Announcements

- Reading Chapter 19
- MT#2 re-grade requests due by Tuesday
  - Must be submitted in writing to grades.cs.umd.edu
  - Provide paper copy of exam to me

Summary of Scores to Date

	P0	P1	P2	MT1	P3	P4	MT2	Overall
Count	50	52	51	52	51	49	49	52
Minimum	90	30	0	32	0	0	26	8.7
Maximum	100	100	100	93	100	100	98	49.8
Mean	99.4	96.15	87.25	63.19	65.2	51.84	67.41	35.93
Std Dev	2.18	14.16	22.37	13.88	30.11	30.8	18.32	8.55

Computer Threat Model

- must consider acceptable risks
  - value of item to be protected
  - $2,000 of computer time to steal 50 cents of data
    - this is a sufficient deter someone
    - **but** computers keep getting faster
- Basic Ideas:
  - confine access to only the highest level needed
    - run programs as root only if needed
    - don't give system access to all users

Authentication

- How does the computer know who is using it?
  - need to exchange some information to verify the user
  - types of information exchanged:
    - pins
      - numeric passwords
      - too short to be secure in most cases
    - passwords
      - a string of letters and numbers
      - often easy to guess
    - challenge/response pairs
      - user needs to be apply to apply a specific algorithm
      - often involve use of a calculator like device
      - can be combined with passwords
    - unique attributes of the person
      - i.e. signature, thumb print, DNA?
      - sometimes these features can change during life

Authentication (cont.)

- How does a user know what computer they are using?
- Need to have *mutual authentication*
  - computer presents some information that only it could contain
  - example: Windows <ctrl>-<alt>-<del> to login
    - user software can't trap that information
    - assumes that the kernel itself is secure
- telephone example:
  - never give banking/credit card info over the phone unless you placed the phone call
    - i.e. you use the telco namespace for authentication

Example (UNIX passwords)

- use a function that is hard to invert
  - "easy" to compute f(x) given x
  - hard to compute x given f(x)
  - the function used is a variation on the DES algorithm
    - changes selected items in the transformation matrix to prevent hardware attacks
  - store only f(x) in the filesystem
- to login:
  - user supplies a password x'
  - compute f(x') and compare to f(x)
- salt
  - add an extra two characters to x so that the same x will produce different values on different machines
- dictionary attach
  - if its to easy to compute f(x)
  - can "guess" many passwords and try them out

Types of Software Threats

- Trojan Horse
  - a program that looks like a normal program
  - for example a login program written by a user
  - UNIX example: never put "." early in your path
- Trap door
  - hole left by the programmers to let them into the system
  - "system" password set to a default value by the vendor
- Worms
  - programs that clone themselves and use resources
  - Internet worm:
    - exploited several bugs and "features" in UNIX
      - .rhosts files
      - bug in finger command (overwrite strings)
      - sendmail "debug" mode to run commands

Viruses

- Most common on systems with little security
  - easy to write to boot blocks, system software
  - never run untrusted software with special privileges
  - Don’t perform daily operations with root/system privileges
- Possible to write system independent viruses
  - MS Word virus
    - uses macros to call into the OS

Access Matrix

- Abstraction of protection for objects in a system.
  - Rows are domains (users or groups of users)
  - Columns are objects (files, printers, etc.)
  - Items are methods permitted by a domain on an objects
    - read, write, execute, print, delete, …
- Representing the Table
  - simple representation (dense matrix) is large
  - sparse representation possible: each non-zero in the matrix
  - observation: same column used frequently
    - represent groups of users with a name and just store that
  - create a default policy for some objects without a value
- Revocation of access
  - when are access rights checked?
  - selective revocation vs. global

Access Matrix

	F1	F2	F3	Laser Printer	
D1	read		execute		
D2			execute	print	
D3	read, write		execute		
D4			execute		
D5		delete			

- Rows represent users or groups of users
- Columns represent files, printers, etc.

Capabilities

- Un-forgeable Key to access something
- Implementation: a string
  - I.e. a long numeric sequence for a copier)
- Implementation: A protected memory region
    - tag memory (or procedures) with access rights
      - example - x86 call gate abstraction
    - permit rights amplification