


Chapter 13:

File-System Interface



Outline

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

Objectives

- To explain the function of file systems
- To describe the interfaces to file systems
- To discuss file-system design tradeoffs, including access methods, file sharing, file locking, and directory structures
- To explore file-system protection

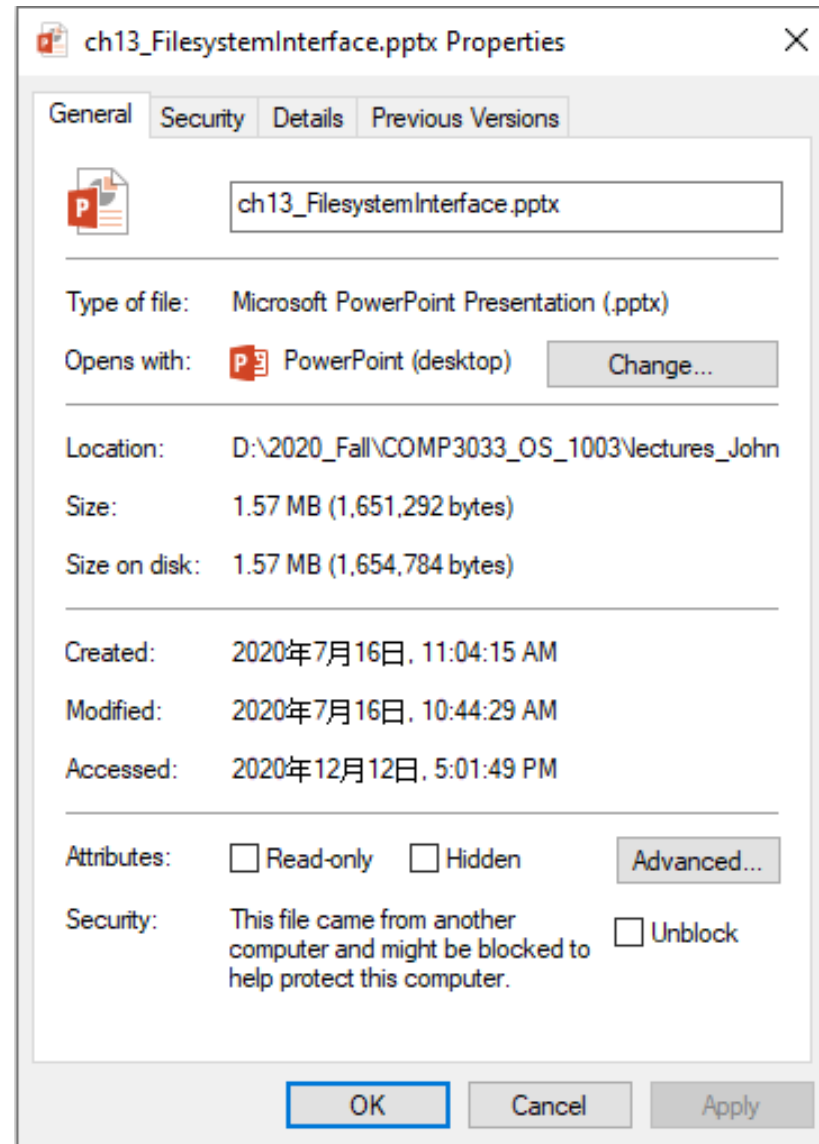
File Concept

- A file is
 - a named collection of related information that is recorded on secondary storage (e.g. Disk, SSD, Flash, etc.)
 - a sequence of bits, bytes, lines, or records, the meaning of which is defined by the file's creator
 - Smallest logical storage unit for user view
- A file
 - uses contiguous logical address space
 - represents programs or data
- File system resides on secondary storage (disks)
 - 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

File Attributes

- A file usually has following basic 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 ([metadata](#)) are kept in the directory structure, which is maintained on the disk

File info Window on OS X and Win10



File Operations

- File is an abstract data type.
- Six basic file operations
 1. Create a file
 2. Write a file
 3. Read a file
 4. Reposition within a file
 5. Truncate a file
 - ▶ File is reset to length zero
 - ▶ File space is released
 - ▶ File attributes are unchanged
 6. Delete a file
- Except file creation and deletion, all other operations need to open the file first and then close the file after the operation

Open File Locking

- Lock of file is provided by some operating systems
- Similar to reader-writer locks
- Two kinds of locks
 - ▶ Shared lock
 - similar to reader lock
 - several processes can acquire concurrently
 - ▶ Exclusive lock
 - similar to writer lock
 - ▶ Mandatory lock
 - once a process acquires an exclusive lock, the operating system will prevent any other process from accessing the locked file
 - E.g., Windows
 - ▶ Advisory lock
 - it is up to software developers to ensure that locks are appropriately acquired and released
 - E,g, Unix/Linux

File Locking Example – Java API (Cont.)

```
import java.io.*;
import java.nio.channels.*;
public class LockingExample {
    public static final boolean EXCLUSIVE =
    false;
    public static final boolean SHARED = true;
    public static void main(String arsg[])
    throws IOException {
        FileLock sharedLock = null;
        FileLock exclusiveLock = null;
        try {
            RandomAccessFile raf = new
            RandomAccessFile("file.txt", "rw");
            // get the channel for the file
            FileChannel ch = raf.getChannel();
            // this locks the first half of the file
            - exclusive
            exclusiveLock = ch.lock(0,
            raf.length()/2, EXCLUSIVE);
            /** Now modify the data . . . */

            // release the lock
            exclusiveLock.release();
```

```
        // this locks the second half of
        the file - shared
        sharedLock =
        ch.lock(raf.length()/2+1,
        raf.length(), SHARED);
        /** Now read the data . . . */

        // release the lock
        sharedLock.release();
    }catch (java.io.IOException ioe){
        System.err.println(ioe);
    }finally {
        if (exclusiveLock != null)
            exclusiveLock.release();
        if (sharedLock != null)
            sharedLock.release();
    }
}
```

File Types – Name, Extension

- The OS uses the **extension** to indicate the type of the file
- Most common types
 - .c
 - .exe
 - .docx
 - .pdf
 - .jpg
 - .png
 - .zip

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

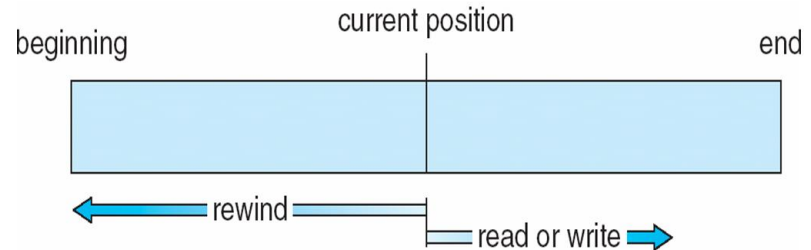
Common file types

File Access Methods

■ Sequential Access

- Commands

- ▶ `read_next`
- ▶ `write_next`
- ▶ `reset` (to the beginning)



■ Direct/Random Access

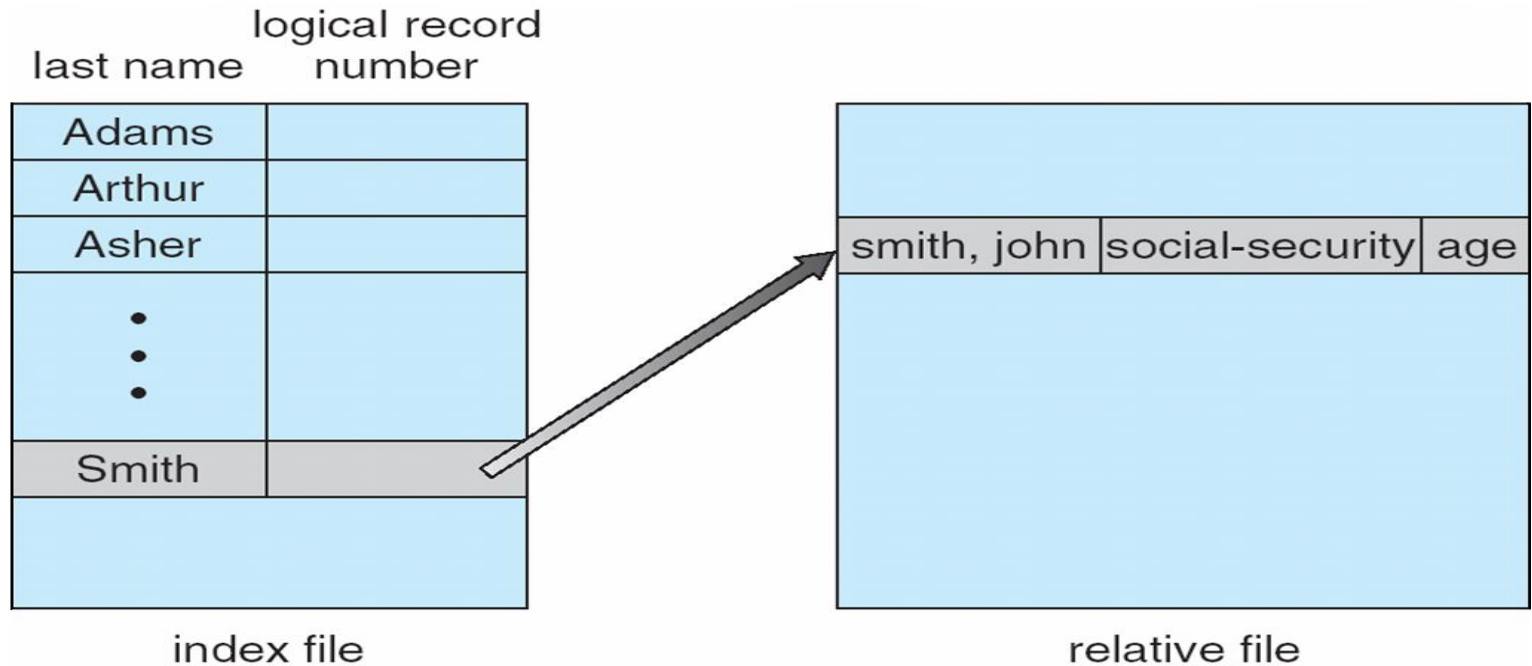
- Commands

- ▶ `read next`
- ▶ `write next`
- ▶ `read n`
- ▶ `write n`
- ▶ `position to n`
- ▶ `rewrite n`

sequential access	implementation for direct access
<i>reset</i>	$cp = 0;$
<i>read next</i>	$read\ cp;$ $cp = cp + 1;$
<i>write next</i>	$write\ cp;$ $cp = cp + 1;$

(n = an index relative to the beginning of the file)

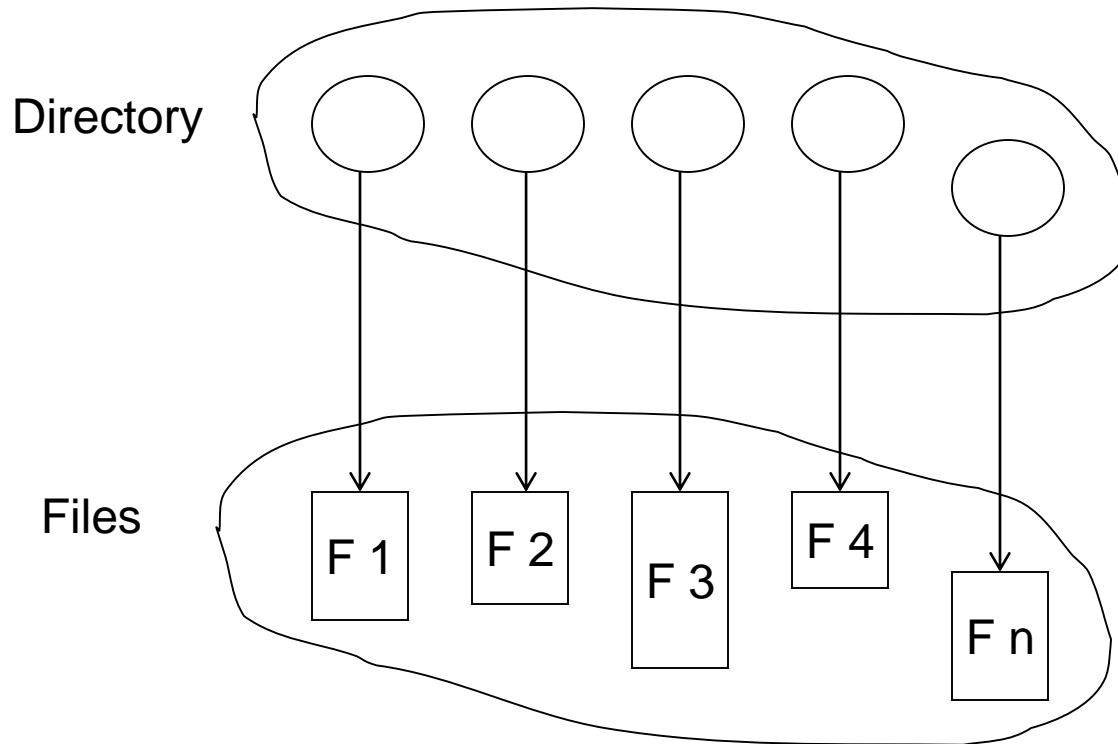
Other Access Method: Index and Relative Files



- Keep index in memory for fast determination of location of data to be operated on
 - E.g., VMS OS

Directory Structure

- A **directory structure** (per file system) is used to organize the files

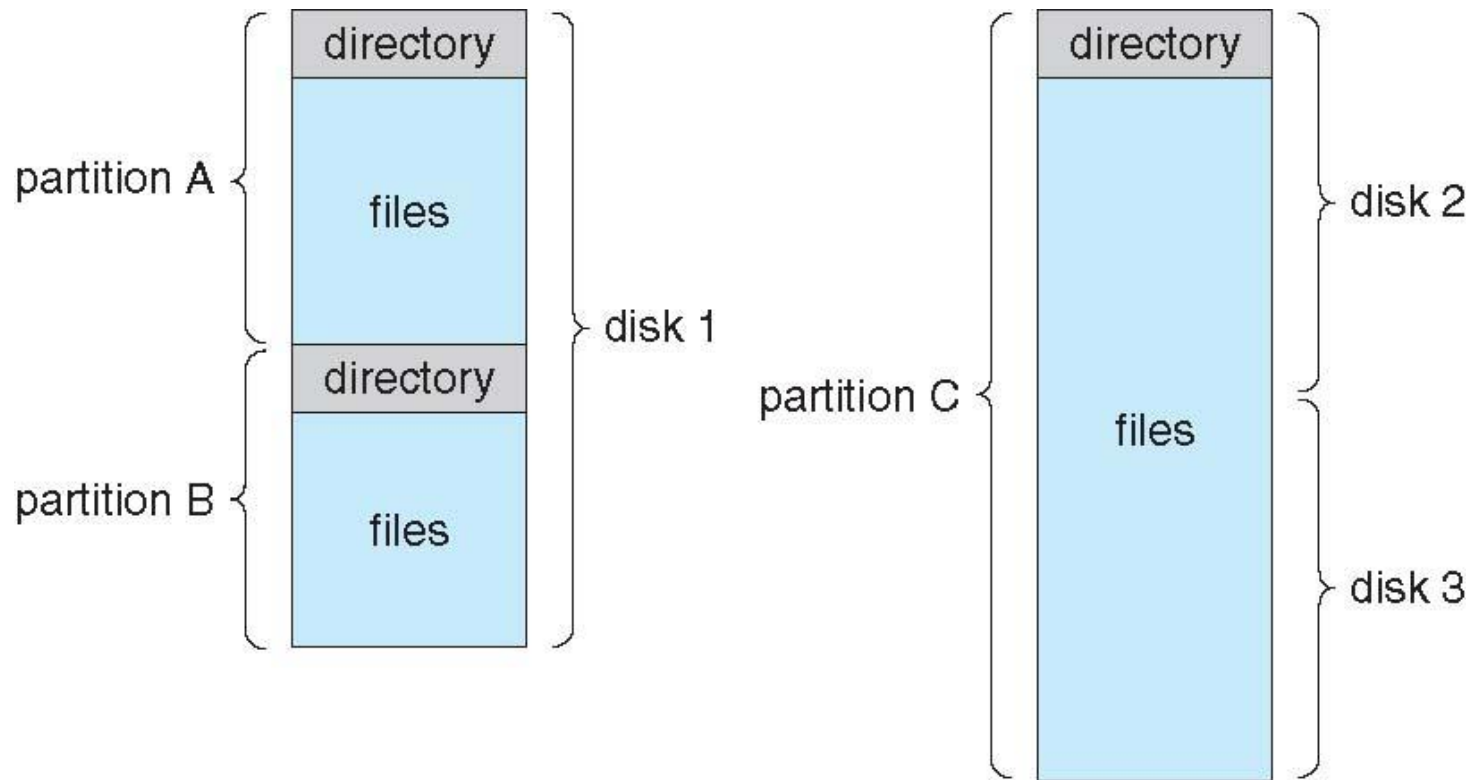


Both the directory structure and the files reside on disk

Disk Structure

- A **drive** is a **physical block** disk.
- A **drive** can be subdivided into several **logic block disks (partitions)**
- Disk or partition can be used **raw**, i.e., without a file system
- **Partition** can be formatted with a **file system (cooked)**
 - **Root partition** contains the **OS kernel**, and other system files
 - **Other partitions** can hold other OSes, other file systems, or be **raw**.
- A **volume** is a single accessible storage area with a single file system, a volume can contain multiple partitions
- File system types
 - **General-purpose file systems**
 - ▶ Mostly used, like Windows
 - **Special-purpose file systems**
 - ▶ E.g., Solaris's contract file system

A Typical File-system Organization



A typical file-system organization

Operations Performed on Directory

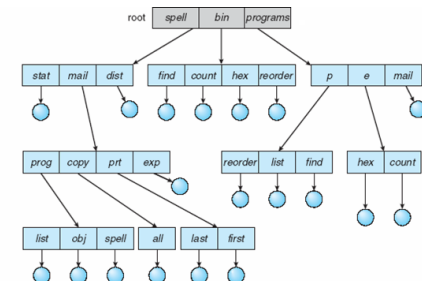
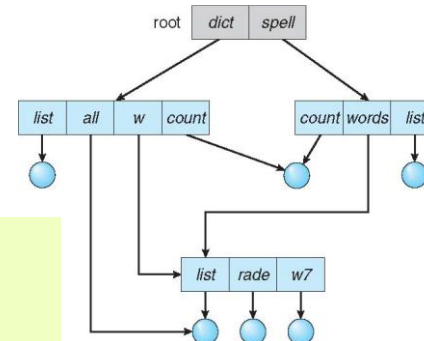
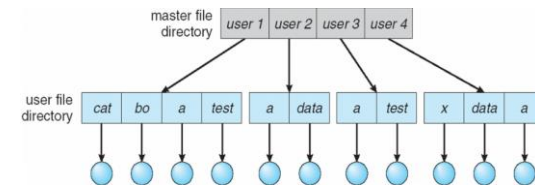
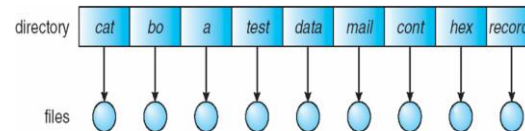
- Basic operations
 - Search for a file
 - Create a file
 - Delete a file
 - List a directory
 - Rename a file
 - Traverse the file system

Operations Performed on Directory

- The directory is organized logically to obtain
 - Efficient location of a file
 - Naming convenient to users
 - ▶ Two files in different directories can have the same name
 - ▶ The same file can have several different names
 - Grouping of files with similar properties
 - ▶ E.g., put photos in a directory

- Directory organization

- Single level directory
- Two-level directory
- Tree-structured directories
- Acyclic graph directories

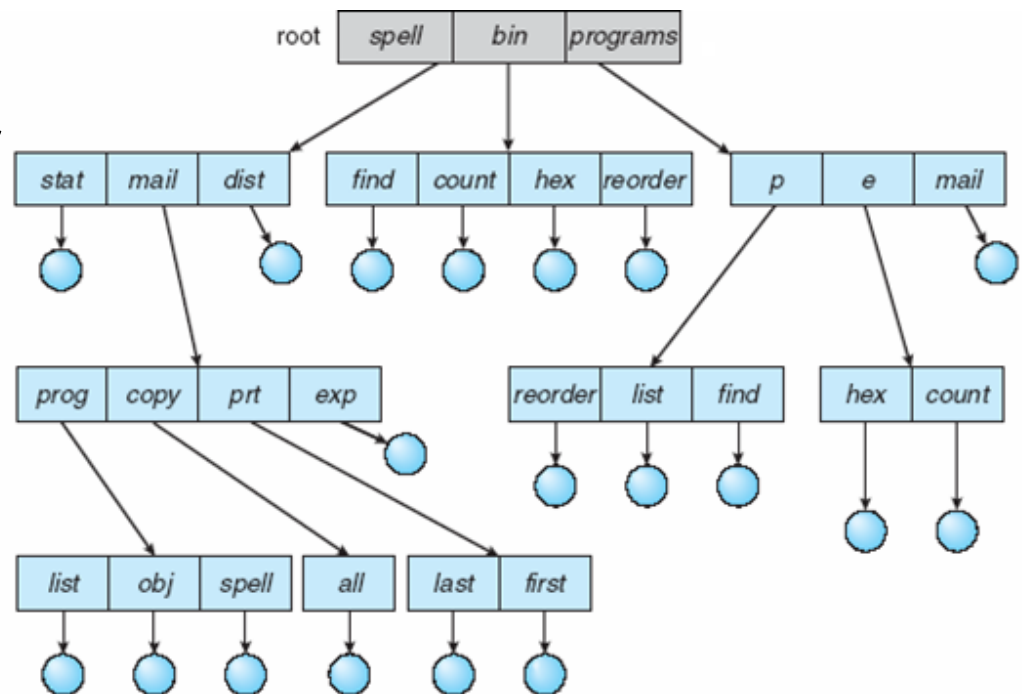


Thinking:

1. Can you tell the organizations of these four examples?
2. Disadvantages for single-level and two-level?

Tree-Structured Directories

- Tree structures is a tree of arbitrary height, **mostly common directory structure**
- It allows users to create their own subdirectories and to organize their files accordingly.
- Advantages
 - Efficient searching
 - Grouping capability



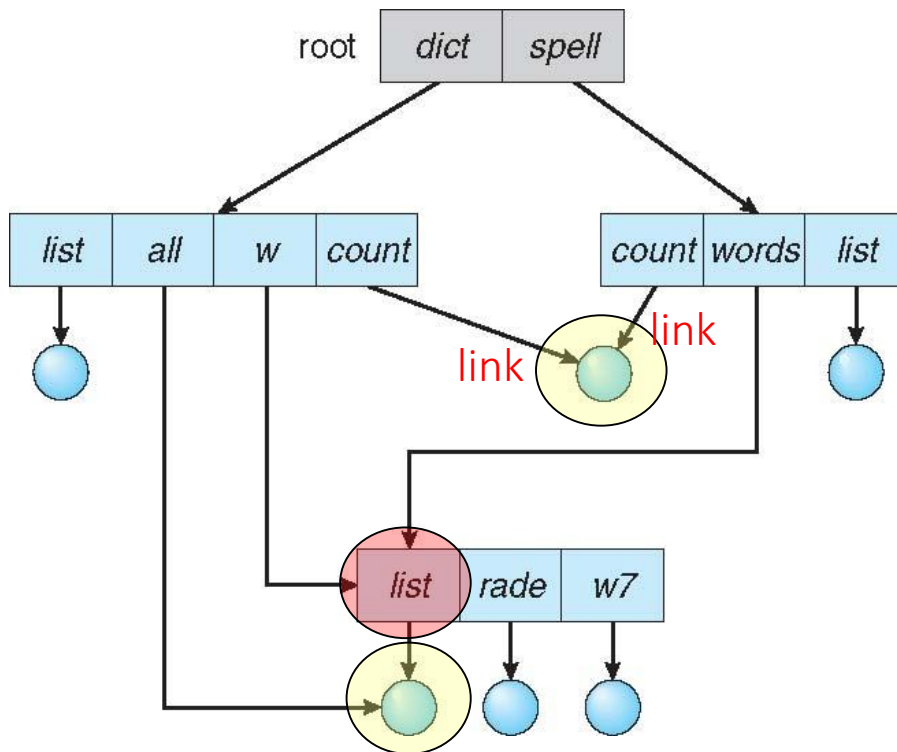
Tree-Structured Directories

- Current directory
- Absolute path name
- Relative path name

```
C:\Users\irlxi>cd documents
C:\Users\irlxi\Documents>mkdir test1
C:\Users\irlxi\Documents>mkdir test2
C:\Users\irlxi\Documents>cd test1
C:\Users\irlxi\Documents\test1>cd ../test2
C:\Users\irlxi\Documents\test2>cd \test1
系统找不到指定的路径。
C:\Users\irlxi\Documents\test2>cd C:\Users\irlxi\documents\test1
C:\Users\irlxi\Documents\test1>_
```

Acyclic-Graph Directories

- A tree structure prohibits the sharing of files or directories.
- An **acyclic graph** (a graph with no cycles) allows directories to share **subdirectories** and **files**
- A file may have **multiple different path names**
- Some OSes simply do not allow shared directories or links

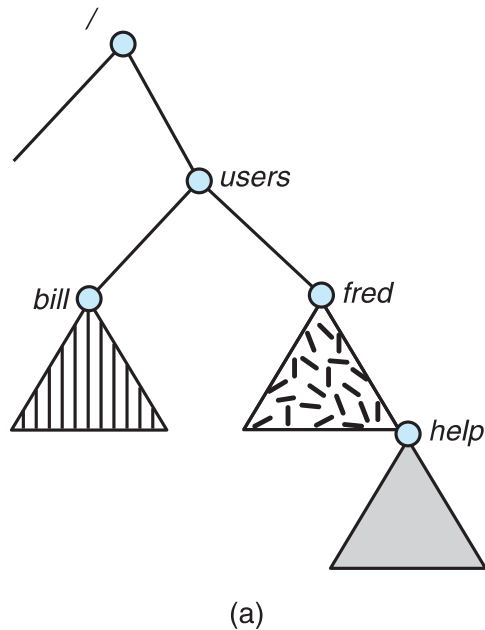


How do we guarantee no cycles?

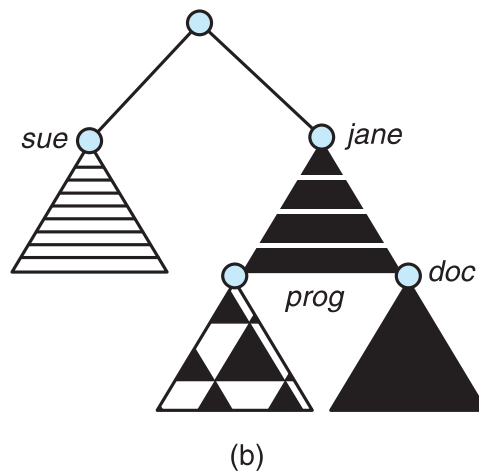
1. Allow **only links to file not subdirectories**
2. Every time a new link is added, use a **cycle detection algorithm** to determine whether it is OK.
Very expensive.

File System Mounting

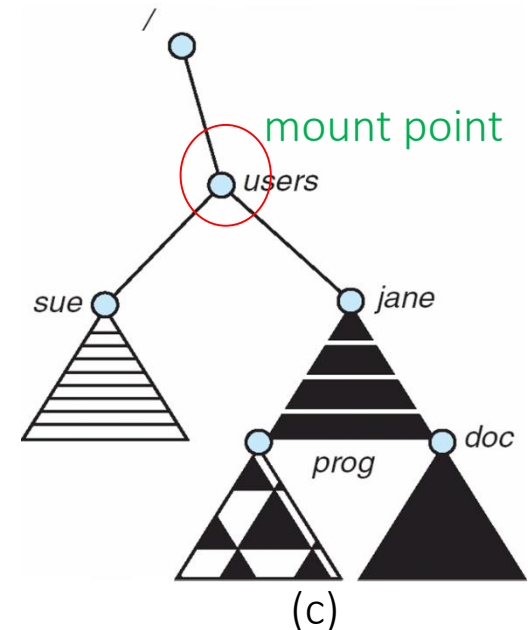
- A **file system** must be mounted before it can be accessed
- An unmounted file system can be mounted at a **mount point**
- The **mount point** must be **an empty directory**, i.e., original file system at the mount point must be removed if exists



Existing system

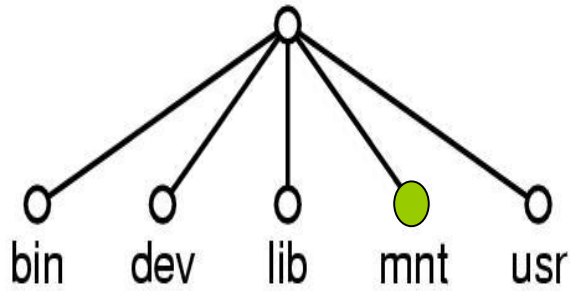


Unmounted volume

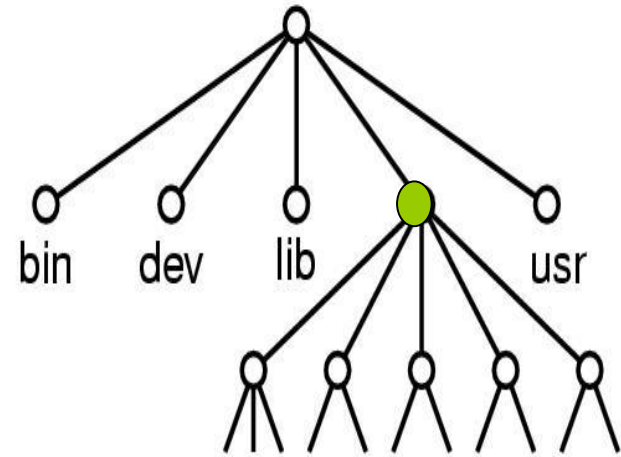


Mounted volume

Mount Point: Another Example



(a)



(b)

File Sharing

- A **file** can be shared by multiple users
- Access might be protected
- The **owner** of a file is the user who can change attributes and grant access and who has the most control over the file.
- The **group** attribute defines a subset of users who can share access to the file
- The **owner and group IDs** of a given file (or directory) are stored as part of file attributes
 - **User IDs** identify users, allowing permissions and protections for a user
 - **Group IDs** allow users to be in groups, permitting group access rights

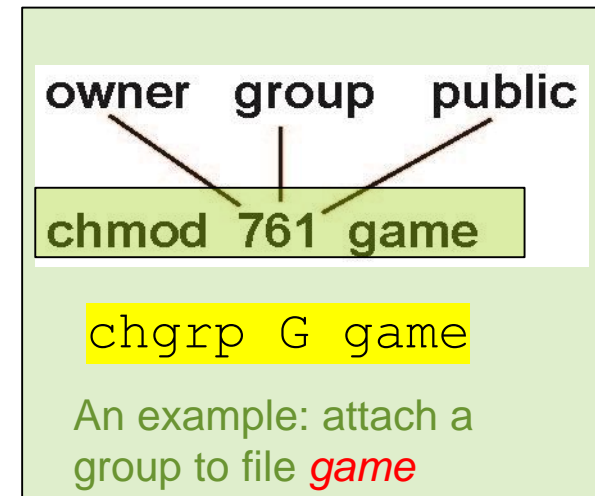
Protection

- When information is stored in a computer system, we want to keep it safe from **physical damage** (the issue of **reliability**) and **improper access** (the issue of **protection**)
- File **owner/creator** should be able to control:
 - what can be done
 - by whom
- Types of access
 - ▶ Read: Read from the file.
 - ▶ Write: Write or rewrite the file.
 - ▶ Execute: Load the file into memory and execute it.
 - ▶ Append: Write new information at the end of the file.
 - ▶ Delete: Delete the file and free its space for possible reuse.
 - ▶ List: List the name and attributes of the file
- Other operations, such as **renaming**, **copying**, and **editing** the file, are high levels, done through read, write access.

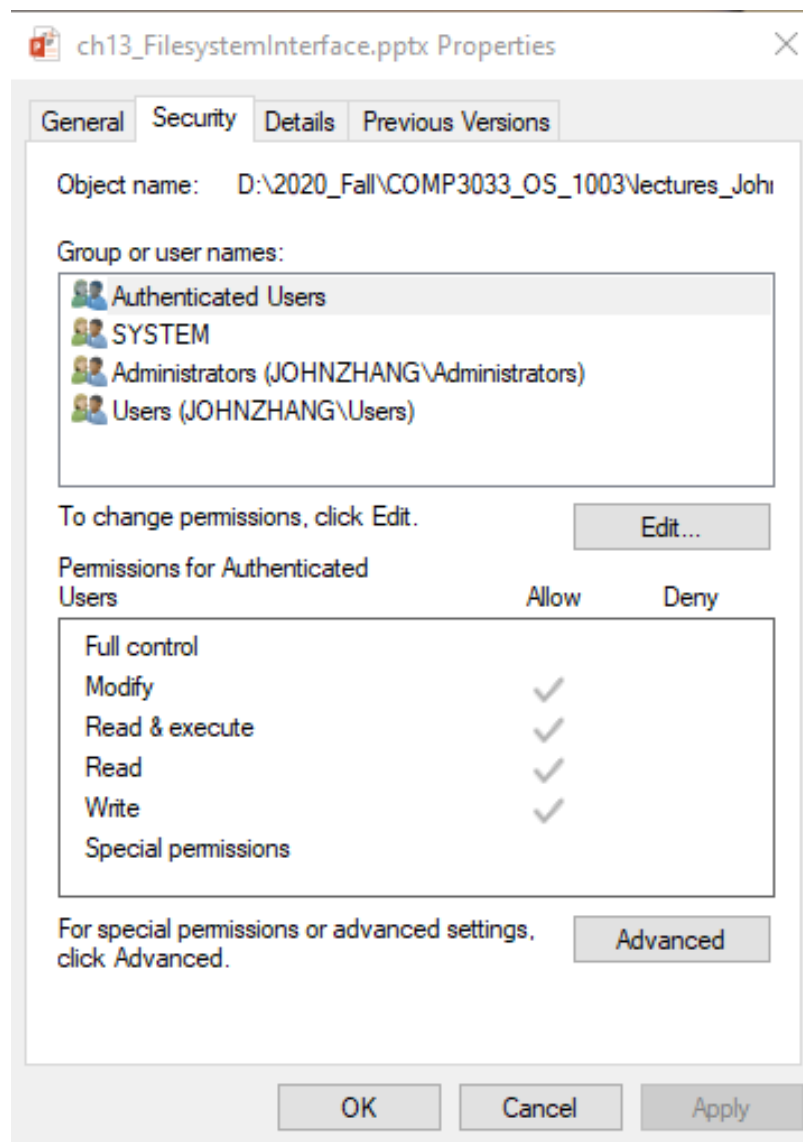
Access Lists and Groups

- Each file and directory has an **access-control list (ACL)**
- lists user names and the types of access allowed for each user.
- Three modes of access
 - Read (R),
 - Write (W)
 - Execute (X)
- The UNIX system defines three fields (owner, group, public) of three bits (**RWX**)
- Three classes of users on Unix / Linux

			RWX
creator	a) owner access	7 \Rightarrow	1 1 1
			RWX
A set of users who are sharing the file and need similar access	b) group access	6 \Rightarrow	1 1 0
			RWX
All other users in the system	c) public access	1 \Rightarrow	0 0 1



Windows 10 Access-Control List Management



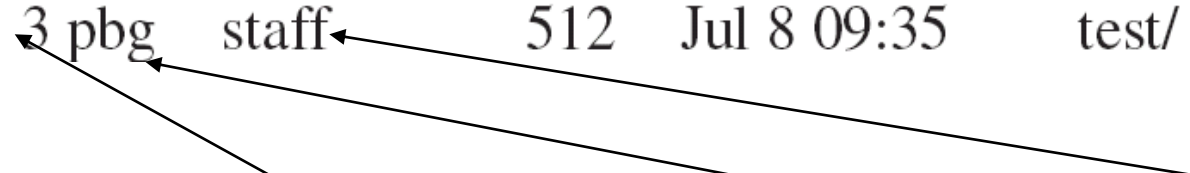
A Sample UNIX Directory Listing

Three fields:

owner, group, universe

each **three bits**: RWX

- 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/



Also shown are the number of links (shared) to the file, **the owner's name**, the group's name, the size of the file in bytes, the date of last modification, and finally the file's name (with optional extension).

Remote File Sharing

- Four implementation methods for remote file sharing
 - **Manually transferring** files between machines via programs like **ftp** (file transfer protocol)
 - ▶ both anonymous and authenticated access
 - Use a **distributed file system (DFS)**, in which remote directories are visible from a local machine
 - ▶ tighter integration between the machine
 - The **World Wide Web**, is a reversion to the first. Use a browser to gain access to the remote files, and separate operations (essentially a **wrapper for ftp**)
 - ▶ anonymous access
 - **Cloud computing**
 - ▶ delivers computing, storage, and even applications as a service across a network.

End of Chapter 13

