Malicious Software

UT CS361S

FALL 2020

LECTURE NOTES

What is “Malicious Code”?

“Software or firmware intended to perform an ***unauthorized*** process that will have ***adverse impacts*** on the confidentiality, integrity, or availability of a system. A virus, worm, Trojan horse, or other code-based entity that infects a host. Spyware and some forms of adware are also examples of malicious code.”

NIST Special Publication 800-53, Revision 5

Common Malware Classes

Malware can be classified by how it spreads or generically behaves

- Virus – typically has to be attached to another program (infection)
- Worm – typically spreads via network vulnerabilities
- Trojan Horse – typically appears benign but contains hostile operations

Malware can also be classified by its behavior

- Spyware – typically designed to steal information, observe behavior, etc.
- Adware – typically designed to “trap” a user into viewing certain ads
- Ransomware – typically locks data unless the user pays a ransom

Two Primary Components

Payload – The code that performs the (harmful) action

Attack Vector/Exploit/Delivery – The code that enables the payload

- May include a transmission component
- May include a stealth component
- May include a mechanism for bypassing security
- May be as simple as an email with an attachment

Malware History

Much of early computer security driven by military concerns

The biggest concern was an unauthorized user or program

Other concerns developed over time

The following slides are a very brief overview/highlight

1972 Government Report:

The technical issue of multilevel computer security is concerned with the concept of malicious threat. By this we recognize that the nature of shared use multilevel computer systems present to a malicious user a unique opportunity for attempting to subvert through programming the mechanism upon which security depends (i. e., the control of the computer vested in the operating system). This threat, coupled with the concentration of the application (data, control system, etc.) in one place (the computer system) makes computers a uniquely attractive target for malicious (hostile) action. Recognition of the implication of malicious threat is important to understanding the security limitations surrounding application of contemporary computer systems. The threat that a single user of a system operating as a hostile agent can simply modify an operating system to by-pass or suspend security controls, and the fact that the operating system controlling the computer application(s) is developed outside of USAF control, contribute strongly to the reluctance to certify (i. e., be convinced) that contemporary systems are secure or even can be secured.

https://apps.dtic.mil/sti/pdfs/AD0758206.pdf

1987: Fred Cohen's Viruses

"Computer Viruses: Theory and Experiments" Fred Cohen, 1987

Introduced the concept of a self-replicating, evil program

The program attaches to a "good" program infecting it

When the infected program is run, the virus runs

The virus does it's evil AND spreads itself to other programs

Concepts first proposed by Cohen in 1984

1988: Morris Worm

Robert Morris wrote a self-spreading piece of code (worm)

Spread using exploits in:

- send mail,
- Finger
- rsh/rexec

Also guessed weak passwords

Copied code to new machine, compiled, and executed

Accidentally re-infected machines until machines became unusable

DOS attack brought down the Internet

Impact of Early Viruses

Amazingly, most early malware ***DID MINIMAL DAMAGE***

Often just a delivery system with a weak payload

Many viruses spread for the sake of spreading

Even the Morris worm was disruptive by accident

There were exceptions, but also plenty of hype

1995: Concept Macro Virus

Microsoft Word and Excel have limited scripting (“Macros”)

Concept was the first virus written completely as a macro

It was a delivery system only with no payload

But was an important proof-of-concept

Users often open documents directly from email

2000: I Love You

Visual Basic Script virus

Appears as email attachment:

- LOVE-LETTER-FOR-YOU.txt.vbs
- The .vbs often hidden on Windows

When executed:

- Damaged many office files
- Sent email out to email address book ***automatically***

Spread worldwide in hours

2004: MyDoom

Fastest spreading mass mailer virus at the time
- Slows overall internet performance by about 10%
- Slows average web page load times by about 50% percent
- Responsible for approximately one in ten e-mail messages.

Appears as a delivery error, mail error, etc

Includes an attachment that, if clicked on, mails out copies

Also attempted to spread via P2P vile sharing Kazaa

Opened a back door for remote control

Attempted to launch a DDOS against the SCO Group's website

2005: Sony Rootkit

Sony CD's from the 2004-2005 era installed a "Rootkit"

- Rootkit, as name implies, usually installs with elevated access
- Using this elevated access, it can change the OS
- This bypasses usual security such as antivirus, etc
- Also usually very good at being undetectable

Installed at root with an EULA ***that did not mention the software***

In 2005, ***US-CERT ISSUED AN ADVISORY!!!***

Texas, ***under Greg Abbot***, was the first state to sue

Why is the Sony Rootkit So Bad?

In addition to violations of privacy, etc, caused:

- Slowing the system, consuming resources
- False alarms from antivirus

OPENED HOLES FOR ADDITIONAL MALWARE

- “Stinx-E trojan”

2013: Cryptolocker

Modern Ransomware

(1980's had a ransomware called CyberAIDS)

Locks up system and uses public key crypto

In addition to fiat currency, accepted BitCoin

2016: Mirai

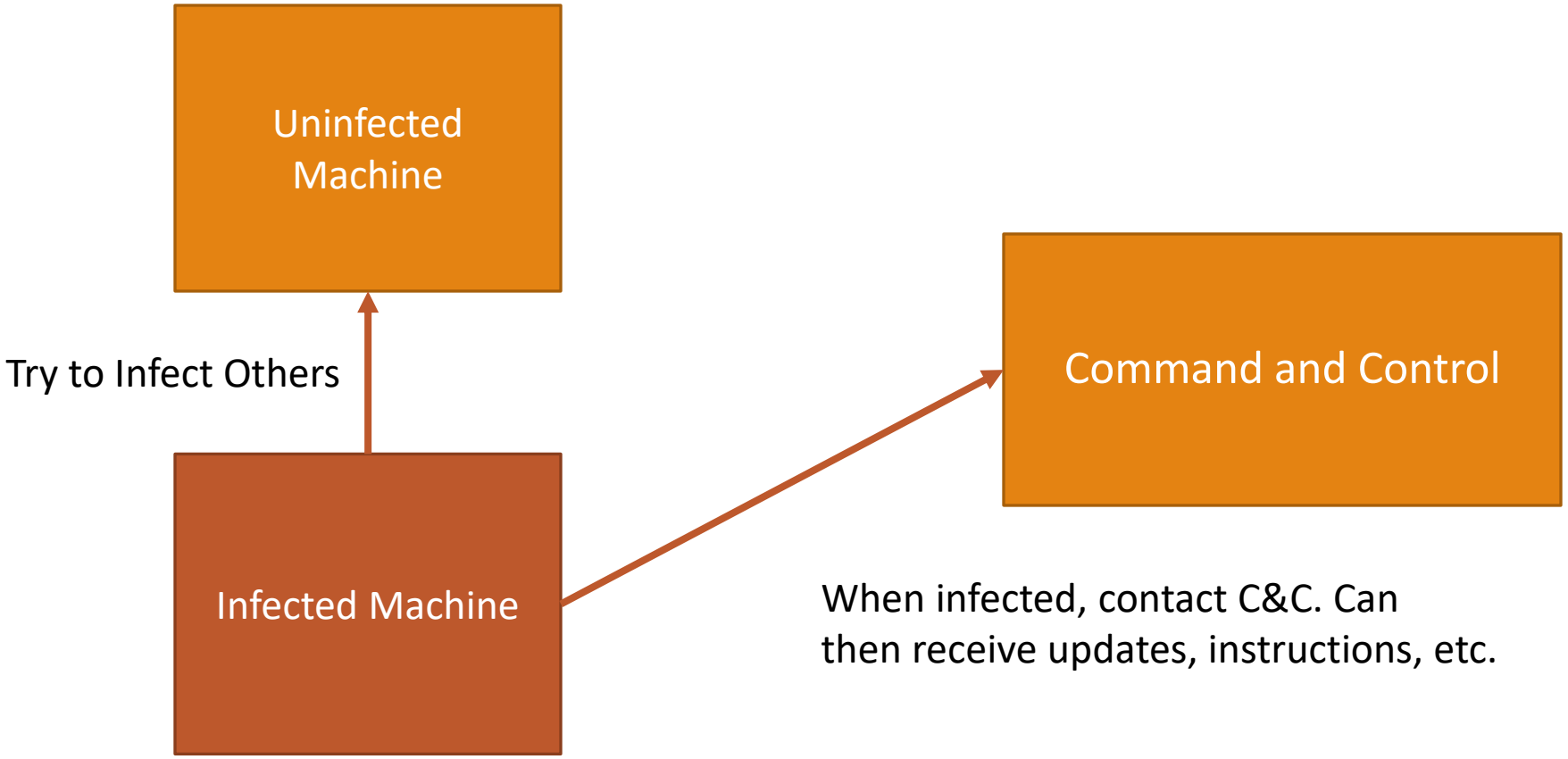
Worm that finds vulnerabilities in IoT devices

Takes over the device (“Zombie”)

Corrdinates all devices with a Command and Control

Launched a powerful DDOS against “krebs on security”

Command and Control Concept



Zero-day Vulnerabilities

What is a **ZERO DAY** exploit?

It is a vulnerability to a system for which a fix/defense is not yet available

Notice this isn't a very technical term

Could be in any form

- Virus
- Worm
- Trojan

Typically, the context is *serious* vulnerability

- Privilege escalation
- Login bypassing

Simultaneously New and Old

Obviously, by definition, zero day vulnerabilities are not new

- Every vulnerability starts out as a zero-day

But the *CONTEXT* has changed so drastically in recent years

- Political shifts (Drones, “Smart Power”)
- Ubiquity of Computing
- Reliance on Computing
- Automation
- Speed of information dissemination
- Broader knowledge of computing among the masses
- Tools

Zero Days from 2005 until Now

In the last ten years, a true “market” has emerged for buying zero-day’s

December 2005 – Fearwall tried to sell an exploit on Ebay

2006 – Unnamed security firm began selling exploits to US defense firms

2006 – Unnamed security firm had Charlie Miller selling exploits to U.S.

- I was there at the time but wasn’t involved
- Didn’t really know what was going on
- My current knowledge is from the book “Countdown to Zero Day”

2011 – “Middlemen” that broker deals

Currently, companies like VUPEN, etc

What are Zero-Days used for?

Defensive testing

- Some companies/agencies/groups buy zero-days for advanced penetration testing
- If one system is compromised, how do the others do?
- Why don’t they report these?

Bug bounties

Private Black Market

- Anonymous
- Russian Mafia

Military

Militarization of Cyberspace

Why Cyberweapons are seductive

- Plausible deniability
- Secrecy
- Potential to minimize casualties on both sides
- Cost effectiveness
- Political convenience
- CURRENTLY, ambiguity of law and regulation

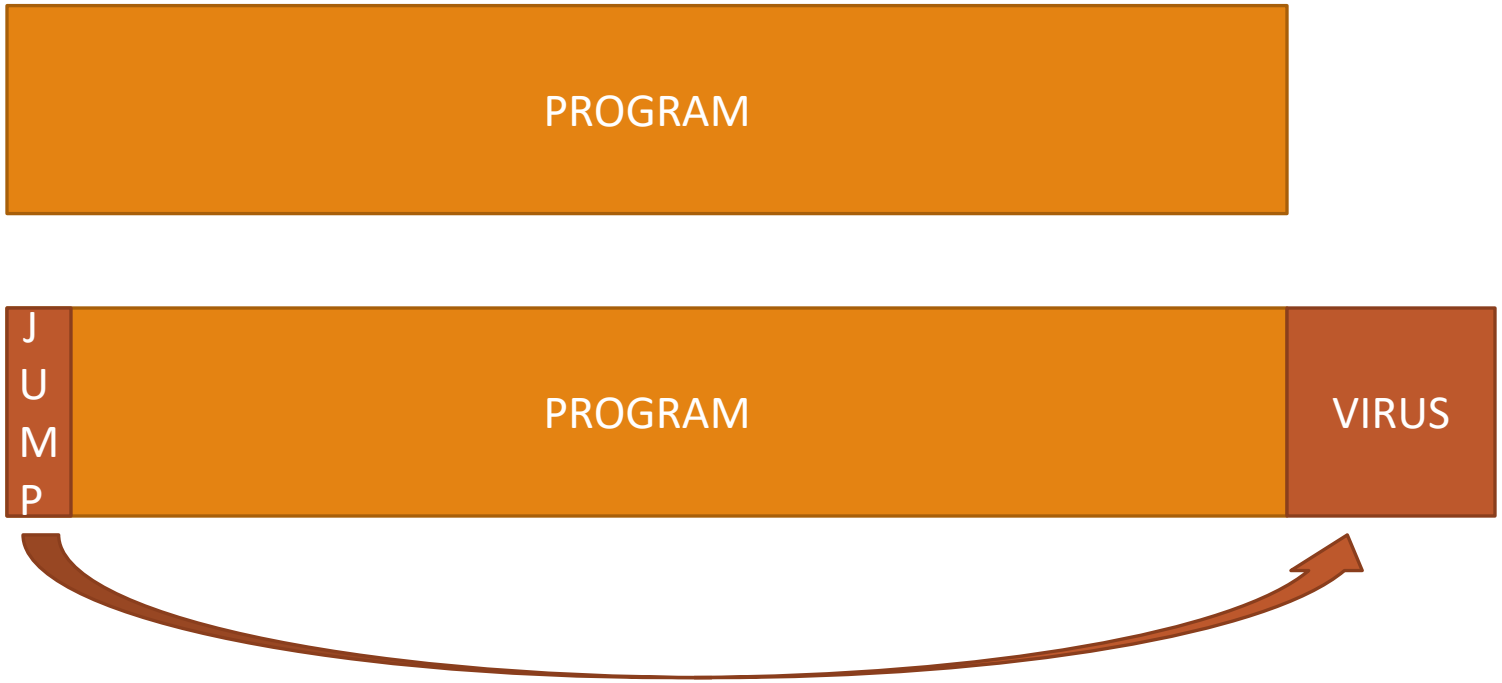
Risks of Cyberweapons

ONE: Everything is interconnected

TWO: Humanity is reliant on automation

THREE: Law of Unintended Consequences

Early DOS Viruses



Early “Anti Virus”

“Virus Bulletin” started in 1989

Still available at www.virusbulletin.com

Used to print ***BYTE SEQUENCES*** of known viruses

8 Tunes - CER: The virus probably originates in Germany and infects COM and EXE files. The length of the virus code is 1971 bytes. When triggered, it will play one out of eight different tunes. The virus attempts to deactivate two anti-virus programs: Bombsquad and Flushot+.

```
8 Tunes          33F6 B9DA 03F3 A550 BB23 0353 CB8E D0BC ; Offset variable
```

Virus Bulletin, January 1991

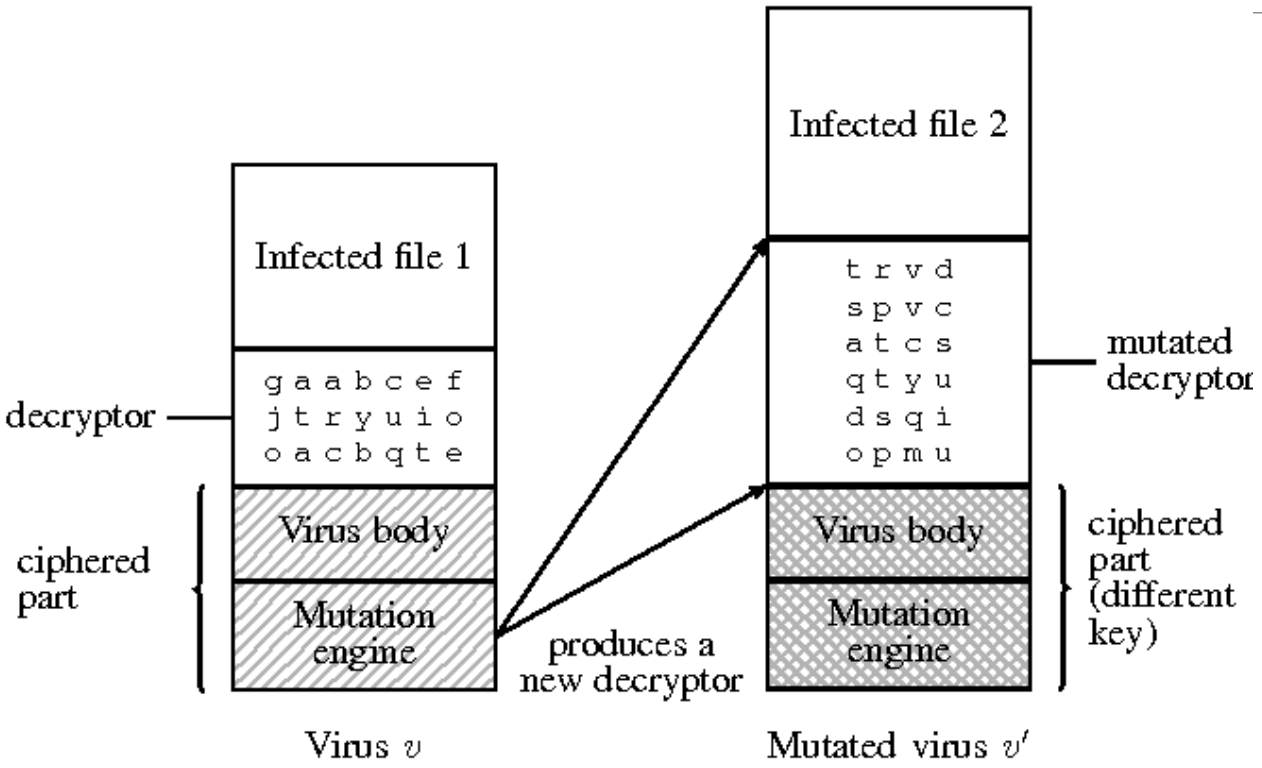
Virus Advancements

Antivirus scanners emerged with “libraries” of virus signatures

In response, viruses becamse “polymorphic”’

- Each infection encrypts virus under a different key
- Decryption engine decrypts virus for operations
- Encryption means that each infection has unique bytes

Polymorphic Virus Diagram



"Automated extraction of polymorphic virus signatures using abstract interpretation" by Chaumette and Tabary

Antivirus Solution Classes

Static Analysis

- Analyze the actual code of the virus

Dynamic Analysis

Solutions

Advanced Signatures

- Signature is not just a byte sequence
- Each “signature” is a mini-program of detection instructions

Partial Interpreter

- Virus usually takes control early
- Interpret the first bytes to see if its decrypting

Heuristics

- Look for “telltale” signs
- Minimally effective; too many false positives

Virus Delivery vs Payload

Much of the focus on Viruses is the delivery

How does it spread?

How does it bypass security?

Payload can be deleting

1984: Thompson's Reflections

"Reflections on Trusting Trust" by Ken Thompson, 1984

Demonstrated creating an evil compiler

Would compile a login program with a backdoor

BUT! Compiler code is compiled ***by a compiler***

"Clean" compiler source code compiled by an evil compiler ***is evil!***

Proved that a "source code review" can't catch all evil

SUPPLY CHAIN