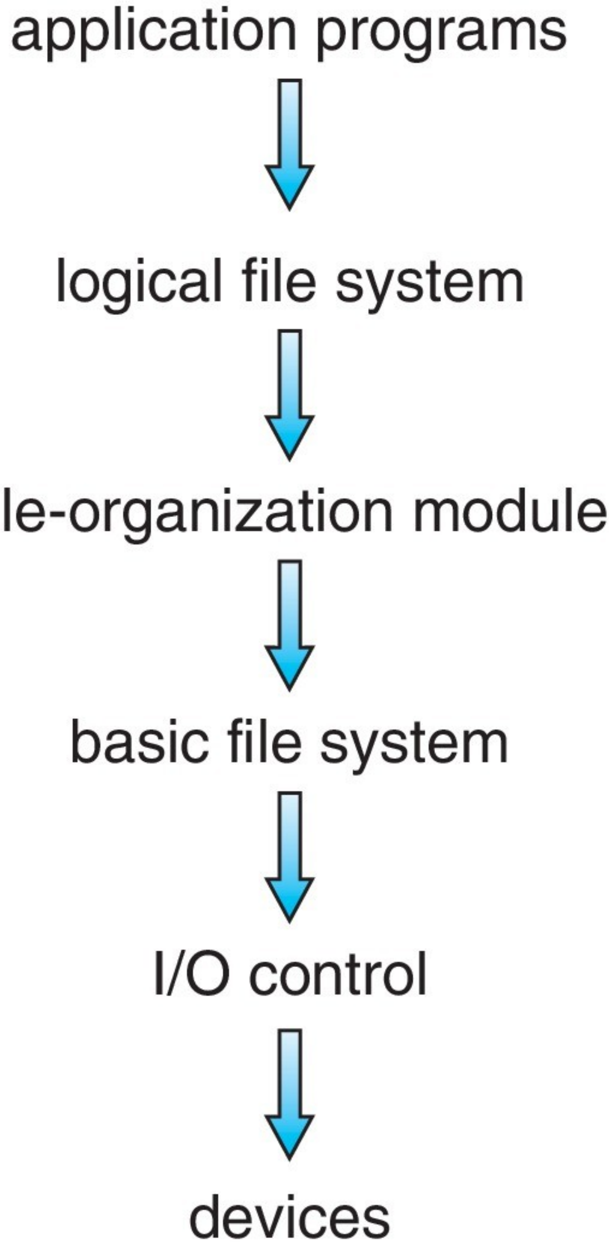
File System

File-System Structure

- File structure
  - Logical storage unit
  - Collection of related information
- **File system** resides on secondary storage (disks)
  - Provided user interface to storage, mapping logical to physical
  - Provides efficient and convenient access to disk by allowing data to be stored, located retrieved easily
- Disk provides in-place rewrite and random access
  - I/O transfers performed in **blocks** of **sectors** (usually 512 bytes)
- **File control block (FCB)** – storage structure consisting of information about a file
- **Device driver** controls the physical device
- File system organized into layers

Layered File System



application programs

logical file system

file-organization module

basic file system

I/O control

devices

File System Layers

- **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

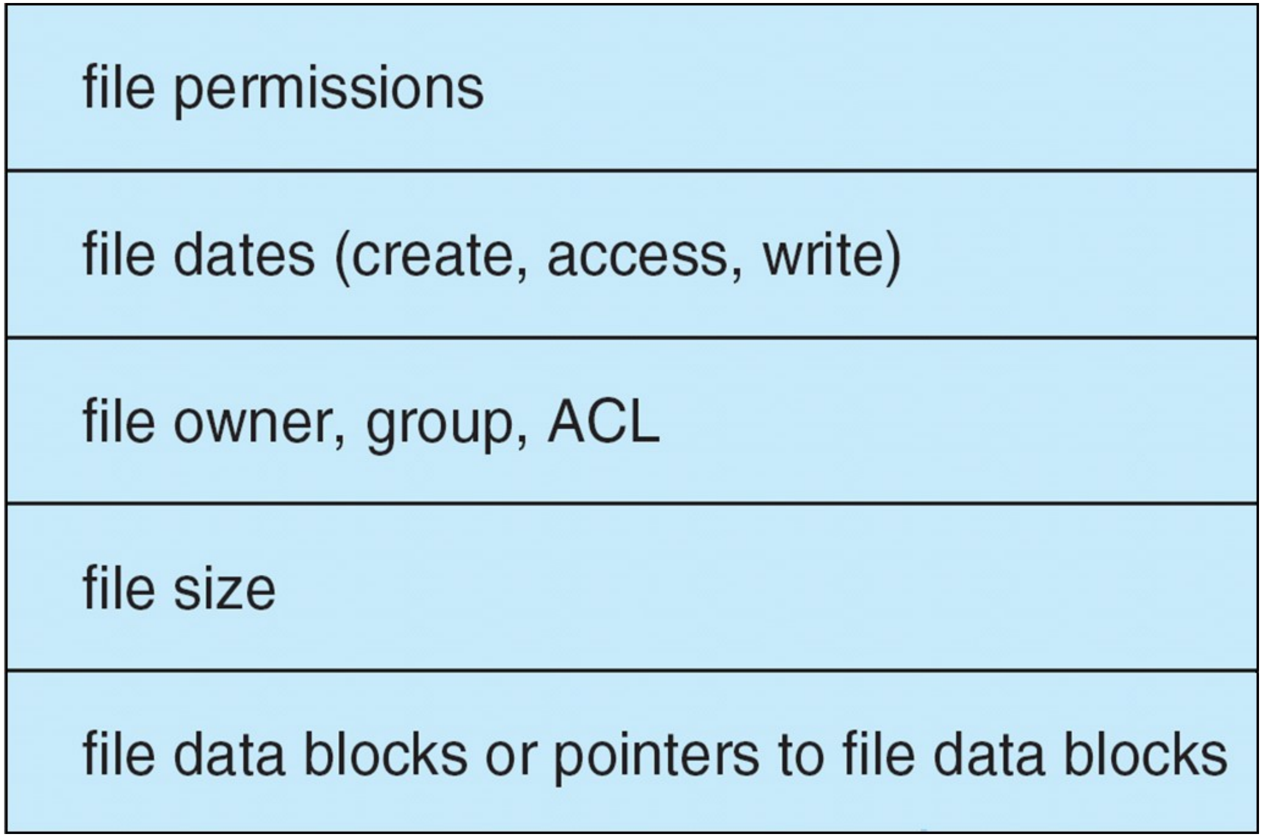
- **Logical file system** manages metadata information
  - Translates file name into file number, file handle, location by maintaining file control blocks (**inodes** in UNIX)
  - Directory management
  - Protection
- Layering useful for **reducing complexity** and **redundancy**, but adds overhead and can decrease performance.
- Logical layers can be implemented by any coding method according to OS designer

File-System Operations

- We have system calls at the API level, but how do we implement their functions?
  - On-disk and in-memory structures
- **Boot control block** contains info needed by system to boot OS from that volume
  - Needed if volume contains OS, usually first block of volume
- **Volume control block (superblock, master file table)** contains volume details
  - Total # of blocks, # of free blocks, block size, free block pointers or array
- Directory structure organizes the files
  - Names and inode numbers, master file table

File-System Implementation

- Per-file **File Control Block (FCB)** contains many details about the file
  - Typically inode number, permissions, size, dates
  - NFTS stores into in master file table using relational DB structures



file permissions

file dates (create, access, write)

file owner, group, ACL

file size

file data blocks or pointers to file data blocks

Directory Implementation

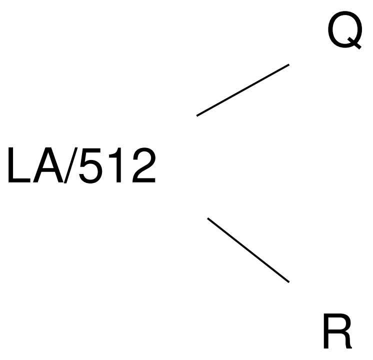
- **Linear list** of file names with pointer to the data blocks
  - Simple to program
  - Time-consuming to execute
    - Linear search time
    - Could keep ordered alphabetically via linked list.
- **Hash Table** – linear list with hash data structure
  - Decreases directory search time
  - Collisions – situations where two file names hash to the same location
  - Only good if entries are fixed size, or use chained-overflow method

Allocation Methods- Contiguous

- An allocation method refers to how disk blocks are allocated for files:
- Each file occupies set of contiguous blocks
  - Best performance in most cases
  - Simple – only starting location (block #) and length (number of blocks) are required
  - Problems include:
    - Finding space for file,
    - Knowing file size,
    - External fragmentation, need for compaction off-line (downtime) or on-line

Contiguous Allocation

- Mapping from logical to physical (block size =512 bytes)

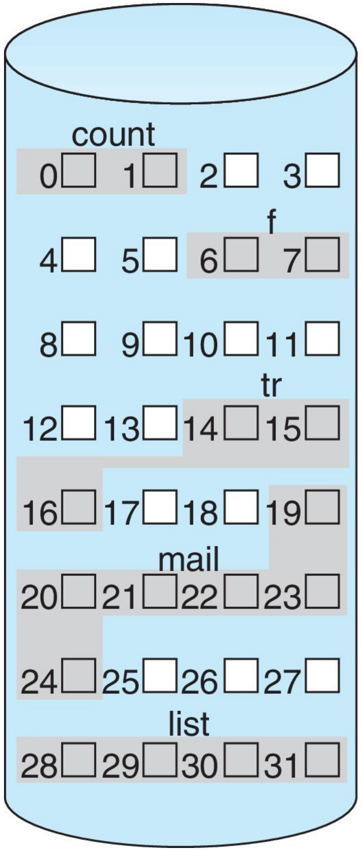


Q

LA/512

R

- Block to be accessed = starting address + Q
- Displacement into block = R



directory

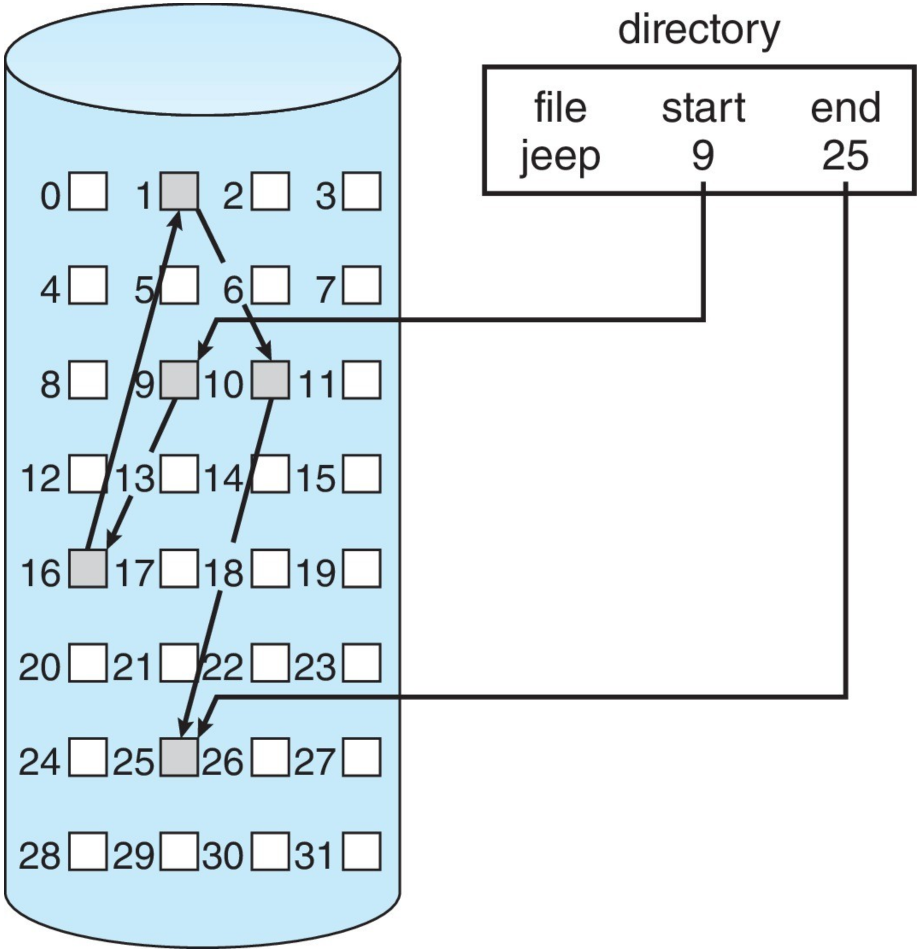
file	start	length
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2

Allocation Methods- Linked

- Each file a linked list of blocks
- File ends at nil pointer
- No external fragmentation
- Each block contains pointer to next block
- No compaction, external fragmentation
- Free space management system called when new block needed
- Improve efficiency by clustering blocks into groups but increases internal fragmentation
- Reliability can be a problem
- Locating a block can take many I/Os and disk seeks.

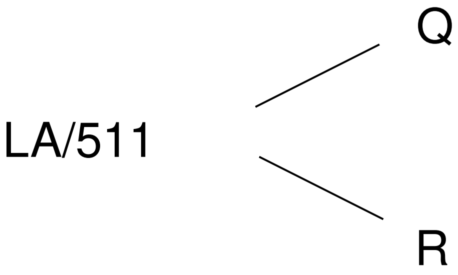
Linked Allocation

- Each file is a linked list of disk blocks: blocks may be scattered anywhere on the disk
- Scheme



Linked Allocation

- Mapping



- Block to be accessed is the Qth block in the linked chain of blocks representing the file.
- Displacement into block = R + 1