CS 155 Spring 2006

How things goes wrong

John Mitchell

Lecture 2 April 5

Announcements

- My office hours
  - Thursdays 2:30-3:30, Gates 476 (or Bytes Café?)
- Course discussion section
  - Friday 3:15-4:05pm in Gates B01 (live on E3)
  - Start Friday 4/14
- Final exam time
  - Tuesday June 13, 7-10 PM
- Other issues?

General concepts in this course

- Vulnerabilities
  - How hackers break into systems
    - Circumvent security mechanisms (e.g., dictionary attack)
    - Use code for purpose it was not intended (buffer overflow)
- Defensive programming
  - Build all software with security in mind
  - Make sure your video game is not a boot loader
- Security Mechanisms
  - Authentication, Access control, Network protocols, Rights management, System monitoring, …

This lecture: Security Problems

- Anatomy of an attack
  - What attackers want
  - Steps in standard break in
- Some ways we help them do it
  - Weak input checking
  - Buffer overflow
  - Inappropriate logging
  - Unintended functionality
  - Inappropriate privilege
  - Race conditions
  - Misconfigured systems
  - Lack of diversity

What attackers want

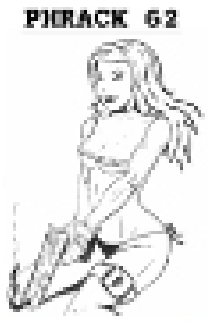
- Create havoc
  - Make the newspaper, tell their friends
- Embarrass or harass someone
  - Deface web pages
- Shut down systems
  - DoS eBay in last 59 minutes of auction
  - DoS sites of business rival or political enemy
- Steal information
  - Product activation codes for popular games
  - User name and password for bank site
  - Credit card or phone card numbers, identity theft
  - Steal business information or government secrets
  - Break copy protection mechanisms

Some hacker resources

- Web sites and archives (use Google to find more …)
  - *Phrack*, www.phrack.org
  - *The Hack FAQ*, www.nmrc.org/pub/faq/hackfaq/
  - *Piracy: The Art of Cracking*, www.textfiles.com/piracy/CRACKING/, including "How To Crack pretty Much Anything", by +ORC
- IMPORTANT NOTICE
  - We provide these links so you can see how hackers operate and learn to prevent attacks.
  - Do not use these attacks on anyone!!!

This course gives you information that can be used for good or evil. It is your ethical responsibility to use this information carefully and considerately. If you do not plan to do so, you are free to drop this class.

Hacker culture



Ranges from amusing to offensive ... probably not written by a 60-year-old in a business suit

Steps in a standard break-in

- Get your foot in the door
  - Steal a password file and run dictionary attack
  - Sniff passwords off the network, social engineering
  - Use input vulnerability in other network code
- Use partial access to gain root (admin) access
  - Break some mechanism on the system
- Set up some way to return
  - Install login program or web server with back door
- Cover your tracks
  - Disable intrusion detection, virus protection, tripwire program, system functions that show list of running programs, ...

Other kinds of attack ...

- Key loggers
  - Install software that reports stolen information
- DOS attacks
  - Use compromised machines to flood network

Black Hat Europe 2006
28 February - 3 March
the Netherlands
Briefings & Training

- Philippe Biondi, & Fabrice Desclaux
  Silver Needle in the Skype
  - This presentation will uncover some Skype secrets, hidden behind many levels of obfuscation, showing how bad security by obscurity can be. It will also describe many technics and tools used to go through obfuscation layers and speak Skype
- Cesar Cerrudo
  WLSI - Windows Local Shellcode Injection
  - A new technique to create 100% reliable local exploits for Windows operating systems, the technique uses a Windows operating systems design weaknesses that allow low privileged processes to insert data on almost any Windows processes no matter if they are running under higher privileges
- many more ...

http://www.blackhat.com/html/bh-media-archives/bh-archives-2006.html

Weak input checking

- General problem
  - Lots of programs have input
    - User input
    - Function calls from other modules
    - Configuration files
    - Network packets
    - Web form input
  - Many web site examples
    - Scripting languages with string input
  - Extensible systems also have serious problems
    - Modules designed assuming calls come from trusted code
    - Extend system so untrusted code can call trusted module

Example: PHP passthru

- Idea
  - PHP passthru(*string*) executes command
  - Pages can construct *string* from user input
  - Put ";" in user input to run your favorite command
    - Morris Internet worm did something similar using "|"
- Example
  - passthru("find . –print | xargs cat | grep $test");
- User input ; ls /

  Runs find . –print | xargs cat | grep ; ls /

2001

Example: Cold Fusion CFEXECUTE

- Example web site code

```
<CFSET #STRING#="c: "' & #form.text# & '"C:\inetput\wwroot\*">
<CFEXECUTE
        NAME = 'c:\winnt\system32\findstr.exe'
        ARGUMENTS=#STRING#
        OUTPUTFILE="c:\inetpub\wwwroot\output.txt"
        TIMEOUT="120">
</CFEXECUTE>
```

- Displayed web page

- User input

  x" c:\winnt\repair\sam ... " ...

  Executes findstr.exe ... c:\winnt\repair\sam ... ...

  possibly with admin privileges

See Hoglund and McGraw, *Exploiting Software* for more info

Unicode vulnerabilities

- Some web servers check string input
  - Disallow sequences such as ../ or \
  - But may not check unicode %c0%af for '/'
- IIS Example, used by Nimda worm

http://victim.com/scripts/../../winnt/system32/cmd.exe?<some command>

  - passes <some command> to cmd command
  - scripts directory of IIS has execute permissions
- Input checking would prevent that, but not this

http://victim.com/scripts/..%c0%af../%c0%afwinnt/system32/...

  - IIS first checks input, then expands unicode

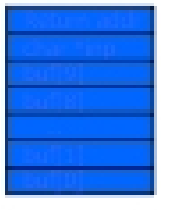
see www.sans.org/rr/threats/unicode.php

Buffer overflow

- Imagine simple password-checking code

```
passwd() { ...
 int funct(char *inp) {
    char buf[10];
    strcpy(buf,inp); }
 ...
  }
```

- Function storage allocated on run-time stack
  - First return address (4 B)
  - Then locations for input parameter
  - Then space for buffer (10 chars)
- What if strlen(inp) > 10 ?
  - Fill up buffer
  - Write over function parameter
  - Write over return address
  - "Return" will jump to location determined by input



(All fixed)

Some examples

- MSFT indexing service, an extension to IIS

```
telnet <victim> 80
GET /somefile.idq?<long buffer>
```

  - Telnet to port 80 and send http GET with buffer over 240 bytes
  - Attacker can take over server
  - Form of attack used by Code Red to propagate
- TFTP server in Cisco IOS
  - Use overflow vulnerability to take over server (long filename)
- MS Xbox
  - James Bond 007 game has a save game option
  - Code to restore game has buffer overflow vulnerability
  - Can boot linux or run other code using game as "boot loader"

Many, many more examples

(All fixed)

Inappropriate logging

- PDG soft: web transaction processing system
  - Creates logfile that is world-readable: /cgi_bin/PDG_cart/order.log
  - File contains mailing addresses, credit card numbers, ...
  - Can use (or could've use) Google to find sites that have this file
  - Bug discovered a few years ago
    - PDG issued patch:
      - changed protection domain of log file, encrypts log file
    - 1.5 years later, FBI reports: still lots of sites vulnerable
    - Admins don't install patches ... Why?
- Cisco Resource Manager (CRM)
  - Administrative tool, runs on admin machine
  - Logs everything admin does (including username/pwd)
  - World-readable file; anyone on system can read it
- Legato Networker, 2002
  - Also logs username/pwds
  - Log file not protected

Unintended functionality

- Idea
  - Designer tries to add useful features
  - Introduces vulnerability in the process
- Example
  - %pipe in postscript file allows Ghostview to read, delete files
  - Partial protection: "ghostview -d SAFER" helps
- Related examples
  - Similar attack on some Unix, Linux PDF readers
    - Victim clicks on a hyperlink in malicious PDF file
    - Shell used to start external program to handle hyperlink
    - Attacker executes arbitrary command with privileges of victim
  - Macro languages (e.g., Word macros)
- Lesson
  - Think about security implications of features