HA-4

1. Analyze the significance of striping and mirroring in RAID configurations, and how do they impact performance and data protection.

- **Striping (RAID 0)**: Faster, no redundancy, total failure if one disk fails.
- **Mirroring (RAID 1)**: Redundant, slower writes, better fault tolerance.
- **RAID 10**: Combines both for speed and redundancy.

2. Illustrate the role of file systems in managing files and directories in an operating system. Demonstrate how file access permissions, metadata, and indexing structures contribute to efficient file retrieval and security.

Role of File Systems in OS

- Manages Files & Directories: Organizes, stores, and retrieves data efficiently.
- File Access Permissions: Controls read, write, and execute rights for security.
- Metadata: Stores file attributes (size, type, timestamps) for quick access.
- Indexing Structures: Uses inodes, B-trees, or FAT for fast file retrieval.

3. Apply thread synchronization using semaphores in a scenario where multiple threads access a shared resource.

Semaphore Example: Shared Printer

```
sem_t printer;
sem_init(&printer, 0, 1);

void* print(void* id) {
   sem_wait(&printer);
   printf("Thread %d printing...\n", *(int*)id);
   sleep(2);
   sem_post(&printer);
}
```

- sem_wait(): Locks printer.
- sem_post(): Releases printer.
- Ensures one thread prints at a time.

4. Demonstrate how the buffer cache in a UNIX environment enhances file system performance by applying your knowledge of cache management strategies.

Buffer Cache in UNIX for File System Performance

- Caching Mechanism: Stores frequently accessed disk blocks in memory, reducing disk I/O.
- Write-back Strategy: Writes data to cache first, delaying disk writes for efficiency.
- LRU (Least Recently Used): Removes least-used blocks to make space for new ones.
- Read-ahead: Preloads expected data to speed up sequential file access.
- Improvement: Reduces latency, enhances throughput, and optimizes disk performance.

5. Apply the principles of embedded systems to design a basic real time controller for an automotive application.

Real-Time Engine Temp Controller

```
void* monitor(void* arg) {
  while (1) {
    int temp = rand() % 120;
    printf("Temp: %d°C\n", temp);
    if (temp > 100) printf("Overheating!\n");
    sleep(1);
  }
}
```

- Monitors engine temp in real-time.
- Alerts if temp > 100°C.
- Ensures timely response.

6. Apply the concepts of contiguous, linked, and indexed file allocation techniques to illustrate their differences in a storage system

File Allocation Techniques

1. **Contiguous Allocation**
   - Files stored in consecutive blocks.
   - Fast access, but fragmentation occurs.
   - Example: Hard disk partitions.

2. **Linked Allocation**
   - Each file block points to the next.
   - No fragmentation, but slow access.
   - Example: FAT file system.

3. **Indexed Allocation**
    - Uses an index block to track file blocks.
    - Fast access, but extra metadata overhead.
    - Example: ext4, NTFS.

7. Identify the steps involved in building a distributed file system. Give Distributed File System Architecture and explain Sun's Network File System (NFS).

Building a Distributed File System (DFS)

1. Define requirements.
2. Design architecture (centralized, peer-to-peer).
3. Implement data distribution.
4. Manage naming & directories.
5. Ensure consistency & security.
6. Deploy and optimize.

DFS Architecture

- Client-Server Model: Clients request files from servers.
- Replication & Caching: Improves performance & fault tolerance.

Sun's NFS

- Uses RPC for remote access.
- Stateless protocol for efficiency.
- Supports mounting for seamless integration.

8. Explain how virtual devices are used to abstract physical hardware.

Virtual Devices & Hardware Abstraction

- Definition: Virtual devices emulate physical hardware using software.
- Abstraction: OS interacts with virtual devices instead of direct hardware.
- Examples:
    - Virtual Network Adapters – Simulate physical NICs.
    - Virtual Disks – Act as real storage (e.g., VHD, VMDK).
    - GPU Virtualization – Allocates graphics resources in VMs.
- Benefits: Portability, isolation, resource efficiency, and flexibility.

9. Construct a scenario where dedicated and shared devices are used in a multitasking environment and demonstrate the differences in resource allocation.

Scenario: Print Server in a Multitasking Environment

- Dedicated Device (Printer)
  - Only one process prints at a time.
  - Example: A user submits a document; others wait in a queue.
- Shared Device (Hard Disk)
  - Multiple processes read/write simultaneously.
  - Example: Several users access and modify files concurrently.

Differences in Resource Allocation

Feature	Dedicated Device (Printer)	Shared Device (Hard Disk)
Access	One process at a time	Multiple processes
Queuing	Yes, waits in line	No queuing needed
Usage	Exclusive	Concurrent
Example	Printer, Scanner	Hard Disk, RAM

10. Identify and explain two methods for storing partition and boot information on a disk. Use your understanding to create a diagram of the UNIX file system layout, showing the disk drive, partitions, and the file system structure.

Partition & Boot Info Storage

1. MBR – First sector, bootloader + partition table (4 partitions).
2. GPT – UEFI-based, supports 128 partitions, better redundancy.

UNIX File System Layout

```
+------------------+  <-- Disk Drive
| Boot Sector      |  <-- MBR/GPT
+------------------+
| / (Root)         |  <-- OS Files
+------------------+
| /home            |  <-- User Data
+------------------+
| Swap             |  <-- Virtual Memory
+------------------+
```

11. Assume that the file system has been mounted and thus that the superblock is already in memory. Everything else (i.e., inodes, directories) is still on the disk. Illustrate the flow of operation during the activity of writing a file to disk.

Flow of Writing a File to Disk

1. User Request – A process requests file creation.
2. Allocate Inode – OS fetches a free inode from disk.
3. Update Directory – Directory entry maps filename to inode.
4. Allocate Data Blocks – OS assigns free disk blocks for file data.
5. Write Data – File content is written to allocated blocks.
6. Update Metadata – Modify inode (size, timestamps).
7. Commit to Disk – Buffer cache writes changes to disk asynchronously.

12. Give the Structure of VSFS(Very Simple File System) and explain data bitmap & inode bitmap. A VSFS file system has 4 KB blocks and 4-byte disk addresses. What is the maximum file size if i-nodes contain 12 direct entries, and one single, double, and triple indirect entry each.

VSFS Structure

1. Superblock – File system metadata.
2. Data Bitmap – Tracks free/used data blocks.
3. Inode Bitmap – Tracks free/used inodes.
4. Inode Table – Stores file metadata.
5. Data Blocks – Hold file contents.

Max File Size Calculation

- Direct (12 × 4KB) = 48KB
- Single Indirect (1024 × 4KB) = 4MB
- Double Indirect (1024^2 × 4KB) = 4GB
- Triple Indirect (1024^3 × 4KB) = 4TB

Total Max Size ≈ 4TB

13. Apply encryption techniques to protect data stored on a disk,and demonstrate how it ensures data security.

Disk Encryption for Data Security

1. Full Disk Encryption (FDE)
    - Encrypts entire disk (e.g., BitLocker, LUKS).
    - Protects data even if disk is stolen.
2. File-Level Encryption
    - Encrypts specific files (e.g., AES, GPG).
    - Ensures selective protection.

Example: AES Encryption

```
from cryptography.fernet import Fernet
key = Fernet.generate_key()
cipher = Fernet(key)
encrypted = cipher.encrypt(b"Sensitive Data")
print(encrypted)
```

- Ensures confidentiality & prevents unauthorized access.

14. Apply the concept of redundancy to design a simple fault tolerant network that continues to operate even when one node fails.'

Fault-Tolerant Network Design

1. Redundant Links – Use multiple paths (e.g., mesh topology) to prevent single points of failure.
2. Backup Nodes – Secondary nodes take over if the primary fails.
3. Load Balancing – Distributes traffic across multiple paths for reliability.
4. RAID for Storage – Ensures data redundancy in networked storage.

Example: Dual-Ring Network

- Primary Ring – Normal data flow.
- Secondary Ring – Activates if a node fails, maintaining connectivity.

15. Consider a scenario in which you have four hard drives of varying sizes. Describe how you would configure a RAID 10 (1+0) setup using these drives and explain the benefits of this configuration in terms of both performance and redundancy.

RAID 10 (1+0) Setup with Four Drives

Configuration:

1. Pair drives into two mirrored sets (RAID 1).
2. Stripe data across these mirrored pairs (RAID 0).

Example:

- Drives: 500GB, 1TB, 1TB, 500GB
- Mirrored Pairs: (500GB + 500GB) & (1TB + 1TB) → Limited by smallest size
- Usable Capacity: 1TB (500GB + 500GB)

Benefits:

- Performance: RAID 0 striping speeds up read/write operations. Redundancy: RAID 1 mirroring ensures fault tolerance (1 drive per pair can fail).

16. Assume that the file system has been mounted and thus that the superblock is already in memory. Everything else (i.e., inodes, directories) is still on the disk. Illustrate the flow of operation during the activity of writing a file to disk.

Flow of Writing a File to Disk

1. User Request – Process requests file creation.
2. Allocate Inode – OS assigns a free inode from disk.
3. Update Directory – Links filename to inode.
4. Allocate Data Blocks – Assigns free blocks for file data.
5. Write Data – Saves file content to blocks.
6. Update Metadata – Modifies inode (size, timestamps).
7. Commit to Disk – Buffer cache writes changes asynchronously.

17. Explain the role of I/O traffic control in operating systems and evaluate their impact on system performance.

Role of I/O Traffic Control in OS

1. Manages I/O Requests – Prioritizes and schedules disk, network, and device access.
2. Reduces Bottlenecks – Prevents congestion by optimizing resource allocation.
3. Ensures Fairness – Balances multiple processes' I/O needs.

Impact on System Performance

✅ Improves Throughput – Efficient scheduling reduces wait times.
✅ Reduces Latency – Prioritization ensures faster response for critical tasks.
✅ Prevents Starvation – Ensures all processes get I/O access fa