Computer & Information Security (3-721-460-1)

Malware

Dept. of Software and Information Systems Engineering, Ben-Gurion University

Prof. Yuval Elovici, Dr. Asaf Shabtai, Dr. Mordechai Guri

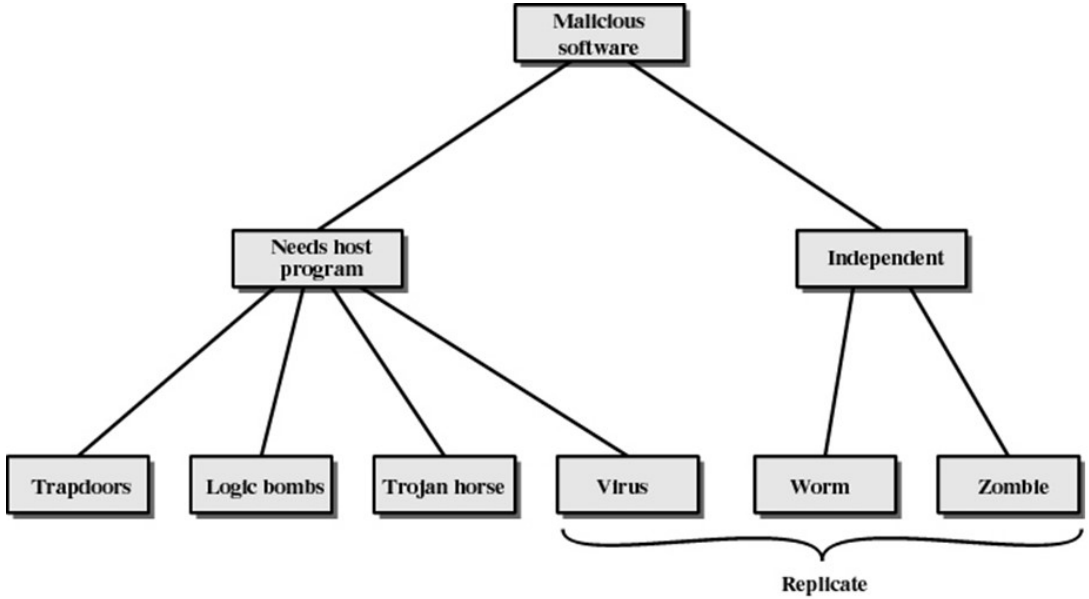
{elovici, shabtaia}@bgu.ac.il

gurim@post.bgu.ac.il

Spring, 2019

Malware

- Malware (*MALicious softWARE*) - a software designed to disrupt computer operation, delete files, gather sensitive information, commit fraud (e.g., clicks, premium numbers), gain unauthorized access to a computer system, use the computer for an attack



Malware

"a program that is inserted into a system, usually covertly, with the intent of compromising the confidentiality, integrity, or availability of the victim's data, applications, or operating system or otherwise annoying or disrupting the victim." [NIST05]

Malicious Software

- programs exploiting system or human vulnerabilities
  - program fragments that need a host program
    - e.g. viruses, logic bombs, and backdoors
  - independent self-contained programs
    - e.g. worms, bots
  - replicating or not
- sophisticated threat to computer systems

Malware Terminology

Name	Description
Adware	Advertising that is integrated into software. It can result in pop-up ads or redirection of a browser to a commercial site.
Attack Kit	Set of tools for generating new malware automatically using a variety of supplied propagation and payload mechanisms
Auto-rooter	Malicious hacker tools used to break into new machines remotely.
Backdoor (trapdoor)	Any mechanisms that bypasses a normal security check; it may allow unauthorized access to functionality in a program, or onto a compromised system.
Downloaders	Code that installs other items on a machine that is under attack. It is normally included in the malware code first inserted on to a compromised system to then import a larger malware package.
Drive-by-Download	An attack using code in a compromised web site that exploits a browser vulnerability to attack a client system when the site is viewed.
Exploits	Code specific to a single vulnerability or set of vulnerabilities.
Flooders (DoS client)	Used to generate a large volume of data to attack networked computer systems, by carrying out some form of denial-of-service (DoS) attack.
Keyloggers	Captures keystrokes on a compromised system.
Logic bomb	Code inserted into malware by an intruder. A logic bomb lies dormant until a predefined condition is met; the code then triggers an unauthorized act.

Malware Terminology

Name	Description
Macro Virus	A type of virus that uses macro or scripting code, typically embedded in a document, and triggered when the document is viewed or edited, to run and replicate itself into other such documents.
Mobile code	Software (e.g., script, macro, or other portable instruction) that can be shipped unchanged to a heterogeneous collection of platforms and execute with identical semantics.
Rootkit	Set of hacker tools used after attacker has broken into a computer system and gained root-level access.
Spammer programs	Used to send large volumes of unwanted e-mail.
Spyware	Software that collects information from a computer and transmits it to another system by monitoring keystrokes, screen data and/or network traffic; or by scanning files on the system for sensitive information.
Trojan horse	A computer program that appears to have a useful function, but also has a hidden and potentially malicious function that evades security mechanisms, sometimes by exploiting legitimate authorizations of a system entity that invokes the Trojan horse program.

Malware Terminology

Name	Description
Virus	Malware that, when executed, tries to replicate itself into other executable machine or script code; when it succeeds the code is said to be infected. When the infected code is executed, the virus also executes.
Worm	A computer program that can run independently and can propagate a complete working version of itself onto other hosts on a network, usually by exploiting software vulnerabilities in the target system.
Zombie, bot	Program activated on an infected machine that is activated to launch attacks on other machines.

Classification of Malware

- **Ransomware**
- **Fileless malware**
- **Bootkit**

APT


The Kill Chain
Define Target
Find and organize accomplices
Build or acquire tools
Research target infrastructure/ employees
Test for detection
Deployment
Initial intrusion
Outbound connection initiated
Expand access and obtain credentials
Strengthen foothold
Exfiltrate data
Cover tracks and remain undetected

Malware

- PC
- Smartphone
- IoT
- Cars
- Embedded systems
- SCADA

Classification of Malware

- **propagation method** - how it spreads or propagates to reach the desired targets
- **action/payload** - the actions or payloads it performs once a target is reached
- **need a host program** (parasitic code such as viruses) vs **independent, self-contained programs** (worms, Trojans, and bots)
- **does not replicate** (Trojans and spam e-mail) vs **does replicate** (viruses and worms)

Types of Malicious Software (Malware)

- propagation mechanisms include:
  - infection of existing content by viruses that is subsequently spread to other systems
  - exploit of software vulnerabilities by worms or drive-by-downloads to allow the malware to replicate
  - social engineering attacks that convince users to bypass security mechanisms to install Trojans or to respond to phishing attacks

Types of Malicious Software (Malware)

- payload actions performed by malware once it reaches a target system can include:
  - corruption of system or data files
  - theft of service/make the system a zombie agent of attack as part of a botnet
  - theft of information from the system/keylogging
  - stealthing/hiding its presence on the system

Viruses

- piece of software that infects programs
  - modifies them to include a copy of the virus
  - replicates and goes on to infect other content
  - easily spread through network environments
- when attached to an executable program a virus can do anything that the program is permitted to do
  - executes secretly when the host program is run
- specific to operating system and hardware
  - takes advantage of their details and weaknesses

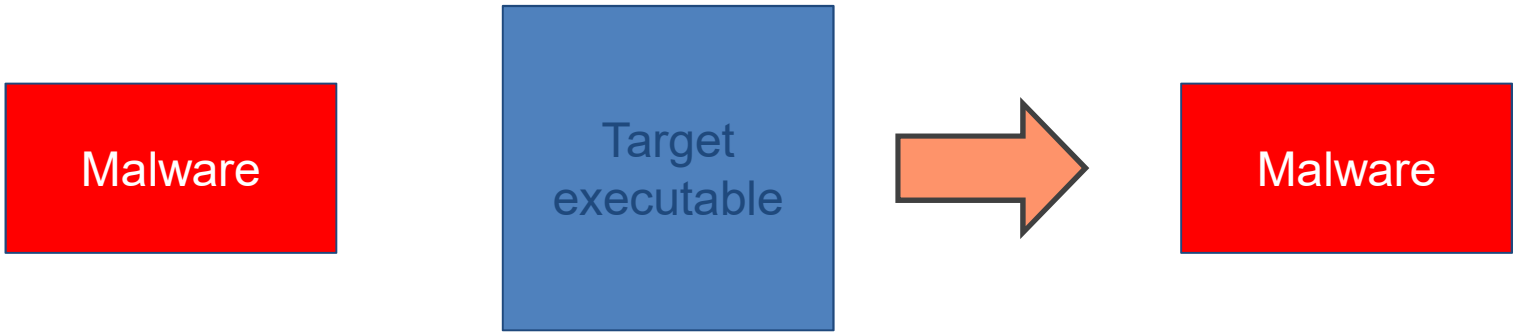
Virus Components

- Infection mechanism
  - means by which a virus spreads or propagates
  - also referred to as the infection vector
- Trigger
  - event or condition that determines when the payload is activated or delivered
  - sometimes known as a logic bomb
- Payload
  - what the virus does (besides spreading)
  - may involve damage or benign but noticeable activity

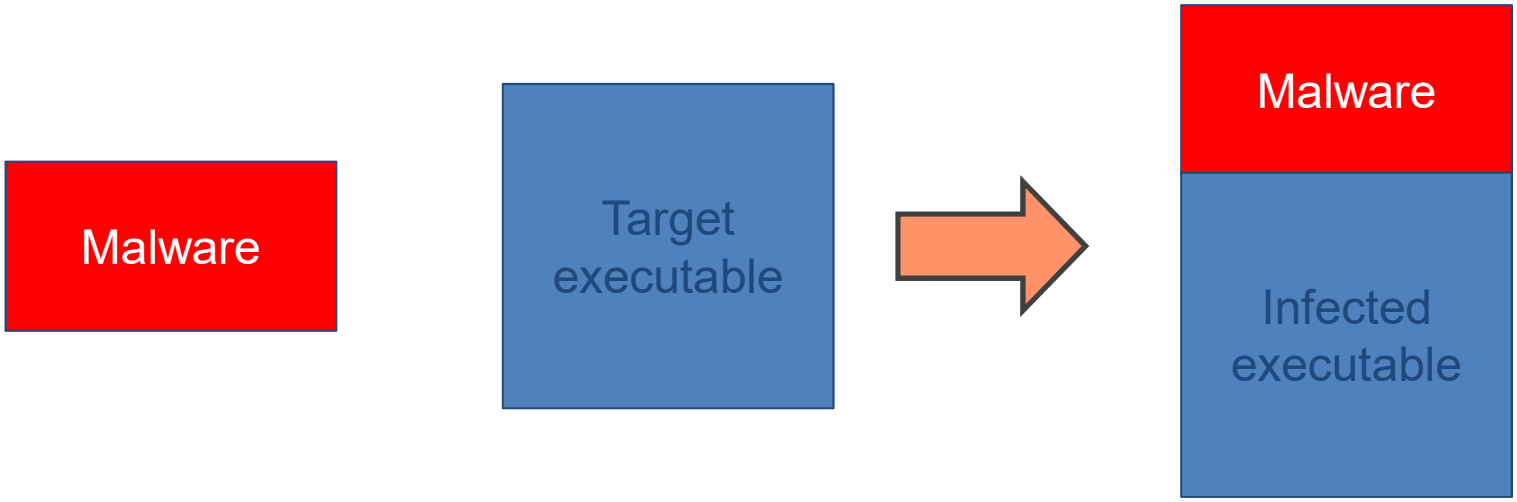
Virus Phases

- dormant phase
  - virus is idle
  - will eventually be activated by some event
  - not all viruses have this stage
- propagation phase
  - virus places a copy of itself into other programs or into certain system areas on the disk
  - may not be identical to the propagating version
  - each infected program will now contain a clone of the virus which will itself enter a propagation phase
- triggering phase
  - virus is activated to perform the function for which it was intended
  - can be caused by a variety of system events
- execution phase
  - function is performed
  - may be harmless or damaging

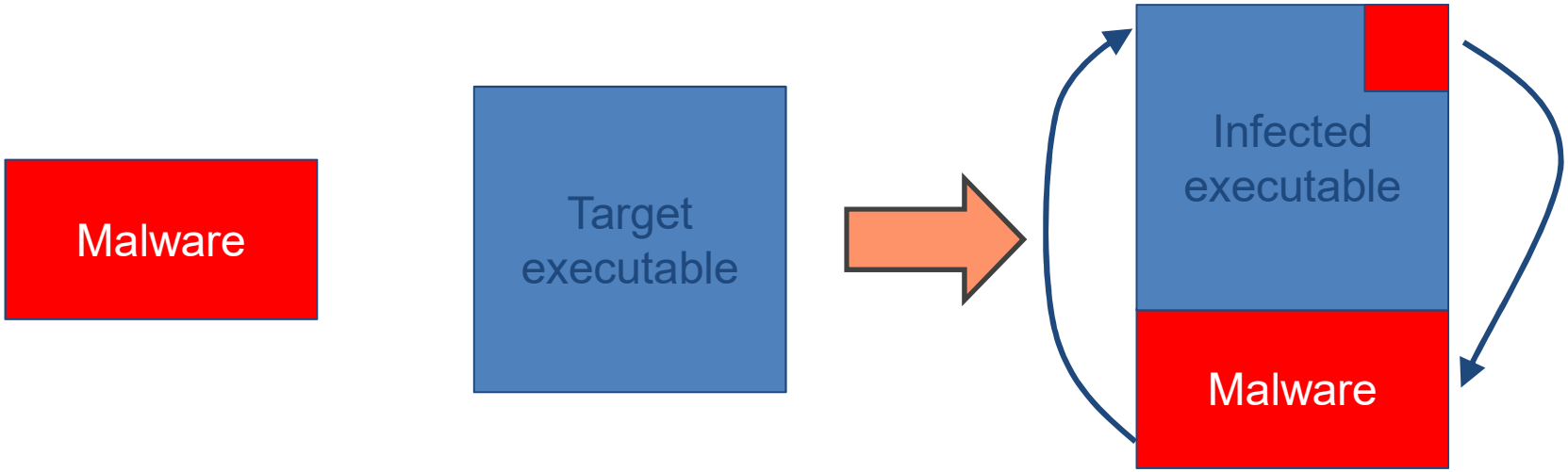
Overwriting



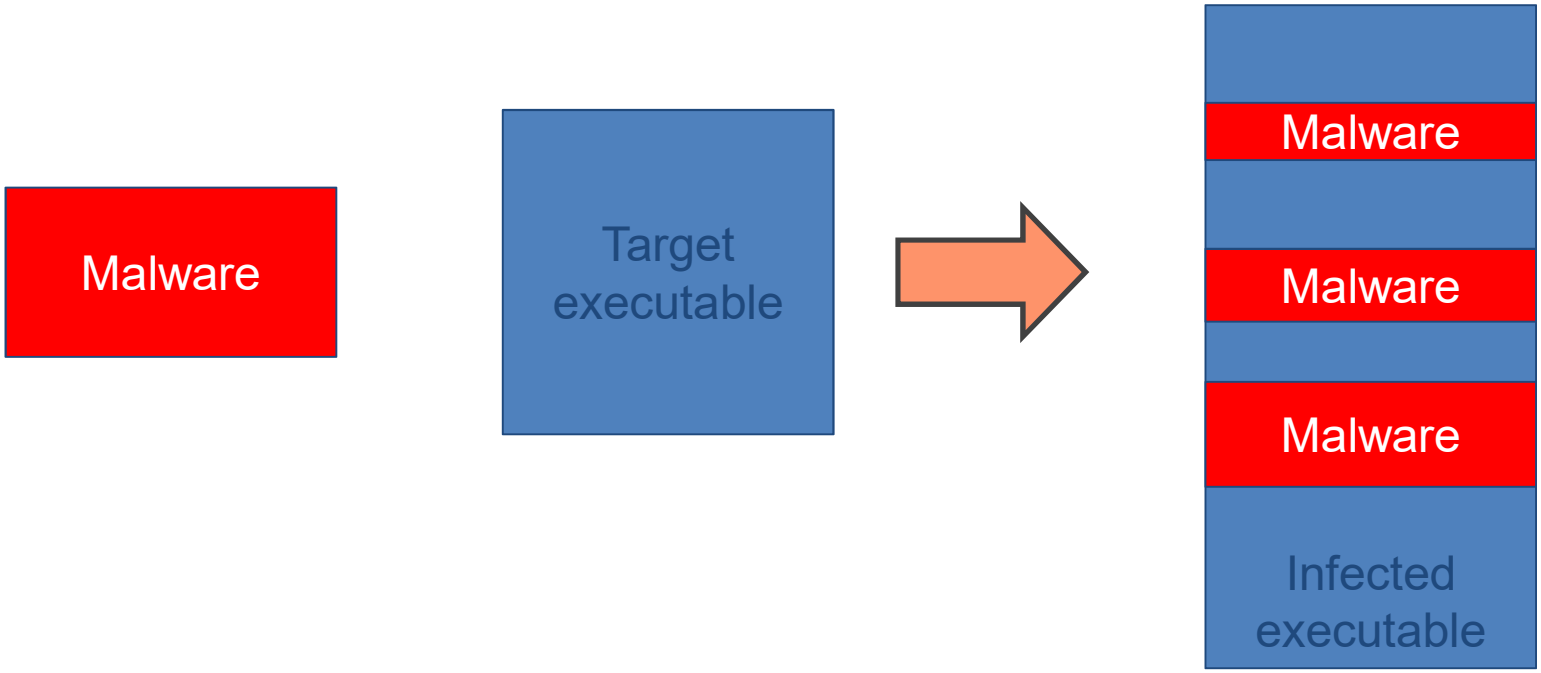
Prepending



Appending

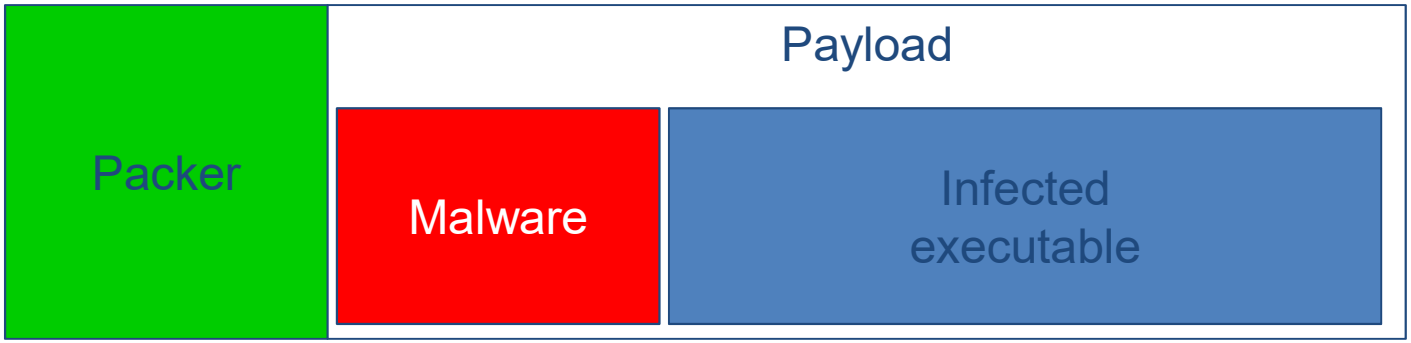


Cavity


Malware
Target executable
Malware
Malware
Malware
Infected executable

Executable packers

- Commonly used by virus writers
- Can process wide range of files: Windows PE executables, DOS executables, DOS COM files…
- Functions: compress, encrypt, randomize (polymorphism), anti-debug techniques, add-junk, anti-VM…



Virus Structure

```
    program V :=

{goto main;
    1234567;

    subroutine infect-executable :=
        {loop:
        file := get-random-executable-file;
        if (first-line-of-file = 1234567)
           then goto loop
           else prepend V to file; }

    subroutine do-damage :=
        {whatever damage is to be done}

    subroutine trigger-pulled :=
        {return true if some condition holds}

main:   main-program :=
        {infect-executable;
        if trigger-pulled then do-damage;
        goto next;}

next:

}
```

Compression Virus Logic

```
    program CV :=

{goto main;
    01234567;

    subroutine infect-executable :=
          {loop:
               file := get-random-executable-file;
          if (first-line-of-file = 01234567) then goto loop;
        (1)    compress file;
        (2)    prepend CV to file;
        }

main:   main-program :=
          {if ask-permission then infect-executable;
        (3)    uncompress rest-of-file;
        (4)    run uncompressed file;}
        }
```

Virus Classifications

classification by target

- boot sector infector
  - infects a master boot record or boot record and spreads when a system is booted from the disk containing the virus
- file infector
  - infects files that the operating system or shell considers to be executable
- macro virus
  - infects files with macro or scripting code that is interpreted by an application
- Memory
  - dynamic memory
- multipartite virus
  - infects files in multiple ways

Classification by concealment strategy

- encrypted virus
  - a portion of the virus creates a random encryption key and encrypts the remainder of the virus
- stealth virus
  - a form of virus explicitly designed to hide itself from detection by anti-virus software
- polymorphic virus
  - a virus that mutates with every infection
- metamorphic virus
  - a virus that mutates and rewrites itself completely at each iteration and may change behavior as well as appearance

NSA planted surveillance software on hard drives, report says

Security vendor Kaspersky outs a group capable of inserting spying software onto hard drives around the world, while Reuters fingers the NSA as the culprit.

In a **new report**, Kaspersky revealed the existence of a group dubbed The Equation Group capable of directly accessing the firmware of hard drives from Western Digital, Seagate, Toshiba, IBM, Micron, Samsung and other drive makers. As such, the group has been able to implant spyware on hard drives to conduct surveillance on computers around the world.

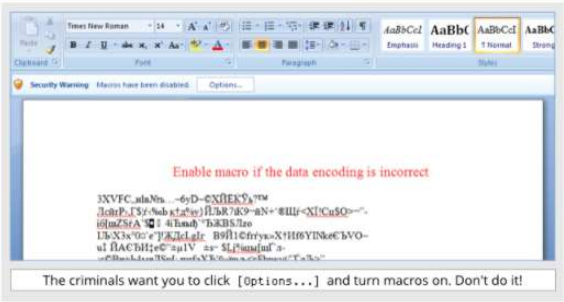
Macro/Scripting Code Viruses

- common in mid-1990s
  - platform independent
  - infect documents (not executable portions of code)
  - easily spread
- exploit macro capability of MS Office applications
  - more recent releases of products include protection
- various anti-virus programs have been developed so these are no longer the predominant virus threat

"Locky" ransomware – what you need to know

The most common way that Locky arrives is as follows:

- You receive an email containing an attached document (**Troj/DocDl-BCF**).
- The document looks like gobbledegook.
- The document advises you to enable macros "if the data encoding is incorrect."



- If you enable macros, you don't actually correct the text encoding (that's a subterfuge); instead, you run code inside the document that saves a file to disk and runs it.
- The saved file (**Troj/Ransom-CGX**) serves as a downloader, which fetches the final malware payload from the crooks.
- The final payload could be anything, but in this case is usually the Locky Ransomware (**Troj/Ransom-CGW**).

Locky scrambles all files that match a long list of extensions, including videos, images, source code, and Office files.

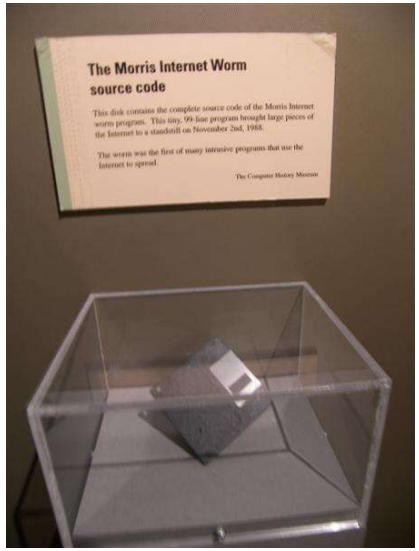
Locky even scrambles `wallet.dat`, your Bitcoin wallet file, if you have one.

E-Mail Viruses

- more recent development
- e.g. Melissa
  - exploits MS Word macro in attached doc
  - if attachment opened, macro activates
  - sends email to all on users address list
  - and does local damage
- hence much faster propagation

Worms

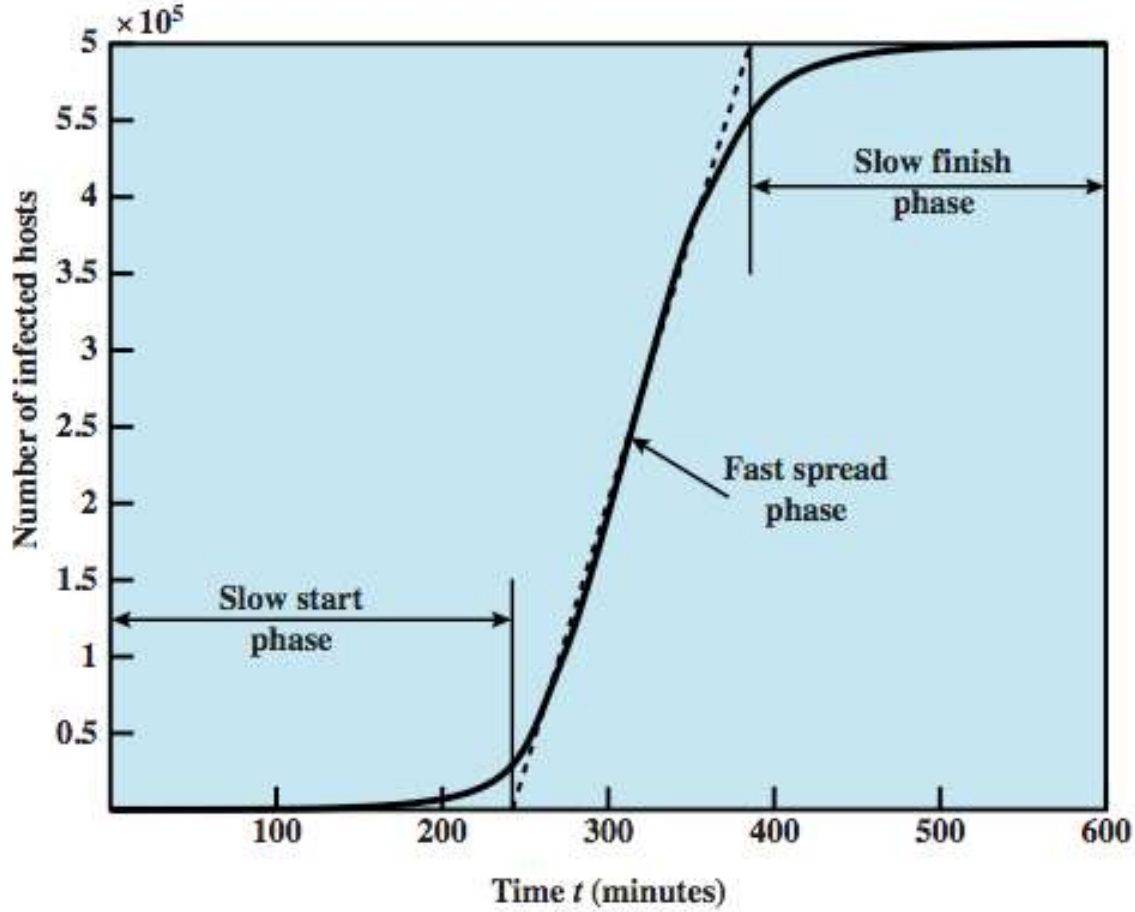
- program that actively seeks out more machines to infect and each infected machine serves as an automated launching pad for attacks on other machines
  - exploits software vulnerabilities in client or server programs
  - can use network connections to spread from system to system
  - spreads through shared media (USB drives, CD, DVD data disks)
  - e-mail worms spread in macro or script code included in attachments and instant messenger file transfers
- upon activation the worm may replicate and propagate again
- usually carries some form of payload
- first known implementation was done in Xerox Palo Alto Labs in the early 1980s
- Morris Worm 1988



Worm Replication

- electronic mail or instant messenger facility
  - worm e-mails a copy of itself to other systems
  - sends itself as an attachment via an instant message service
- file sharing
  - creates a copy of itself or infects a file as a virus on removable media
- remote execution capability
  - worm executes a copy of itself on another system
- remote file access or transfer capability
  - worm uses a remote file access or transfer service to copy itself from one system to the other

Worm Propagation Model



Known Worm Attacks

Melissa	1998	e-mail worm first to include virus, worm and Trojan in one package
Code Red	July 2001	exploited Microsoft IIS bug probes random IP addresses consumes significant Internet capacity when active
Code Red II	August 2001	also targeted Microsoft IIS installs a backdoor for access
Nimda	September 2001	had worm, virus and mobile code characteristics spread using e-mail, Windows shares, Web servers, Web clients, backdoors
SQL Slammer	Early 2003	exploited a buffer overflow vulnerability in SQL server compact and spread rapidly
Sobig.F	Late 2003	exploited open proxy servers to turn infected machines into spam engines
Mydoom	2004	mass-mailing e-mail worm installed a backdoor in infected machines
Warezov	2006	creates executables in system directories sends itself as an e-mail attachment can disable security related products
Conficker (Downadup)	November 2008	exploits a Windows buffer overflow vulnerability most widespread infection since SQL Slammer
Stuxnet	2010	restricted rate of spread to reduce chance of detection targeted industrial control systems

SQL Slammer

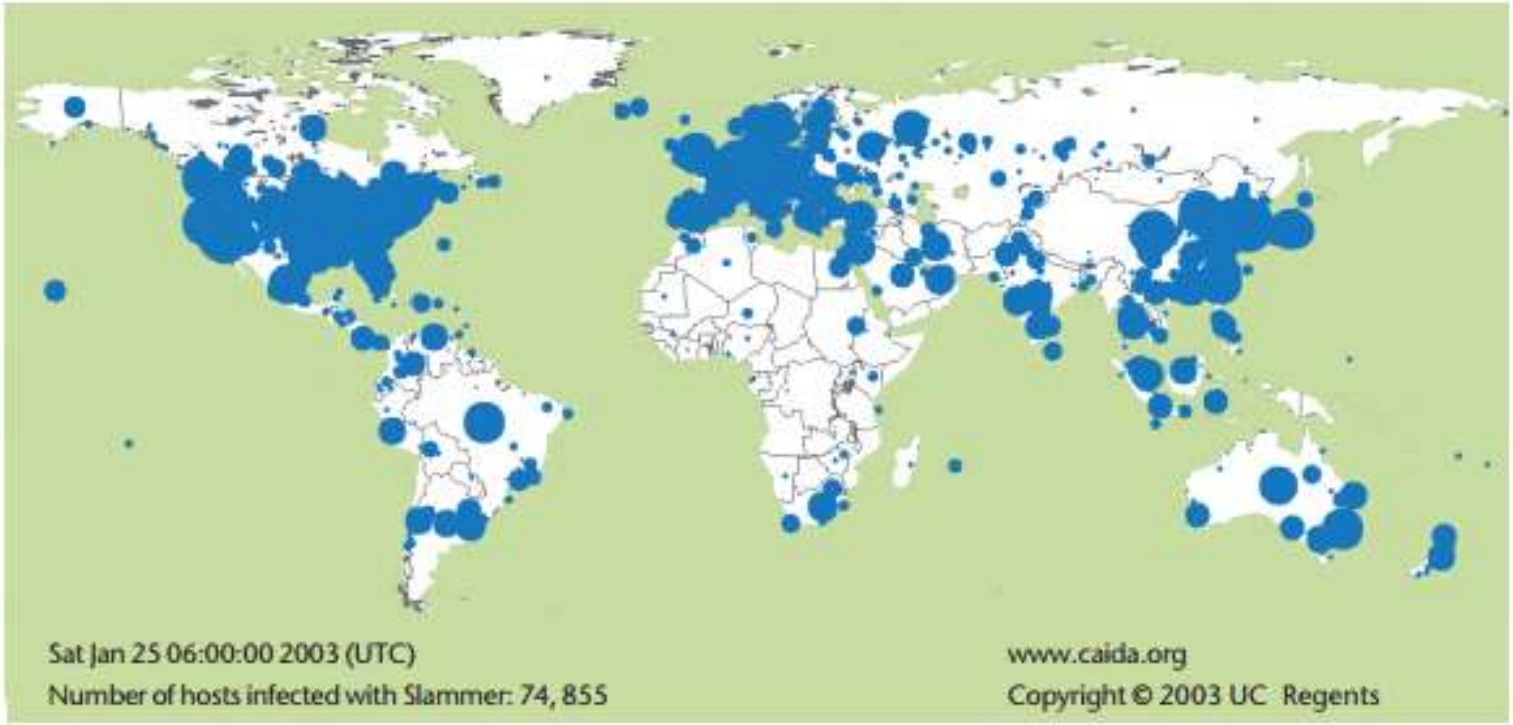


Figure 1. The geographical spread of Slammer in the 30 minutes after its release. The diameter of each circle is a function of the logarithm of the number of infected machines, so large circles visually underrepresent the number of infected cases in order to minimize overlap with adjacent locations. For some machines, we can determine only the country of origin rather than a specific city.

Mobile Code

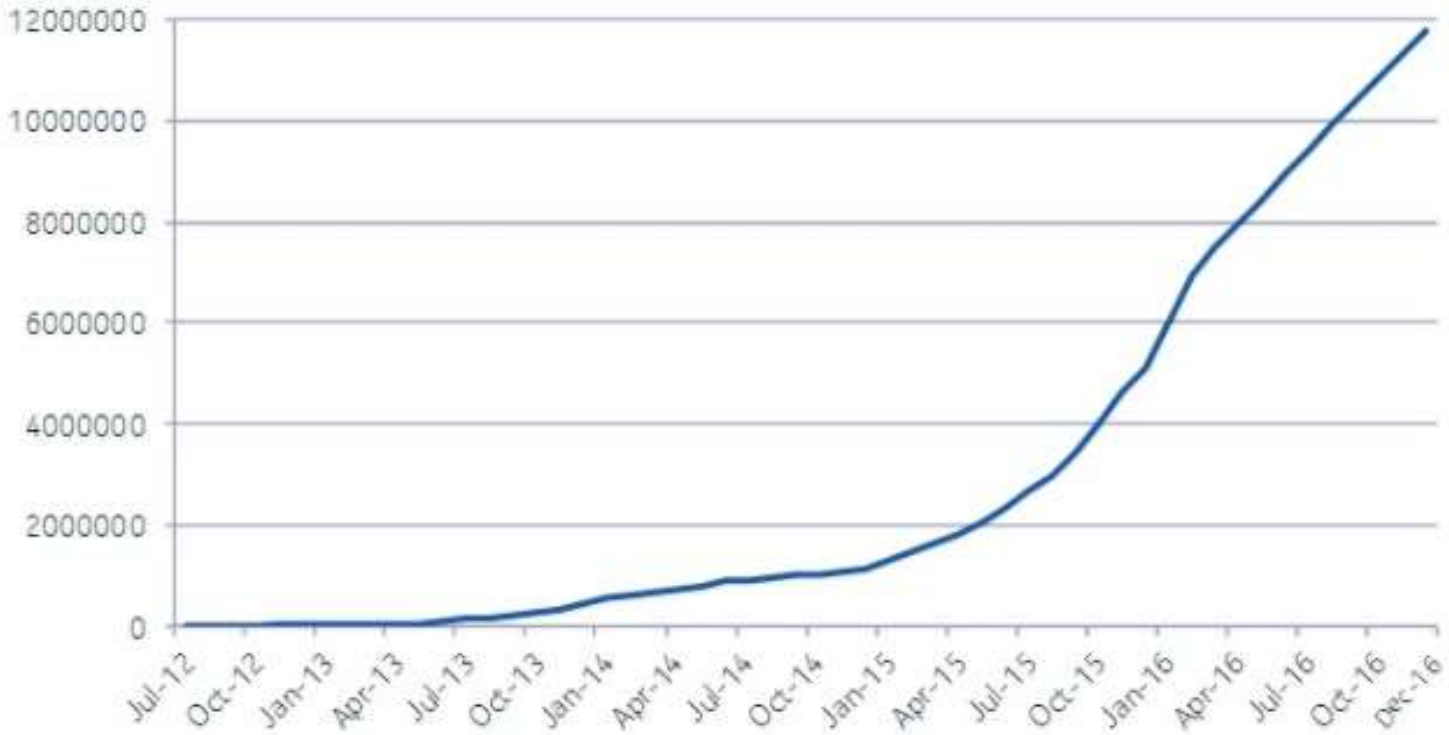
- programs that can be shipped unchanged to a variety of platforms
- transmitted from a remote system to a local system and then executed on the local system
- often acts as a mechanism for a virus, worm, or Trojan horse
- takes advantage of vulnerabilities to perform it own exploits
- popular vehicles include Java applets, ActiveX, JavaScript and VBScript

Mobile Phone Malware

- first discovery was Cabir worm in 2004
- then Lasco and CommWarrior in 2005
- communicate through Bluetooth wireless connections or MMS
- target is the smartphone
- can completely disable the phone, delete data on the phone, or force the device to send costly messages
- CommWarrior replicates by means of Bluetooth to other phones, sends itself as an MMS file to contacts and as an auto reply to incoming text messages

Mobile Phone Malware

Figure 4. Mobile malware samples since July 2012 through to December 2016



Source: Nokia Threat Intelligence Report

Drive-By-Downloads

- exploits browser vulnerabilities to download and install malware on the system when the user views a Web page controlled by the attacker
- in most cases does not actively propagate
- spreads when users visit the malicious Web page

Social Engineering

- "tricking" users to assist in the compromise of their own systems
- spam
  - unsolicited bulk e-mail
  - significant carrier of malware
  - used for phishing attacks
- Trojan horse
  - program or utility containing harmful hidden code
  - used to accomplish functions that the attacker could not accomplish directly
- mobile phone Trojans
  - first appeared in 2004 (Skuller)
  - target is the smartphone

Payload System Corruption

- data destruction
  - Chernobyl virus
    - first seen in 1998
    - Windows 95 and 98 virus
    - infects executable files and corrupts the entire file system when a trigger date is reached
  - Klez
    - mass mailing worm infecting Windows 95 to XP systems
    - on trigger date causes files on the hard drive to become empty
  - ransomware
    - encrypts the user's data and demands payment in order to access the key needed to recover the information
    - PC Cyborg Trojan (1989)
    - Gpcode Trojan (2006)
    - WannaCry (2017)

Payload System Corruption

- real-world damage
  - causes damage to physical equipment
    - Chernobyl virus rewrites BIOS code
  - Stuxnet worm
    - targets specific industrial control system software
    - there are concerns about using sophisticated targeted malware for industrial sabotage
  - logic bomb: code embedded in the malware that is set to "explode" when certain conditions are met

Botnets

- **Bots** – a malicious software that runs autonomously and automatically on a compromised computer (zombie) without owner's consent to perform some malicious/disturbing activity; e.g., SPAM, DoS…

- **Botnet** (Bot army) – controlled by a master via some Command & Control (C&C) channel

Botnets

- takes over another Internet attached computer and uses that computer to launch or manage attacks
- botnet - collection of bots capable of acting in a coordinated manner
- uses:
  - distributed denial-of-service (DDoS) attacks
  - spamming
  - sniffing traffic
  - keylogging
  - spreading new malware
  - installing advertisement add-ons and browser helper objects (BHOs)
  - attacking IRC chat networks
  - manipulating online polls/games

Remote control

- distinguishes a bot from a worm
  - worm propagates itself and activates itself
  - bot is initially controlled from some central facility
- previous typical means of implementing the remote control facility is on an IRC server
  - bots join a specific channel on this server and treat incoming messages as commands
  - more recent botnets use covert communication channels via protocols such as HTTP
  - distributed control mechanisms use peer-to-peer protocols to avoid a single point of failure

Remote Control Facility

- Encrypted services
- DNS
- Websites

Russian State Hackers Use Britney Spears Instagram Posts to Control Malware

By Catalin Cimpanu June 6, 2017 01:50 PM 0

- Bridging airgap

Payload – Information Theft Keyloggers and Spyware

- Keylogger
  - captures keystrokes to allow attacker to monitor sensitive information
  - typically uses some form of filtering mechanism that only returns information close to keywords ("login", "password")
- Spyware
  - subverts the compromised machine to allow monitoring of a wide range of activity on the system
  - monitoring history and content of browsing activity
  - redirecting certain Web page requests to fake sites
  - dynamically modifying data exchanged between the browser and certain Web sites of interest

Payload – Information Theft Phishing

- exploits social engineering to leverage the user's trust by masquerading as communication from a trusted source
  - include a URL in a spam e-mail that links to a fake Web site that mimics the login page of a banking, gaming, or similar site
  - suggests that urgent action is required by the user to authenticate their account
  - attacker exploits the account using the captured credentials
- spear-phishing
  - recipients are carefully researched by the attacker
  - e-mail is crafted to specifically suit its recipient, often quoting a range of information to convince them of its authenticity

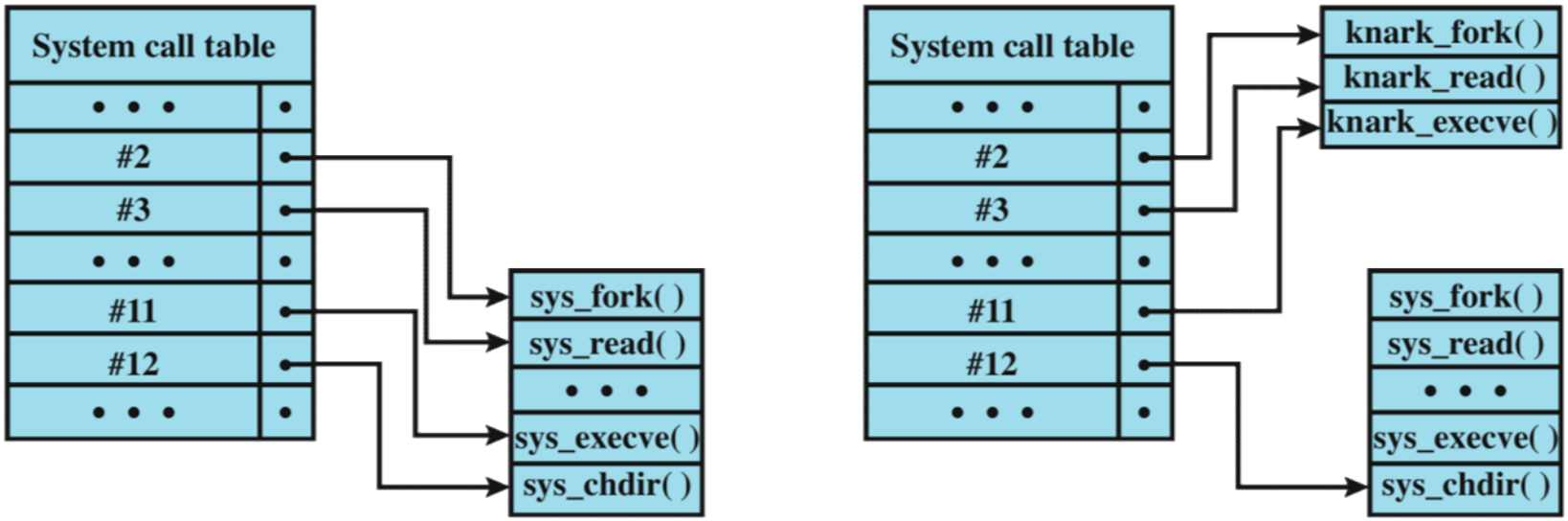
Payload – Stealthing Backdoor

- also known as a trapdoor
- secret entry point into a program allowing the attacker to gain access and bypass the security access procedures
- maintenance hook is a backdoor used by programmers to debug and test programs
- difficult to implement operating system controls for backdoors in applications

Payload - Stealthing Rootkit

- stealth software component that maintains a persistent and undetectable presence on the machine
- set of hidden programs installed on a system to maintain covert access to that system
- hides by subverting the mechanisms that monitor and report on the processes, files, and registries on a computer
- gives administrator (or root) privileges to attacker
  - can add or change programs and files, monitor processes, send and receive network traffic, and get backdoor access on demand

System Call Table Modification

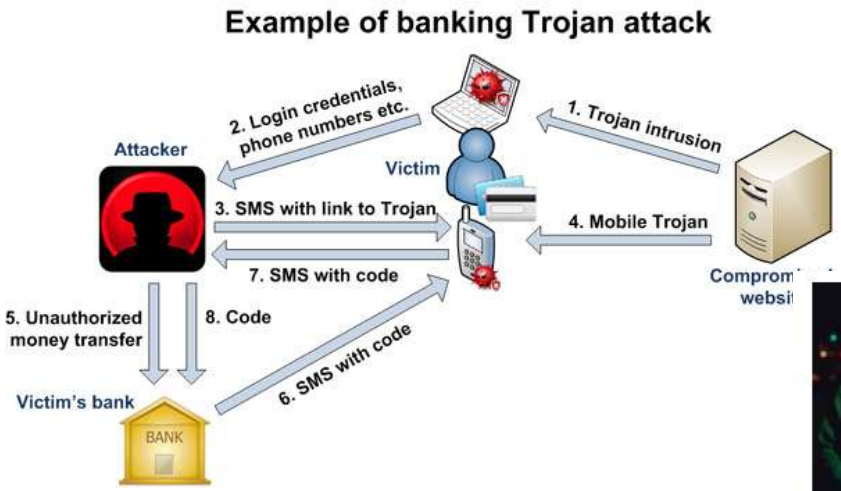


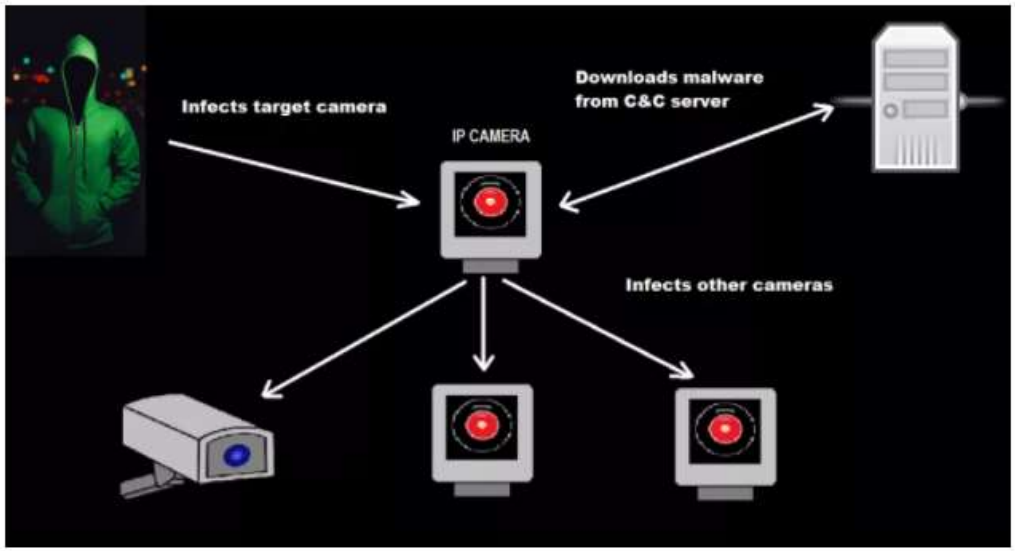
(a) Normal kernel memory layout

(b) After nkark install

Figure 6.5 System Call Table Modification by Rootkit (based on [LEVI06])

Advances malware/attacks





Virus Countermeasures

- ideal solution to the threat of malware is prevention
- four main elements of prevention:
  - policy
  - awareness
  - vulnerability mitigation
  - threat mitigation

- prevention fails, technical mechanisms can be used to support the following threat mitigation options:
  - detection
  - identification
  - removal

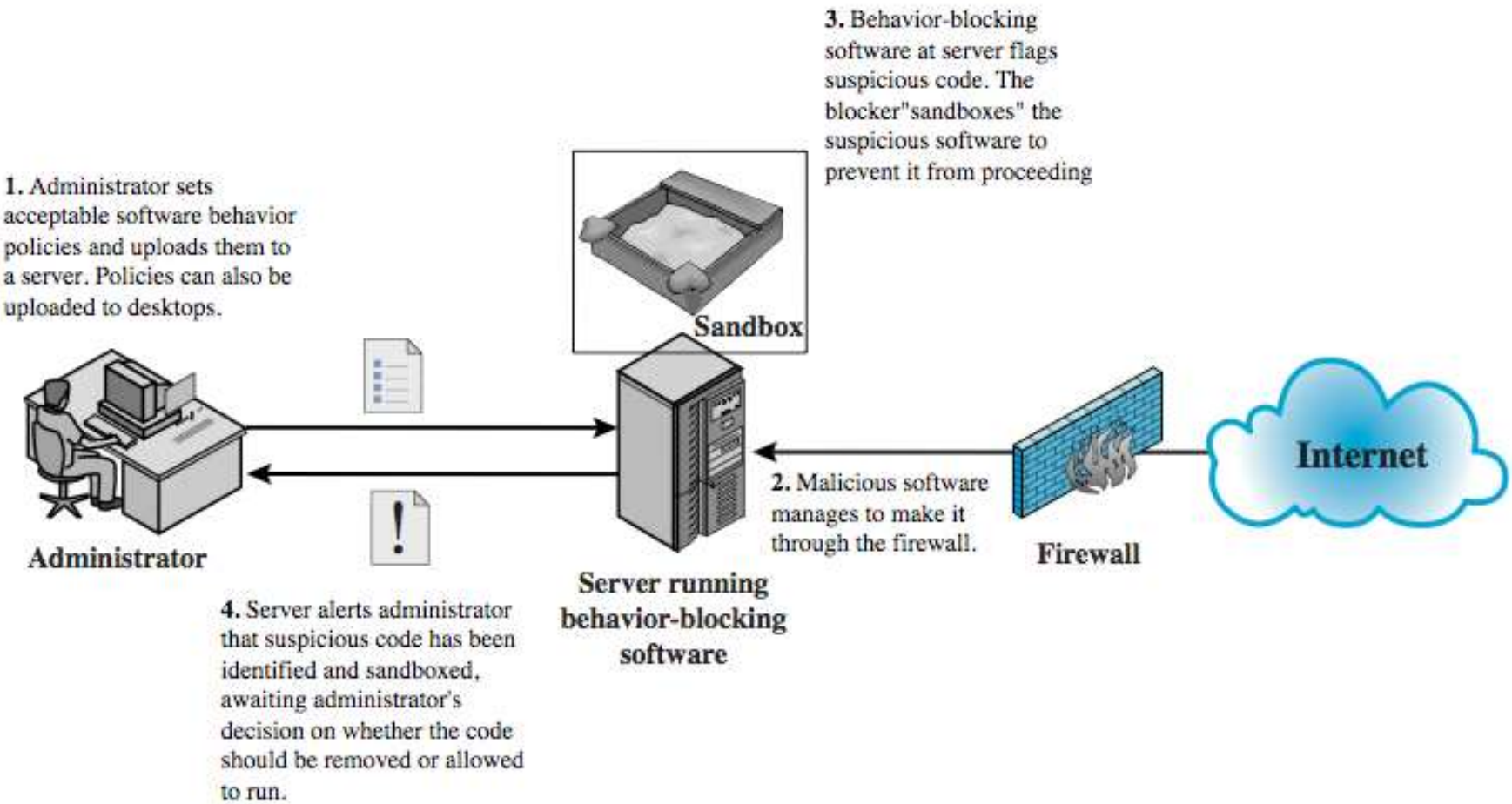
Anti-Virus Evolution

- virus & antivirus tech have both evolved
- early viruses simple code, easily removed
- as become more complex, so must the countermeasures
- generations
  - first - signature scanners
  - second - heuristics
  - third - identify actions
  - fourth – full featured solution

Generic Decryption

- runs executable files through GD scanner:
  - CPU emulator to interpret instructions
  - virus scanner to check known virus signatures
  - emulation control module to manage process
- lets virus decrypt itself in interpreter
- periodically scan for virus signatures
- issue is long to interpret and scan
  - tradeoff chance of detection vs time delay

Behavior-Blocking Software



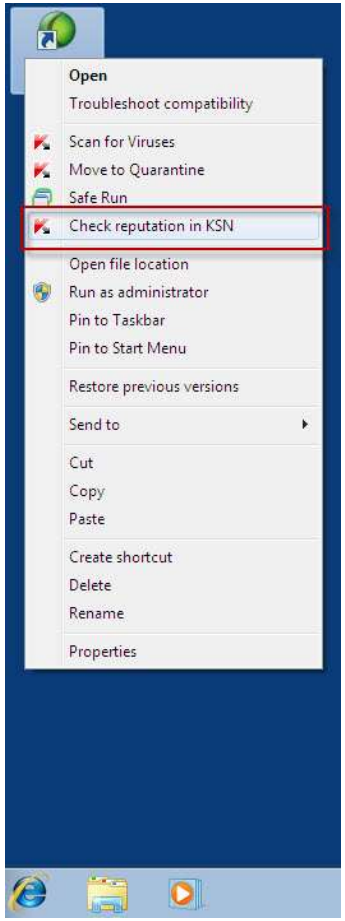
Host-Based Behavior-Blocking Software

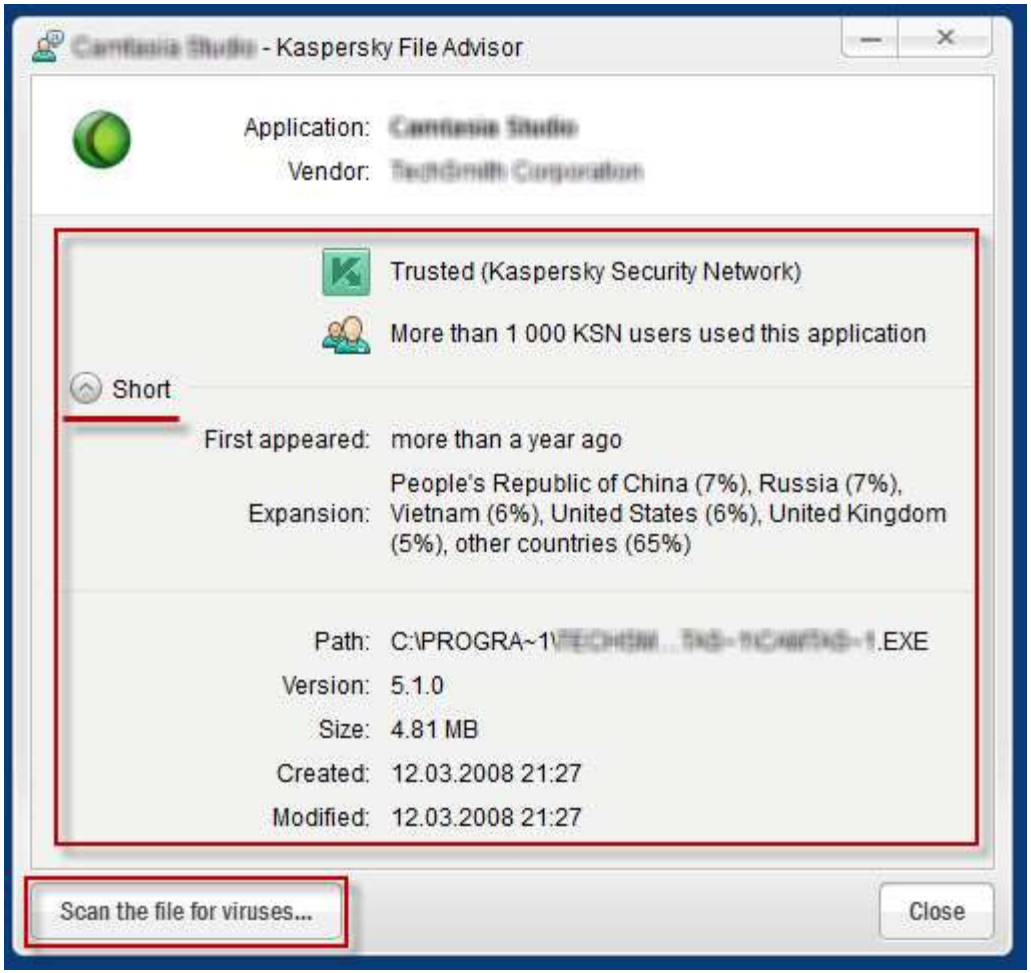
- integrates with the operating system of a host computer and monitors program behavior in real time for malicious action
  - blocks potentially malicious actions before they have a chance to affect the system
  - blocks software in real time so it has an advantage over anti-virus detection techniques such as fingerprinting or heuristics
- limitations
  - because malicious code must run on the target machine before all its behaviors can be identified, it can cause harm before it has been detected and blocked

Detection techniques

- Integrity check
- Static analysis
  - Signature-based
  - Behavioral analysis
- Dynamic behavioral analysis
- Reputation of files

Reputation-based





Static analysis

- Leverage structural information (e.g., sequence of bytes)
- Attempts to detect malware before the program under inspection executes
- Information about the program or its expected behavior consists of explicit and implicit observations in its binary/source code
- Implemented using the *signature-based* method which relies on the identification of unique strings in the binary code
  - useless against unknown malicious code
- Behavioral/Heuristic static analysis

Dynamic analysis

- leverage runtime information (e.g., network usage, files and memory modifications, system calls)
- attempts to detect malicious behavior during program execution or after program execution
- Difficult to simulate the appropriate conditions in which the malicious functions of the program, such as the vulnerable application that the malware exploits, will be activated
- It is not clear what is the required period of time needed to observe the appearance of the malicious activity for each malware

Goals

- Clean network traffic from known malware
- Clean network traffic from unknown malware
- Detect malware at the host
- Rank unknown files by level of suspicion
- Generate unique signature – automatic signature generation (ASG)

On the effectiveness of malware protection on Android

Test Case 1: Altered Malware

	1	2	3	4	5	6	7	8	9	10	Total
avast	✓	–	✓	✓	–	✓	✓	–	✓	–	6/10
AVG	–	–*	–	✓	–	✓	–*	–	–	–	2/10
BitDefender	✓	–	–	✓	–	✓	✓	–	–	–	4/10
ESET	✓	✓	✓	✓	✓	✓	✓	✓	✓	–	9/10
F-Secure	✓	–	✓	✓	–	–	✓	✓	✓	–	6/10
Kaspersky	✓	–	–	✓	–	–	✓	–	–	–	3/10
Lookout	✓	–	–	✓	–	✓	✓	✓	✓	✓	7/10
McAfee	✓	–	–	✓	–	–	–	–	–	–	2/10
Norton	✓	–	–	✓	–	✓	–	–	–	–	3/10
Sophos	–	–	–	–	–	–	–	–	–	–	0/10
Trend Micro	✓	–	–	✓	–	–	–	–	✓	–	3/10

Test Case 3: Altered Root Exploits

	Exploid	GingerBreak	RageAgainstTheCage	Total
avast	–	✓	–	1/3
AVG	–	–	–	0/3
BitDefender	–	–	–	0/3
ESET	–	–	–	0/3
F-Secure	–	–	–	0/3
Kaspersky	–	–	–	0/3
Lookout	✓	✓	–	2/3
McAfee	–	–	–	0/3
Norton	–	–	–	0/3
Sophos	–	–	–	0/3
Trend Micro	–	–	–	0/3

Test Case 6: Unknown Malware

	Dropper	Core Malware	USB Infection Tool	Total
avast	–	–	–	0/3
AVG	–	–	–	0/3
BitDefender	–	–	–	0/3
ESET	–	–	–	0/3
F-Secure	–	–	–	0/3
Kaspersky	–	–	–	0/3
Lookout	–	–	–	0/3
McAfee	–	–	–	0/3
Norton	–	–	–	0/3
Sophos	–	–	–	0/3
Trend Micro	–	–	–	0/3

Static analysis
Portable Executable (PE) data

- Windows OS Portable Executable
  - File format for executables, object code, sys drivers, DLLs
  - Contains information necessary for the Windows OS loader to manage the executable code
- Data is extracted from certain parts of .exe files stored in Win32 PE binaries
- PE Header that describes physical structure of a PE binary (e.g., creation/modification time, machine type, file size)
  - Import Section: which DLLs were imported and which functions from which imported DLLs were used
  - Exports Section: which functions were exported (if the file being examined is a DLL)
  - Resource Directory: resources used by a given file (e.g., pics)
  - Version Information (e.g., internal and external name of a file, version number)

Static analysis
Portable Executable (PE) data

PE Explorer - C:\Program Files\Movie Maker\moviemk.exe

File View Tools Help

HEADERS INFO

Address of Entry Point: 0115FF8B Real Image Checks

Field Name	Data Value	Description
Machine	014Ch	i386®
Number of Sections	0004h	
Time Date Stamp	48025298h	13/04/2008 18:36:08
Pointer to Symbol Table	00000000h	
Number of Symbols	00000000h	
Size of Optional Header	00E0h	
Characteristics	010Fh	
Magic	010Bh	
Linker Version	0A07h	
Size of Code	002CC200h	
Size of Initialized Data	0009BA00h	
Size of Uninitialized Data	00000000h	
Address of Entry Point	0115FF8Bh	
Base of Code	00001000h	
Base of Data	002CE000h	
Image Base	01000000h	

PE Explorer Dependency Scanner - C:\Program Files\Movie Maker\moviemk.exe

File Help

moviemk.exe
- msvcrt.dll
- advapi32.dll
- kernel32.dll
- gdi32.dll
- user32.dll
- comdlg32.dll

Path: C:\WINDOWS\system32\comdlg32.dll

Version Info

```
Info:            VS_VERSION_INFO
Signature:       FEEF04BDh
Struc Version:   1.0
File Version:    6.0.2900.5512
```

PE Explorer - C:\Program Files\Movie Maker\moviemk.exe

File View Tools Help

RESOURCE EDITOR

- RT_JPG
  - SAMPLE1
  - SAMPLE2
- RT_PNG
  - NEWS
  - PAINT
- RT_PRX
- RT_XML
- TYPELIB
- Icon Entry
- Dialog
- String
  - 2049
  - 2050
- Group Icon
- Version
  - 1
- Manifest

```
Length Of Struc: 03C4h
Length Of Value: 0034h
Type Of Struc:   0000h
Info:            VS_VERSION_INFO
Signature:       FEEF04BDh
Struc Version:   1.0
File Version:    2.1.4026.0
Product Version: 2.0.0.0
File Flags Mask: 0.63
File Flags:
File OS:         WINDOWS32
File Type:       DLL
File SubType:    UNKNOWN
File Date:       00:00:00  00/00/0000

     Struc has Child(ren). Size: 872 bytes.

Child Type:          StringFileInfo
Language/Code Page:  1033/1200
Comments:
```

Malware code detection as text categorization (opcode n-grams)

- Disassembles executables into assembly instructions (e.g., by using IDA-Pro)
- Creates sequences (n-grams) of operation codes

```
.text:004014D1    push    0                ; hTemplateFile
.text:004014D3    push    80h              ; dwFlagsAndAttributes
.text:004014D8    push    3                ; dwCreationDisposition
.text:004014DA    push    0                ; lpSecurityAttributes
.text:004014DC    push    1                ; dwShareMode
.text:004014DE    push    80000000h        ; dwDesiredAccess
.text:004014E3    mov     eax, [ebp+arg_4]
.text:004014E6    push    dword ptr [eax] ; lpFileName
.text:004014E8    call    CreateFileA
.text:004014ED    mov     edi, eax
```

Shabtai et al, Detecting Unknown Malicious Code by Applying Classification Techniques on OPCODEs Representation, Security Informatics, 2012

Malware code detection as text categorization opcode n-grams

```
.text:00DCC7B0  55 8B EC 83 E4 F8 81 EC-BC 02 00 00 53 56 57 E8  "U1°_T_J__ǂ..SVW_"
.text:00DCC7C0  EC EE FF FF 33 DB 38 1D-86 38 3F 01 0F 85 AF 02  "__  3_8ІҐІ?8Ґ»ǂ "
.text:00DCC7D0  00 00 38 1D 88 38 3F 01-0F 85 A3 02 00 00 E8 BD  "..8ІҐІ?8D_ǂ.__ "
.text:00DCC7E0  A0 71 FF E8 58 05 00 00-E8 03 04 00 00 8B 75 10  "_q _X?.._ǂ?..Juϑ"
.text:00DCC7F0  E8 8B FC FF FF 85 C0 0F-85 AA 02 00 00 E8 4E FD  "_ІҐ_І  _J¬ǂ.._N²"
.text:00DCC800  FF FF E8 E9 DF FF FF 84-C0 0F 84 62 02 00 00 E8  "  ___  ПҐ_Пbǂ.._"
.text:00DCC810  6C 03 00 00 3B C3 74 23-38 1E 0F 84 51 02 00 00  "1ǂ..;_t#8ҐПQǂ.. "
```

Disassembly

```
push    ebp
mov     ebp, esp
and     esp, 0FFFFFFF8h
sub     esp, 2BCh
push    ebx
```

Extracting sequences of OpCodes

OpCode 1-gram	push (2)	mov (1)	and (1)	sub (1)
OpCode 2-gram	push,mov (1)	mov,and (1)	and,sub (1)	sub,push (1)
OpCode 3-gram	push,mov,and (1)	mov,and,sub (1)	and,sub,push (1)	

Static analysis
Strings

- String features are based on plain-text strings that are encoded in programs files
- Examples: “windows”, “kernel”, “reloc” etc)
- Possibly can also be used to represent files similarly to the text categorization problem
- May reveal useful information: IRC Commands, SMTP commands, registry keys

Detection Using Machine Learning

File	55 8B EC	E4 F8 81	…	3F 01 0F	Class
File1	1	1		0	Benign
File2	1	0		0	Benign
…					…
	0	0		1	Malware
Filen	0	1		1	Malware

Basic dynamic analysis

- Sandboxes
  - Run in an isolated virtual environment
  - Various flavors with various features
  - Users' PCs
- Virtual machines
  - Pretty safe if no network connection
  - Can also generate fake networks
  - Snapshots
- Honeypots
  - Carefully control all traffic through a honeywall

Malware signatures classification

- **Vulnerability-based signature**
  - describes the properties of a certain bug in the system
  - do not attempt to detect every malicious code
  - can be very effective when dealing with polymorphic malware
  - can be generated only when the vulnerability is discovered
- **Exploit-based signature**
  - describes a piece of code (sequence of commands or inputs) triggered by the malware which actually exploits a vulnerability in the system
  - usually focus on analyzing similarities in network traffic or sequence of commands
- **Payload-based signatures**
  - actual body of the malware

Virus Scanners and Signatures

- Common AV identify the presence of a virus in an exe file, a boot record or memory by using short signatures which consist of sequences of bytes
- Good signature
  - Found in every object infected by the virus
  - Unlikely to be found if the virus is not present
- Typically, a human expert chooses a signature for a new virus by means of a time consuming procedure
  - Converting the binary code of the virus to Assembler
  - Analyzing the assembler code and picking sections of code that appear to be unusual
  - Identifying the corresponding bytes in the machine code