Dept Number	CS 407	Course Title	Advanced Linux/UNIX Programming
Semester Hours	3	Course Coordinator	Norman Carver
Catalog Description	This course builds on the knowledge gained in CS 306, to prepare students to do advanced development on Linux/UNIX platforms. The topics studied are critical for achieving high performance in large-scale, high-load networked software systems. These topics include development techniques such as profiling, concurrent programming and synchronization, network programming for high-load servers, advanced I/O alternatives, and IPC such as shared memory. The course will involve the study of code from Open Source projects like Apache and Nginx. The focus will be on the C language, but other languages will also be considered. Students must complete a significant network software project.		

Textbooks

The Linux Programming Interface. Kerrisk, Michael. No Starch Press, 2010.

References

- *Advanced Programming in the UNIX Environment*, 3rd ed. Stevens, Richard and Rago, Steven. Addison-Wesley, 2013.
- *Advanced UNIX Programming*, 2nd Ed. Rochkind, Marc. Addison Wesley, 2004.

Course Learning Outcomes

- Advancing students C development skills.
- Improving students' knowledge of concurrent programming.
- Improving students' knowledge of network and distributed programming.
- Familiarizing students with advanced Linux/UNIX system calls.
- Familiarizing students with performance and security trade-offs in software.
- Preparing students for advanced software engineering jobs (e.g., Site Reliability Engineering at Google).

Assessment of the Contribution to Student Outcomes

Outcome →	1	2	3	4	5	6	7	8	9	10
Assessed →	X	X	X	X	X					

Prerequisites by Topic

CS 306 & 335 with grades of C or better, or grad standing with C language & Linux system programming experience.

Major Topics Covered in the Course

1. Advanced C Development
   - Compilers: GCC vs. Clang
   - C vs. C++ vs. Objective C
   - Compiler options (optimization, etc.)
   - Code disassembly and analysis
   - Debugging from core files
   - Performance profiling
   - Library creation and use
2. Concurrent Programming
   - Issues in concurrent programming
   - Process vs. threads comparison
   - Pthreads calls and usage
   - Thread synchronization: mutexes, condition variables
   - Process synchronization: semaphores, signals
   - Thread/process pools
   - Thread-safe and async-signal-safe functions
   - Event-based (event-driven) programming
3. Signals
   - Signal characteristics in detail
   - Signal usage patterns
   - Writing proper signal handlers
   - Async-signal-safe functions
   - Real-time signals
   - Signals vs. file descriptors (e.g., signalfd())
4. Advanced Network Programming
   - TCP vs. UDP servers and clients
   - Alternative server models
   - The SCTP protocol
   - UNIX sockets
   - Raw sockets
   - Distributed programming and RPC
5. Advanced I/O
   - Non-blocking I/O
   - Scatter/gather I/O
   - Multiplexed/interleaved I/O (poll() and select ())
   - Epoll API (Linux-specific) and UNIX alternatives
   - Signal-based I/O
   - Async I/O (AIO)
   - Sendfile () and splice (), and equivalents
   - Issues in handling large numbers of devices/clients
   - Understanding kernel internals

Major Topics Covered in the Course

6. Advanced IPC
   - Message queues
   - Shared memory
   - Memory mapped files
   - Understanding kernel internals
7. Devices
   - Terminals and terminal I/O
   - Pseudo terminals and pty
   - Drivers
8. Writing Secure Programs
   - Security considerations in C
   - Program privileges
   - Linux capabilities and UNIX alternatives