EECS 678: Introduction to Operating Systems
About This Class

• Textbook: Operating System Concepts
• Objectives: Learn OS basics and practical system programming skills
• Audience: Senior and Junior undergraduate (grad students)
• Course website: http://ittc.ku.edu/~heechul/courses/eecs678/S15
About This Class

• Course structure
  – Lecture: TR 8:00 – 9:15 LEA 2112
    • Office hour: TR: 9:15 - 10:15 @ 3040 Eaton
  – Lab: Mon12:00 – 01:50 p.m. 1005B Eaton
    Wed 9:00 - 10:50 a.m 1005B Eaton
    Wed 4:00 - 5:50 p.m. 1005D Eaton
    Fri12:00 – 01:50 p.m. 1005B Eaton
  – Programming projects

• Grading
  – Class attendance: 10%
  – Midterm exam: 20%
  – Final exam: 30%
  – Lab evaluations: 20%
  – Programming projects: 20%
Disclaimer: some slides are adopted from Prof. Kulkarni’s and the book authors’ slides with permission
Introduction – Objectives

- To describe the basic organization of computer systems
- To provide an overview of the major components of operating systems
What is an Operating System?
What is an Operating System?

• A program that acts as an intermediary **between a user** of a computer and the computer **hardware**

• Operating system goals:
  • Make the computer system **easy to use**
  • Use the computer hardware **efficiently**
Roadmap

• CPU management
• Memory management
• Disk management
• Network and security
• Virtual machine and distributed systems
Computer System Structure

- User 1
- User 2
- User 3
- User n

- Compiler
- Assembler
- Text Editor
- System and Application Programs
- Database System
- Operating System
- Computer Hardware
Computer System Structure

- Hardware
  - Provides basic computing resources: CPU, memory, I/O devices

- Operating system
  - Controls and coordinates use of hardware among various applications and users

- Application programs
  - Solve the computing problems of the users: word processors, compilers, web browsers, video games, ...

- Users
  - People, other computers
What Does an Operating System Do?

- Provide abstractions
  - For easy of use (User’s goal)
  - E.g.,) Process, thread, file, directory, virtual memory, socket,…

- Manage hardware resources
  - For efficient use of the hardware (System’s goal)
  - E.g.,) CPU scheduling, memory management, I/O scheduling, …
User's View of the Operating System

- Hide the details of the underlying hardware
- Provide powerful APIs
  - e.g.) write(fileno, buf, len);
- Provide various services
  - e.g.) Execute a program
    - loading program into memory, initializing program state, maintaining program counter, stopping the program
    - instead user types: `gcc hello.c` and then `./a.out`
  - fair scheduling, filesystem, performance monitoring, ...
System's View of the Operating System

- Allocate resources
  - manage CPU time, memory space, file storage space, I/O devices, network, etc.

- Control programs/hardware
  - Controls execution of programs to prevent errors and improper use of the computer
A Typical Computer System

– One or more CPUs, device controllers connect through common bus providing access to shared memory

– Concurrent execution of CPUs and devices competing for memory cycles
CPU Architecture

- **Unicore**, Multicore, Multiprocessor
CPU Architecture

- Unicore, **Multicore**, Multiprocessor
CPU Architecture

• Unicore, Multicore, Multiprocessor
Storage Structure

• Main memory – only large storage media that the CPU can access directly
  – Random access
  – Typically volatile

• Secondary storage – extension of main memory that provides large nonvolatile storage capacity
  – Hard disks – rigid metal or glass platters covered with magnetic recording material
  – Solid-state disks – faster than hard disks, nonvolatile
Storage-Device Hierarchy
Caching

- Important principle, performed at many levels in a computer (in hardware, operating system, software)
- Information in use copied from slower to faster storage temporarily
- Faster storage (cache) checked first to determine if information is there
  - If it is, information used directly from the cache (fast)
  - If not, data copied to cache and used there
- Cache smaller than storage being cached
  - Cache management important design problem
  - Cache size and replacement policy
How a Modern Computer Works

A von Neumann architecture
Operating System Operations

- Bootstrapping and Interrupt handling
- User/Kernel mode transition
- Process management
- Memory management
- Storage management
- Security/Protection
- Virtualization/Cloud Computing
Bootstrap

- **Bootstrap program** for the computer to start running
  - Initialize memory (DRAM) and boot device
  - locate and load the OS kernel
- OS starts executing the first program
  - waits for some event (timer, I/Os, syscall) to occur
  - an operating system is interrupt driven
Interrupt Handling

- Interrupt handling process
  - save CPU states (registers)
  - execute the associated interrupt service routine
  - restore the CPU states
  - return to the interrupted program