Introduction

Nima Honarmand
What is an Operating System? (1)

• All of the stuff between you/your application and the hardware
  • Kernel
  • Device Drivers
  • API libraries
  • UI

• Our focus is mostly on the kernel and some advanced API
What is an Operating System? (2)

• What are the main tasks of an OS?
  • Abstract the hardware for convenience and portability
  • Virtualize the hardware to share it among multiple applications
  • Multiplex the virtualized resources over physical resources
  • Provide services to applications and users
    • Process isolation
    • Inter-process communication
    • UI
    • ...

Why Operating Systems?

• Primary Goal: Demystify how computers work
  • Lots of abstractions and heuristics between your application and the hardware
  • A good computer scientist should understand what happens inside the system when one types a command

• Secondary: Learn how to write robust programs
  • OSes like Linux have many users and work on a wide range of hardware
  • Deal with subtle issues: concurrency, consistency, etc.
About This Course

• This course is being revised
  • After recent changes in CSE 320 and making it a prerequisite

• Bear with me as we try to finalize the course syllabus

• Some basic OS concepts and API already taught in CSE 320
  • We will review those quickly
  • We will cover more advanced API here

• More importantly, we will talk about internal design and implementation issues of Oses
  • Not covered in CSE 320
Course Format :: Lectures (1)

- Basic OS ideas: abstractions, interfaces and algorithms on core issues
  - Memory
  - CPU
  - Multi-threading and synchronization
  - Storage — Disks (HDDs and SSDs) and File systems
  - Networking
  - Device programming
  - Inter-process communication and isolation issues

- Supplement background on hardware programming
Course Format :: Lectures (2)

• Several more recent topics (time permitting)
  • OS Security
  • Virtual machines
  • Advanced file systems
  • OS in data centers and cloud
  • Embedded OS issues
  • High performance networking
  • etc.
Course Format :: Lectures (3)

• Discuss and supplement reading material

• An important chance to clarify issues
  - Questions are encouraged!

• I expect you to arrive prepared to answer and ask questions about the reading material

• Everything in lectures may appear on the exams, even if not in the book
Course Format :: Labs (1)

• Learn by doing

• This course is **coding intensive**
  • You should know C, or be prepared to remediate quickly
  • You will learn basic, inline x86 assembly
  • You must learn to **learn on your own**/with lab partner

• You will make substantial modifications to xv6, a simple x86 Unix variant
  • Code is written to be easy to understand, but lacks many modern OS features
  • Challenging work, but a very marketable skill
Course Format :: Labs (2)

• We may have one or two advanced user-mode programming labs to use advanced OS API not covered in CSE 320
  • Unlike xv6 labs that will be kernel-mode programming

• One way or the other, there will be five labs in total
  • Combination of xv6 and user-mode programming

• The exact combination will be determined as we go
Course Format :: Labs (3)

• Each student will have a Linux virtual machine on a departmental cluster
  • Comes with basic development tool chain that you need for the course (GCC, GDB, Qemu, etc.)
  • You have sudo access and can install new software as you need

• You can also use your own laptops but the tech staff won’t support them

• See https://compas.cs.stonybrook.edu/~nhonarmand/courses/fa17/cse306/labs.html for complete details
Lab Late Hours

• Each lab team gets 72 late hours
  • List how many you use in slack.txt
  • Each hour after these are gone costs 2% on the assignment

• It is your responsibility to use these to manage:
  • Holidays, weddings, research deadlines, conference travel, Buffy marathons, release of the next Zelda game, etc.

• 3 Exceptions: illness (need doctor’s note), death in immediate family, accommodation for disability
Lab Teams

• Can work alone or as a pair

• Choose your own partners

• Remains the same for the entire course
  • Changes only with instructor permission
Readings & Books

• **Required Readings**
  • Assigned from the OSTEP book + maybe some papers
  • **Should be completed before the lecture**
  • Required reading material may appear on the exams, even if not discussed in lecture

• **Highly recommended (optional) books**
  • K&R for C programming
  • Understanding the Linux Kernel (3rd edition) for Linux internals
  • Advanced Programming in the UNIX Environment (3rd edition), the UNIX bible

• Many other references on the website
Other Course Information

• TA: TBD

• Course website:
  • compas.cs.stonybrook.edu/~nhonarmand/courses/fa17/cse306
  • Syllabus, schedule, homework, etc. posted here

• Course newsgroup
  • Blackboard → Discussions → “General Discussions” Forum
  • Main venue for all class-related discussions
  • Sign up ASAP to avoid missing anything
  • Goal: Everyone can learn from general questions
  • Do not post code or other solutions here
Prerequisites

• CSE 219 (CS III) or CSE 260 (CS B, Honors)

• CSE 320 (Systems Fundamentals II) or ESE 380 (Embedded Microprocessor Design I)

• The background courses are necessary

• In some cases, industry experience is ok
  • In-class quiz, due before you leave
  • If you can’t answer 50% of these questions you are not prepared

• C programming

• Basic Unix command-line proficiency
C Programming

• You should have learned C in the prerequisite courses

• If you are not sure, you should read “The C Programming Language” by Kernighan and Ritchie (K&R) cover to cover this week
  • And complete all exercises in the book

• If you can do this in the next week or so, you will be prepared to complete this course on schedule
Other administrative notes

• Read syllabus completely

• Subscribe to the discussion forum on Blackboard

• The exams cover lectures, labs, assigned readings and blackboard discussions

• Lab VMs aren’t ready yet

• Back up your lab work in a private repo
  • Department provides git repos to let you backup your work
    • Send an email to rt@cs.stonybrook.edu to have yours activated
    • Or use BitBucket or Github

• Do not make your repos publicly available
# Grading

<table>
<thead>
<tr>
<th>What?</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Quiz</td>
<td>0</td>
</tr>
<tr>
<td>Labs</td>
<td>60</td>
</tr>
<tr>
<td>Midterm exam</td>
<td>20</td>
</tr>
<tr>
<td>Final exam</td>
<td>20</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>100</strong></td>
</tr>
</tbody>
</table>

- Guaranteed grades: [A, A-, B+, ..., D, F] = [85, 80, 75, ..., 45, <45]
  - I may use a curve on top of this (but there is no guarantee)
- Midterm grade will be the max(midterm, final)
- Grades solely determined by your performance in the course
  - Not whether they are needed for graduation, qualifiers, etc.
Academic Integrity

• We take cheating very seriously. It can end your career.

• Share ideas but not code
  • Acknowledge students that give you good ideas

• In a gray area, it is your job to stay on right side of line

• Never show your code to anyone except course staff
• Never look at anyone else’s code (including other universities)
  • Do not debug each other’s code

• Fully read the Academic Integrity text on the website
Questions?

Remember:

• Hand in the survey

• VMs and Lab 1 coming out soon

• Reading assigned for next class (Thursday)
  • No class on Tuesday (Labor Day Observance)
Survey Quiz