CISC 3325 - Information Security

Malware and Internet of Things

Malware:
Malicious Software

https://www.welivesecurity.com/2017/10/19/malware-firmware-exploit-sense-security/

Malware

- Programs planted by an agent with malicious intent to cause unanticipated or undesired effects
- Virus
  - A program that can replicate itself and pass on malicious code to other nonmalicious programs by modifying them
- Worm
  - A program that spreads copies of itself through a network
- Trojan horse
  - Code that, in addition to its stated effect, has a second, nonobvious, malicious effect

Viruses, Worms, Trojans, Rootkits

- Malware can be classified into several categories, depending on propagation and concealment
- Propagation
  - Virus: human-assisted propagation (e.g., open email attachment)
  - Worm: automatic propagation without human assistance
- Concealment
  - Rootkit: modifies operating system to hide its existence
  - Trojan: provides desirable functionality but hides malicious operation
- Various types of payloads, ranging from annoyance to crime

Harm from Malicious Code

- Harm to users and systems:
  - Sending email to user contacts
  - Deleting or encrypting files
  - Modifying system information, such as the Windows registry
  - Stealing sensitive information, such as passwords
  - Attaching to critical system files
  - Hide copies of malware in multiple complementary locations
- Harm to the world:
  - Some malware has been known to infect millions of systems, growing at a geometric rate
  - Infected systems often become staging areas for new infections

Transmission and Propagation

- Setup and installer program
- Attached file
- Document viruses
- Autorun
- Using nonmalicious programs:
  - Appended viruses
  - Viruses that surround a program
  - Integrated viruses and replacements

Malware Activation

- One-time execution (implanting)
- Boot sector viruses
- Memory-resident viruses
- Application files
- Code libraries

Malware Types

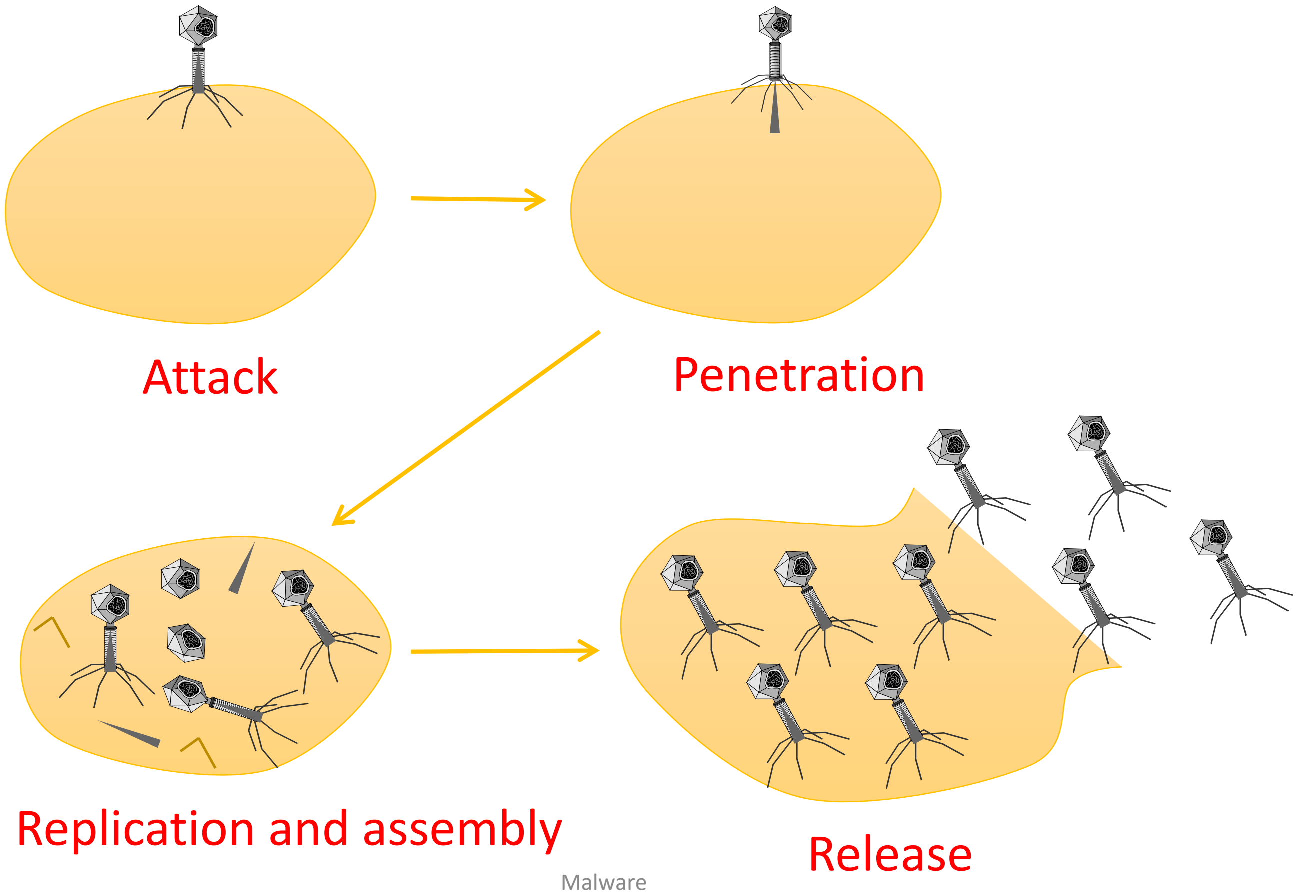
Computer Viruses

- A **computer virus** is computer code that can replicate itself by modifying other files or programs to insert code that is capable of further replication.
- This self-replication property is what distinguishes computer viruses from other kinds of malware, such as logic bombs.
- Another distinguishing property of a virus is that replication requires some type of **user assistance,** such as clicking on an email attachment or sharing a USB drive.

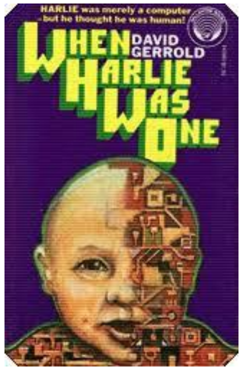
https://www.yelp.com/biz/40-computer-virus-removal-reseda

Biological Analogy

- Computer viruses share some properties with Biological viruses

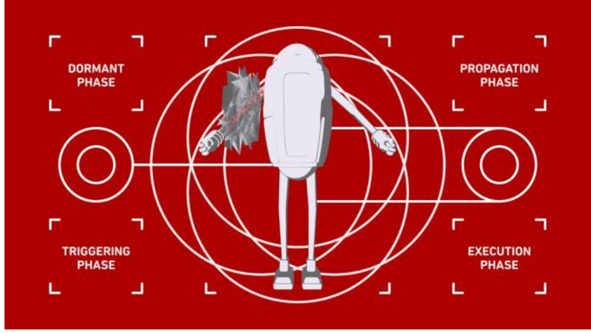


Early History



- 1972 sci-fi novel “When HARLIE Was One” features a program called VIRUS that reproduces itself
- First academic use of term ‘virus’ by PhD student Fred Cohen in 1984
  - credits advisor Len Adleman with coining it
- In 1982, high-school student Rich Skrenta wrote first virus released in the wild: Elk Cloner
  - A boot sector virus
- Brain, by Basit and Amjood Farooq Alvi in 1986, credited with being the first virus to infect PCs

Virus Phases

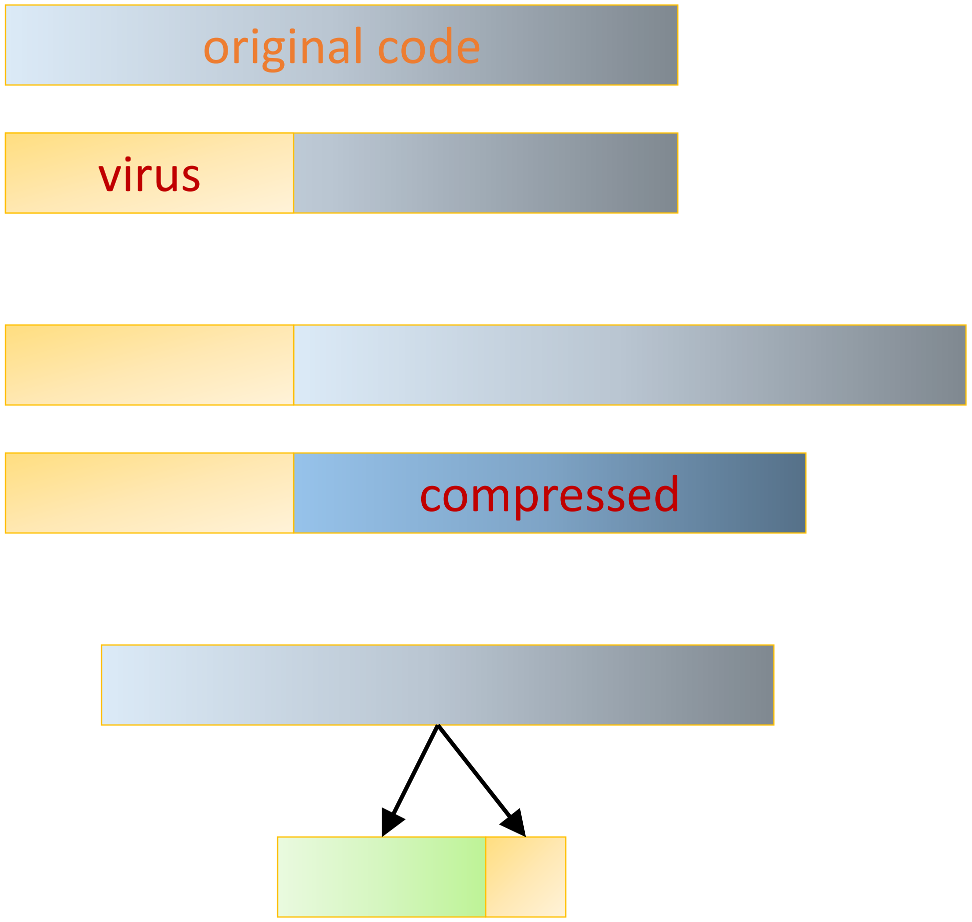


- **Dormant phase.** During this phase, the virus just exists—the virus is laying low and avoiding detection.
- **Propagation phase.** During this phase, the virus is replicating itself, infecting new files on new systems.
- **Triggering phase.** In this phase, some logical condition causes the virus to move from a dormant or propagation phase to perform its intended action.
- **Action phase.** In this phase, the virus performs the malicious action that it was designed to perform, called **payload.**
  - This action could include something seemingly innocent, like displaying a silly picture on a computer's screen, or something quite malicious, such as deleting all essential files on the hard drive.

https://festival.vconline.org/2017/films/nuwa/

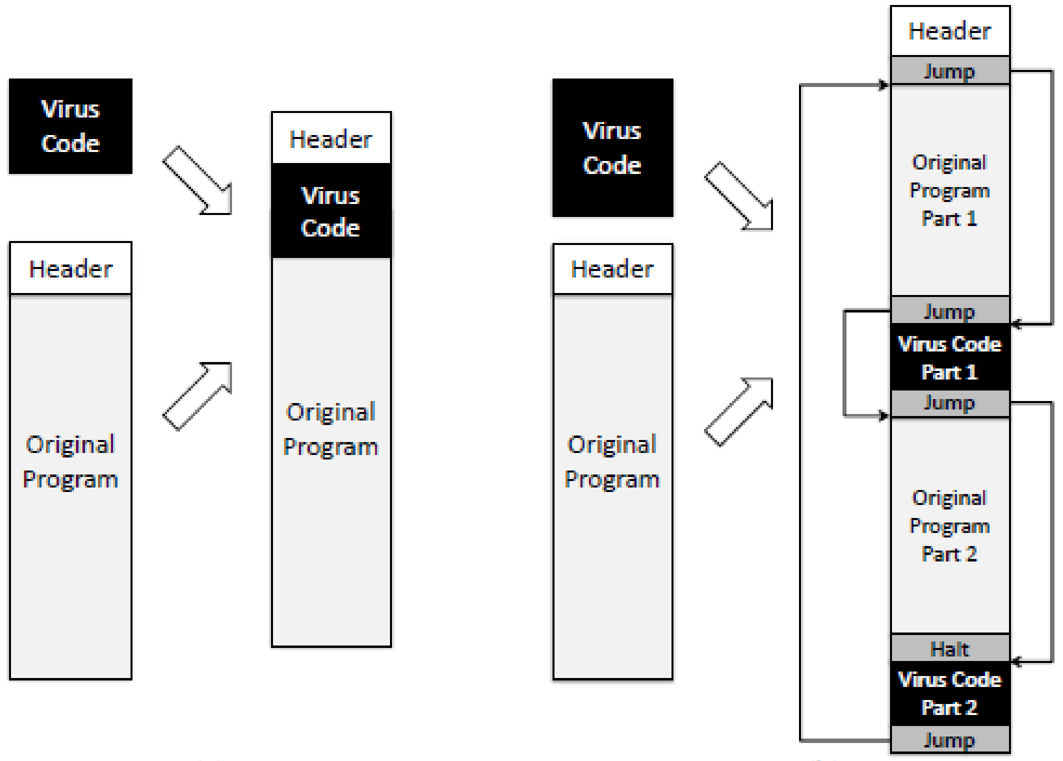
Infection Types

- Overwriting
  - Destroys original code
- Pre-pending
  - Keeps original code, possibly compressed
- Infection of libraries
  - Allows virus to be memory resident
  - E.g., kernel32.dll
- Macro viruses
  - Infects MS Office documents
  - Often installs in main document template

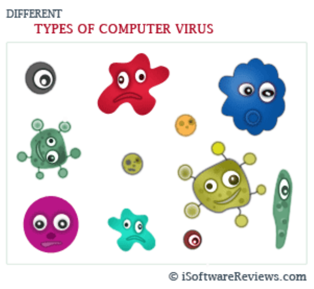


Degrees of Complication

- Viruses have various degrees of complication in how they can insert themselves in computer code.



(a) (b)



Concealment

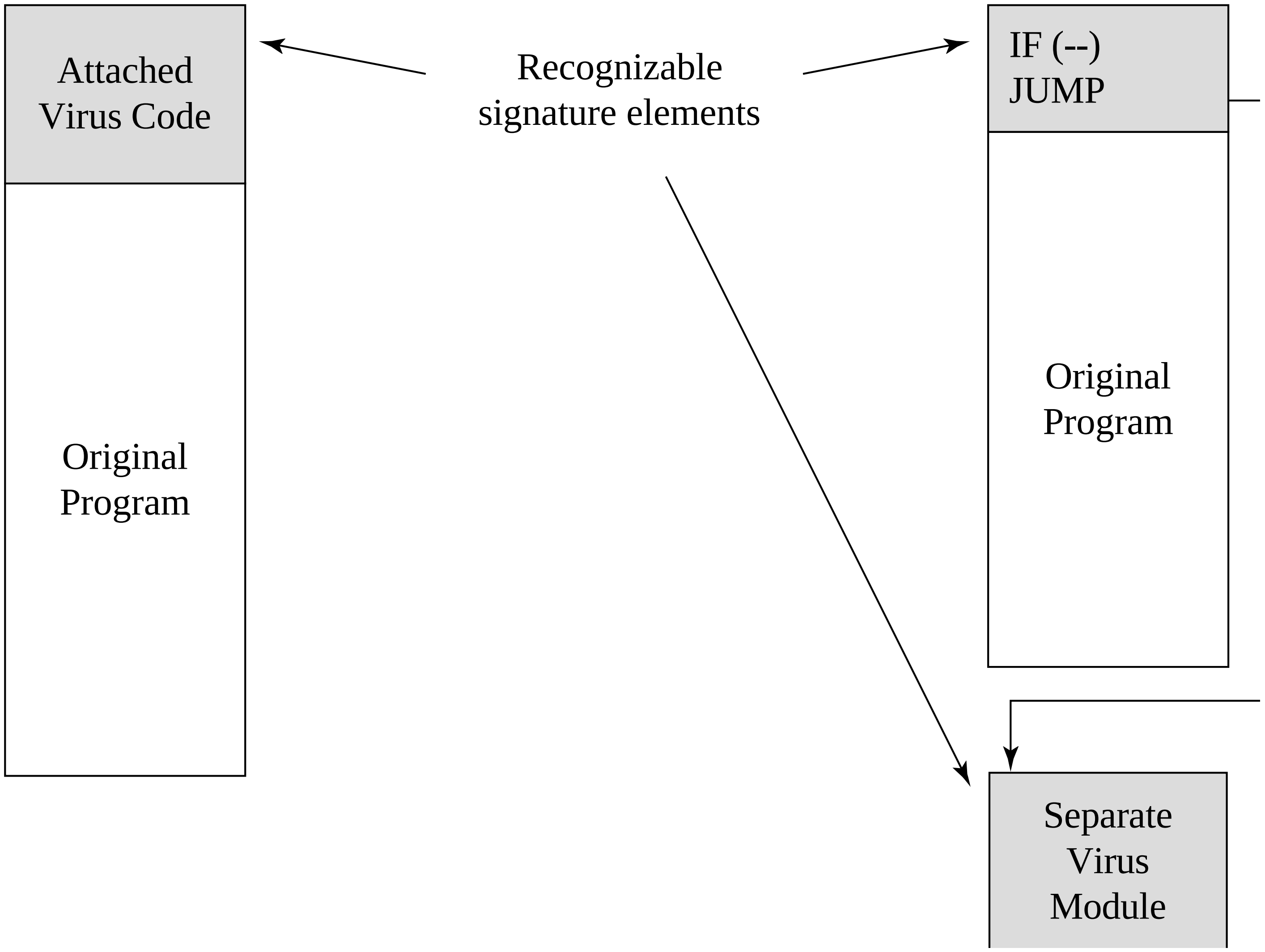
- Encrypted virus
  - Decryption engine + encrypted body
  - Randomly generate encryption key
  - Detection looks for decryption engine
- Polymorphic virus
  - Encrypted virus with random variations of the decryption engine (e.g., padding code)
  - Detection using CPU emulator
- Metamorphic virus
  - Different virus bodies
  - Approaches include code permutation and instruction replacement
  - Challenging to detect

http://www.hoax-slayer.com/really-bad-virus-warning.shtml, https://sites.google.com/site/a14g32/home/different-types-of-computer-viruses

Virus Detection

- Virus scanners look for signs of malicious code infection using signatures in program files and memory
- Traditional virus scanners have trouble keeping up with new malware—detect about 45% of infections
- Detection mechanisms:
  - Known string patterns in files or memory
  - Execution patterns
  - Storage patterns

Virus Signatures



Computer Worms

- A **computer worm** is a malware program that spreads copies of itself without the need to inject itself in other programs, and usually without human interaction.
- Thus, computer worms are technically not computer viruses (since they don't infect other programs), but some people nevertheless confuse the terms, since both spread by self-replication.
- In most cases, a computer worm will carry a malicious payload, such as deleting files or installing a backdoor.

https://www.youtube.com/watch?v=Jz83nFMH-04

Early History

- First worms built in the labs of John Shock and Jon Hepps at Xerox PARC in the early 80s
- CHRISTMA EXEC written in REXX, released in December 1987, and targeting IBM VM/CMS systems was the first worm to use e-mail service

Early History (cont.)

- The first internet worm was the Morris Worm, written by Cornell student Robert Tappan Morris and released on November 2, 1988
  - First person to be indicted under the Computer Fraud and Abuse Act
    - Sentenced to probation

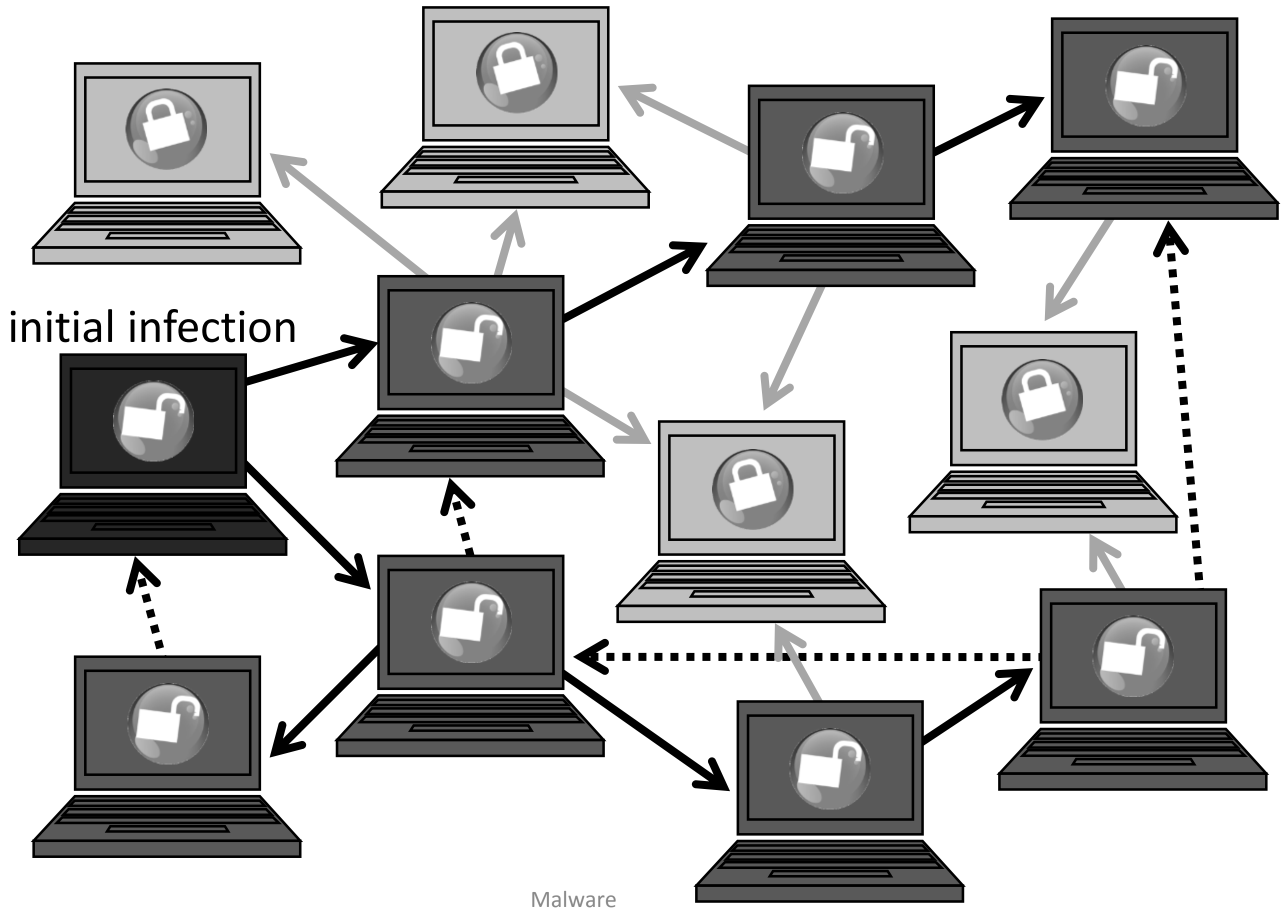
Worm Development

- Identify vulnerability still unpatched
- Write code for
  - Exploit of vulnerability
  - Generation of target list
    - Random hosts on the internet
    - Hosts on LAN
    - Divide-and-conquer
  - Installation and execution of payload
  - Querying/reporting if a host is infected
- Initial deployment on botnet

- Worm template
  - Generate target list
  - For each host on target list
    - Check if infected
    - Check if vulnerable
    - Infect
    - Recur
- Distributed graph search algorithm
  - Forward edges: infection
  - Back edges: already infected or not vulnerable

Worm Propagation

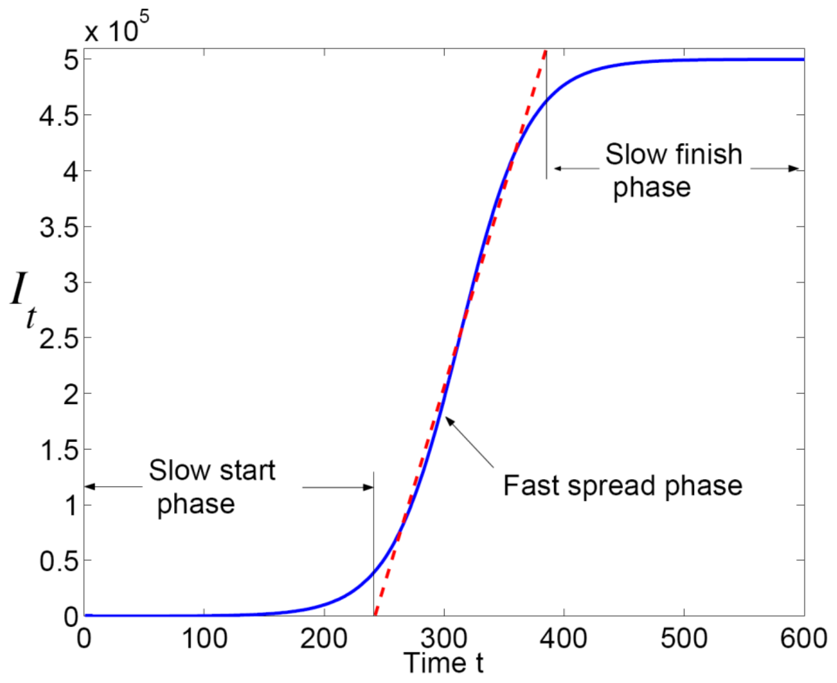
- Worms propagate by finding and infecting vulnerable hosts.
  - They need a way to tell if a host is vulnerable
  - They need a way to tell if a host is already infected.



Propagation: Theory

- Classic epidemic model
  - N: total number of vulnerable hosts
  - I(t): number of infected hosts at time t
  - S(t): number of susceptible hosts at time t
  - I(t) + S(t) = N
  - b: infection rate
- Differential equation for I(t):
- dI/dt = bI(t) S(t)
- More accurate models adjust propagation rate over time

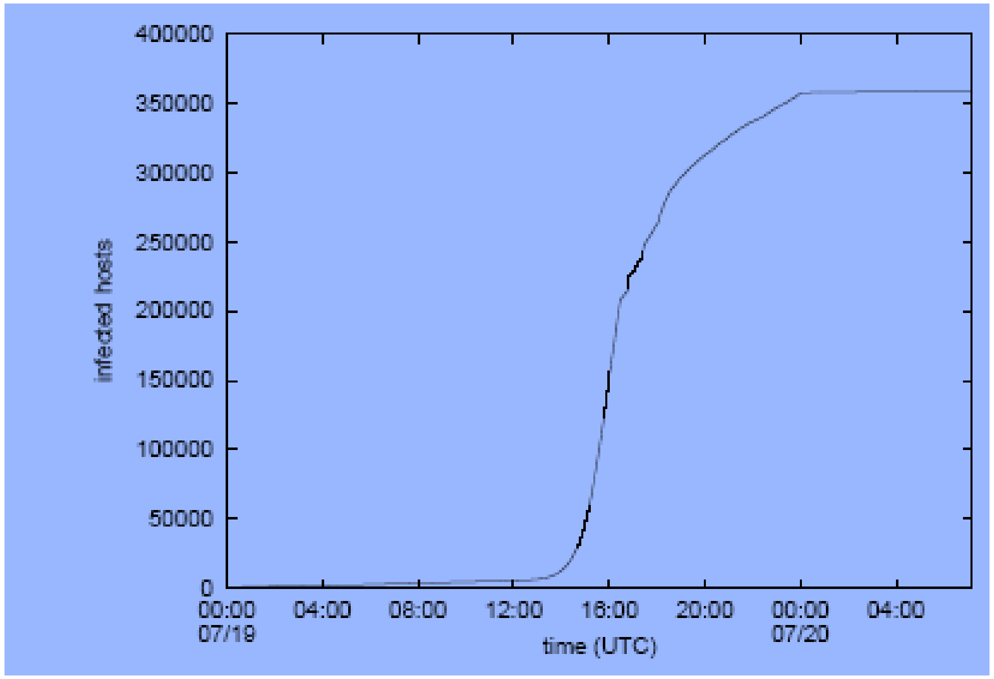
Source:
Cliff C. Zou, Weibo Gong, Don Towsley, and Lixin Gao. The Monitoring and Early Detection of Internet Worms, IEEE/ACM Transactions on Networking, 2005.



Propagation: Practice

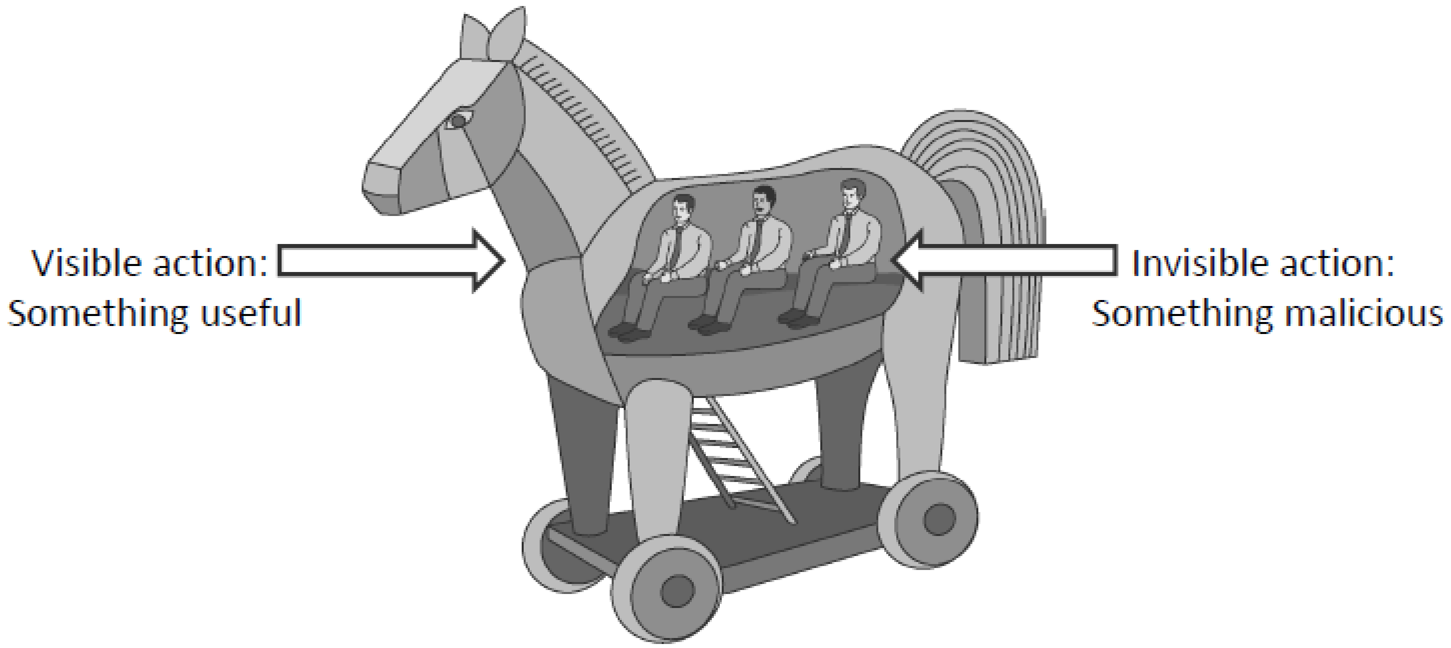
- Cumulative total of unique IP addresses infected by the first outbreak of Code-RedI v2 on July 19-20, 2001

Source:
David Moore, Colleen Shannon, and Jeffery Brown. Code-Red: a case study on the spread and victims of an Internet worm, CAIDA, 2002



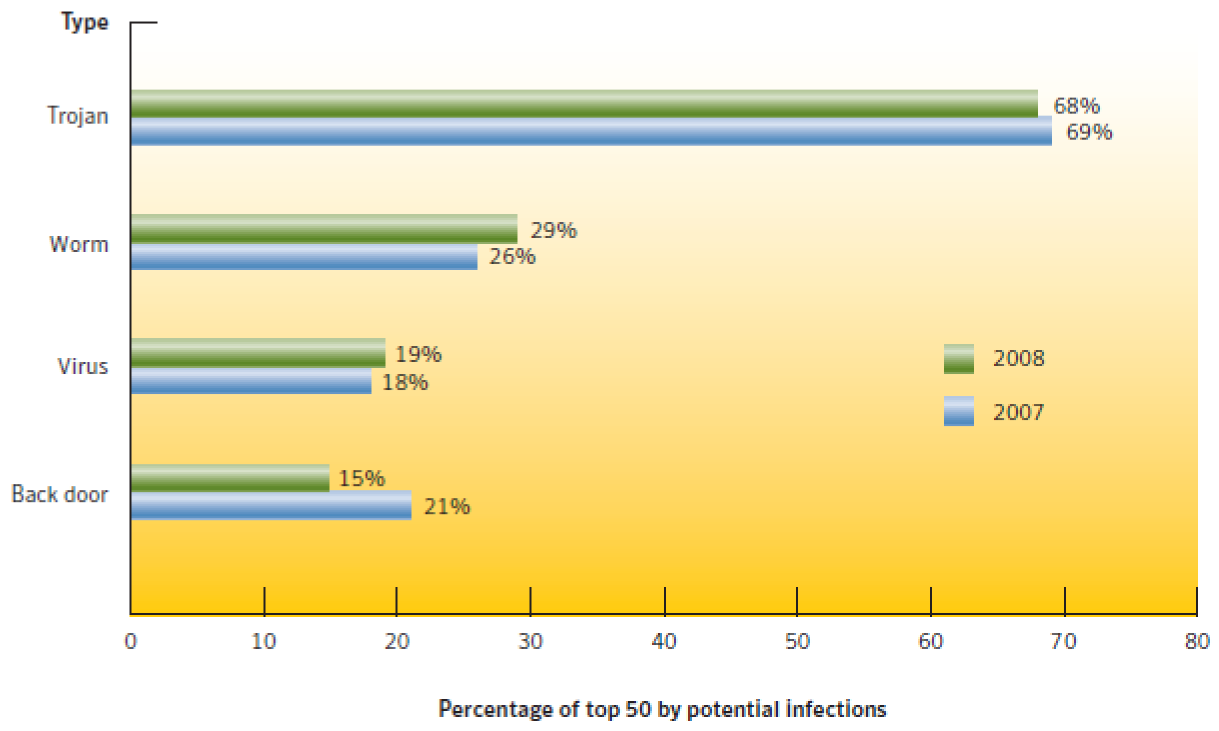
Trojan Horses

- A **Trojan horse (or Trojan)** is a malware program that appears to perform some useful task, but which also does something with negative consequences (e.g., launches a keylogger).
- Trojan horses can be installed as part of the payload of other malware but are often installed by a user or administrator, either deliberately or accidentally.



Current Trends

- Trojans currently have largest infection potential
  - Often exploit browser vulnerabilities
  - Typically used to download other malware in multi-stage attacks



Source:
Symantec Internet Security Threat Report, April 2009

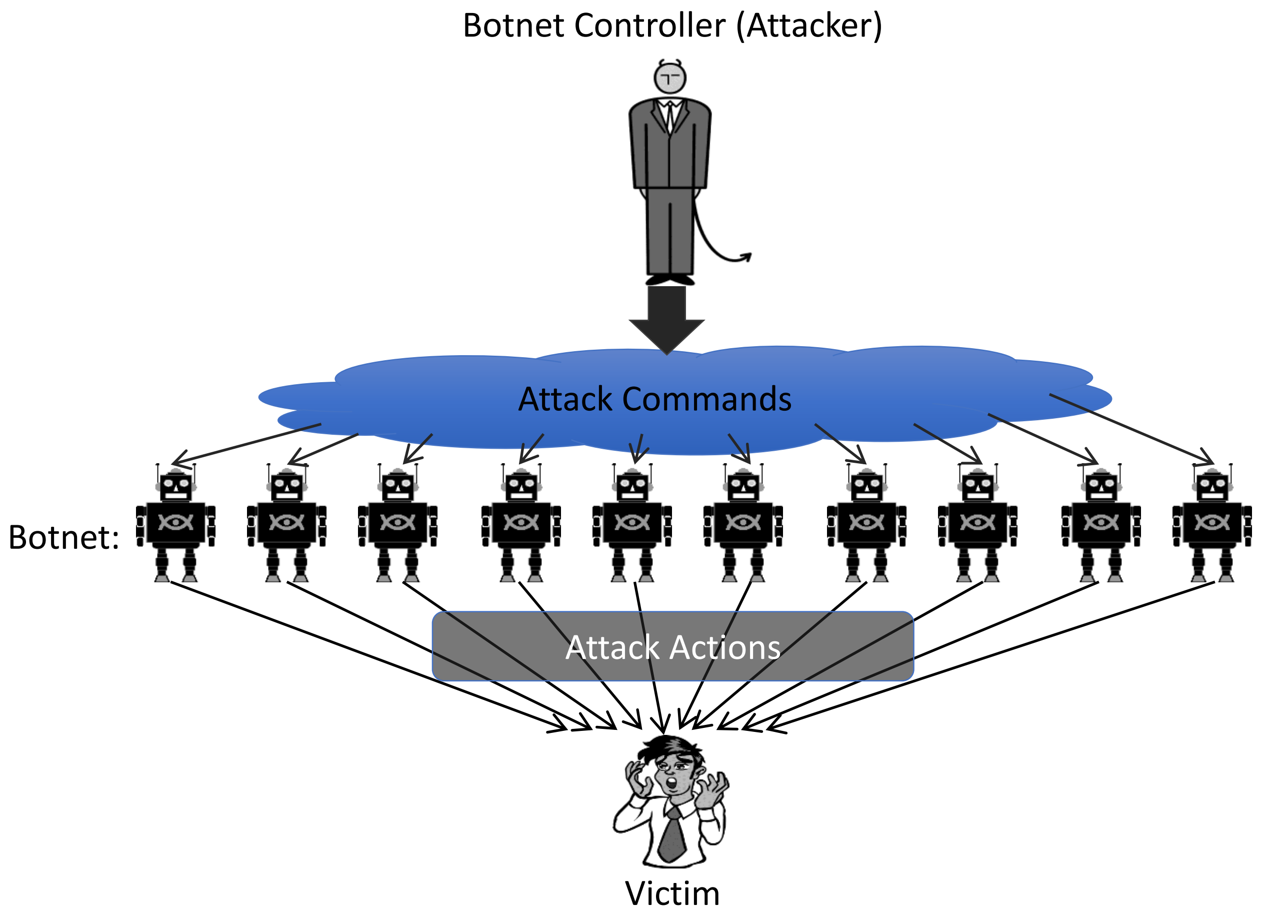
Rootkits

- A rootkit modifies the operating system to hide its existence
  - E.g., modifies file system exploration utilities
  - Hard to detect using software that relies on the OS itself
- RootkitRevealer
  - By Bryce Cogswell and Mark Russinovich (Sysinternals)
  - Two scans of file system
  - High-level scan using the Windows API
  - Raw scan using disk access methods
  - Discrepancy reveals presence of rootkit
  - Could be defeated by rootkit that intercepts and modifies results of raw scan operations

https://www.softshop.eu/en/information/library/computer-threats/rootkit/

Malware Zombies

- Malware can turn a computer in to a **zombie**, which is a machine that is controlled externally to perform malicious attacks, usually as a part of a **botnet**.



Insider Attacks

- An **insider attack** is a security breach that is caused or facilitated by someone who is a part of the very organization that controls or builds the asset that should be protected.
- In the case of malware, an insider attack refers to a security hole that is created in a software system by one of its programmers.

https://www.observeit.com/blog/top-motivating-factors-inside-threats/

Backdoors

- A **backdoor,** which is also sometimes called a **trapdoor,** is a hidden feature or command in a program that allows a user to perform actions he or she would not normally be allowed to do.
- When used in a normal way, this program performs completely as expected and advertised.
- But if the hidden feature is activated, the program does something unexpected, often in violation of security policies, such as performing a privilege escalation.
- Benign example: **Easter Eggs** in DVDs and software

https://www.computerhope.com/jargon/b/backdoor.htm

Logic Bombs

- A **logic bomb** is a program that performs a malicious action as a result of a certain logic condition.
- The classic example of a logic bomb is a programmer coding up the software for the payroll system who puts in code that makes the program crash should it ever process two consecutive payrolls without paying him.
- Another classic example combines a logic bomb with a backdoor, where a programmer puts in a logic bomb that will crash the program on a certain date.

The Omega Engineering Logic Bomb

- An example of a logic bomb that was actually triggered and caused damage:
  - Programmer Tim Lloyd was convicted of using on his former employer, Omega Engineering Corporation.
  - On July 31, 1996, a logic bomb was triggered on the server for Omega Engineering's manufacturing operations
  - Ultimately cost the company millions of dollars in damages, led to lay-off of many of its employees.

The Omega Bomb Code

- The Logic Behind the Omega Engineering Time Bomb included the following strings:
- 7/30/96
  - Event that triggered the bomb
- F:
  - Focused attention to volume F, which had critical files
- F:\LOGIN\LOGIN 12345
  - Login a fictitious user, 12345 (the back door)
- CD \PUBLIC
  - Moves to the public folder of programs
- FIX.EXE /Y F:\*.*
  - Run a program, called FIX, which actually deletes everything
- PURGE F:\/ALL
  - Prevent recovery of the deleted files

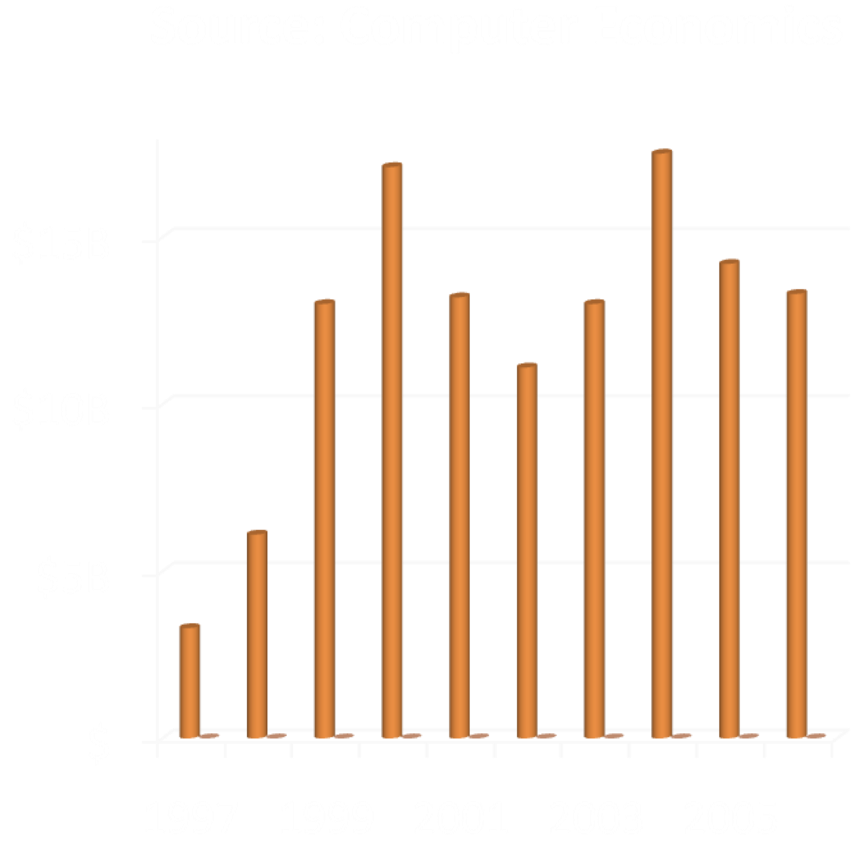
Defenses against Insider Attacks

- Avoid single points of failure.
- Use code walk-throughs.
- Use archiving and reporting tools.
- Limit authority and permissions.
- Physically secure critical systems.
- Monitor employee behavior.
- Control software installations.

https://www.cpni.gov.uk/insider-risk-assessment

Financial Impact

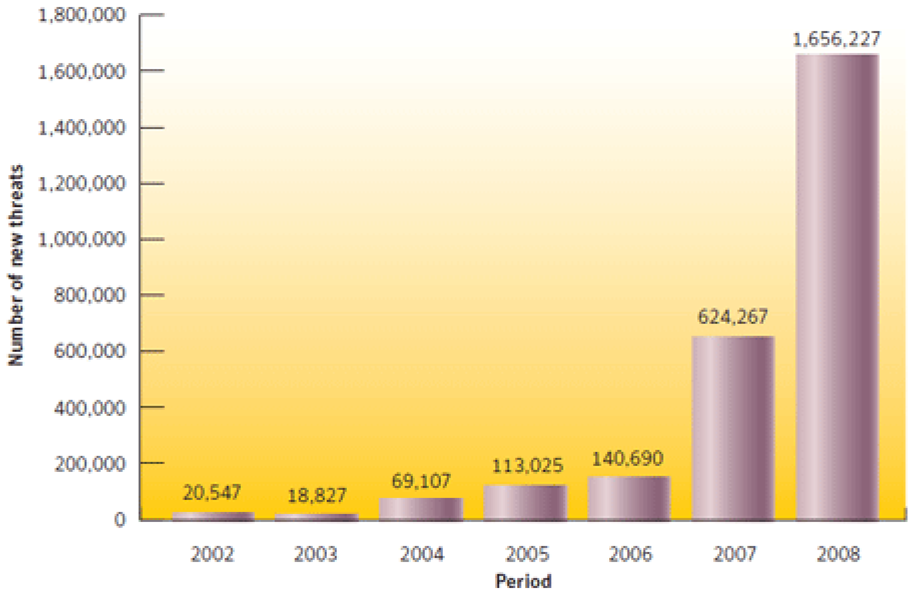
- Malware often affects a large user population
- Significant financial impact, though estimates vary widely, up to $100B per year (mi2g)
- Examples
  - LoveBug (2000) caused $8.75B in damages and shut down the British parliament
  - In 2004, 8% of emails infected by W32/MyDoom.A at its peak
  - In February 2006, the Russian Stock Exchange was taken down by a virus.



Economics of Malware

- New malware threats have grown from 20K to 1.7M in the period 2002-2008
- Most of the growth has been from 2006 to 2008
- Number of new threats per year appears to be growing an exponential rate.

Source:
[Symantec Internet Security Threat Report](), April 2009



Professional Malware

- Growth in professional cybercrime and online fraud has led to demand for professionally developed malware
- New malware is often a custom-designed variations of known exploits, so the malware designer can sell different “products” to his/her customers.
- Like every product, professional malware is subject to the laws of supply and demand.
  - Recent studies put the price of a software keystroke logger at $23 and a botnet use at $225.

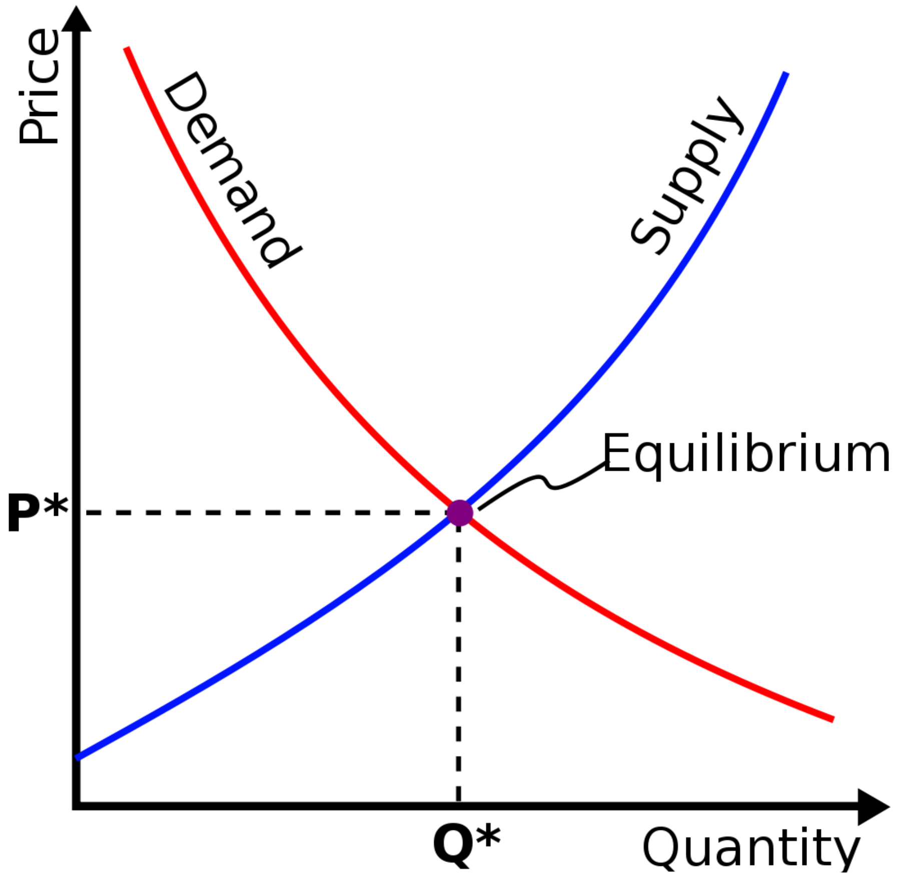
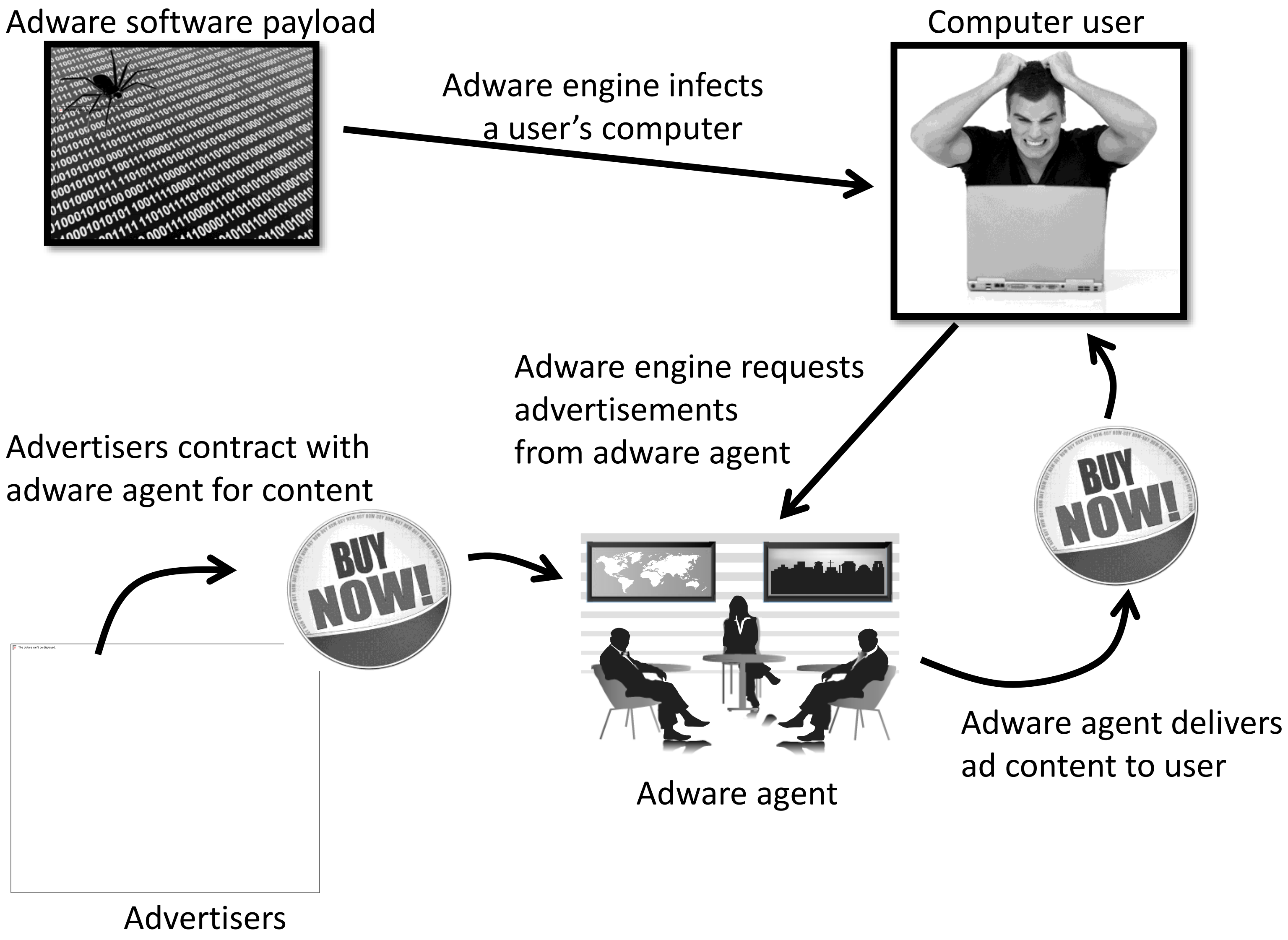


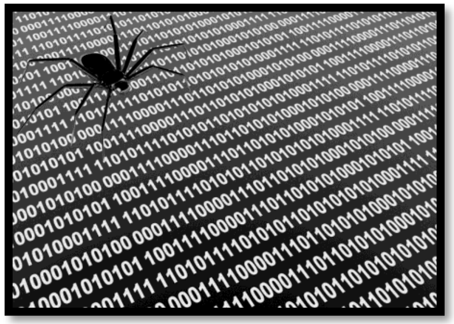
Image by User:SilverStar from http://commons.wikimedia.org/wiki/File:Supply-demand-equilibrium.svg used by permission under the *Creative Commons Attribution ShareAlike 3.0 License*

Adware


Adware software payload
Adware engine infects a user's computer
Computer user
Adware engine requests advertisements from adware agent
Advertisers contract with adware agent for content
BUY NOW!
BUY NOW!
Adware agent
Adware agent delivers ad content to user
Advertisers

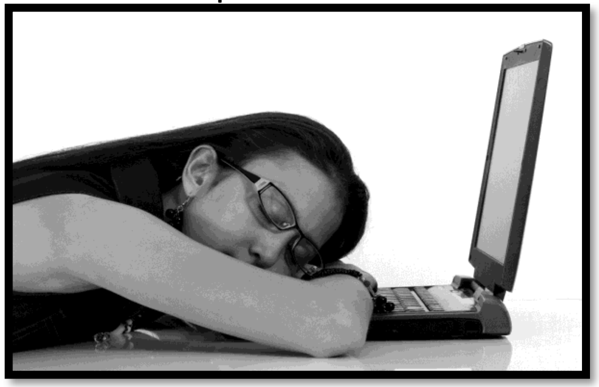
Spyware

Spyware software payload



1. Spyware engine infects a user's computer.

Computer user



2. Spyware process collects keystrokes, passwords, and screen captures.

3. Spyware process periodically sends collected data to spyware data collection agent.

Spyware data collection agent

Types of Malware

Code Type	Characteristics
Virus	Code that causes malicious behavior and propagates copies of itself to other programs
Trojan horse	Code that contains unexpected, undocumented, additional functionality
Worm	Code that propagates copies of itself through a network; impact is usually degraded performance
Rabbit	Code that replicates itself without limit to exhaust resources
Logic bomb	Code that triggers action when a predetermined condition occurs
Time bomb	Code that triggers action when a predetermined time occurs
Dropper	Transfer agent code only to drop other malicious code, such as virus or Trojan horse
Hostile mobile code agent	Code communicated semi-autonomously by programs transmitted through the web
Script attack, JavaScript, Active code attack	Malicious code communicated in JavaScript, ActiveX, or another scripting language, downloaded as part of displaying a web page

Types of Malware (cont.)

Code Type	Characteristics
RAT (remote access Trojan)	Trojan horse that, once planted, gives access from remote location
Spyware	Program that intercepts and covertly communicates data on the user or the user's activity
Bot	Semi-autonomous agent, under control of a (usually remote) controller or "herder"; not necessarily malicious
Zombie	Code or entire computer under control of a (usually remote) program
Browser hijacker	Code that changes browser settings, disallows access to certain sites, or redirects browser to others
Rootkit	Code installed in "root" or most privileged section of operating system; hard to detect
Trapdoor or backdoor	Code feature that allows unauthorized access to a machine or program; bypasses normal access control and authentication
Tool or toolkit	Program containing a set of tests for vulnerabilities; not dangerous itself, but each successful test identifies a vulnerable host that can be attacked
Scareware	Not code; false warning of malicious code attack

Malware

- We see that many types of malware exist
- Consequences of attacks can be significant
- How do we protect against them?

Countermeasures for Users

- Use software acquired from reliable sources
- Test software in an isolated environment
- Only open attachments when you know them to be safe
- Treat every website as potentially harmful
- Create and maintain backups

Security Principles

Making decisions about technology in an uncertain world

https://criticaluncertainties.com/category/security/saltzer-and-schroeder-principles/

Security Principles

- How to design of secure software systems?
- Saltzer and Schroeder's design principles
  - Published in The Protection of Information in Computer Systems [1975]

Design Principles for Security

- Least privilege
- Economy of mechanism
- Fail-safe defaults
- Open design
- Complete mediation
- Separation of privilege
- Least common mechanism
- Psychological Acceptability - Ease of use

Least privilege

- Each program and user of a computer system should operate with the bare **minimum privileges necessary** to function properly.
  - If this principle is enforced, abuse of privileges is restricted, and the damage caused by the compromise of a particular application or user account is minimized.
  - The military concept of **need-to-know** information is an example of this principle.

Fail-safe defaults

- This principle states that the default configuration of a system should have a **conservative protection scheme**.
    - For example, when adding a new user to an operating system, the default group of the user should have minimal access rights to files and services. Unfortunately, operating systems and applications often have default options that favor usability over security.
    - This has been historically the case for a number of popular applications, such as web browsers that allow the execution of code downloaded from the web server.

Economy of mechanism

- This principle stresses **simplicity** in the **design** and **implementation** of security measures.
  - While applicable to most engineering endeavors, the notion of simplicity is especially important in the security domain, since a simple security framework facilitates its understanding by developers and users and enables the efficient development and verification of enforcement methods for it.

Open design

- According to this principle, the security architecture and **design** of a system should be made **publicly available**.
  - Security should rely only on keeping cryptographic keys secret.
  - Open design allows for a system to be scrutinized by multiple parties, which leads to the early discovery and correction of security vulnerabilities caused by design errors.
  - The open design principle is the opposite of the approach known as **security by obscurity,** which tries to achieve security by keeping cryptographic algorithms secret and which has been historically used without success by several organizations.

Kerckhoffs's principle

- A cryptosystem should be secure even if everything about the system, except the key, is public knowledge
  - Auguste Kerckhoffs, [19th century]

Kerckhoffs's principle

- Design principles for military ciphers:
  - System must be practically indecipherable
    - If not mathematically
  - Should not require secrecy
    - it should not be a problem if it falls into enemy hands
  - It must be possible to communicate and remember the key
    - Without written notes
    - Parties must be able to change or modify it at will;
  - Should be portable and applicable to telegraph communications
    - should not require several persons to handle or operate;
  - System must be easy to use
    - should not require users to know and comply with a long list of rules.
      - given the circumstances in which it is to be used, the

Open design

- According to this principle, the security architecture and **design** of a system should be made **publicly available**.
  - Security should rely only on keeping cryptographic keys secret.
  - Open design allows for a system to be scrutinized by multiple parties, which leads to the early discovery and correction of security vulnerabilities caused by design errors.
  - The open design principle is the opposite of the approach known as **security by obscurity,** which tries to achieve security by keeping cryptographic algorithms secret and which has been historically used without success by several organizations.

Complete mediation

- The idea behind this principle is that every access to a resource must be checked for **compliance with a protection scheme**.
  - As a consequence, one should be wary of performance improvement techniques that save the results of previous authorization checks, since permissions can change over time.
  - For example, an online banking web site should require users to sign on again after a certain amount of time, say, 15 minutes, has elapsed.

Separation of privilege

- This principle dictates that **multiple conditions** should be required to achieve access to restricted resources or have a program perform some action.

Least common mechanism

- In systems with multiple users, mechanisms allowing resources to be **shared by more than one user should be minimized**.
  - For example, if a file or application needs to be accessed by more than one user, then these users should have separate channels by which to access these resources, to prevent unforeseen consequences that could cause security problems.

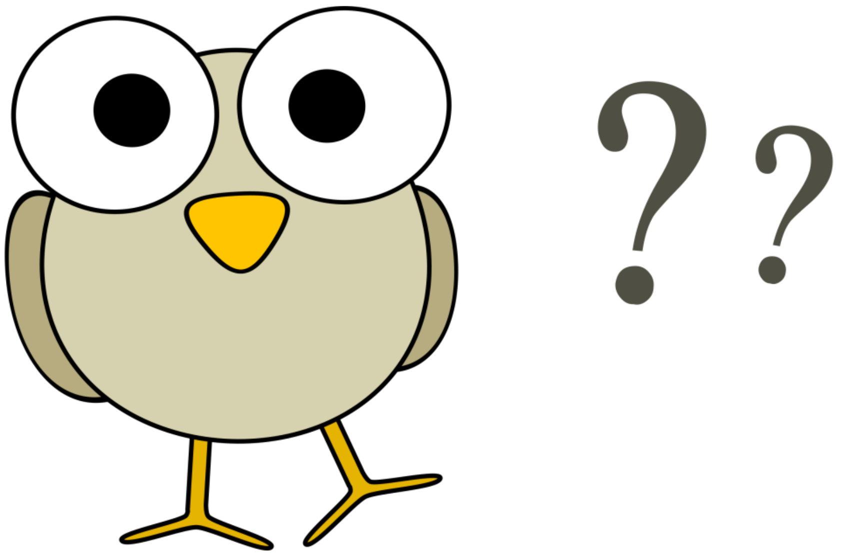
Psychological acceptability - Ease of use

- This principle states that user interfaces should be **well designed and intuitive**, and all security-related settings should adhere to what an ordinary user might expect.

Summary

- Malware can have a variety of harmful effects depending on its characteristics, including resource usage, infection vector, and payload
- Developers can use a variety of techniques for writing and testing code for security
- Following design principles helps protect systems against attacks

- Questions?



What about Internet of Things?

Internet of Things (IoT)

- A network of physical devices, vehicles, home appliances and other items
- Devices are embedded with electronics, software, sensors, actuators
- Network connectivity enables these objects to connect and exchange data
  - But can be a source of vulnerabilities

“Internet of Things” devices

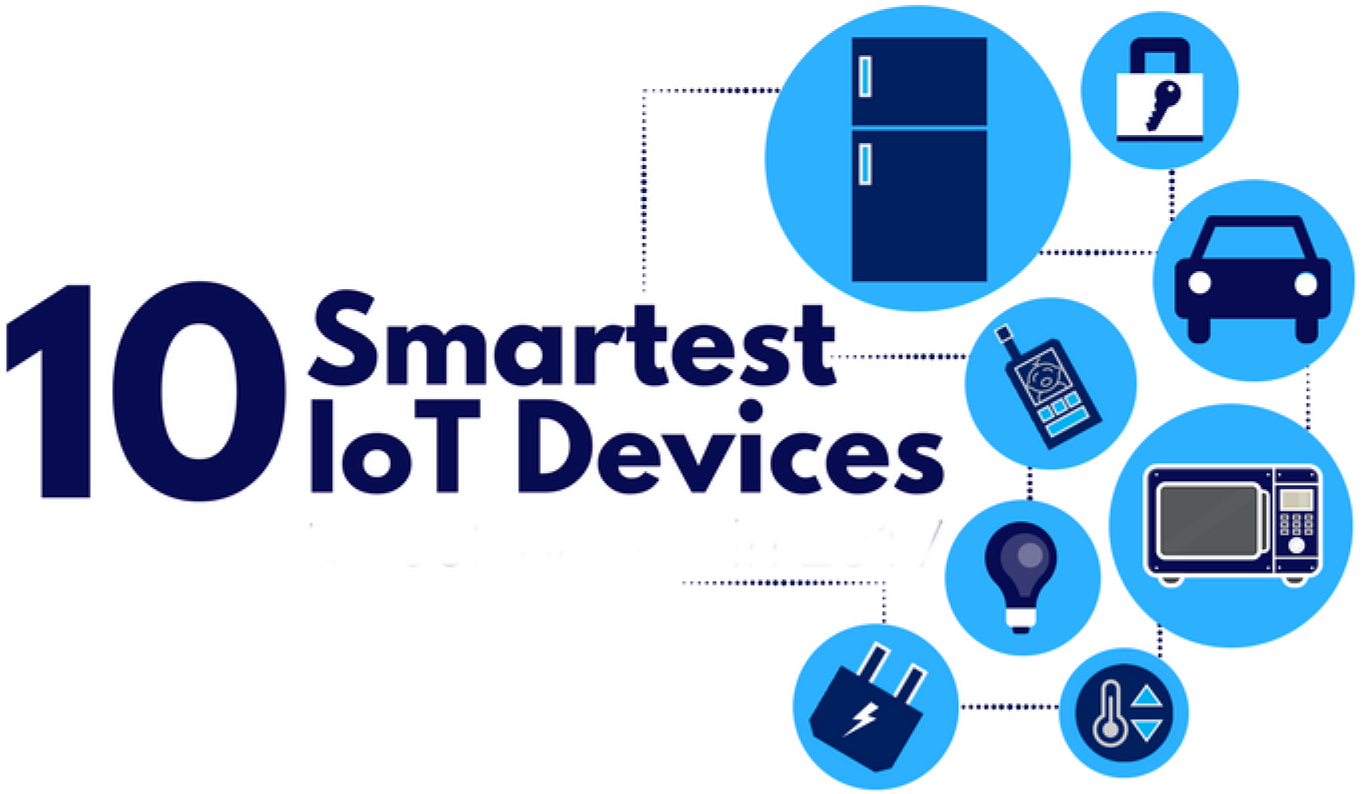
- “Internet of things” devices are connected through the network
- Attackers may access device vulnerabilities through the network
- Devices are typically very cost sensitive
  - Therefore, very little support after purchase
  - User can not tell if they are secure

Example of IoT devices

- Wearable devices
- Medical devices
- Home automation
  - Refrigerators, stoves, etc.
- Automative
- Etc.

https://www.pcworld.com/article/3150801/security/privacy-protections-for-wearable-devices-are-weak-study-says.html
https://www.slideshare.net/Losant/what-is-the-internet-of-things-iot

IoT Devices



https://www.pentasecurity.com/blog/10-smartest-iot-devices-2017/

“Internet of Things” devices

- Device only communicates through a central service
- Most of the companies running the service are “Data Asset” companies
  - Make their money from advertising, not from the product itself
    - Product may actually be subsidized considerably
  - Companies include Google, Amazon, Salesforce, etc…
  - Apple HomeKit provides a higher level of security
    - But you still need to trust that it does not report to a third party

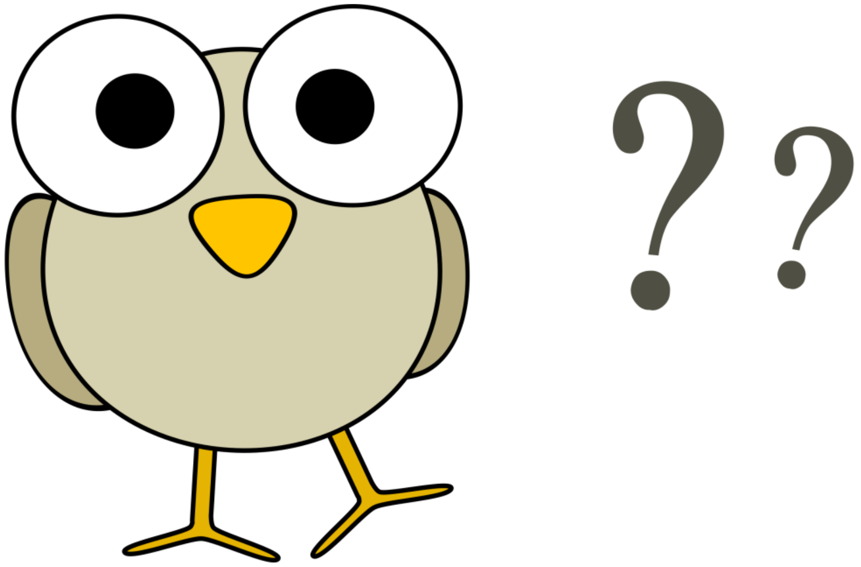
Risks in using IoT devices

- Devices collect information about user via device
- Many devices do not require strong passwords (or any passwords)
- Devices may use unencrypted network services
- User interface of applications may be vulnerable to different attacks
  - Shoulder surfing, etc.

“Internet of Things” devices

- Goal: Increase security of devices
- Possible defenses:
  - Develop applications with security in mind
    - Consider security throughout the entire development lifecycle of IoT devices
      - Not treat it separately or as an “add-on”
  - Integrate human understanding and algorithms
    - Take in account the “human factor”

- Questions?



Computer Security Quiz

What is a computer network?

- A. A super computer owned only by the government
- B. A web of connected computers or devices
- C. A computer vulnerability
- D. An Internet service provider
- E. All of the above

What is a computer network?

- A. A super computer owned only by the government
- [✓ B. A web of connected computers or devices]
- C. A computer vulnerability
- D. An Internet service provider
- E. All of the above

Why are cyber vulnerabilities unlikely to ever go away?

- A. Criminals need them to steal identities.
- B. They are side effects of the freedom and ease of communicating online.
- C. They're protected in a secret base on the moon.
- D. The government won't allow people to fix them.

Why are cyber vulnerabilities unlikely to ever go away?

- A. Criminals need them to steal identities.
- [✓ B. They are side effects of the freedom and ease of communicating online.]
- C. They're protected in a secret base on the moon.
- D. The government won't allow people to fix them.

The size and complexity of networks grew enormously when:

A. Only governments and universities owned computers.

B. Spamware caused some computers to break down

C. The number of personal computers greatly increased.

D. The hacktivists started using the internet

The size and complexity of networks grew enormously when:

A. Only governments and universities owned computers.

B. Spamware caused some computers to break down

[✓ C. The number of personal computers greatly increased.]

D. The hacktivists started using the internet

- Questions?