COP 4225 Advanced Unix Programming

File Systems

Chi Zhang

czhang@cs.fiu.edu

File Attributes

- **Name** – only information kept in human-readable form.
- **Type**
  - Extension-based
  - Magic numbers stored at the beginning of a file (Unix)
- **Location** – pointer to file location on device.
- **Size** – current file size.
- **Time**, **date**, **and owner identification**
- **Protection** – controls who can do r/w/x.
- Information about files are kept in the directory structure, which is maintained on the disk.

File Operations

- Create / Delete
- Truncate: erase but keep attributes (except the length
- Open(F_i) – move the content from disk to memory.
- Write / Read /Seek
- Close (F_i) – move the content in memory to directory structure on disk.

Tree-Structured Directories

- Current directory (working directory)
  - **cd** /spell/mail/prog
  - **type** list
- **Absolute** or **relative** path name
- Creating a new file is done in current directory.
- Delete a file: **rm** <file-name>
- Creating a new subdirectory is done in current directory.

  mkdir <dir-name>

Acyclic-Graph Directories

- Directories can have shared subdirectories and files.
  - A link is a pointer to another file or subdirectory
- Symbolic link
  - Link deletion does not affect the original file
  - File deletion leaves the links dangling
- Hard links
  - A reference count is kept with the file.
  - Preserve the file until the count is zero

File System Mounting

- The directory structure can be built out of multiple partitions (file systems)
- A file system must be **mounted** before it can be accessed.
- The root partition (contains OS) is mounted at boot time.
- A unmounted file system is mounted at a **mount point** (typically an empty dir).
- Traversing the directory structure switches among file systems

File Sharing and Protection

- Sharing of files on multi-user systems is desirable.
- Sharing may be done through a *protection* scheme.
- File owner/creator should be able to control: what can be done by whom
- Types of access
  - R/W/X
  - Append / Delete / List

Access Lists and Groups

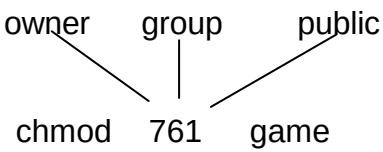
- Three classes of users

			RWX
a) **owner access**	7	⇒	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.
  - A user can join multiple groups.
- For a particular file (say *game*) or subdirectory, define an appropriate access.

Attach a group to a file

chgrp G game



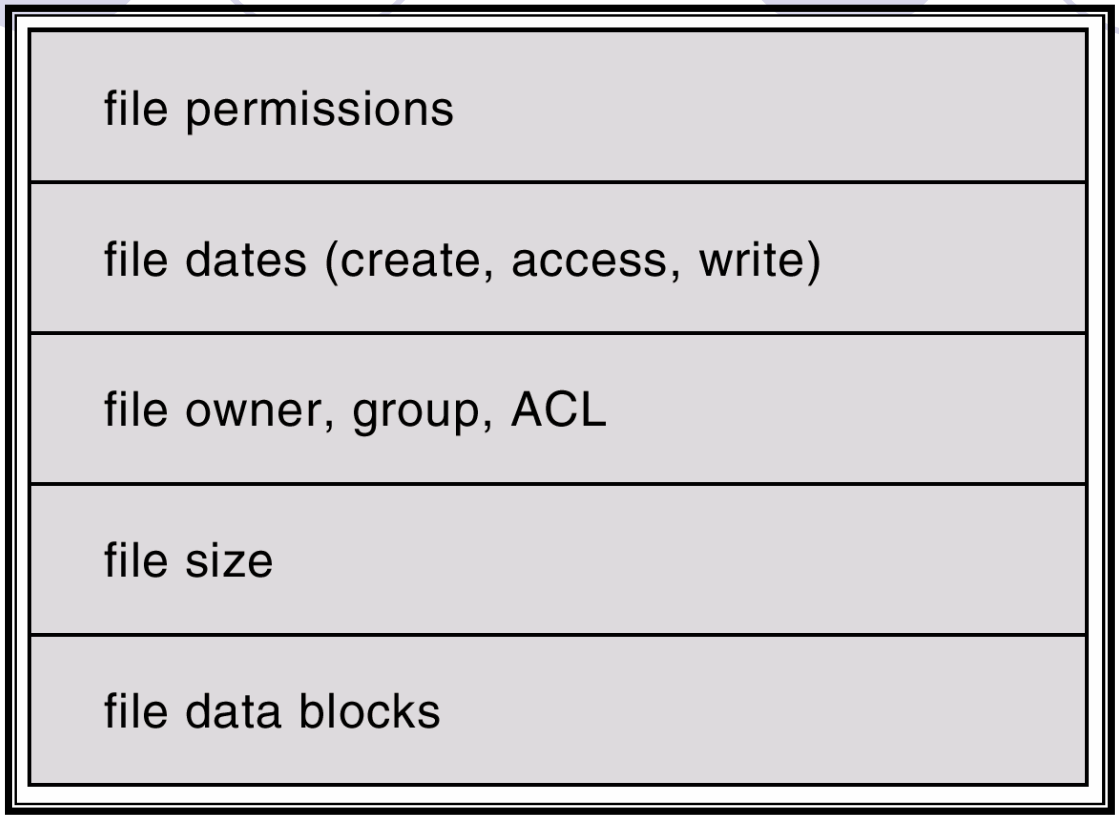
File-System Structure

- I/O transfer between memory and disk are performed in blocks.
  - Each block is one or more sectors. (e.g. 512 bytes)
- Map the logical file system onto the physical disks
- Allocate / Release disk blocks.

On-Disk File Structure

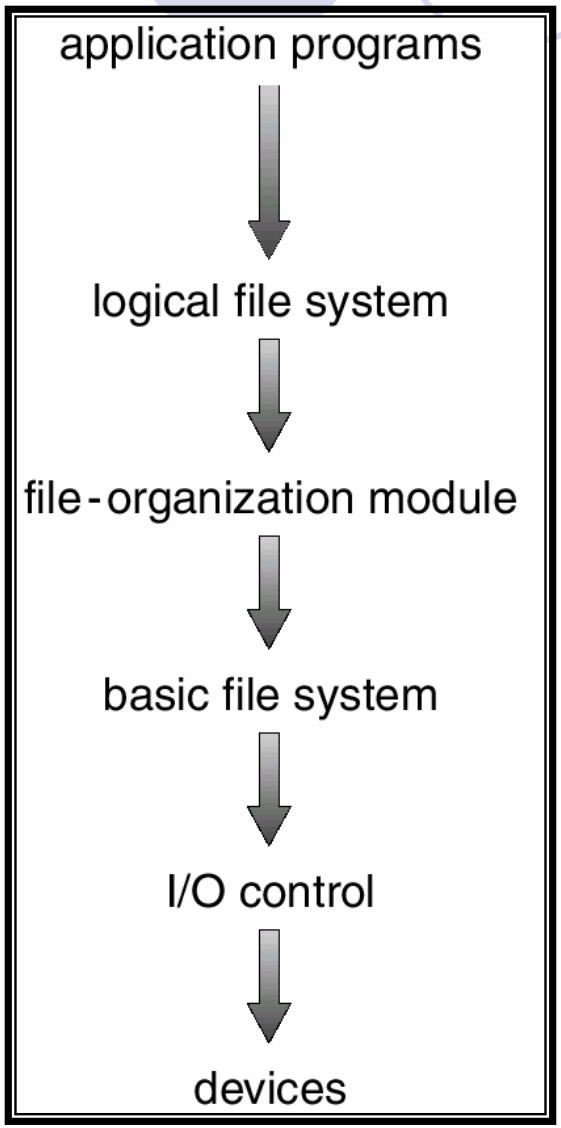
- Boot control block: How to boot the OS
- Partition control block (Superblock):
  - partition details, number of blocks, free block count …
- Directory structure
- *File control block* – storage structure consisting of information about a file.

A Typical File Control Block



In Unix, a directory is treated exactly as a file, with a type field indicating it is a directory.

Layered File System



- **I/O control**: device drivers and interrupt handler.
  - E.g. “retrieve cylinder 73, track 2, sector 10”
- **Basic file system** (device independent): issue generic commands to the appropriate device driver to r/w blocks on disks.
  - E.g. “retrieve drive 1 block 123”
- **File organization module**: translate logical blocks of a file into physical block addresses; disk free-space manager (e.g. append).
  - E.g. “ retrieve the i^{th} logical block of *file”*.
- **Logical file system:** manage file system structure and directory structure. Protection and security.
- The I/O control and the basic file system code can be shared by multiple file systems

In-Memory File System Structures

- In-memory partition table about each mounted partition
- In memory directory structure
  - Recently accessed directories
  - For directories of mount point, a pointer to an entry in the partition mounting table.
- System-wide open-file table
- Per-process open-file table

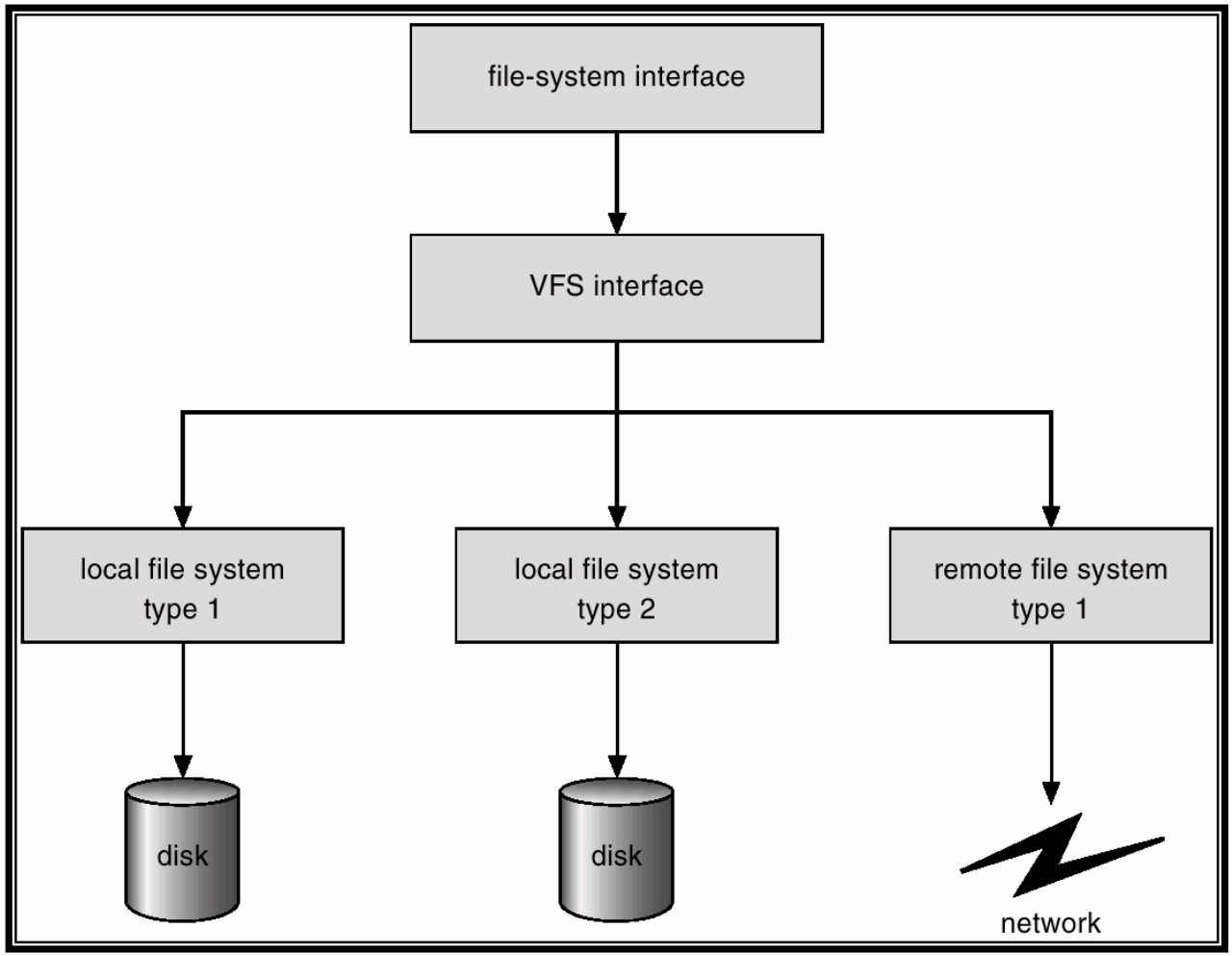
File Operations

- The open system call will return a pointer to the entry in the open-file table, which is used in all subsequent I/O operations.
- Per-process tables tracks all files that a process has open.
  - file pointer (for sequential access), Access mode …
- Each entry in the per-process table has a pointer that points to a process-independent entry in the system-wide open-file table.
  - File open count, FCB (Disk location …)
  - When the count is 0, the updated file information is copies back to disk directory structure.

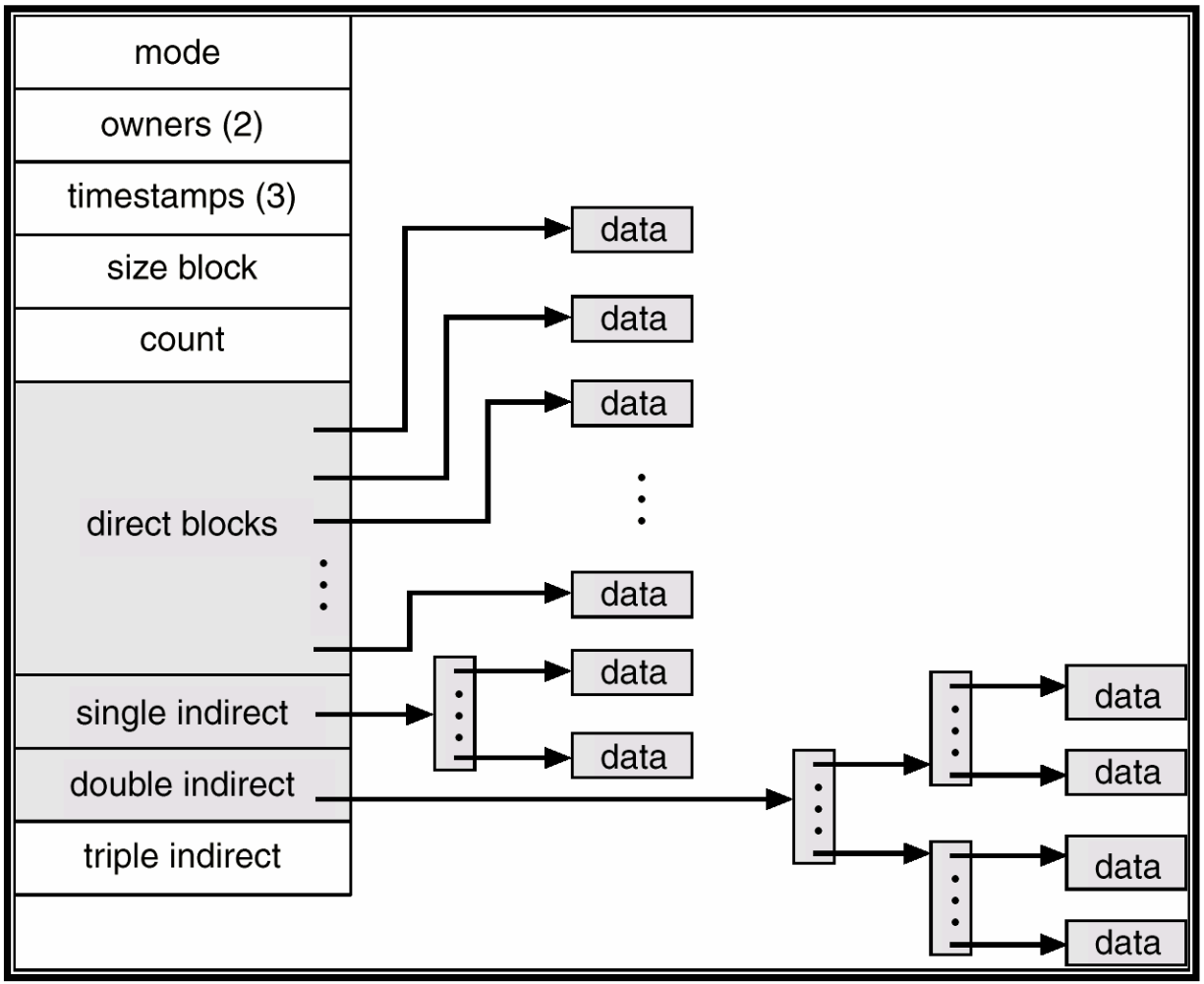
Virtual File Systems

- Virtual File Systems (VFS) provide an object-oriented way of implementing file systems.
- VFS allows the same system call interface (the API) to be used for different types of file systems.
- The API is to the VFS interface, rather than any specific type of file system.

Schematic View of Virtual File System

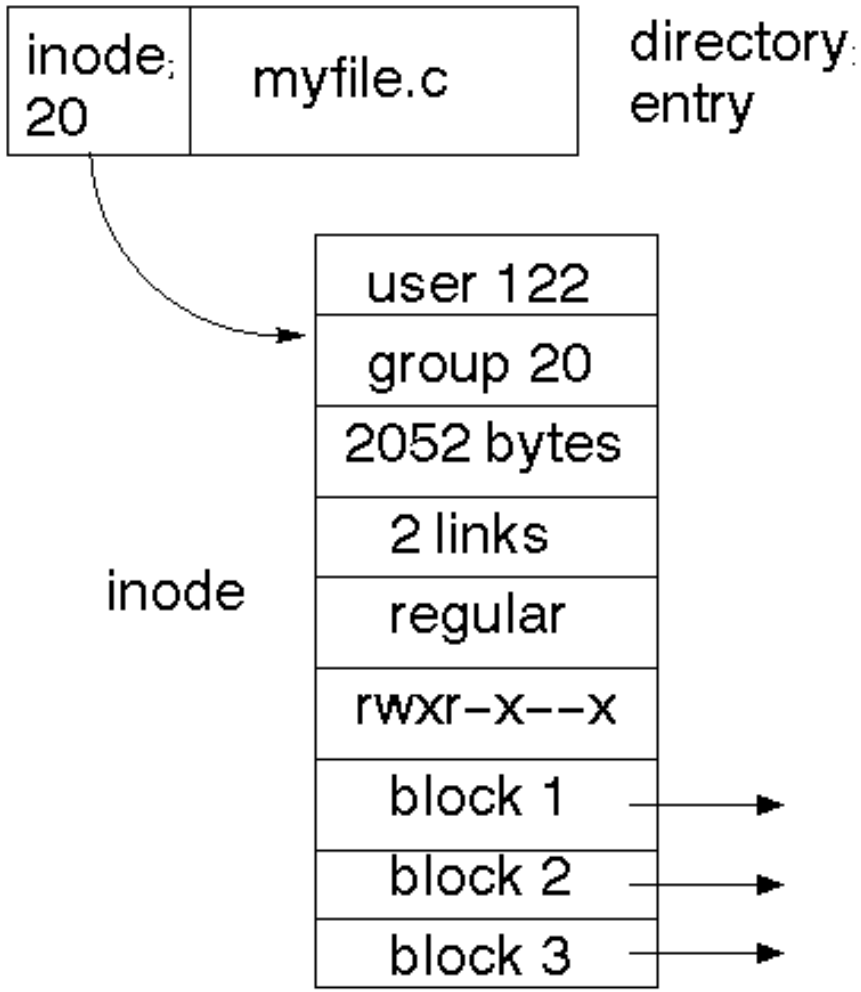


Indexed Allocation: UNIX (4K bytes per block)



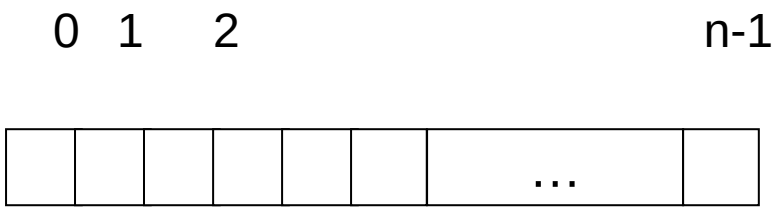
Pre-allocated inodes.

Unix Directory Implementation



Free-Space Management

- Bit vector (*n* blocks)



$$\text{bit}[i] = \begin{cases} 0 \Rightarrow \text{block}[i] \text{ free} \\ 1 \Rightarrow \text{block}[i] \text{ occupied} \end{cases}$$

Free-Space Management (Cont.)

- Easy to find the first *n* consecutive free blocks
- Bit map requires extra space. Example:

  block size = 2^{12} bytes

  disk size = 2^{30} bytes (1 gigabyte)

  $n = 2^{30}/2^{12} = 2^{18}$ bits (or 32K bytes)
- Bit Vectors are inefficient unless the entire vector is kept in main memory.
  - Written to disk occasionally for recovery needs

Free-Space Management (Cont.)

- Bit map
  - Copy in memory and disk may differ (e.g. power off).
  - Cannot allow for block[*i*] to have a situation where bit[*i*] = 1 in memory (allocated) and bit[*i*] = 0 on disk.
- Solution:
  - Set bit[*i*] = 1 in disk.
  - Allocate block[*i*] (what if outage in the middle?)
  - Set bit[*i*] = 1 in memory

Performance

- Performance
  - On-board cache in the disk to buffer a track when a sector is requested
  - disk cache – separate section of main memory for frequently used blocks
    - Asynchronous data writes
  - LRU disk cache for random accesses
  - free-behind and read-ahead – techniques to optimize sequential access

Recovery

- Consistency checking – compares data in directory structure with data blocks on disk, and tries to fix inconsistencies.
- Memory is more up-to-date than the disk because of disk cache
  - Changes in metadata (space allocation, inodes) are written to the disk synchronously, before the data blocks are written.
- Use system programs to *back up* data from disk to another storage device (floppy disk, magnetic tape).