Assignment - 2

Name : Ria J. Tambve
Roll No : 242011057

1) Explain different file access methods

→ File access methods describe how data is retrieved from files stored on disk. Primary access methods are :

1. Sequential Access :

- Data is accessed sequentially, one record after the other. This is the simplest access method and is suitable for tasks like reading log files.

Eg : read_next() , write_next()

2. Direct Access (Random Access) :

Data is accessed directly using a specific offset or position in the file. This method is ideal for applications that require frequent updates or retrieval from known locations such as databases.

Eg : read_at(x) , write_at(x)

3. Indexed Access :

An index is maintained for file, mapping key values to locations. This method allows for quick retrieval based on search keys, commonly used databases.

Eg : Searching record using an index table.

4. Memory-mapped file access:
Maps a file into the processe's virtual memory space, allowing applications to access it as if it were a part of memory. This is useful for performance critical applications.

Eg: mmap() in UNIX / LINUX

2) Explain file system structure.

→ The file system organizes and manages how data is stored and retrieved.
It's structure includes:

1. Boot Control Block: Contains metadata for booting the operating system, usually located at a fixed block

Eg: MBR in Windows

2. Volume Control Block: Stores details about a specific volume, such as total blocks, free blocks, block size and file system type.

3. Directory Structure: Manages file names and hierarchical organization of files within directories.

Eg: Single level, multi-level or tree based directory structures.

4. File Control Block : Contains metadata about files including file size, permissions, timestamps and locations of data blocks.

5. Data Blocks :
The actual storage locations for content of files.

3) How to do failure analysis ? Explain about OS performances tuning.

→ Failure Analysis :
Failure Analysis helps determine the root cause of system or application crashes.
Steps include :

1. Collecting logs : System logs, application logs or crash dumps.
2. Analyzing core dumps : Inspecting the memory and process state at the time of failure.
3. Tracing execution : Using tools like strace or dtrace to trace system calls.
4. Identifying bottleneck : Monitoring CPU, memory disk and network usage
5. Reproducing the issue : Simulating the failure in a controlled environment

OS Performance Tuning :
Performance tuning involves optimizing the operating system for better performance.

Examples include :

1. CPU tuning - Adjusting scheduling algorithms,

assigning process priorities

2. Memory tuning: Managing virtual memory, page sizes and swap space efficiently

3. Disk tuning: Optimizing disk I/O, caching and defragmentation

4. Network Tuning: Modifying socket buffer sizes or TCP configurations

5. Kernel Optimization: Adjusting kernel parameters for better resource management

4) Explain principles of OS performance tuning.

→ Performance Tuning focuses on minimizing system efficiency for a specific application set principles include:

1. Identify Bottlenecks:
Use profiling tools to determine resource intensive operations.

Eg: High CPU usage, memory leaks or excessive I/O

2. Set Realistic goals:
Define clear performance objectives such as reducing latency or improving throughput

3. Optimize for workload:
Tune the system for most critical applications

Eg: Databases, web servers

4. Use efficient algorithms :
Replace inefficient algorithms in software or kernel with optimized ones.

5. Balance Resources :
Ensure even distribution of CPU, memory, disk and network usage to avoid overloading specific components.

6. Test and Validate :
Continuously monitor performance metrics and validate improvements under realistic workloads

7. Iterative tuning :
Performance optimization is an iterative process, analyze, implement charges, test and repeat.