SLIIT

Discover Your Future

IE2022 – Introduction to Cyber Security

Lecture - 11

Malicious Software (Malware)

Mr. Amila Senarathne

Reading Assignment

- W. Stallings and L. Brown, “Computer Security, Principles and Practice, 2nd edition, Pearson, 2012, Chapter 6.

Topics to be discussed

- Definition of malwares
- Malware propagation
- Malware payloads
- Malware countermeasures
- Malware detection mechanisms

Malicious Software (Malware)

Malicious Software (Malware) is defined by NIST as:

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.

–Malware is one of the most significant threats to computer systems

- Application programs
- Utility program (editor, compiler)
- Kernel program
- Websites and server
- Spam emails to trick users, etc

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.

Table 6.1 lists the common malware terminology used throughout Chapter 6 : From Stallings & Brown textbook

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.
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.

Table 6.1 lists the common malware terminology used throughout Chapter 6 : From Stallings & Brown textbook

Malware Classification

- There is no universally accepted classification
  - One classification is based on:
    - How malware first spreads/propagates to reach its target
    - The payloads/actions malware performs on the target
  - Other malware classification:
    - Parasitic software that needs a host program (e.g., virus)
    - Self contained software (e.g., worms, trojans)
    - Malware that do not replicate (trojans, email spam)
    - Malware that replicates (virus, worms)
- Malware can be created using a crimeware – a toolkit that can be used to create malware with various propagation and payload
  - e.g., Zeus Crimeware Toolkit
- The source of malware can be individuals and organizations

Malware Propagation/Payloads

Propagations:

- By infection - Infecting existing program that spread to other system (e.g., virus)
- By exploiting vulnerability - Attacking software vulnerabilities that allow malware to be downloaded/spread, e.g., worm
- By social engineering - Tricking users to install the malware (trojan, phishing)

Payloads:

- Corrupting host systems and data
- Stealing system resources/service to make it zombie/botnet
- Stealing system information (login, password, other personal details)
- Stealthing – hiding within the host system to avoid detection

Propagation by Infection

- Virus: a fragment of program that attaches to some executable code
  - First appeared in early 1980's
  - The name is by Fred Cohen
- Virus modifies existing program with a routine to replicate the virus code to go infecting other content
- The program fragment can be:
  - Machine code that infects existing programs
  - Scripting code that is used to support data files in MS Words, Excels, and Adobe PDF
- The operations that the virus can do depends on the rights of the program it is attached to:
  - It operates secretly when the host program runs
  - It can erase files/programs

Virus components and phases

- Virus has three components (Aycock, J, 2006):
  - Infection mechanism:– the tool for the virus to propagate
    - also called infection vector
  - Trigger: when the payload is activated
    - also called logic bomb
  - Payload: what the virus does
- Virus has four phases in its lifetime:
  - Dorman phase: the virus is idle
    - It will eventually be activated by some events
  - Propagation phase: the virus put a copy of itself into other program or disk
    - the copy may or may not be identical to avoid detection
  - Triggering phase: the virus is activated to perform its intended function
  - Execution phase: the function is performed
    - It can be harmless but annoying or damaging

Executable Virus

- A machine executable code virus can be
  - Pre-pended or post-pended to some executable program
  - Embedded into some executable program
- The infected program will first execute the virus code before executing the original program
- The general virus structure is shown as follows (Fig. 6.3, textbook; also from Cohen 94)
  - It can be easily detected since the infected program is longer (in bytes) than the original
  - A simple way to avoid easy detection is by compressing the code so that both infected and original program are the same size (See Figure 6.2, textbook)

Simple virus – example-1 (from Stallings & Brown)

```
    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:

}
```

Figure 6.1 A Simple Virus

Virus logic with compression

Consider a program P1 infected with virus CV and an uninfected program P2. Assume P1+CV becomes P1'.

When P1 is executed, its virus will do the following:

1) Compress P2 to create P2' such that the size of P2'+CV=P2
2) Prepend a copy of CV to P2'
3) Uncompress P1'
4) P1 is executed.

- This example shows how the virus propagate undetected.
- Virus infection can be avoided if the virus can be blocked from entering the system.

Simple virus – example-1 (from Stallings & Brown)

```
    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;}
       }
```

Figure 6.2 Logic for a Compression Virus

Compression –operations (from Stallings & Brown)

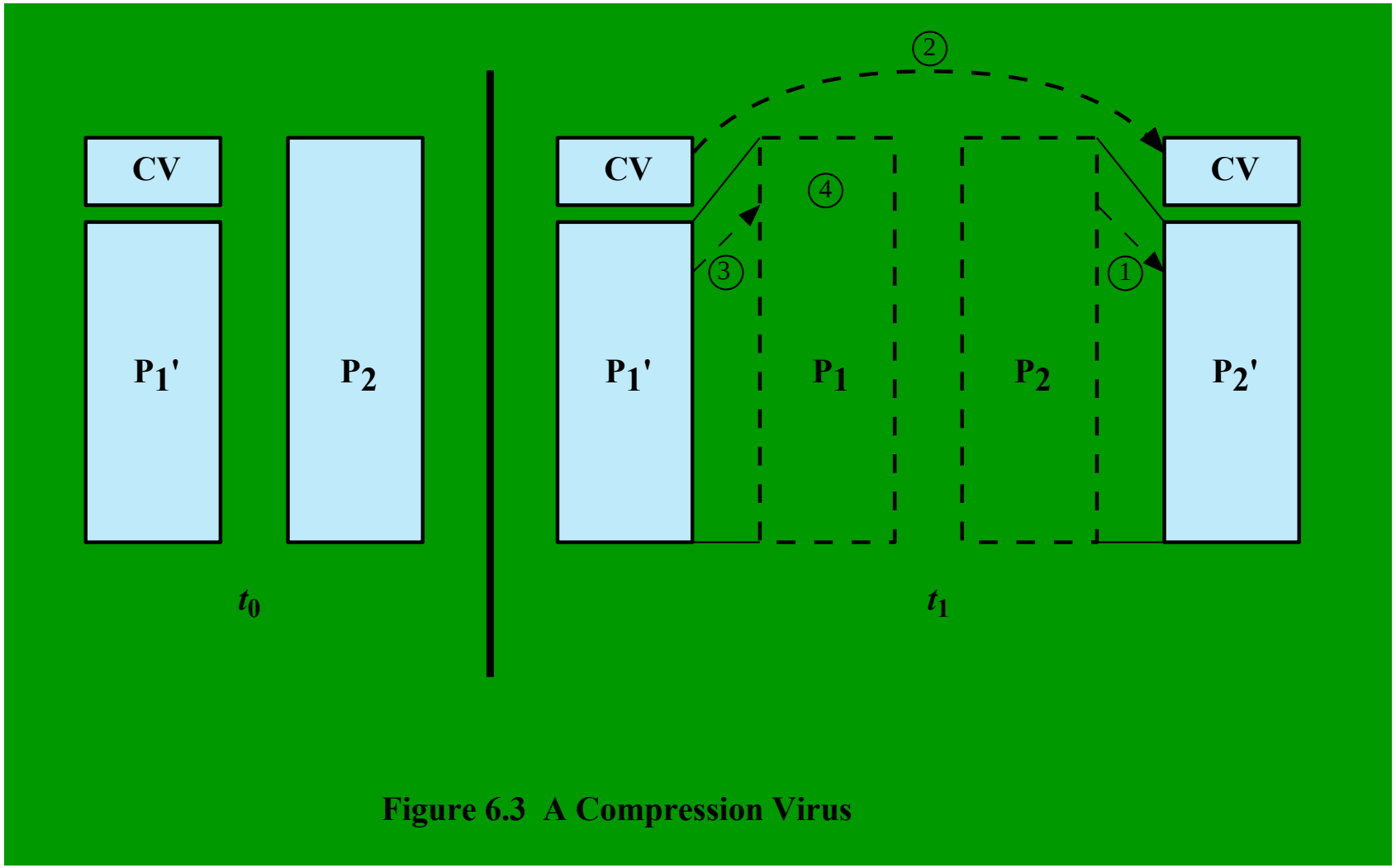

Figure 6.3 A Compression Virus

Virus Classification

- Aycock, classifies viruses into two classes:
  - By the targets that the viruses attack
  - By the method the viruses hide from detection

- Virus Categories by its targets:
  - Boot Sector Infector – infecting the master boot record and spreading when the system is booted
  - File Infector – infecting executable files
  - Macro virus – infecting macro or scripting files interpreted by the application
  - Multipartite virus – infecting files in multiple ways

Virus Classification

- Virus categories by it concealment:
  - Encrypted virus
    - A portion of the virus creates a random encryption key and encrypts the remainder of the virus
    - Store the key with the virus
    - When an infected program is executed, the virus decrypts itself
    - When the virus spreads, it creates a different random key.
    - No constant bit pattern is noticed since each virus has different key
  - Stealth virus – a form of virus and its payload that are intentionally designed to hide from detection
  - Polymorphic virus – a virus that is hard to detect by its signature since it mutates/changes with every infection
  - Metamorphic virus – like polymorphic, it mutates in every infection
    - However, metamorphic completely rewrites itself at every iteration that make it even harder to detect

Macro and Scripting Viruses

Threats of macro or scripting code viruses:

- A macro virus is platform independent.
  - Any system that support the applications using the macro can be infected
  - MS office, PDF
- It infects documents, not code
  - More documents are input to the system than code
- It can easily spread
  - Documents are commonly shared
- Traditional file system access controls are not effective in preventing their spread
  - Users are expected to modify documents

Propagation by exploiting vulnerability

- A worm is a program that looks for other machines to infect
  - Each infected machine is also used to find other machines
  - John Brunner (1975) uses the term “worm” and its concept in his novel book “The Shockware Rider”
  - The first known worm was not malicious, implemented in Xerox labs
  - The Morris worm: a well-known worm spread on Unix in 1988
- Several worms are also viruses, e.g., Melissa, Nimda
- A worm exploits software vulnerabilities in either client or server.
- Worms can spread through network connections, shared media (USB, CD, DVD), and email
  - A worm can spread faster than a virus, e.g., Code Red worm infected 360k computers in 14 hours

Means for worm replications

- Electronic mail or instant messenger facility
  - Melissa, Nimda, Mydoom, Warezov
- File sharing
  - Conficker, Stuxnet
- Remote execution capability
- Remote file access or transfer capability
- Remote login capability

Worm phases

- Similar to virus, worms has four phases: dormant, propagation, triggering, execution.
- Propagation phase performs:
  - Search for means to access other systems to infect
    - Host tables, address books, buddy lists, trusted peers, target host addresses, and others.
  - Use the access to transfer a copy of itself and execute the copy
  - Worm can check if the system has been infected

How does worm find a target?

- The first step, scanning or fingerprinting, is a function for network worm to search for other system to infect
  - Identify systems running vulnerable service.

- Types of network address scanning strategies
  - Random – use random IP addresses
  - Hit-List – compile a long list of vulnerable machines and infect the machines on the list
  - Topological – use information contained on infected machine to find more hosts to scan
  - Local subnet – look for targets within the same local network behind firewalls

Worm propagation model

- The speed of propagation depends on:
  - the mode of propagation: by email? By file sharing? Etc.
  - the exploited vulnerability,
  - the similarity to previous attack

- Three phases of propagation:
  - Initial phase: the number of host increases exponentially
  - Middle phase: linear growth
  - Finish phase: slow since remaining hosts are mostly infected

Trojan Horses

- A trojan horse contains a hidden code that when called performed unwanted or harmful function
  - It may be a useful program or utility
- Some possible harmful function:
  - Gain access to sensitive personal information and send a copy of it to the attacker.
- Users must be careful when downloading software from unknown source

Payload – System Corruption

- Payload is the action that the malware takes on the target
  - Some malware does not have payload
- Some possible payload: data destruction, real-world damage, logic bomb
- Examples of malware that destruct data:
  - Chernobyl virus: it deletes data on the infected system by overwriting the first megabyte of the hard disk with zeroes
  - Klez mass-mailing worm: on trigger date, it causes files on hardware to become empty
  - Cyborg trojan and Gpcode encrypt the user's data and ask for ransom to decrypt the data

Payload – System Corruption

- Examples of malware that cause real world damage:
  - The payload aims to damage the physical system
  - Chernobyl virus also attempts to overwrite the BIOS code that boot the computer
    - If successful, the BIOS chip must be replaced or reprogrammed
  - Stuxnet worm: targets specific industrial control system software
    - The worm replaces the original code and drive the controller equipment beyond its normal operation to cause failure and damage
- Logic bomb is a code as part of a malware that will explode when certain conditions are met such as,
  - Presence or absence of certain files or devices
  - A particular date or day
  - A particular user running the program

Payload – Attack Agent

- A bot is a compromised machine that can be remotely controlled by the attacker (the bot master)
  - Also called robot, drone, or zombie
  - Botnet is a collection (hundreds, thousands, even millions) of bots under the control of the bot master.
- Some uses of botnet:
  - Distributed DoS
  - Spamming
  - Sniffing traffic
  - Keylogging
  - Spreading new malware
  - Manipulating online polls/games

Payload – Information Theft

- Some malware (keyloggers, phishing, spyware) gathers data stored on the infected system for use by the attacker
  - Login and password, Bank account, Gaming, etc.
  - These attacks target the information confidentiality.
- The attacker installs keylogger that captures keystrokes on the system
  - This allows the attacker get the login and password even when they are sent over encrypted channels (e.g., HTTPS).
  - Keyloggers can return only desired keywords using some form of filtering mechanism
    - e.g., login, password, paypal.com
  - Countermeasure: use graphical applet to enter critical information that cannot be captured by the traditional keyloggers
  - More general spyware can monitor a wide range of activity on the victim

Payload – Information Theft

- The attacker can send a SPAM with URL that links to a fake Web site similar to some banking → phishing attack
  - Careless users may follow the link and provide their critical information
  - Spear phishing attack is targeting better researched victims that include information specific to the victims

Payload – Stealthing

- Some malware hides its existence on the victim's machine but provides covert access to that system
  - backdoor, rootkit
- A backdoor or trapdoor is a secret entry into a program that allows the attacker to gain access without going through regular security check
  - Programmers use this backdoor, called maintenance hook, to debug and test program
  - It is difficult to implement OS control for backdoor in applications
- A rootkit is a malware with supervisory access rights
  - It has access to all OS services and functions
  - It can add/change programs and files, monitor processes, and hide
  - It hides by corrupting the mechanisms for monitoring processes, files, and registries on the computer

Countermeasures

- The best solution: prevention
  - do not allow malware to get into the system or prevent the malware to modify the system
  - In general almost impossible
- Four elements of prevention (NIST): policy, awareness, vulnerability mitigation, and threat mitigation.
- If prevention fails, technical mechanism can be used for threat mitigation:
  - Detection: know that infection has occurred and where it is located
  - Identification: find out the specific malware
  - Removal: remove all traces of the infection to prevent further spread
- If detection succeeds but identification and removal fail, replace all infected files with clean files from backup

Requirements for countermeasures

- Generality: the approach can address a wide variety of attacks
- Timeliness: it should respond quickly
- Resiliency: it is resistant against the attacker's hiding technique
- Minimal denial of service cost: it does not significantly reduce the system capacity and disrupt normal operation
- Transparency: it does not require modification to application and system software as well as hardware
- Global and local coverage: it can deal with attack from both inside and outside the network

Where to run antivirus program?

- Run some host-based antivirus program on the infected system
- Run antivirus on the perimeter security mechanisms in the firewall or as part of the intrusion detection mechanism
- Use distributed mechanism that gather data from both host-based and perimeter

Host-based scanners

First generation: simple scanners

- Require a malware signature to identify it
- Can detect only known malware
- The scanner may keep the length of programs and looks for changes in length

Host-based scanners

Second generation: heuristic scanners

- It does not rely on malware signature but uses heuristic rules to search for probable malware instances
- It looks for fragments of code that are often associated with malware
- It uses integrity checking
  - It may add checksum to each program
    - if malware modifies the program without changing the checksum, it will be detected.
  - More sophisticated malware is able to change the checksum when it modify the program
    - Counter this using encrypted hash function with the encryption key stored somewhere else

Host-based scanners

Third generation: activity traps

- It is memory resident program that identify malware from its action rather than its structure
- It does not need signatures or heuristics for wide variety of malware
- It needs to identify small set of actions that indicate malicious activity

Host-based scanners

Fourth generation: full-featured protection

- It includes scanning and activity traps
- It also includes access control capability

Questions ?

Thank you