CS-477

Advanced operating systems

Kashyap Sanidhya

Cursus	Sem.	Type
Computer science minor	H	Opt.
Computer science	MA1, MA3	Opt.
Cyber security minor	H	Opt.
Cybersecurity	MA1, MA3	Opt.
SC master EPFL	MA1, MA3	Opt.

Language of teaching	English
Credits	6
Session	Winter
Semester	Fall
Exam	During the semester
Workload	180h
Weeks	14
Hours	**5 weekly**
Courses	2 weekly
Exercises	1 weekly
Lab	2 weekly
Number of positions	

Summary

This course teaches the design and implementation of modern operating systems through the lens of the Linux kernel. Using an extensive hands-on approach, students study how classical OS principles are realized in a production kernel and explore new design ideas proposed over the past two decades.

Content

- OS architecture: kernel design space, isolation, and the system call interface
- Kernel data structures and debugging
- Concurrency and synchronization: locking, lock-free data structures, and multicore scalability
- Process management and scheduling
- Interrupts and kernel execution contexts
- Memory management: address spaces, virtual memory, and memory allocation
- Storage: VFS, page cache, file systems, and the block layer
- Virtualization
- OS security: data security, integrity, and authentication

Note that topics are tentative and will be explored at different depths. Material discussed in previous years is an indication of what may be covered, but not a guarantee, as operating systems remain a fast-moving target. In-class participation is strongly recommended.

Keywords

Operating systems, Linux kernel, concurrency, scalability, memory management, file systems, virtualization, OS security

Learning Prerequisites

Required courses

CS-200 Computer architecture
CS-202 Computer systems (or COM-208 Computer networks / CS-323 Introduction to operating systems)
CS-214 Software construction (or CS-210 Functional programming)
CS-300 Data-Intensive Systems (or CS-322 Introduction to database systems)

Important concepts to start the course

This is a hands-on course that requires a solid background in operating systems, computer architecture, and systems programming in C. Students must be comfortable reading and writing low-level code.

Learning Outcomes

By the end of the course, the student must be able to:

- Identify and manage key components of operating systems
- Choose or critique design choices for OS
- Create and design an OS with practical choices
- Report performance and possible optimizations for applications

Teaching methods

- In-class lectures
- Weekly exercise sessions preparing for the labs
- Programming labs
- A semester-long project

Expected student activities

- Attend lectures and tutorial sessions
- Participate actively in the course
- Complete programming exercises

Assessment methods

- Labs
- Project
- Quiz

Supervision

Office hours	Yes
Assistant.e.s	Yes

Resources

Moodle Link

- https://go.epfl.ch/CS-477