Virtualization and File System Management

Module 6

Dr. Naveenkumar J

Associate Professor,

PRP- 217 - 4

File System - Definition

- A file system is a structure and set of rules that dictates how data is stored, organized, and managed on storage devices like hard drives, SSDs, and USB drives.
- Without a file system, the OS would see a storage device as a single, undifferentiated block of data, making it impossible to distinguish between different files.

File System - Definition

- A file system acts as an index for all the data on a storage device, allowing users and applications to create, read, update, and delete files in an organized manner.

File System - Definition

- **Files**
  - A file is a named collection of related information, such as a document, program, or image, recorded on storage. File systems define conventions for naming files, including length and character limitations
- **Directories**
  - Also known as folders, directories are used to group files and other directories. This creates a hierarchical structure, with the "root" directory at the top.

File System - Definition

- **Partitions**
  - Before a file system can be used, a storage device is typically divided into one or more partitions.
  - Each partition is a distinct region of storage that the OS manages separately, and each can be formatted with a different file system.
  - This separation can improve performance, security, and data integrity.

File System - Definition

- **Metadata**
  - Along with the actual data in a file, the file system stores metadata.
  - This includes information such as the file's name, size, creation date, access permissions, and its location within the directory structure.

File System Interface

- The file system interface defines how the operating system presents files and directories to the user and how applications can interact with them.

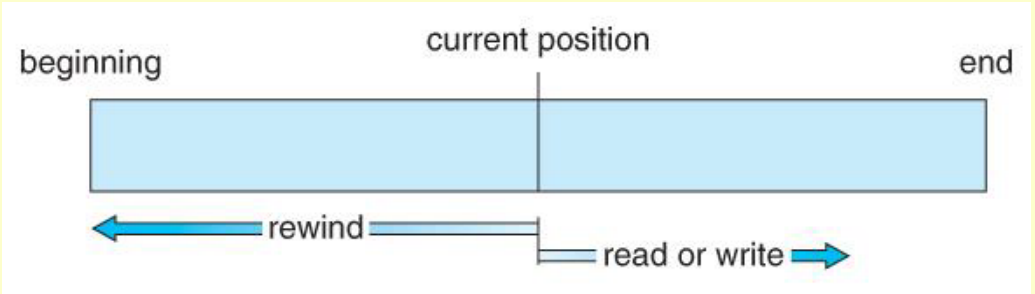
- It abstracts the physical properties of storage devices into a logical storage unit

File System – Access Methods

- Access methods determine how the information within a file can be accessed and read.

- **Sequential Access**
  - This is the most common method.
  - Information in the file is processed in order, one record after another.
  - A read operation advances a file pointer to the next position, and a write operation appends to the end of the file or overwrites subsequent data.

- A sequential access file emulates magnetic tape operation, and generally supports a few operations:
  - Readnext – read a record and advance the tape to the next position.
  - writenext-write a record and advance the tape to the next position.
  - Rewind
  - Skip n records – May or may not be supported. N may be limited to positive numbers or may be limited to +/-1.



File System – Access Methods

- **Direct Access** Also known as relative access,
  - This method allows a program to read or write information from a file in any order, without reading from the beginning.
  - The file is viewed as a numbered sequence of blocks or records. This is useful for database applications where data needs to be accessed rapidly and in a non-linear fashion.

- Jump to any record and read that record. Operations supported include:
  - read n - read record number n. (Note an argument is now required.)
  - write n - write record number n. (Note an argument is now required.)
  - jump to record n - could be 0 or the end of file.
  - Query current record - used to return back to this record later.
  - Sequential access can be easily emulated using direct access. The inverse is complicated and inefficient.

sequential access	implementation for direct access
reset	cp = 0;
read_next	read cp ; cp = cp + 1;
write_next	write cp; cp = cp + 1;

File System – Access Methods

- **Other Methods**
  - More complex access methods can be built on top of direct access.
  - One common approach is to create an index for the file.
  - The index, like an index in a book, contains pointers to various blocks in the file, allowing data to be found quickly without searching the entire file