What Problem Do Operating Systems Solve?
Why Should You Learn About Them?

Michael E. Locasto

Department of Computer Science
UofC
CPSC 457

September 10, 2014
Review of Previous Homeworks (10 minutes)
Why Study Operating Systems? (15 minutes)
→ the sermon
→ what problem(s) do(es) an OS solve?
Skills I Want You to Learn (5 minutes)
Links From Wiki (10 minutes)
→ http://lxr.cpsc.ucalgary.ca/lxr/
What is a program? (10)
→ considerations
→ demo
What Problem Do Operating Systems Solve?
Approaches to Teaching OS

(a) Concepts only
(b) Dinosaur book: Java code
(c) Toy OS, e.g., Nachos
(d) Doug Comer: OS from scratch on a 802.11 AP hardware
(e) Minix: experimental research OS (see www.minix3.org)
(f) Nieh: Linux kernel development / Android development
(g) UofC: Linux or Android on Raspberry Pi (coming soon!)
I’ve heard both students and professors here and abroad say that OS should no longer be a core part of the Computer Science curriculum. Let’s examine that claim.

Students are often launched into an OS course by being told “what” an operating system is, but are seldom given an opportunity to understand “why” this topic is either interesting or relevant.
There is always a temptation to label CS courses according to the technology presented in them. “That’s the Java course.” “This is the Python course.” “There is the SPARC course.” “This is the OS course.” To me, “operating system” as a term is about the software that *operates* some interesting system (possibly hardware).

Although this course focuses on Linux, it is not about Linux per se; we simply demonstrate abstract OS concepts relative to a mature, real-world, and complex piece of software.
Measurement
Concurreny
Resource Management

All of these occur in complex software and hardware systems; OS is but one expression of them. We could probably teach “OS” by looking at systems like Firefox, qmail, OpenSSH, a windowing system, or a RDBMS.

---

1 This is not just about “speed”, but rather the ability to intercept and inspect behavior, state, and system properties
Every part of the motors was an embodied answer to “Why?” and “What for?” – like the steps of a life-course chosen by the sort of mind she worshipped. The motors were a moral code cast in steel. (p245-246)

Operating systems (and large, well designed software systems in general) are like great works of art and engineering combined — filled with purpose.
Main Problem

What main problem do operating systems solve?
→ Manage time and space multiplexed resources
→ Mediate hardware access for application programs
→ Load programs and manage execution
Not only do I want you to come out of this course with an understanding of operating systems principles and the main algorithms involved in resource management, I want you to absorb and nurture some practical skills:

- using a command line
- accessing software documentation (i.e., the man pages)
- experience using C for both user level and kernel programming (and knowing the difference in available environments)
- understanding and practicing how to use assembly code to invoke system calls
- use of a version control systems (i.e., svn)
- how to write, compile, and load loadable kernel modules (LKM)s
- gaining familiarity and comfort navigating a very large code base and;
- comfort using the LXR tool to navigate such a large code base
- the process of configuring and building a large piece of software, including the expense of compiling ”real” large software
Understand the OS environment as a confluence of several disciplines:

- computer architecture
- "real" software engineering of a complex code artifact
- computer science concepts for managing concurrency and sets of resources

This environment allows you to:

- study how to design and manage hardware to support complex applications
- see the kernel as a case study of complex applications
- put into practice some of the software programming and problem solving skills you’ve learned so far in your undergraduate career
(a) source code?
(b) assembly code?
(c) machine code / binary?
(d) a process?

goto demo (also a demo of the command line)
Next week, we will begin to examine the question of what a process is (i.e., its properties and characteristics) and how it is created.

Also, guest lecture next Wednesday (Prof. Carey Williamson)