Week 7 - Wednesday

COMP 4290

Last time

- What did we talk about last time?
- Exam 1 post mortem
- Program security
- Non-malicious software flaws

Questions?

Project 2

Exponent properties

- If you want some review on how exponents work, try Khan Academy:
  - https://www.khanacademy.org/math/cc-eighth-grade-math/cc-8th-numbers-operations/cc-8th-exponent-properties/v/exponent-properties-involving-products

Kyle Hinkle Presents

Case Study: Therac-25

Therac-25 background

- Therac-25 was a radiation therapy machine built by the Atomic Energy of Canada Limited
- It was the successor to the Therac-6 and Therac-20 machines
- The machine had low power and high power modes
- The low power mode shot a beam directly at the patient
- The high power mode created X-rays by shooting the beam at a target, spread these X-rays with a flattening filter, shaped the beam with movable blocks, and tested the strength of the beam with an X-ray ion chamber

Tragedies

- In some situations, the high power beam was activated without the spreader in place
- The software and hardware systems did not catch this particular problem
- Over 100 times the intended dose was given
- At least 2 people died and there were at least 6 overdoses total
- Software bugs actually kill people!

Direct causes

- A certain unusual combination of keystrokes had to happen within 8 seconds
- There were no hardware interlocks to prevent the problem if the user overrode the error code
- Error codes were not well-documented and were displayed as a number
- Software was reused from previous models that **did** have hardware interlocks
- Arithmetic overflow caused safety checks to fail in some cases

Indirect causes

- The software/hardware combination had never been tested before use
- Personnel did not believe complaints due to confidence in the system
- Code was not independently reviewed
- Errors were easily overridden

Malicious Code

Malicious code

- Obviously, it's a problem
- It's very difficult to stop
  - You never really know what's getting installed on your computer
  - You're downloading thousands of files from the Internet every day
  - Even if you had the source code for every program, could you catch all the dangerous stuff?
- Malicious code has been around since at least 1970

Terminology

- **Malicious code** (or a **rogue program**) is our blanket term for any code that has undesirable effects that were intentionally designed
- The **agent** is the person who writes the code
- A **virus** is a program that can replicate itself and add malicious code to non-malicious programs
  - A **transient virus** runs when its host program is running
  - A **resident virus** lives in memory and can be active anytime

Viruses

- Terminology is inconsistent
- Popular culture tends to call lots of things a virus
- Sometimes we will too, but here are some other terms
- Almost all of these are, by definition, Trojan horses
- Worms differ from viruses primarily because they spread across networks

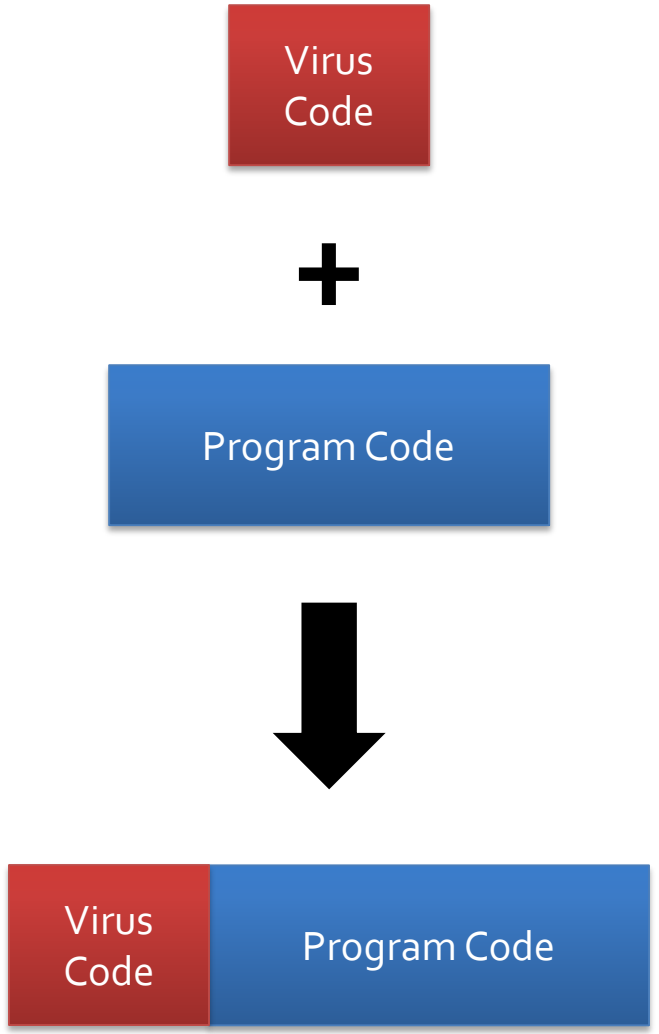
Type	Characteristics
Virus	Attaches itself to a program and propagates copies of itself to other programs
Trojan horse	Contains unexpected, additional functionality
Logic bomb	Triggers action when condition occurs
Time bomb	Triggers action when specified time occurs
Trapdoor	Allows unauthorized access to functionality
Worm	Propagates copies of itself through a network
Rabbit	Replicates itself without limit to exhaust resources
Spyware	Covertly communicates user data or user activities
Ransomware	Transfers data offsite or encrypts it, demanding money for the data or decryption key

How viruses attach

- A virus is not dangerous unless it's active
- Just having an infected file on your hard drive won't cause a problem unless it's accessed
- But files get opened all the time
  - Programs call other programs
  - Just previewing files can be dangerous
  - E-mail programs can open attachments automatically
- How do these viruses infect code?

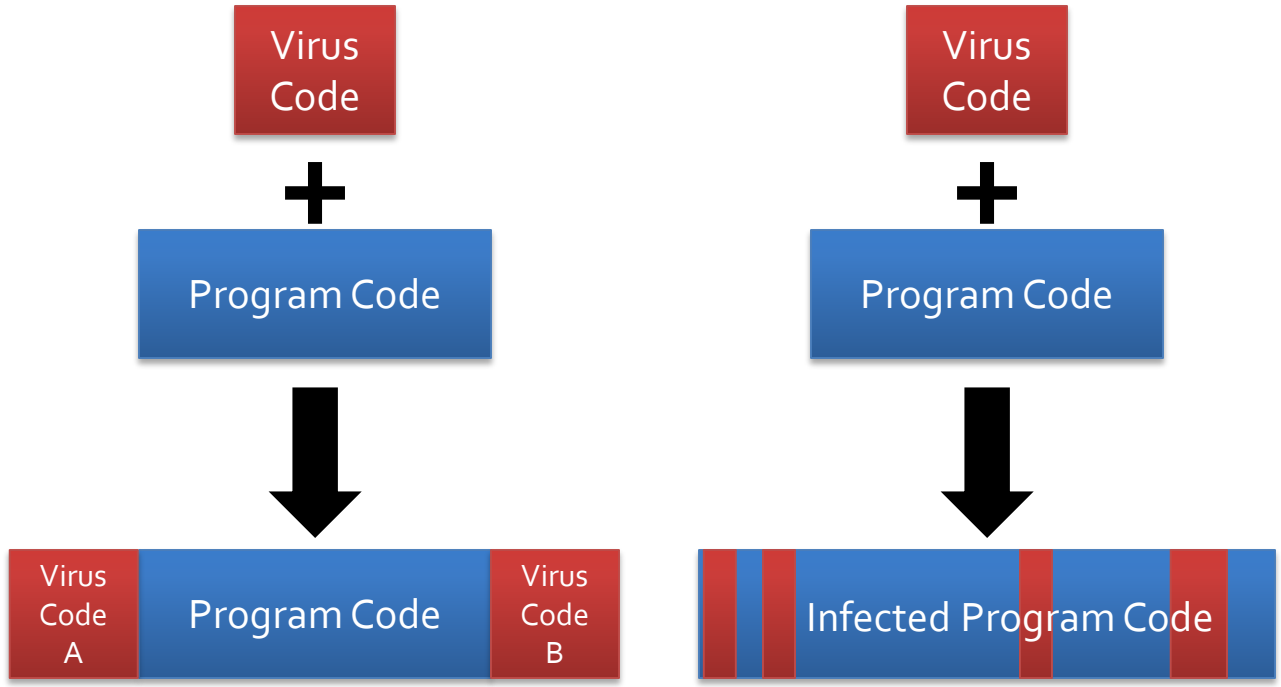
Appended viruses

- A virus can be designed so that it starts running before the real program does
  - Machine code for the virus is inserted before the machine code for the beginning of the program
  - After the virus runs, it transfers control to the real program
  - The real program runs as if nothing happened
- This kind of virus is easy to write
- It is also relatively easy to catch for antivirus software



Surrounding or integrating

- Another possibility is viruses that surround a program, gaining control before and after execution
  - The code may not be at the beginning and end of the executable, but that's how the control flow works
- Viruses can also be spread throughout the code



Document viruses

- The most common form of virus used to be a **document virus**
- A document virus is an infected document (instead of an executable file)
- Nevertheless, the macros that can be stored in Word, Excel, Access, and other similar complex documents are powerful enough to cause just as much damage as any other virus
- Document viruses are why Word and Excel force you to put documents into Edit mode to do stuff with them after you download them from the Internet

Where Viruses Live

The perfect virus

- If you're making a virus, the following characteristics are ideal:
  - Hard to detect
  - Difficult to destroy or deactivate
  - Spreads infection widely
  - Capable of re-infecting its host or other programs
  - Easy to create
  - Machine and OS independent
- It's difficult to make a virus that meets all these criteria

One-time execution

- Many viruses will be executed just once
- This could be on running a pirated (and infected) file
- One of the most common avenues of attack is through an e-mail attachment

Boot sector viruses

- The **boot sector** is the part of a hard drive that says what code to load to start your OS
- The details are technical, but a **boot sector virus** is stored in the chain of code that starts up your whole computer
- A virus that can start this early can circumvent or disable antivirus
- It has complete control over your system
- It's also not obvious from the file system

Memory resident viruses

- Some programs start up and then never really die
- They are low level parts of the OS that need to keep running
  - Sometimes called TSR (terminate and stay resident)
- Because these programs are always running, they are an attractive home for a virus
- Even if you delete the original infected file, the memory resident virus can replace it

Somewhere else …

- As with everything in security, the assumption is that attackers do not play by the rules
- A virus does not have to live where we expect it to
- A few other places that are sensible:
  - Applications
  - Libraries
  - Compilers (infect programs as you create them)
  - Antivirus software

Virus Signatures

A fundamental problem with Trojan horses

- Ken Thompson's seminal paper *Reflections on Trusting Trust*:
  - He added a backdoor to the Unix **`login`** program
  - Too easy to trace, so he added a backdoor to the C compiler to insert the backdoor in any program called **`login`**
  - Too easy to trace, so he added a backdoor in the compiler compiler to insert code that would insert the backdoor in any program called **`login`**
  - And so on, and so on …
- You can't trust anything you didn't completely create yourself
- Some amount of trust is necessary

Virus signatures

- Viruses are difficult to detect, but we can still classify them by the way they change code or the way they execute
- We call these tell-tale signs a **signature**
- Antivirus programs work by searching for certain signatures in code

Storage patterns

- At simplest, this is just a particular string of code in the binary
- Often this code is at the beginning of a program so that it gets control immediately
- Craftier viruses will put themselves other places that get jumped to early in execution
- An antivirus program can check:
  - The size of a file
  - The functioning of the code compared to some standard
  - It can look for suspicious execution patterns (weird JUMP instructions)
  - The program against a hash digest for the program

Execution patterns

- Viruses are also suspicious because of the way they execute
- The virus should:
  - Spread infection
  - Avoid detection
  - Cause harm
- How do these behaviors look like normal programs?
- How do they look abnormal?
- It's not easy to tell …

Polymorphic viruses

- Because virus scanners try to match strings in machine code, virus writers design **polymorphic viruses** that change their appearances
- **No-ops**, code that doesn't have an impact on execution, can be used for simple disguises
- Clever viruses can break themselves apart and hide different parts in randomly chosen parts of code
  - Similar to code obfuscation
- Advanced polymorphic viruses called **encrypting viruses** encrypt parts of themselves with randomly chosen keys
  - A scanner would have to know to decrypt the virus to detect it
- Virus scanners can't catch everything

Virus effects and causes

Virus Effect	Virus Cause	
Attach to executable program	▪ Modify file directory	▪ Write to executable program file
Attach to data or control file	▪ Modify directory ▪ Rewrite data	▪ Append to data ▪ Append data to self
Remain in memory	▪ Intercept interrupt by modifying interrupt handler address table	▪ Load self in non-transient memory area
Infect disks	▪ Intercept interrupt ▪ Intercept OS system call	▪ Modify system file ▪ Modify ordinary executables
Conceal self	▪ Intercept system calls	▪ Classify self as hidden file
Spread infection	▪ Infect boot sector ▪ Infect system program	▪ Infect ordinary program ▪ Infect data ordinary program reads
Prevent deactivation	▪ Activate before deactivating program	▪ Store copy to re-infect after deactivation

Prevention of infection

- It's impossible to prevent infection entirely
- Some guidelines:
  - Use only professional software acquired from reliable, well-established vendors
    - Open source is often good, but there's a huge spectrum of quality
  - Test all new software on an isolated computer
  - Open attachments only when you know them to be safe
  - Make a recoverable system image and store it safely
  - Make and retain backup copies of executable system files
  - Use virus detectors regularly and update them daily

Upcoming

Next time…

- Web security

Reminders

- Anu Regmi presents
- Read sections 4.1 – 4.4
- Work on Project 2