Malware

Malware

- **Malware** (**mal**icious soft**ware**): Attacker code running on victim computers
  - Sometimes called **malcode** (**mal**icious **code**)
- Catch-all term for many different types of attacker code, including code that:
  - Deletes files
  - Sends spam email
  - Launches a 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
- Today: How does malware propagate?
  - Propagation: Spread copies of the code from machine to machine
  - Strategies for automatic propagation

Self-Replicating Code

- **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
  - The malicious code sends copies of itself to other users
- **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
  - Some malware uses both approaches together
  - Example: Trojan malware does not self-propagate, but instead requires user action

Application of Malware: Botnets

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

Viruses

Viruses

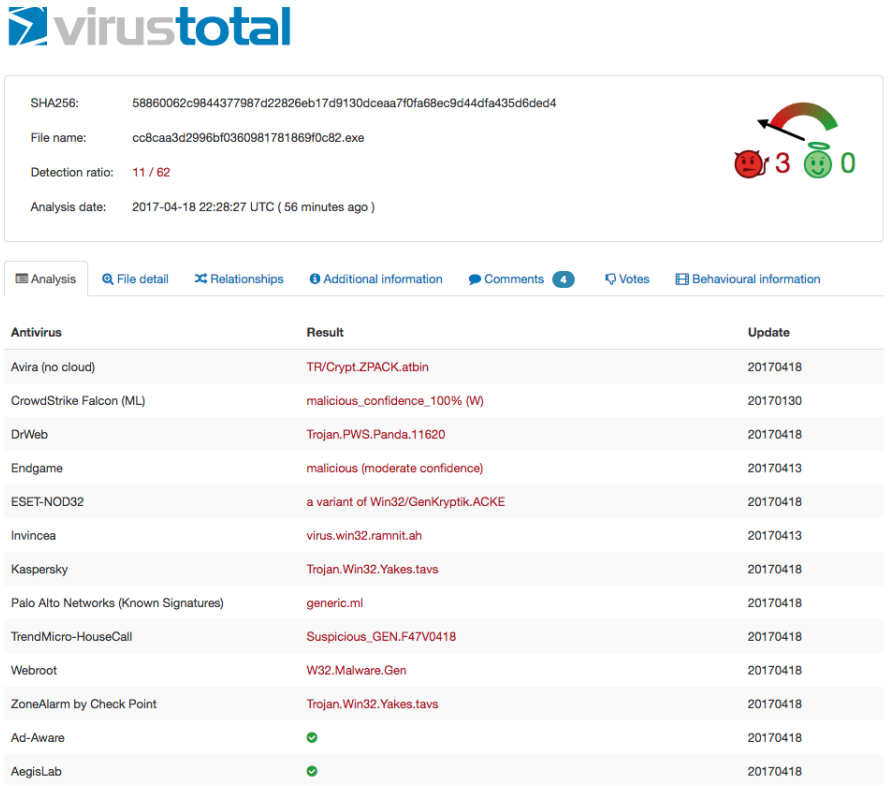
- **Virus**: Code that requires user action to propagate
  - Usually infects a computer by altering some stored code
  - When the user runs the code, the code spreads the virus to other users

Propagation Strategies

- Infect existing code that will eventually be executed by the user
  - Example: Code that runs when opening an app
  - Example: Code that runs when the system starts up
  - Example: Code that runs when the user opens an attachment
- Modify the existing code to include the malcode
- 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)

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
- Antivirus
  - Antivirus software usually includes a checklist of common viruses



virustotal

SHA256: 58860062c9844377987d22826eb17d9130dceaa7f0fa68ec9d44dfa435d6ded4

File name: cc8caa3d2996bf0360981781869f0c82.exe

Detection ratio: 11 / 62

Analysis date: 2017-04-18 22:28:27 UTC (56 minutes ago)

3 0

Analysis | File detail | Relationships | Additional information | Comments 4 | Votes | Behavioural information

Antivirus	Result	Update
Avira (no cloud)	TR/Crypt.ZPACK.atbin	20170418
CrowdStrike Falcon (ML)	malicious_confidence_100% (W)	20170130
DrWeb	Trojan.PWS.Panda.11620	20170418
Endgame	malicious (moderate confidence)	20170413
ESET-NOD32	a variant of Win32/GenKryptik.ACKE	20170418
Invincea	virus.win32.ramnit.ah	20170413
Kaspersky	Trojan.Win32.Yakes.tavs	20170418
Palo Alto Networks (Known Signatures)	generic.ml	20170418
TrendMicro-HouseCall	Suspicious_GEN.F47V0418	20170418
Webroot	W32.Malware.Gen	20170418
ZoneAlarm by Check Point	Trojan.Win32.Yakes.tavs	20170418
Ad-Aware	✓	20170418
AegisLab	✓	20170418

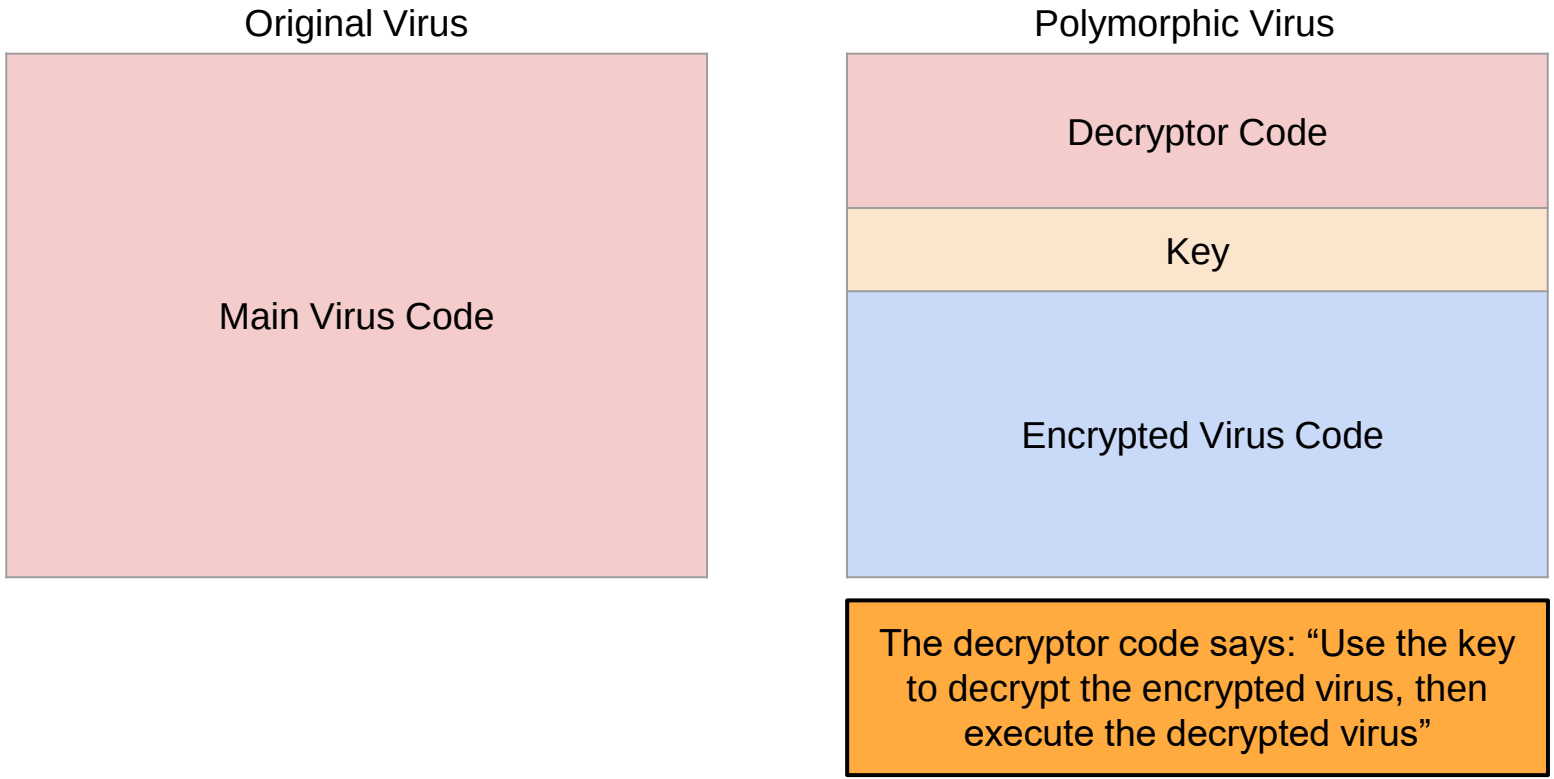
Arms Race

- Viruses have existed for decades
- Active arms race between attackers writing viruses and antivirus companies detecting viruses
- This arms race has influenced the evolution of modern malware
- Attackers look for **evasion** strategies
  - Change the appearance of the virus so that each copy looks different
  - Makes signature-based detection harder
  - Need to automate the process of changing the virus's appearance
- Attackers have a slight advantage
  - Attackers can see what detection strategies the antivirus software is using
  - The antivirus cannot see what attacks the attacker is planning

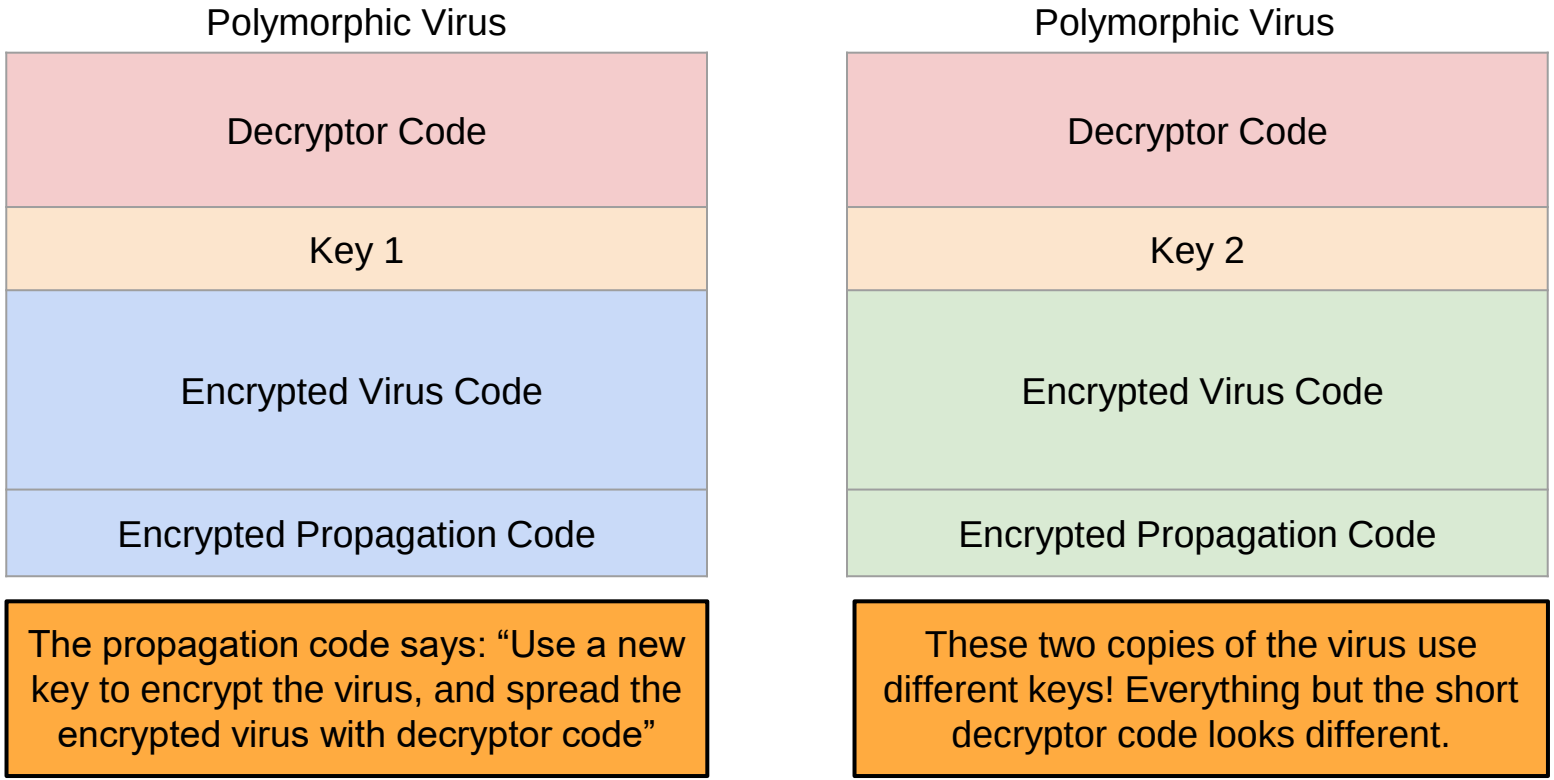
Polymorphic Code

- **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
- Recall: Encryption schemes produce different-looking output on repeated encryptions
  - Example: Using a different IV for each encryption
  - 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



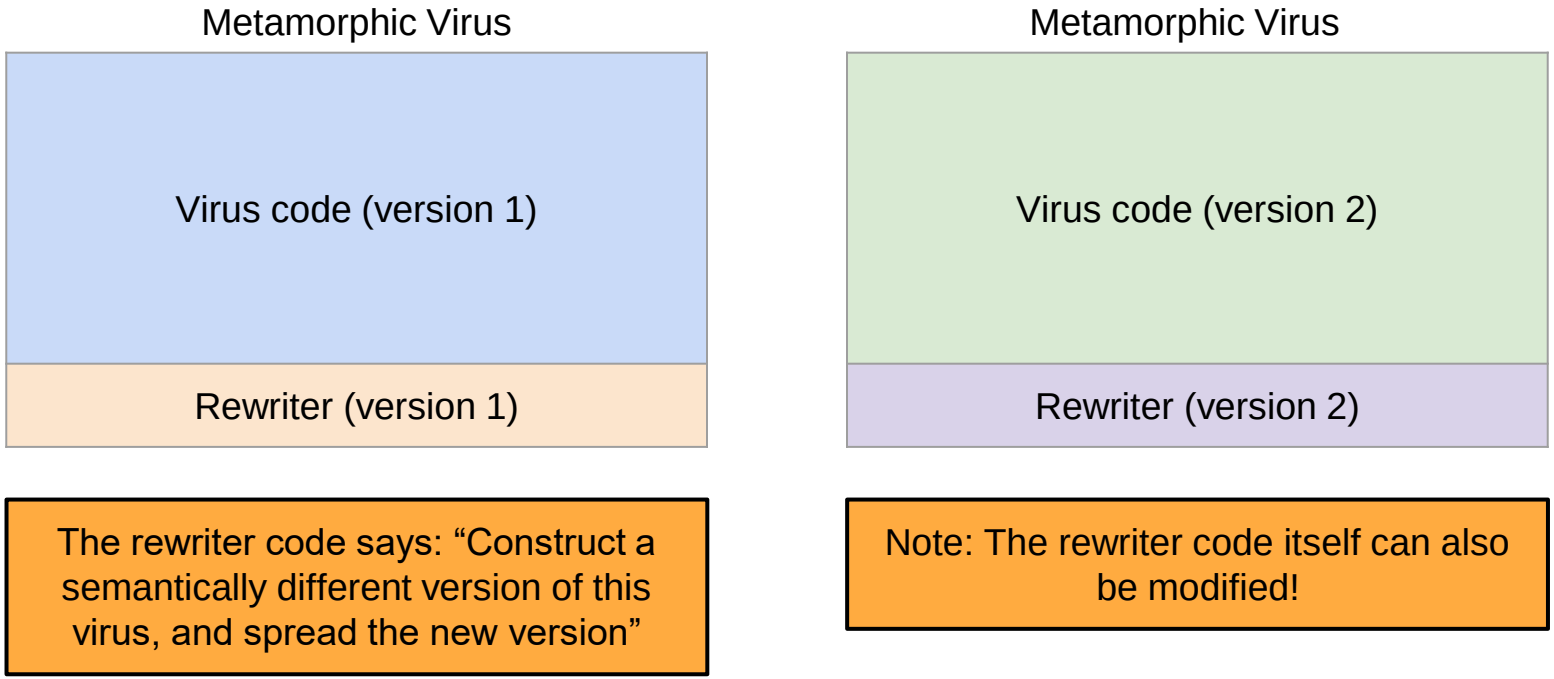
Polymorphic Code: Defenses

- 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
- 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.

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
- 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



Metamorphic Code: Defenses

- 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, and antivirus software analyzes code for the 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
  - A perfect algorithm reduces to the halting problem, which is unsolvable (and out of scope)
- Antivirus software instead looks 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)

Defense: Flag Unfamiliar Code

- Flagging unfamiliar code is a powerful defense
  - You have a detector for malicious behavior (e.g. signature detection)
  - Now you also have a strategy for people avoiding your first detector
- Attacker is in trouble either way:
  - If the attacker doesn’t modify the code for each propagation, it will have a detectable signature
  - If the attacker modifies the code each time, it always appears as new and suspicious
  - When avoiding one strategy, the attacker will be caught by the other strategy!

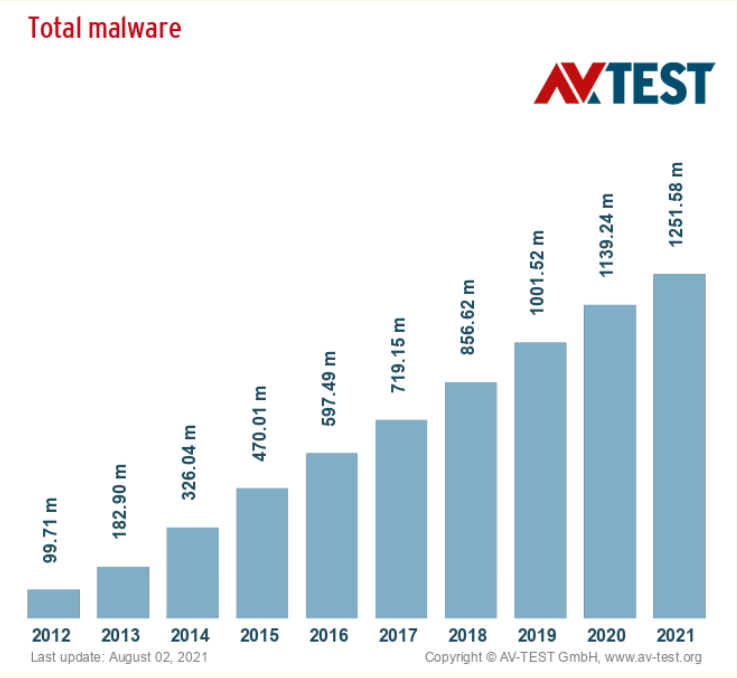
Counting Viruses

- Polymorphism and metamorphism can cause a single virus to be incorrectly counted as thousands of different viruses
- Antivirus companies may want to exaggerate the number of different viruses to convince the public to buy their software
- Antivirus companies may create signatures for every variant of a virus, then advertise the number of signatures in their software (even though fewer, stronger signatures would be better)

Counting Viruses

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. Visualisation programs then transform the results into diagrams that can be updated and produce current malware statistics.



Link

Takeaway: Antivirus companies might overcount different versions of one virus

Worms

Worms

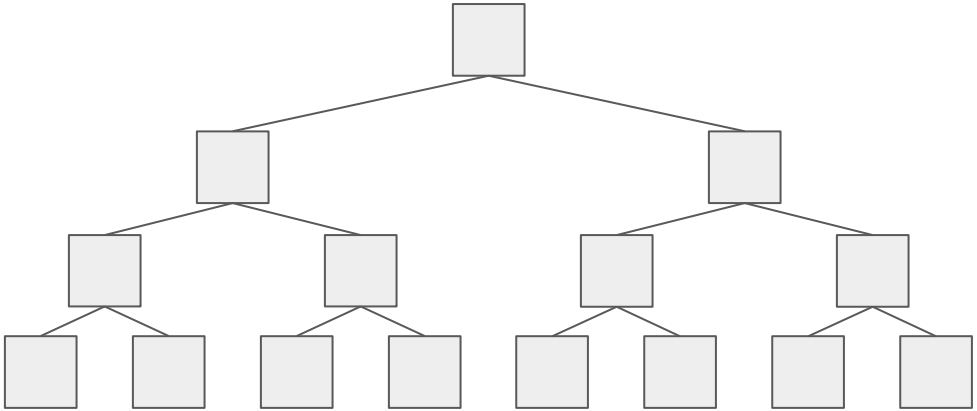
- **Worm**: Code that does not require user action to propagate
  - Usually infects a computer by altering some already-running code
  - Unlike malware, no user interaction is required for the worm to spread to other users

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
- 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

- Worm propagation can be modeled as an infectious epidemic
  - We can use the same models that biologists use to model the spread of infectious diseases
- The spread of the worm depends on:
  - The size of the population
  - The proportion of the population vulnerable to infection
  - The number of infected hosts
  - The contact rate (how often an infected host communicates with other hosts)

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