CS 177 - Computer Security

Malicious Software
(Malware)
Thanksgiving Special

Material in this deck will *not* be covered by the final exam
Introduction

• Malicious Code (Malware)
  – software that fulfills malicious intent of author
  – mass-market malware vs. targeted attacks (e.g., “advanced persistent threats”)
  – term often used equivalent with virus (due to media coverage)
  – however, many different types exist
  – classic viruses account for only 3% of malware in the wild
Taxonomy
Reasons for Malware Prevalence

- Software vulnerabilities
  - unfortunately, still frequent
- Homogeneous computing base
  - Windows, Android, iOS are tempting targets
- Unprecedented connectivity
- Large and often security-unaware user base
- Malicious code has become profitable
Virus

Definition

A virus is a program that reproduces its own code by attaching itself to other executable files in such a way that the virus code is executed when the infected executable file is executed
Virus Infection Strategies

• **File infectors**
  – simple overwrite virus (damages original program)
  – parasitic virus
    • append virus code and modify program entry point
  – cavity virus
    • inject code into unused regions of program code

• **Boot virus**
  – master boot record (MBR) of hard disk (first sector on disk)
  – boot sector of partitions
  – rather old, but interest is growing again
    • diskless workstations, virtual machine virus (SubVirt)
    • attacks against UEFI
Virus Infection Strategies

- **Code Integration**
  - merge virus code with program
  - requires disassembly of target
    - difficult task on x86 machines
  - W95/Zmist is a classic example for this technique
(Traditional) Virus Defense

- **Antivirus Software**
  - working horse is signature-based detection
    - database of byte-level or instruction-level signatures that match virus
    - wildcards can be used, regular expressions
  - heuristics (check for signs of infection)
    - code execution starts in last section
    - incorrect header size in PE header
    - suspicious code section name
    - patched import address table
Polymorphism and Metamorphism

• Polymorphic viruses
  – change layout (shape) with each infection
  – payload is encrypted
  – using different key for each infection
  – makes static string analysis practically impossible
  – of course, encryption routine must be changed as well
  – otherwise, detection is trivial

• Metamorphic techniques
  – create different “versions” of code that look different but have the same semantics (i.e., do the same)
Chernobyl (CIH) Virus

5B 00 00 00 00 00 8D 4B 42 51 50 50 0F 01 4C 24 FE 5B 83 C3 1C FA 8B 2B

pop ebx
lea ecx, [ebx + 42h]
push ecx
push eax
push eax
sidt [esp - 02h]
pop ebx
add ebx, 1Ch
cli
mov ebp, [ebx]
Dead Code Insertion

```
5B 00 00 00 00    pop ebx
8D 4B 42    lea ecx, [ebx + 42h]
51    push ecx
50    push eax
90    nop
50    push eax
40    inc eax
0F 01 4C 24 FE    sidt [esp - 02h]
48    dec eax
5B    pop ebx
83 C3 1C    add ebx, 1Ch
FA    cli
8B 2B    mov ebp, [ebx]
```
# Instruction Reordering

<table>
<thead>
<tr>
<th>mnemonic</th>
<th>opcodes</th>
<th>comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>5B 00 00 00 00</td>
<td>pop ebx</td>
<td></td>
</tr>
<tr>
<td>EB 09</td>
<td>jmp &lt;S1&gt;</td>
<td></td>
</tr>
<tr>
<td>S2:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>push eax</td>
<td></td>
</tr>
<tr>
<td>0F 01 4C 24 FE</td>
<td>sidt [esp - 02h]</td>
<td></td>
</tr>
<tr>
<td>5B</td>
<td>pop ebx</td>
<td></td>
</tr>
<tr>
<td>EB 07</td>
<td>jmp &lt;S3&gt;</td>
<td></td>
</tr>
<tr>
<td>S1:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8D 4B 42</td>
<td>lea ecx, [ebx + 42h]</td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>push ecx</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>push eax</td>
<td></td>
</tr>
<tr>
<td>EB F0</td>
<td>jmp &lt;S2&gt;</td>
<td></td>
</tr>
<tr>
<td>S3:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>83 C3 1C</td>
<td>add ebx, 1Ch</td>
<td></td>
</tr>
<tr>
<td>FA</td>
<td>cli</td>
<td></td>
</tr>
<tr>
<td>8B 2B</td>
<td>mov ebp, [ebx]</td>
<td></td>
</tr>
</tbody>
</table>

```
5B 00 00 00 00 EB 09 50 0F 01 4C 24 FE 5B EB 07 8D 4B 42 51 50 EB F0 83 C3 1C FA 8B 2B
```
Instruction Substitution

<table>
<thead>
<tr>
<th>Instruction</th>
<th>decode</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>5B 00 00 00 00 00</td>
<td>pop ebx</td>
<td></td>
</tr>
<tr>
<td>8D 4B 42</td>
<td>lea ecx, [ebx + 42h]</td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>push ecx</td>
<td></td>
</tr>
<tr>
<td>89 04 24</td>
<td>mov eax, [esp]</td>
<td></td>
</tr>
<tr>
<td>83 C4 04</td>
<td>add 04h, esp</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>push eax</td>
<td></td>
</tr>
<tr>
<td>0F 01 4C 24 4E</td>
<td>sidt [esp - 02h]</td>
<td></td>
</tr>
<tr>
<td>83 04 24 0C</td>
<td>add 1Ch, [esp]</td>
<td></td>
</tr>
<tr>
<td>5B</td>
<td>pop ebx</td>
<td></td>
</tr>
<tr>
<td>8B 2B</td>
<td>mov ebp, [ebx]</td>
<td></td>
</tr>
</tbody>
</table>

```
5B 00 00 00 00 8D 4B 42 51 89 04 24 83 C4 04 50 0F 01 4C 24 FE 83 04 24 0C 5B 8B 2B
```
Advanced Virus Defense

- Malware techniques very effective against (pure) static analysis

- Thus, dynamic analysis techniques introduced
  - virus scanner equipped with emulation engine
  - executes actual instructions (no disassembly problems)
  - runs until polymorphic part unpacks actual virus
  - then, signature matching can be applied

- Difficulties
  - virus can attempt to detect emulation engine
  - time execution, use exotic (unsupported) instructions, …
  - insert useless instructions in the beginning of code to deceive scanner
Advanced Virus Defense

- **Stalling loops**
  - exploit overhead of analysis system
  - execute “slow” operation many (millions of) times

```
1 unsigned count, tick;
2
3 void helper() {
4    tick = GetTickCount();
5    tick++;
6    tick++;
7    tick = GetTickCount();
8 }
9
10 void delay() {
11    count=0x1;
12    do {
13        helper();
14        count++;
15    } while (count!=0xe1c1);
16 }
```

Real host - A few milliseconds
Sandbox - Ten hours

Figure 1. Stalling code found in real-world malware (W32.DelfInj)
Advanced Virus Defense

• Behavior-based detection on the endpoint
  – EDR (Endpoint detection and response)
  – Observe system call activity
  – Match against models of “bad” behaviors
A self-replicating program able to propagate itself across networks, typically having a detrimental effect.

(Oxford English Dictionary)

- Worms either
  - exploit vulnerabilities that affect large number of hosts
  - send copies of worm body via email

- Difference to classic virus is *autonomous* spread over network
Target Locator

- Email harvesting
  - consult address books (W32/Melissa)
  - files might contain email addresses
    - inbox of email client (W32/Mydoom)
    - Internet Explorer cache and personal directories (W32/Sircam)
  - even Google searches are possible
    - search worms (W32/MyDoom.O)

- Network share enumeration
  - Windows discovers local computers, which can be attacked
  - some worms attack everything, including network printers
  - prints random garbage (W32/Bugbear)
Target Locator

• Scanning
  – randomly generate IP addresses and send probes
  – interestingly, many random number generators flawed
    • static seed
    • not complete coverage of address space
  – scanning that favors local addresses (topological scanning)
    – some worms use hit-list with known targets (shorten initial phase)

• Service discovery and OS fingerprinting performed as well
Email-Based Worms

- Often use social engineering techniques to get executed
  - fake from address
  - promise interesting pictures or applications
  - hide executable extension (.exe) behind harmless ones (.jpeg)

- Many attempt to hide from scanners
  - packed or zipped
  - sometimes even with password (ask user to unpack)

- Speed of spread limited because humans are in the loop
  - can observe spread patterns that correspond to time-of-day
Email-Based Worms

Hi,

It's me

I am a student. I'm studying international relationships. I would like to find an interesting and active man for serious relations. Sitting at home it is not for me. I like to go out to the theater, cinema, and nightclubs.

Attached file tells everything.

---

Subject: FW: microsoft patch
From: Microsoft Corporation Security Assistance [mailto:gpddonwyregof@newsletters.net]
Sent: Friday, September 19, 2003 8:35 AM
To: MS Corporation Client

Subject: microsoft patch

MS Client

This is the latest version of security update, the "September 2003, Cumulative Patch" update which resolves all known security vulnerabilities affecting MS Internet Explorer, MS Outlook and MS Outlook Express. Install now to maintain the security of your computer from these vulnerabilities, the most serious of which could allow an malicious user to run executable on your system. This update includes the functionality of all previously released patches.

System requirements
Windows 95/98/Me/2000/NT/XP

This update applies to:
MS Internet Explorer, version 5.1 and later
MS Outlook, version 5.0.0 and later
MS Outlook Express, version 5.0 and later

Recommendations
Customers should install the patch at the earliest opportunity.

How to install
Run attached file. Choose Yes on displayed dialog box.

How to use
You don't need to do anything after installing this item.

Microsoft Product Support Services and Knowledge Base articles can be found on the Microsoft Technical Support web site. For security-related information about Microsoft products, please visit the Microsoft Security Advisory web site, or Contact Us.

Thank you for using Microsoft products.

Please do not reply to this message. It was sent from an automated email address and we are unable to respond to any replies.

The names of the actual companies and products mentioned herein are the trademarks of their respective owners.
Email-Based Worms
Exploit-Based Worms

• Require no human interaction
  – typically exploit well-known network services
  – can spread much faster

• Propagation speed limited either
  – by network latency
    worm thread has to establish TCP connection (Code Red)
  – by bandwidth
    worm can send (UDP) packets as fast as possible (Slammer)

• Spread can be modeled using classic disease model
  – worm starts slow (only few machines infected)
  – enters phase of exponential growth
  – final phase where only few uncompromised machines left
Exploit-Based Worms

![Graph showing the infection rate of the Code Red Worm over time](image-url)
Worm Defense

• Malware scanners
  – most effective against email-based worms
  – email attachments can be scanned as part of mail processing

• Host level defense
  – mostly targeted at underlying software vulnerabilities
  – code audits
  – stack-based techniques
    • StackGuard, MS VC compiler extension
  – address space layout randomization (ASLR)
    • attempt to achieve diversity to increase protection
Botnets

• **Bot**
  – autonomous programs performing tasks

• **Benign bots**
  – first bots were programs used for Internet Relay Chat (IRC)
  – react to events in IRC channels
  – typically offer useful services

• **Early definition of bot**

  *An IRC user who is actually a program. On IRC, typically the robot provides some useful service. Examples are NickServ, which tries to prevent random users from adopting nicks already claimed by others.*
Botnets

• Bots today
  – malware running on compromised machines
  – incorporates different modules to carry out malicious tasks (spamming, DoS, …)
  – remote controlled by criminal entity (called bot master, bot herder)

• Bots are incorporated in network of compromised machines
  → Botnets (can be hundreds of thousands of infected machines)
Botnet Creation

• Hosts infected by one of
  – *email attachments*
  – *infected version of legitimate program (video players)*
  – vulnerabilities (drive-by-downloads, network services)
  – existing backdoor (from previous infection)
Botnet Architectures

• Bot overlay network
  – centralized
    • IRC server (Internet relay chat)
    • web server (HTTP)
    • multiple controllers for robustness
  – peer-to-peer: self organizing
    • each host can be a worker or a proxy; decided dynamically
    • multi-level hierarchies possible

• Push versus pull designs
  – Attacker sends out message to tell bots what to do (push)
  – Worker bots “ask” for work to do (pull)
Centralized Botnet
Botnets

they can tell when they get knocked offline if the server dies... like 100 bots

**X1-[44325]** (anym@irc.com-19255.plano1.tx.home.com)

quit [05:29] Connection reset by peer

**X1-[23831]** [Packing]: Halted!

**X1-[23831]** Pepsi Attack Started On < IP: 207.71.92.193 Amount: 1000 Size: 180 Port: 80 >

**X1-[93881]**

**X2-[38556]**

**X1-[44882]**

**X1-[78622]**

**X1-[73958]**

**X1-[8860]**

**X1-[92898]**

**X1-[93881]**

**X2-[20149]**

**X2-[38247]**

**X2-[42096]**

**X2-[69963]**

<sigh> X1-[33165]

<sigh> ban that

<sigh> X1-[44325]

<sigh> well

<sigh> im using that bot

207.71.92.193

grc.com

PING? PONG!
Botnet Evolution

- IRC server
  - often easy to take down certain hard-coded IP (dynamic DNS)
  - traffic easier to detect (switch to HTTP)

- HTTP
  - rotating domains (*rendez-vous* points)
    - computation based on current date
    - hard to take down many domains, must also do it quickly
    - reverse engineering domain generation algorithm important
  - Torpig
    - one new domain name per week, multiple TLDs
  - Conficker
    - list of 250 domains, 8 times per day
    - send queries to Google to obtain current time
Botnet Evolution

• Fast flux
  – network of bots with fast changing DNS records
  – many IP addresses for single DNS name (A records)
  – advanced type also change NS records (double flux)
  – used to hide mother-ship (content) behind proxy network
Botnet Evolution

**Diagram: Single-Flux**
- "bullet-proof" hosted DNS server
- 1) Query: flux.example.com
- 2) Referral: ns.example.com
- 3) Query: 10.10.10.10
- 4) Answer: ns.example.com

**Diagram: Double-Flux**
- "mothership" (zombie home PC)
- 1) Query: flux.example.com
- 2) Referral: ns.example.com
- 3) Query: 10.10.10.10
- 4) Query redirected & Response returned
- 5) Answer: ns.example.com

**DNS Resolution Comparison**
Botnet Evolution

dhcp-41-209:~ chris$ dig canadian-pharmacy.com

; <<< DiG 9.3.5-P2 <<<> canadian-pharmacy.com
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 688
;; flags: qr rd ra; QUERY: 1, ANSWER: 7, AUTHORITY: 4, ADDITIONAL: 4

;; QUESTION SECTION:
;canadian-pharmacy.com. IN A

;; ANSWER SECTION:
canadian-pharmacy.com. 1789 IN A 69.25.27.170
 canadian-pharmacy.com. 1789 IN A 69.25.27.173
 canadian-pharmacy.com. 1789 IN A 63.251.171.80
 canadian-pharmacy.com. 1789 IN A 63.251.171.81
 canadian-pharmacy.com. 1789 IN A 66.150.161.136
 canadian-pharmacy.com. 1789 IN A 66.150.161.140
 canadian-pharmacy.com. 1789 IN A 66.150.161.141
Example – Storm P2P Botnet

Bot master

HTTP proxies

Proxy bots

Overnet

Worker bots
Botnet Applications

• Spam

• Denial-of-Service (DoS)

• Information theft / data exfiltration

• Mining of cryptocurrencies
Spam

• Use bots
  – to avoid blacklisting (such as Spamhaus DNSBL)
  – in addition to using open proxies

~95% of bots listed in one or more blacklists

vs. ~80% on average
Denial-of-Service

- Botnet of infected IoT devices (Mirai)
- Botnet of infected cloud hosts (Mantis)
  - Cloudflare reported 26M HTTPS requests / second attack in 06/2022
Cryptomining

- Typically focuses on mining Monero
  - provides strong privacy guarantees
  - can be successfully mined without specialized hardware
Botnet Defense

- Hosted-based malware detection
- Rule-based
  - monitor outbound network connections
    block certain ports (25, 6667, ...)
- Network content
  - Match network packet contents to known command strings (keywords)
    e.g., DoS command – .ddos.httpflood
- Network traffic monitoring
  - IP addresses, domains, URLs (reputation)
  - connection patterns (NDR)
- Take down command and control infrastructure
Network Detection and Response (NDR)

- **IDS-based sensor suite for malware event detection**
  - remote-to-local exploit detection
  - Botnet component downloads
  - Machine-learning-based detection of suspicious network behaviors
    - network scans
    - command and control connections / beaconing
    - lateral spread

- **Event correlation**
  - combines information from sensors to recognize bots that infect and coordinate with your internal network assets
Time Series Analysis

- Look for unexpected connections / volume over time
Beaconing Detector

- Look for (semi)-periodic check-ins with similar properties
Generic Infection Lifecycle
TLS Detector

- Learn set of features in TLS connection setup that indicates malicious activity
Ransomware

- Malware that encrypts victim's files, making them inaccessible, and demands a ransom payment to decrypt them
- Problem surged over the last few years
- Many high-profile incidents and targets
- Enabled by use of cryptocurrencies for ransom payments

Average ransom demands compared to average ransom payments in 2020 and 2021, according to Unit 42 incident response data

https://unit42.paloaltonetworks.com/2022-ransomware-threat-report-highlights/
Ransomware

- Early families used weak encryption or did not protect keys properly
  - this has changed
- Victims can respond by restoring data from backups
  - disruptive and costly, but better than complete loss of data
  - this led to double extortion attacks (where victim is threatened with making data public)

https://unit42.paloaltonetworks.com/2022-ransomware-threat-report-highlights/
Rootkits

- Tools used by attackers after compromising a system
  - hide presence of attacker
  - allow for return of attacker at later date
  - gather information about environment
  - attack scripts for further compromises

- Traditionally trojaned set of user-space applications
  - system logging (syslogd)
  - system monitoring (ps, top)
  - user authentication (login, sshd)
Kernel Rootkits

- **Kernel-level rootkits**
  - kernel controls view of system for user-space applications
  - malicious kernel code can intercept attempts by user-space detector to find rootkits

- **Modifies kernel data structures**
  - process listing
  - module listing

- **Intercepts requests from user-space applications**
  - system call boundary
  - VFS fileops struct