AI224: Operating Systems — Lecture 01

Dr. Karim Lounis

Jan - Jun 2025

Who am I?

Dr. Karim Lounis

Assistant Professor at ENISA (since October 2022).

Research in Information Security.

Ph.D. in Computing (Queen's Univ, CA, 2020).

MSc in IS Security (Paris-XII Univ, FR, 2014).

MSc in Networks & DSs (USTHB, DZ, 2013).

BSc in Networks & Telecom (USTHB, DZ, 2011).

Research Assistant in cybersecurity (DE, LUX, & CA).

Can visit my website for more details: https://lounis.weebly.com/.

Shoot an email to: karim.lounis@ensia.edu.dz.

Walk in to office: L2-111 (Appointment by email).

Operating Systems

Some Questions

What is this course about?

Some Questions

Why are you taking this course?

Some Questions

Why should you care about taking this course?

Motivation

There are various reasons why you should like and enjoy this course:

- If you are using a computer (e.g., WS, PC, mac, tablet, smartphone, etc), then you are using an operating system. Could you use the computer, as it is supposed to be used, without having an operating system?
- You've been using an operating system, but you may not know what is it, and you may certainly not know its functions. What's happening under the hood? what happens when you run a program?
- You've learned how to write and run interesting programs, but you do not know how they interact with an operating system. What happens when you double-click on a program? or when it crashed (& why)?
- With advencement in operating systems, you are capable of running multiple programs and applications simultaniously without issues. How is this possible? I may have one processor and limited memory.

Motivation

There are various reasons for why you should like and enjoy this course:

- You may have 8 to 32GB of memory (RAM) to run programs. How is it possible to smoothly run a 60GB program (e.g., GTA5) without any issues (& even simultaniously run various large programs)?
- You are used to simultaniously download files, make a skype-call, browse the net, and send emails without encountering any kind of interference. You only have one network card, how is this happening?
- You've certainly got the chance to write a program that runs a dirty infinite loop, while other programs are running. How comes that this dirty program did not influcene and block the others from running?
- You usually run multiple programs at the same time without any issues. Have ever gone through a scenario where the webpage you requested appeared in your skype call app, where the face of the person you were talking to appeared on the web-browser (did not get mixed up)?

Motivation

There are various reasons for why you should like and enjoy this course:

- You may have created and used files in various applications (e.g., text, charts, database, etc). Have you ever wondered how different files are stored in your disk drives?
- You can run various applications (or apps) on your system. Did you ask your self how does the operating system keep track of each individual application without problems?
- Your operating system provides you with ability to control access to your files, folders, and resources. Do you know how does it do that, and which mechanisms are used?
- You may have experienced the scenario where your computed was hacked, or a malware was cracking up it fireworks. How do malware exploit your operating system flaws to harm your system)?

What does all of this have with the course?

Let's start by figuring out what an Operating System is

What is an Operating System?

An operating system is ...?

What is an Operating System?

An operating system is an intermediary layer, between the hardware (tangible part) and users' software (e.g., applications)

What is an Operating System?

Definition

It is a set of programs (called modules) that cooperate for the good management and use of computer resources.

These modules can be classified into two classes, primary and secondary. The primary modules constitute the kernel of an operating system.

- **Primary modules (Kernel):**
  - Process management.
  - Memory managment module.
  - File system module.
  - Interrupt handling module.
  - Other?
- **Secondary modules:**
  - Drivers.
  - Networking module.
  - Security managment module.
  - The Shell
  - Others?

So, a set of programs (modules) that create a program-friendly environment for user programs to fairly, safely, and reliably execute till completion.

What is an Operating System?

Primary modules: [Details not discussed today — other chapters]

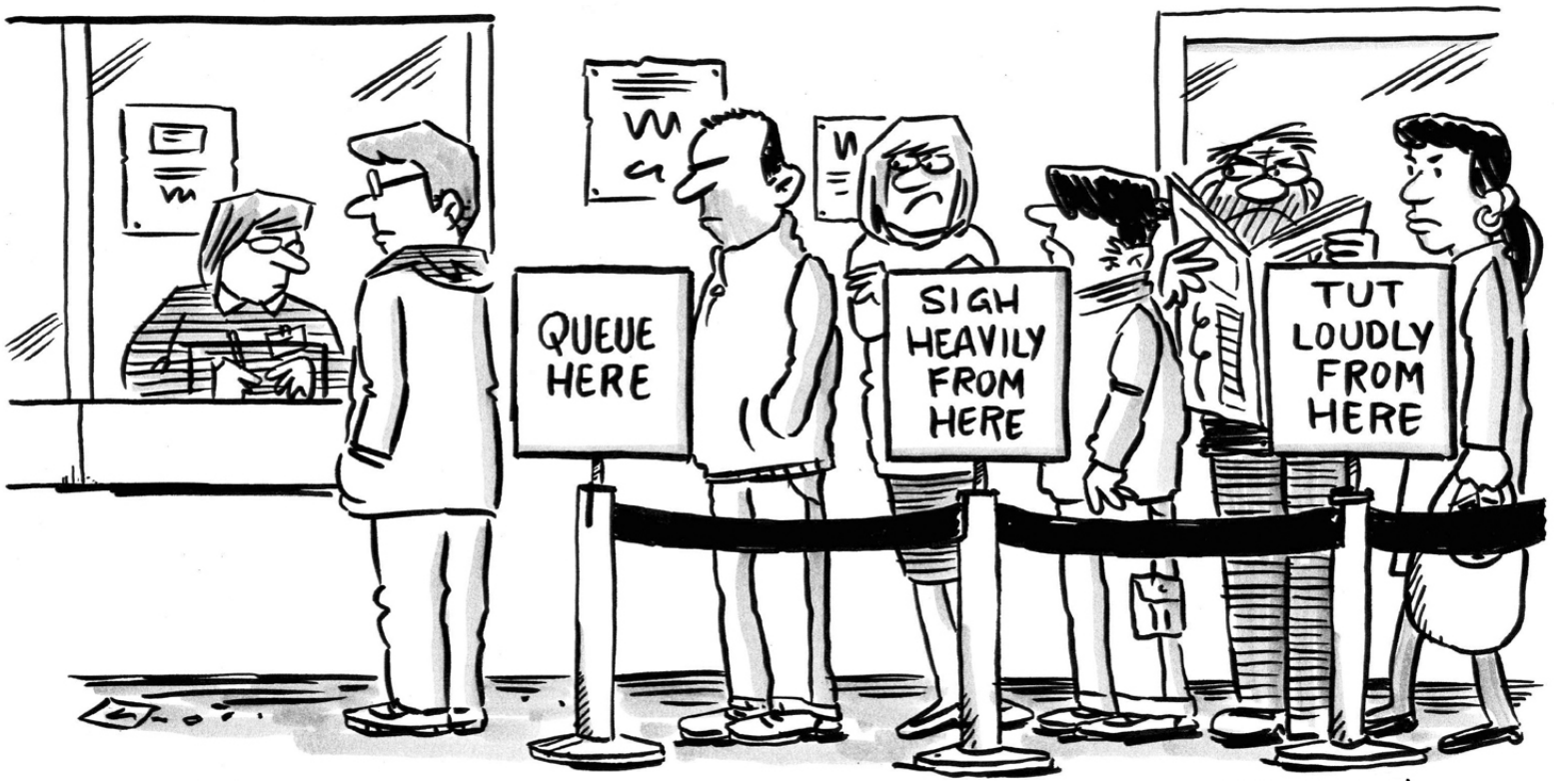
- Process management module. Responsible for creating, executing, and terminating processes. It includes a process scheduler, that decides which process will run next and when it should be stoped, based on a scheduling algorithm, e.g., FIFO, SJF, STRF, RR, and PR.
- Memory managment module. Decides where to load a process in the memory (RAM) and how to keep track of each portion of the memory. It allocates and frees space in the central memory — RAM.
- File system module. Keeps track of files, their identity, type, how they are stored, and where. Also, how the hard drive is logically structured (file systems: NTFS, FAT16, FAT32, EXT4, etc).
- Interrupt handling module. Handles interrupts: hardware (e.g., network traffic, keystrokes, mouse, etc), exception (e.g., division by 0, overflow, etc), and system calls (e.g., read and write on I/O devices).

What is an Operating System?

Secondary modules: [Details not discussed today — other chapters]

- Drivers. These are system programs that interface between the operating system and certain hardware devices, e.g., advanced keyboard, network cards, printer, graphical card, etc.
- Networking module. It implements a list of functions and services, called communication protocols (e.g., TCP/IP protocol suite), to allows inter/inner-computer communication.
- Security managment module. Provides security services, such as encryption, authentication, data integrity, and availability.
- The Shell. Is the program that implements the interface of interaction, either as command-line interface (CLI) and/or a graphical user interface (GUI), between the user and the kernel. In the GUI-Shell, users can run programs by clicking and executing them, whereas in CLI-Shell, programs are run by executing commands (e.g., `kill -9 pid`).

Which Services does an Operating System Provide to Programs (Applications) and Users?



Which Services does an Operating System Provide?

Services provided by the operating system to the users ({programs}):

User Interface. Could be a Batch Interface[1] [1945-1968], a CLI (Command Line Interface)[1969-] or GUI (Graphical User Interface) [1970-].

Program Execution. Programs need to be loaded into the main memory and get executed till completion. The operating system ensures that.

I/O Operations. Running programs may require I/O operations, the operating system provides special functions to talk to I/O devices.

File System Manipulation. Programs may need to manipulate files. The operating system provides special functions for such manipulations.

Communication. Programs may need to communicate with each other, locally (e.g., pipes & shared memory) or remotely. The OS provides special functions to guarantee a safe, secure, and reliable communication.

[1] Batch OS — consists of having an operator rearranging and submiting group of similar jobs (a.k.a., batch) for a sequential execution — punchcards in & punchcards out.

Which Services does an Operating System Provide?

Error Detection. The operating system needs to detect and correct errors when they occur during programs execution (e.g., device failure). For each type of error, the operating system takes the appropriate action.

Services provided by the operating system to the users and system itself:

Resource Allocation. The operating system ensures that resources are shared in an optimal manner among multiple users and jobs.

Accounting. The operating system keeps track of which user uses how much and what kind of resources for accounting and statistics.

Protection and Security. The operating system provides certain security services to its users (or processes) to preserve their security and privacy.

When a program executes, it constitutes a **process**, a **task**, or a **job**.

How Does an Operating System Start? (several stages)

To launch the OS, the **kernel** needs to be located and loaded into the memory then executed (Booting the system is loading the **kernel**):

1. When you power on you computer, the processor (CPU) starts executing a micro-program, called BIOS[2], which initialize hardware components and invokes the bootstrap loader.
2. The bootstrap loader, looks up for the a program called bootloader (e.g., GRUB in GNU/Linux). The bootloader starts execution, looking for a kernel to load (may ask user in case of dual-boot).
3. The bootloader loads the kernel into memory and executes it (we say, the system is running — has booted).

[2]UEFI (Unified Extensible Firmware Interface) is a successor to BIOS (Basic Input/Output System), aiming to address its technical shortcomings.

Wait! Has the Course Already Started?

Not yet. This is just a warm up.

What resources are available to swim in this course?.

The course website (your best reference):

https://sites.google.com/ensia.edu.dz/ai224-w25

You will find everything there (well, not everything, but things you aresupposed to have access to — you'll not find the exam sheet there).

The syllabus, course objectives, course evaluation, teaching team, etc.

What Do You Need to Do to Succeed in this Course?

Few things:

1. Attend the course lectures, focus while I am talking, ask questions if not clear. Make sure you leave the session with minimum ambiguities.
2. Feel free to use the Question Box — answers will be posted.
3. Prepare and complete your labs.
4. If you did not understand something during the lecture or lab sessions, try not to miss office hours.
5. In the worst case, shoot an email (avoid after 9:00 PM).

More things: Class policy.

When Do We Start Then?

?

We've Actually Started

Types of Operating Systems (Terminology)

From operating system's program execution perspective, we find:

- Single-tasking operating systems — limited systems.
- Multiprogramming operating systems — exploiting RAM to it max.
- Multiprocessing operating systems — exploiting multiple CPUs.
- Multithreading operating systems — supporting complex processes.
- Multi-tasking operating systems (Time-sharing systems)

Vs

Types of Operating Systems (Terminology)

From operating system's structural perspective, we find:

- Traditional operating systems — used on general-purpose computers.
- Firmware with OS-like features systems — used on network devices.
- Pure firmware — Used on special devices, e.g., washing machine.

Types of Operating Systems (Terminology)

From operating system's application perspective, we find:

- General-purpose "personal" operating systems.
- Embedded operating systems.
- Real-time operating systems.
- Distributed operating systems.
- Network operating systems.
- Mobile operating systems.

- End