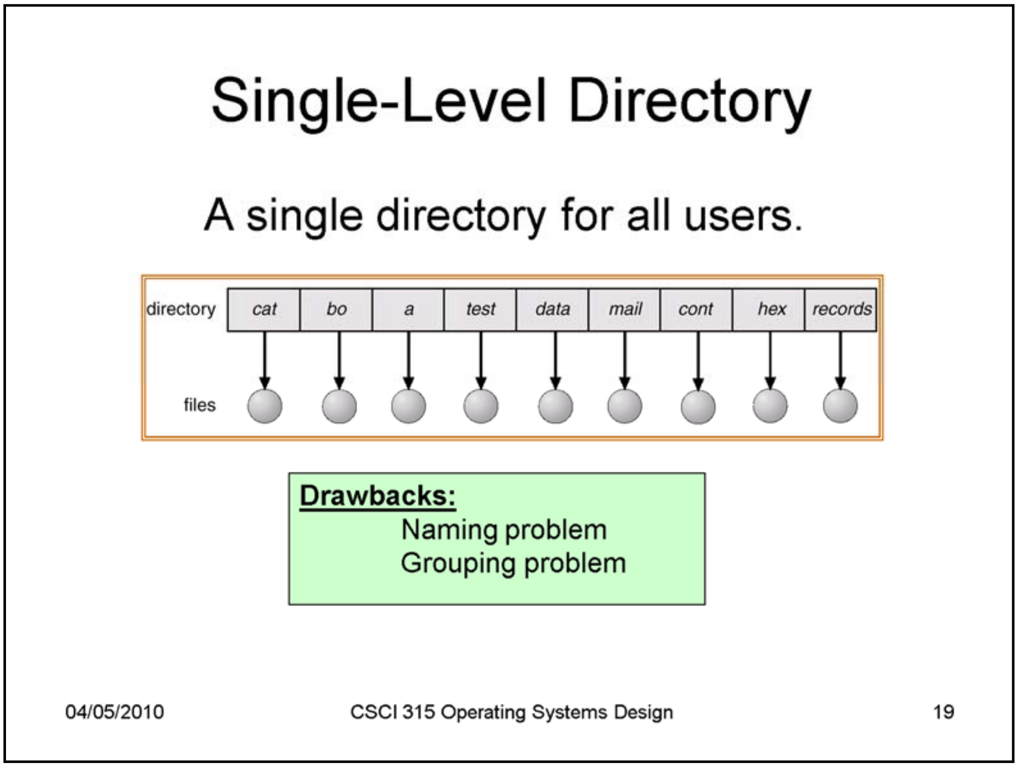
Operations on Directories

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



Goals of Directory Logical Organization

- **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, …)

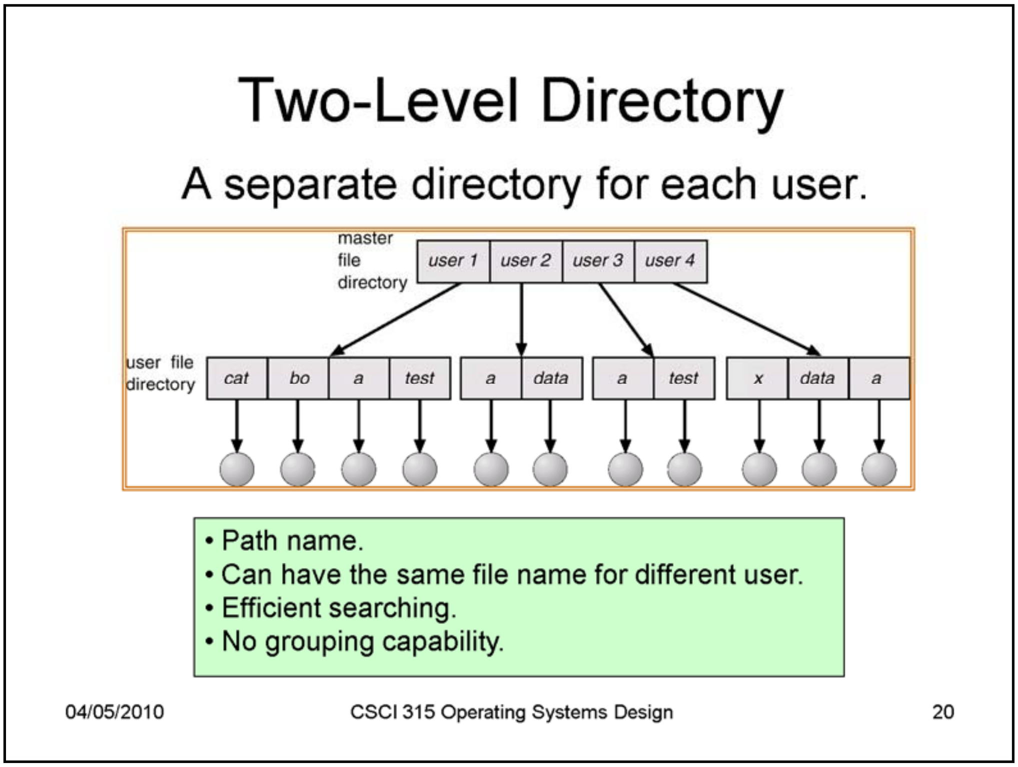


Single-Level Directory

A single directory for all users.

Drawbacks:

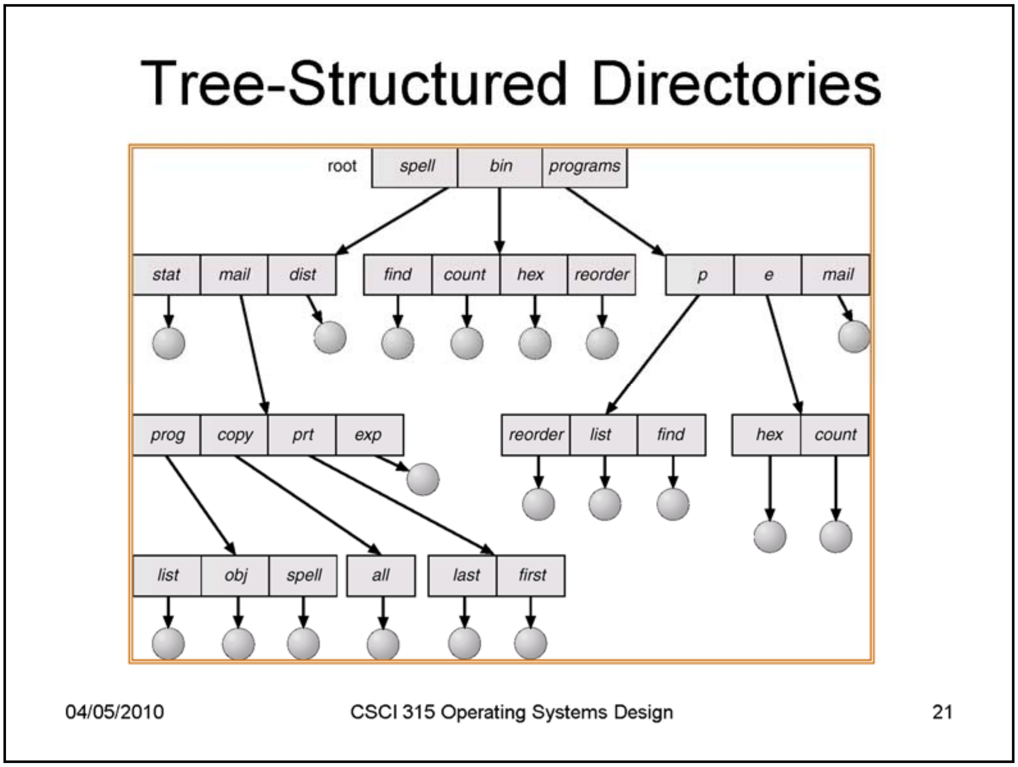
Naming problem
Grouping problem



Two-Level Directory

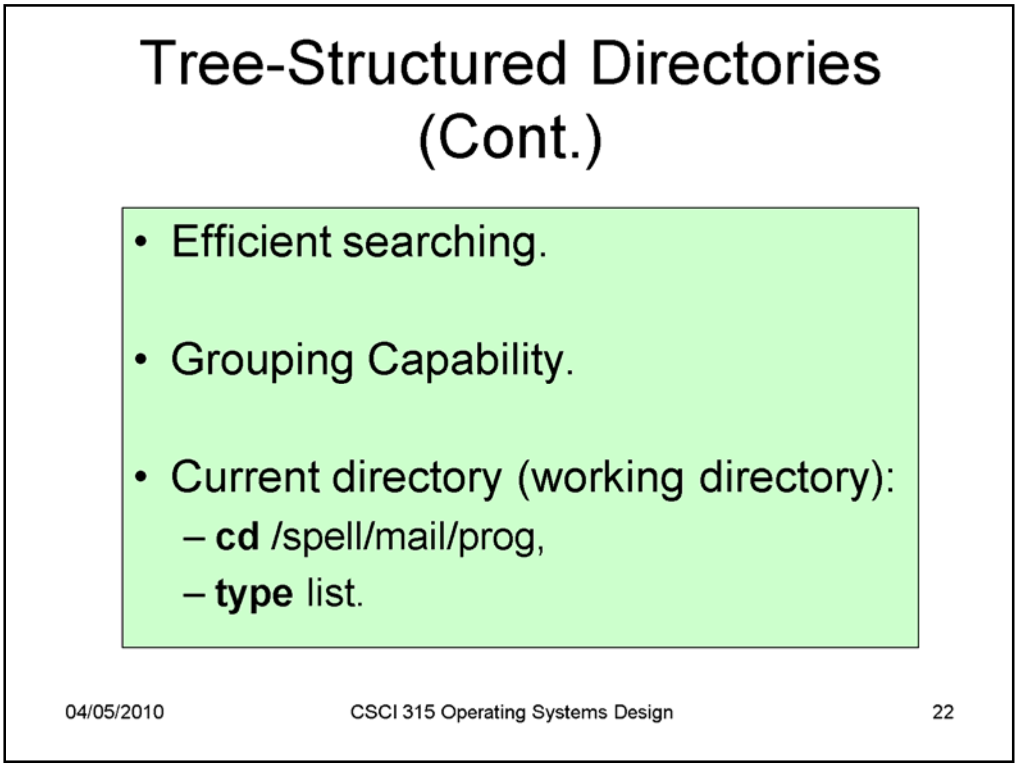
A separate directory for each user.

- Path name.
- Can have the same file name for different user.
- Efficient searching.
- No grouping capability.



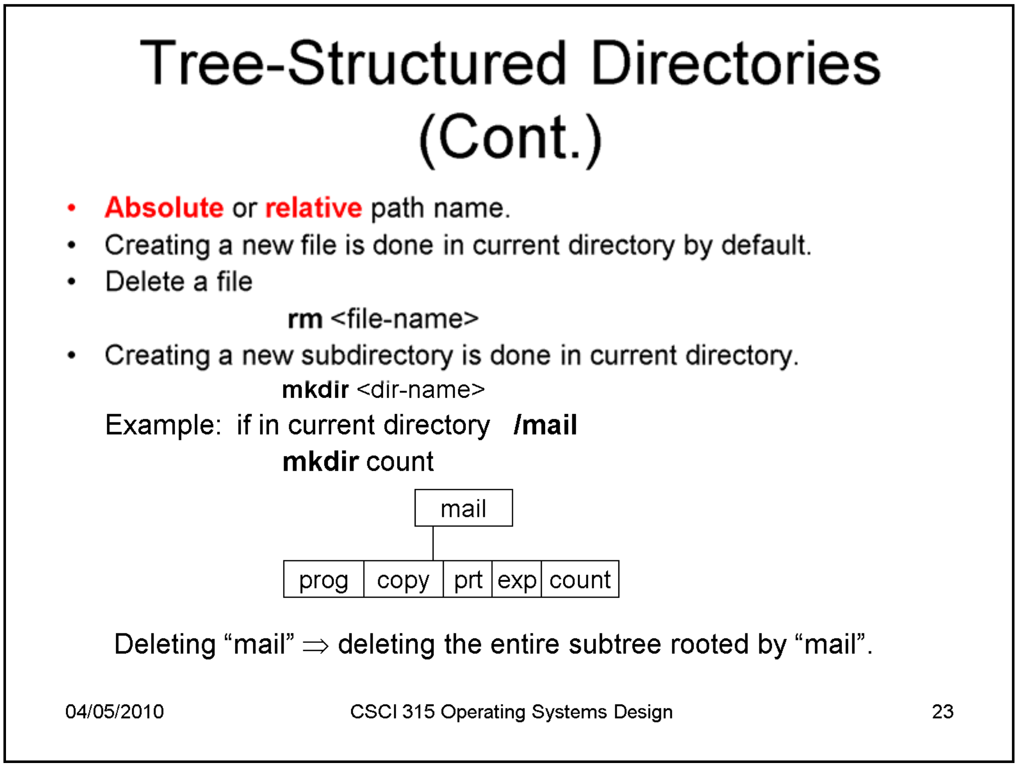
Tree-Structured Directories

04/05/2010 CSCI 315 Operating Systems Design 21



Tree-Structured Directories (Cont.)

- Efficient searching.
- Grouping Capability.
- Current directory (working directory):
  - **cd** /spell/mail/prog,
  - **type** list.



Tree-Structured Directories (Cont.)

- **Absolute** or **relative** path name.
- Creating a new file is done in current directory by default.
- Delete a file

  rm <file-name>
- Creating a new subdirectory is done in current directory.

  mkdir <dir-name>

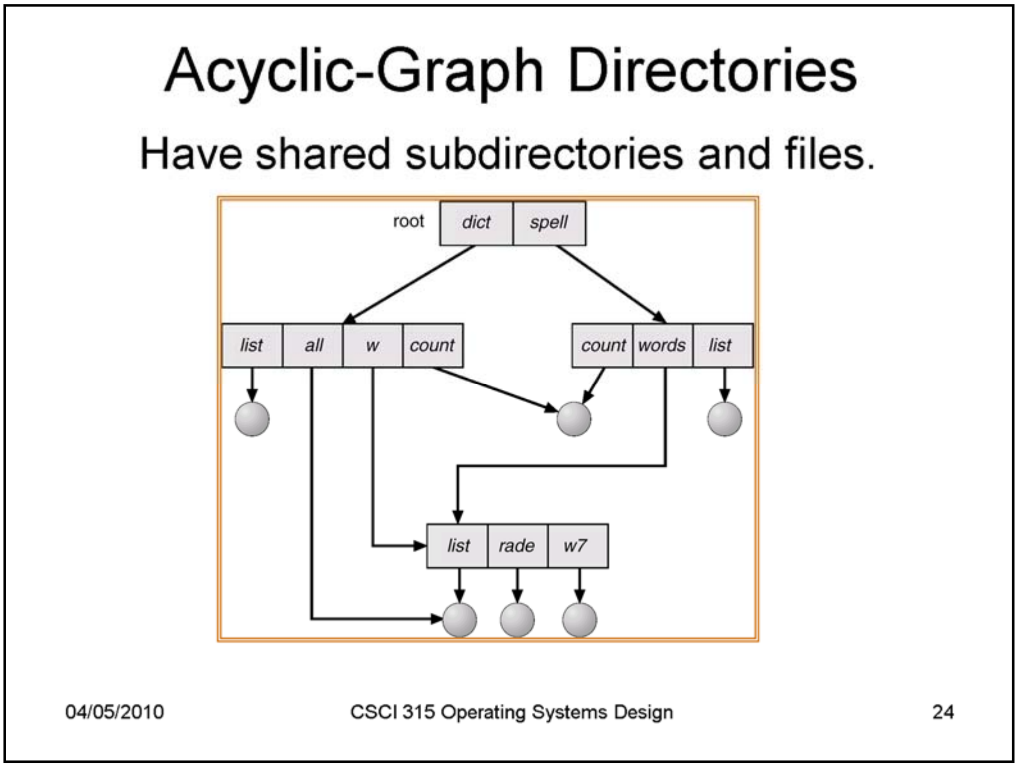
  Example: if in current directory **/mail**

  mkdir count

mail

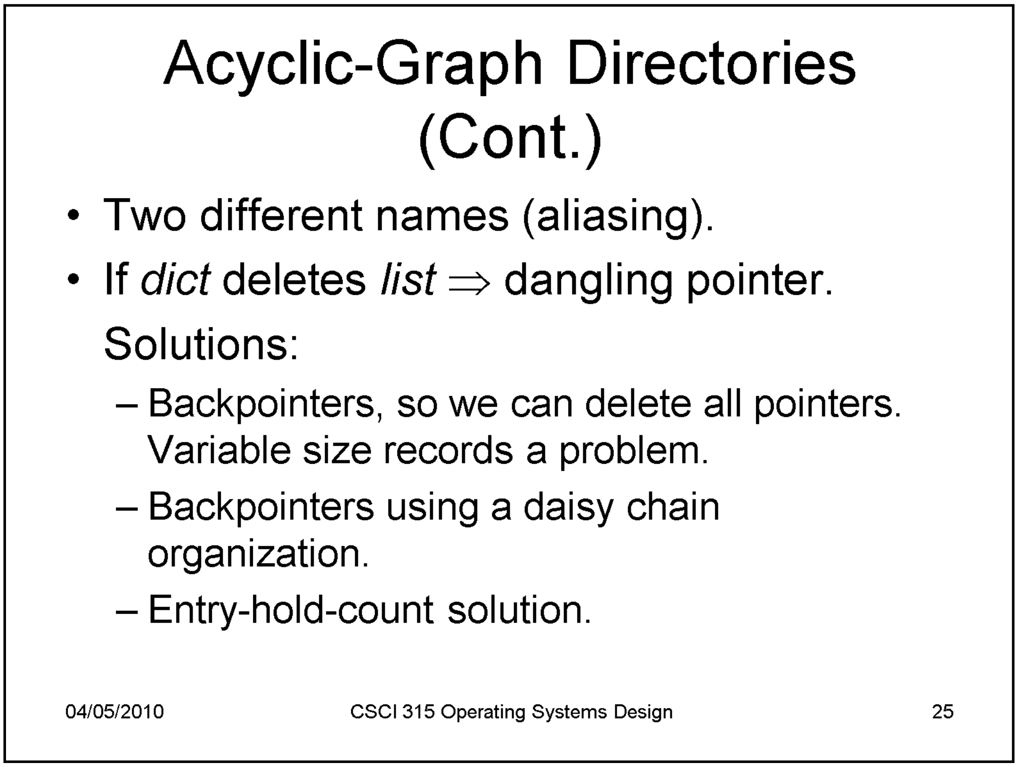
prog	copy	prt	exp	count

Deleting “mail” ⇒ deleting the entire subtree rooted by “mail”.



Acyclic-Graph Directories

Have shared subdirectories and files.

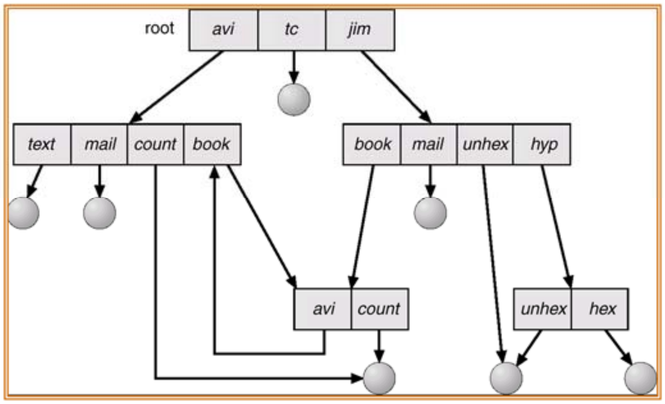


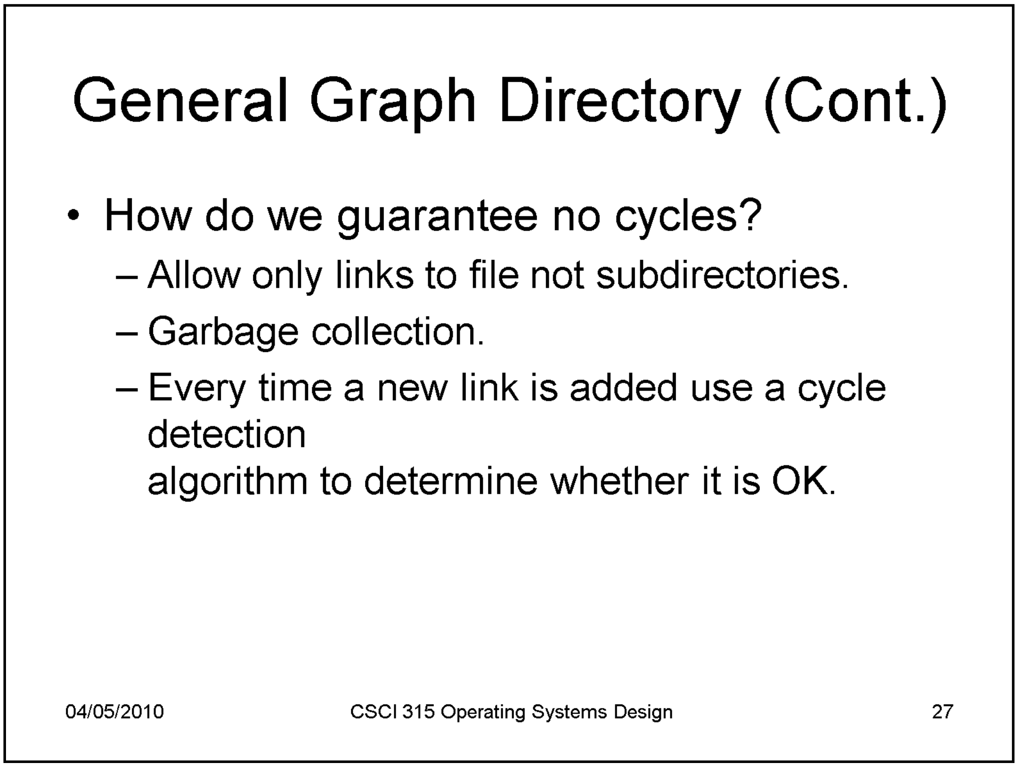
Acyclic-Graph Directories (Cont.)

- Two different names (aliasing).
- If *dict* deletes *list* ⇒ dangling pointer.

  Solutions:
  - Backpointers, so we can delete all pointers. Variable size records a problem.
  - Backpointers using a daisy chain organization.
  - Entry-hold-count solution.

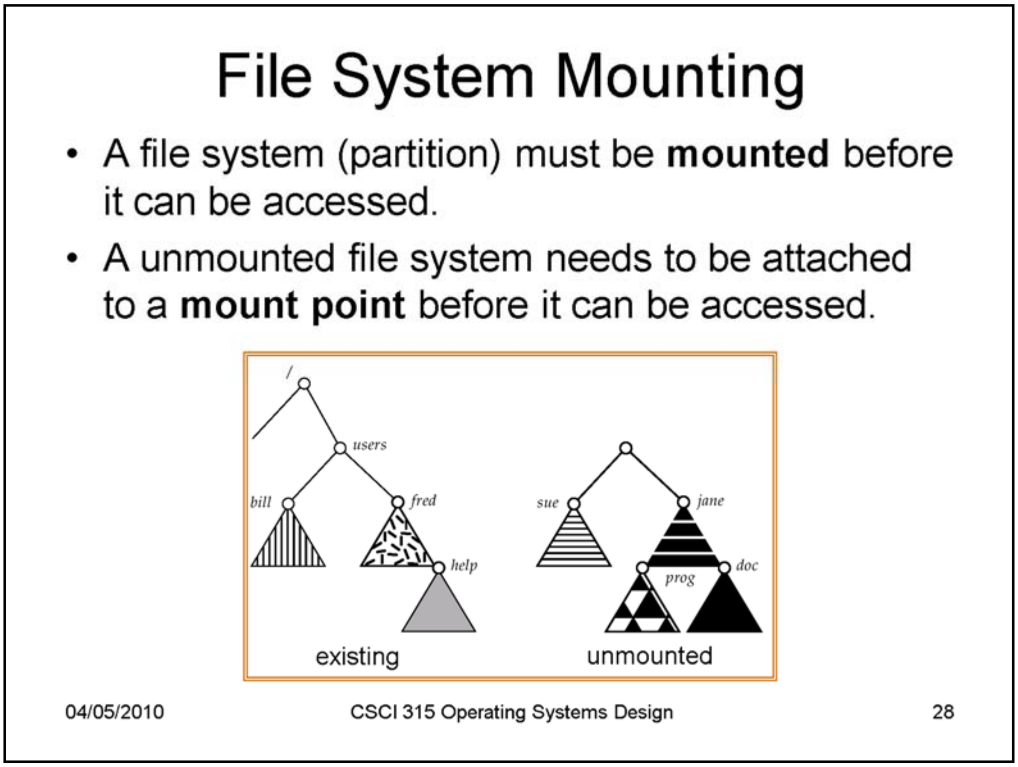
General Graph Directory





General Graph Directory (Cont.)

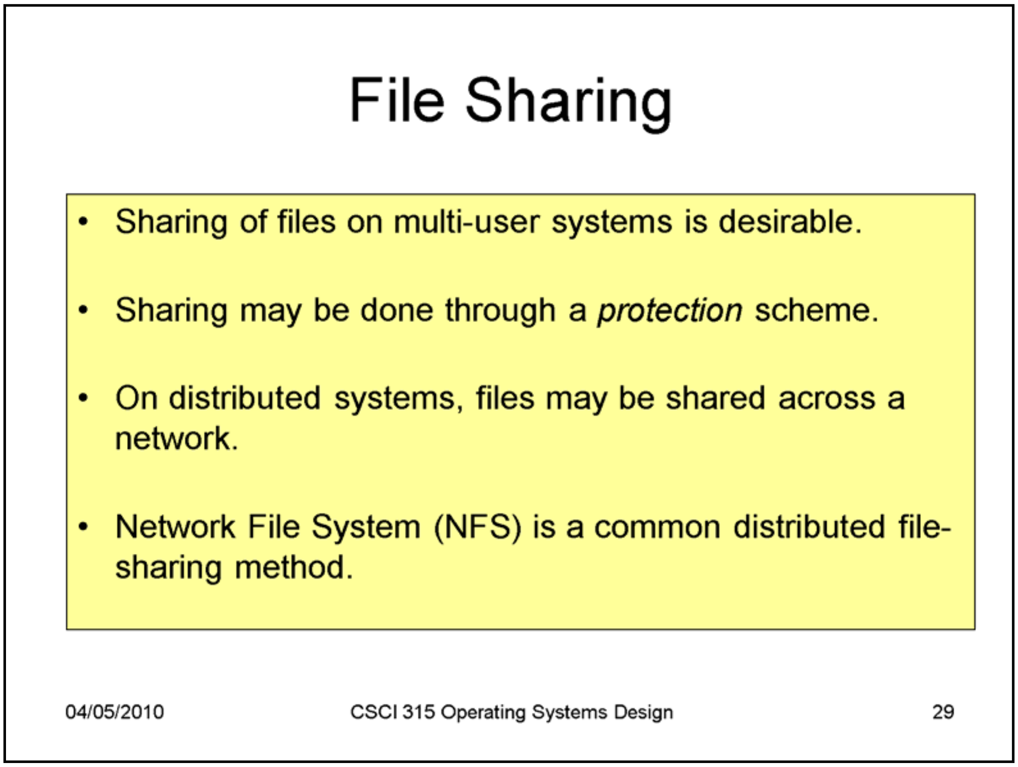
- How do we guarantee no cycles?
  - Allow only links to file not subdirectories.
  - Garbage collection.
  - Every time a new link is added use a cycle detection
    algorithm to determine whether it is OK.



File System Mounting

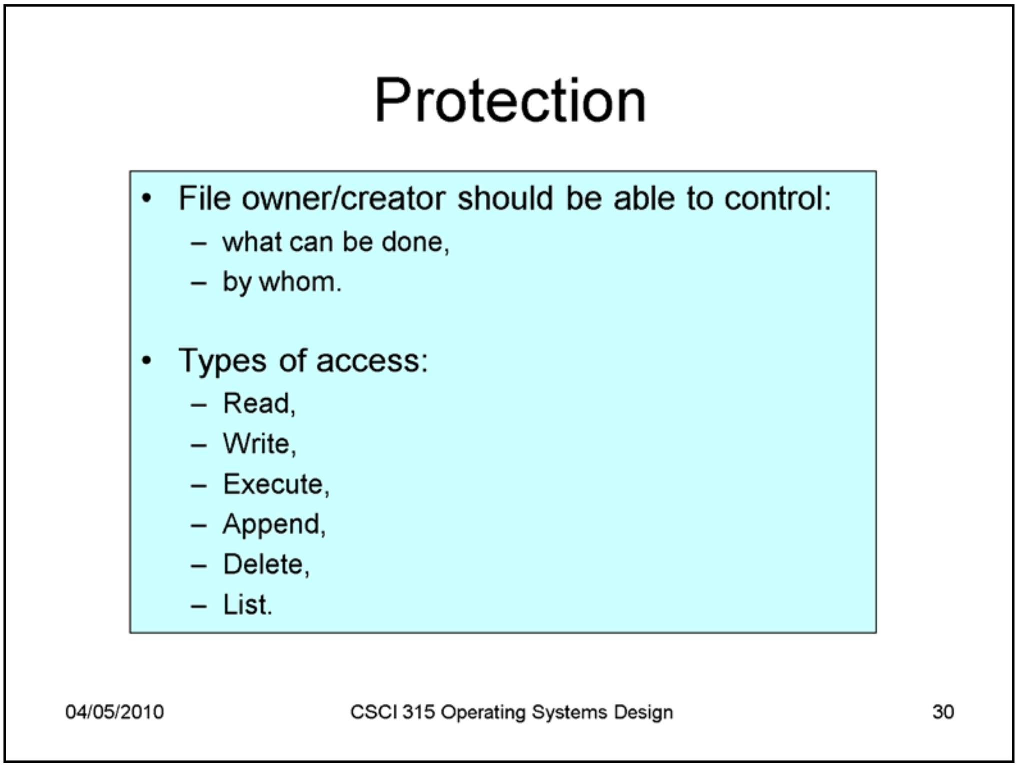
- A file system (partition) must be **mounted** before it can be accessed.
- A unmounted file system needs to be attached to a **mount point** before it can be accessed.

/
users
bill
fred
help
sue
jane
prog
doc
existing
unmounted



File Sharing

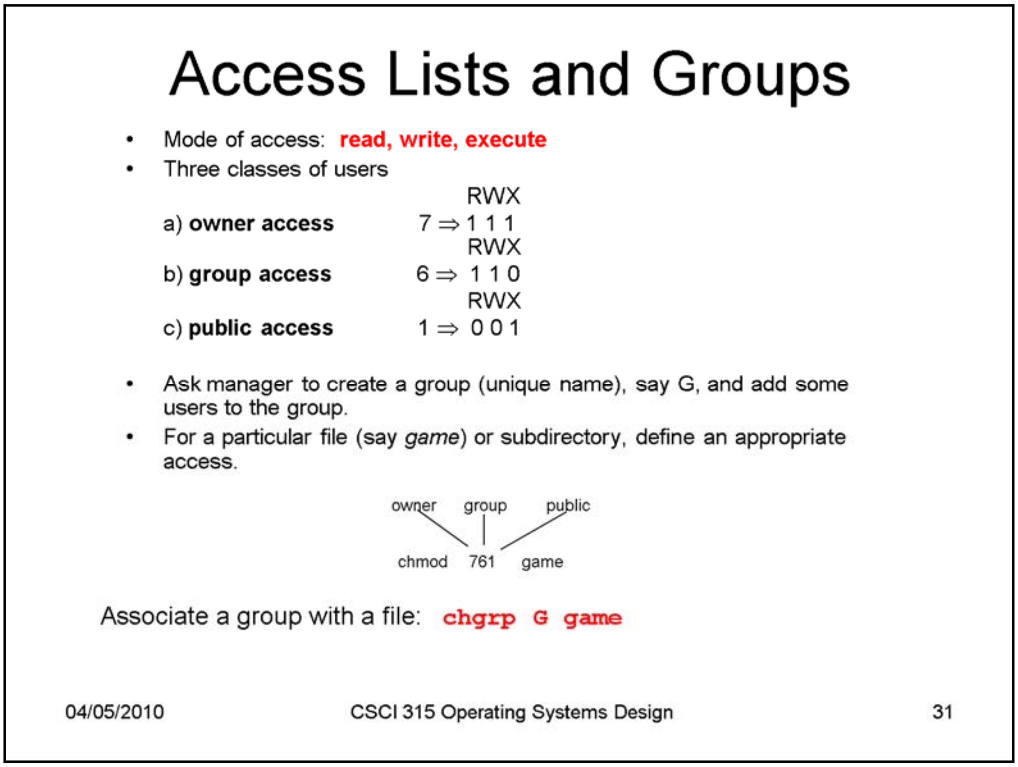
- Sharing of files on multi-user systems is desirable.
- Sharing may be done through a *protection* scheme.
- On distributed systems, files may be shared across a network.
- Network File System (NFS) is a common distributed file-sharing method.



Protection

- File owner/creator should be able to control:
  - what can be done,
  - by whom.

- Types of access:
  - Read,
  - Write,
  - Execute,
  - Append,
  - Delete,
  - List.



Access Lists and Groups

- Mode of access: **read, write, execute**
- Three classes of users

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

owner group public

chmod 761 game

Associate a group with a file: `chgrp G game`