CMSC414 Computer and Network Security

Firewalls, Malware

Yizheng Chen | University of Maryland
surrealyz.github.io

April 30, 2026

Agenda

- Firewalls
- Malware
- Viruses
- Worms
- Infection cleanup and rootkits

Overview: Denial of Service and Firewalls

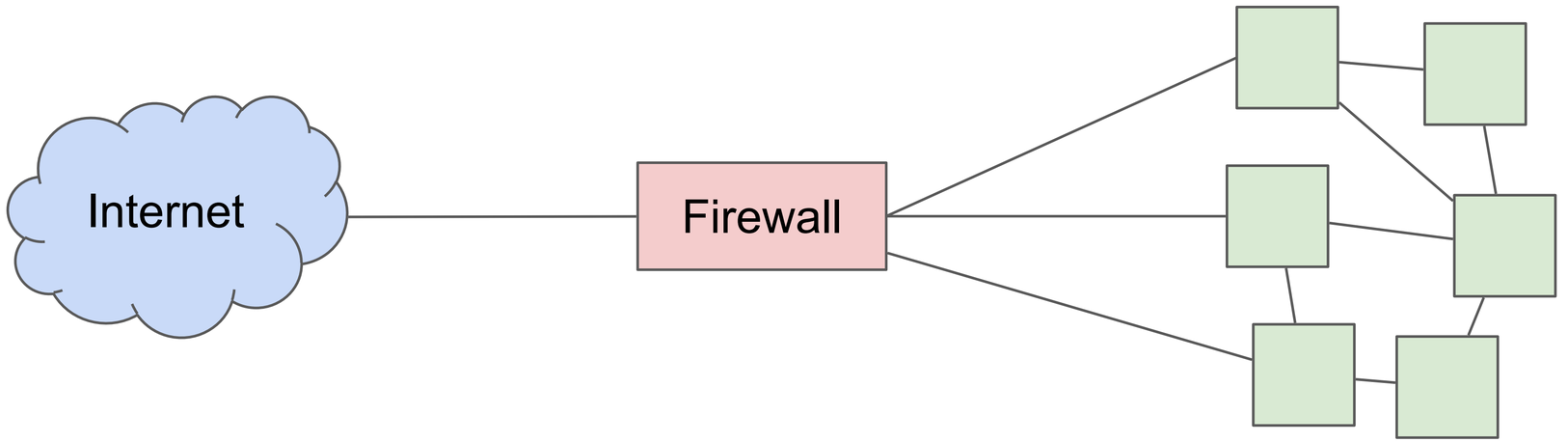
- Denial of service
  - Availability
  - Application-level DoS
    - Algorithmic complexity attacks
  - Network-level DoS
    - Distributed DoS (DDoS)
    - Amplified DoS
  - SYN flooding
    - SYN cookies
  - Defenses
- Firewalls
  - Packet filters
    - Stateless/stateful packet filters
  - Proxy firewalls

Motivation: Scalable Defenses

- How do you protect a set of systems against external attack?
  - Example: A company network with many servers and employee computers
- Observation: More network services = more risk
  - Each network connection creates more opportunities for attacks (greater attack surface)
  - Turning off all network services is often infeasible (print services, SSH services, etc.)
- Observation: More networked machines = more risk
  - What if you have to secure hundreds of systems?
  - What if the systems have different hardware, operating systems, and users?
  - What if there are some systems in the network that you aren’t aware of?
- Instead of securing individual machines, we want to secure the entire network!

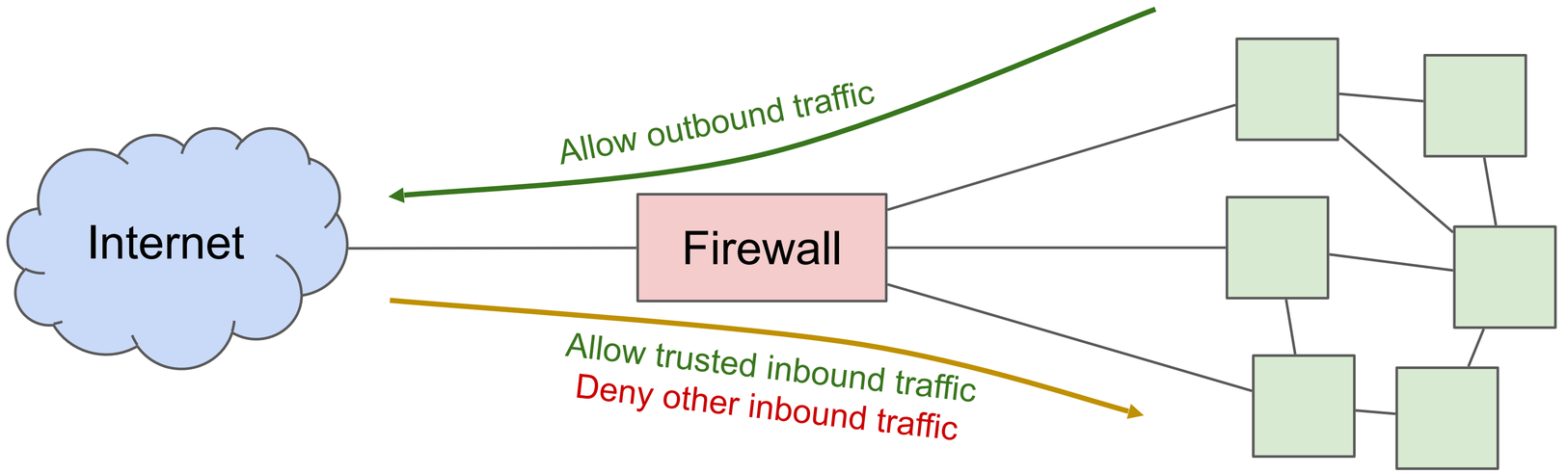
Firewalls and Security Policies

- Idea: Add a single point of access in and out of the network, with a monitor
  - “Ensure complete mediation”
  - Any traffic that could affect vulnerable systems must pass through the firewall
- Network access is controlled by a **policy**
  - Defines what traffic is allowed to exit the network (**outbound policy**)
  - Defines what traffic is allowed to enter the network (**inbound policy**)
  - Policy model based on our threat model: We usually assume users “inside” the network are trusted, and those outside are not



Firewalls and Security Policies

- What's the policy of a standard home network?
  - Outbound policy: Allow outbound traffic
    - Users inside the network can connect to any service
  - Inbound policy: Only some traffic is able to enter the network
    - Allow inbound traffic in response an outbound connection
    - Allow inbound traffic to certain, trusted services (e.g. SSH)
    - Deny all other inbound traffic



Default Security Policies?

- How should we handle traffic that isn't explicitly allowed or denied?
  - **Default-allow policy**: Allow all traffic, but deny those on a specified **denylist**
    - As problems arise, add them to the denylist
  - **Default-deny policy**: Deny all traffic, but allow those on a specified **allowlist**
    - As needs arise (or users complain), add them to the allowlist?

- Which default policy is best?
  - Default-allow is more flexible, but flaws are vulnerabilities and can be catastrophic
  - Default-deny is more conservative, but flaws are less painful
  - Default-deny is generally accepted to be the best default policy ("consider fail-safe defaults")

Use Fail-Safe Defaults

- Choose default settings that “fail safe,” balancing security with usability when a system goes down
  - e.g., Content Security Policy: By default, reject JavaScript from all websites, use an allowlist to accept some JavaScript from trustworthy website

Stateless Packet Filters

- Firewalls are often **packet filters**, which inspect network packets and chooses what to do with them
  - Option #1: Allow the packet to pass through the firewall, forwarding it onwards
  - Option #2: Deny the packet from passing through the firewall, dropping it

- Stateless packet filters
  - Packet filters that have no history
  - All decisions must be made using only the information in the packet itself
  - Can have trouble implementing complex policies that require knowledge of history

Stateless Packet Filters

- Consider implementing the typical home network policy from earlier:
  - Allow outbound traffic
  - Allow inbound traffic in response to an outbound connection
  - Deny all other inbound traffic

- Issue: How do we know what inbound traffic is in response to an outbound connection?
  - TCP: Can be implemented with a hack
    - Allow inbound traffic with the ACK flag set
    - Deny inbound traffic without an ACK flag set
    - If the internal computer sees an ACK packet without having formed a connection, it will ignore it or send a RST
  - UDP: Impossible to implement
    - UDP “connections” are typically implemented at the application layer, so we can’t inspect much

Stateful Packet Filters

- A better idea: Keep state in the implementation of the packet filter
  - The filter keeps track of inbound/outbound connections
    - Notice: All connections have packets going in both directions, so a stateless filter could not do this
  - Rules define what connections are allowed or denied
  - Ultimately, packets are still either forwarded or dropped

- Example rules:
  - **`allow tcp connection 4.5.5.4:* -> 3.1.1.2:80`**
    - Allow connections from **`4.5.5.4`** to **`3.1.1.2`** with destination port 80
  - **`allow tcp connection *:*/int -> *:80/ext`**
    - Allow outbound connections with destination port 80
  - **`allow tcp connection *:*/int -> *:*/ext`**
    - Allow all outbound connections
  - **`allow tcp connection *:*/ext -> 1.2.2.3:80`**
    - Allow inbound connections to **`1.2.2.3`** with destination port 80

Stateful Packet Filters

- Stateful packet filters can also track the state of well-known applications
  - Example: Decoding and tracking HTTP requests/responses
  - Example: Tracking the files sent in an FTP (File Transfer Protocol) connection

Other Types of Firewalls

- **Proxy firewall**: Instead of forwarding packets, form two TCP connections: One with the source, and one with the destination
  - The firewall is really just a MITM
  - Avoids problems with packets, since the firewall has direct access to the TCP byte streams

- **Application proxy firewall**: Certain protocols allow for proxying at the application layer
  - Example: HTTP proxies will make an HTTP request on behalf of the user then return the HTTP response to the client

Alternatives to Allowing Firewall Traffic

- **Virtual private network** (**VPN**): A set of protocols that allows direct access to an internal network via an external connection
  - Creates an encrypted tunnel to allow internal network traffic to be sent securely over the Internet
  - Intuition: The encrypted tunnel is an emulated Ethernet cable that allows you to connect “inside” the network
  - The firewall allows VPN traffic, which allows arbitrary traffic to be tunneled inside

Firewall Pros and Cons

- Pros
  - Centralized management of security policies (single point of control)
  - Transparent operation to end users
  - Mitigates security vulnerabilities on end hosts (e.g. block anything that looks like shellcode)
- Cons
  - Reduced network connectivity
    - Some applications don’t work well inside a firewall
  - Vulnerability to “insiders”
    - Employees could be bribed or threatened
    - Devices are often brought from into the network outside (e.g. cell phones, laptops)
    - Once one device is compromised, attackers can quickly spread through the network
    - Could be mitigated by layering firewalls for more sensitive devices

Summary: Denial of Service

- **Availability**: Making sure users are able to use a service
  - DoS attacks availability of services
- **Application-level DoS**: Attacks the high-level applications
  - Algorithmic complexity attacks: Attack using inputs that cause the worst-case runtime of an algorithm
  - Defense: Identification, isolation, and quotas
  - Defense: Proof of work
- **Network-level DoS**: Attacks the network of a service
  - Typically floods either the network bandwidth or the packet processing capacity
  - Distributed DoS: Use multiple computers to flood a network at the same time
  - Amplified DoS: Use an amplifier to turn a small input into a large output, spoofing packets so the reply goes to the victim
  - Defense: Packet filtering
- All DoS attacks can be defended against by overprovisioning

Summary: SYN Cookies

- **SYN flooding**: A type of DoS that causes a server to allocate state for unfinished TCP connections, upon receiving a SYN packet
  - **SYN cookies**: Instead of allocating state when receiving a SYN, send the state back to the client in the sequence number
  - The client returns the state back to the server, which it only then allocates state for

Summary: Firewalls

- **Firewalls**: Defend many devices by defending the network
  - **Security policies** dictate how traffic on the network is handled
- **Packet filters**: Choose to either forward or drop packets
  - **Stateless packet filters**: Choose depending on the packet only
  - **Stateful packet filters**: Choose depending on the packet and the history of the connection
  - Attackers can subvert packet filters by splitting key payloads or exploiting the TTL
- **Proxy firewalls**: Create a connection with both sides instead of forwarding packets

Agenda

- Firewalls
- Malware
- Viruses
- Worms
- Infection cleanup and rootkits

Malware

- **Malware** (**mal**icious soft**ware**): Malicious code that is stored on and runs on victim's system
  - Sometimes called malcode (malicious code)

What can malware do?

- Deletes files
- Sends spam email
- Launches a Denial of Service (DoS) attack
- Steals private information
- Records user inputs (keylogging, screen capture, webcam capture)
- Encrypts files and demands money to decrypt them (ransomware)
- Physically damages machines
- …

How does malware get to run?

- Attacks a user- or network-facing vulnerable service
- Backdoor: Added by a malicious developer for remote access
- Social engineering: Trick the user into running/clicking/installing
- Trojan horse: Offer a good service, add in the bad
- Drive-by download: Webpage surreptitiously installs without user knowing
- Attacker with physical access downloads & runs it

Potentially from any mode of interaction (automated or not), provided sufficient vulnerability

When does malware run?

- **Some delay based on a trigger**
  - Time bomb: triggered at/after a certain time, e.g., 1st through the 19th of any month…
  - Logic bomb: triggered when a set of conditions hold, e.g., If I haven't appeared in two consecutive payrolls…
  - Can also include a backdoor to serve as ransom, e.g., "I won't let it delete your files if you pay me by Thursday…"

When does malware run?

- **Some delay based on a trigger**
  - Time bomb: triggered at/after a certain time, e.g., 1st through the 19th of any month…
  - Logic bomb: triggered when a set of conditions hold, e.g., If I haven't appeared in two consecutive payrolls…
  - Can also include a backdoor to serve as ransom, e.g., "I won't let it delete your files if you pay me by Thursday…"
- **Some attach themselves to other pieces of code**
  - Viruses: run when the user initiates something, e.g., Run a program, open an attachment, boot the machine
  - Worms: run while another program is running. No user intervention required

Self-Replicating Code

- Propagation: Spread copies of the code from machine to machine
- **Self-replicating code**: A code snippet that outputs a copy of itself
- Can be used to automatically propagate malware
  - When malware is run, the self-replicating code outputs a copy of itself and sends the code to other computers

Viruses and Worms

- Viruses and worms are both malware that automatically self-propagate
- **Virus**: Code that requires user action to propagate
  - Usually infects a computer by altering some stored code
  - When the user runs the code, the code spreads the virus to other users
- **Worm**: Code that does not require user action to propagate
  - Usually infects a computer by altering some already-running code
  - No user interaction required for the worm to spread to other users

Viruses and Worms

- Viruses and worms are both malware that automatically self-propagate
- **Virus**: Code that **requires user action** to propagate
  - Usually infects a computer by altering some stored code
  - When the user runs the code, the code spreads the virus to other users
- **Worm**: Code that **does not require user action** to propagate
  - Usually infects a computer by altering some already-running code
  - No user interaction required for the worm to spread to other users
- The difference between a virus and a worm is not always clear in some malware
  - Some malware uses both approaches together
  - Example: Trojan malware does not self-propagate, but instead requires user action

Malware: Technical Challenges

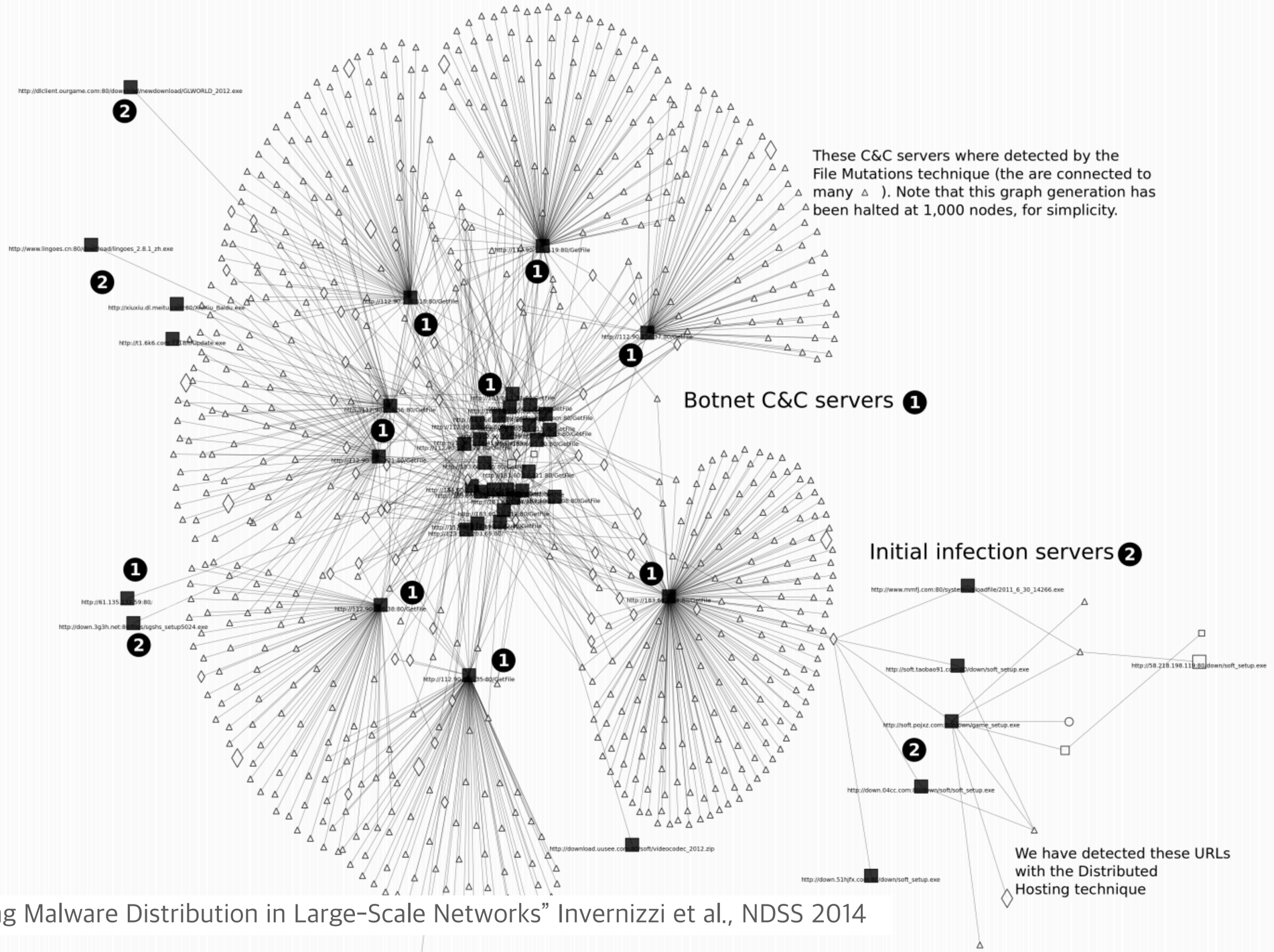
- **Viruses: Detection**
  - Antivirus software wants to detect
  - Virus writers want to avoid detection for as long as possible
  - **Evade** human response
- **Worms: Spreading**
  - The goal is to hit as many machines and as quickly as possible
  - **Outpace** human response

Botnets

- **Botnet**: A set of compromised machines (“bots”) under central control
  - Use a virus or a worm to infect many different computers
  - Every infected computer is now under the attacker’s control
  - A huge amount of resources (e.g. can be used for DoS)

Botnets

- **Botnet**: A set of compromised machines (“bots”) under central control
  - Use a virus or a worm to infect many different computers
  - Every infected computer is now under the attacker’s control
  - A huge amount of resources (e.g. can be used for DoS)
- **C&C Server**: A command-and-control (C&C) server is a computer controlled by an attacker or cybercriminal which is used to send commands to systems compromised by malware



"Nazca: Detecting Malware Distribution in Large-Scale Networks" Invernizzi et al., NDSS 2014

Viruses

- **Virus**: Code that requires user action to propagate
  - Usually infects a computer by altering some stored code
  - When the user runs the code, the code spreads the virus to other users

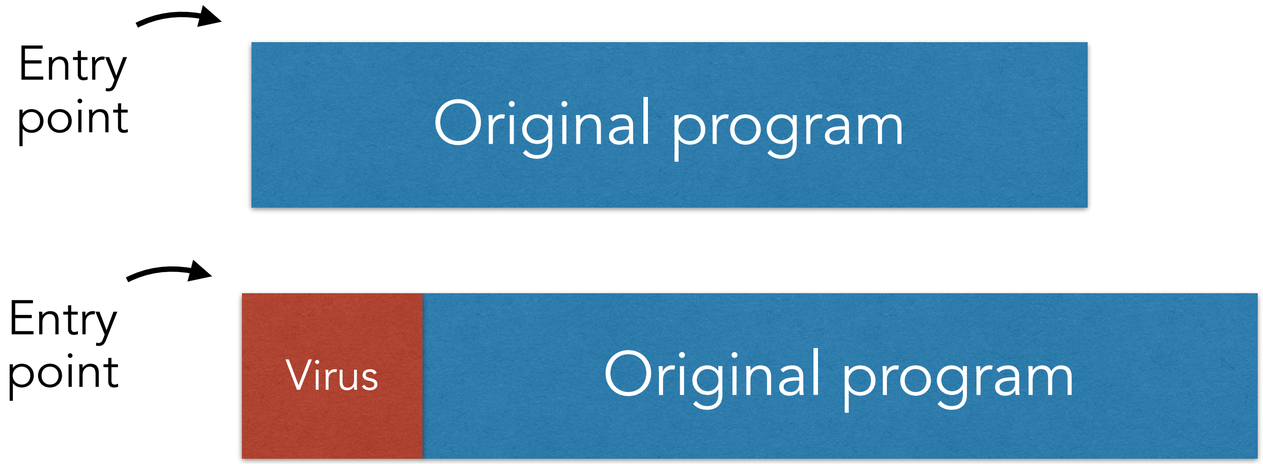
Viruses are classified by what they infect

- Infect / Modify existing code that will eventually be executed by the user
  - **Document Viruses**: Code that runs when the user opens an attachment
    - Implemented within a formatted document
    - Word documents (very rich macros)
    - PDF (Acrobat permits javascript)
    - **(Why you shouldn't open random attachments)**
  - **Boot Sector Viruses**: Code that runs when the system starts up
    - Boot sector: small disk partition at a fixed location, supposed to load the OS -> loads the virus
    - Similar: any AutoRun
    - **(Why you shouldn't plug random USB drives into your computer)**
  - **Mobile App Viruses**: Code that runs when opening an app

Propagation Strategies

- When the malcode runs, it looks for opportunities to infect more systems
  - Example: Send emails to other users with the code attached
  - Example: Copy the code to a USB flash drive (so any other users who run the files on the USB drive will be infected too)
  - Again: Don’t open random attachments! Don’t plug in random USB drives!

How Viruses Affect Other Programs



Take over the entry point

Detection Strategies

- Signature-based detection
  - Viruses replicate by using copies of the same code
  - Capture a virus on one system and look for **bytes corresponding to the virus code** on other systems
  - Example: YARA rules, can match hex code, regex, multiple conditions, etc

Simple Example: Detect Strings that Demand Money

```
1 rule Example_One
2 {
3     strings:
4         $string1 = "pay"
5         $string2 = "immediately"
6
7     condition:
8         ($string1 and $string2)
9 }
```

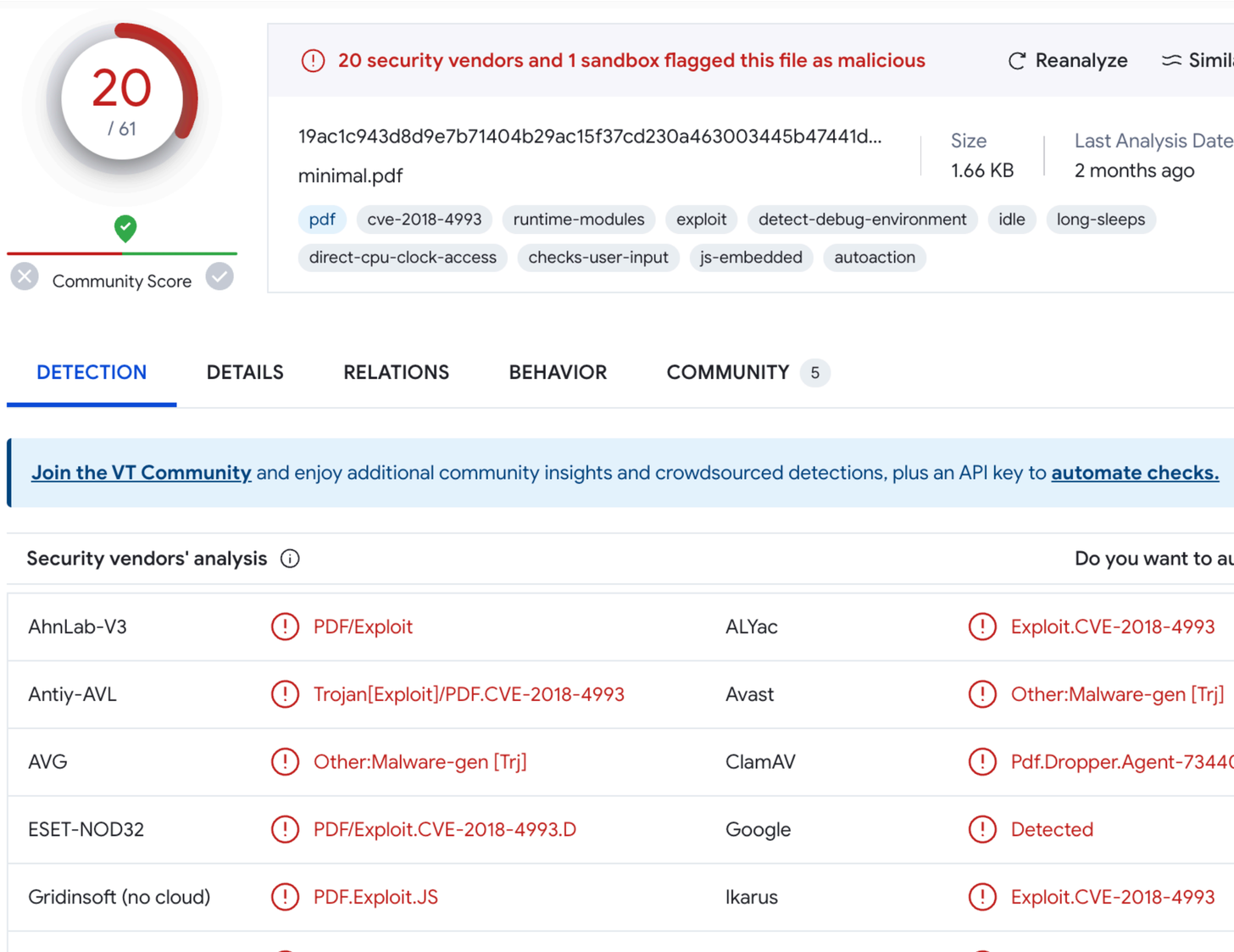
Simple Example: Prevent specific website links or names

```
rule Example_Two
{
    strings:
        $MaliciousWeb1 = "www.scamwebsite.com"
        $MaliciousWeb2 = "www.notrealwebsite.com"
        $Maliciousweb3 = "www.freemoney.com"
        $AttackerName1 = "hackx1203"
        $AttackerName2 = "Hackor"
        $AttackerName3 = "Hax"

     condition:
        any of them
}
```

Antivirus Software

- Antivirus software usually includes a checklist of common viruses
- Example on the right from VirusTotal:
  - 20 out of 61 AV engines detected this file as malicious



Arms Race

Mechanisms for evasive **propagation**

Mechanisms for **detection** and prevention

Want to be able to claim wide coverage for a long time

Want to be able to claim the ability to detect *many* viruses

- Attackers look for **evasion strategies**
- This arms race has influenced the **evolution** of modern malware

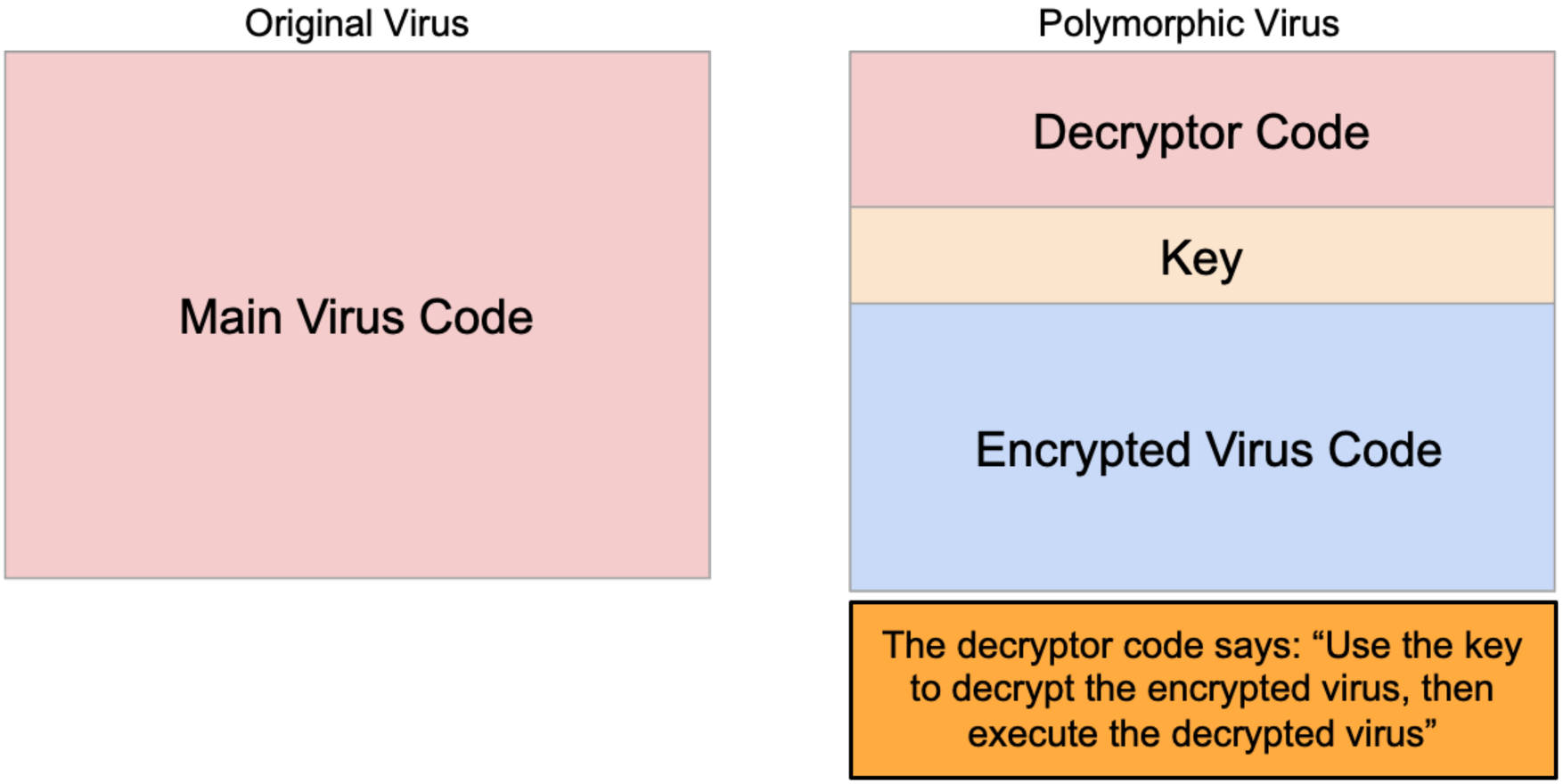
Polymorphic Code

- **Polymorphic code**: Each time the virus propagates, it inserts an encrypted copy of the code
  - The code also includes the key and decryptor
  - When the code runs, it uses the key and decryptor to obtain the original malcode
- Encryption schemes can produce different output on repeated encryptions
  - Example: Using a different key for each encryption

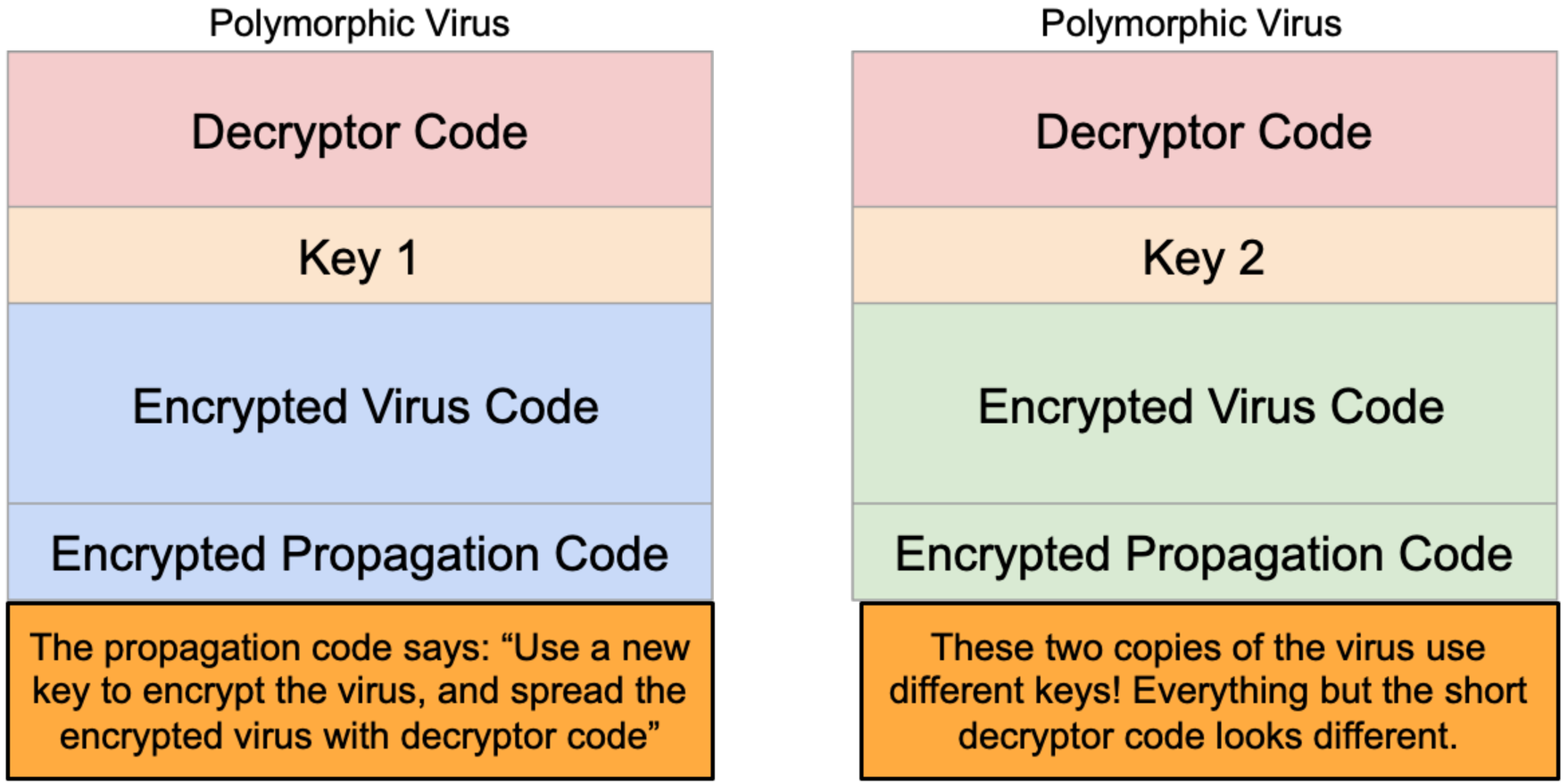
Polymorphic Code

- **Polymorphic code**: Each time the virus propagates, it inserts an encrypted copy of the code
  - The code also includes the key and decryptor
  - When the code runs, it uses the key and decryptor to obtain the original malcode
- Encryption schemes can produce different output on repeated encryptions
  - Example: Using a different key for each encryption
- Encryption is being used for **obfuscation**, not confidentiality
  - The goal is to evade detection by making the virus look different
  - The goal is not to prevent anyone from reading the virus contents
  - Weaker encryption algorithms can be used, and the key can be stored in plaintext

Polymorphic Code



Polymorphic Code



Polymorphic Code: Defenses

- Strategy #1: Add a signature for detecting the decryptor code
  - Issue: Less code to match against → More false positives
  - Issue: The decryptor code could be scattered across different parts of memory

Polymorphic Code: Defenses

- Strategy #1: Add a signature for detecting the decryptor code
  - Issue: Less code to match against → More false positives
  - Issue: The decryptor code could be scattered across different parts of memory
- Strategy #2: Safely check if the code performs decryption
  - Execute the code in a sandbox
  - Analyze the code structure without executing the code
  - Issue: Legitimate programs might perform similar operations too (e.g. decompressing ZIP files)
  - Issue: How long do you let the code execute? The decryptor might only execute after a long delay.

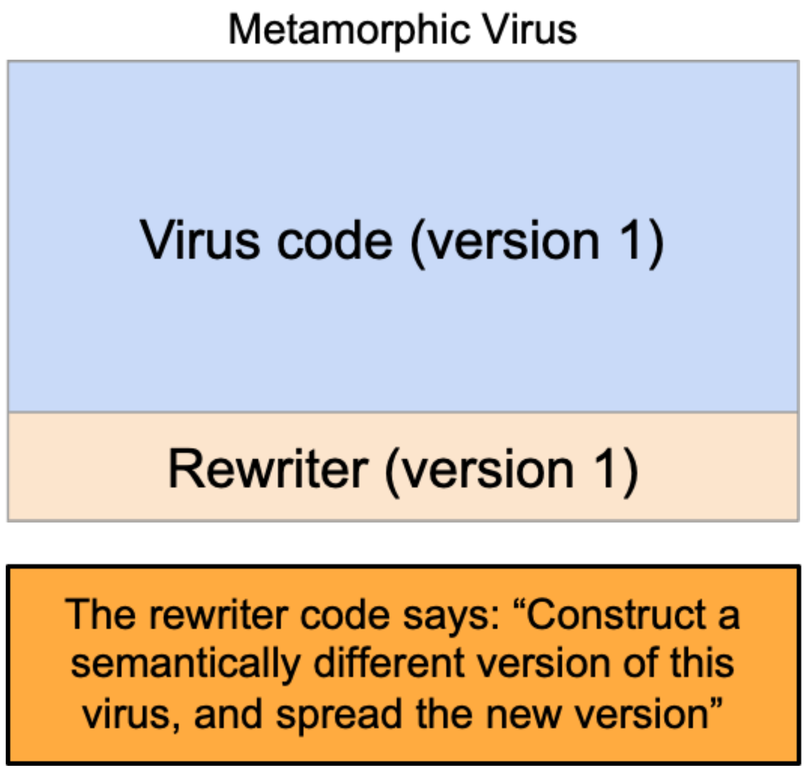
Arms Race: How to Evade?

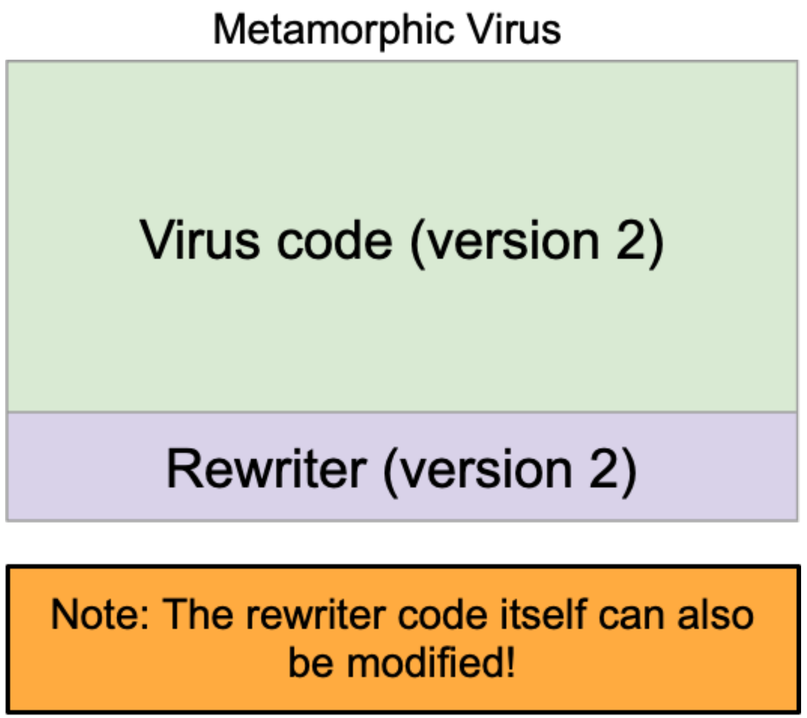
- Idea #1: Change the decrypter
  - True polymorphic viruses: use endless number of decrypters
- Idea #2: Change the decrypted code itself

Metamorphic Code

- **Metamorphic code**: Each time the virus propagates, it generates a **semantically different version** of the code
  - The code performs the **same high-level action**, but with **minor differences in execution**
  - Difference in low-level semantics
- Include a code rewriter with the virus to change the code randomly each time
  - Renumber registers
  - Change order of conditional (if/else) statements
  - Reorder independent operations
  - Replace a low-level algorithm with another (e.g. mergesort and quicksort)
  - Add some code that does nothing useful (or is never executed)

Metamorphic Code





```
5A              pop  edx
BF04000000      mov  edi,0004h
8BF5            mov  esi,ebp
B80C000000      mov  eax,000Ch
81C288000000    add  edx,0088h
8B1A            mov  ebx,[edx]
899C8618110000  mov  [esi+eax*4+00001118],ebx

58              pop  eax
BB04000000      mov  ebx,0004h
8BD5            mov  edx,ebp
BF0C000000      mov  edi,000Ch
81C088000000    add  eax,0088h
8B30            mov  esi,[eax]
89B4BA18110000  mov  [edx+edi*4+00001118],esi
```

Figure 4: Win95/Regswap using different registers in new generations

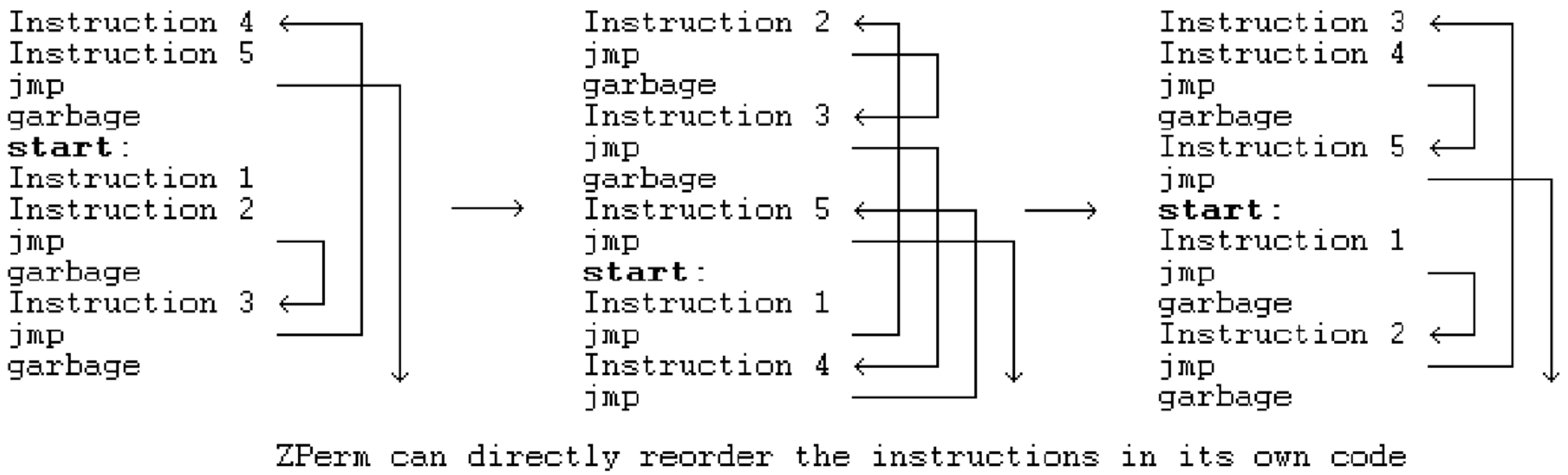


Figure 7. Zperm.A inserts JMP instruction into its code

```
a. An early generation:

C7060F000055   mov       dword ptr [esi],5500000Fh
C746048BEC5151 mov       dword ptr [esi+0004],5151EC8Bh

b. And one of its later generations:

BF0F000055     mov       edi,5500000Fh
893E           mov       [esi],edi
5F             pop       edi
52             push      edx
B640           mov       dh,40
BA8BEC5151     mov       edx,5151EC8Bh
53             push      ebx
8BDA           mov       ebx,edx
895E04         mov       [esi+0004],ebx

c. And yet another generation with recalculated ("encrypted") "con-
stant" data.

BB0F000055     mov       ebx,5500000Fh
891E           mov       [esi],ebx
5B             pop       ebx
51             push      ecx
B9CB00C05F     mov       ecx,5FC000CBh
81C1C0EB91F1   add       ecx,F191EBC0h ; ecx=5151EC8Bh
894E04         mov       [esi+0004],ecx
```

Figure 6: Example of code metamorphosis of Win32/Evol

Metamorphic Code: Defense

- Behavioral detection
  - Need to analyze **behavior** instead of **syntax**
  - Look at the effect of the instructions, not the appearance of the instructions
  - Antivirus company analyzes a new virus to find a **behavioral signature**

- Example: Ransomware encrypts files *and* changes the victim's desktop background
- Really hard to craft behavioral signatures

Metamorphic Code: Defense

- Behavioral detection
  - Need to analyze **behavior** instead of **syntax**
  - Look at the effect of the instructions, not the appearance of the instructions
  - Antivirus company analyzes a new virus to find a **behavioral signature**
- Subverting behavioral detection
  - Delay analysis by waiting a long time before executing malcode
  - Detect that the code is being analyzed (e.g. running in a debugger or a virtual machine) and choose different behavior
  - Antivirus can look for these subversion strategies and skip over them

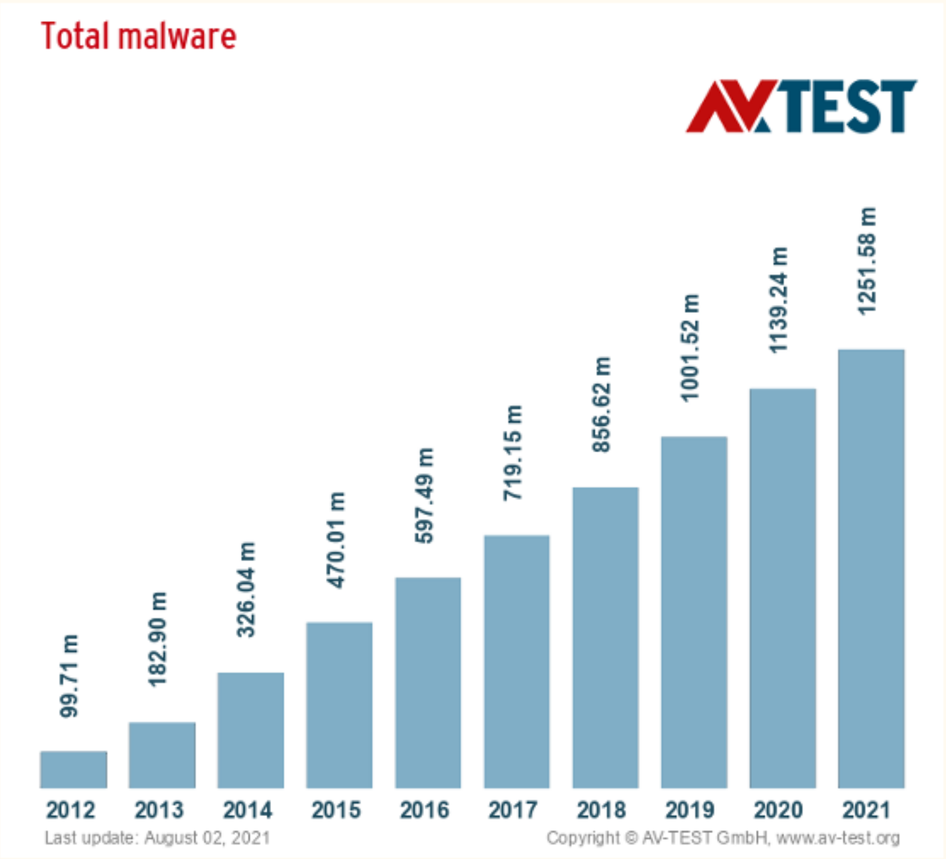
Defense: Flag Unfamiliar Code

- It is impossible to write a perfect algorithm to separate malicious code from safe code
- Antivirus software can look for new, unfamiliar code
  - Keep a central repository of previously-seen code
  - If some code has never been seen before, treat it as more suspicious
  - The central repository can store secure cryptographic hashes of previously-seen code snippets for efficiency (the software hashes code and see if the hash matches a hash in the repository)
- Issue: false positives

Virus Counts May Be Exaggerated

AV-TEST
The Independent IT-Security Institute
Magdeburg Germany

Every day, the AV-TEST Institute registers over 350,000 new malicious programs (malware) and potentially unwanted applications (PUA). These are examined and classified according to their characteristics and saved. Visualization programs then transform the results into diagrams that can be updated and produce current malware statistics.



Takeaway: Antivirus companies might overcount different versions of one virus

Agenda

- Firewalls
- Malware
- Viruses
- Worms
- Infection cleanup and rootkits

Worms

- **Worm**: Malware code that does not require user action to propagate
  - Usually infects a computer by altering some already-running code
  - Unlike malware, no user interaction is required for the worm to spread to other users

Propagation Strategies

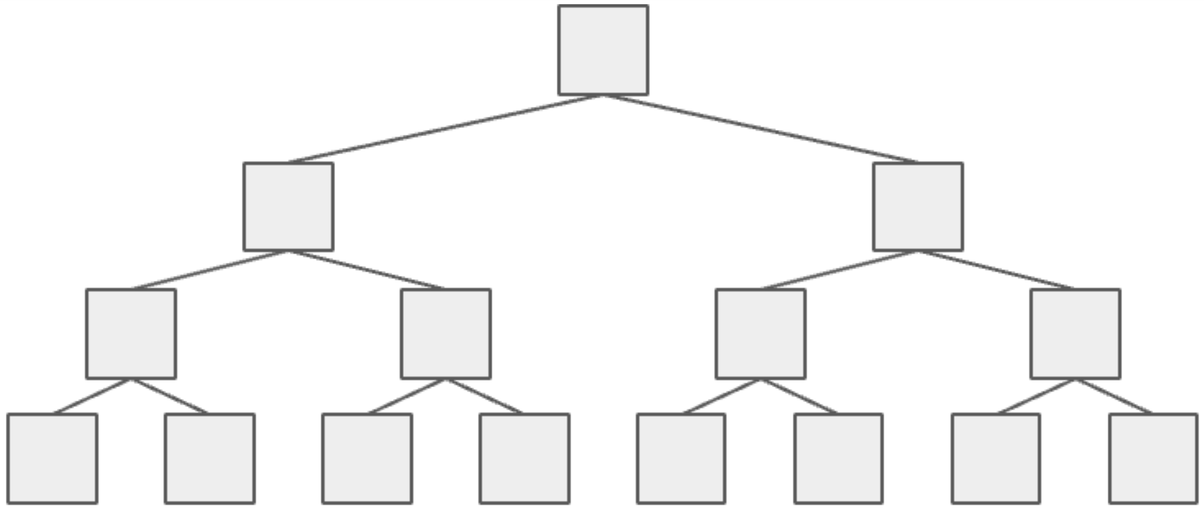
- How does the worm find new users to infect?
  - **Randomly choose machines**: generate a random 32-bit IP address and try connecting to it
  - **Search worms**: Use Google searches to find victims
  - **Scanning**: Look for targets (can be limited by bandwidth)
  - **Target lists**
    - Pre-generated lists (hit lists)
    - Lists of users stored on infected hosts
    - Query a third-party server that lists other servers
  - **Passive**: Wait for another user to contact you, and reply with the infection

Propagation Strategies

- How does the worm find new users to infect?
  - **Randomly choose machines**: generate a random 32-bit IP address and try connecting to it
  - **Search worms**: Use Google searches to find victims
  - **Scanning**: Look for targets (can be limited by bandwidth)
  - **Target lists**
    - Pre-generated lists (hit lists)
    - Lists of users stored on infected hosts
    - Query a third-party server that lists other servers
  - **Passive**: Wait for another user to contact you, and reply with the infection
- How does the worm force code to run?
  - Buffer overflows for code injection
  - A web worm might propagate with an XSS vulnerability

Modeling Worm Propagation

- Worms can potentially spread extremely quickly because they parallelize the process of propagating/replicating
- More computers infected = more computers to spread the worm further
- Viruses have the same property, but usually spread more slowly, since user action is needed to activate the virus



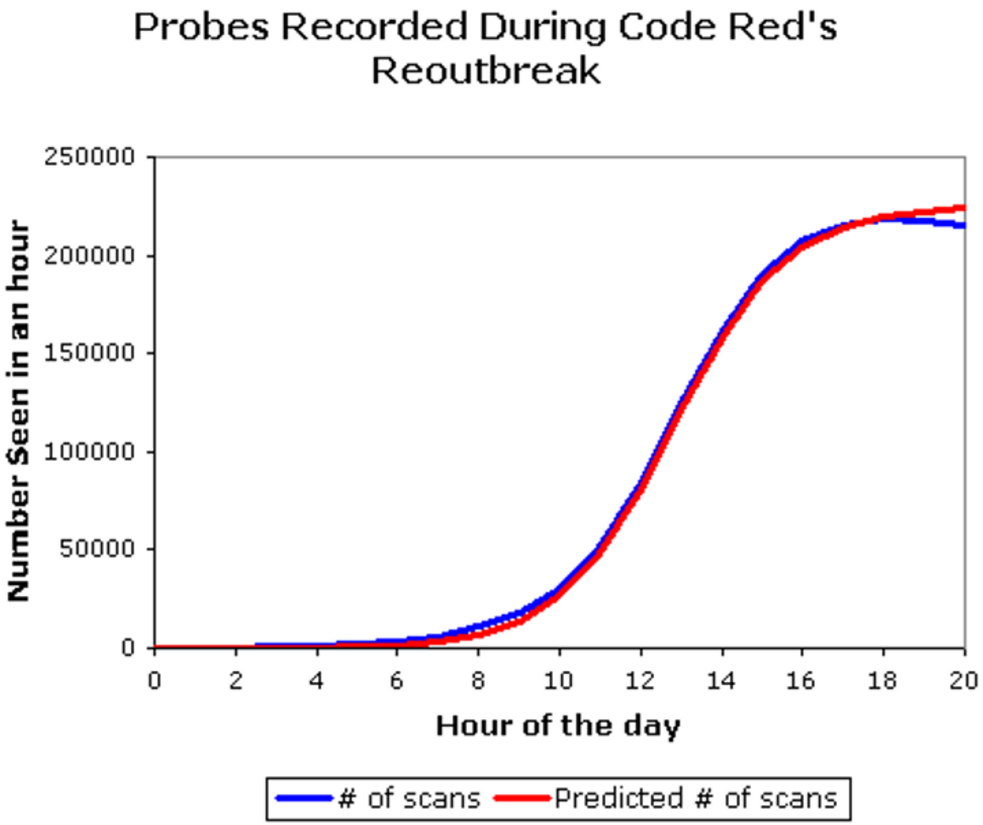
If each infected computer can infect two more computers, we get exponential growth!

Modeling Worm Propagation

- The number of infected hosts grows **logistically**
  - **Initial growth is exponential**:

    More infected hosts = more opportunities to infect
  - **Later growth slows down**: Harder to find new non-infected hosts to infect

- Logistic growth is a good model for worm propagation
  - e.g., Code Red Worm, DoS attack against the White House



Infection Cleanup

- If we find malware on a system, how do we get rid of it?
- May require restoring and repairing many files
  - Antivirus companies sell software that helps with disinfection
- What if the malware executed with administrator privileges?
  - The entire computer is potentially compromised
  - The operating system might be compromised too
  - Best solution: Rebuild the system from data backups and a fresh copy of the operating system
- What if malware infected the tools used to rebuild the operating system?
  - There is no good way of cleaning up malware using only tools in the system!

Rootkit

- Rootkit: Malcode in the operating system that hides its presence
  - Note that the operating system controls disk storage, running processes, etc.
- Rootkits are can be very hard to detect and eliminate
  - Often the best recovery solution is to delete everything and start over

Virus Case Studies

History of Worms: Morris Worm

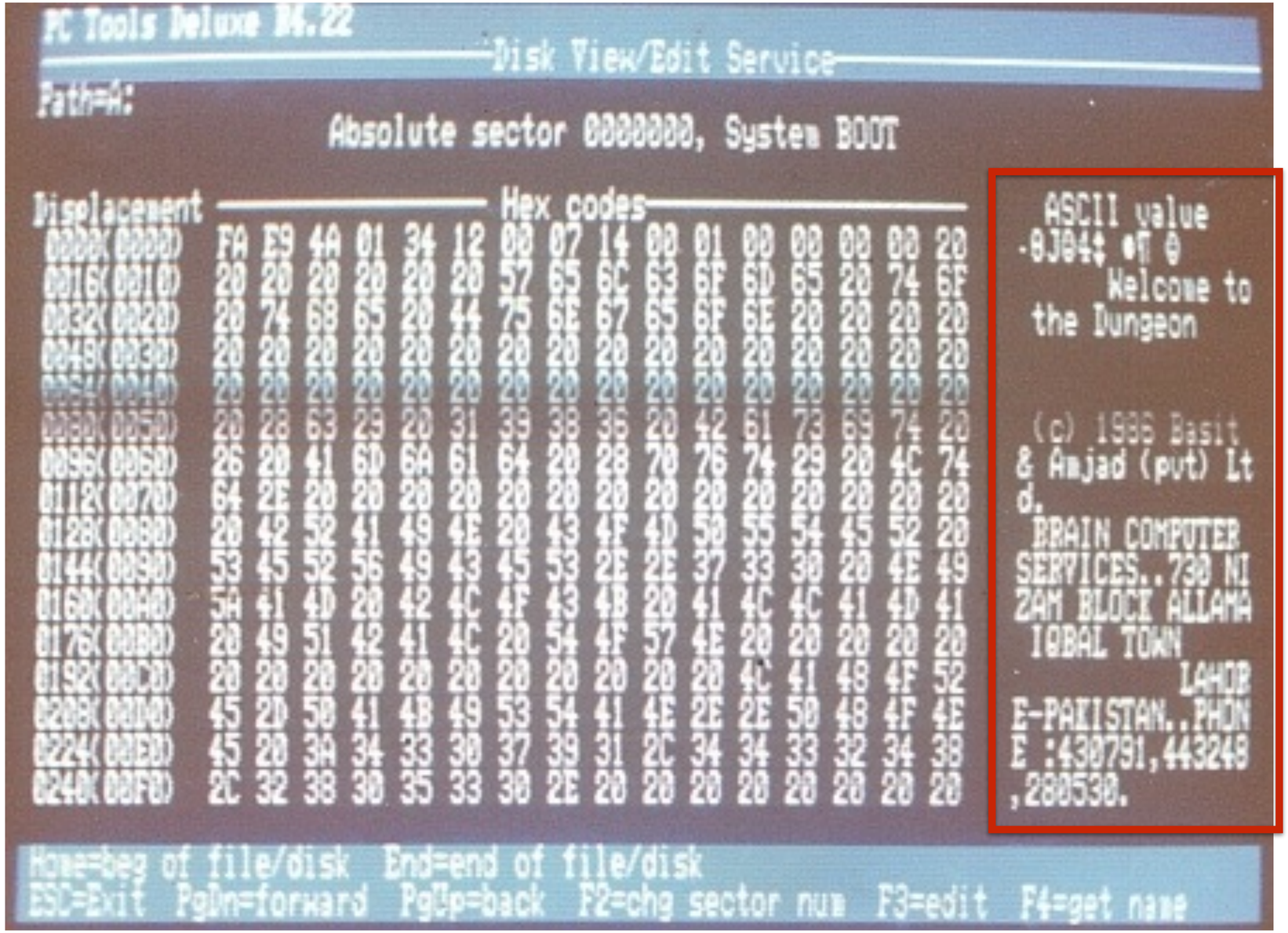
- Morris Worm: November 2, 1988
  - Generally considered the first Internet worm
  - Influenced generations of future worms (and malware)
- Strategies to infect systems
  - Exploit multiple buffer overflows
  - Guess common passwords
  - Activate a “debug” configuration option that provided shell access
  - Exploit common user accounts across different machines
- Strategies to find users to infect
  - Scan local subnet
  - Machines listed in the system’s network configuration
  - Look through user files for mention of remote hosts

BRAIN

First IBM PC virus (1987)

- Propagation method
  - Copies itself into the boot sector
  - Tells the OS that all of the boot sector is “faulty” (so that it won’t list contents to the user)
    - Thus also one of the first examples of a **stealth** virus
  - Intercepts disk read requests for 5.25” floppy drives
    - Sees if the 5th and 6th bytes of the boot sector are 0x1234
    - If so, then it’s already infected, otherwise, infect it

- Payload:
  - Nothing really; goal was just to spread (to show off?)
  - However, it served as the template for future viruses



ROOTKITS

Malicious code that hides from discovery

- Ways to hide:
  - By intercepting system calls, patching the kernel, etc.
  - Often effectively done by a man in the middle attack

- Rootkit revealer: analyzes the disk offline and through the online system calls, and compares

- Mark Russinovich ran a rootkit revealer and found a rootkit in 2005…

SONY XCP ROOTKIT

Detected 2005

- Goal: keep users from copying copyrighted material

- How it worked:
  - Loaded thanks to autorun.exe on the CD
  - Intercepted read requests for its music files
  - If anyone but Sony's music player is accessing them, then garble the data
  - Hid itself from the user (to avoid deletion)

- How it messed up
  - Morally: violated trust
  - Technically: Hid ***all files*** that started with "sys"
  - Seriously?: The uninstaller did not check the integrity of the code it downloaded, and would not delete it afterwords.

STUXNET

June 2010

- Virus in that it initially spread by infected USB stick
  - Once inside a network, it acted as a worm, spreading quickly

- Exploited **four** zero-day exploits
  - Zero-day: Known to only the attacker until the attack
  - Typically, one zero-day is enough to profit
  - Four was unprecedented
    - Immense cost and sophistication on behalf of the attacker

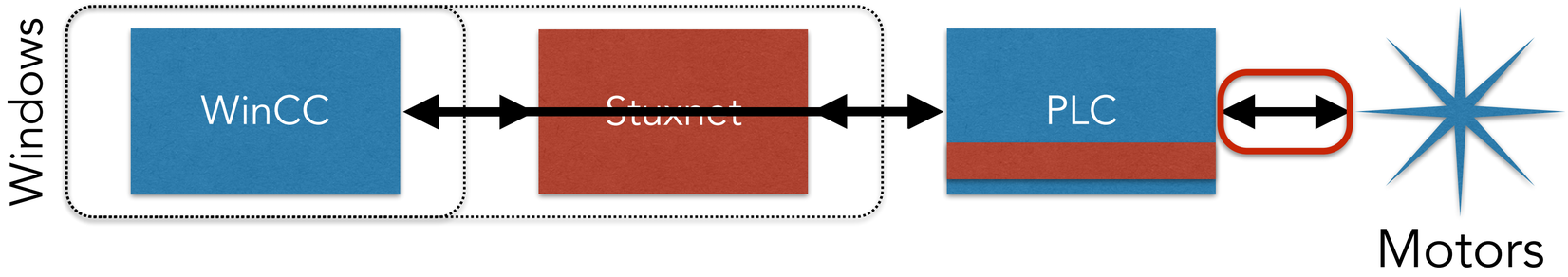
- Rootkit: installed *signed* device drivers
  - Thereby avoiding user alert when installing
  - Signed with certificates stolen from two Taiwanese CAs

STUXNET: PAYLOAD

- Do nothing
- Unless attached to particular models of frequency converter drives that operate at 807-1210Hz
  - You know, like those in Iran and Finland
  - .. those ones that are used to operate centrifuges
  - .. for producing enriched uranium for nuclear weapons
- In which case, slowly increase the freq to 1410Hz
  - You know, enough to break the centrifuge
  - .. all the while sending “looks good to me” readings to the user
  - .. then drop back to normal range

STUXNET: PAYLOAD

- Targets industrial control systems by overwriting programmable logic boards
- Man-in-the-middle between Windows and Siemens control systems; looked like it was working properly to the operator



- In reality, it sped up and slowed down the motors
- Result: Destroy (or at least decrease the productivity of) nuclear centrifuges

STUXNET FALLOUT

- Iran denied they had been hit by Stuxnet
- Then claimed they were, but had contained it
- Understood now that it took out 1k of Iran's 5k centrifuges
- Security experts believe the U.S. did it (possibly along with Israel) due to its sophistication and cost
- **Legitimized cyber warfare**