Operating Systems & Linux

Comprehensive study guide organized from Lec 1 and Section 1.

1. Introduction to Operating Systems

What is a Computer System?

A computer system consists of four main components:

- **Hardware:** Provides basic computing resources (CPU, Memory, I/O devices).
- **Operating System (OS):** Controls and coordinates the use of hardware among applications and users.
- **Application Programs:** Define ways resources are used to solve user problems (Word processors, browsers, games).
- **Users:** People, machines, or other computers.

What Operating Systems Do (Views)

The role of an OS depends on the point of view:

- **PC Users:** Want convenience, ease of use, and good performance. They **don't care about resource utilization**.
- **Shared Computers (Mainframes):** The OS must keep all users happy and allocate resources efficiently.
- **Mobile Devices:** Resource poor, optimized for usability and battery life.

What is an Operating System?

- It is a **Resource Allocator** (manages and allocates CPU, memory, etc. efficiently) and a **Control Program** (manages execution of user programs to prevent errors and improper use).
- **The Kernel:** "The one program running at all times on the computer".
- Everything else is either a system program (ships with OS but not kernel) or an application program.
- Modern OS includes **Middleware** (frameworks providing additional services like databases or graphics).

Main OS Duties:

- Memory Management
- I/O Management

- CPU Scheduling
- File System Management
- Multitasking / Multiprogramming
- Communications & User Interface

💡 توضيح بالعربي: يعني إيه OS و Kernel؟

الكمبيوتر كـ Hardware غبي، الـ OS هو "المدير" اللي بينظم الشغل. من وجهة نظرك كمستخدم عادي، إنت عايز الجهاز يكون سريع وسهل (مش فارق معاك هو بيستهلك رامات قد إيه). لكن الـ OS كـ (Resource Allocator) وظيفته إنه يقسم الرامات والبروسيسور على البرامج بالعدل وكفاءة عالية.

أهم جزء في الـ OS اسمه **الـ Kernel (النواة)**. ده البرنامج الوحيد اللي شغال طول الوقت من أول ما تفتح الجهاز لحد ما تقفله، وهو اللي معاه الصلاحيات الكاملة. أي حاجة تانية بتطلب من الـ Kernel إنه ينفذلها شغلها.

2. The Boot Process (How a PC Starts)

When you turn on a computer, it goes through a specific sequence to load the Operating System into memory.

1. **BIOS / UEFI**
   Performs a Power-On Self Test (POST) to check if essential hardware (RAM, CPU, GPU) is working.
2. **Bootloader**
   Loaded by BIOS. Responsible for loading the Kernel into RAM. (e.g., GRUB for Linux, Windows Boot Manager).
3. **Kernel Loads**
   The Kernel takes control. It identifies devices, loads device drivers, and prepares the system.
4. **Init Process**
   Kernel starts the 'init' process. This launches all other services (login screen, network, background daemons).

💡 توضيح بالعربي: قصة الـ Boot Process

لما بتدوس على زرار الباور، الكهربا بتوصل للمذربورد. أول حاجة بتشتغل برنامج صغير متخزن على شريحة اسمها **BIOS**. الـ BIOS بيعمل حاجة اسمها **POST** (عشان يتأكد إن الرامات والبروسيسور راكبين وشغالين).

بعدين الـ BIOS بيروح للهارديسك يدور على برنامج اسمه **Bootloader** (زي الـ GRUB في لينكس). وظيفة الـ Bootloader الوحيدة إنه يروح يجيب **الـ Kernel** من الهارديسك ويحطه في الرامات (RAM).

أول ما الـ Kernel يصحى في الرامات، بيبدأ يتعرف على كارت الشاشة والماوس والكيبورد ويحمل تعريفاتهم (Drivers). وبعدين يشغل أول عملية في النظام اسمها **Init process**، ودي اللي بتفتحلك شاشة اللوجين وبتشغل باقي النظام.

3. System Architecture & Caching

Computer-System Architecture

- Most systems use a single general-purpose processor, but **Multiprocessor systems** (parallel/tightly-coupled) are growing.
- **Advantages of Multiprocessors:** 1. Increased throughput (do more work), 2. Economy of scale, 3. Increased reliability (fault tolerance or graceful degradation).

Asymmetric Multiprocessing (AMP)
Each processor is assigned a specific task. (Boss-worker relationship).

Symmetric Multiprocessing (SMP)
Each processor performs all tasks. Most common architecture today.

Caching

Information in use is copied from slower to faster storage temporarily.

- Faster storage (cache) is checked first. If data is there (cache hit), it's used directly.
- If not (cache miss), data is copied from slower storage to cache and then used.
- In a Multiprocessor environment, **Cache Coherency** is critical so all CPUs have the most recent value in their cache.

💡 توضيح بالعربي: الـ Caching والـ SMP

الـ SMP: تخيل عندنا مطعم فيه 4 طباخين (Processors). لو كل طباخ بيعمل كل حاجة (بيطبخ، بيغسل، بيقدم)، ده اسمه SMP (كلهم متساويين). أغلب الأجهزة الحديثة شغالة كدة.

الـ Caching: الرامات بطيئة جداً مقارنة بسرعة البروسيسور. فعملنا ذاكرة صغيرة جداً وسريعة جداً جوه البروسيسور اسمها **Cache**. لو البروسيسور محتاج داتا بيشوف الـ Cache الأول، لو لقاها بيستخدمها فوراً، لو ملقاهاش بيروح

يجيبها من الرامات ويحط نسخة منها في الـ Cache عشان لو احتاجها تاني.
المشكلة؟ لو عندنا كذا بروسيسور وكل واحد عنده Cache خاص بيه، لازم النظام يتأكد إن لو بروسيسور غير قيمة معينة، باقي البروسيسورات تعرف القيمة الجديدة دي، وده اللي بنسميه **Cache Coherency**.

4. Interrupts & I/O Structure

What is an Interrupt?

An operating system is **interrupt driven**. When a device (like a keyboard or disk) finishes a task, it sends a hardware signal (Interrupt) to the CPU.

- The OS preserves the state of the CPU by storing registers and the program counter.
- It looks up the **Interrupt Vector** (an array of memory addresses) to find the specific Interrupt Service Routine (ISR) for that device.
- **Trap or Exception:** This is a *software-generated* interrupt caused by an error (e.g., division by zero) or a user request (System Call).

Direct Memory Access (DMA)

Used for high-speed I/O devices (able to transmit close to memory speeds).

- Device controller transfers blocks of data from its local buffer **directly to main memory** without CPU intervention.
- Only **one interrupt is generated per block**, rather than one interrupt per byte. This saves a massive amount of CPU time.

💡 توضيح بالعربي: الـ Interrupts والـ DMA

الـ Interrupt: البروسيسور مش بيفضل يسأل الماوس "هل اتحركت؟ هل اتحركت؟". لا، البروسيسور بيشوف شغله العادي ولما انت تحرك الماوس، الماوس بيبعت إشارة (Interrupt) للبروسيسور يقوله "أنا اتحركت". البروسيسور بيسيب اللي في إيده، يحفظ هو كان واقف فين، ويروح ينفذ كود الماوس، ويرجع يكمل شغله.

الـ Trap/Exception: ده Interrupt بس بيحصل من الـ Software مش الـ Hardware. مثلاً برنامج حاول يقسم على صفر (Error)، أو برنامج طلب يفتح ملف من الهارديسك (System Call).

الـ DMA (Direct Memory Access): تخيل لو بننقل فيلم مساحته 2 جيجا من الهارديسك للرامات. لو البروسيسور هو اللي هينقل بايت بايت، هيتعطل ومش هيعمل حاجة تانية وهتجيله ملايين الـ Interrupts. الحل؟ شريحة الـ DMA بتنقل الفيلم كله للرامات لوحدها، ولما تخلص خالص بتبعت Interrupt **واحد بس** للبروسيسور تقوله "الفيلم اتنقل".

5. Dual-Mode Operation

Hardware provides at least two modes to protect the OS and other system components from malicious or buggy programs.

User Mode (Mode bit = 1)

Where normal application programs execute. Cannot execute privileged instructions directly (like accessing hardware or managing memory).

Kernel Mode (Mode bit = 0)

Also called supervisor mode. Complete access to all hardware and memory. OS runs in this mode.

How it works (The System Call)

1. User process executes in User Mode.
2. Process needs something from OS (e.g., read a file), so it makes a **System Call**.
3. This causes a **Trap**, and the hardware switches the **mode bit to 0** (Kernel Mode).
4. OS executes the requested service safely.
5. OS sets the **mode bit back to 1** (User Mode) and returns control to the user process.

The Timer

Used to prevent an infinite loop or a process from hogging resources. The timer interrupts the computer after a specific period, returning control to the OS to schedule another process.

💡 توضيح بالعربي: حماية النظام بالـ Dual Mode والـ Timer

لو أي برنامج (زي لعبة أو فيروس) قدر يتحكم في الهاردوير براحته، النظام هيقع. عشان كدة قسمنا الشغل لمستويين:

User Mode (وضع المستخدم): ده سجن للبرامج العادية، مفيش برنامج يقدر يكلم الهاردوير مباشرة.

Kernel Mode (وضع النواة): ده للمدير بس (الـ OS).

لما البرنامج بيعوز يطبع ورقة مثلاً، بيعمل طلب رسمي اسمه **System Call**. النظام بيغير المود لـ Kernel، يروح يطبع الورقة بأمان، وبعدين يرجع المود لـ User تاني.

الـ Timer: تخيل برنامج دخل في Infinite Loop (حلقة مفرغة)، البروسيسور هيفضل مشغول للأبد والجهاز هيهنج! الحل إن الـ OS بيظبط "منبه" (Timer) في الهاردوير، كل شوية المنبه ده يرن (يعمل Interrupt)، فيسحب التحكم من البرنامج ويرجعه للـ OS عشان يشوف لو في برنامج تاني محتاج يشتغل.

6. Process Management & Threads

Program: Passive entity (a file on disk).
Process: Active entity (a program in execution). It's an execution environment with restricted rights.

Inside a Process Address Space:

- **Code / Text:** The compiled program instructions.
- **Data:** Global and static variables.
- **Heap:** Dynamically allocated memory during runtime.
- **Stack:** Local variables and function call data.
- Owns memory, file descriptors, and sockets.

Threads (Virtual Cores)

- A single-threaded process has one program counter (executes one instruction at a time).
- A **multi-threaded process** has one program counter per thread.
- Threads within the same process **share the same memory (Code, Data, Heap) and files**, but each thread has its own **Registers and Stack**.

Multiprogramming & Multitasking

- **Multiprogramming (Batch system):** Organizes jobs so the CPU always has one to execute. When a job waits for I/O, OS switches to another.
- **Timesharing (Multitasking):** Logical extension where CPU switches jobs so frequently that users can interact with each job while it's running (response time < 1 sec).

💡 توضيح بالعربي: الـ Process والـ Thread

الـ Program: هو الأبلكيشن وهو مقفول وموجود على الهارديسك (ميت).
الـ Process: هو الأبلكيشن لما تفتحه (بقى حي واشتغل في الرامات). الـ OS بيعزل كل Process عن التاني عشان لو متصفح كروم هنج، الـ Word ميفصلش معاه.

الـ Thread: جواه الـ Process، ممكن نعمل كذا مهمة في نفس الوقت. الـ Thread هو "عامل" جوه المصنع (الـ Process).
مثلاً: فاتح كذا تاب في كروم؟ كل تاب عبارة عن Thread. العمال (Threads) جوه نفس المصنع بيتشاركوا في الموارد (Data, Heap, Files)، بس كل عامل ليه شنطة عدته الخاصة بيه (Stack & Registers).

7. Memory & Storage

Storage Definitions

- **Bit:** The basic unit of storage (0 or 1).
- **Byte:** 8 bits (smallest convenient chunk).
- **Word:** Native unit of data for CPU (e.g., 64-bit architecture uses 8-byte words).
- **Capacities:** Kilobyte (KB) = 1,024 bytes, Megabyte (MB) = $1,024^2$ bytes, Gigabyte (GB) = $1,024^3$ bytes, Terabyte (TB) = $1,024^4$ bytes, Petabyte (PB) = $1,024^5$ bytes.

Storage Hierarchy

Storage systems are organized in a hierarchy based on Speed, Cost, and Volatility.

Level	Name	Typical Size	Volatility	Managed By
1 (Fastest/Costly)	Registers	< 1 KB	Volatile	Compiler
2	Cache	< 16 MB	Volatile	Hardware
3	Main Memory (RAM)	< 64 GB	Volatile	Operating System
4	Solid-State Disk (SSD)	< 1 TB	Non-volatile	Operating System
5 (Slowest/Cheap)	Magnetic Disk (HDD)	< 10 TB	Non-volatile	Operating System

Memory Management

- To execute a program, all (or part) of its instructions and data must be in memory.
- **Swapping:** If processes don't fit in memory, the OS moves them in and out of the disk to run.
- **Virtual Memory:** Allows execution of processes that are not completely in memory, giving the illusion of a massive RAM.
- **Address Translation:** The OS (with hardware TLB/Page Tables) translates "Virtual Addresses" generated by the CPU to actual "Physical Addresses" in memory.

💡 توضيح بالعربي: الـ Virtual Memory والـ Swapping

الرامات مساحتها محدودة (مثلاً 8 جيجا). لو فتحنا برامج محتاجة 12 جيجا، الجهاز مش هيقفل. ليه؟
الـ OS بيستخدم جزء من الهارديسك كأنه رامات احتياطية (بنسميه **Virtual Memory**). البرامج اللي مش شغالة حالياً (في الخلفية)، الـ OS بياخدها من الرامات يرميها على الهارديسك (عملية اسمها **Swapping Out**)، عشان يفضي مساحة للبرنامج اللي انت شغال عليه دلوقتي. ولما ترجع للبرنامج القديم، بيجيبه من الهارديسك للرامات تاني (**Swapping In**).

عشان كدة كل Process فاكرة إن معاها الذاكرة كلها لوحدها (Virtual Address)، والـ OS هو اللي بيترجم العناوين الوهمية دي للعناوين الحقيقية (Physical Address) في الرامات.

8. Protection and Security

- **Protection:** Any mechanism for controlling access of processes or users to resources defined by the OS. (Ensuring software doesn't step on each other).
- **Security:** Defense of the system against internal and external attacks (e.g., denial-of-service, worms, viruses, identity theft).
- **User Identities:** Systems distinguish among users via **User IDs** and **Group IDs** to determine access control.
- **Privilege Escalation:** Allows a user to temporarily change to an effective ID with more rights (like "Run as Administrator").

💡 توضيح بالعربي: Protection vs Security

الفرق بينهم دقيق بس مهم جداً:

الـ Protection (الحماية الداخلية): دي القواعد اللي بتمنع برنامج يمسح ملفات برنامج تاني بالغلط، أو إن يوزر عادي يطفي السيرفر. دي سياسة داخلية لتنظيم الموارد.

الـ Security (الأمان الخارجي والداخلي): ده خط الدفاع ضد الهجمات المتعمدة زي الفيروسات، الهاكرز، أو إن حد يسرق الباسوورد.

9. Computing Environments

Client-Server Computing

Dumb terminals supplanted by smart PCs. Many systems now act as servers responding to requests generated by clients.

- **Compute-server:** interface to client to request services (e.g., database query).
- **File-server:** interface for clients to store and retrieve files.

Mobile Computing

Handheld smartphones and tablets. They have extra OS features (GPS, gyroscope) for apps like augmented reality. Leaders: Apple iOS, Google Android.

Cloud Computing

Delivers computing, storage, even apps as a service across a network. It uses **Virtualization** as its base.

- **SaaS (Software as a Service):** Applications available via Internet (e.g., Google Docs, word processors).
- **PaaS (Platform as a Service):** Software stack ready for application use via the Internet (e.g., a database server).
- **IaaS (Infrastructure as a Service):** Servers or storage available over Internet (e.g., AWS EC2).

10. Linux Deep Dive (Section 1)

History of Linux

- **1969 - MULTICS:** Created by Bell Labs (AT&T), GE, and MIT.
- **1970 - UNIX:** Bell Labs continued to create UNIX (written in C). Licensed to academics and companies.
- **GNU Project:** Richard Stallman started GNU to make open-source software free (created the GPL license).
- **1991 - Linux:** Linus Torvalds developed a new kernel based on UNIX and called it Linux. It combined with GNU tools to become what we use today.

Linux Architecture

```
Hardware➜ Kernel➜ Shell (bash) / API Libs ➜ Terminal / Applications ➜ User
```

- **Kernel:** Core of the OS. Handles devices and memory. Stays in RAM.
- **Shell:** Interface for user to communicate with kernel. Parses commands. (e.g., `bash`).
- **Terminal:** Gives the shell a place to accept typed commands and display results.

User Interactivity (How to use Linux)

- **GUI (Graphical User Interface):** Interacting visually with windows and mouse.
- **CLI (Command Line Interface):** Typing commands via CMD or Shell.
- **Remote Login (SSH Protocol):** A secure network protocol used to securely log into remote computers and transfer files safely.

Linux Distributions (Distros)

A distro is the Linux Kernel packaged with different software, tools, and UI. Major families:

- **Debian Family:** Ubuntu, Linux Mint.
- **RHEL Family:** CentOS, Fedora.
- **SUSE Family:** openSUSE, SLES.

Basic Linux Commands

```
ls                                List files in current directory.
ls -l                             List in "long format" (details).
ls -l file_name                   List details for a specific file.
```

```
cat filename                      Show contents of a file.
cat -n filename                   Show contents with line numbers.
```

```
uname                             Print kernel name.
uname -n                          Print user/network name.
uname -a                          Print all system information.
```

```
cal 10 2023; date; uname          Semicolon ; runs multiple commands sequentially.
```

```
command --help
man command                       Different ways to get help/manuals for a command.
whatis command
```

Useful Shortcuts:

- **Ctrl + C:** Interrupt a command that's taking too long.
- **Ctrl + L:** Clear the terminal screen.