Outline for October 23, 2014

Reading: *text*, §22.4–22.5, 22.7, 23.3–23.4

1. Types of malicious logic
   a. Computer worm
   b. Bacterium, rabbit
   c. Logic bomb
2. Ideal: program to detect malicious logic
   a. Can be shown: not possible to be precise in most general case
   b. Can detect all such programs if willing to accept false positives
   c. Can constrain case enough to locate specific malicious logic
3. Some defenses
   a. Type checking (data vs. instructions)
   b. Limiting rights (sandboxing)
   c. Limiting sharing
   d. Preventing or detecting changes to files
   e. Prevent code from acting beyond specification (proof carrying code)
   f. Static signature checking
   g. Behavioral analysis
   h. Check statistical characteristics of programs
4. Vulnerability models
   a. PA model
   b. RISOS
   c. NRL
   d. Aslam
5. Example Flaws
   a. *fingerd* buffer overflow
   b. *xterm* race condition
6. RISOS
   a. Goal: Aid managers, others in understanding security issues in OSes, and work required to make them more secure
   b. Incomplete parameter validation—failing to check that a parameter used as an array index is in the range of the array;
   c. Inconsistent parameter validation—if a routine allowing shared access to files accepts blanks in a file name, but no other file manipulation routine (such as a routine to revoke shared access) will accept them;
   d. Implicit sharing of privileged/confidential data—sending information by modulating the load average of the system;
   e. Asynchronous validation/Inadequate serialization—checking a file for access permission and opening it non-atomically, thereby allowing another process to change the binding of the name to the data between the check and the open;
   f. Inadequate identification/authentication/authorization—running a system program identified only by name, and having a different program with the same name executed;
   g. Violable prohibition/limit—being able to manipulate data outside one's protection domain; and
   h. Exploitable logic error—preventing a program from opening a critical file, causing the program to execute an error routine that gives the user unauthorized rights.