P2L1 Malicious Software

Types of Malware

Needs a host program-

There are several ways to imbed into programs:

- Trap doors, logic bombs, trojan horses, viruses, browser plug-ins, extensions, scripts

Independent-

- Worms, botnets, APT (Advanced Persistent Threats)

Trap Doors (or Back doors)

A secret entry point to a program or system.
It typically works by recognizing some special input or special user ID.
The flight simulator Easter Egg embedded in Microsoft software is an example of a back door.

Logic Bomb

Embedded in some legitimate program
It will explode or perform malicious activities when certain conditions are met.
For example: it will perform on a specific time and date.

Trojan Horses

Hidden in an apparently useful host program
Performs some unwanted/harmful function when the host program is executed

- Unwanted functions might be keylogging

Viruses

Infects a program by modifying it
It can self copy into the program to spread itself
There are 4 stages of a virus

- Dormant phase - the program has been infected, but the virus has not been triggered.
- Propagation phase- the virus is being spread, for example through email attachments.
- Triggering phase - when the host program is run and the virus is run.
- Execution phase- when the virus runs and performs malicious activities. It also looks for occasions to spread.

Host Required Malware Quiz #1

virus = email attachment that when opened will send itself to all people in the user's address book

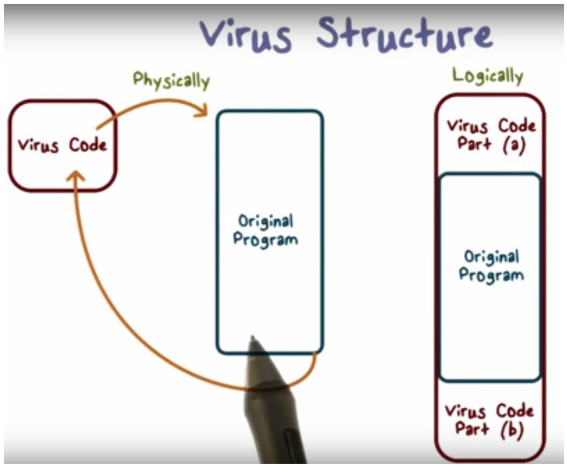
trojan horse = a customized keyboard app that logs user input and sends it to a server on the Internet

logic bomb = part of a program will only run if the computer is at the user's home, then it will upload all MS Word docs to a website.

trapdoor = a login program with an undocumented option that will allow an attacker to supply any username and password to gain access to the computer.

Host-Required Malware Quiz #2
Spy on employees of a specific company - use a trojan horse
Cripple an organization's computers - use a logic bomb
Quickly spread information and drive traffic to a specific website - use a virus

Virus Structure



The virus code has to be physically inserted into the program code.

The virus code runs first, then the original program runs. The user should not suspect that there is a virus, the program should run correctly. The virus may also run at the end of the program to do clean up to avoid detection.

- The first line of the host program will say to go to the virus program. This will guarantee that the virus will run first.
- The second line of the host program will have a special flag to signify that it has been infected. Otherwise the host program will be repeatedly infected.
- The main of the virus program
    - find uninfected program - infect them
    - do something damaging to the system
    - "go to" first line of the host program - do normal work

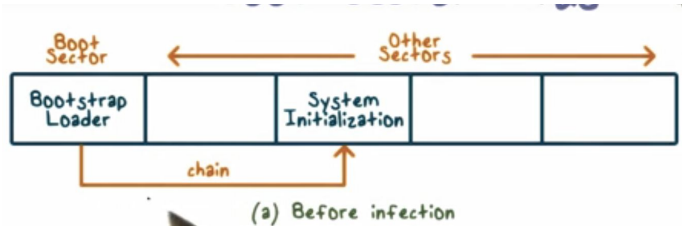
- Avoid detection by looking at the size of the program
    - Compress/decompress the host program

Types of Viruses

Parasitic virus: scan/infect programs
Memory-resident virus: infect running programs
Macro virus: run/spread whenever the system is booted
Polymorphic virus: encrypt part of the virus program using a randomly generated key

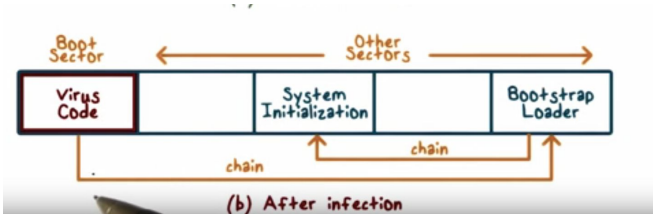
Any virus can be polymorphic

Boot Sector Virus



Before infection:

Boot sector is a sector on the hard drive. The code loaded here always runs first, it is called Bootstrap Loader.



After infection the virus code is the first code that is executed in the boot sector.
Then control is transferred to the bootstrap loader.

Macro Viruses

Macro: an executable program embedded in a word processing document. For example: MS Word.

How a macro virus is spread:

- A virus macro is attached to a word document
- Document is loaded and opened in the host system
- When the macro executes, it copies itself to the global macro file
- The global macro can be activated/spread when new documents are opened

Macro Viruses are embedded in documents
Boot Sector Viruses reside on the hard drive and execute before the OS is loaded
Memory resident viruses are OS level viruses

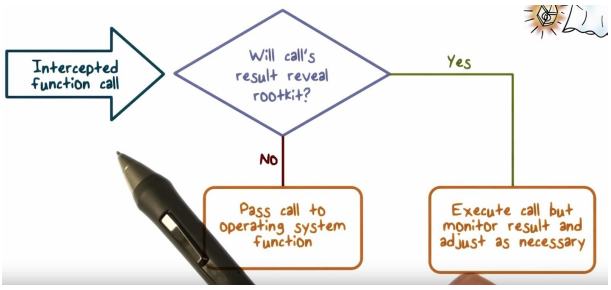
Rootkit

Resides in operating systems

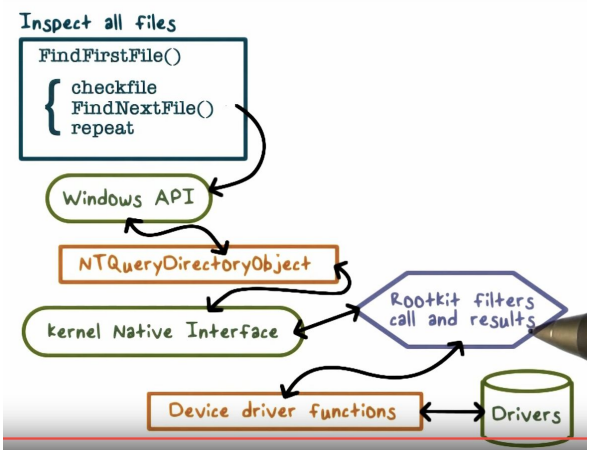
- Modifies OS code and data structure
    - Rootkits can be used to hide itself from the user by not listing itself during an ls command.

Assume the user uses ‘dir’ to look at files in a directory.
The command is implemented with a loop that keeps looking for the next file in the directory.

Files and directories are stored on the hard drive, which is controlled by the OS. Any access to the hard drive must go through the OS. The rootkit must hide itself from the user.



The rootkit will intercept the function call to the OS and check to see if it is revealed to the user. If it is, it will filter the results that are shown to the user.



The flow chart for how a rootkit modifies the output of the OS.

Linux, iOS, Windows, Android operating systems have all been affected by rootkits.

-All operating systems can be affected by rootkits -
-Rootkits can modify hidden and read-only files
-Rootkits can spread in any form
-Rootkits cannot remain in memory after reboot - but …. the Rootkit is part of the OS, so when the OS starts again, the Rootkit returns.
-Rootkits cannot infect hardware - hardware that does not have firmware.
-Rootkits are always malevolent

Worms
Use network connections to spread from system to system.
Worms later evolved into botnets.

The First Internet Worm

- Determine where it could spread
- Spread its infection
- Remain undiscovered and undiscoverable

The effect of the worm:

Resource exhaustion because repeated infection due to programming bug.
Servers were disconnected from the Internet by system admin to stop the infection

An internet worm:

- exploits security flaws
    - Such as weak passwords. It tries to guess password
    - fingerD: computers running the fingerD program experienced buffer overflow
    - sendmail: mail that had trap doors (the email accepts shell commands)

The worm spread:

- Bootstrap loader is put on the target machine, then fetches the rest of the code
- It uses a password authentication to make sure only the worm can execute the boot strap loader software

To Remain Undiscoverable:

- Load code in memory, encryp, remove file
- It periodically changes it name and process ID

What was learned from the worm:

- security scanning and patching must be done. The worm was so successful because the servers had security flaws.
- Need a CERT (computer emergency response team)

Worm Quiz

The following methods can be used to spread a worm:

email
instant messaging
downloading files
watching a video on netflix
clicking on a popup
using facebook

Malware Prevention and Detection Approaches

Prevention: limit contact to outside world
Detection and Identification:
Removal

Prevention really hampers productivity, so detection is the method that is preferred.

4 Generations of Antivirus Software:

- Simple scanners - use signatures of known viruses. Not effective against polymorphic viruses because they do not have a unique signature

- Heuristic scanners: Integrity checking = making sure the checksum is the same as before infection. This can be defeated by compressing the file to have the same size as the pre-infection file
- Activity traps: they look for specific activities that malware performs. For example: modification of a password file. These traps are not effective against newer malware.
- Full-featured analysis: this is state of the art. It is host-based, network-based, and sandbox-based. A sandbox is used to run a piece of executable code in a method that will do no permanent harm to the system

Why are signature-based anti-virus solutions still used, even though they are not effective?

- they are very efficient
- they are effective against known malware
- they are a good first-line of defense