

# Chapter 19 MALICIOUS LOGIC

## 19.1 Introduction

- Computer systems can be attacked with 3 types of malicious logic:
  - Trojan horses
  - computer viruses
  - computer worms
- **DEFINITION:** Malicious logic is a set of instructions that cause a site's security policy to be violated.
- **EXAMPLE:**

```
cp /bin/sh /tmp/.xyzzzy
chmod u+s,o+x /tmp/.xyzzzy
rm ./ls
ls $*
```

UNIX OS user identity: UID

UNIX script ls is placed in a directory.

Creates a copy of the UNIX shell that is setuid (set user ID) to the user executing this program.

This program is deleted and the correct ls command is executed.

On most systems, it is against the policy to trick someone into creating a shell that is setuid to themselves!

If someone is tricked into executing this script, a violation of the (implicit) security policy occurs.

# Chapter 19 MALICIOUS LOGIC

## 19.2 Trojan horses

- **DEFINITION:** A Trojan horse is a program with an overt (documented or known) effect and a covert (undocumented or unexpected) effect.
- **EXAMPLE:** The previous script is a Trojan horse
  - The overt purpose is to list the files in a directory.
  - The covert purpose is to create a shell that is setuid to the user executing the script.
- This term was named by Dan Edwards in the Anderson Report (J. Anderson, “Computer Security Technology Planning Study,” Technical Report ESD-TR-73-51, Electronic Systems Division, Hanscom Air Force Base, Hanscom, MA, 1974).
- **EXAMPLE:** NetBus program
  - Allows an attacker to control a Windows NT workstation remotely.
  - The program acts as a server, accepting and executing commands for remote administrator:
    - Includes intercepting keystrokes and mouse motions and sending them to the attacker.
    - Also allows the attacker to upload, download files.
    - Would work if the victim NT system has a server with which the NetBus program can communicate.

# Chapter 19 MALICIOUS LOGIC

- A Trojan horse (propagating Trojan horse or a replicating Trojan horse) can make copies of itself.
- One of the earliest Trojan horses was a version of the game animal.
  - When the game was played, it created an extra copy of itself.
  - The program was modified to delete one copy of the earlier version and create two copies of the modified program.
    - As the modified version spread more rapidly than the earlier version, it completely supplanted the earlier version.
    - After a preset date, each copy of the modified version deleted itself after it was played.
- Karger and Schell, and later Thompson examined detection of Trojan horses.
  - They constructed a Trojan horse that propagated itself slowly and in a manner that was difficult to detect.
  - The central idea: The Trojan horse modifies the compiler to insert itself into specific programs, including later version of the compiler.

# Chapter 19 MALICIOUS LOGIC

## 19.3 Computer viruses

- When the Trojan horse can propagate freely and insert a copy of itself into another file, it becomes a computer virus.
- **DEFINITION:** A computer virus is a program that inserts itself into one or more files and performs some (possibly null) action.
- The first phase is the insertion phase: the virus inserts itself into a file.
- The second phase is the execution phase: the virus performs some (possibly null) action.

# Chapter 19 MALICIOUS LOGIC

## Pseudocode

```
beginvirus:  
  if spread-condition then begin  
    for some set of target files do begin  
      if target is not infected then begin  
        determine where to place virus instructions  
        copy instructions from beginvirus to endvirus into target  
        alter target to execute added instructions  
      end;  
    end;  
  end;  
  perform some action(s)  
  goto beginning of infected program  
endvirus:
```

The insertion phase must be present:

- Need not always be executed.
  - The Lehigh virus would check for an uninfected boot file (the spread condition in the pseudocode).
  - If one is found, it would infected the file (the set of target files).
  - The virus would then increment a counter and test to see if the counter = 4. }
  - If 4 is reached, the virus would erase the disk. }
- The actions

# Chapter 19 MALICIOUS LOGIC

- **Authorities differ on whether or not a computer virus is a type of Trojan horse.**
  - **Some argue that the answer is YES.**
    - Overt action = infected program's actions
    - Covert action = virus' actions (infect, execute)
  - **Some argue that the answer is NO.**
    - Overt purpose = virus' actions (infect, execute)
    - Covert purpose = none
- **In some sense, this disagreement is semantic.**
- **In any case, defenses against a Trojan horse inhibit computer viruses.**
- **History of computer viruses**
  - **Programmers wrote the first computer viruses on Apple II computers.**
  - **In 1983, Fred Cohen was a graduate student at the USC. He described a type of Trojan horse that an instructor (Len Adleman) named a “computer virus.”**
    - Cohen designed a computer virus to acquire privileges on a VAX-11/750 running the UNIX OS.
    - He obtained all system rights within half an hour on the average (longest time=an hour, shortest time less than 5 minutes).
    - The virus did not degrade the respond time noticeably. Most users never knew the system was under attack.

# Chapter 19 MALICIOUS LOGIC

- **History of computer viruses (continued)**
  - In 1984, Cohen's experiment on a UNIVAC 1108 showed that viruses could spread throughout that system.
    - Unlike the UNIX system, the UNIVAC partially implemented the Bell-LaPadula model, using mandatory protection mechanisms (the simple security condition was implemented but the \*-property was not).
    - As writing was not inhibited (no \*-property enforcement), viruses spread easily.
  - The Brain (Pakistani) virus was created in early 1986.
    - Written for IBM PCs
    - Alters the boot sectors of floppy disks, possibly corrupting files in the process.
    - Also spreads to any uninfected floppy disks inserted into the system.
  - In 1987, computer viruses infected Macintosh, Amiga, and other computers.
    - The MacMag Peace virus would print a "universal message of peace" on March 2, 1988, and then delete itself.
  - In 1987, Tom Duff experimented on UNIX systems with a small virus that copied itself into executable files.
    - 48 infected programs were placed on the most heavily used machine in the computer center. The virus spread to 46 systems in 8 days.

# Chapter 19 MALICIOUS LOGIC

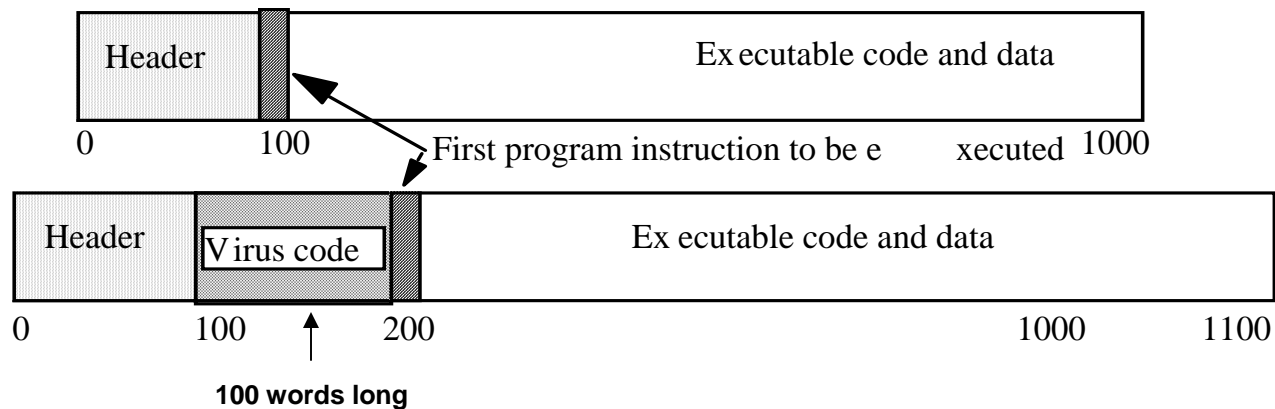
- **History of computer viruses (continued)**
  - In 1989, Harold J. Highland developed a virus for Lotus 1-2-3.
    - The virus was stored as a set of commands in a spreadsheet, and was loaded when a file was opened.
    - The virus was intended for a demonstration only. It changed a value in a specific row, column and then spread to other files.
- **Types of Viruses**
  - A boot sector infector
  - An executable infector
  - A multipartite virus
  - A terminate and stay resident (TSR) virus
  - A stealth virus
  - An encrypted virus
  - A polymorphic virus
  - A macro virus

# Chapter 19 MALICIOUS LOGIC

- A boot sector infector is a virus that inserts itself into the boot sector of a disk.
- **Example: The Brain virus for the IBM PC is a boot sector infector.**
  - When the system boots from an infected disk, the virus is in the boot sector and is loaded.
  - It moves the disk interrupt vector from 13H or 19 to 6DH or 109, and sets the disk interrupt vector location to invoke the Brain virus now in memory.
  - It then loads the original boot sector and continues the boot.
  - When the user reads a floppy, the interrupt at location 13H is invoked.
  - The Brain virus checks for the signature 1234H in the word location at 4.
  - If the signature is present, control is transferred to the interrupt vector at location 6DH so that a normal read can proceed.
  - Otherwise, the virus infects the disk.

# Chapter 19 MALICIOUS LOGIC

- An executable infector is a virus that infects executable programs.
- The PC variety of executable infectors are called COM or EXE viruses because they infect the programs with those extensions.



- The above figure shows how infection can occur.
- The executable infector inserts itself into the program so that the virus code will be executed before the application code.
- The virus can prepend itself to the executable or append itself.

# Chapter 19 MALICIOUS LOGIC

- A multipartite virus is a virus that can infect either boot sectors or executables.
- Such a virus typically has two parts, one for each type.
- When it infects an executable, it acts as an executable infector.
- When it infects a boot sector, it works as a boot sector infector.

# Chapter 19 MALICIOUS LOGIC

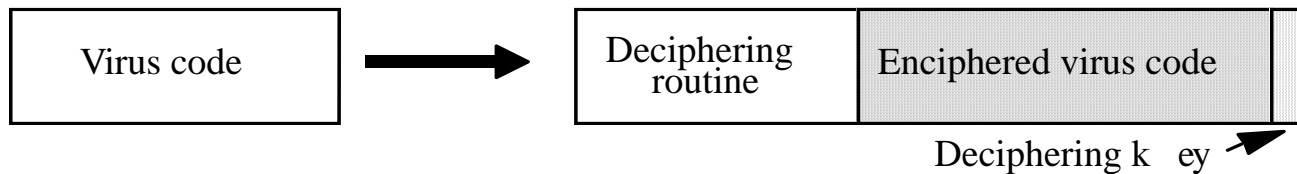
- A terminate and stay resident (TSR) virus is a virus that stays active in memory after the application (or bootstrapping, or disk mounting) has terminated.
- TSR viruses can be boot sector infectors or executable infectors.
- The Brain virus is a TSR virus.

# Chapter 19 MALICIOUS LOGIC

- A stealth virus is a virus that conceals the infection of files.
- These viruses intercept calls to the operating system that access files.
  - If the call is to obtain file attributes, the original attributes of the file are returned.
  - If the call is to read the file, the file is disinfected as its data is returned.
  - If the call is to execute the file, the infected file is executed.
- The Stealth virus (also called the IDF virus or the 4096 virus) is an executable infector.
  - It modifies the DOS service interrupt handler as follows:
    - If the request is for the length of the file, the length of the uninfected file is returned.
    - If the request is to open the file, the file is temporarily disinfected, and it is reinfected on closing.
    - If the request is to load the file for execution, the infected file is loaded.
  - The Stealth virus also changes the time of last modification of the file in the file allocation table to indicate that the file is infected.

# Chapter 19 MALICIOUS LOGIC

- An encrypted virus is a virus that is enciphered except for a small decryption routine.
- Computer virus detectors often look for known sequences of code to identify computer viruses.
- To conceal these sequences, some viruses encipher most of the virus code, leaving only a small decryption routine and a random cryptographic key in-the-clear.
- The below figure shows the ordinary virus on the left, and the deciphering routine, the enciphered virus, and the deciphering key on the right.



# Chapter 19 MALICIOUS LOGIC

- A polymorphic virus is a virus that changes its form each time it inserts itself into another program.
- An encrypted virus will not be detected using known sequences of instructions. However, the decryption algorithm can be detected.
- Polymorphic viruses are designed to prevent this.
- They change the instructions in the virus to something equivalent but different.
  - The decryption routine is the segment of the virus that is changed.
  - Polymorphic viruses are successors to the encrypted viruses and are often used in conjunction with them.
- At the instruction level, the instructions are substituted.
- At the algorithm level, different algorithms are used to achieve the same result.

# Chapter 19 MALICIOUS LOGIC

- A macro virus is a virus composed of a sequence of instructions that is interpreted rather than executed directly.
- Macro viruses can infect either executables (Duff's shell virus) or data files (Highland's Lotus 1-2-3 spreadsheet virus).
- Duff's shell virus can execute on any system that can interpret the instructions.
- A spreadsheet virus executes when the spreadsheet interprets these instructions.
- The Melissa virus infected Word 97 and Word 98 documents on Windows and Macintosh systems.
  - It is invoked when the program opens an infected file.
  - It installs itself as the "open" macro and copies itself into the Normal template so that any files that are open are infected.
  - It then invokes a mail program and sends copies of itself to everyone in the user's address book that is associated with the program.

# Chapter 19 MALICIOUS LOGIC

## 19.4 Computer worms

- A computer virus infects other programs.
- **DEFINITION:** A computer worm is a program that copies itself from one computer to another.
- **Research into computer worms began in the mid-1970s.**
  - Schoch and Hupp developed distributed programs to do computer animation, broadcast messages, and perform other computations.
  - These programs probed workstations.
  - If the workstation was idle, the worm copied a segment onto the system.
  - The segment was given data to process and communicated with the worm's controller.
  - When any activity other than the segment's began on the workstation, the segment shut down.

# Chapter 19 MALICIOUS LOGIC

- **EXAMPLE: Internet Worm of 1988**
  - On November 2, 1988, a program targeting Berkeley and Sun UNIX-based computers entered the Internet.
  - Within hours, it had rendered several thousand computers unusable.
  - Among other techniques, this program used a virus-like attack to inject instructions into a running process on the target machine and arranged for those instructions to be executed.
  - To recover, these machines had to be disconnected from the network and rebooted.
    - Several critical programs had to be changed and recompiled to prevent reinfection.
    - The only way to determine if the program had suffered other malicious side effects (e.g., deletion of files) was to disassemble it.
    - Fortunately, the only purpose of this virus was self-propagation.

# Chapter 19 MALICIOUS LOGIC

- **EXAMPLE: Father Christmas worm**
  - An electronic Christmas card passed around several IBM-based networks.
  - The card was an electronic letter instructing the recipient to save it and run it as a program.
  - The program drew a Christmas tree with blinking lights and printed “Merry Christmas!”
  - It then checked the recipient’s list of previously received mail and the recipient’s address book to create a new list of email addresses.
  - Finally, it sent copies of itself to all these addresses.
  - The worm quickly overwhelmed the IBM networks and forced the networks and systems to be shut down.
  - This worm had the characteristics of a macro worm.
    - It was written in a high-level job control language, which IBM systems interpreted.
    - Like the Melissa virus, the Father Christmas worm was never directly executed, but its effects were just as serious.

# Chapter 19 MALICIOUS LOGIC

## 19.5 Other forms of malicious logic

- Malicious logic can have other effects, alone or in combinations with Trojan horses, computer viruses, and computer worms.
- Rabbits and Bacteria
  - Some malicious logic multiplies so rapidly that resources become exhausted. This creates a denial of service attack!
  - DEFINITION: A bacterium or a rabbit is a program that absorbs all of some class of resource.
  - EXAMPLE:

```
while true
do
    mkdir x
    chdir x
done
```
  - This shell script would quickly exhaust either disk space or file allocation table (inode) space.
  - However, the user who caused the crash using this program would be immediately identified when the system was rebooted.

# Chapter 19 MALICIOUS LOGIC

- **Logic Bombs**
  - Some malicious logic triggers on an external event, such as a user logging in or the arrival of midnight, Friday the 13<sup>th</sup>.
  - **DEFINITION:** A logic bomb is a program that performs an action that violates the security policy when some external event occurs.
  - **EXAMPLE:**
    - Disaffected employees who plant Trojan horses in systems use logic bombs.
    - A program that deletes company's payroll records when one particular record is deleted.
    - The particular record is usually that of the person writing the logic bomb.
    - The idea is if (when) he or she is fired, and the payroll record deleted, the company loses all those records.
  - **EXAMPLE:**
    - In the early 1980's, a program posted to the USENET news network promised to make administering systems easier.
    - The directions stated that the *shar* archive containing the program had to be unpacked, and the program compiled and installed, as *root*.
    - Midway down the *shar* archive were the lines `cd /` and `rm -rf *`.
    - Anyone who followed the instructions caused the lines to be executed. These commands deleted all files in the system.

# Chapter 19 MALICIOUS LOGIC

## 19.6 Defenses

- Defending against malicious logic takes advantage of several different characteristics of malicious logic to detect, or to block, its execution.
- The defenses inhibit the suspect behavior and the mechanism are imprecise.
  - They may allow malicious logic that does not exhibit the given characteristic to proceed.
  - They may prevent programs that are not malicious but do exhibit the given characteristic from proceeding.
- Malicious logic acting as both data and instructions
  - A computer virus inserts code into another program.
    - The object being written into the file is data: the set of virus instructions.
    - The virus then executes itself.
    - The approach is to treat “data” and “instructions” as separate types, and require a certifying authority to approve conversion.

# Chapter 19 MALICIOUS LOGIC

- **EXAMPLE: Logical Coprocessor Kernel (LOCK)**
  - A system designed to meet the highest level of security under the US Department of Defense TCSEC (the Trusted Computer System Evaluation Criteria).
  - The compiled programs are type “data”
  - The compiled programs cannot be executed until a sequence of specific, auditable events changes the label to “executable.”
  - The executable objects cannot be modified.
  - Hence, the viruses cannot insert themselves into programs.
  - This scheme recognizes that viruses threat programs as data (when they infect them by changing the file’s contents) and as instructions (when the program executes and spreads the virus) and rigidly separating the two.

# Chapter 19 MALICIOUS LOGIC

- **Malicious logic assuming the identity of a user**
  - Because a user (unknowingly) executes malicious logic, that code can access and affect objects within the user's protection domain.
  - Limiting the accessibility of objects should limit spread of malicious logic and effects of its actions.
  - The approach draws on mechanisms for confining information.
  - **Information Flow Metrics**
    - **Flow distance metric  $fd(x)$ :**
      - Initially, all information  $x$  has  $fd(x) = 0$
      - Whenever info  $y$  is shared,  $fd(y)$  increases by 1
      - Whenever  $y_1, \dots, y_n$  used as input to compute  $z$ ,  $fd(z) = \max(fd(y_1), \dots, fd(y_n))$
      - Information  $x$  is accessible if and only if for some parameter  $V$ ,  $fd(x) < V$
    - **EXAMPLE:**
      - Anne, Bill, and Cathy work on the same computer.
      - The system uses the flow distance metric to limit the flow of information.
      - Anne:  $V_A = 3$ ; Bill, Cathy:  $V_B = V_C = 2$ .
      - Anne creates program P containing a computer virus.
      - Bill executes P. P tries to write to Bill's program Q.
      - It works, as  $fd(P) = 0$ , so  $fd(Q) = 1 < V_B$ .
      - Cathy executes Q. Q tries to write to Cathy's program R.
      - It fails, as  $fd(Q) = 1$ , so  $fd(R)$  would be 2.

# Chapter 19 MALICIOUS LOGIC

- **Reducing the rights**
  - The basic idea is to remove the rights from a process so it can only perform its function.
  - This follows from the application of the principle of least privilege.
  - The principle of least privilege states that a subject should be given only those privileges that it needs in order to complete its task.
- **Sandboxing**
  - Sandboxes and virtual machines restrict process rights.
  - A common implementation is to restrict the program by modifying it.
    - Special instructions inserted into the object code cause traps when an instruction violates the security policy.
    - If the executable dynamically loads libraries, special libraries with the desired restrictions replace the standard libraries.

# Chapter 19 MALICIOUS LOGIC

- **Malicious logic crossing protection domain boundaries by sharing**
  - Inhibiting users in different protection domains from sharing programs or data will inhibit malicious logic from spreading among those domains.
  - If the idea is carried to its extreme, it would result in isolation of each domain.
  - Because sharing would not be possible, no viruses could propagate. Unfortunately, such a system would not be useful!
- **Malicious logic altering files**
  - Mechanisms using manipulation detection codes (MDCs) generate a signature block for each file.
  - If the signature block is recomputed, and the result differs from the stored signature block, this implies that the file has changed (possibly as a result of malicious logic altering the file).
- **Malicious logic performing actions beyond specifications**
  - Fault-tolerant techniques keep systems functioning correctly when the SW or HW fails to perform to specifications.

# Chapter 19 MALICIOUS LOGIC

- **Malicious logic altering statistical characteristics**
  - Like natural languages, programs have specific statistical characteristics that malicious logic may alter.
  - Detection of such changes may lead to detection of malicious logic.
- **The notion of trust**
  - The effectiveness of any security mechanism depends on the security of the underlying base.
  - The design of any mechanism for enhancing computer security must attempt to balance the cost of the mechanism against the level of security desired and the degree of the trust in the base that the site accepts as reasonable.