Seongwoo Choi
CMPS 111
Winter 2017
Note 1

Week 1
The First Lecture
Note: Professor Long was not teaching this week. Professor Sobn came into the class as a substitute professor.

Introduction
The operating system is very hard to create for a company or very hard to learn and create for us when we create it because it needs to deal with multiple components.

Topics:
These topics are for this quarter. We will learn more in detail on each of these topics later on.

- Introduction, concepts, review & history
- Processes
    - Synchronization
    - Scheduling
    - Deadlocks
- Memory management, address translation, and virtual memory
- Operating system management of I/O.
- File systems
- Virtualization
- Multi-core systems
- Security & Protection

Note: Every topic will be an assignment.

Programming projects
We will concentrate mostly on these:

- Modify FreeBSD
    - Runs on x86 hardware
    - Virtual machine software runs on Mac OS, Windows OS, Linux
    - Tool set runs on freeBSD.

What is OS?
An Operating system (OS) is a computer program that provides use abstraction of hardware to applications. It interacts with multiple hardware and also it acts as an intermediary between computer and users. Also, an OS provides the interface to the user across different types of hardware. It is a resource manager.

- Presents user with a virtual machine, easier to use.

The Second Lecture

Note: *Professor Long was not teaching this week. Professor Sobn came into the class as a substitute professor.*

Spooling

- Is a specialized form of multi-programming for the purpose of copying data between different devices. Like drag and drop on Windows

Multi-core CPUs

There are two different types of cache in a CPU.

- Shared cache
- Structured cache

Storage Pyramid

- Goal: really large memory with very low latency.
    - Latencies are smaller at the top of the hierarchy
    - Capacities are larger at the bottom of the hierarchy
- Solution: move data between levels to create illusion

Disk drive structure.

In this structure, data are stored on surfaces.

- In total, there are up to two surfaces per platter.
- There can have one or more platters per disk.

Data in concentric tracks.
Tracks broken into sectors.

Flash memory structure

Flash is divided into erase blocks.
Flash is read and written in pages. (Typically 64-256 pages.)
Flash Translation Layer (FTL) manages the device.

Memory

Single base/limit pair: set for each process.
Two base/limit registers: one for program, one for data.

Anatomy of a device request

Left: sequence as seen by hardware.

- Request sent to controller, then to disk.
- Disk responds, signals disk controller which tells interrupt controller.
- Interrupt controller notifies CPU.

Right: interrupt handling (software point of view)

Processes

- Process: program in execution.
    - Address space (memory) the program can use.
    - State (registers, including program counter & stack pointer)

- OS keeps track of all processes in a process table.
- Processes can create other processes.

Inside a (UNIX) process.

- Processes have three segments
    - Text: program code
    - Data: program data
        - Statically declared variables
- Stack
    - Automatic variables
- Address space growth.

Deadlock

Two things:

1. Processes.
2. Resources

Hierarchical file systems:

Interprocess communication

- Processes want to exchange information with each other.

System calls

- OS runs in privileged mode.
- Programs want the OS to perform a service
- Access a file
- Create a process
- Others
- Accomplished by system call.
- Some operations are permitted only in privileged mode.
- User programs run in user mode and can't do the operations.

How system calls work.

- User program enters supervisor mode
    - Must enter via well-defined entry point.
- Program passes relevant information to OS.
- OS performs the service if

Making a System call

- System call

- Program pushes arguments, calls library.
- Library sets up trap, calls OS.
- OS handles system call.
- Control returns to library.
- Library returns to user program.

Operating system structure.

The Operating System is composed of lots of pieces.

- Memory management
- Process management
- Device drivers
- File system
- How do the pieces of the operating system.

Monolithic OS structure

- All of the OS is one big “program”
- Sometimes modular (as with Linux and freeBSD)
    - Extra pieces can be dynamically added.
    - Extra pieces become part of the whole.
- Easy to write, but harder to get right.

Layered OS.

CPU supports “layers” of privilege.

- Intel supports 4.
- User code at outermost layer.

User privilege layers to enforce

Microkernels (client-server)

Processes (clients and OS servers) don’t share memory.

- Communication via message-passing.
- Separation reduces risk of “byzantine” failures

Examples include

- Mach (used by MacOS X)
- MINIX

Prof. Sobn mentioned that running Linux on Macintosh