Chapter 4: The Process

- The process is one of the most fundamental abstractions provided to users.
- A process is a running program
- The OS creates the illusion of CPU's by virtualizing the CPU.
    - By running one process, stopping it, then running another.
    - Time sharing allows the CPU to run multiple processes at once.
        - Possibly at the expense of performance.
- Mechanisms are low level-methods or protocols that implement a needed piece of functionality.
- Policies are algorithms for making some kind of decision within the OS.
    - A scheduling policy determines when things are run.

4.1 A Process

- A process is the abstraction provided by the OS of a running program.
    - At any given time a process can be summarized by taking an inventory of the different pieces of the system it accesses or affects during its execution.
- Understanding a process requires understanding its machine state.
    - What a program can read or update when it is running.
    - What parts of a machine are importatnt to execution.
- Memory is important.
    - Instructions reside in memory.
    - The data for the process to read and write is in memory.
    - Address space is the memory a process can address.
- Registers are important

    - Many instructions explicitely read or update registers.
    - Special registers that are important to a process:
        - Instruction pointer.
            - Tells which instruction the process was executing.
        - Stack pointer.
            - used to manage the stack.

4.2 Process API

- What must be included in any interface of an operating system.
- The following API's are available in some form on any modern operating system.
    - Create: A method to create a new process.
    - Destory: A method to destroy processes.
    - Wait: A waiting interface
    - Miscellaneous COntrol: Controls other than the above.
    - Status: Get status information about a process.

4.3 Process Creation

- How programs are transformed into processes.
    - Load the code and any static data into memory into the address space of the process.
        - Read bytes of the program from disk and move to memory.
        - In early OS the loading process was done all at once, also known as eagerly.
        - Modern OS load what's needed when it's needed, also called lazily.
    - Allocate memory for the programs stack.
    - Allocate memory for the program heap.

    - In UNIX each process has three open file descriptors:
        - stdin, stdout, stderr.

4.4 Process States

- A process can have different states at any given time.
    - Running: the process is running on the CPU.
    - Ready: The process is ready to be executed but not currently being executed.
    - Blocked: Waiting on some external event to ooccur.
- The states can move in the following ways:
    - Running to blocked.
    - Blocked to Ready.
    - Running to Ready or Ready to Running.
        - Descheduled: moving from Running to Ready.
        - Scheduled: moving from Ready to Running.
- The OS scheduler makes the decisions on when a process is scheduled or descheduled.

Data Structures:

- The OS is a program just like any other program.
- The OS most likely keeps a process list of ready processes and keep track of which processes are running.
- Blocked processes must also be kept track of.
- Register Context:
    - holds a all of the contents of the registers of a stopped process.
    - This will be saved to a memory location.