Lecture notes for Multics retrospective

- Multics overview
  - Time-sharing OS 1964-2000
  - MIT w/ GE/Honeywell & Bell Labs
  - Extremely influential to CS
    - Hierarchical file system, arbitrary length filenames, symlinks
    - ACLs on files
    - Dynamic linking (even to other executables...)
    - Hot swapping of CPUs, RAM, disks
    - Engineered from the start for security
    - Ring protection
    - Per-process per-ring stacks in kernel
    - 36 bit words (in 1964!)
  - Bell Labs left in 1969, developed UNICS (later UNIX)
  - [Compare to DOS of the 1980s and early 1990s... not even a concept of a user or of protection]
    - So what happened?
      - [Shift from time-sharing mainframes to single-user systems; people believed single-user systems required no security]

- Multics required all customers to design around its security
  - All apps must work with Multics' controls
  - [Compare to Windows & Vista user accounts control]

- PL/I [pee el one / programming language one]
  - Possible to program buffer overflow, but takes work
  - [Compare to C: possible to program non-overflow, but takes work]
  - Design choices: the easy thing to do should be the secure thing

- Backdoors / malicious developers
  - Testing is poor way to discover backdoors... unlikely to test critical values
  - Experimental backdoors discovered only with manual code audit
  - Backdoors put in programs since (sendmail debug mode, Excel flight simulator)

- Boot sector viruses
  - Virtual machine based rootkit

- Ring protections
  - Multics had 9 rings
  - x86 processors have 4
  - 0: hypervisor, 1: kernel, 2: unused, 3: apps

- Compiler backdoor inserter
  - Malicious compiler on system (as an executable binary)
  - Inserts a backdoor into every program it compiles
    - Source-code review of apps will **not** find the backdoor
    - Source-code review of compiler will not find the backdoor inserter
  - Recompile compiler?
    - But recompile with the malicious compiler
    - Detects own recompilation, inserts the backdoor inserter into output (or simply copies itself over to the final output)

- "Software without pedigree"
  - Military use of COTS
  - E-voting systems