Operating Systems

Introduction

Lecturer
Asaad Alhijaj
2019

Content of this lecture

- Course information (personnel, policy, prerequisite, agenda, etc.)
- Why learning OS?
- What is an OS? What does it do?
- Summary

Why learning OS?

- Fulfill requirement?
- Operating System knowledge is important
  - http://www.youtube.com/watch?v=-3Rt2_9d7Jg
    - Which course is this?
    - http://matt-welsh.blogspot.ca/2010/10/in-defense-of-mark-zuckerberg.html
  - Software companies love OS students
  - Most big software companies have system positions
- Academic research in OS is very influential

Goals of this course

- Understand operating system concepts
- How OS works, and more importantly, ***why***?
  - What are the reasons motivated each design?
- Basis for future learning
- Other hands
  - You will *implement* parts of a real OS
- ***Train your problem solving skills!***
  - *Face a problem, solve it, instead of come up with a theory and find applications*

Prerequisite

- Programming experiences (C, C++, Java)
  - How many of you know C ? Java ?
    - You will be programming in C (it's OK if you only know Java)
- Computer organizations
- What is an ***Instruction*** (e.g., *load*, *store*)?
  - What is ***CPU*** ? ***Memory*** ? ***Registers*** ?
  - What is ***Stack*** ? ***Stack pointer*** ?
  - What is ***Program Counter*** (PC)?

Course Contents

Overview

Introduction

Operating system structures

Process Management

Processes

Threads

CPU Scheduling

Process Synchronization

Deadlocks

Memory management

Main Memory

Virtual memory.

Storage management

File System Interface

Advanced topics

What to Expect From Lab Assignments

- Building an OS is difficult
  - Perhaps the hardest lab in your undergraduate study
  - OS: one of the hardest program to write & debug
- Principles may sound easy, implementation is *extremely* hard
  - The labs give specifications, not implementations
- Hack into a large, unfamiliar code base and implement additional features
- You will spend a lot of time on the lab assignments
  - Allows for imagination
  - Allows for errors and frustration
  - Lab instructions ask that you design well, before you code
  - Assume that you will do the design/coding outside lab hours

But it is rewarding!

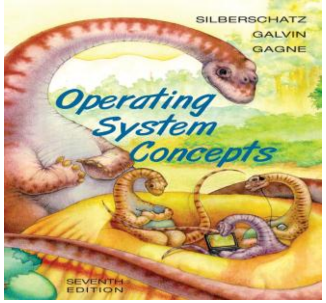
- Solid understanding of how an OS works
  - Appreciations on the implementation efforts that make things work
- Technical capability
  - Again: OS is one of the hardest programs to write and *debug*
  - Quickly hack into unfamiliar code base
- You will work in ***groups of 2*** for the lab assignments
  - Make sure you know what your partner is implementing
  - Learn to coordinate and be efficient
  - ***Form your group by March 28th, 10:30 AM***

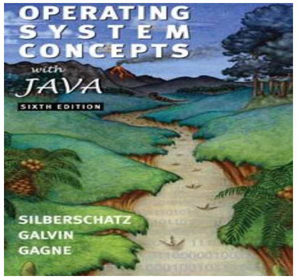
Suggested Textbooks

(Main textbook)

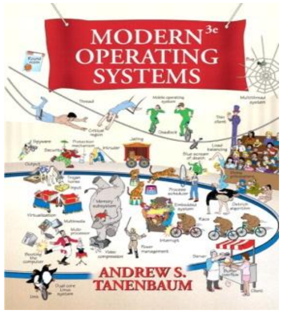
- Operating system Concepts, Silberschatz, Galvin, and Gagne, 7th edition, 2005,

  john wiley & sons inc., USA.





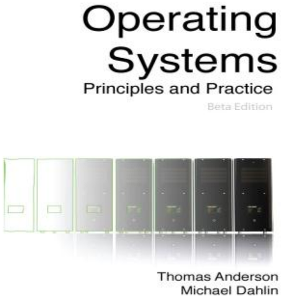
- Modern Operating Systems, 3nd Edition , Andrew S. Tanenbaum



- Operating Systems: Principles and Practice

(Further reading)

Thomas Anderson, Michael Dahlin



Communications

- Class web site available from instructor’s home page
  - http://un.uobasrah.edu.iq/lecturer_signin.php
  - http://www.edmodo.com/ the group code is: Each student should register as a student in the website
  - Provides slides, agenda, grading policy, etc.
  - All information regarding the labs

Exam

- 1st **Exam** Thr 11/4/2019
  - Covers first half of class
- 2nd **Exam** Thr 8/5/2019
  - Covers second half of class
- 3rd Exam Thr 16/5/2019 (**Optional**)
  - Covers selected topics from first half and second half of class
- Final
  - Covers second half of class + selected material from first part
- Project-related knowledge may be included in the exams
  - *So do your project and do NOT copy!*
  - the project cannot be done in the last few days!

What is an OS?

- Anyone?

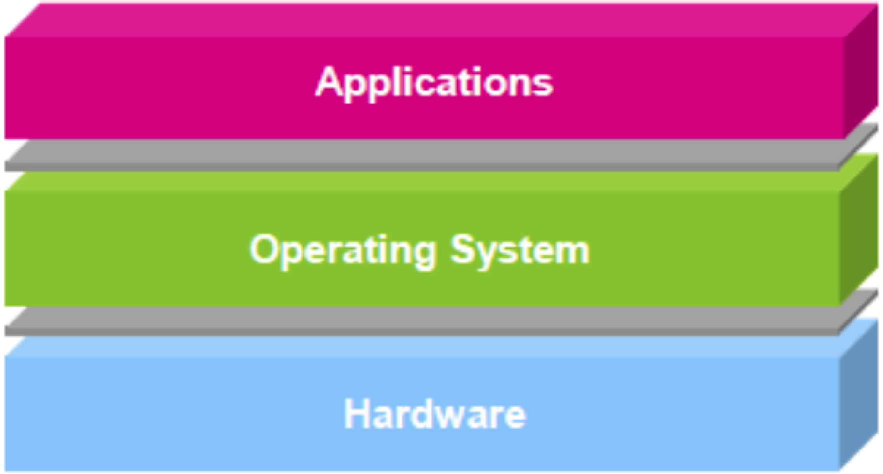
- Give a few names of an OS?
  - Desktops?
  - Smart phones?

What is an OS?

- "Code" that:
  - Sits between programs & hardware
  - Sits between different programs
  - Sits between different users
- But what does it do?
  - Managing the hardware resource
  - Provide a clean set of interface to programs
- A good OS is a piece of software that normally you shouldn't notice of its existence
  - But you feel the pain if it goes wrong
- Real life analogy?
  - Government

OS is…

- *Software* layer between **hardware** and **applications**



- The OS is "all the code that you didn't have to write" to implement your application

An example comparing life with/without OS

Life with an OS

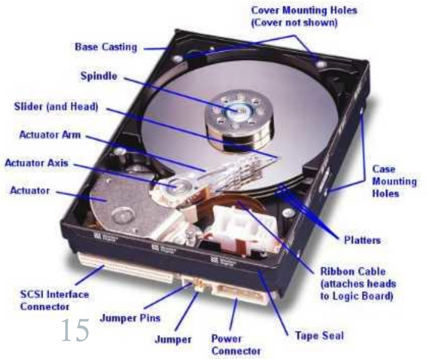
```
file = open (“test.txt”,
    O_WRONLY);

write (file, “test”, 4);

close (file);
```

Life without an OS

- Blocks, platter, track, and sector
- Where is this file on disk? Which platter, track, and sectors?
- Code needs to change on a different system



OS and hardware

- The OS abstracts/controls/mediates access to hardware resources (what resources?)
  - Computation (CPUs)
  - Volatile storage (memory) and persistent storage (disk, etc.)
  - Communication (network, modem, etc.)
  - Input/output devices (keyboard, display, printer, etc.)

Benefits to Applications

- Simpler
  - no tweaking device registers
- Device independent
  - all disks look the same
- Portable
  - same program runs on Windows95/98/ME/NT/2000/XP/Vista/Windows 7/Windows 8
- Worry less about interference from other applications

What does an OS do?

- Resources
  - Allocation
  - Protection
  - Reclamation
  - Virtualization

What does an OS do?

- Resources
  - Allocation
  - Protection
  - Reclamation
  - Virtualization

- Finite resources
- Competing demands

- Examples:
  - CPU
  - Memory
  - Disk
  - Network

Government:
Limited budget,
Land,
Natural resources

What does an OS do?

- Resources
  - Allocation
  - Protection
  - Reclamation
  - Virtualization

- You can't hurt me, I can't hurt you.
- Some degrees of safety and security

Government:

Law and order

What does an OS do?

- Resources
  - Allocation
  - Protection
  - Reclamation
  - Virtualization

- The OS gives,
  The OS takes away

- Some times involuntarily

Government:
Income Tax

What does an OS do?

- Resources
  - Allocation
  - Protection
  - Reclamation
  - Virtualization

- Illusion of infinite, private resources
  - Memory vs. disk
  - Time-shared CPU

Government:
Social welfare and insurance

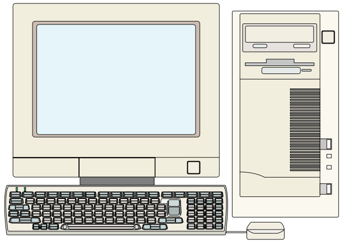
Why you want to learn OS?

- Foundation to other software
  - Databases, Browsers, Computational software, … …
- OS is one of the hardest software piece to write & debug
  - Directly talks to hardware (very ugly interfaces)
  - Abstract into clean interfaces
  - They are BIG
  - Lines of code:
    - Windows Vista (2006): 50M (XP + 10M) million lines of code
    - Linux 3.6: 15.9 M
    - Android 4.0: > 1M

Why you want to learn OS?

- Many OS concepts (e.g., protection, resource management) is needed in other places
  - E.g., browser
- OS is used everywhere
  - Your car is running on Linux/Windows

Computing Devices Everywhere



Computing Devices Everywhere

- Operating Systems drive the inner workings of virtually every computer in the world today
- PCs, servers, iPods, cell phones, missile guidance systems, etc. all have an OS that dictate how they operate.
- The OS manages many aspects of how programs run, and how they interact with hardware and the outside world.

Before the next class

- Browse the course web
- Start thinking about partners for project groups
- Read chapter 1
- Send me messages through the course web page if you have any questions
- Let the fun begin!