ECE 353: Systems Software
Lecture 12

First Review

1.1.0

Jon Eyolfson

February 2, 2023

There are 3 Major Concepts in This Course

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

- Virtualization
- Concurrency
- Persistence

Kernel Interfaces Operate Between CPU Mode Boundaries

The lessons from the lecture:

- Code running in kernel mode is part of your kernel
- System calls are the interface between user and kernel mode
  - Every program must use this interface!
- File format and instructions to define a simple “Hello world” (in 168 bytes)
  - Difference between API and ABI
  - How to explore system calls
- Different kernel architectures shift how much code runs in kernel mode

Unix Systems Clone Processes with a Parent/Child Relationship

- You can only create new processes with `fork`
- After a `fork` both processes are exactly the same
    - except for the value of `pid` (the child is always 0)
- The scheduler decides when to run either process

You’re Responsible for Managing Processes

The operating system maintains a strict parent/child relationship

You should be able to identify (and prevent) the following:

- Zombie processes
- Orphan processes

We Explored Basic IPC in an Operating System

Some basic IPC includes:

- `read` and `write` through file descriptors (could be a regular file)
- Redirecting file descriptors for communcation
- Signals

Signals are like interrupts for user processes

The kernel has to handle all 3 kinds of “interrupts”

Scheduling Involves Trade-Offs

We looked at few different algorithms:

- First Come First Served (FCFS) is the most basic scheduling algorithm
- Shortest Job First (SJF) is a tweak that reduces waiting time
- Shortest Remaining Time First (SRTF) uses SJF ideas with preemptions
- SRTF optimizes lowest waiting time (or turnaround time)
- Round-robin (RR) optimizes fairness and response time

Scheduling Gets Even More Complex

There are more solutions, and more issues:

- Introducing priority also introduces priority inversion
- Some processes need good interactivity, others not so much
- Multiprocessors may require per-CPU queues
- Real-time requires predictability
- Completely Fair Scheduler (CFS) tries to model the ideal fairness

Operating Systems Provide the Foundation for Libraries

We learned:

- Dynamic libraries and a comparison to static libraries
  - How to manipulate the dynamic loader
- Example of issues from ABI changes without API changes

Example Midterm

`https://eyolfson.com/media/courses/ucla/cs111/21fall/midterm.pdf`

This is the style of midterm I typically write

We should be able to do every thing except virtual memory (next week!)