Lecture 1 —Introduction, Operating Systems, Security

Seyed Majid Zahedi (smzahedi@uwaterloo.ca)

Slides adapted from Jeff Zarnett (jzarnett@uwaterloo.ca)

Department of Electrical and Computer Engineering
University of Waterloo

SE 350

Course Syllabus

As our first order of business, let's go over the course syllabus.

Collaborative Course

To access all course material, go to `https://eceweb.uwaterloo.ca/~smzahedi/crs/se350/`.

To enroll in the course on Piazza, go to `https://piazza.com/uwaterloo.ca/winter2026/se350`.

To report errors in notes/slides, open an issue at `https://github.com/smajidzahedi/SE-OS`.

If you know how to use `git` and LaTeX, then you can go to the URL and submit a pull request (changes) for me to look at and incorporate!

Readings

There is no mandatory textbook for this course; the lecture notes should be sufficient.

Optional references:

- Operating Systems: Principles and Practice
- Operating Systems: Three Easy Pieces
- Operating System Concepts

Let’s talk about labs!

Introduction to Operating Systems

Operating systems are those programs that interface the machine with the applications programs. The main function of these systems is to dynamically allocate the shared system resources to the executing programs.

- What Can Be Automated?: The Computer Science and Engineering Research Study, MIT Press, 1980

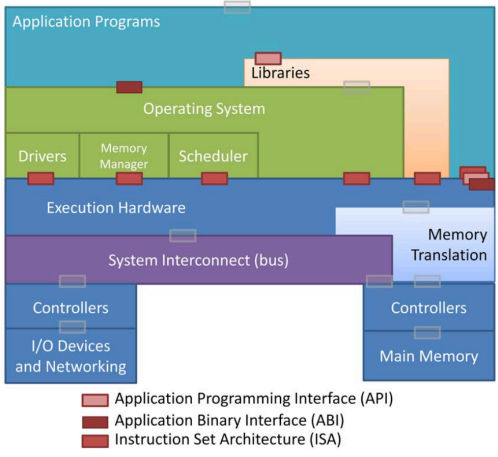
Introduction to Operating Systems

An operating system (OS) sits between the hardware and programs.

It has many goals, that often conflict with one another.

Its job is to make it so other programs can run efficiently.

Structural Diagram of a Modern Computer



OS: Resource Manager

The OS is responsible for resource management and allocation.

Resources like CPU time or memory space are limited.

The OS must decide how to allocate & to keep track of system resources.

In the event of conflicting requests, choose the winner.

OS: Environment Provider

The OS enables useful programs like Photoshop or Microsoft Word to run.

The OS is responsible for abstracting away the details of hardware.

This is so program authors do not have to worry about the specifics.

Imagine Hello World had to be written differently for different hardware.

OS: Multitasking

Multiple programs means some resources are shared.
→ A source of conflicts!

OS creates and enforces the rules so all can get along.

Sometimes processes want to co-operate and not compete.
The OS can help them to do so.

OS: Efficiency

Another goal may be to use the computer hardware efficiently.

Image Credit: Argonne National Laboratory

Any moment when the supercomputer is not doing useful work is a waste.

OS: What is it, really?

Operating systems tend to be large and do a lot of things.

We expect now that an OS comes with a web browser, an e-mail client, some method for editing text, et cetera.

The part of the operating system we will study is the Kernel.

The kernel is the “core”; the portion of the OS that is always present in main memory and the central part that makes it all work.

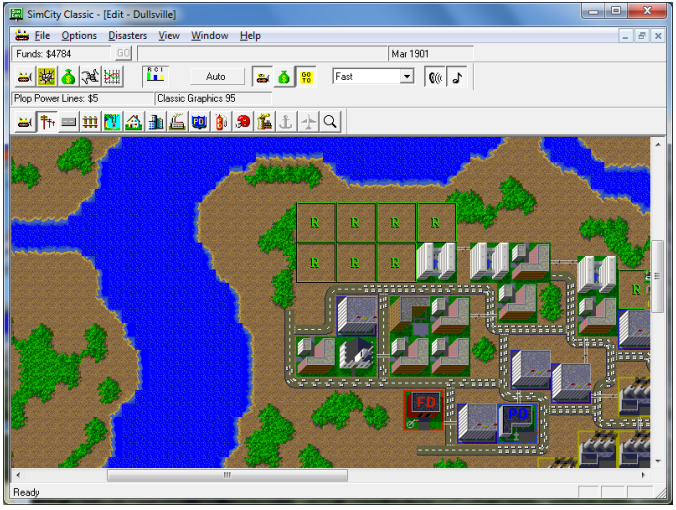
OS: Evolution

Operating systems will evolve over time.

There will be new hardware released, new types of hardware, new services added, and bug fixes.

Evolution is constrained: a need to maintain compatibility for programs.

Example: SimCity and Windows 95



Security Now

Security Now

In many textbooks, security and protection are left to the end.

Security has to be designed from the beginning; cannot be bolted on.

An OS supports multiple users with multiple tasks.

Tool Support

OS designers create policies and policy tools.

There is a tradeoff with usability.

You do NOT want to have to report a data breach.

Enforcement is Everything

The OS must enforce the configured policies.
Otherwise, malicious users exploit it.

Three desirable properties: Confidentiality, Integrity, Availability.

Protection vs Security

Protection is about internal threats.

Security is about external threats.

We'll take them in order.

Protection

Protection

Most of the discussion in this course is about protection.

Following the rules is important to a functioning system or society.

Even in the absence of malicious intent.

Goals of Protection

Enforce policies about responsible usage.

Responsible and reasonable vary across systems.

Examples of access control: file permissions, walls between processes.

Enforcement is Key

Rules take effort to enforce.

Exceptions are allowed and administrators can override policies!

Other kinds of rules can exist.

Context is Key

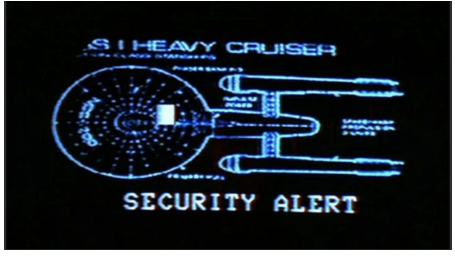
If I edit a lecture video, it consumes a lot of CPU and RAM.

Is that bad?

It's legitimate – but might set off alarms.

So we may be lax – which gives an opening to malicious users…

Security



Let’s consider a few bad things attackers can do.

Why? We need this in mind when designing the system.

Security Problems

- Breach of Confidentiality
- Breach of Integrity
- Breach of Availability
- Theft of Service
- … and others.

Most likely the weak link is people.

Specific Attacks

A few potential specific attacks:

- **Excessive Requests**
- **Malformed Requests**
- **Back Door**
- **Intercepting Messages**
- **Trojan Horse**

Arms Race

Security is an arms race.

Can we break things if it’s a security problem?

No easy answers.

Security, Conclusion

This is not a course in security, but we can’t ignore it.

Security needs to be included from the beginning!

The Plan

A broad overview of the major topics of the course:

1. Introduction, Security, Review of Architecture
2. Processes, Inter-Process Communication
3. Threads, Concurrency,Synchronization, Deadlock
4. Memory
5. Scheduling
6. I/O Devices, File Systems