Announcements

- Reading Chapter 19 (Security)
- Project #5 is due Today
- Project #6 will be out by Friday
Security

- **security vs. protection**
  - protection provides a mechanism to control access to resources
  - security also includes external features such as users

- **security requires precluding unauthorized**
  - access to data
  - modification of data
  - destruction of data

- **several major types of security**
  - physical: must protect access to resource itself
    - if you have physical access to a machine, you can break security.
  - users: if a user gives away access (or info) computer security if useless
  - software: OS and system software must provide protection
Who do you trust?

- It’s easy to get paranoid
- Do I trust a login prompt?
- Do I trust the OS that I got from the vendor?
- Do I trust the system staff?
  - should I encrypt all my files?
- Networking
  - do you trust the network provider?
  - do you trust the phone company?
- How do you bootstrap security?
  - always need one “out of band” transfer to get going
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, MD5 SHA1 algorithms
    - 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
  - need to store salt along with password

- dictionary attack
  - if it’s too easy to compute \( f(x) \)
  - can “guess” many passwords and try them out
  - salt makes this much harder
Types of Software Threats (Malware)

- **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