Agenda

- Lecture
  - Security
- Tutorial
  - TA Evaluations
  - Pick up quizzes
  - Take up quizzes
  - More security?

Who needs security?

- Amount of security needed is proportional to risk exposure
  - Risk Exposure = Probability × Effect
  - Two factors are not independent
- Probability increases when you have more vulnerabilities
  - Desktop computer with no connectivity vs. web application/ service
- Effect increases when the value of assets increase
  - Data, game credits, cash

Defending Yourself

- Step 1. Who are the attackers?
- Step 2. What do they want?
- Step 3. Setting up countermeasures

- Script kiddies
  - Root access for bragging rights
- Spam networks
  - Root access to enslave your computer
- Your own users
  - Gain advantage in your game
- Industrial and political espionage
  - Information or intelligence
- Link spammers

Root Access

- root user has complete control over computer
  - can install software, add/remove users
  - reformat hard drive

- Countermeasures
  - Don't allow login as root
    - sudo
  - Don't allow web server to run as root
    - Use a web or http user and group

Server-Level Countermeasures

- Red Hat is a better Linux distro for servers
  - Slower to add new packages, more sensible defaults
- Keep software up to date
  - But not on the bleeding edge
- Turn off services that you don't need
  - Reduces open ports

Protecting Ports

- Common attack is to try ports until an open one is found
  - Then move on to account attacks
- Move ssh from port 22
  - ~50 000 range
- Firewall
  - iptables and netfilter
  - tarpit
- Decrease response time
  - Configure to fail silently than return errors
  - Increase time-out interval
  - Keep connection alive longer

Authentication

- Users
  - "You can't come in without a password"
- Web sites
  - "How do I know you're a real web site"
  - Phishing attacks and social engineering
  - SSL certificates and robust browsers
- Third-party apps
  - Shared Secret multi-directional protocol

Attacks on Accounts and Passwords

- To gain access, need login and password
  - Remember disallowing root to login
- Most common attacks involve brute force guessing
  - Using dictionaries
- Next level of sophistication is table attacks
  - Compare the contents of tables rather than one at a time

Protecting Passwords

- Don't store as plain text
- Store in a format, so that data is still protected even if attacker has access to the password table
- Current best practice:
  - hash(password+salt)
  - Hash = SHA256
  - Salt is 32 bytes; can be reused
  - Note: mysql only hashes, does not salt

Protecting the Comm Channel

- Requires end-to-end authentication for both the user and the site
- Primary mechanism is SSL using a certificate from a signed authority
  - Otherwise warnings are generated by maximal security browser
- Certificates are specific to a domain name and ip address
  - 128 or 256 bits
  - More bits, more encryption
  - Signing authorities charge for service; charge more for guarantees (insurance)

Protecting Apache

- Keep software up to date
  - But not on the bleeding edge
- Turn off mod_rewrite on apache
  - Improves performance too
- On http, use POST, not GET for login/password submission
  - GET contents are stored in the access log

Protecting Service Requests

- Server accepts requests through URL (standard practice)
- Traffic with a client can be monitored and request format can be reverse engineered
- Need to ensure that server data/contents are not tampered with
  - Interception Attack
  - Replay Attack
  - Cross-Site Scripting
- Use Shared Secrets Protocol
  - Send important data + unimportant data
  - Send hash(important data + shared secret + timestamp)

Protecting Data

- Input to your application from an external source must be treated as “dirty data”
  - Insertion attacks
- Apply “washing” to make it clean
  - Consists of rejection or alteration
- Examples of checks
  - Length
  - Legal characters
  - Escaping characters
  - SQL injection