CS 111: Operating System Principles
Lecture 1

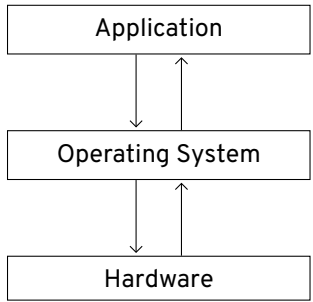
Overview
1.0.1

Jon Eyolfson
March 30, 2021

“All problems in computer science can be solved by another level of indirection”

- David Wheeler

An Operating System Sits between Applications and Hardware



The primary role of an operating system is to manage and coordinate resources

Ubuntu and Android are Considered Different Operating Systems

Both use a Linux kernel, but they run different applications

There isn't a clear line, especially with "Linux"

For desktop applications, you'd draw the line at the Display System

"Linux" uses Wayland, and Android uses SurfaceFlinger

Operating Systems Allow Running More than One Application

Without an operating system, a CPU starts executing code at a fixed address

You could put your application here, but it would be the only one

You would have to handle specific hardware in your application

Instead, we start executing an operating system at boot

Our First Abstraction is a Process

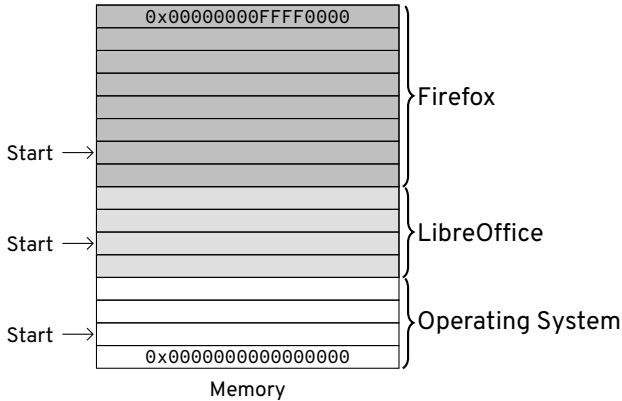
Each process contains its own set of registers, including the program counter

When starting a process, it specifies where the CPU should start executing

The operating systems has to:

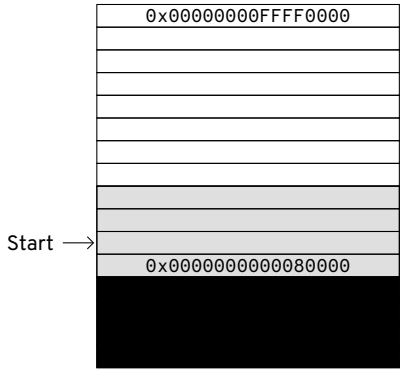
- Keeps track of registers for each process
- Switch between different processes
- Decide when to switch between processes

We Could Put Applications in Different Parts of Memory

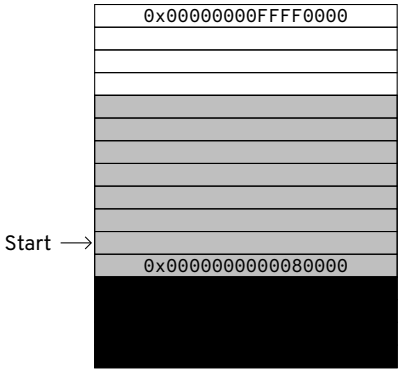


This isn't very flexible

Virtualization Fools Something into Thinking it Has All Resources



"LibreOffice Memory"



"Firefox Memory"

Virtual Memory Abstracts Away Physical Memory

Each process believes it has access to all the memory

Different processes can have the same starting address

The operating system has to:

- Map virtual memory access to physical memory
- Keep track of memory usage (allocate and deallocate)
- Handle out-of-memory scenarios

Virtualization is a Powerful Concept

Applies to both processes and virtual memory

We can extend this to an entire machine

A single physical machine can run multiple operating systems at once

Concurrency is Multiple Things Happening at the Same Time

We want multiple applications running at once

We want applications to do multiple things at once

We don't want applications isolated

We want applications and libraries to communicate

Concurrency is Necessary for Operating Systems

Running one application at a time isn't a good experience

Completely isolated applications aren't useful

The simplest applications still communicate with the terminal

The operating system has to:

- Allow multiple executions at once, safely
- Manage abstractions for different kinds of inter-process communication (IPC)
- Provide permission checking and access control

Finally, We Need Persistence for a Basic Operating System

We want to be able to access data between boots

A file system specifies how to organize data on a storage medium

The operating system has to:

- Store and retrieve data
- Ensure integrity of data

File Descriptors Abstract Both Communication and Persistence

A file descriptor is just a number identifier (per process) that you can:

- Read bytes from
- Write bytes to

The operating system can direct the bytes to whatever it represents

You could imagine it representing a file, or one side of communication

Security is Another Consideration

We want our computers to only do what we tell them to

The operating system has to:

- Encrypt of sensitive data
- Prevent bypassing access control
- Only execute applications the user wants

Most Kernel Code is Device Drivers

Device drivers implement the abstractions we'll learn to the physical hardware

It involves reading manufacturer specifications, and finding bugs

Sometimes there's inconsistencies between documentation and reality

An Actual Comment Linux Source (`arch/x86/kernel/apm_32.c`)

```
/*
 *  Check for clue free BIOS implementations who use
 *  the following QA technique
 *
 *      [ Write BIOS Code ]<------
 *               |                ^
 *      < Does it Compile >----N--
 *               |Y               ^
 *      < Does it Boot Win98 >-N--
 *               |Y
 *           [Ship It]
 *
 *      Phoenix A04  08/24/2000 is known bad (Dell Inspiron 5000e)
 *      Phoenix A07  09/29/2000 is known good (Dell Inspiron 5000)
 */
```

Believe It or Not, This Is “Hello World”

```
0x7F 0x45 0x4C 0x46 0x02 0x01 0x01 0x03 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x02 0x00 0x3E 0x00 0x01 0x00 0x00 0x00 0x78 0x00 0x01 0x00 0x00 0x00 0x00 0x00
0x40 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x00 0x00 0x00 0x40 0x00 0x38 0x00 0x01 0x00 0x40 0x00 0x00 0x00 0x00 0x00
0x01 0x00 0x00 0x00 0x05 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x00 0x01 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x01 0x00 0x00 0x00 0x00 0x00
0xB2 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0xB2 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x01 0x00 0x00 0x00 0x00 0x00 0x00 0x48 0xC7 0xC0 0x01 0x00 0x00 0x00 0x48
0xC7 0xC7 0x01 0x00 0x00 0x00 0x48 0xC7 0xC6 0xA6 0x00 0x01 0x00 0x48 0xC7 0xC2
0x0C 0x00 0x00 0x00 0x0F 0x05 0x48 0xC7 0xC0 0xE7 0x00 0x00 0x00 0x48 0xC7 0xC7
0x00 0x00 0x00 0x00 0x0F 0x05 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x77 0x6F 0x72 0x6C
0x64 0x0A
```

There are 4 Major Concepts in This Course

You'll learn how the following applies to operating systems:

- Virtualization
- Concurrency
- Persistence
- Security