CS 290
Host-based Security and Malware

Christopher Kruegel
chris@cs.ucsb.edu

Malicious Code

Overview

- Introduction to malicious code
  - taxonomy, history, life cycle
- Virus
  - infection strategies, armored viruses, detection
- Worms
  - email- and exploit-based worms, spreading strategies
- Trojan horses
  - key logger, rootkits, botnet, spyware
  - demos that demonstrate danger of malicious code on local host

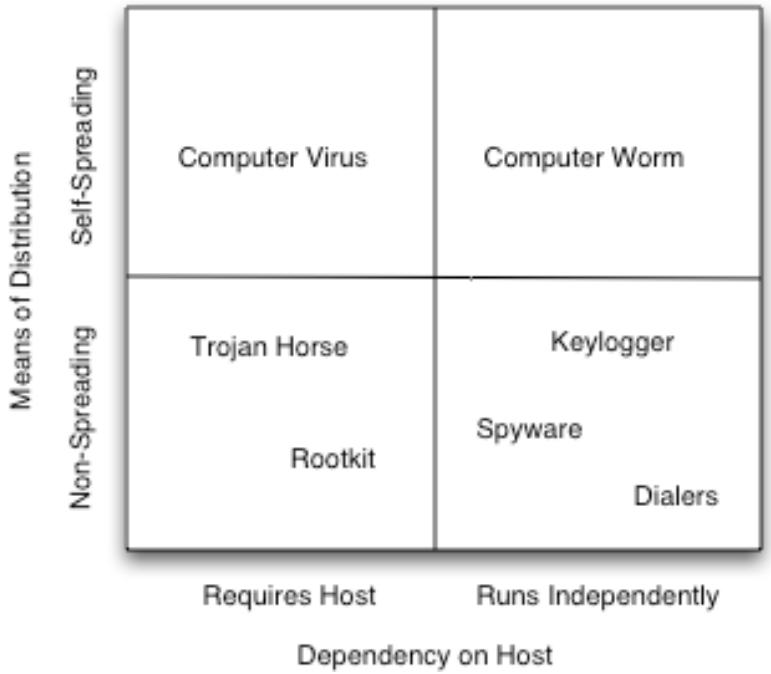
Introduction

- Malicious Code (Malware)
  - software that fulfills malicious intent of author
  - 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

- 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

Taxonomy


Means of Distribution
Self-Spreading
Non-Spreading
Computer Virus
Computer Worm
Trojan Horse
Keylogger
Spyware
Rootkit
Dialers
Requires Host
Runs Independently
Dependency on Host

Taxonomy

- Virus
  - self-replicating, infects files (thus requires host)
- Worm
  - self-replicating, spreads over network
- Interaction-based worms (B[e]agle, Netsky, Sobig)
  - spread requires human interaction
  - double-click and execute extension
  - follow link to download executable
- Process-based worms (Code Red, Blaster, Slammer)
  - requires no human interaction
  - exploits vulnerability in network service

Reasons for Malware Prevalence

- Mixing data and code
  - violates important design property of secure systems
  - unfortunately very frequent
- Homogeneous computing base
  - Windows is just a very tempting target
- Unprecedented connectivity
  - easy to attack from safety of home
- Clueless user base
  - many targets available
- Malicious code has become profitable
  - compromised computers can be sold (e.g., spam, DoS, banking)

Virus Lifecycle

- Lifecycle
  - reproduce, infect, run payload
- Reproduction phase
  - viruses balance infection versus detection possibility
  - variety of techniques may be used to hide viruses
- Infection phase
  - difficult to predict when infection will take place
  - many viruses stay resident in memory (TSR or process)
- Attack phase
  - e.g., deleting files, changing random data on disk
  - viruses often have bugs (poor coding) so damage can be done
    - Stoned virus expected 360K, floppy, corrupted sectors

Infection Strategies

- Boot viruses
  - master boot record (MBR) of hard disk (first sector on disk)
  - boot sector of partitions
  - e.g., Pakistani Brain virus
  - rather old, but interest is growing again
    - diskless work stations, virtual machine virus (SubVirt)
    - *MebRoot*
- 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

Infection Strategies

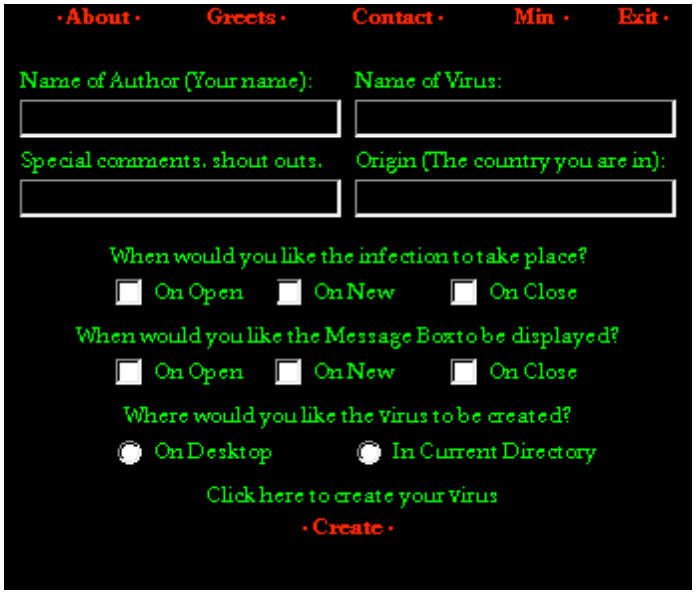
- Entry Point Obfuscation
  - virus scanners quickly discovered to search around entry point
  - virus hijacks control later (after program is launched)
  - overwrite import table addresses
  - overwrite function call instructions

- 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

Macro Viruses

- Many modern applications support macro languages
  - Microsoft Word, Excel, Outlook
  - macro language is powerful
  - embedded macros automatically executed on load
  - mail app. with Word as an editor
  - mail app. with Internet Explorer to render HTML

I made this program to all those people who want to write Word 2000 virii, but don't know what the hell to do.



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
- Sandboxing
  - run untrusted applications in restricted environment
  - simplest variation, do not run as Administrator

Tunneling and Camouflage Viruses

- To minimize the probability of its being discovered, a virus could use a number of different techniques
- A tunneling virus attempts to bypass antivirus programs
  - idea is to follow the interrupt chain back down to basic operating system or BIOS interrupt handlers
  - install virus there
  - virus is “underneath” everything – including the checking program
- In the past, possible for a virus to spoof a scanner by camouflaging itself to look like something the scanner was programmed to ignore
  - false alarms of scanners make “ignore” rules necessary

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

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

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]
```

```
5B 00 00 00 00 8D 4B 42 51 50 90 50 40 0F 01 4C 24
FE 48 5B 83 C3 1C FA 8B 2B
```

Instruction Reordering

```
5B 00 00 00 00      pop ebx
EB 09               jmp <S1>
S2:
50                  push eax
0F 01 4C 24 FE      sidt [esp - 02h]
5B                  pop ebx
EB 07               jmp <S3>
S1:
8D 4B 42            lea ecx, [ebx + 42h]
51                  push ecx
50                  push eax
EB F0               jmp <S2>
S3:
83 C3 1C            add ebx, 1Ch
FA                  cli
8B 2B               mov ebp, [ebx]
```

```
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

```
5B 00 00 00 00     pop ebx
8D 4B 42           lea ecx, [ebx + 42h]
51                 push ecx
89 04 24           mov eax, [esp]
83 C4 04           add 04h, esp
50                 push eax
0F 01 4C 24 FE     sidt [esp - 02h]
83 04 24 0C        add 1Ch, [esp]
5B                 pop ebx
8B 2B              mov ebp, [ebx]
```

```
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

- Most virus techniques very effective against 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
  - emulation must be fast
  - ANUBIS
- 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

Computer Worms

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
- Speed of spreading is constantly increasing
- Make use of techniques known by virus writers for long time

Worm Components

- Target locator
  - how to choose new victims
- Infection propagator
  - how to obtain control of victim
  - how to transfer worm body to target system
- Life cycle manager
  - control different activities depending on certain circumstances
  - often time depending
- Payload
  - nowadays, often a Trojan horse (we come back to that later)

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)

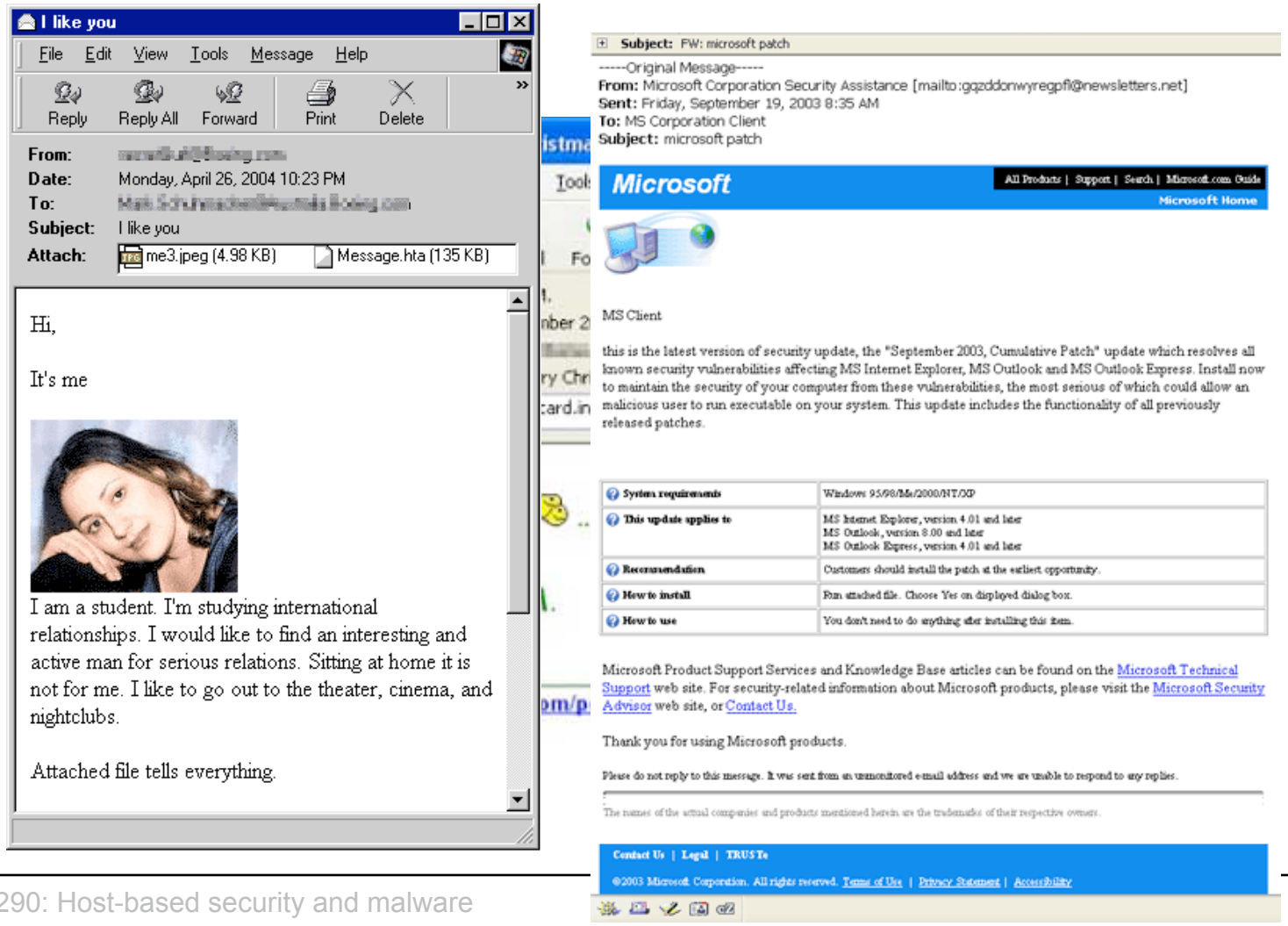
Target Locator

- Scanning
  - more Google searches
    - search for vulnerable web applications (Santy)
  - 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

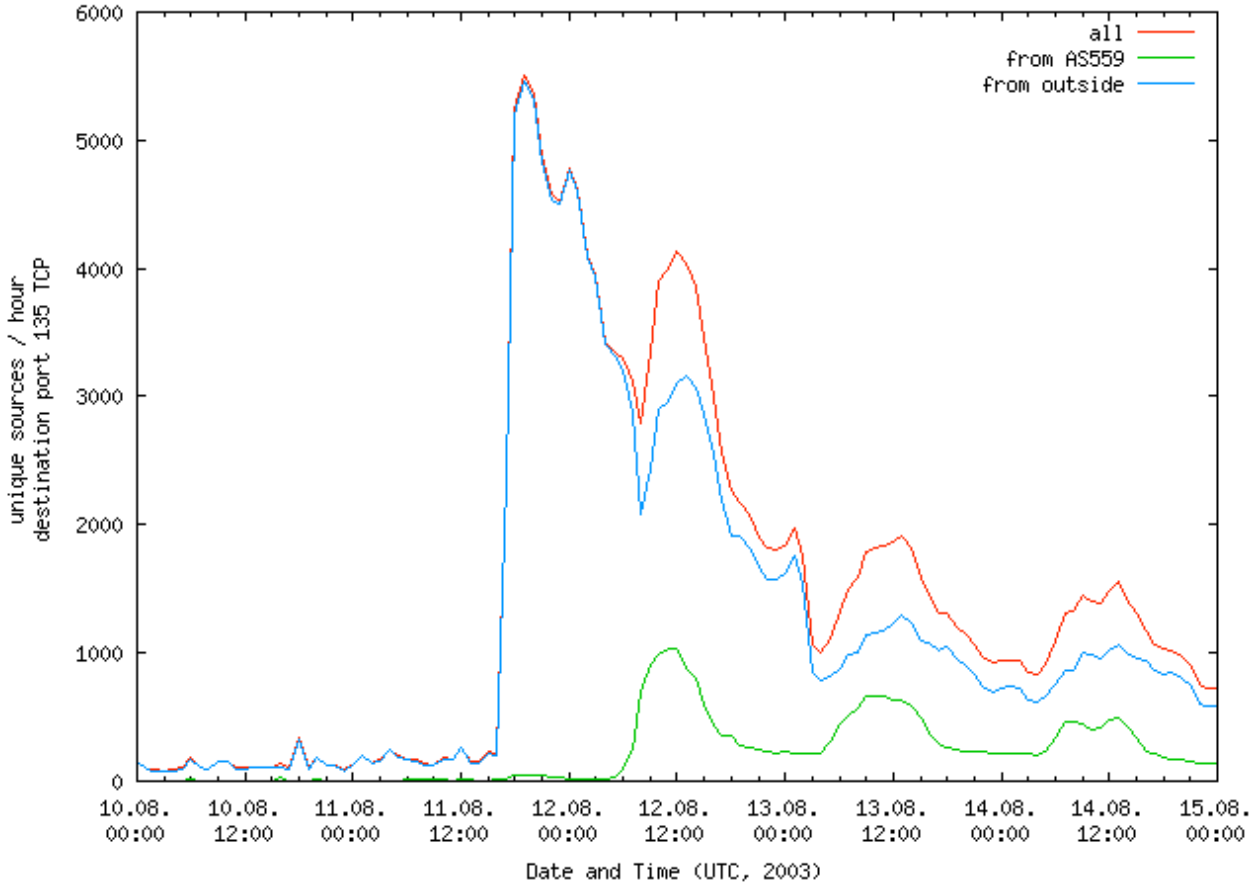
Email-Based Worms

- 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)
- Some exploit Internet Explorer bugs when HTML content is rendered
- Significant impact on SMTP infrastructure
- Speed of spread limited because humans are in the loop
  - can observe spread patterns that correspond to time-of-day

Email-Based Worms



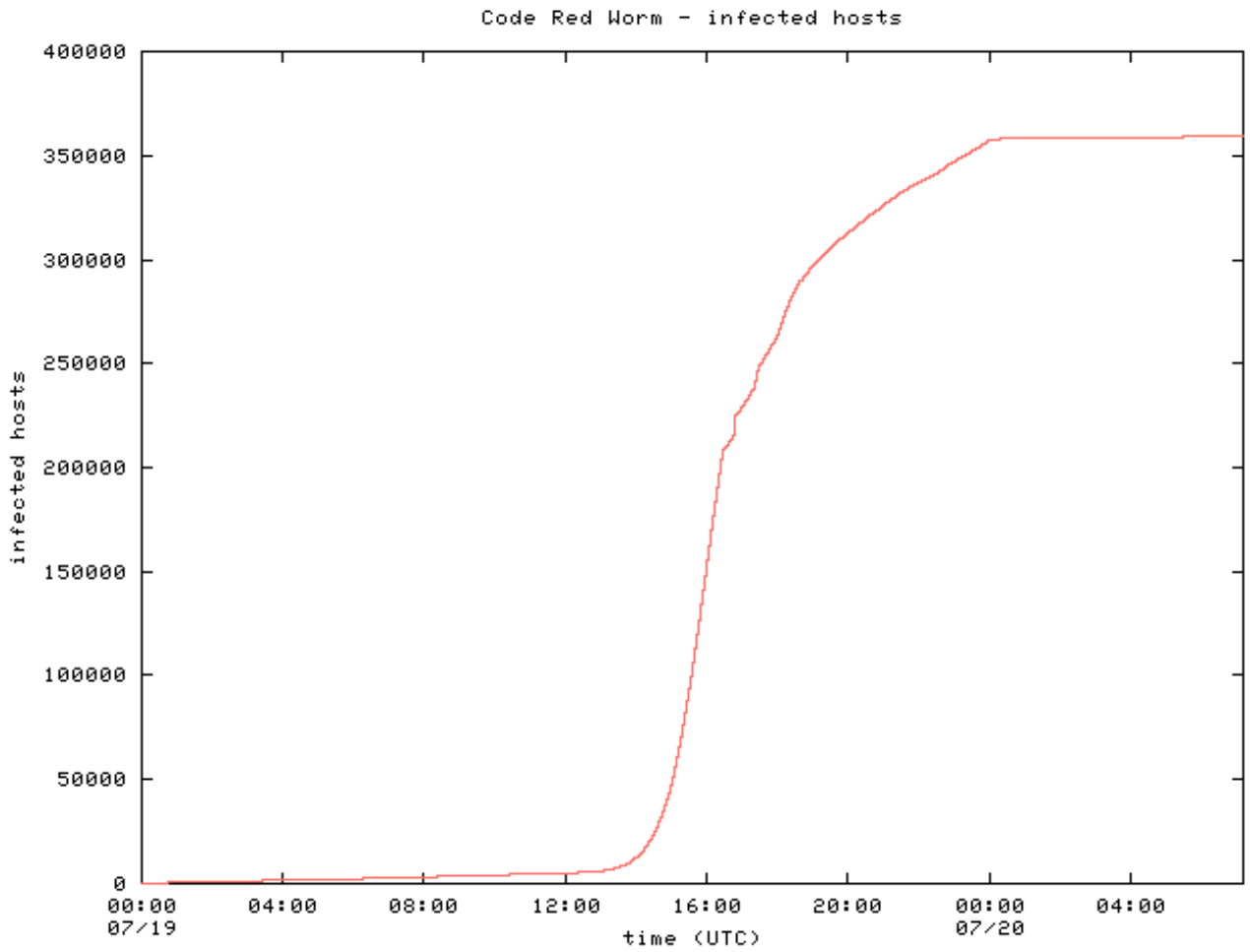
Email-Based Worms



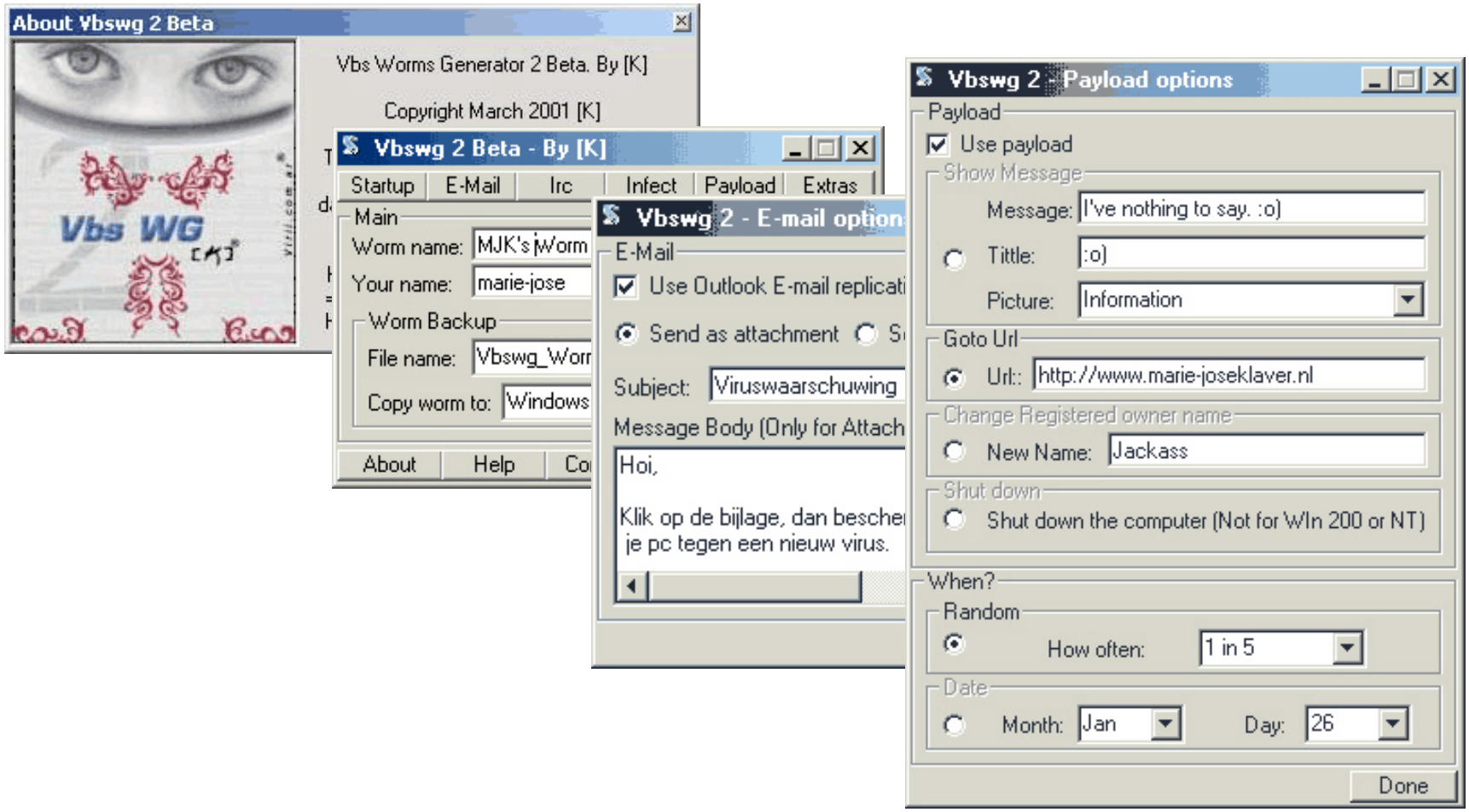
Exploit-Based Worms

- 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



Worm Generators



Worm Defense

- Virus scanners
  - 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

Worm Defense

- Network level defense
  - intrusion detection systems
    - scan for known attack patterns
    - automatic signature generation (Early Bird, Autograph, Polygraph)
  - rate limiting
    - allow only certain amount of outgoing connections
    - helps to contain worms that perform scanning
  - personal firewall
    - block outgoing SMTP connections (from unknown applications)

Trojan Horse

- Trojan horse is a malicious program that is disguised as legitimate software
  - software may look useful or interesting (or at the very least harmless)
  - term derived from the classical myth of the Trojan Horse

- Two types of Trojan horses
  1. malicious functionality is included into useful program
     - disk utility, screensaver, weather alert program
     - famous compiler that generated backdoor into code
  2. malware is stand-alone program
     - possibly disguised file name (sexy.jpg.exe)

Trojan Horse

- Many different types and functions
  - spy on (sensitive) user data
    - log keystrokes, monitor surfing activity
  - disguise presence
    - rootkits
  - allow remote access
    - file transfer, remote program execution
    - base for further attacks, mail relay (for spammers)
    - Back Orifice, NetBus, SubSeven
  - damage routines
    - corrupting files
    - participate in denial of service attacks

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

Linux Kernel Rootkits

- Linux kernel exports well-defined interface to modules
- Examples of legitimate operations
  - registering device with kernel
  - accesses to devices mapped into kernel memory
  - overwriting exported function pointers for event callbacks
- Kernel rootkits violate these interfaces
- Examples of illegal operations
  - replacing system call table entries (knark)
  - replacing VFS fileops (adore-ng)

Linux Kernel Rootkits

- System call table hijacking

```
orig_getuid = sys_call_table[__NR_getuid]; sys_call_table
[__NR_getuid] = give_root;
```

- VFS hijacking

```
pde = proc_find_tcp();
o_get_info_tcp = pde->get_info;
pde->get_info = n_get_info_tcp;
```

- Works pretty much the same for Windows
  - anyone remember the Sony rootkit discussion?

Windows Kernel Rootkits

WIRED NEWS

Search: Wired Ne

Top | Technology | Culture | Politics | News Wires

University of Phoenix ONLINE

Real Story of the Rogue Rootkit

PRINT MAIL RANTS + RAVES

By Bruce Schneier | Also by this reporter
02:00 AM Nov, 17, 2005

It's a David and Goliath story of the tech blogs
mega-corporation.

On Oct. 31, Mark Russinovich broke the story in
Sony BMG Music Entertainment distributed a co
scheme with music CDs that secretly installed a
computers. This software tool is run without you
consent -- if it's loaded on your computer wit
hacker can gain and maintain access to your sy
wouldn't know it.

The Sony code modifies Windows so you can't t
a process called "cloaking" in the hacker world.
spyware, surreptitiously sending information ab
Sony. And it can't be removed: trying to get rid

[text only] [mobil]

derStandard.at/Web

NEWSROOM | Politik | Investor | Web | Sport | Panorama | Etat
LIVINGROOM | Innovationen | IT-Business | Telekom | Netzpolitik | PDA | Games

derStandard.at | WebStandard | Archiv

27. März 2006
15:09

Sony-Chef entschuldigt sich für aggressiven Kopierschutz
"Inhalte und Technologie sind merkwürdige Bettgenossen"

Der Chef des Unterhaltungs-elektronikkonzerns Sony, Howard Stringer, hat sich für den aggressiven Kopierschutz des Musiklabels Sony BMG entschuldigt.

Der Chef des Unterhaltungselektronikkonzerns Sony, Howard Stringer, hat sich für den aggressiven Kopierschutz des Musiklabels Sony BMG entschuldigt. "Sony BMG hatte nicht die Absicht, den Konsumenten zu bestrafen", sagte Stringer auf der Consumer Electronics Show (CES) in Las Vegas.

Kopierschutzprogramme

Das Label hatte einzelne Musik-CDs mit Kopierschutzprogrammen ausgestattet, die sich wie Computerviren auf den PCs der Kunden einnisten konnten und die Rechner Sicherheitsrisiken aus dem Internet aussetzten. Nach massiver Kritik der Kunden und verschiedenen Klagen hatte Sony BMG von

Link

Sony BMG

Nachlese

Sony BMG legt "Rootkit"-Sammelklage bei

Mehr zum Thema

Internet
SPRIT.ORG Domain & Webhosting
Domainreg. ab € 9.90 /Jahr inkl.

ZuZweit
Partnersuche auf derStandard.at/ZuZweit

Wissen

Windows Kernel Rootkits

- Sony rootkit filters out any files/directories, processes and registry keys that contain `$sys$`
- System call dispatcher
  - uses system service dispatch table (SSDT)
  - Windows NT kernel equivalent to system call table
  - entries can be manipulated to re-route call to custom function

`ZwCreateFile`
  - used to create or open file

`ZwQueryDirectoryFile`
  - used to list directory contents (i.e. list subdirectories and files)

`ZwQuerySystemInformation`
  - used to get the list of running processes (among other things)

`ZwEnumerateKey`
  - used to list the registry keys below a given key

Rootkit Defense

- `tripwire`
  - user-space integrity checker
- `chkrootkit`
  - user-space, signature-based detector
- `kstat, rkstat, St. Michael`
  - kernel-space, signature-based detector
  - implemented as kernel modules or use /dev/kmem
- Limitations
  - typically, rootkit must be loaded in order to detect it
  - thus, detectors can be thwarted by kernel-level rootkit
  - also suffer from limitations of signature-based detection

Rootkit Defense

- Kernel rootkits
  - have complete control over operating system
  - operating system is part of trusted computing base, thus applications can be arbitrarily fooled
  - this includes all rootkit or Trojan detection mechanisms
  - at best, an arms race can be started

- Proposed solutions
  - trusted computing platform
    - can enforce integrity of operating system
  - smart cards
    - attacker can not influence computations on card, but has still full control of computations performed on machine and information displayed on screen

Spyware

- Any software that monitors and collects information about a user in a covert and unsolicited manner
- Goal of spyware
  - collect sensitive user information and surfing habits
- Task of spyware
  - component must monitor user behavior
  - component must leak information to environment (OS, network)
- Often implemented as browser extensions
  - Internet Explorer Browser Helper Object (BHO)
  - COM object that can hook into Microsoft's Internet Explorer
  - monitor/modify events

Spyware

- Interaction
  - between browser and spyware component
    - COM function invocations (exported by Internet Explorer)
  - between spyware component and operating system
    - Windows API calls

- In addition, it typically has a real company behind it that is making money from the information gathered
  - Adware is any software that injects unsolicited advertisements into a user's workspace
  - Scumware is a specific type of adware that hides other advertisements with those from its own controlling source

Spyware

Typical routes of infection:

1. spyware is bundled with legitimate software package
   - end-user license agreement (EULA) even informs about this fact
   - EULA is very long (often hundreds of pages), user accepts
   - classic examples are shareware programs
     - P2P file-sharing clients (e.g., Kazaa)

2. “drive-by” downloads
   - exploit browser bug, in particular, vulnerabilities of Internet Explorer
   - WMF (Windows meta file) exploit, around Christmas 2005
   - arbitrary code execution via mismatched DOM objects (December 2005)
   - insufficient ActiveX security settings

3. fake dialogs
   - display “Would you like to optimize your Internet” and perform installation when user agrees

Botnets

- Recent trend in malicious code development
- Often part of payload that is downloaded as Trojan horse or part of worm
- 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.

- IRC (Internet Relay Chat)
  - instant message (communication service)
  - allows for many-to-many communication in channels

Botnets

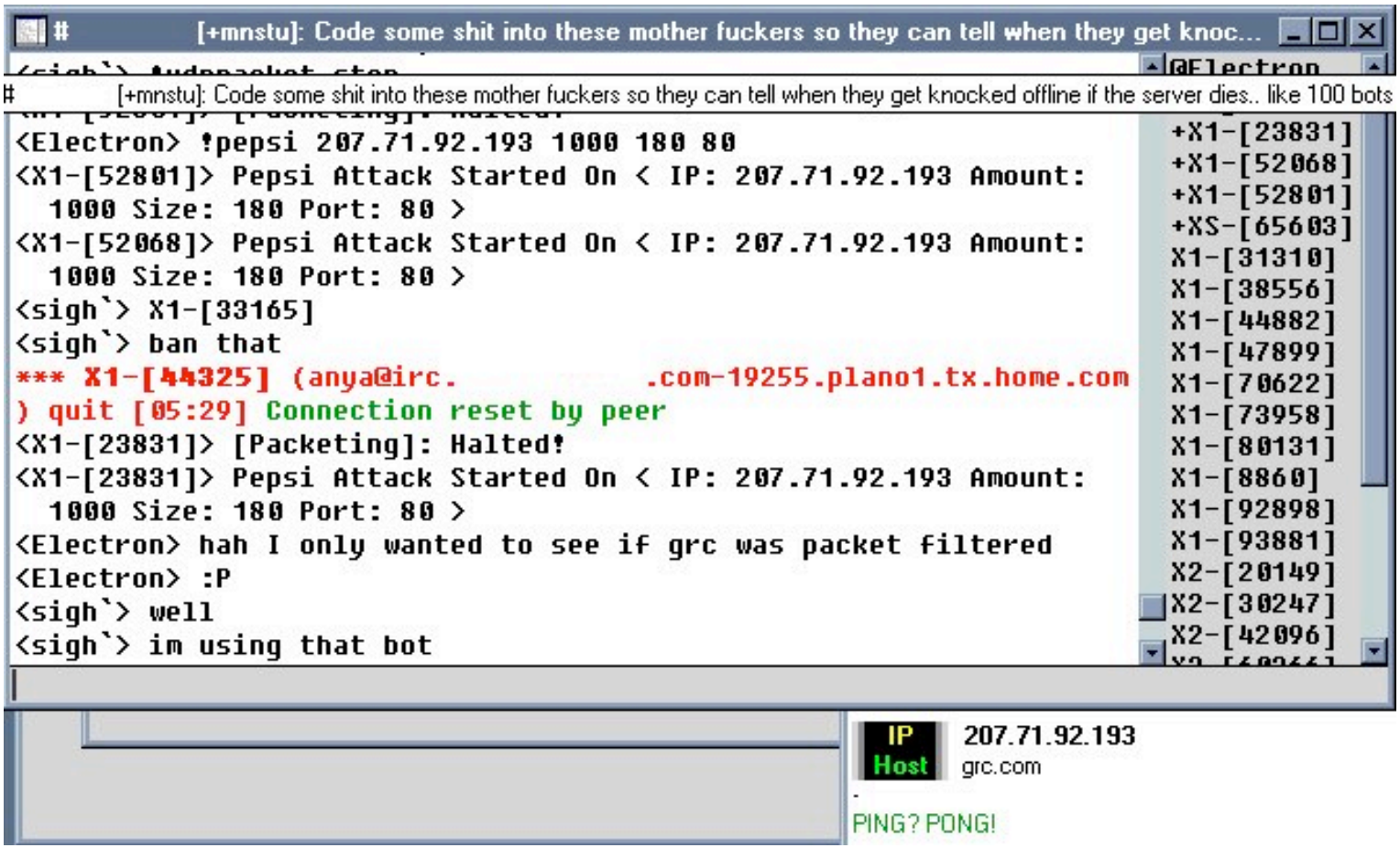
- Bots
  - first bots were programs used for Internet Relay Chat (IRC)
  - react at events in IRC channels
  - typically offer useful services
  - malicious bots started to evolve
    - takeover wars to control certain IRC channels
    - often involved denial of service to force IRC net split
  - nowadays, term refers to remote program loaded on a computer after compromise
  - usage of IRC for command and control of these programs

Botnets

- Bots today
  - implementation of several commands (e.g., DDoS)
  - spreading mechanism to propagate further
  - other functionality possible
    - key logger
    - SOCKS proxy
    - spam relay
  - some bots are even open source
    - caused massive distribution and variations
    - SDBot, AgroBot

- Bots can be incorporated in network of compromised machines
  → Botnets (sizes up to tens of thousands)

Botnets


[+mnstu]: Code some shit into these mother fuckers so they can tell when they get knoc...
[+mnstu]: Code some shit into these mother fuckers so they can tell when they get knocked offline if the server dies.. like 100 bots
<Electron> !pepsi 207.71.92.193 1000 180 80
<X1-[52801]> Pepsi Attack Started On < IP: 207.71.92.193 Amount: 1000 Size: 180 Port: 80 >
<X1-[52068]> Pepsi Attack Started On < IP: 207.71.92.193 Amount: 1000 Size: 180 Port: 80 >
<sigh`> X1-[33165]
<sigh`> ban that
*** X1-[44325] (anya@irc. .com-19255.plano1.tx.home.com
) quit [05:29] Connection reset by peer
<X1-[23831]> [Packeting]: Halted!
<X1-[23831]> Pepsi Attack Started On < IP: 207.71.92.193 Amount: 1000 Size: 180 Port: 80 >
<Electron> hah I only wanted to see if grc was packet filtered
<Electron> :P
<sigh`> well
<sigh`> im using that bot
+X1-[23831]
+X1-[52068]
+X1-[52801]
+XS-[65603]
X1-[31310]
X1-[38556]
X1-[44882]
X1-[47899]
X1-[70622]
X1-[73958]
X1-[80131]
X1-[8860]
X1-[92898]
X1-[93881]
X2-[20149]
X2-[30247]
X2-[42096]
IP
Host
207.71.92.193
grc.com
PING? PONG!

Botnet Defense

- Attack command and control infrastructure
  - take IRC channel off-line
  - when dynamic DNS is used for central command server, route traffic to black hole

- Honeypots
  - vulnerable computer that serves no purpose other than to attract attackers and study their behavior in controlled environments
  - when honeypot is compromised, bot logs into botnet
  - allows defender to study actions of botnet owners

Malware and Vulnerable Software

- Malicious software (Malware) and benign software that can be exploited to perform malicious actions (Badware) are two facets of the same problem

  → execution of unwanted code

- Malware
  - viruses, worms, Trojan horses, rootkits, and spyware are evolving to become resilient to eradication and to evade detection
- Badware
  - services and applications (especially web-based) are vulnerable to a wide range of attacks, some of which novel

Conclusions

- Malware
  - sophisticated technology developed for more than 20 years
  - combined with automatic spread mechanisms
  - tools to generate malware significantly lower technological barrier
- Trojan Horses
  - particularly dangerous because they infest trusted computing base
  - typically full control of platform and applications
- Defense Techniques
  - mostly reactive
  - using signatures to detect known instances
  - use best programming practice for application development, educate employees, keep infrastructure well maintained (patched)