

ECE560

Computer and Information Security

Fall 2024

Malware

Tyler Bletsch
Duke University

Introduction

MALWARE:

Software that violates confidentiality, integrity, or availability of a system.

- Two main questions:
 - How does it get onto a system?
 - What is it trying to achieve?

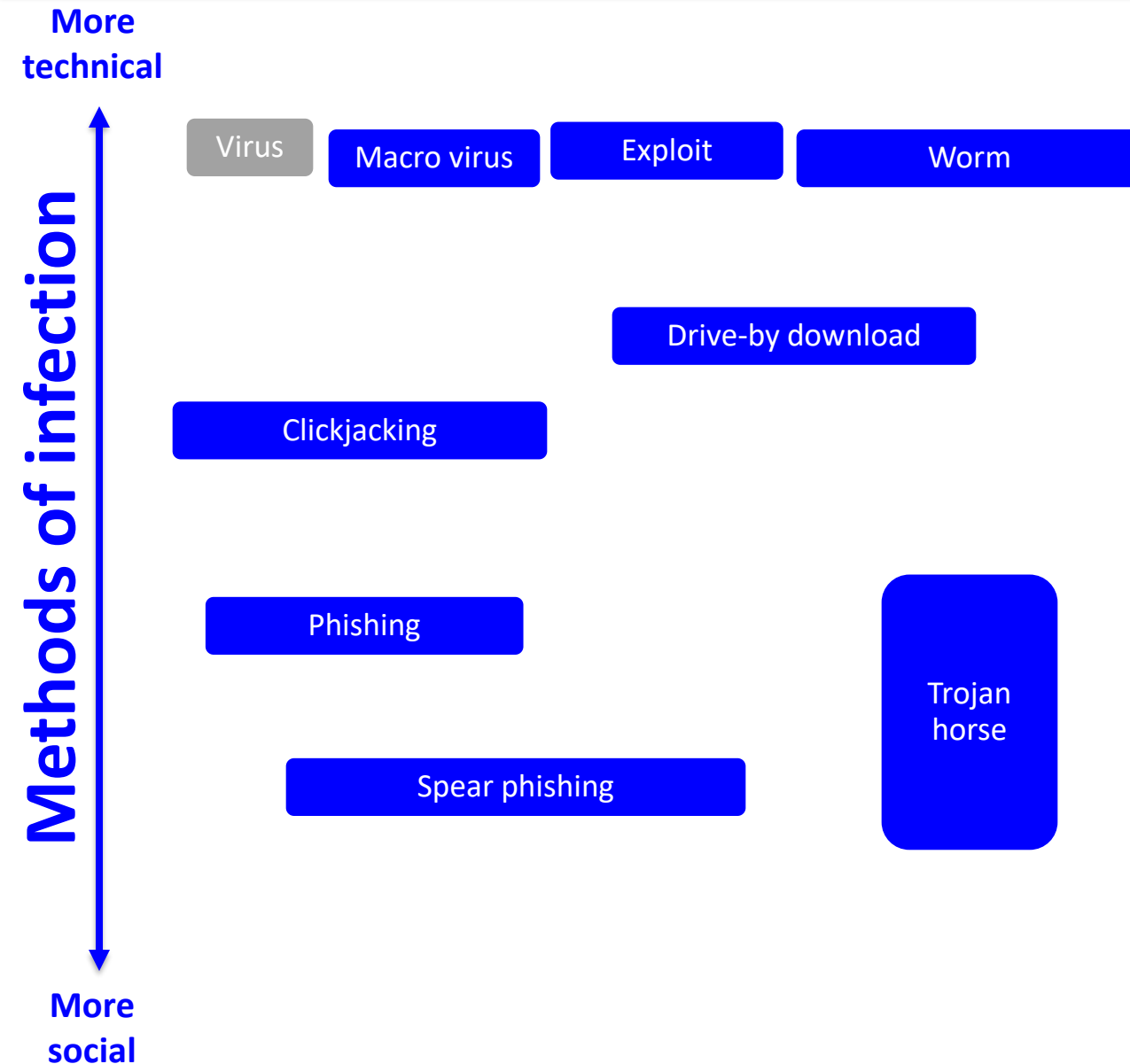
The universe of malware

Methods of infection



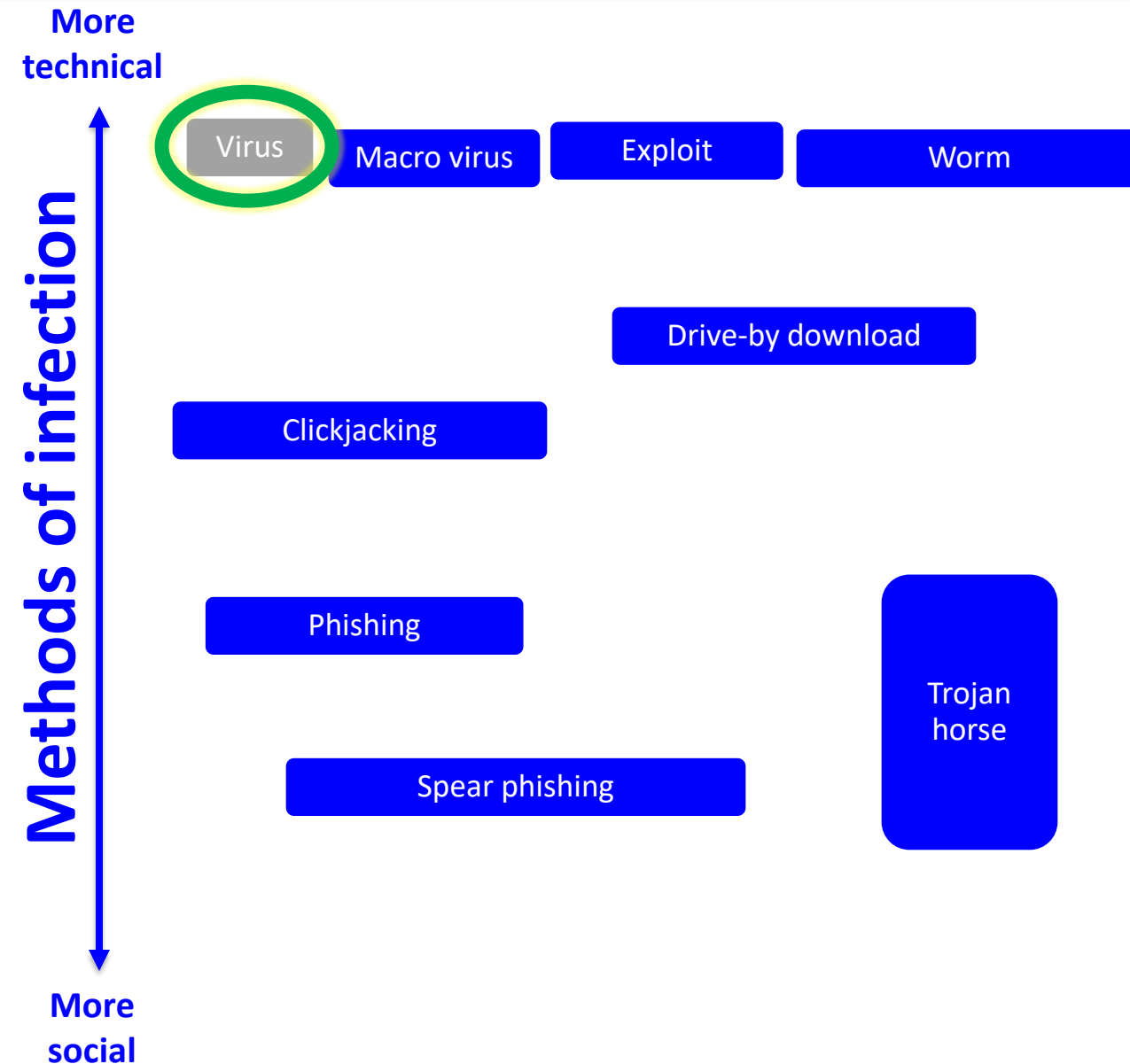
Goals of attacker

Vectors of infection



- Can classify by amount of technical engineering vs. social engineering

Vectors of infection



- Can classify by amount of technical engineering vs. social engineering

Viruses

- Software that “infects” (modifies) existing programs
 - Modifies executables to include code to spread further
 - Has same permissions as that program, runs in secret
 - Is OS- and platform-specific
- Classical targets of viruses
 - Boot sector virus – modifies launch firmware [OBSOLETE]
 - File infector – modifies executable files [DETECTABLE]
 - Macro virus – the “program” is automation code inside of documents [DYING]
- They may employ concealment strategies such as:
 - Encrypted virus – Code gets decrypted at launch, keeps hashes unique
 - Polymorphic virus – Mutates or changes with every infection

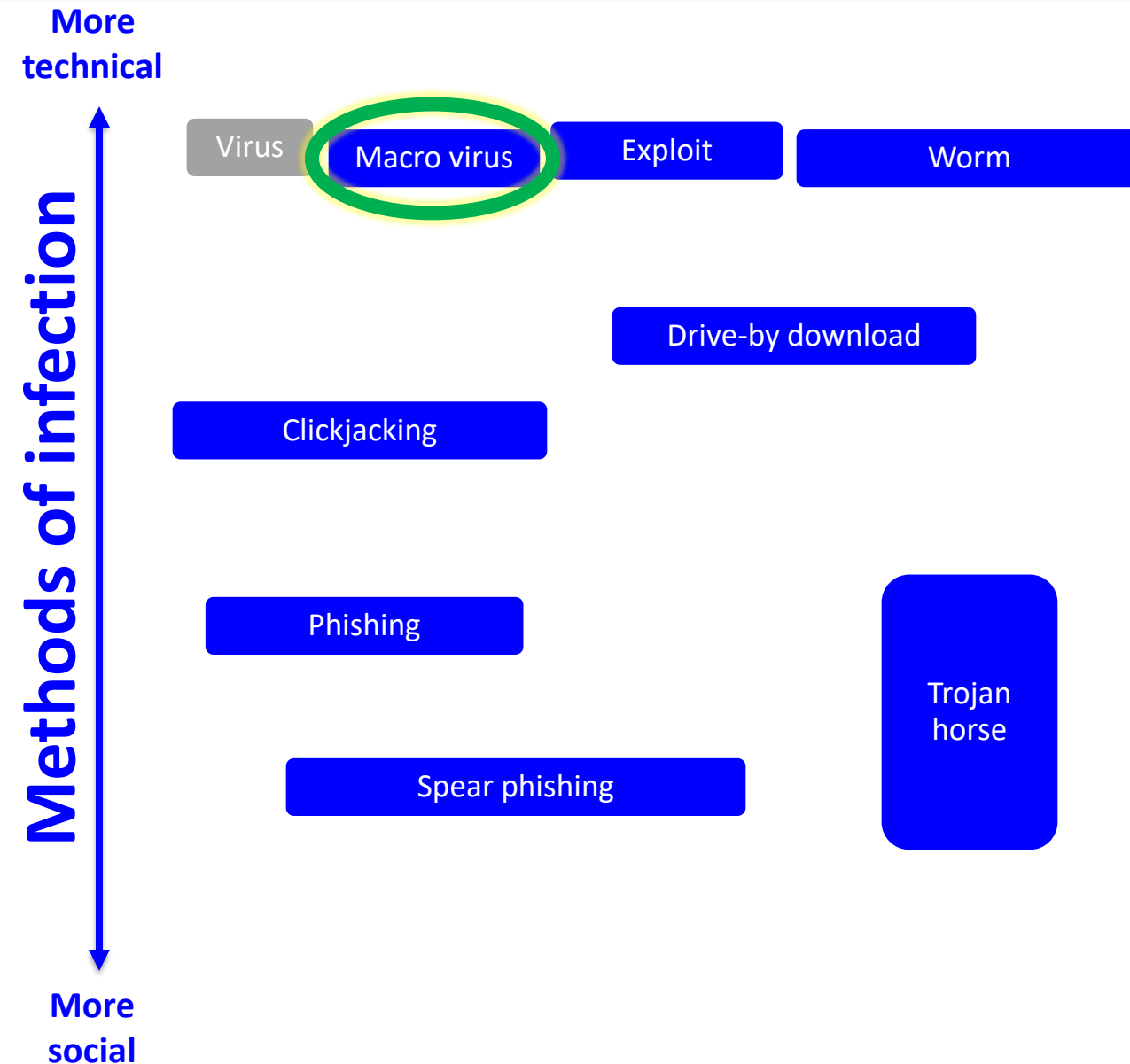
Viruses in the modern era

- Observation: *Viruses modify binary executables*
- Solutions?
 - Don't let unprivileged users modify binaries
 - Track hashes of binaries, notice when they change
 - Require cryptographic signing of binaries
- Bottom line: virus infection strategy is *peculiar*, can be detected
- Result? ***Classical viruses aren't really a thing in the modern era.***
 - ...but the uninformed press and non-computing public keeps using the term



**It's not
a virus.**

Vectors of infection

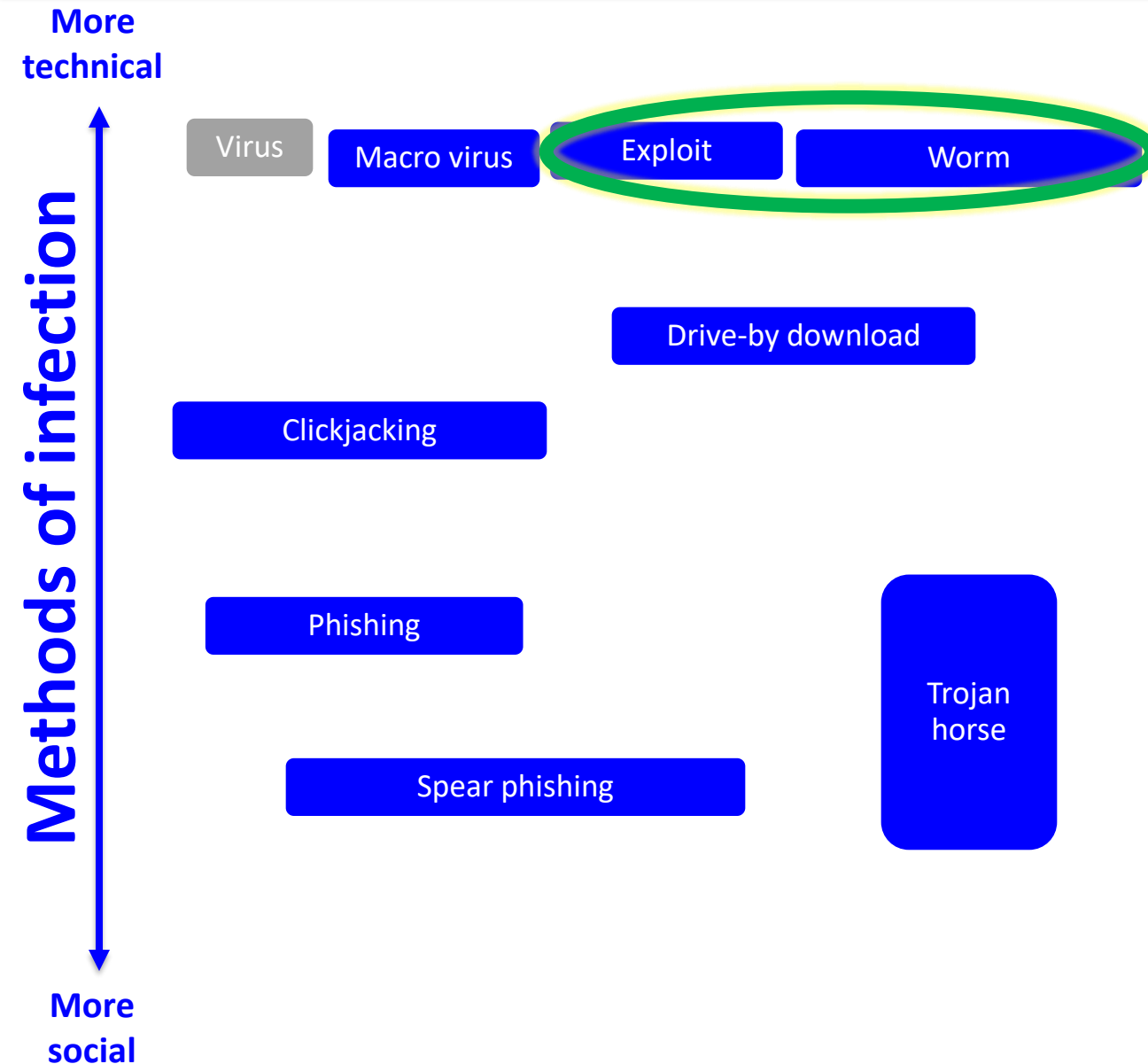


- Can classify by amount of technical engineering vs. social engineering

Macro viruses

- Many document formats have some form of scripting to allow custom automation, e.g. Microsoft Office
- Attackers make document macros that infect other documents when opened.
- **This category should be dead.** How hard is it to not let macros do dangerous stuff?
 - Kept alive by a need for backwards compatibility.
- Nowadays much harder:
 - Macros come with big giant warnings and, in Office, a separate file extension (.xlsm instead of .xlsx)
 - But you still see workarounds and macro-based attacks working sometimes 😞

Vectors of infection



- Can classify by amount of technical engineering vs. social engineering

Worms

- **Worm:** A program that seeks out more *machines* to infect
 - Each infected machine is a launching pad for attacks on other machines
- Methods of spread:
 - Exploit **software vulnerabilities** in client or server programs
 - Can use **network connections** to spread from system to system
 - Example: Web app bug allows uploading of new code
 - Example: SSH dictionary attack to infect bad credential'd hosts
 - Spreads through **shared media** (USB drives, CD, DVD data disks)
 - Example: Automatically write autostart executable to attached USB stick
 - Can include **social techniques** (email, instant messaging, etc.)
 - Example: Email to everyone in address book with “me-nude.jpg.exe”
- Usually carries some form of payload

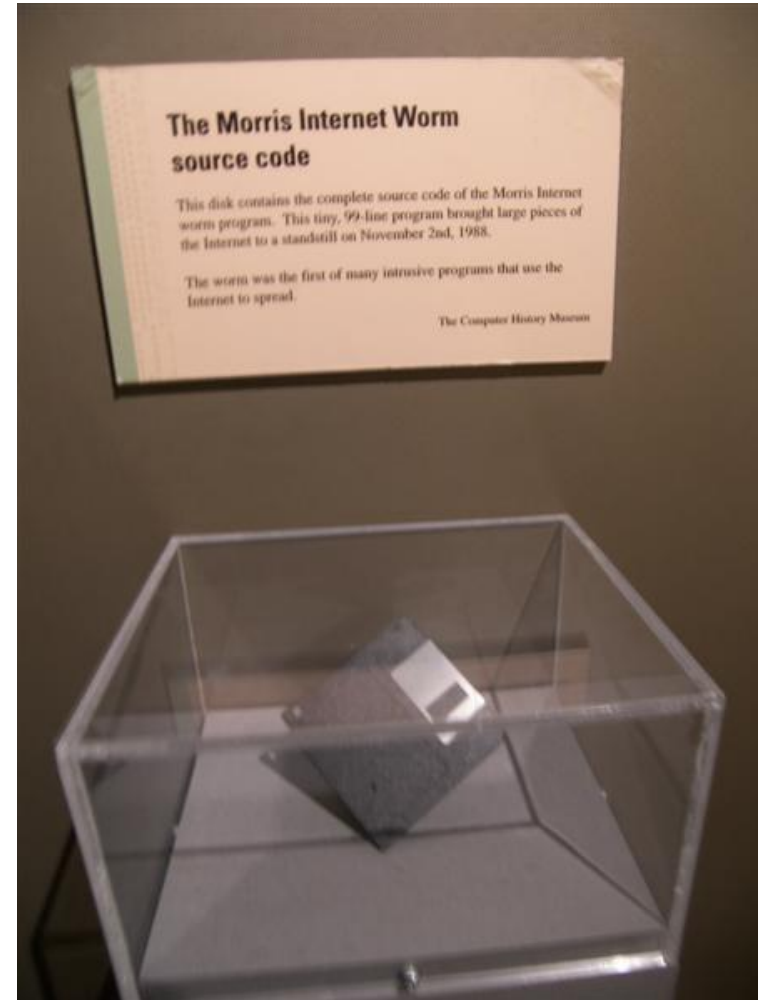
How a network worm tries to spread

Many possible strategies. Examples:

- **Random:** Each compromised host probes random addresses in the IP address space using a different seed. High traffic, may be inadvertently disruptive.
- **Hit-list:** The attacker first compiles a long list of potential vulnerable machines, includes in worm itself, infections scan part of the list.
 - Example of *targeted attack* – common if attacker wants to achieve something specific with stealth (e.g. the Stuxnet worm)
- **Topological:** Use info in or about the victim machine, such as “automagic” file sharing services
- **Local subnet:** Target hosts nearby on network; especially good if the worm lands behind a NAT or firewall

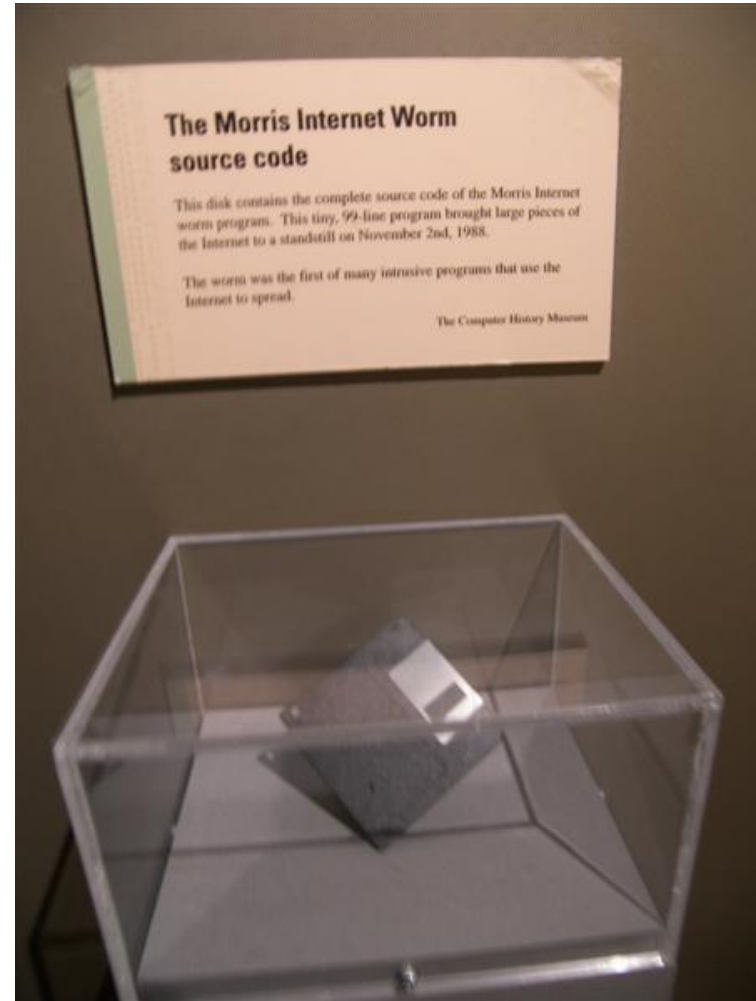
Malware history: the Morris worm (1988)

- First major worm on the internet
- Released by Robert Morris in 1988 as an experiment
 - The experiment got out of hand...
- Attributes:
 - Dictionary-attacked weak passwords
 - Exploited bugs in “fingerd” (user info tool) and “sendmail” (email server)



Morris Worm, continued

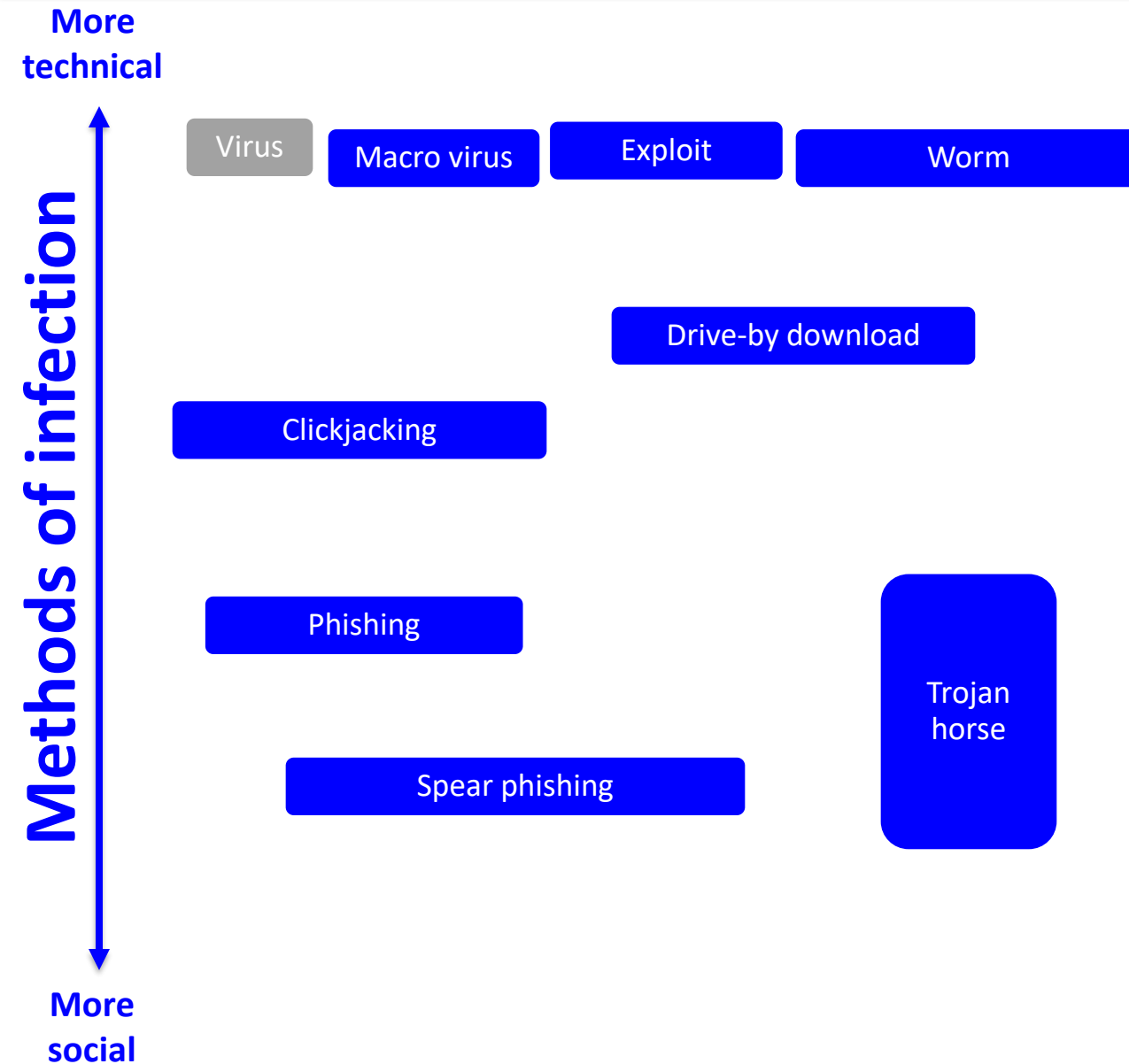
- Not supposed to cause damage
 - Had an intentional 1 in 7 chance of re-infecting an infected system in case the already-infected detector had been fooled
 - This was dumb due to math
 - Damage due to re-infection choking system with thousands of worm processes
- Internet was literally segmented during cleanup
- First conviction under the 1986 Computer Fraud and Abuse Act
 - Probation, community service, and \$10k fine



Worms today

- Worms are alive and well today, but the term “worm” is less common
- Reason: increasing monetization of attacks
 - Example: The 2017 “WannaCry” attack was a worm that encrypted data and demanded a bitcoin payment
 - It’s both **ransomware** and a **worm**, people call it “ransomware” more

Vectors of infection



What about
rootkits?

What is a rootkit?

- How do you tell if a system is running process X?
 - Ask the OS (e.g. the **ps** command)
 - What if the OS lies???????
- **Rootkit**: A program that uses root privilege to modify the running operating system's behavior
 - This implies you have root privilege!
(Achieved by another attack or rootkit exploits an OS bug to get root)
- Change kernel code or data to change behavior of system calls
- **Not a method of infection**; it's a method of stealth and continued access (back door)!

"Runs at boot" doesn't imply rootkit – needs to mess with OS behavior!

Rootkit properties

- Persistent vs. in-memory:
 - **Persistent:** Activates on system boot; requires persistent storage. Can be easier to detect (can look at storage offline).
 - **In-memory:** No persistent code (can't survive a reboot). Can be harder to detect (have to look at RAM; usually need OS to do so).
- Location:
 - **User mode:** Replace system tools (ls, cat, etc.) or their shared libraries.
 - Example: LD_PRELOAD on Linux -- put a custom library in front of any executed program; can catch all libc calls.
 - **Kernel mode:** Modify kernel memory; can control all syscalls.
 - **Virtual machine based:** Install a lightweight hypervisor and run the operating system in a virtual machine.
 - **External:** Control something outside the plain CPU, such as the BIOS or system management mode, so it can directly access hardware.

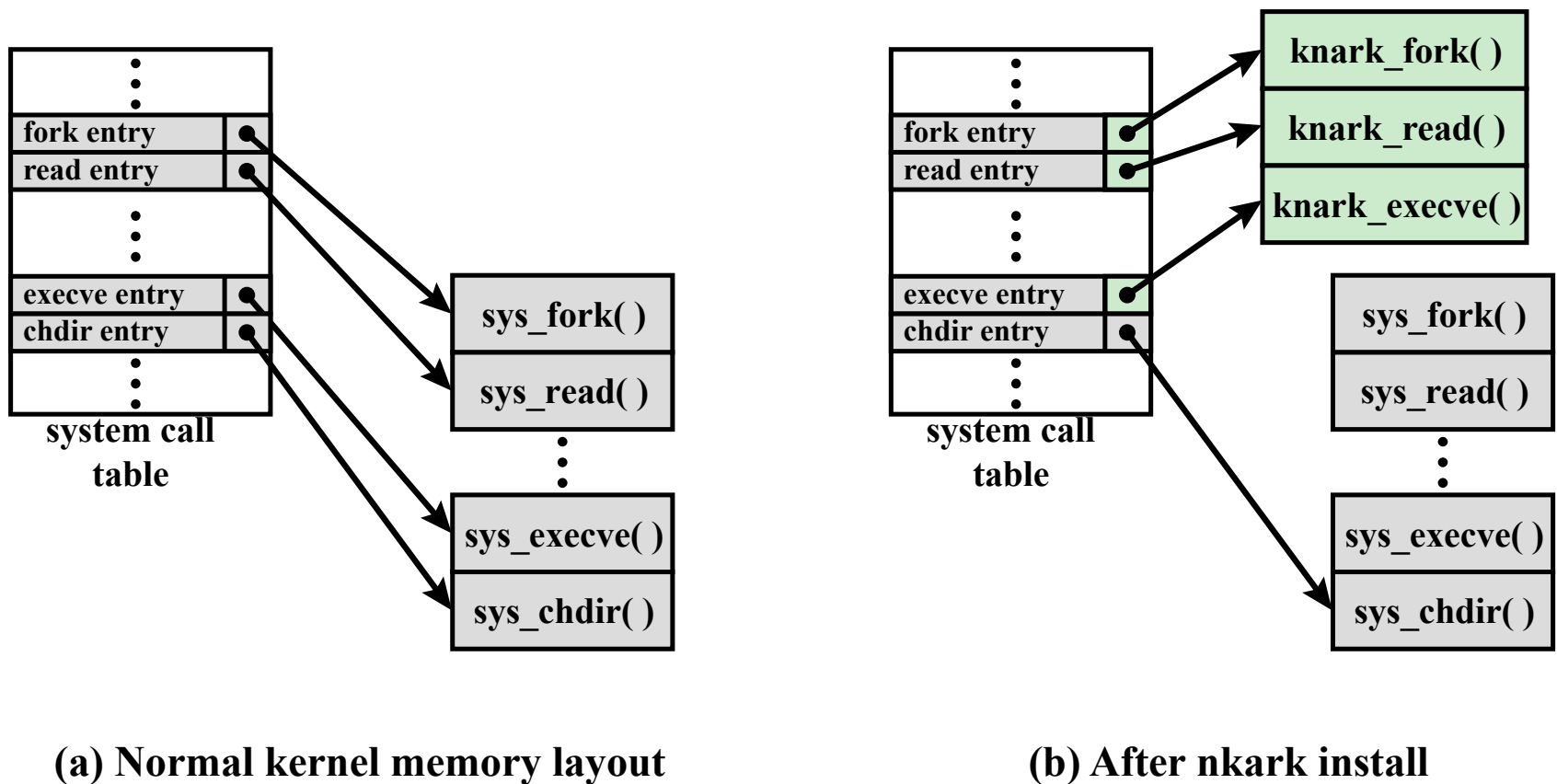


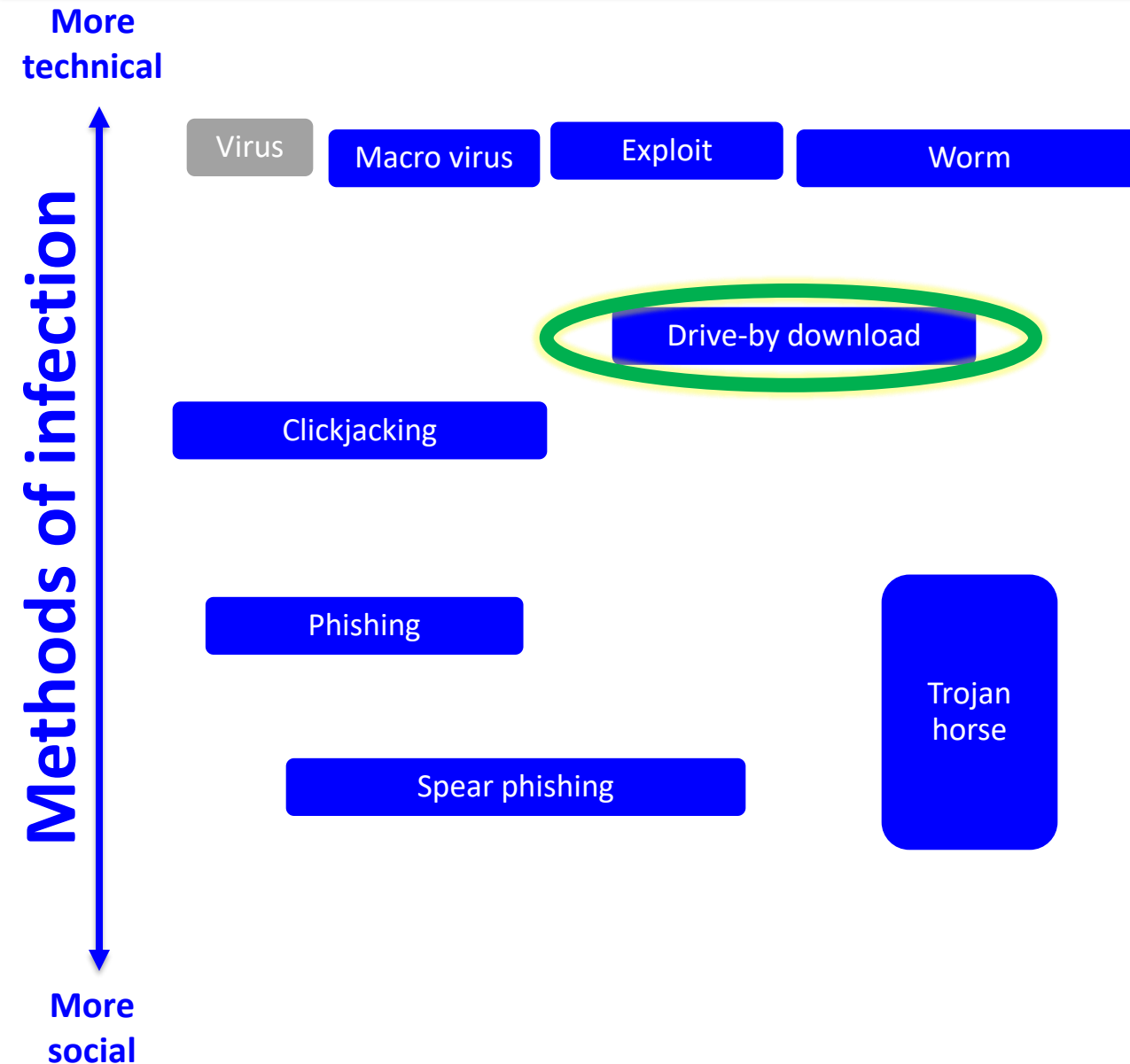
Figure 6.4 System Call Table Modification by Rootkit

Example of a kernel rootkit

```
-bash
tkblets@engr-ras-101:~$ ps -A | head -30
PID TTY          TIME CMD
  1 ?            00:00:01 init
  2 ?            00:00:00 kthreadd
  3 ?            00:00:00 migration/0
  4 ?            00:00:00 ksoftirqd/0
  5 ?            00:00:00 stopper/0
  6 ?            00:00:00 watchdog/0
  7 ?            00:00:04 events/0
  8 ?            00:00:00 events/0
  9 ?            00:00:00 events_long/0
 10 ?            00:00:00 events_power_ef
 11 ?            00:00:00 cgroup
 12 ?            00:00:00 khelper
 13 ?            00:00:00 netns
 14 ?            00:00:00 async/mgr
 15 ?            00:00:00 pm
 16 ?            00:00:00 sync_supers
 17 ?            00:00:00 bdi-default
 18 ?            00:00:00 kintegrityd/0
 19 ?            00:00:04 kblockd/0
 20 ?            00:00:00 kacpid
 21 ?            00:00:00 kacpi_notify
 22 ?            00:00:00 kacpi_hotplug
 23 ?            00:00:00 ata_aux
 24 ?            00:00:17 ata_sff/0
 25 ?            00:00:00 ksuspend_usbd
 26 ?            00:00:00 khubd
 27 ?            00:00:00 kseriod
 28 ?            00:00:00 md/0
 29 ?            00:00:00 md_misc/0
tkblets@engr-ras-101:~$ . infect
```

```
tkblets@engr-ras-101:~$ ps -A | head -30
  1 ?            00:00:00 top
  2 ?            00:00:00 ps
  4 ?            00:00:00 grep
  8 ?            00:00:00 LOL
 16 ?            00:00:00 if
 32 ?            00:00:00 you
 64 ?            00:00:00 think
128 ?            00:00:00 this
256 ?            00:00:00 process
512 ?            00:00:00 list
1024 ?           00:00:00 is
2048 ?           00:00:00 remotely
4096 ?           00:00:00 valid.
8192 ?           00:00:00 you
16384 ?          00:00:00 got
32768 ?          00:00:00 Owned
65536 ?          00:00:00 by
131072 ?         00:00:00 l33t
262144 ?         00:00:00 haxors
524288 ?         00:00:00 ps
1048576 ?        00:00:00 grep
tkblets@engr-ras-101:~$
```

Vectors of infection



- Can classify by how amount of technical engineering vs. social engineering

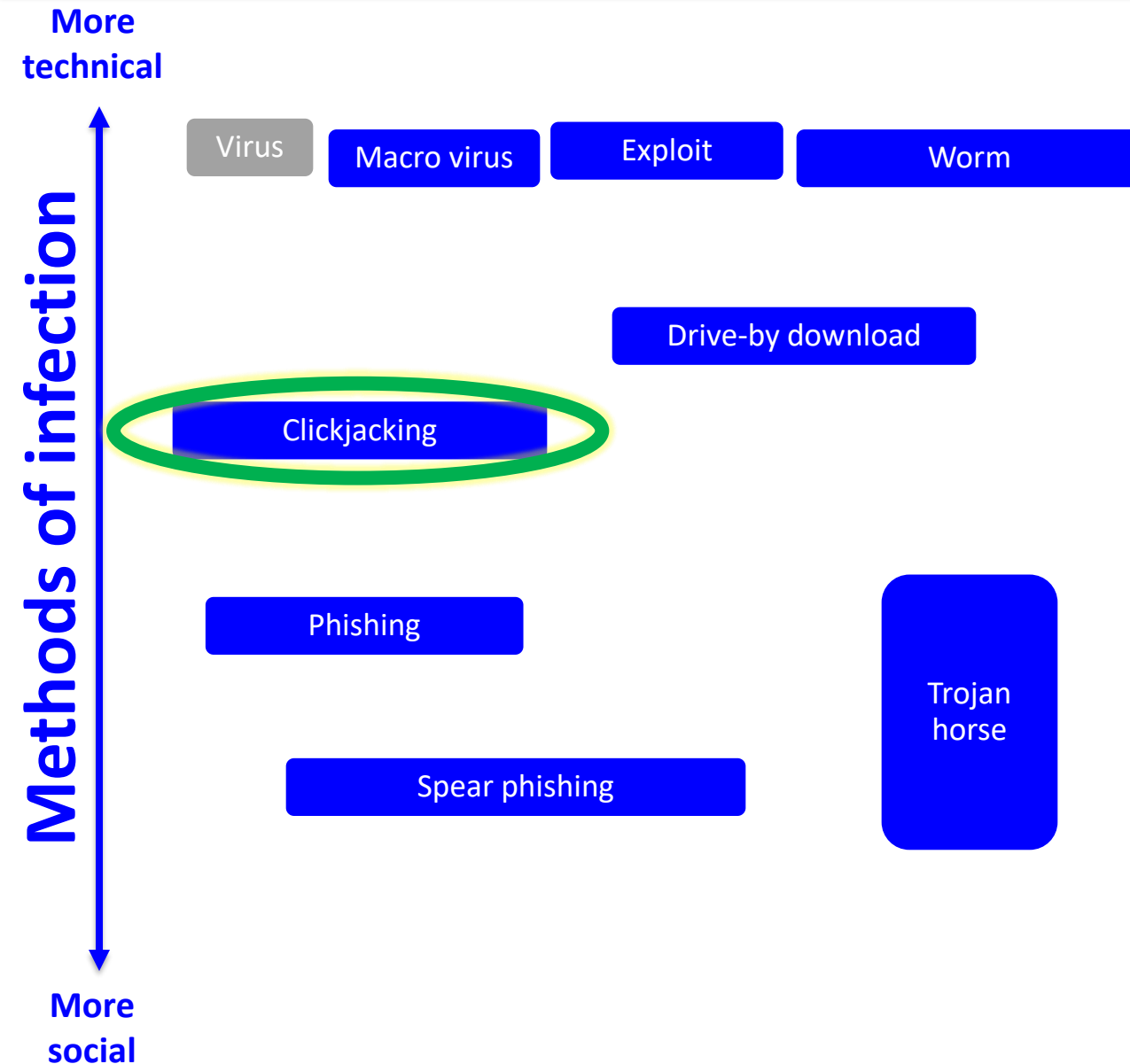
Drive-By Downloads

- Exploit browser vulnerabilities to download and installs malware on the system when the user views a Web page controlled by the attacker
 - Usually happens automatically and invisibly
- This is stupid – why are there so many critical browser vulnerabilities that this kind of attack was given a name?



- It's happened on other browsers, but not as much by far...

Vectors of infection



- Can classify by how amount of technical engineering vs. social engineering

Clickjacking

- What do you use to authorize something on your computer?

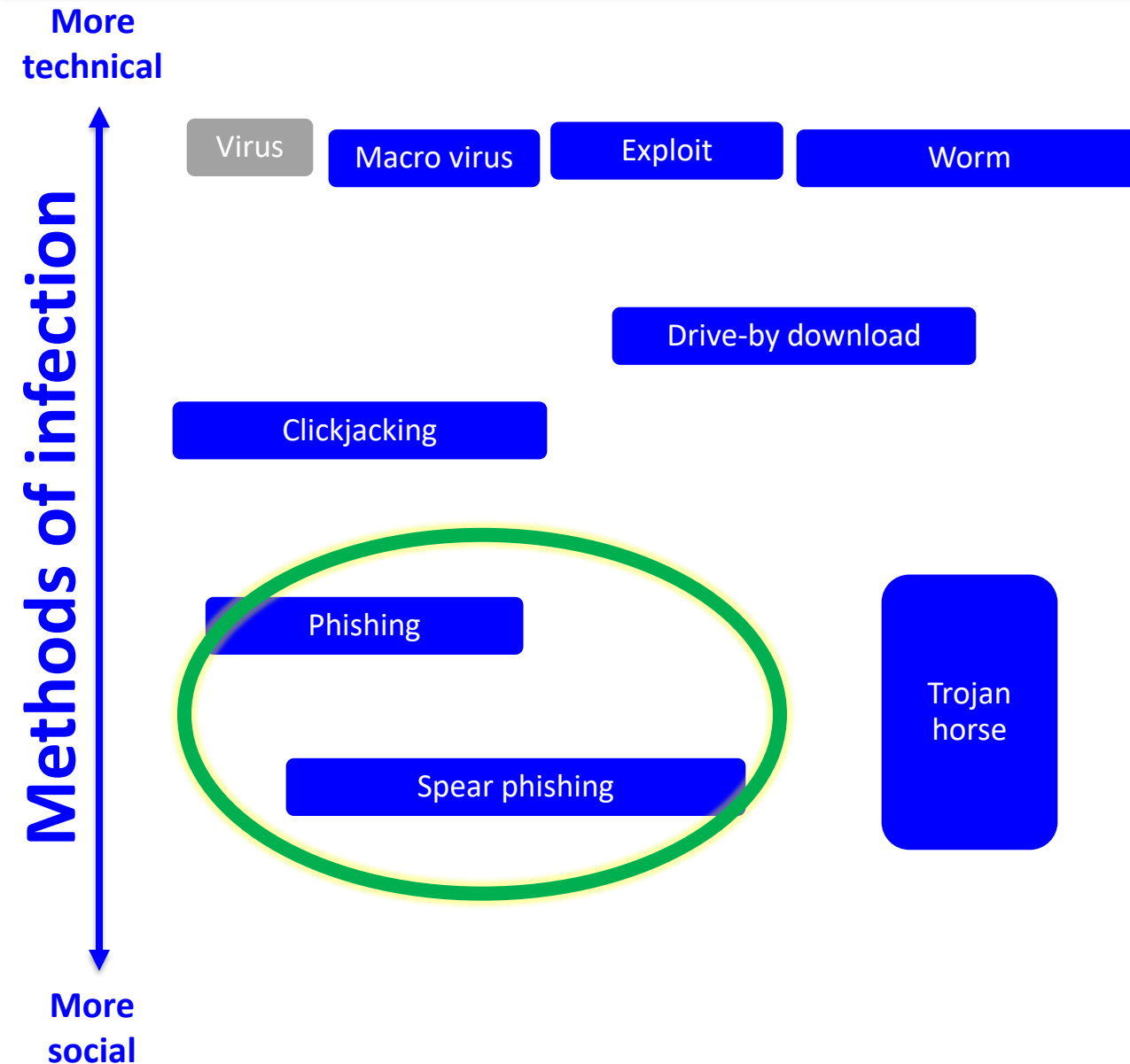


- **Clickjacking:** Creating a situation where a user will give a mouse or keyboard input to do one action, but due to timing or circumstance, it will actually perform a different action
 - Timing example: Click the puppy to continue! *Download confirmation appears with “OK” button right on top of the puppy in 200ms*
 - Circumstance example: An authorization website is loaded as transparent on top of an unrelated interface, so that clicking the puppy actually creates a user account on another system.

Clickjacking demo

- Simple web click harvesting:
 - <https://www.youtube.com/watch?v=gxyLbpIdmuU>
- A more personalized example:
 - <http://people.duke.edu/~tkb13/courses/ece560/resources/clickjackdemo.html>
(Tested Oct 2023, incognito, Chrome on Windows)
- This is why the browser waits before it lets you say yes to certain things

Vectors of infection



- Can classify by how amount of technical engineering vs. social engineering

Phishing

- **Phishing:** A social engineering attack where the attacker pretends to be a trusted source, induces victim to take an action
 - Possible “sources”: Your IT department, a voicemail system, a cloud storage provider, a friend or colleague, an authority at your company, etc.
 - Possible actions: Click a link, open an attachment, reply with info, change a setting, transfer money, run a program (like a trojan horse – next topic!), etc.



[img src](#)

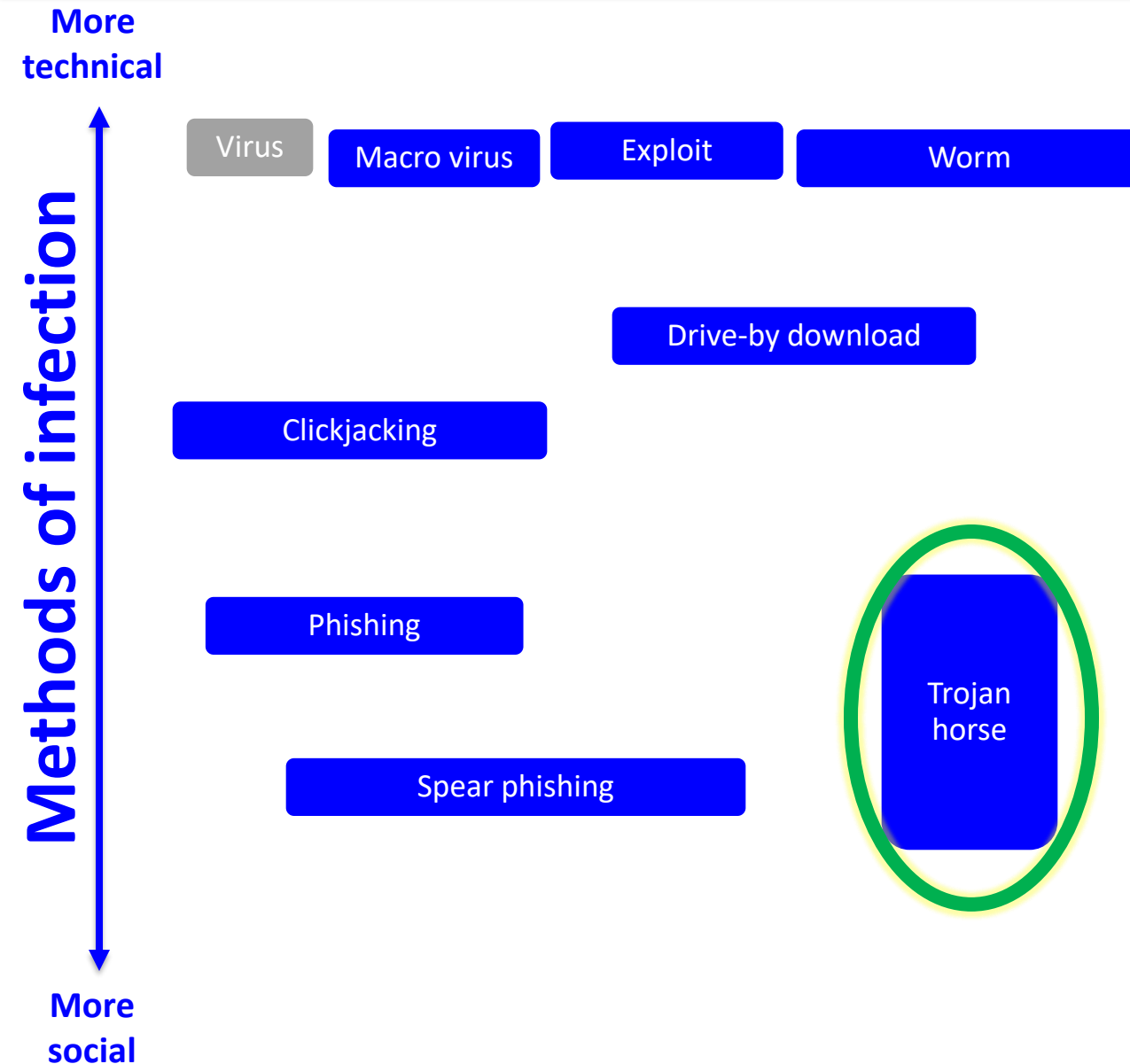
It's not just about stealing credentials!

- **Spear phishing:**
 - *Normal* phishing is usually broadcast to large number of potential victims
 - *Spear* phishing is specific and targeted
 - Create a deeper narrative for specific victim(s)
 - Leverage facts already found from other investigation/attacks
- We'll cover this more when we discuss social engineering in depth

Phishing defenses

- Users:
 - Be skeptical of all communications
 - Be aware of what you make public (that's background for attackers!)
 - Inspect the link, and if in doubt, don't click it
 - For links to known services, just go there yourself
- Organizations:
 - Train users in the above
 - Have clear chain-of-command procedures so that a random email won't be an expected method of doing anything critical
 - Deploy email **link screening system** (e.g. the "urldefense" links you see in Duke email)
 - Allows you see all outgoing clicks
 - Can spot trends, black hole those links at the forwarding stage

Vectors of infection



- Can classify by how amount of technical engineering vs. social engineering

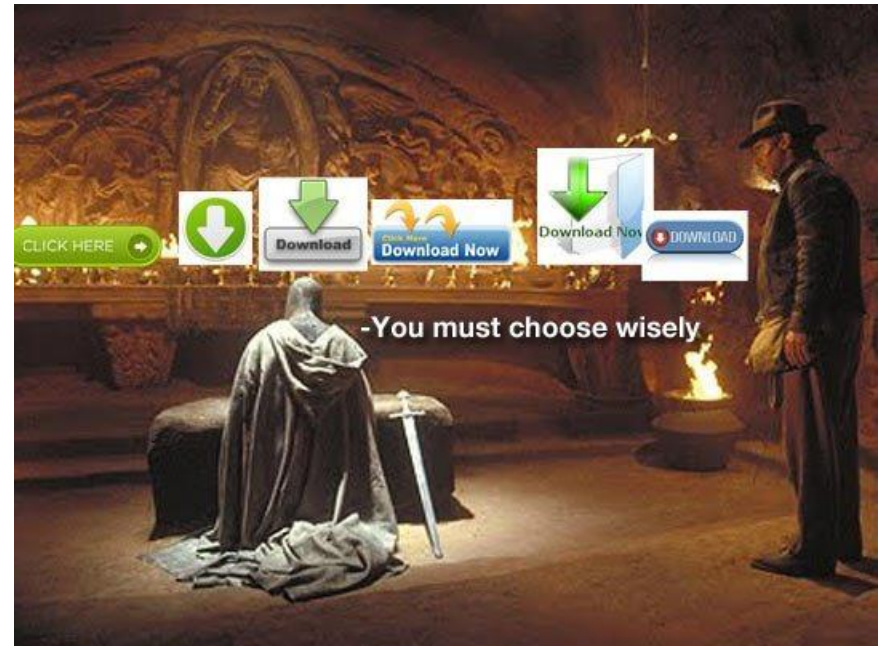
Trojan horse

What if, instead of all that complicated stuff, we just got people to just, like, run the malware themselves?



Trojan horse

- **Trojan horse:** Malware that the attacker tries to get the victim to run themselves.
- Example approaches:
 - Pretend to be installer for a program the user wants
 - Better: actually **be** the installer, but with something extra...
 - Pretend to be a **non-program**, e.g. the classic “adorablecat.jpg.exe”
 - Make a web ad that looks like what the user wants →
 - Pretend to be a necessary step to something else, e.g. “you must install MalMeeting plugin to attend this webinar”



Summary of commonly confused malware terms

- **Viruses:** Infect executables to spread
- **Worms:** Infect machines to spread
 - Via exploits or automated social attacks
- **Trojan:** Infect machines if you're dumb enough to run them
- **Rootkit:** Infect kernels to avoid detection/removal
 - Requires root access (or privilege escalation exploit to achieve root access)

The universe of malware

Methods of infection

Goals of attacker




Goals of attacker (1)

- Remote access
 - **Back door:** Maintain access
 - **Network access:** Penetrate private network (e.g. a tunnel or VPN)
 - **Spyware:** Gather info on user/system activities
 - **Keylogger:** Specifically monitor keyboard (passwords, bank info, etc.)
 - **Data theft:** Steal data, either generally or in a targeted way
 - **Scams:** Use malware as part of a larger scam to get user money/info, e.g. the currently common “Microsoft support” scam
- **Business Email Compromise (BEC)**
 - Usually an extended, personalized attack:
 1. **Gain access** to a victim via spearphishing, malware, or other technique(s)
 2. **Learn** about the business: who makes decisions, how they’re communicated
 3. **Impersonate** a decision maker (and possibly others) via email to arrange a money transfer or other profitable action

Goals of attacker (2)

- **Zombie/Botnet:** Enlist victim as a node in a network of victims
 - **Distributed Denial of Service (DDOS):** Have all bots flood a target
 - Spam sender
 - Mine stupid cryptocurrencies
- Endpoint attacks
 - **Ransomware:** Encrypt files and sell the user back the key to decrypt
 - This one is HUGE right now!!
 - Usually demand payment in *Bitcoin* or other cryptocurrency
 - **Adware:** Inject ads into normal browsing or just have them pop up
 - **Damage/defacement:** Just mess things up, sign the attacker's work, etc.
 - **Charge for services:** Use SMS, phone, pay services, etc. to rack up bills
- **Attack kit:** Download tools to further an active intrusion

The universe of malware



But who?

Classes of attackers

- **Explorer:** An individual just testing things
 - Increasingly rare...
- **Criminal:** Out for money (either directly or indirectly)
 - Increasingly common...
- **Hacktivist:** Political motivation
- **Advanced Persistent Threat (APT)**
 - Nation-states and large collective organizations
 - *Advanced:* Has access to unpublished vulnerabilities and custom tools; will deploy multiple malware systems in a concerted attack
 - *Persistent:* Has specific targets of interest and will work on them over time
 - *Threats:* Did you read the above two?

Who APTs are and what they do

- Gets paid in bitcoin? You're a **criminal**.
- Work in an office; get paid by the government? You're in an **APT**.



More on APTs

- They've been growing in recent years
- Analysts number or name a groups and attempt to track them
- Example:
 - **Fancy Bear** is also known as **APT28** (by Mandiant), **Pawn Storm**, **Sofacy Group** (by Kaspersky), **Sednit**, **Tsar Team** (by FireEye) and **STRONTIUM** (by Microsoft)
 - Russian APT group that hacks groups/resources associated with western democracies, such as the U.S. Democratic National Committee
 - Identified in 2018 by the U.S. as Russian GRU Unit 26165
- Many, many more. They don't advertise, so attribution is hard.

The universe of malware



Defenses?



Optimal defense: prevention

- Prevention is best, but prevention is also hardest.
- Keys to malware prevention:
 - **Policy:** Adopt preventative maintenance policies and clear procedures for how systems are used
 - **#1 example:** Keep systems up to date with security updates!
 - **Awareness:** Most infection methods involve some amount of social engineering
 - Training reduces effectiveness of the social dimension
 - **Vulnerability mitigation:** Deploy defenses against *classes of vulnerability*
 - Example: Software can be written to do no runtime memory allocation, thus eliminating the possibility of memory allocation bugs (e.g., Wireguard VPN)
 - **Threat mitigation:** Deal with specific malware threats and behavior patterns
 - This includes **Host-Based Intrusion Detection Systems (HIDS)**, **Network-Based Intrusion Detection Systems (NIDS)**, and other things we'll cover later in the course

What about anti-virus?

- Slide from the text book:

Worm Countermeasures

- Considerable overlap in techniques for dealing with viruses and worms
- Once a worm is resident on a machine anti-virus software can be used to detect and possibly remove it
- Perimeter network activity and usage monitoring can form the basis of a worm defense
- Worm defense approaches include:
 - Signature-based worm scan filtering
 - Filter-based worm containment
 - Payload-classification-based worm containment
 - Threshold random walk (TRW) scan detection
 - Rate limiting
 - Rate halting



True?

Rootkits ruin everything (1)

- If *any* malware has ever *touched* a system, the following is possible:
 - The malware may have exploited an unpublished kernel vulnerability to get root access
 - Using this access, it may have installed a rootkit, rendering all OS calls suspect
 - RESULT: Even if it looks like known malware, that may be a ruse, and your kernel has a persistent infection that you cannot detect
- CONCLUSION: “Flatten and reinstall”
 - Destroy all data and restore from install files and/or known-good backup

Rootkits ruin everything (2)

- Higher levels of paranoia can also exist:
 - If a bare metal computer, malware may have exploited a hardware vulnerability to install outside the OS, e.g. the system firmware, hard disk firmware, video card firmware, out-of-band management system, etc.
 - If a VM, malware may have exploited a hypervisor vulnerability to take over the hypervisor and may be in command of *all* VMs (and may do the hardware stuff listed above).
- In that case...

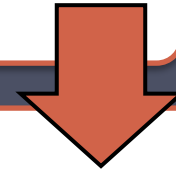


You gotta trash the box

Generations of Anti-Virus Software

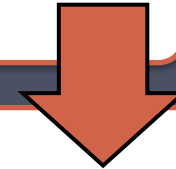
First generation: simple scanners

- Requires a malware signature to identify the malware
- Limited to the detection of known malware



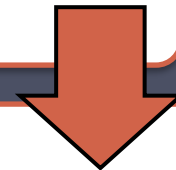
Second generation: heuristic scanners

- Uses heuristic rules to search for probable malware instances
- Another approach is integrity checking



Third generation: activity traps

- Memory-resident programs that identify malware by its actions rather than its structure in an infected program



Fourth generation: full-featured protection

- Packages consisting of a variety of anti-virus techniques used in conjunction
- Include scanning and activity trap components and access control capability

Beyond antivirus

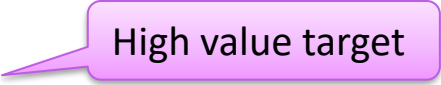
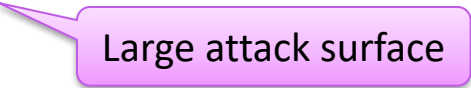
- Commercial antivirus has a bad reputation, OS vendors provide free equivalents today that are quite good
- Has evolved into “**Endpoint Detection and Response**” (EDR), does the usual basic signature-based scanning plus more.
- Key differences:
 - **Enterprise audience:** Deployed in an organization rather than by individuals
 - **Phones home:** Monitors and alerts security team based on detected behavior, heuristics
 - **Ensures compliance:** Watches OS state (e.g., are you installing OS updates?)
 - **Config management:** Allows security team to remotely change config or policy of machines
- Example: *CrowdStrike*, installed by Duke on many machines

The anti-malware cat-and-mouse game

- **Attacker:** I'll encrypt my malware with a random key:
First part of code decrypts the rest, foils signature checking
- **Defender:** What if we run it in a CPU emulator and see if the code transforms into something recognizable? ("**Generic Decryption**")
- **Attacker:** I will detect that I'm running in an emulated environment
- **Defender:** I will add complexity to make the simulation more authentic
- **Attacker:** I will look for bugs in your complex emulator
- **Defender:** I will **sandbox** the whole detection engine
(disabling most system calls)
- **Attacker:** I will look for a sandbox bypass

etc., etc.

An alternative take on anti-virus software (1)

- Modern anti-virus is weak against *novel* threats, and in some cases can be harmful!
- Antivirus software must:
 1. Hook in at kernel level High value target
 2. Parse and process every piece of code and data you see using a wide variety techniques (lots of code) Large attack surface
- Modern tools can get worse...
 - Sometimes installs root CA or has browser dump encryption internals to intentionally man-in-the-middle all SSL traffic to scan it!
- Result: In antivirus software,

QUALITY MATTERS!

Example: a tale of two antiviruses

- Symantec/Norton



[Reference](#)

Also, a here's
[a separate Symantec privilege escalation bug](#)

- Microsoft Defender

- Runs malware analysis entire in a sandbox (code unable to use most system calls)

[Reference](#)



An alternative take on anti-virus software (2)

- For consumers, vendors push software into pre-installed market with limited “subscription” to updates, fleece buyers into paying for subscription
 - Basically, extortion
- Both Windows and Mac have built-in protections now that obviate this...don't pay the consumer antivirus mafia.
- For organizations, it's better...
Endpoint Detection and Response (EDR) systems answer to the security team

Perimeter scanning

Various approaches to look for malware at network borders:

- **Network Intrusion Detection Systems (NIDS):** Scanning network traffic content for malware signatures (covered later in the course)
- **Protocol-specific checks.** For example, for email: scanning for malicious attachments, intercepting/blocking email links, etc.
- **Honeypots:** Fake targets to attract attackers – can trigger automated reactions, such as blocking origin traffic

Summary

- Methods of infection
 - Virus
 - Worm
 - Trojan horse
 - Drive-by download
 - Phishing/Spear phishing
 - Clickjacking
 - Exploit
- Rootkits
- Attacker types
 - Includes **APT**s
- Attacker payloads
 - Back door
 - Spyware
 - Zombie/Bot
 - DDOS
 - Ransomware
 - Adware
- Countering
 - Policy
 - Awareness
 - Vulnerability mitigation
 - Threat mitigation