Task1

(1)

Spinning Disk (HDD):

- Mechanical parts (platters, read/write head).
- Slower data access (~100 MB/s).
- More prone to damage.
- Cheaper, larger capacity.

Solid-State Disk (SSD):

- No moving parts (flash memory).
- Faster data access (~500–7000 MB/s).
- Durable and energy-efficient.
- More expensive, smaller capacity.

(2)

Logical Block Addressing (LBA): A method of addressing blocks on a storage device, where each block is identified by a linear number instead of cylinder/head/sector.

Maximum Disk Size:

1. **24-bit LBA:**
   - Blocks: 224=16,777,2162^{24} = 16,777,216224=16,777,216
   - Assuming 512 bytes per block: 16,777,216×512=8,589,934,59216,777,216 \times 512 = 8,589,934,59216,777,216×512=8,589,934,592 bytes = **8 GB**
2. **28-bit LBA:**
   - Blocks: 228=268,435,4562^{28} = 268,435,456228=268,435,456
   - Assuming 512 bytes per block: 268,435,456×512=137,438,953,472268,435,456 \times 512 = 137,438,953,472268,435,456×512=137,438,953,472 bytes = **137 GB**

(3)

Hard Disk Interface:
The medium/standard used to connect a hard drive to a computer for data transfer.

Key HDD Interfaces:

1. **IDE (PATA):**

    - Max speed: 133 MB/s.
    - Parallel data transfer.
    - Limited cable length (18 inches).
2. **SATA:**
    - Max speed: Up to 6 Gb/s (SATA III).
    - Serial data transfer.
    - Thin, long cables (1m).
3. **SCSI:**
    - Max speed: ~320 MB/s.
    - Supports multiple devices.
    - Used in servers.
4. **NVMe (via PCIe):**
    - Max speed: ~7,000 MB/s.
    - Direct PCIe communication.
    - Ideal for SSDs, low latency.
5. **USB (External HDDs):**
    - USB 3.2: ~20 Gb/s.
    - Plug-and-play, portable.

(4)

Reading/Writing on CHS Disk:

1. **CHS (Cylinder-Head-Sector):** Specifies the physical location of data on a hard disk.
    - **Cylinder:** Tracks on multiple platters aligned vertically.
    - **Head:** Read/write head for each platter surface.
    - **Sector:** Smallest addressable storage unit within a track.
2. **Process:**
    - The disk controller moves the head to the specified **cylinder** (seek).
    - Waits for the required **sector** to rotate under the head (rotational delay).
    - Data is read/written once the sector is reached.

Seek Time vs. Rotational Delay:

- **Seek Time:** Time to move the read/write head to the correct track.
- **Rotational Delay:** Time for the desired sector to rotate under the head.
    - **Seek Time** depends on head movement, while **Rotational Delay** depends on spindle speed.

CHS to LBA Mapping & Seek Time Reduction:

- **Mapping:** LBA treats the disk as a linear array of blocks, simplifying addressing (block 0 to N).
- **Reduction:** LBA eliminates the need to compute physical cylinder, head, and sector, optimizing disk access and minimizing head movement for sequential reads.

Task2

(1)

- **Data Organization:** Separate OS, applications, and personal files for better management.
- **Improved Performance:** Reduces search time by limiting data access to specific partitions.
- **Backup and Recovery:** Easier to isolate and recover data without affecting other partitions.
- **Multi-OS Support:** Allows installation of multiple operating systems on the same drive.
- **Disk Space Management:** Allocate space efficiently for different purposes.

(2)

A **primary partition** is a main division of a hard drive that can store an operating system or data. It can be made bootable and is limited to a maximum of four per disk. These partitions are created directly on the disk.

A **logical partition**, on the other hand, exists within an extended partition. It cannot be booted directly and is mainly used for additional storage. Unlike primary partitions, there is no strict limit on the number of logical partitions that can be created.

(3)

A **partition table** is a data structure stored on a hard disk that contains information about how the disk is divided into partitions. It includes details such as the start and end of each partition, the size of the partitions, and the type of partitions (e.g., NTFS, ext4). This table is crucial for the operating system to understand how to access and manage the data stored on the disk.

This partition structure is used in **MBR-based partitioning** and is only capable of managing up to 4 primary partitions (or 3 primary + 1 extended partition).

- **Boot Code (446 bytes)**:
    - This is the initial executable code for booting the system (e.g., the bootloader).
- **Partition Entries (16 bytes each, 4 entries total)**:

- Each partition entry describes one of the 4 partitions on the disk (Primary or Extended).
- Information included in each entry:
    - Boot flag (1 byte).
    - Partition type (1 byte).
    - Starting sector (4 bytes).
    - Ending sector (4 bytes).
    - Partition size (4 bytes).

- **Disk Signature ("55 AA") (2 bytes)**:

    - The "55 AA" byte signature indicates a valid MBR, marking the end of the MBR structure.

(4)

To display the boot signature of your hard disk, you can use the following shell command:

```
hexdump -n 2 -s 510 /dev/sda
```

Explanation:

- **`-n 2`**: Reads 2 bytes.
- **`-s 510`**: Skips the first 510 bytes of the MBR, starting the reading from the 511st byte (the boot signature location).
- **`/dev/sda`**: your specific disk (e.g., `sda`, `sdb`).

This command will show the last two bytes of the Master Boot Record (MBR), which should be the boot signature "55 AA" if the disk is correctly formatted.

(5)

To display the **Stage 1 Boot Loader** (the first 512-byte sector) on your hard disk, you can use the following shell command:

```
dd if=/dev/sdX bs=512 count=1 | hexdump -C
```

Explanation:

- **`dd if=/dev/sda`**: Reads the input from your disk. Replace `/dev/sda` with your specific disk

- **`bs=512`**: Sets the block size to 512 bytes, which is the size of the MBR (Master Boot Record).
- **`count=1`**: Reads only 1 block (the first 512 bytes).
- **`hexdump -C`**: Displays the output in a readable hexadecimal format, including ASCII representation on the side.

This will show the content of the **Stage 1 boot loader** found in the first sector (MBR) of your disk.

(6)

To display the **partition type** of the first partition on your hard disk, you can use the following shell command:

```
fdisk -l /dev/sdX | grep "^/dev/sdX1" | awk '{print $6}'
```

Explanation:

- **`fdisk -l /dev/sda`**: Lists all partitions on the specified disk. Replace `/dev/sda` with your specific disk (e.g., `/dev/sda`).
- **`grep "^/dev/sda1"`**: Filters the output to show only the first partition (`/dev/sda1`).
- **`awk '{print $6}'`**: Extracts the 6th field from the output, which corresponds to the partition type (e.g., Linux, NTFS, etc.).

This command will give you the partition type of the first partition (`/dev/sda1`) on your disk.

(7)

- **Linux (ext4)**
  - Partition Type: **83**
- **Windows NTFS**
  - Partition Type: **07**
- **Swap (Linux)**
  - Partition Type: **82**

- **FAT32 (Windows)**
  - Partition Type: **0C**
- **Extended Partition**
  - Partition Type: **05**

(8)

Run the `fdisk` command to interact with the disk (e.g., `/dev/sda`):

```
sudo fdisk /dev/sda
```

Replace `/dev/sda` with your specific disk (e.g., `/dev/sda`).

2. Create Two Primary Partitions:

- **Primary Partition 1**:
  - Type `n` to create a new partition.
  - Choose `p` for primary.
  - Partition number: `1`.
  - Set the size (e.g., `+10G` for 10 GB).
- **Primary Partition 2**:
  - Type `n` again to create another primary partition.
  - Choose `p` for primary.
  - Partition number: `2`.
  - Set the size (e.g., `+20G` for 20 GB).

3. Create Extended Partition (for Logical Partitions):

- Type `n` to create a new partition.
- Choose `e` for extended (this is required to create logical partitions).
- Partition number: `3` (it will be the extended partition).
- Set the size (e.g., the remaining free space).

4. Create Logical Partitions (within the Extended Partition):

- Type `n` to create a logical partition inside the extended partition.
- Choose `l` for logical.
- Partition numbers: `4` to `9`.
- Set the size for each partition (e.g., `+5G` for 5 GB).

Task3

(1)

File System:

A **file system** is a method used by an operating system to organize and manage data stored on a storage device (e.g., hard drive, SSD). It defines how data is stored, retrieved, and named, allowing users and applications to access files systematically. Examples include **NTFS**, **ext4**, **FAT32**, and **HFS+**.

Journaling File System:

A **journaling file system** is a type of file system that keeps track of changes (updates) in a journal (log) before they are applied to the main file system. This helps in preventing data corruption during unexpected shutdowns, system crashes, or power failures. Common examples include **ext4**, **NTFS**, and **XFS**. The journal ensures that, in the event of a crash, the system can replay or discard incomplete operations to maintain integrity.

Functionalities of a Good File System:

1. **Data Integrity**: Ensures that data is not corrupted, especially in cases of power failure or crashes (e.g., through journaling).
2. **Efficiency**: Minimizes overhead in accessing files, efficiently manages space on the disk, and reduces fragmentation.
3. **Security**: Provides mechanisms for data access control, permissions, and encryption.
4. **Scalability**: Supports large file sizes and large volumes, allowing the file system to grow as needed.
5. **Data Recovery**: Offers features like file backups, redundancy, and quick recovery after system failures or crashes.
6. **File and Directory Management**: Organizes files into directories, supports naming conventions, and allows easy file access and modifications.
7. **Performance**: Offers fast read/write operations and minimizes system latency.

(2)

To display the list of currently loaded file system drivers, use the following command:

```
lsmod | grep fs
```

Explanation:

- **`lsmod`**: Lists all currently loaded kernel modules.
- **`grep fs`**: Filters the output to show only the modules related to file systems (those containing "fs" in their name).

(3)

. ext3 (Linux)

- **Max File Size**: 2 TB
- **Max Partition Size**: 16 TB

2. ext4 (Linux)

- **Max File Size**: 16 TiB (Tebibytes)
- **Max Partition Size**: 1 EiB (Exbibyte)

3. vfat (FAT32)

- **Max File Size**: 4 GB
- **Max Partition Size**: 8 TB (with a 32 KB cluster size)

4. NTFS (Windows)

- **Max File Size**: 16 TiB
- **Max Partition Size**: 256 TiB

5. ZFS (Linux, FreeBSD, Solaris)

- **Max File Size**: 16 EiB
- **Max Partition Size**: 256 ZiB (Zebibytes)

(4)

```
lsblk -o NAME,TYPE,FSTYPE,PARTTYPE,SIZE,MODE
```

Explanation:

- **`lsblk`**: Lists information about all available block devices (disks and partitions).

- `-o`: Specifies the columns to display

(5)

Assign the Label:

```
sudo e2label /dev/sda5 pucit9
```

Undo the Label:

```
sudo e2label /dev/sda5 ""
```

(6)

Format the Partition to NTFS:

```
sudo mkfs.ntfs /dev/sdb2
```

Confirm the Partition Format:

```
sudo blkid /dev/sdb2
```