Lecture notes for 5.4 OS design

[Comment about multi-factor authentication & phishing attacks. Two kinds attacks: data collection for later use & active MITM.]

Design principles

- Least privilege [for users & programs]
- Economy [keep trusted code small as possible, easier to analyze & test]
- Open design [security by obscurity does not work]
- Complete mediation [every access checked, attempts to bypass must be prevented]
- Default deny [vs. default allow]
- Ease of use [users avoid security that gets in their way]

Security features of common OSes [some things we saw in chap 4]

- Authentication
- Memory protection
- Access control to files, hardware, & general objects
  - Mandatory access control (SELinux)
  - Discretionary access control (standard file permissions)
- IPC
- Protection of data used by OS for other protection mechanisms (OS must protect itself)
- Starvation prevention

Security features of trusted OSes

- Object reuse protection
  - Disk blocks, memory frames reused
  - Process can allocate disk or memory, then look to see what's left behind
  - Trusted OS should zero out objects before reuse
  - Secure file deletion: overwrite with varying patterns of zeros & ones
  - Secure disk destruction: degaussing, physical destruction
- Complete mediation of accesses
- Trusted path from user to secure system
  - Prevents programs from spoofing interface of secure components
  - Prevents programs from tapping path (e.g. keyloggers)

- Audit log showing object accesses
  - Only useful if you /look/ at the log
- Intrusion detection
  - Detect unusual use of the system
  - Masquerade detection

Kernel design

- Security kernel enforces all security mechanisms
- Good isolation, small size for verifiability, keeps security code together
- Reference monitor controls access to objects (monitors all references to objects)
  - Tamperproof [impossible to break or disable]
  - Unbypassable [always invoked, complete mediation]
  - Analyzable [small enough to analyze & understand]
- Trusted computing base (mentioned previously)
  - All parts of OS needed for correct enforcement of security policy
  - Handles primitive I/O, clocks, interrupt handling, hardware, capabilities
- Create security kernel by identifying critical components of large system
- Create security kernel by design
- Monolithic kernel vs. microkernel

Virtualization

- Virtual memory provides memory isolation, logical process separation
- Virtual machine provides hardware isolation, logical OS separation