- Kernel / Protection
    - Interpreting vs. direct execution on hardware
    - Kernel vs User Space
        - Privilege bit; privileged instructions
        - Memory protection
        - Timer
    - Avoiding boundary crossings
        - Big operations
        - Caching
- Virtual Address Space
    - Kernel allocated space
    - Address translation to physical
    - Differences between kalloc and malloc
- Executing Kernel Code
    - Synchronous vs. asynchronous transfer
    - Interrupts
        - Timer
    - Traps
        - System Call Process
        - Motivation for single access point.
    - Exceptions
    - Distinguishing different scenarios
    - System call process
- Processes
    - What is a process?
        - Process vs. thread
    - CreateProcess() vs. fork()
    - fork() vs. exec()
    - wait()
    - Orphans / zombies
    - Context switching
        - Pre-emptive (timer) vs. non-preemptive (yield(), sleep())
    - Process States and Transitions
    - Scheduling in xk
- File Descriptors
    - Open
    - Close
    - Dup
    - Read/Write
    - Stat

- Synchronization
    - Process/thread wait()
    - Spinning vs. Sleeping
        - Spinlocks
            - Disabling interrupts - when does it work?
        - Atomic Operations
            - Test-and-set
            - Compare-and-swap
        - Basic idea of optimistic / Non-blocking sync
        - Sleeplocks
            - Cost of changing to sleep state
        - What situations would you use each?
    - Condition Variables
        - What are the xk equivalents to cond_wait, cond_signal, and cond_broadcast?
        - How are these calls related?
- Inter Process Communication (IPC) / Pipes
    - How are pipes different than files?
    - How are they created/used?
    - What happens when you close a pipe?
- Inter Process Communication (IPC) / Signal Handling
    - What is it?
        - Example: Ctrl-C on a user process, how does that work?