Operating System

1. Operating System (OS)

Operating System ဆိုတာ ကွန်ပျူတာတစ်လုံးကို လည်ပတ်စေတဲ့ အဓိက software ပါ။

👉 It acts like a **boss or manager** between:

- **Hardware** (CPU, RAM, disk, keyboard, screen)
- **User & applications** (Chrome, Word, games, apps)

Operating system မရှိရင် ကွန်ပျူတာကို အသုံးပြုလို့မရပါဘူး။

What does an Operating System do?

1️⃣ Controls hardware

OS က hardware ကို ဘာလုပ်ရမယ်၊ ဘယ်အချိန်မှာ လုပ်ရမယ်ဆိုတာကို ပြောပြပါတယ်။.

- CPU: which program runs
- RAM: what data is stored
- Disk: where files are saved

2️⃣ Runs applications

Apps တွေက hardware နဲ့ တိုက်ရိုက် ဆက်သွယ်လို့ မရပါဘူး။

သူတို့က OS ကို မေးပြီး OS က ကိုင်တွယ်ပါတယ်။

Example:

- Browser wants internet → OS uses network card
- App wants to save file → OS uses hard disk

3️⃣ Manages memory (RAM)

- ပရိုဂရမ်တစ်ခုစီတွင် memory မည်မျှရှိသည်ကို ဆုံးဖြတ်သည်
- ပရိုဂရမ်များ တစ်ခုနှင့်တစ်ခု ပျက်စီးခြင်းမှ ကာကွယ်ပေးသည်

4️⃣ Manages processes

- Starts programs
- Stops programs
- Allows multitasking (many apps at once)

5️⃣ Handles files and folders

- Create, read, update, delete files
- Organizes data in folders

6️⃣ Provides user interface

Two types:

- **GUI** – Graphical (windows, icons, mouse)
- **CLI** – Command line (text commands)

Examples of Operating Systems

💻 Desktop / Laptop

- Windows
- macOS
- Linux (Ubuntu, Fedora)

📱 Mobile

- Android

- iOS

🗄️ **Servers**

- Linux
- Windows Server
- UNIX

Operating System and Networking (important for you)

The OS:

- Contains the **networking stack**
- Handles **TCP/IP**
- Uses **DNS, DHCP**
- Manages **ports, sockets, connections**

Apps ask the OS:
👉 “I want to connect to this website”
OS does the networking work.

One-line summary 🧠

operating system သည် ဟာ့ဒ်ဝဲကို စီမံခန့်ခွဲခြင်း၊ အပလီကေးရှင်းများကို လုပ်ဆောင်ခြင်းနှင့် အသုံးပြုသူများကို ကွန်ပျူတာနှင့် အပြန်အလှန်ဆက်သွယ်နိုင်စေသည့် အဓိကဆော့ဖ်ဝဲဖြစ်သည်။

2. Kernel

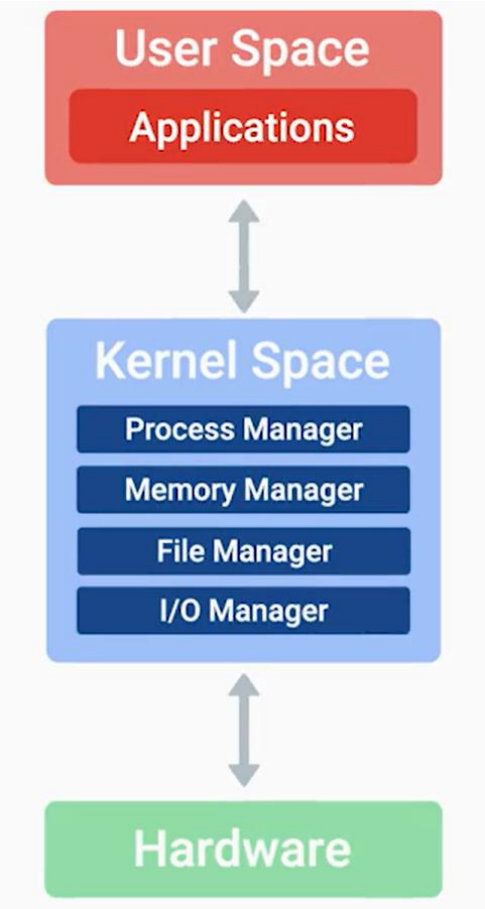
What is the Kernel?

Kernel ဆိုတာ operating system ရဲ့ **core (heart)** ပါ။.

👉 It sits **between applications and hardware**
👉 It has **full control** over the computer

App တွေက hardware နဲ့ တိုက်ရိုက် ဆက်သွယ်လို့ မရဘူး။

kernel ကနေ ဖြတ်သွားရမယ်။



Looking at the diagram (top → bottom)

🔴 **User Space (Applications)**

This is where:

- Chrome
- Word
- Games
- Mobile apps

run.

⚠️ Applications:

- ဝင်ရောက်ခွင့် အကန့်အသတ်ရှိတယ်
- hardware ကို တိုက်ရိုက် မထိနိုင်ဘူး
- kernel ကနေ အကူအညီတောင်းရမယ်

🔵 **Kernel Space (This is the Kernel)**

ဒါက OS ရဲ့အစွမ်းထက်ဆုံး အပိုင်းပဲ။

ယုံကြည်စိတ်ချရတဲ့ code တွေပဲ ဒီမှာ လည်ပတ်တယ်။

kernel မှာ manager အများအပြားပါဝင်ပါတယ်-

Process Manager
Controls running programs

- Starts processes
- Stops processes
- program တွေကြားမှာ CPU ကို ပြောင်းပါတယ် (multitasking)
- program တစ်ခုက system တစ်ခုလုံးကို freeze မလုပ်အောင် သေချာလုပ်ပါတယ်

Example:

You open Chrome + Music Player → Process Manager shares CPU time

---

2️⃣ Memory Manager

Controls RAM usage

- program တွေကို memory ပေးပါတယ်
- program တွေပိတ်သွားတဲ့အခါ memory ကို ပြန်ယူပါတယ်
- program တွေရဲ့ memory ကို တစ်ယောက်နဲ့တစ်ယောက် မသုံးအောင် တားဆီးပေးပါတယ်

Example:

app တစ်ခုမှာရှိတဲ့ bug တစ်ခုဟာ OS တစ်ခုလုံးကို crash မလုပ်ပါဘူး

3️⃣ File Manager

Controls files and storage

- Read files
- Write files
- Delete files
- Manage permissions

Example:

App asks: “Save this file” → Kernel handles disk access

4️⃣ I/O Manager (Input / Output)

Controls hardware devices

- Keyboard
- Mouse
- Screen

- Disk
- Network card
- USB devices

Example:

You press a key → kernel receives input → app sees it

🟢 Hardware

This is the **physical part**:

- CPU
- RAM
- Hard disk
- Network card
- Printer

Hardware က low-level instructions တွေကိုသာ နားလည်တာကြောင့် kernel က သူ့ရဲ့ဘာသာစကားကို ပြောတာပါ။

How Applications Talk to the Kernel

Applications use **system calls**:

Application

↓ (system call)

Kernel

↓

Hardware

Example:

- App wants internet → system call → kernel → network card
- App wants file → system call → kernel → disk

Why Kernel is VERY Important

✓ Controls everything
✓ Protects hardware
✓ Prevents crashes
✓ Enables multitasking
✓ Makes system secure

If the **kernel crashes**:
👉 The **whole system crashes** (blue screen / kernel panic)

One-line Summary 🧠

kernel က hardware၊ memory၊ processes၊ files နဲ့ device input/output တွေကို စီမံခန့်ခွဲတဲ့ operating system ရဲ့ core ပါ။

3. I/O Manager

What is the I/O Manager?

I/O (Input / Output) မန်နေဂျာဆိုတာ kernel ရဲ့ အစိတ်အပိုင်းတစ်ခုဖြစ်ပြီး software နဲ့ hardware device တွေကြားက ဆက်သွယ်မှုအားလုံးကို ထိန်းချုပ်ပေးပါတယ်။

👉 ကွန်ပျူတာထဲကို data ဘယ်လိုဝင်ထွက်လဲဆိုတာ ဆုံးဖြတ်ပေးပါတယ်။

What counts as Input and Output?

🟢 **Input devices**

- Keyboard
- Mouse / Touchpad
- Microphone
- Camera
- Network (incoming data)

🔵 **Output devices**

- Monitor
- Printer
- Speakers
- Disk (writing data)
- Network (sending data)

What does the I/O Manager do?

1️⃣ **Controls hardware access**

Application များသည် hardware ကို တိုက်ရိုက်ဝင်ရောက်၍မရပါ။

So the I/O Manager:

- application များမှ request များကို လက်ခံသည်

- device driver များနှင့် စကားပြောသည်
- hardware သို့ instruction များပေးပို့သည်

2️⃣ Uses Device Drivers

Each hardware device needs a **driver**.

Application

↓

I/O Manager

↓

Device Driver

↓

Hardware

Example:

- Printer driver
- Network card driver
- Disk driver

I/O Manager သည် မည်သည့် **driver** ကို အသုံးပြုရမည်ကို ဆုံးဖြတ်သည်။

3️⃣ Manages data transfer

- စက်ပစ္စည်းများနှင့် မှတ်ဉာဏ်အကြား ဒေတာများကို ရွှေ့ပြောင်းသည်
- buffering (ယာယီသိုလှောင်မှု) ကို ကိုင်တွယ်သည်
- အမှားအယွင်းများကို ကိုင်တွယ်သည် (စက်ပစ္စည်းကို ရှာမတွေ့၊ အလုပ်များနေ၊ ချိတ်ဆက်မှုပြတ်တောက်နေ)

Example:

Copying a file from USB to hard disk

4️⃣ Makes I/O efficient

I/O Manager:

- Queues တောင်းဆိုမှုများ
- စက်ပစ္စည်းဝင်ရောက်ခွင့်ကို အချိန်ဇယားဆွဲသည်
- အက်ပ်နှစ်ခုသည် တူညီသောစက်ပစ္စည်းကို ပျက်စီးခြင်းမှ ကာကွယ်ပေးသည်

Example:

Two apps printing at the same time → no chaos 😄

5️⃣ Supports networking (very important)

Network I/O is also handled here:

- Sending data packets
- Receiving data packets
- Talking to the network card

This is how browsers, apps, and servers send/receive data.

Simple Real-Life Example 🧠

Think of the I/O Manager as a **traffic police officer** 🚦 :

- Apps = cars
- Devices = roads
- Drivers = traffic rules

The I/O Manager:
✓ Controls traffic
✓ Prevents accidents
✓ Decides who goes first

Why I/O Manager is important

✓ Hardware safety
✓ System stability
✓ Faster performance
✓ Secure device access

Without it:

- Apps could damage hardware
- System would crash easily

One-line summary (exam-ready)

I/O မန်နေဂျာသည် စက်ပစ္စည်းဒရိုက်ဘာများကို အသုံးပြု၍ အပလီကေးရှင်းများနှင့် ဟာ့ဒ်ဝဲစက်ပစ္စည်းများအကြား ဆက်သွယ်မှုကို စီမံခန့်ခွဲသည့် kernel အစိတ်အပိုင်းတစ်ခုဖြစ်သည်။

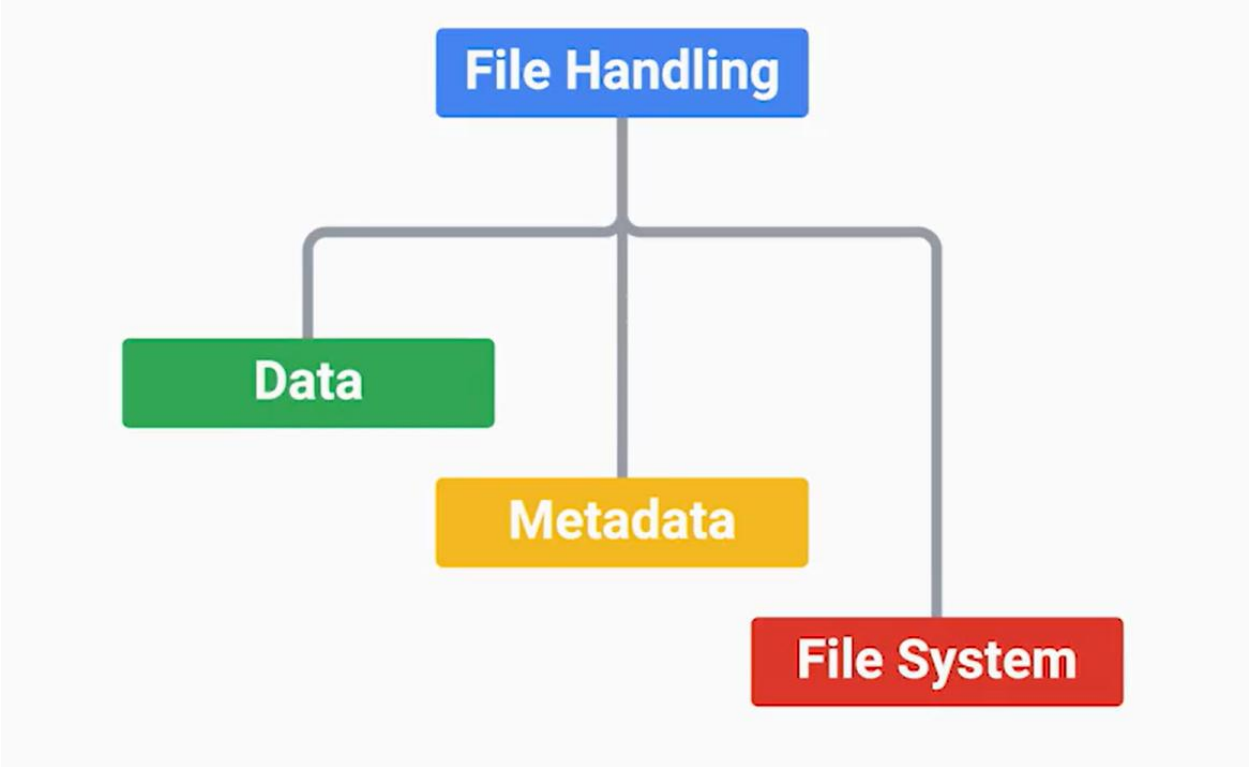
4. File System

What is a File System?

File system ဆိုတာ operating system ရဲ့ အစိတ်အပိုင်းတစ်ခုဖြစ်ပြီး storage devices (hard disk, SSD, USB) တွေမှာ ဖိုင်တွေကို စီစဉ်၊ သိမ်းဆည်း၊ စီမံခန့်ခွဲပေးပါတယ်။

👉 It tells the OS:

- **Where data is stored**
- **How files are named**
- **Who can access them**



File Handling (Top of the diagram)

ဖိုင်ကိုင်တွယ်ခြင်းဆိုတာ user တွေနဲ့ application တွေ လုပ်ဆောင်တဲ့အရာပါ။

- Create files
- Open files
- Read files
- Write files
- Delete files

Example:

Saving a Word document
Opening a photo
Deleting a video

Applications ask the OS to do these actions.

Data (Green)

Data = the **actual content** of the file.

Examples:

- Text inside a .txt file
- Image pixels in a .jpg
- Video content in a .mp4

👉 ဒါက user တစ်ယောက်အနေနဲ့ သင်ဂရုစိုက်တဲ့အရာပါ။

Metadata (Yellow)

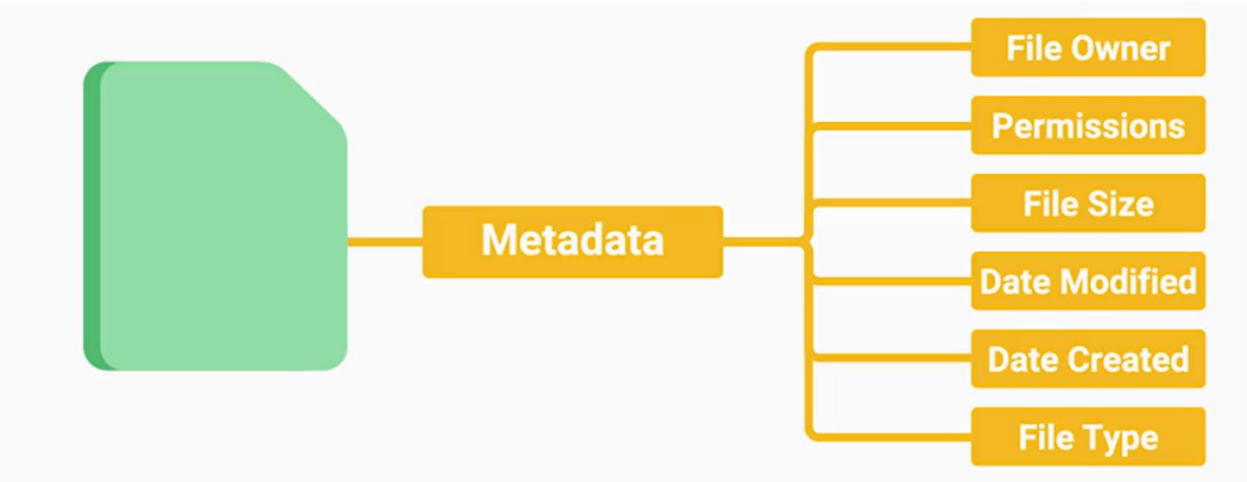
Metadata = information about the file (not the content)

Examples:

- File name
- File size
- Creation date
- Last modified date
- Owner
- Permissions (read/write/execute)
- File location on disk

🧠 Easy way to remember:

Metadata is data about data



File System (Red)

The **file system** controls:

- data and metadata ကို မည်သို့သိမ်းဆည်းသည်
- Directory (folder) ဖွဲ့စည်းပုံ
- ဖိုင်ခွင့်ပြုချက်များ
- Disk space ခွဲဝေခြင်း

Common file systems:

- **Windows:** NTFS, FAT32
- **Linux:** ext4, XFS
- **macOS:** APFS

How it works together (simple flow)

Application

↓

File Handling

↓

File System

↓

Data + Metadata stored on disk

Example:

You save a file
→ File system stores the **data**
→ File system updates the **metadata**

Why File Systems are Important

✓ Keeps files organized
✓ Prevents data loss
✓ Controls access & security
✓ Improves performance

Without a file system:

- Data would be a mess of random bits
- Files would be impossible to find

One-line exam-ready summary 🧠

ဖိုင်စနစ်ဆိုသည်မှာ သိုလှောင်ရေးကိရိယာများတွင် ဒေတာနှင့် မက်တာဒေတာများကို မည်သို့သိမ်းဆည်း၊ စီစဉ်ပြီး ဝင်ရောက်ကြည့်ရှုသည်ကို စီမံခန့်ခွဲသည့် OS အစိတ်အပိုင်းတစ်ခုဖြစ်သည်။

5. Block Storage

What is Block Storage?

Block storage က data တွေကို fixed-size block တွေနဲ့ သိမ်းဆည်းပါတယ်။

block တစ်ခုချင်းစီမှာ ထူးခြားတဲ့ address တစ်ခုစီရှိပေမယ့် file name မပါသလို file တွေလို metadata လည်း မပါပါဘူး။

👉 operating system က ဒီ block တွေကို ဘယ်လိုအသုံးပြုရမလဲဆိုတာ ဆုံးဖြတ်ပါတယ်။

Simple definition 🧠

Block storage = OS က file system နဲ့ format လုပ်ထားတဲ့ အရွယ်အစားတူ block တွေအဖြစ် ပိုင်းခြားထားတဲ့ raw storage။

How Block Storage Works

Disk / SSD

├─ Block 1

├─ Block 2

├─ Block 3

├─ Block 4

- Each block:
  - Same size (e.g. 4 KB)
  - Stores part of data
- Disk က file ဒါမှမဟုတ် folder တွေအကြောင်း မသိပါဘူး
- file system က file တွေကို map လုပ်ပါတယ် → block

Block Storage + File System

Block storage က သူ့အလိုလို လူသားတွေအတွက် အသုံးမဝင်ပါဘူး 😄

format လုပ်ပြီးရင် အသုံးဝင်လာပါတယ်။

Block Storage

↓

File System (NTFS, ext4)

↓

Files & Folders

Example:

- Hard disk = block storage
- ext4 / NTFS = file system
- Files = built on top

Where Block Storage is Used

💻 **Local storage**

- Hard disks (HDD)
- Solid State Drives (SSD)
- USB drives

☁️ **Cloud storage**

- AWS EBS
- Azure Managed Disks
- Google Persistent Disk

🗄️ **Servers & Databases**

- Operating system disks
- Database storage (very important!)

Why Block Storage is Powerful

✓ Very fast
✓ Low latency
✓ Random access
✓ Perfect for databases

That's why:

- Databases များသည် block storage ကို နှစ်သက်ကြသည်
- Operating systems များသည် block storage မှ boot လုပ်သည်

Block Storage vs File Storage (quick compare)

Feature	Block Storage	File Storage
Data unit	Blocks	Files
File system	Required	Built-in
Speed	Very fast	Slower
Use case	OS, DB, VM	Shared files

Real-life analogy 🧠

📦 Block storage = နံပါတ်များပါသော ဗလာသေတ္တာများ

📁 ဖိုင်စနစ် = ညွှန်ကြားချက်လက်စွဲ

📄 ဖိုင်များ = စနစ်တကျစီစဉ်ထားသော စာရွက်စာတမ်းများ

One-line exam answer

Block storage ဆိုသည်မှာ ဒေတာများကို ပုံသေအရွယ်အစား ဘလောက်များတွင် သိမ်းဆည်းပြီး ဖိုင်များကို စနစ်တကျစီစဉ်ရန် ဖိုင်စနစ်တစ်ခု လိုအပ်သော သိုလှောင်မှုအမျိုးအစားတစ်ခုဖြစ်သည်။

6. File Extension

File extension ဆိုတာ dot (.) ပြီးရင် ပေါ်လာတဲ့ ဖိုင်နာမည်ရဲ့ အဆုံးပိုင်းပါ။

ဒါက operating system ကို ဘယ်လိုဖိုင်အမျိုးအစားလဲ၊ ဘယ် program က ဖွင့်သင့်လဲဆိုတာ ပြောပြပါတယ်။

Example

report.pdf

photo.jpg

music.mp3

program.exe

- report → file name
- .pdf → file extension

Why File Extensions Matter

✓ OS ကို ဖိုင်အမျိုးအစားကို ခွဲခြားသိရှိနိုင်အောင် ကူညီပေးတယ်

✓ ဘယ် application က ဖိုင်ကို ဖွင့်နိုင်လဲဆိုတာ ပြောပြတယ်

✓ ဖိုင်ထဲမှာ ဘာတွေပါလဲဆိုတာ အသုံးပြုသူတွေ နားလည်အောင် ကူညီပေးတယ်

Example:

- .docx → Microsoft Word
- .jpg → Image viewer
- .mp4 → Video player

Common File Extensions

📄 **Documents**

- .txt – Text file
- .pdf – PDF document
- .docx – Word document

🖼 **Images**

- .jpg, .png, .gif

🎵 **Audio / Video**

- .mp3, .wav
- .mp4, .avi

💻 **Programs**

- .exe (Windows)
- .sh (Linux script)
- .apk (Android app)

Important Note ⚠️

Extension ကို ပြောင်းလဲခြင်းသည် ဖိုင်အကြောင်းအရာကို မပြောင်းလဲစေပါ။

❌ photo.jpg → photo.txt
✓ Still an image, not text

extension သည် label တစ်ခုသာဖြစ်ပြီး တကယ့် data format မဟုတ်ပါ။

File Extension vs File System

- **File extension**: အမည်ပေးခြင်းဆိုင်ရာ convention (user-level)
- **File system**: disk ပေါ်တွင် data များကို မည်သို့သိမ်းဆည်းသည် (OS-level)

အချို့သော system များ (Linux ကဲ့သို့) သည် extension များအပေါ် မှီခိုမှုနည်းပါးပြီး ဖိုင်ခွင့်ပြုချက်များနှင့် content များအပေါ် မှီခိုမှုပိုမိုများပြားသည်။

One-line exam answer 🧠

File extension ဆိုသည်မှာ ဖိုင်အမျိုးအစားနှင့် ဆက်စပ် application ကို ညွှန်ပြသည့် ဖိုင်အမည်တွင် dot တစ်ခုပြီးနောက် suffix ဖြစ်သည်။

7. Process

What is a Process?

လုပ်ငန်းစဉ်ဆိုတာ ကွန်ပျူတာမှာ လက်ရှိလည်ပတ်နေတဲ့ ပရိုဂရမ်တစ်ခုပါ။

👉 Program on disk = **program**
👉 Program in memory and executing = **process**

Simple Example

- You double-click **Chrome**
  - Chrome file on disk → **program**
  - Chrome running → **process**

Open Chrome twice?
👉 Two **processes**

What does a Process include?

Each process has:

- Program code
- Data (variables)
- CPU state (registers)
- Memory space
- Process ID (PID)

The OS keeps track of all this.

Who manages processes?

The **Process Manager** (inside the kernel):

- Creates processes
- Schedules CPU time
- Switches between processes (multitasking)
- Terminates processes

Process States (basic)

1️⃣ **New** – being created
2️⃣ **Ready** – waiting for CPU
3️⃣ **Running** – using CPU
4️⃣ **Waiting / Blocked** – waiting for I/O
5️⃣ **Terminated** – finished

Process vs Thread (quick)

Process	**Thread**
Independent	Part of a process
Own memory	Shared memory
Heavier	Lighter
Slower to create	Faster to create

Example:

- Chrome = process
- Each tab = thread (simplified view)

Why Processes are Important

✓ Multitasking
✓ Stability
✓ Security
✓ Resource control

If one process crashes, others keep running.

One-line exam answer 🧠

လုပ်ငန်းစဉ်ဆိုသည်မှာ လည်ပတ်မှုစနစ်မှ စီမံခန့်ခွဲသော အကောင်အထည်ဖော်မှုတွင် ပရိုဂရမ်တစ်ခု၏ ဥပမာတစ်ခုဖြစ်သည်။

8. Time Slice

What is a Time Slice?

Time slice (time quantum လို့လည်းခေါ်တယ်) ဆိုတာ process တစ်ခုအတွက် CPU အချိန်ပမာဏ အနည်းငယ်ကို ပေးထားတာပါ။

👉 CPU က process တစ်ခုကို တစ်ကြိမ်မှာ run ပေမယ့် အရမ်းမြန်မြန် switch လုပ်ပါတယ်။

Why Time Slicing is Needed

Many processes want the CPU at the same time:

- Browser
- Music player
- Antivirus
- Background services

The OS shares the CPU **fairly** using time slices.

How Time Slice Works (Simple)

Process A → 5 ms

Process B → 5 ms

Process C → 5 ms

(repeat)

Each process:

- သူ့ရဲ့ time slice အတွက် run ပါတယ်
- ပြီးရင် CPU က တခြား process တစ်ခုကို switch လုပ်ပါတယ်

ဒီ switching က တစ်စက္ကန့်ကို အကြိမ်ထောင်ပေါင်းများစွာ ဖြစ်ပျက်တာကြောင့် အရာအားလုံးက တစ်ပြိုင်နက်တည်း run နေသလိုပါပဲ။

Who controls Time Slice?

The **CPU scheduler** in the **kernel**:

- Assigns time slices
- Decides which process runs next
- Performs context switching

What happens when Time Slice ends?

When a time slice expires:

- Process is paused
- Its state is saved
- CPU switches to another process

This is called a **context switch**.

Time Slice Length Matters

- **Too long** → slow response (laggy system)
- **Too short** → too many context switches (overhead)

So the OS carefully balances it.

Real-life analogy 🧠

🍕 **Pizza sharing**

- One pizza (CPU)
- Friends (processes)
- Each friend gets a few seconds to eat
- Everyone feels like they're eating at the same time 😄

One-line exam answer

အချိန်အပိုင်းအခြားဆိုသည်မှာ multitasking operating system ရှိ လုပ်ငန်းစဉ်တစ်ခုအတွက် ခွဲဝေပေးထားသော CPU အချိန်ပမာဏဖြစ်သည်။

9. Virtual Memory

What is Virtual Memory?

Virtual memory ဆိုတာ ကွန်ပျူတာမှာ RAM ပိုများတယ်လို့ ထင်ရအောင် operating system က အသုံးပြုတဲ့ နည်းပညာတစ်ခုပါ။

👉 disk space (HDD/SSD) ကို memory အပိုအဖြစ် အသုံးပြုပါတယ်။

Why Virtual Memory is Needed

RAM is:

- Fast ⚪
- Limited
- Expensive

Virtual memory allows:

✓ တစ်ပြိုင်နက်တည်း program တွေ ပိုအလုပ်လုပ်နိုင်တယ်

✓ RAM သေးသေးလေးပေါ်မှာ program ကြီးတွေ အလုပ်လုပ်နိုင်တယ်

✓ system stability ပိုကောင်းတယ်

How Virtual Memory Works (Simple)

Application

↓

Virtual Memory (Address Space)

↓

RAM ↔ Disk (swap/page file)

- Frequently used data → stays in **RAM**
- Rarely used data → moved to **disk**

ဒီ movement ကို paging လို့ခေါ်ပါတယ်။

Key Terms You Should Know

📄 **Page**

- Small block of memory (e.g. 4 KB)

🔁 **Paging**

- RAM နှင့် disk အကြား စာမျက်နှာများ ရွှေ့ပြောင်းခြင်း

💾 **Swap Space / Page File**

- Disk area used as virtual memory
- Linux: **swap**
- Windows: **page file**

Example

Your computer has:

- 8 GB RAM
- 8 GB swap

You run programs using 10 GB memory:

- 8 GB in RAM
- 2 GB stored on disk

The OS moves data in and out automatically.

Important Note ⚠️

Virtual memory သည် RAM ထက် များစွာနှေးကွေးသည်။

စနစ်က swap အလွန်အကျွံအသုံးပြုပါက:

- Computer becomes slow

- This is called **thrashing**

Virtual Memory Benefits

✓ ပရိုဂရမ်များသည် RAM အရွယ်အစားအစစ်အမှန်ကို သိရှိရန်မလိုအပ်ပါ

✓ လုပ်ငန်းစဉ်တစ်ခုစီသည် ၎င်း၏ကိုယ်ပိုင်မှတ်ဉာဏ်နေရာရရှိသည်

✓ ပရိုဂရမ်တစ်ခုသည် အခြားပရိုဂရမ်များကို ပျက်စီးခြင်းမှ ကာကွယ်ပေးသည်

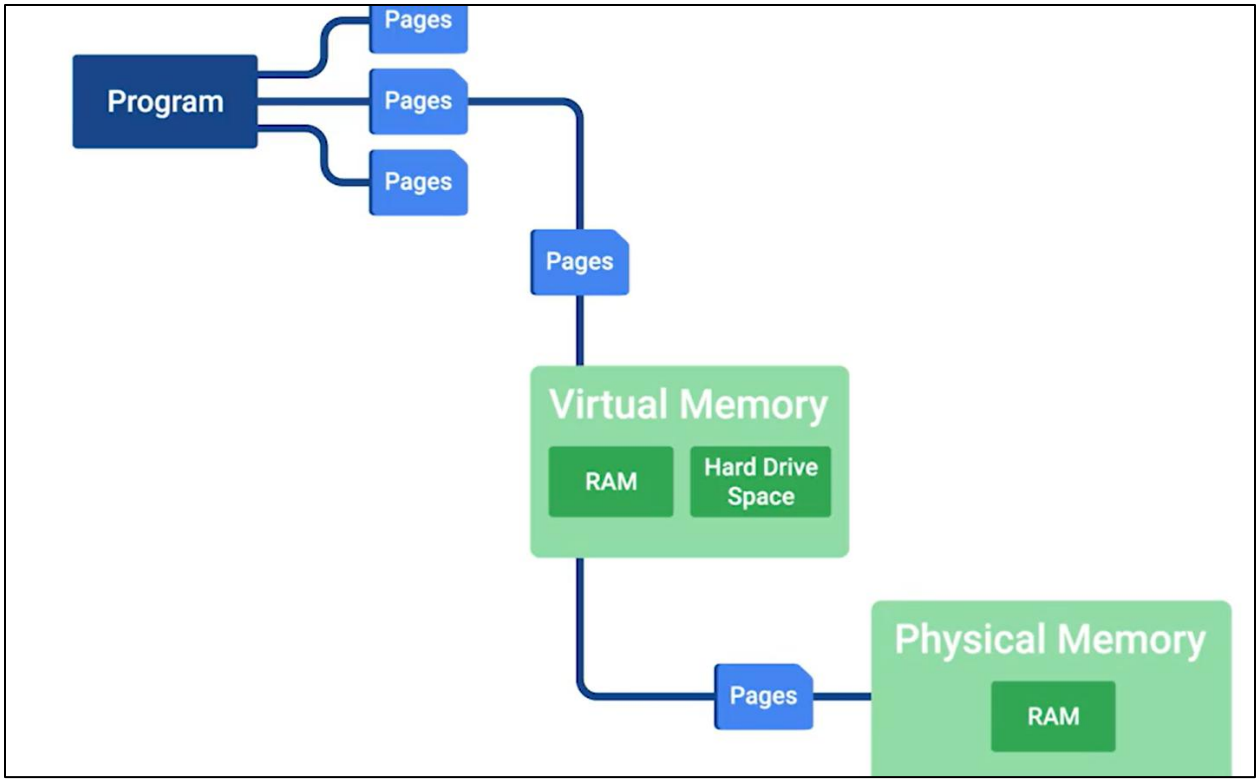
Real-life analogy 🧠

🧠 RAM = desk
📦 Disk = storage cabinet

- Keep current work on desk
- Store old papers in cabinet
- Take them back when needed

One-line exam answer

Virtual memory သည် physical RAM ကို တိုးချဲ့ရန်နှင့် လုပ်ငန်းစဉ်တစ်ခုစီကို ၎င်း၏ကိုယ်ပိုင်လိပ်စာနေရာပေးရန်အတွက် disk space ကို အသုံးပြုသည့် မှတ်ဉာဏ်စီမံခန့်ခွဲမှုနည်းပညာတစ်ခုဖြစ်သည်။


Program
Pages
Pages
Pages
Pages
Virtual Memory
RAM
Hard Drive Space
Pages
Physical Memory
RAM

10. Shell

What is a Shell?

Shell ဆိုတာ user တွေကို operating system နဲ့ ဆက်သွယ်ပြောဆိုနိုင်စေတဲ့ program တစ်ခုပါ။

👉 user နဲ့ kernel ကြားမှာ interface အနေနဲ့ လုပ်ဆောင်ပါတယ်။

What does the Shell do?

- user ဆီကနေ command တွေကို လက်ခံတယ်
- သူတို့ကို နားလည်တယ် (အဓိပ္ပာယ်ဖွင့်ဆိုတယ်)
- kernel ကို request တွေ ပို့တယ်
- output ကို user ဆီ ပြန်ပြတယ်

User → Shell → Kernel → Hardware

Types of Shells

1️⃣ **Command-Line Shell (CLI)**

Text-based interface.

Examples:

- **Bash** (Linux)
- **Zsh**
- **PowerShell** (Windows)
- **Command Prompt**

Example command:

ls

2️⃣ Graphical Shell (GUI)

Visual interface.

Examples:

- Windows Desktop
- macOS Finder
- GNOME / KDE (Linux)

You click icons, menus, windows — that's also a shell.

Why Shell is Important

✓ OS ကို တိုက်ရိုက်ထိန်းချုပ်ခြင်း

✓ admin များနှင့် IT support အတွက် ပိုမိုမြန်ဆန်ခြင်း

✓ အစွမ်းထက်သော scripting နှင့် automation

✓ server များအတွက် မရှိမဖြစ်လိုအပ်သည် (GUI မရှိပါ)

Shell vs Kernel (Very Important)

Shell	Kernel
User interface	Core of OS
Takes commands	Controls hardware
Runs in user space	Runs in kernel space
Can be changed	Cannot be easily replaced

Real-life analogy 🧠

👧 User = customer
🧻 Shell = waiter
🍳 Kernel = kitchen

You tell the waiter (shell) what you want.
The kitchen (kernel) does the real work.

One-line exam answer

shell သည် အသုံးပြုသူများအား kernel သို့ command များပေးပို့ခြင်းဖြင့် operating system နှင့် အပြန်အလှန်ဆက်သွယ်နိုင်စေသည့် user interface တစ်ခုဖြစ်သည်။

11. Logs

What are Logs?

Logs are **records of events** that happen in a system, application, or network.

Logs များသည် system, application, or network တွင်ဖြစ်ပျက်သော အဖြစ်အပျက်များ၏ မှတ်တမ်းများဖြစ်သည်။

👉 ၎င်းတို့သည် ဖြစ်ပျက်ခဲ့သည်များ၊ မည်သည့်အချိန်တွင်ဖြစ်ပျက်ခဲ့သည်နှင့် တစ်ခါတစ်ရံတွင် အဘယ်ကြောင့်ဖြစ်ပျက်ခဲ့သည်ကို ပြောပြသည်။

What do Logs Record?

Logs can include:

- Errors and failures ❌
- Warnings ⚠️
- Successful actions ✅
- Login attempts
- System start/shutdown
- Network connections

Why Logs are Important

✓ ပြဿနာရှာဖွေခြင်း

✓ အမှားများရှာဖွေခြင်း

✓ လုံခြုံရေးစောင့်ကြည့်ခြင်း

✓ စွမ်းဆောင်ရည် ခွဲခြမ်းစိတ်ဖြာခြင်း

✓ စာရင်းစစ်ဆေးခြင်းနှင့် လိုက်နာမှု

👉 တစ်စုံတစ်ခု ပျက်စီးသွားသောအခါ မှတ်တမ်းများသည် သင်ပထမဆုံးစစ်ဆေးရမည့်အရာဖြစ်သည်။

Common Types of Logs

🗄️ **System Logs**

- OS events
- Hardware issues
- Boot and shutdown

Example:

- Linux: /var/log/syslog
- Windows: Event Viewer

👤 **Application Logs**

- App errors
- App activity

Example:

- Web server logs (Apache, Nginx)
- Database logs (PostgreSQL, Oracle)

🔐 **Security Logs**

- Login attempts
- Access failures
- Suspicious activity

Example:

- Authentication logs
- Firewall logs

🌐 **Network Logs**

- Connection attempts
- Packet drops
- DNS activity

Log Format (Simple Example)

2026-01-30 10:45:12 ERROR User login failed (invalid password)

This shows:

- Time
- Severity
- Message

Log Levels (Very Common)

Level	Meaning
INFO	Normal activity
WARN	Something unusual
ERROR	Something failed
CRITICAL	System failure

Where Logs Are Stored?

- Text files
- Databases
- Central log servers

Examples:

- Linux: /var/log/
- Windows: Event Viewer

Real-life analogy 🧠

🧮 Logs = **system diary**

They remember things the human brain can’t.

One-line exam answer

မှတ်တမ်းများသည် စောင့်ကြည့်ခြင်း၊ ပြဿနာရှာဖွေခြင်းနှင့် လုံခြုံရေးအတွက် အသုံးပြုသော စနစ်နှင့် အပလီကေးရှင်းဖြစ်ရပ်များ၏ မှတ်တမ်းများဖြစ်သည်။

12. BIOS / UEFI

What is BIOS / UEFI?

BIOS နဲ့ UEFI တို့ဟာ motherboard မှာသိမ်းဆည်းထားတဲ့ firmware တွေဖြစ်ပါတယ်။

👉 operating system မတင်ခင် computer ကိုစတင်ပါတယ်။

What does BIOS / UEFI do?

1️⃣ hardware ကိုဖွင့်ပါ

2️⃣ hardware ကိုစစ်ဆေးပါ (POST)

3️⃣ boot device (HDD/SSD/USB) ကိုရှာဖွေပါ

4️⃣ operating system ကိုတင်ပါ

Without BIOS/UEFI → **computer cannot boot**

One-line exam answer

BIOS/UEFI သည် motherboard firmware ဖြစ်ပြီး system startup လုပ်နေစဉ်အတွင်း hardware ကို initialize လုပ်ပြီး operating system ကို load လုပ်ပါသည်။

13. Bootloader

What is a Bootloader?

bootloader ဆိုတာ operating system ကို memory ထဲကို load လုပ်တဲ့ program ငယ်လေးတစ်ခုပါ။

👉 BIOS/UEFI ပြီးရင် OS kernel မတိုင်ခင်မှာ run ပါတယ်။

Boot Process (Big Picture)

Power ON

↓

BIOS / UEFI

↓

Bootloader

↓

Operating System (Kernel)

What does the Bootloader do?

1️⃣ Finds the operating system
2️⃣ Loads the OS **kernel** into RAM
3️⃣ Passes control to the kernel
4️⃣ (Optional) lets you choose which OS to boot

1) လည်ပတ်မှုစနစ်ကို ရှာဖွေပါ။
2) OS kernel ကို RAM ထဲသို့ တင်ပါ။
3) kernel သို့ ထိန်းချုပ်မှုကို ပေးပို့ပါ။
4) (ရွေးချယ်နိုင်သည်) သည် မည်သည့် OS ကို boot လုပ်ရမည်ကို ရွေးချယ်နိုင်စေပါသည်။

Why Bootloader is Needed

- BIOS/UEFI **cannot load a full OS**
- Bootloader knows:
  - Where the OS is stored
  - How to start it correctly

Common Bootloaders

🐧 **Linux**

- **GRUB** (most common)
- LILO (older)

🪟 **Windows**

- **Windows Boot Manager**

🍎 **macOS**

- Boot.efi

Bootloader and Dual Boot

If you have:

- Windows + Linux

Bootloader:

- Shows a menu
- Lets you choose which OS to start

Example:

1. Ubuntu
2. Windows 11

Bootloader Location

- **BIOS systems** → MBR (Master Boot Record)
- **UEFI systems** → EFI System Partition (ESP)

Bootloader vs BIOS/UEFI

BIOS / UEFI	Bootloader
Firmware	Software
Starts first	Starts second
Hardware check	Loads OS
On motherboard	On disk

What happens if Bootloader breaks?

❌ OS won't start
❌ Error like "Operating System not found"

Fix usually involves:

- Boot repair
- Reinstalling bootloader

Real-life analogy 🧠

🎬 BIOS/UEFI = cinema opens
🎟️ Bootloader = ticket checker
🎥 OS = movie starts

One-line exam answer

bootloader ဆိုသည်မှာ BIOS/UEFI initialization ပြုလုပ်ပြီးနောက် operating system kernel ကို memory ထဲသို့ load လုပ်သည့် program တစ်ခုဖြစ်သည်။

14. Boot Methods

Boot Methods – Easy Explanation

What is booting?

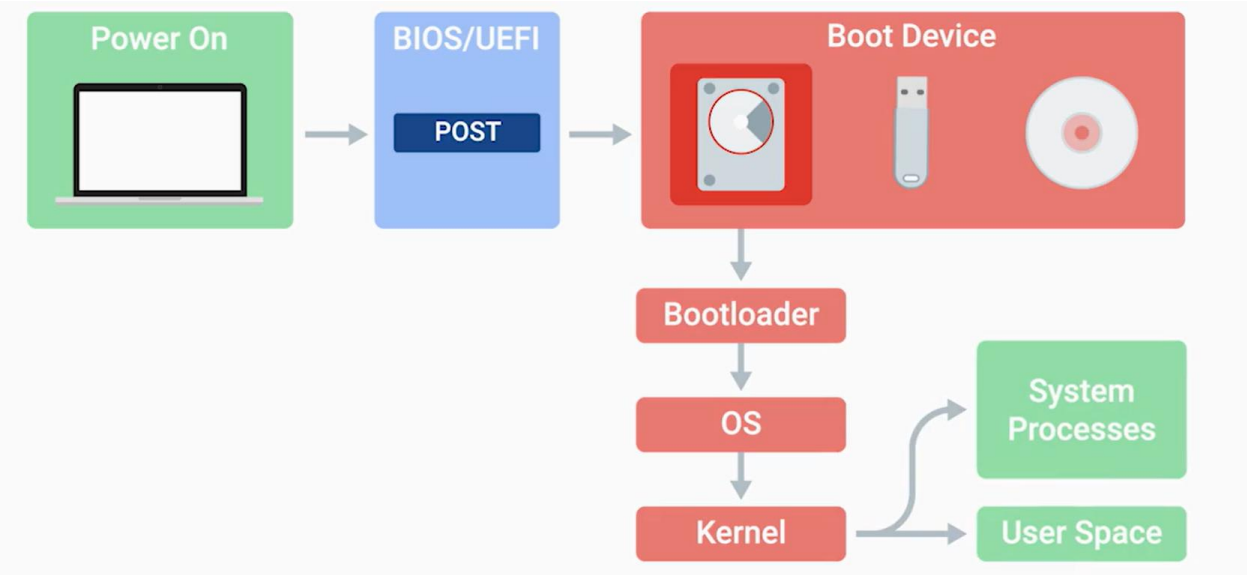
Booting ဆိုတာ ကွန်ပျူတာတစ်လုံးကို စတင်ပြီး operating system (OS) ကို load လုပ်တဲ့ လုပ်ငန်းစဉ်ပါ။

Normally, you:
👉 Press the **power button**
👉 The computer starts automatically

But sometimes:

- The OS is damaged
- The system won't start
- You want to use a **different OS**

That's when **different boot methods** are useful.



The Boot Process (Step by Step)

1️⃣ **Power ON**

2️⃣ **BIOS / UEFI starts**

- Checks hardware (keyboard, RAM, disk, etc.)

3️⃣ **Boot device is chosen**

- Based on **boot order** set in BIOS

4️⃣ **Bootloader is found**

- A small program that loads the OS

5️⃣ **Operating System loads**

- Windows, Linux, macOS, etc.

What is Boot Order?

Boot order is the list of devices the computer checks to find a bootloader.

Example boot order:

1. Hard drive
2. USB drive
3. CD/DVD
4. Network

➡️ BIOS သည် ၎င်းတွေ့ရှိသော ပထမဆုံး bootable device ကို အသုံးပြုသည်။

How to Change Boot Options

Entering BIOS

- Restart the computer
- Press a key like:
  - **DEL**
  - **F2**
  - **F12**
  - **ESC**
- macOS: hold **Option** key

Inside BIOS:

- Go to **Boot Options**
- Change the device order

Boot Method Options

🔹 **External Boot Methods**

1. USB Drive

- OS ပါသော USB ကို အသုံးပြု၍ Boot လုပ်ပါ
- Common for:
  - OS installation
  - System repair

2. Optical Media (CD/DVD)

- Boot from a CD or DVD
- Older but still used

3. Solid State Boot Drive

- Fast storage (SSD or flash drive)
- No moving parts

4. External Hot-Swappable Drive

- External hard drive
- Can be plugged/unplugged easily

5. Network Boot (PXE)

- Boots OS from a **local network**
- Used when:
  - No OS installed
  - Installing OS on many computers
- Requires:
  - LAN connection
  - PXE enabled in BIOS

6. Internet-Based Boot

အင်တာနက်ကို အသုံးပြု၍ ဝေးလံခေါင်သီသောနေရာမှ Boot တက်ခြင်း။

Two methods:

- **Remote Access (iDRAC/IPMI)** – used for servers
- **Wake on LAN (WoL)** – starts system remotely

🔹 Internal Boot Methods

Disk Partitions (Dual Boot)

- hard drive တစ်ခုကို အပိုင်းပိုင်းခွဲထားသည်
- အစိတ်အပိုင်းတစ်ခုစီတွင် မတူညီသော OS ရှိသည်

Example:

- Partition တစ်ခုတွင် Windows
- အခြား partition တစ်ခုတွင် Linux

At startup:
👉 မည်သည့် OS ကို boot လုပ်ရမည်ကို သင်ရွေးချယ်သည်

This is useful if:

- One OS fails
- You need troubleshooting access

Why Boot Methods Are Important

✓ ပျက်စီးနေသော စနစ်များကို ပြန်လည်ရယူပါ

✓ OS ကို ထည့်သွင်းခြင်း သို့မဟုတ် ပြုပြင်ခြင်း

✓ မတူညီသော operating system များကို လည်ပတ်ပါ

✓ Startup ပြဿနာများကို ဖြေရှင်းပါ

✓ server များကို အဝေးမှ စီမံခန့်ခွဲပါ

Key Takeaways (Exam Friendly)

- ကွန်ပျူတာတစ်လုံးသည် နည်းလမ်းများစွာဖြင့် boot လုပ်နိုင်သည်
- Boot order သည် OS ကို မည်သည့်နေရာမှ load လုပ်သည်ကို ဆုံးဖြတ်သည်
- External boot tools များသည် OS ကို ပြုပြင်ခြင်း သို့မဟုတ် install လုပ်ခြင်းကို ကူညီပေးသည်
- Dual boot သည် operating system များစွာကို ခွင့်ပြုသည်
- Network နှင့် internet booting များသည် enterprise environment များတွင် အဖြစ်များသည်

One-line summary:

Boot နည်းလမ်းများသည် ကွန်ပျူတာတစ်ခုသည် ၎င်း၏ operating system ကို မည်သို့နှင့် မည်သည့်နေရာမှ load လုပ်သည်ကို သတ်မှတ်ပေးသည်။

15. Virtual Machine (VM)

ကွန်ပျူတာအတွင်းတွင် လည်ပတ်သော software-based ကွန်ပျူတာတစ်ခုဖြစ်သည်။ ၎င်းသည် ၎င်း၏ကိုယ်ပိုင်ကွန်ပျူတာတစ်လုံးကဲ့သို့ လုပ်ဆောင်သည်-:

- Operating System
- CPU
- Memory (RAM)
- Storage
- Network interface

သို့သော် ဤအရာအားလုံးသည် virtual ဖြစ်ပြီး software မှ ဖန်တီးပြီး စီမံခန့်ခွဲသည်။

How a Virtual Machine Works

1. **Physical machine (Host)**
   Your real computer (Windows, Linux, macOS)
2. **Hypervisor**
   Software that creates and manages virtual machines
   Examples:
   - VirtualBox
   - VMware
   - Hyper-V
3. **Virtual Machine (Guest)**
   ကိုယ်ပိုင် OS ကို အသုံးပြုတဲ့ virtual computer
   - Example: Linux running inside Windows

Types of Hypervisors

Type 1 (Bare-metal)

- Runs directly on hardware
- Used in servers
- Examples:
    - VMware ESXi
    - Microsoft Hyper-V (Server)

Type 2 (Hosted)

- Runs on top of an OS
- Used on personal computers
- Examples:
    - VirtualBox
    - VMware Workstation

Why Use Virtual Machines?

✅ ကွန်ပျူတာတစ်လုံးတည်းမှာ operating system အများအပြားကို run လုပ်ပါ

✅ လုံခြုံတဲ့ testing environment (host OS အတွက် အန္တရာယ်မရှိပါ)

✅ သင်ယူမှု၊ development နဲ့ server တွေအတွက် အသုံးဝင်ပါတယ်

✅ အလွယ်တကူ backup နဲ့ restore လုပ်နိုင်ပါတယ် (snapshot)

✅ dual-boot အစားထိုးကို support လုပ်ပါတယ်

Example

- Host OS: **Windows 11**
- VM 1: **Ubuntu Linux**
- VM 2: **Windows Server**

အားလုံး laptop တစ်လုံးတည်းမှာ တစ်ပြိုင်နက်တည်း run လုပ်ပါတယ် 💻

Virtual Machine vs Physical Machine

Physical Machine	Virtual Machine
Real hardware	Software-based
One OS usually	Multiple OSs
Hard to duplicate	Easy to copy
Needs full device	Uses shared resources

Related Terms You Should Know

- **Host OS** – Main operating system
- **Guest OS** – OS inside the VM
- **Snapshot** – Saved state of a VM
- **Virtual disk (VHD/VMDK)** – VM storage file

16. Boot Methods

What is Booting?

Booting ဆိုတာ ကွန်ပျူတာတစ်လုံးကို စတင်ပြီး လည်ပတ်မှုစနစ် (OS) ကို မှတ်ဉာဏ်ထဲကို ထည့်သွင်းတဲ့ လုပ်ငန်းစဉ်ပါ။

Normally, you:
➡️ Press the **power button**
➡️ Computer follows the **default boot process**

But if:

- The OS is corrupted ❌
- You want to install or use another OS 💿
- The system won't start

➡️ သင်သည် အခြား boot နည်းလမ်းများကိုသာ အသုံးပြုသည်

The Boot Process (Step by Step)

1. **Power On**
2. **BIOS/UEFI starts**
    - Runs hardware checks (POST)
    - Initializes CPU, RAM, storage, keyboard, etc.
3. **Boot Order is checked**
    - Example order:
        1. Internal Hard Drive
        2. USB
        3. CD/DVD
        4. Network
4. **Bootloader is found**
    - Small program stored on a bootable device

5. **Bootloader loads the OS**
    - Windows, Linux, macOS, etc.

👉 If no bootloader is found → system fails to boot

Boot Order (Important Concept)

Boot order = BIOS မှ bootloader အတွက် စစ်ဆေးသော device များ၏ ဦးစားပေးစာရင်း။

၎င်းကို BIOS/UEFI setting များတွင် ပြောင်းလဲနိုင်သည်။

How to Enter BIOS

- Windows / Linux: DEL, F2, F12 (depends on manufacturer)
- macOS: စတင်ချိန်တွင် Option key ကိုဖိထားပါ

Boot Method Options

1. External Boot Methods

✅ **USB Drive**

- OS installer သို့မဟုတ် recovery tools များဖြင့် USB မှ boot လုပ်ပါ
- Common use:
    - Install OS
    - Repair broken system

✅ **Optical Media (CD/DVD/Blu-ray)**

- Older method
- ပြန်လည်ရယူခြင်း သို့မဟုတ် legacy စနစ်များအတွက် အသုံးဝင်ဆဲဖြစ်သည်

✅ **Solid State Boot Drive**

- SSD or flash drive
- Faster than HDD

- No moving parts

✅ **External Hot-Swappable Drive**

- External HDD/SSD
- Can move between computers
- No shutdown required

✅ **Network Boot (PXE)**

- Boots OS from a **local network**
- No storage device needed
- Used in:
    - Companies
    - Labs
    - Diskless systems

✅ **Internet-Based Boot**

Used for **remote management**:

1. **Remote Access (IPMI / iDRAC)**
    - Server management
2. **Wake on LAN (WoL)**
    - Power on system remotely using network signal

---

2. Internal Boot Methods

✅ **Disk Partitions (Dual Boot)**

- One disk, multiple OSes
- Example:
    - Windows on Partition 1
    - Linux on Partition 2
- On startup, you choose which OS to load

📌 Very useful if:

- One OS fails
- You need troubleshooting access

Best Practices (Key Points)

✓ BIOS/UEFI ထဲသို့ မည်သို့ဝင်ရမည်ကို အမြဲသိရှိပါ

✓ ပြန်လည်ရယူရန်အတွက် bootable USB ကို သိမ်းဆည်းထားပါ

✓ dual boot ကို ဂရုတစိုက်အသုံးပြုပါ

✓ Network boot သည် Enterprise ပတ်ဝန်းကျင်များအတွက် အကောင်းဆုံးဖြစ်သည်

✓ တိုက်ခိုက်မှုများကို ရှောင်ရှားရန် အင်တာနက်အခြေပြု boot ကို လုံခြုံစွာထားပါ

✓ boot မအောင်မြင်ခြင်းကို ကာကွယ်ရန် မှန်ကန်သော boot order ကို သတ်မှတ်ပါ

Key Takeaways (Exam-Ready)

- BIOS/UEFI initializes hardware and selects boot device
- Bootloader loads the operating system
- Boot order determines **how** the system boots
- You can boot from:
    - Internal disks
    - USB
    - Optical media
    - Network
    - Internet
- Dual boot allows multiple OSs on one machine

Course 1 – Module 3

Easy Glossary (Simple + Memorable)

🧠 **Operating Systems (Think: "What runs the computer?")**

- **Operating System (OS)**
  👉 ကွန်ပျူတာ၏ အဓိက အကြီးအကဲ။ ဟာ့ဒ်ဝဲကို စီမံခန့်ခွဲပြီး အသုံးပြုသူများကို အက်ပ်များ လည်ပတ်ခွင့်ပြုသည်။

- **Kernel**
  👉 The *heart* of the OS. Controls CPU, memory, and processes.

- **User space**
  👉 Where users and apps live (browsers, music players, settings).

- **Android**
  👉 Mobile OS based on Linux.

- **iOS**
  👉 Apple's mobile OS.

- **Mac OS**
  👉 Apple's desktop OS.

- **Chrome OS**
  👉 Google's Linux-based OS for Chromebooks.

- **Ubuntu**
  👉 The most popular Linux version for everyday users.

- **Distributions (Distros)**
  👉 Different versions of Linux (Ubuntu, Debian, Red Hat).

🧠 **Memory tip:**

Kernel = heart, OS = body

⚡ Booting & Startup (Think: “How the computer starts”)

- **Boot**
  👉 Starting a computer.
- **BIOS/UEFI**
  👉 Wakes up first. Checks hardware and finds the OS.
- **Bootloader**
  👉 The launcher that loads the OS.
- **Install image**
  👉 A downloadable OS file used for installation.
- **Etcher.io**
  👉 Tool to make a USB bootable.

🧠 **Order to remember:**
Power → BIOS → Bootloader → OS

📁 Files & Storage (Think: “How data is stored”)

- **Computer file**
  👉 Any saved data (doc, image, song).
- **File system**
  👉 Rules for storing and organizing files.
- **File extension**
  👉 File type indicator (.txt, .jpg, .mp3).
- **Folders / Directories**
  👉 Containers for files.
- **Finder**
  👉 File manager on Mac.
- **Metadata**
  👉 Information about a file (creator, date, permissions).
- **File handling**
  👉 How programs save and manage files.

- **Block storage**
  👉 Data stored in blocks for faster access.
- **Data blocks**
  👉 Small pieces of data stored in different disk locations.
- **HFS+ / APFS**
  👉 Apple file systems (APFS = newer & encrypted).

🧠 **Memory tip:**

Files live in folders, folders live in file systems

🧮 **Memory & Performance (Think: "Speed & resources")**

- **Memory management**
  👉 Kernel makes sure apps get enough memory.
- **Virtual memory**
  👉 RAM + hard drive space acting as memory.
- **Swap space**
  👉 Disk space used when RAM is full.
- **Time slice**
  👉 Tiny CPU time given to a process.
- **Hardware resource deficiency**
  👉 Not enough RAM, storage, or CPU.
- **Scalability**
  👉 အလုပ်တာဝန် အနည်းနှင့်အများ ကိုင်တွယ်ဆောင်ရွက်နိုင်ခြင်း။

🧠 **Memory tip:**

RAM full? → Swap space saves the day

🔄 Processes & Programs (Think: “What’s running now?”)

- **Application**
  👉 A program made for a specific task.
- **Process management**
  👉 OS decides when and how programs run.
- **System**
  👉 Hardware + software working together.

🧠 **Memory tip:**

App = program, Process = running program

🖥️ Interfaces & Controls (Think: “How humans interact”)

- **Shell**
  👉 Middleman between user and OS.
- **Terminal**
  👉 Text-based interface.
- **CLI (Command Line Interface)**
  👉 Text commands to control the OS.
- **PowerShell**
  👉 Windows command shell.
- **Task bar**
  👉 Quick access bar (time, network, apps).
- **System settings**
  👉 Control panel for system options.

🧠 **Memory tip:**

Shell = interpreter, Terminal = window

🌐 **Networking & Remote Access (Think: “Connecting to other computers”)**

- **Hostname**
  👉 Computer’s name on a network.
- **Remote connection**
  👉 Accessing a computer from far away.
- **RDP (Remote Desktop Protocol)**
  👉 Microsoft’s remote desktop tool.
- **Microsoft Terminal Services Client**
  👉 RDP client.
- **SSH (Secure Shell)**
  👉 Secure remote access protocol.
- **OpenSSH**
  👉 Most common SSH software.
- **SSH client**
  👉 Software to connect to another machine.
- **SSH server**
  👉 Software that accepts SSH connections.
- **SSH authentication key**
  👉 Password-free secure login.
- **Plink (PuTTY Link)**
  👉 Command-line SSH tool.
- **VPN**
  👉 Secure private network over the internet.

🧠 **Memory tip:**

SSH = secure command access

🧪 Tools, Users & Concepts

- **Virtual machine (VM)**
  👉 A computer inside a computer.
- **VirtualBox**
  👉 Software to create VMs.
- **Qwiklabs**
  👉 Online cloud training platform.
- **Power user**
  👉 Advanced computer user.
- **User name**
  👉 Account identifier.
- **Logs**
  👉 System event records.
- **Standardization**
  👉 common IT စံနှုန်းများကို အသုံးပြု
- **PC (Personal Computer)**
  👉 A computer for one person.
- **Input / Output device**
  👉 Keyboard, mouse, monitor, speakers, network card, etc.

🔑 One-Line Super Memory Summary

- **Kernel** = OS core
- **Shell** = command interpreter
- **BIOS** = startup checker
- **Bootloader** = OS launcher
- **VM** = OS inside OS
- **Swap** = backup memory
- **Logs** = system diary