Lecture 07: Malware

Professor Adam Bates
CS 461 / ECE 422
Fall 2019

Goals for Today

- Learning Objectives:
  - Define malware in terms of means and motivates of attackers.
  - Identify and evaluate several examples of malware
- Announcements, etc:
  - Office hours all the time! M-F 5-7pm, 4405 Siebel
  - Prof Bates does not have office hours this week
  - MP1 is live!
    - Checkpoint #2: **Due Sept 18 at 6pm**

Reminder: Please put away devices at the start of class

Malware: Definition and Goals

- "Malware" is short for malicious software and is typically used as a catch-all term to refer to any software designed to cause damage to a single computer, server, or computer network
- Encompasses computer viruses, worms, Trojans, ransomware, spyware, adware, Backdoors, Logic bombs, keyloggers, rootkits…

Malware Prevalence

Malware encounter rates by country:

Country/Region	3Q15	4Q15	1Q16	2Q16
United States	10.8%	12.5%	11.9%	12.0%
China	14.9%	18.9%	19.1%	21.1%
Brazil	29.2%	34.4%	29.9%	29.4%
Russia	22.8%	28.7%	27.2%	24.9%
India	36.5%	44.2%	35.4%	32.6%
Turkey	32.6%	40.3%	34.8%	31.4%
France	18.8%	19.4%	17.0%	15.3%
Mexico	23.9%	28.5%	24.4%	23.8%
United Kingdom	11.9%	13.9%	13.7%	11.5%
Germany	12.2%	13.8%	13.0%	13.0%
Worldwide	*17.8%*	*20.8%*	*18.3%*	*21.2%*

Up-to-date, Interactive Info at https://www.microsoft.com/securityinsights/Malware

Malware Prevalence

Malware encounter rates by country:

Country/Region	3Q15	4Q15	1Q16	2Q16
United States	10.8%	12.5%	11.9%	12.0%
China	14.9%	18.9%	19.1%	21.1%
Brazil	29.2%	34.4%	29.9%	29.4%
Russia	22.8%	28.7%	27.2%	24.9%
India	36.5%	44.2%	35.4%	32.6%
Turkey	32.6%	40.3%	34.8%	31.4%
France	18.8%	19.4%	17.0%	15.3%
Mexico	23.9%	28.5%	24.4%	23.8%
United Kingdom	11.9%	13.9%	13.7%	11.5%
Germany	12.2%	13.8%	13.0%	13.0%
Worldwide	*17.8%*	*20.8%*	*18.3%*	*21.2%*

Why isn't US leading in malware encounters?

Up-to-date, Interactive Info at https://www.microsoft.com/securityinsights/Malware

How does it manage to run?

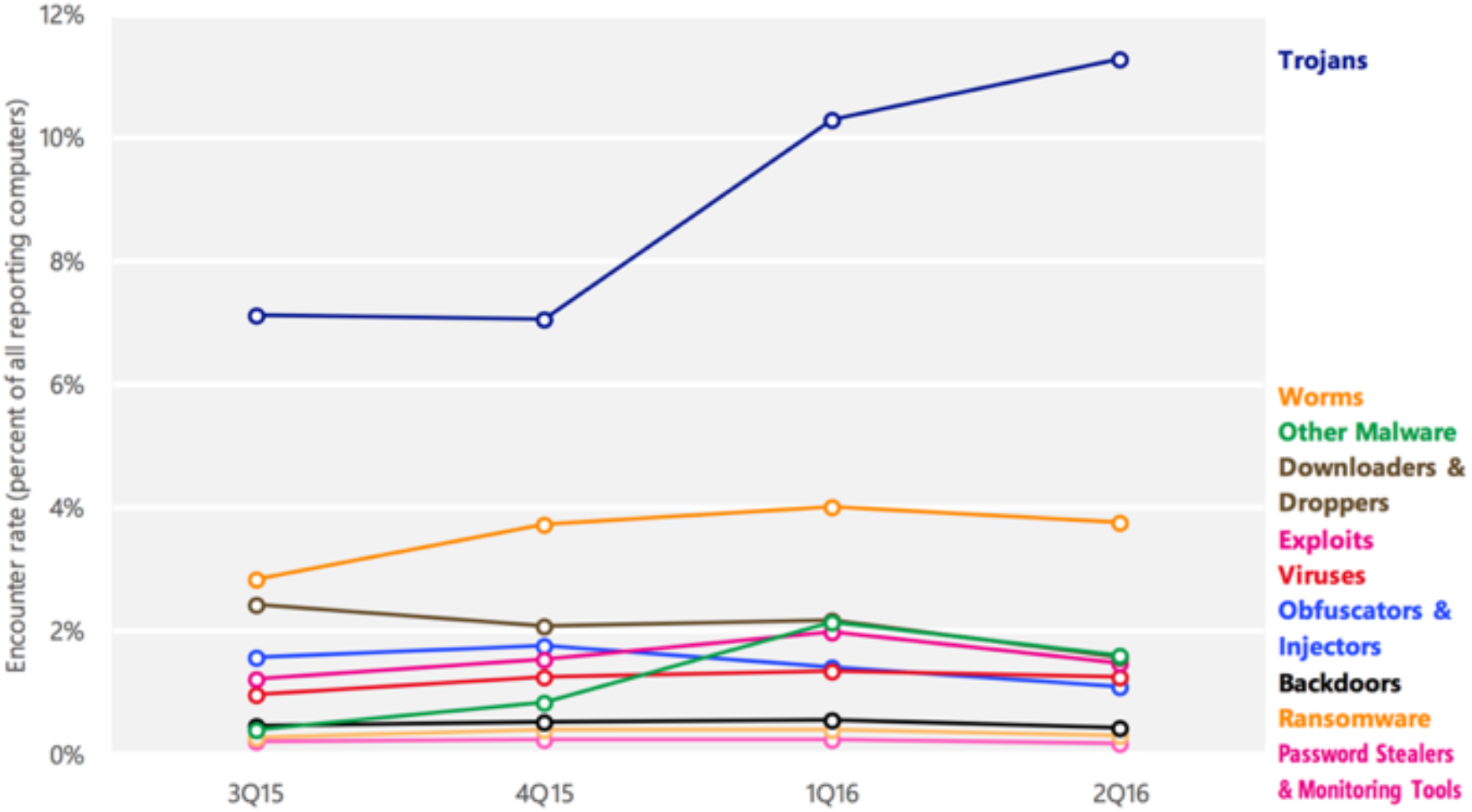
- Buffer overflow in network-accessible vulnerable service
- Vulnerable client (e.g. browser) connects to remote system that sends over an attack (a driveby)
- Social engineering: trick user into running/installing
- "Autorun" functionality (esp. from plugging in USB device)
- Slipped into a system component (at manufacture; compromise of software provider; substituted via MITM)
- Attacker with local access downloads/runs it directly
  - Might include using a "local root" exploit for privileged access, aka privilege escalation

What can Malware do?

- Pretty much anything
  - Payload generally decoupled from how manages to run
  - Only subject to permissions under which it runs
- Examples:
  - Brag or exhort or extort (pop up a message/display)
  - Trash files (just to be nasty)
  - Damage hardware (Stuxnet?)
  - Launch external activity (spam, click fraud, DoS)
  - Steal information (exfiltrate)
  - Keylogging; screen / audio / camera capture
    - Robbins v. Lower Merion School District
- Encrypt files (ransomware)
- Possibly delayed until condition occurs
  - "time bomb" / "logic bomb"

Malware Prevalence

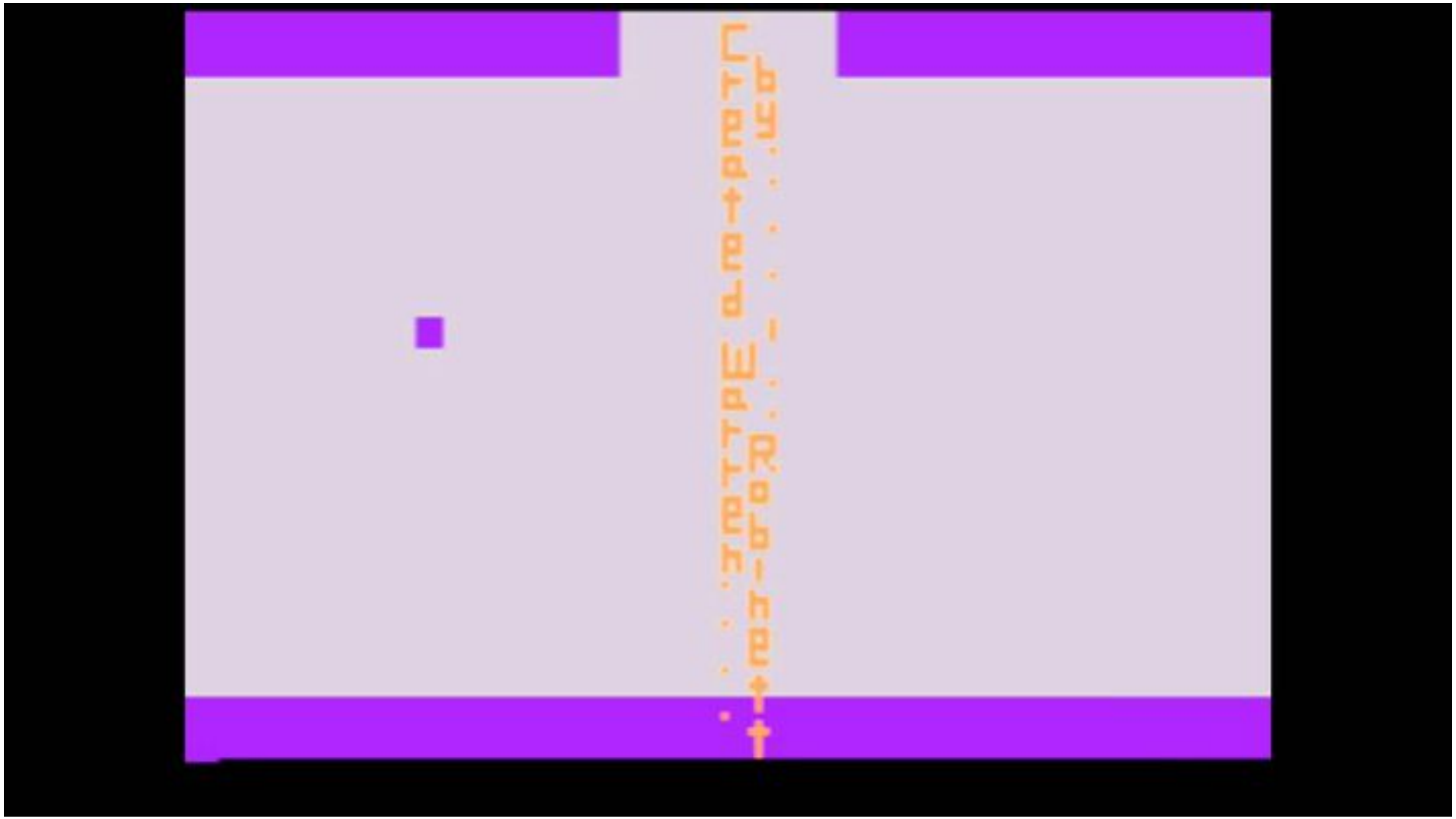
What are the most common types of malware?



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

Easter Eggs: Backdoors?



Adventure (Atari 2600)

Juniper's Surprising Announcement

- Problem: During an internal code review, two security issues were identified:

  1. Administrative Access (CVE-2015-7755) allows unauthorized remote administrative access to the device. Exploitation of this vulnerability can lead to complete compromise of the affected device.

  2. VPN Decryption (CVE-2015-7756) may allow a knowledgeable attacker who can monitor VPN traffic to decrypt that traffic. It is independent of the first issue.

- Affected devices and firmware:
  - Juniper's Secure Services Gateway firewall/VPN appliances
  - Various revisions of ScreenOS 6.2 and 6.3

Juniper Backdoor

- Administrative access backdoor:

```
ROM:0013DBE8                STMFD           SP!, {R4-R8,R11,R12,LR,PC}
ROM:0013DBEC                SUB             R11, R12, #4
ROM:0013DBF0                SUB             SP, SP, #0x10
ROM:0013DBF4                MOV             R5, R0
ROM:0013DBF8                MOV             R6, #0
ROM:0013DBFC                MOV             R7, R6
ROM:0013DC00                MOV             R8, R6
ROM:0013DC04                LDR             R3, =dword_1E7FCF0
ROM:0013DC08                LDR             R12, [R3]
ROM:0013DC0C                CMP             R12, R6
ROM:0013DC10                BEQ             loc_13DC54
ROM:0013DC14                ADD             R0, R0, #0x6C
ROM:0013DC18                BL              sub_402438
ROM:0013DC1C                MOV             R4, R0
ROM:0013DC20                ADD             R0, R5, #0x80
ROM:0013DC24                BL              sub_402438
ROM:0013DC28                LDRH            R2, [R5,#0x68]
ROM:0013DC2C                ADD             R3, R5, #4
ROM:0013DC30                STR             R4, [SP,#0x30+var_30]
ROM:0013DC34                STR             R0, [SP,#0x30+var_2C]
ROM:0013DC38                LDRH            R12, [R5,#0x94]
ROM:0013DC3C                STR             R12, [SP,#0x30+var_28]
ROM:0013DC40                LDRH            R12, [R5,#0x96]
ROM:0013DC44                STR             R12, [SP,#0x30+var_24]
ROM:0013DC48                LDR             R0, =aSCtUUnSSipSDip ; ">>> %s(ct=%u, un='%s',
ROM:0013DC4C                LDR             R1, =aAuth_admin_int ; "auth_admin_internal"
ROM:0013DC50                BL              sub_558810
ROM:0013DC54
ROM:0013DC54 loc_13DC54                             ; CODE XREF: auth_admin_internal+2C↑j
ROM:0013DC54                ADD             R0, R5, #0x6C
ROM:0013DC58                BL              sub_147224
ROM:0013DC5C                MOVS            R0, R0,LSL#16
ROM:0013DC60                MOVNE           R7, #1
ROM:0013DC64                BNE             loc_13DDD8
ROM:0013DC68                LDRH            R12, [R5,#0x68]
ROM:0013DC6C                ADD             R12, R12, #0xFF00
ROM:0013DC70                ADD             R12, R12, #0xFE
ROM:0013DC74                MOV             R12, R12,LSL#16
ROM:0013DC78                CMP             R12, #0x20000
ROM:0013DC7C                BHI             loc_13DCB4
ROM:0013DC80                ADD             R4, R5, #4
ROM:0013DC84                MOV             R0, R4
ROM:0013DC88                BL              sub_14141C
ROM:0013DC8C                CMP             R0, #0
ROM:0013DC90                BLE             loc_13DCB4
ROM:0013DC94                MOV             R8, #1
```

Original

```
ROM:0013DBF0                STMFD           SP!, {R4-R8,R11,R12,LR,PC}
ROM:0013DBF4                SUB             R11, R12, #4
ROM:0013DBF8                SUB             SP, SP, #0x10
ROM:0013DBFC                MOV             R5, R0
ROM:0013DC00                MOV             R6, #0
ROM:0013DC04                MOV             R7, R6
ROM:0013DC08                MOV             R8, R6
ROM:0013DC0C                LDR             R3, =dword_1E7FCF0
ROM:0013DC10                LDR             R12, [R3]
ROM:0013DC14                CMP             R12, R6
ROM:0013DC18                BEQ             loc_13DC5C
ROM:0013DC1C                ADD             R0, R0, #0x6C
ROM:0013DC20                BL              sub_402B9C
ROM:0013DC24                MOV             R4, R0
ROM:0013DC28                ADD             R0, R5, #0x80
ROM:0013DC2C                BL              sub_402B9C
ROM:0013DC30                LDRH            R2, [R5,#0x68]
ROM:0013DC34                ADD             R3, R5, #4
ROM:0013DC38                STR             R4, [SP,#0x30+var_30]
ROM:0013DC3C                STR             R0, [SP,#0x30+var_2C]
ROM:0013DC40                LDRH            R12, [R5,#0x94]
ROM:0013DC44                STR             R12, [SP,#0x30+var_28]
ROM:0013DC48                LDRH            R12, [R5,#0x96]
ROM:0013DC4C                STR             R12, [SP,#0x30+var_24]
ROM:0013DC50                LDR             R0, =aSCtUUnSSipSDip ; ">>> %s(ct=%u, un='%s',
ROM:0013DC54                LDR             R1, =aAuth_admin_int ; "auth_admin_internal"
ROM:0013DC58                BL              sub_558F74
ROM:0013DC5C
ROM:0013DC5C loc_13DC5C                             ; CODE XREF: auth_admin_internal+2C↑j
ROM:0013DC5C                ADD             R0, R5, #0x44
ROM:0013DC60                LDR             R1, =aSUnSU ; "<<< %s(un='%s') = %u"
ROM:0013DC64                BL              strcmp
ROM:0013DC68                CMP             R0, #0
ROM:0013DC6C                BNE             loc_13DC78
ROM:0013DC70                MOV             R0, #0xFFFFFFFD
ROM:0013DC74                LDMDB           R11, {R4-R8,R11,SP,PC}
ROM:0013DC78 ; ---------------------------------------------------------------------------
ROM:0013DC78
ROM:0013DC78 loc_13DC78                             ; CODE XREF: auth_admin_internal+80↑j
ROM:0013DC78                ADD             R0, R5, #0x6C
ROM:0013DC7C                BL              sub_14724C
ROM:0013DC80                MOVS            R0, R0,LSL#16
ROM:0013DC84                MOVNE           R7, #1
ROM:0013DC88                BNE             loc_13DDFC
ROM:0013DC8C                LDRH            R12, [R5,#0x68]
ROM:0013DC90                ADD             R12, R12, #0xFF00
```

Modified

Dual EC DRBG Timeline

Dual_EC_DRBG (Dual Elliptic Curve Deterministic Random Bit Generator) is an algorithm that was presented as a cryptographically secure pseudorandom number generator (CSPRNG) using methods in elliptic curve cryptography.

- Early 2000s: Created by the NSA and pushed towards standardization
- 2004: Published as part of ANSI x9.82 part 3 draft
- 2004: RSA makes Dual EC the default CSPRNG in BSAFE (for $10MM)
- 2005: Standardized in NIST SP 800-90
- 2007: Shumow and Ferguson demonstrate a theoretical backdoor attack
- 2013: Snowden documents lead to renewed interest in Dual EC
- 2014: Practical attacks on TLS using Dual EC demonstrated
- 2014: NIST removes Dual EC from list of approved PRNGs
- 2016: Practical attacks on IKE using Dual EC (this work)

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


IDG

FEBRUARY 23, 1998 VOLUME 15, NUMBER 8

NetworkWorld

THE NEWSWEEKLY OF ENTERPRISE NETWORK COMPUTING

Going with Gigabit Ethernet

GMMC's Niraj Patel has learned firsthand about the high-speed net technology. Page 9.

- An example of a logic bomb that was actually triggered and caused damage is one that 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, which ultimately cost the company millions of dollars in damages and led to it laying off many of its employees.

A view into a network attack

Net administrator charged in $10M "logic bomb" case.

By Ellen Messmer
Bridgeport, Conn.

In one of the costliest reported acts of computer sabotage, an engineering company next month will prosecute its former network administrator for electronically destroying computer files that the company claims cost it about $10 million in sales.

Omega Engineering, Inc. is set to go to trial against Timothy Lloyd, the chief network program designer, who the company said planted a LAN-based logic bomb that went off after his job was terminated. The logic bomb wiped out all the files on the company's Novell, Inc. network-based servers.

What detonated the Omega bomb was not immediately clear.

Security experts said a logic bomb usually is a software program that, once activated by a specific date for example, eats through files or reformats hard drives. The bomber's intent is to hopelessly damage and erase data.

"[Logic bombs] can be as simple as a script that runs a bunch of delete commands," said Chris Byrnes, vice president for servers and systems management strategy

See Bomb, page 16

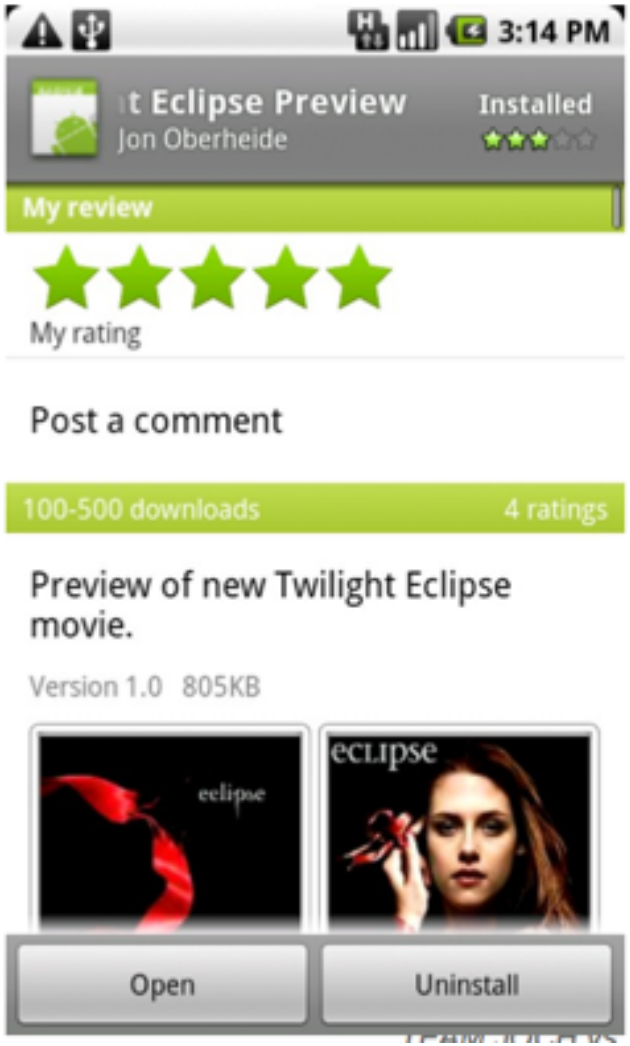
The Omega Bomb Code

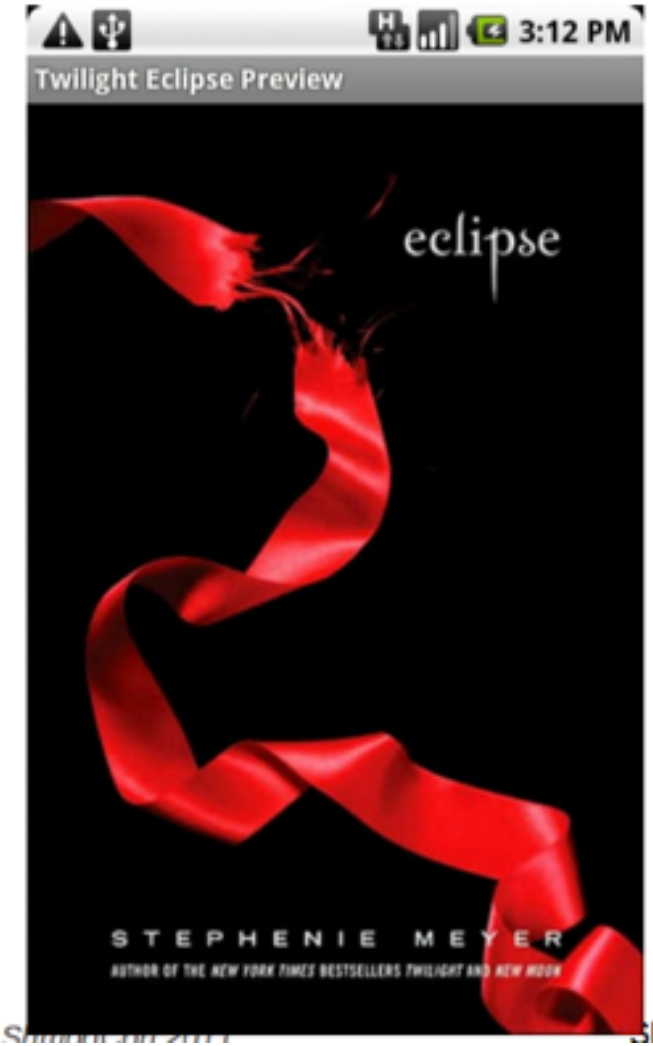
- The Logic Behind the Omega Engineering Time Bomb included the following strings:
- 7/30/96
  - Event that triggered the bomb
- F:
  - Focused attention to volume F, which had critical files
- F:\LOGIN\LOGIN 12345
  - Login a fictitious user, 12345 (the back door)
- CD \PUBLIC
  - Moves to the public 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

Trojan Horse

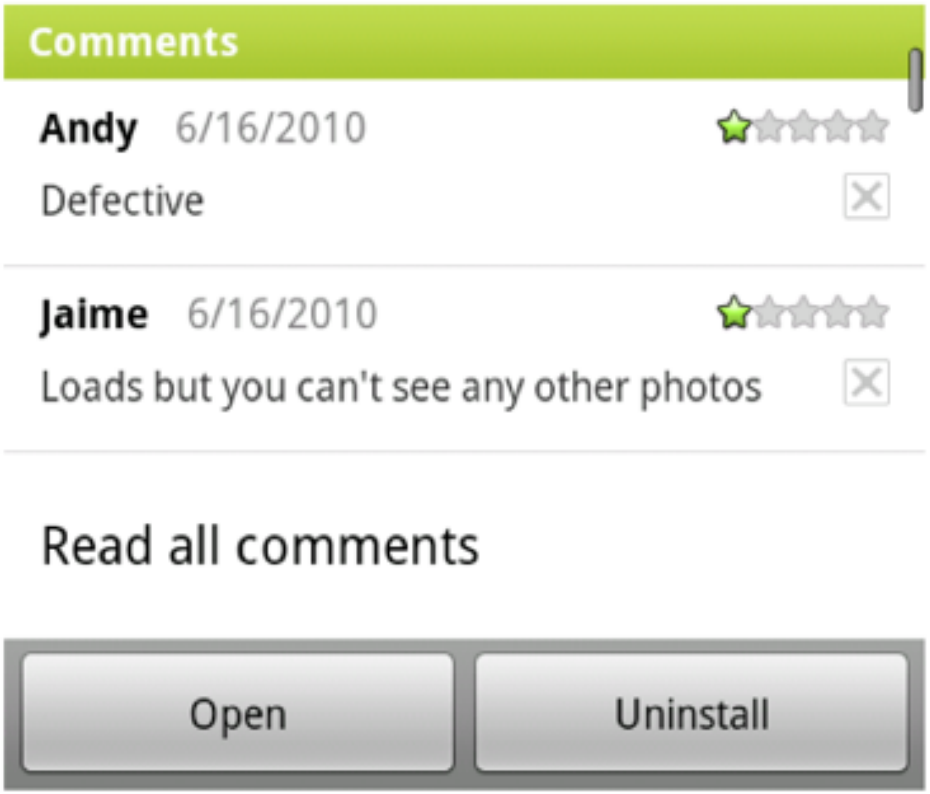
- Software that appears to perform a desirable function but is actually designed to perform undisclosed malicious functions
  - Spyware: installed by legitimate looking programs, then provides remote access to the computer, such as logging keys or sending back documents
  - Adware: shows popup ads
  - Ransomware: encrypts data and requires payment to decrypt

Android Trojan Horse





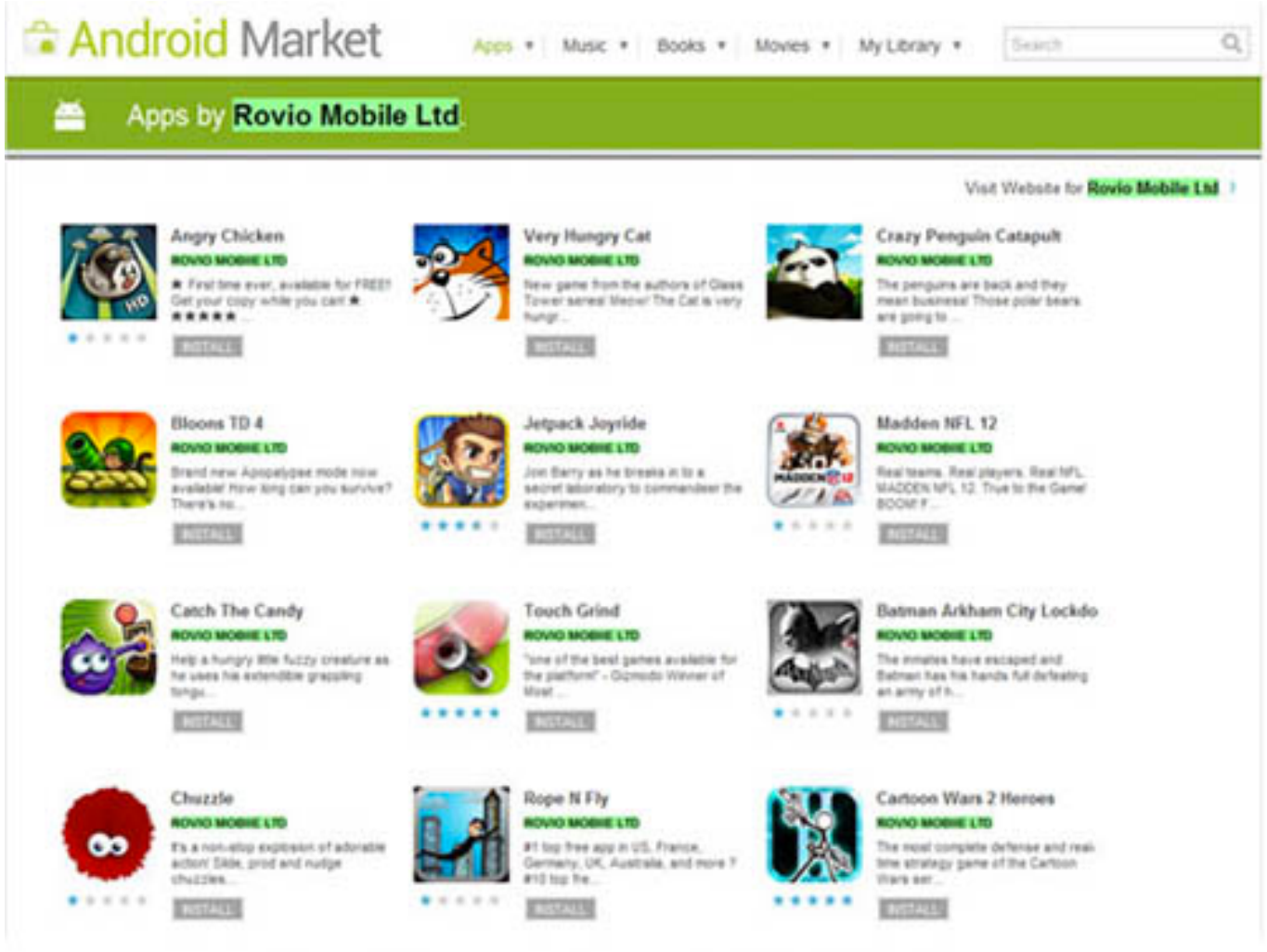
Android Trojan Horse



- Still, 200+ downloads in under 24 hours
- With a legit-looking app/game, you could collect quite an install base for RootStrap

Repackaging

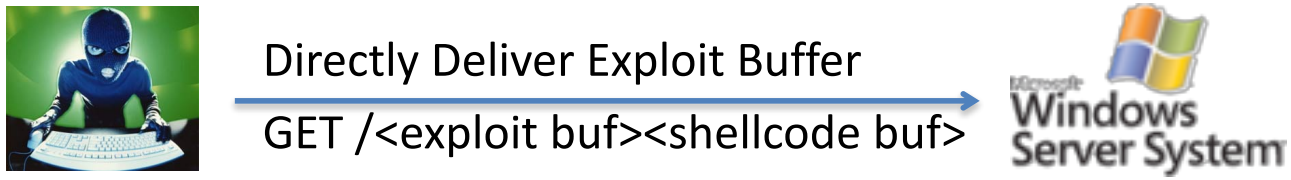
- Repackaging apps is pervasive on 3rd party android markets; trojans are common.



Code Injection Exploits

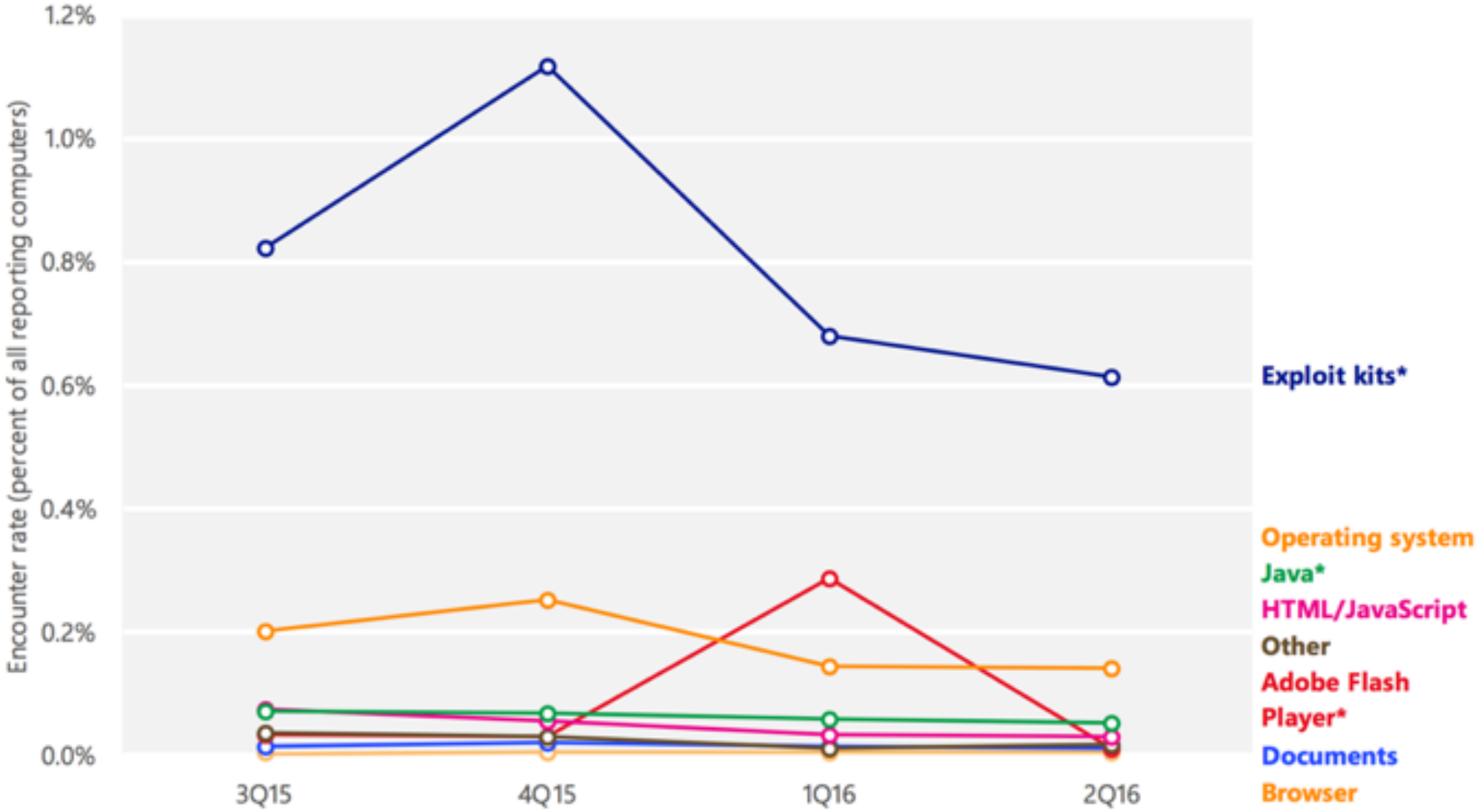
- Client software exploit (e.g. PDF, Flash, MSWord, etc.)

- Network-based exploit (HTTP, File, RPC servers, etc.)



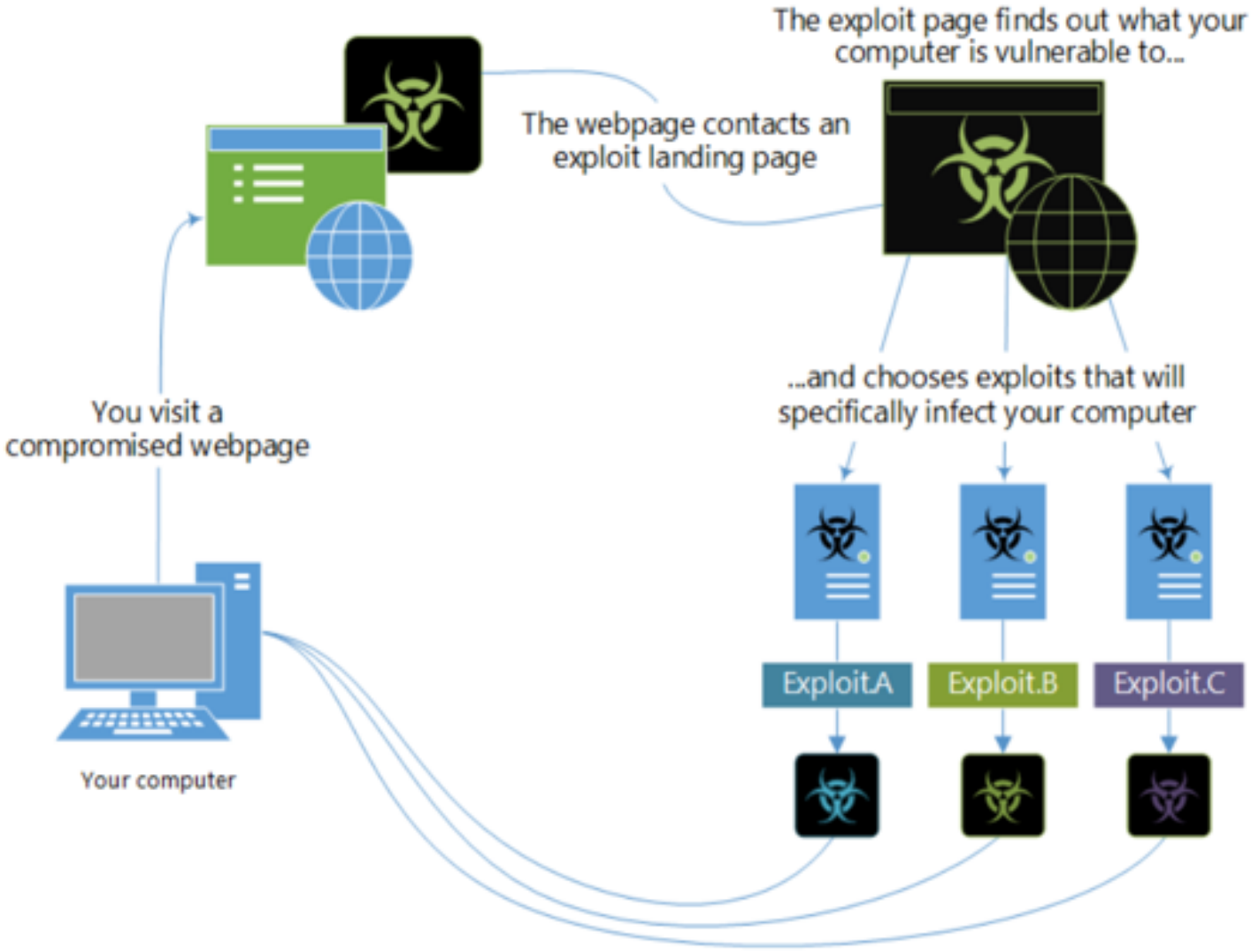
Code Injection Exploits

What are the vulnerabilities being exploited?



Exploit Kits

How a typical exploit kit works:



Malware that Automatically Propagates

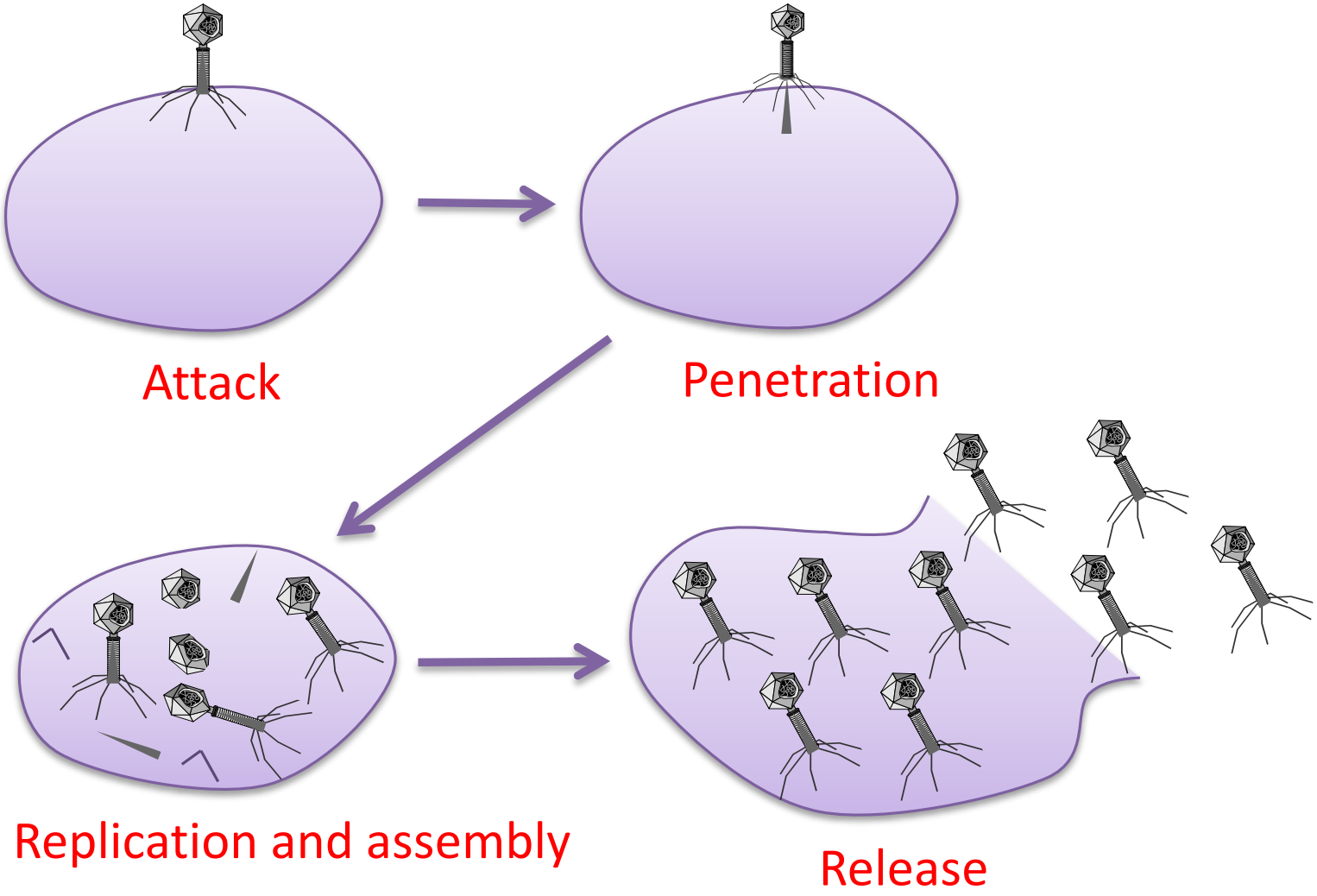
- Virus = code that propagates (**replicates**) across systems by arranging to have itself *eventually executed,* creating anew additional instance
  - Generally infects by altering stored code
  - Typically with the help of a user
- Worm = code that self-propagates/replicates across systems by arranging to have itself *immediately executed* (creating new addl. instance)
  - Generally infects by altering running code
  - No user intervention required
- (Note: line between these isn't always so crisp; plus some malware incorporates both styles)

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.

Computer Viruses

Computer viruses share some properties with biological viruses:

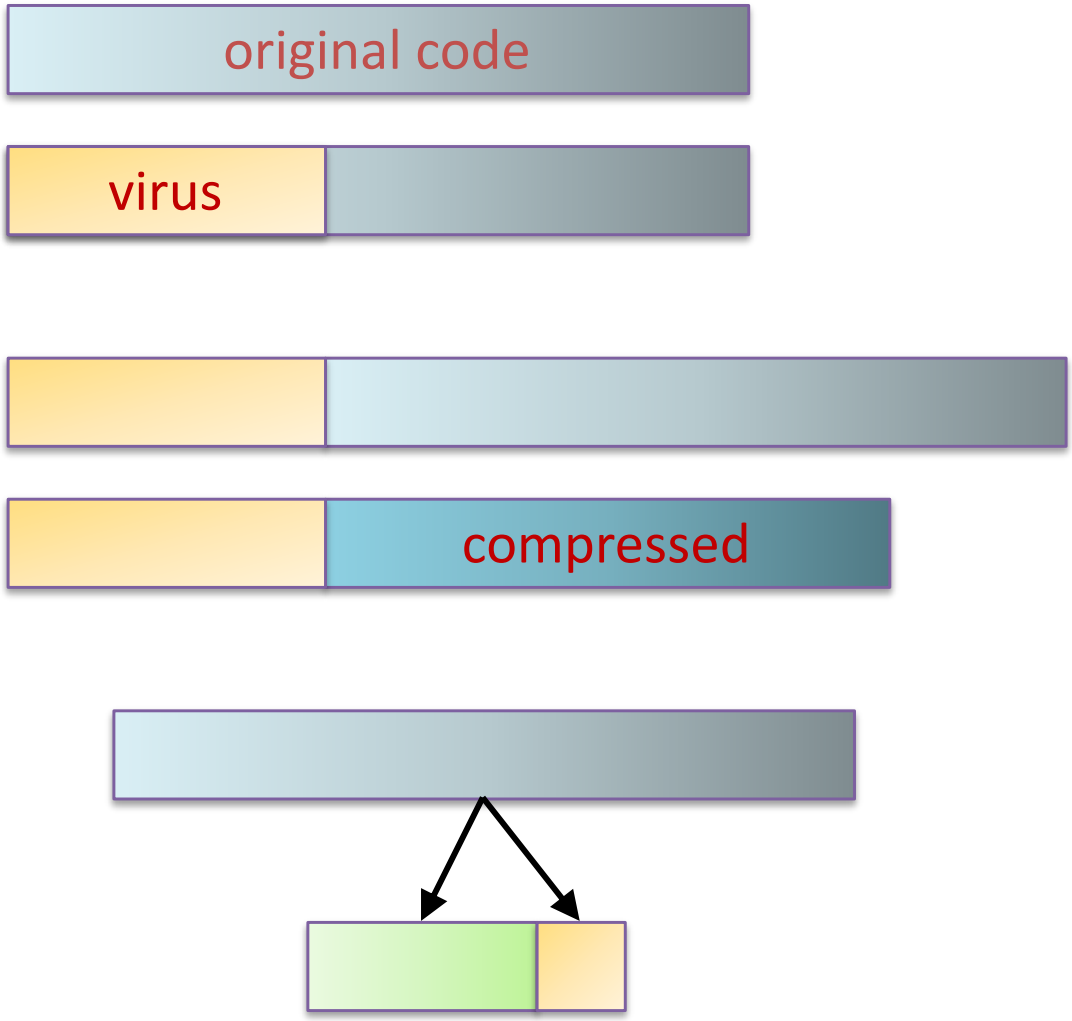


Virus Phases

- **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.

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



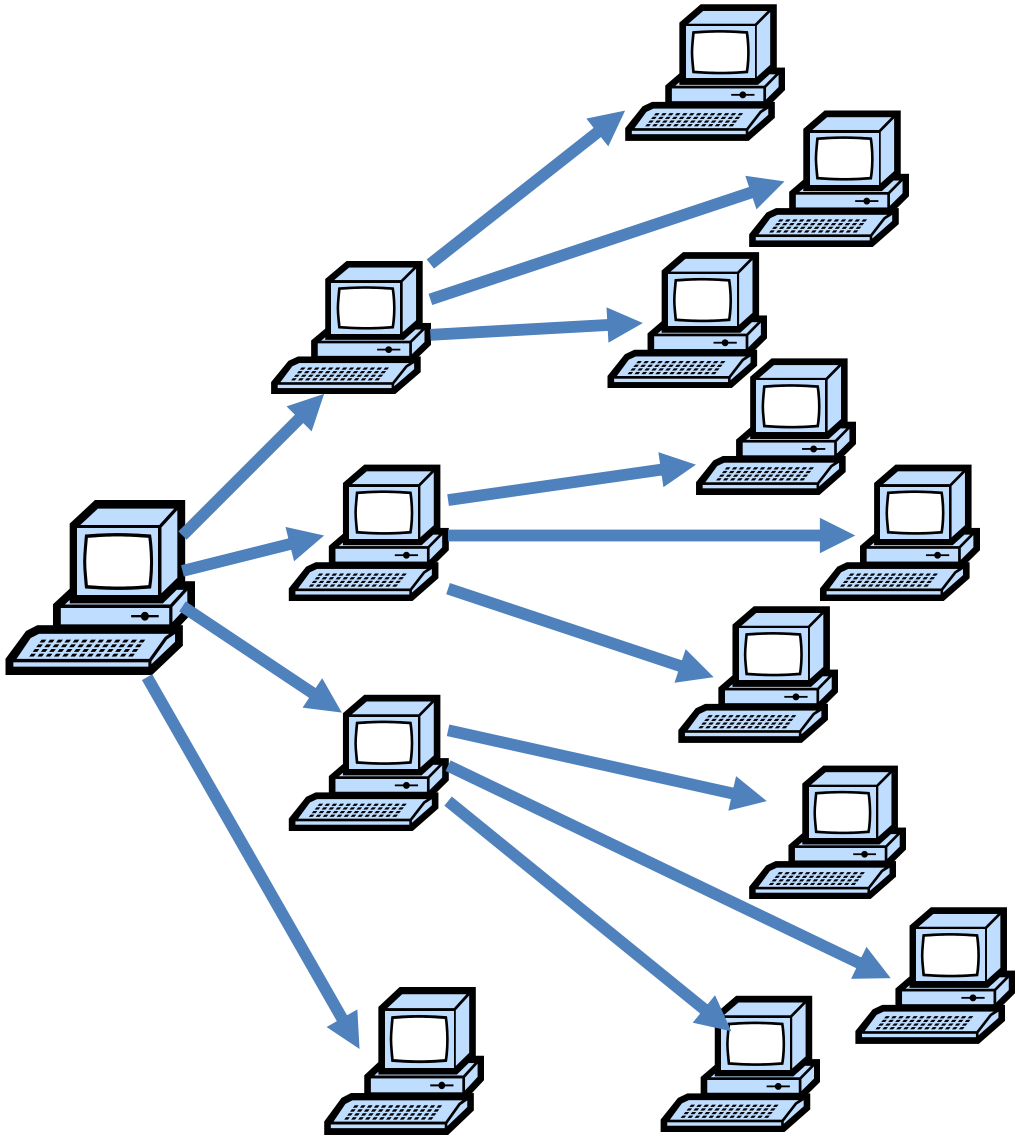
Worms (Preview)

- A worm is code that self-propagates/replicates across systems by arranging to have itself immediately executed
  - Generally infects machines by altering running code
  - No user intervention required

Work Propagation

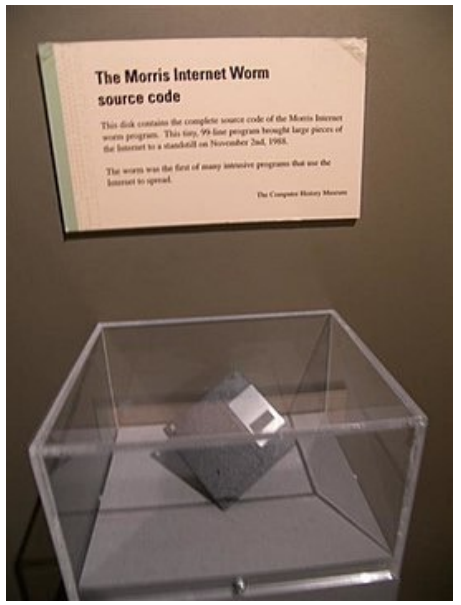
Worms can potentially spread quickly because they parallelize the process of propagating/ replicating.

Same holds for viruses, but they often spread more slowly since they require some sort of user action to trigger each propagation.



Morris Worm

- Used the Internet to infect a large number of machines in 1988
- Three Propagation Vectors
  - sendmail bug
    - default configuration allowed debug access
    - well known for several years, but not fixed
  - fingerd: finger adam@cs
    - fingerd allocated fixed size buffer on stack
    - copied string into buffer without checking length
    - encode virus into string!
  - dictionary attack on weak passwords
- Used infected machines to find/infect others



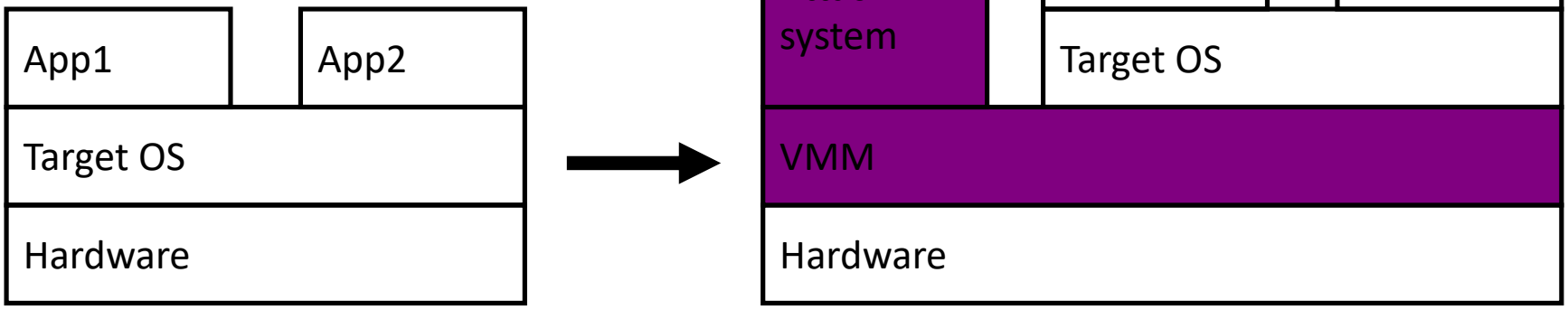
Rootkits

- 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
- Operation:
  - Intercept system calls for listing files, processes, etc.
  - Filter out malware's files and processes
  - Example: Magic prefix -- sysfilename
  - Diagram:

    Applications --> System Call ---> (Rootkit) --> Kernel

    <-- Results --- If call is from rootkit application (e.g. sysrootkit.exe), don't filter!
- 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

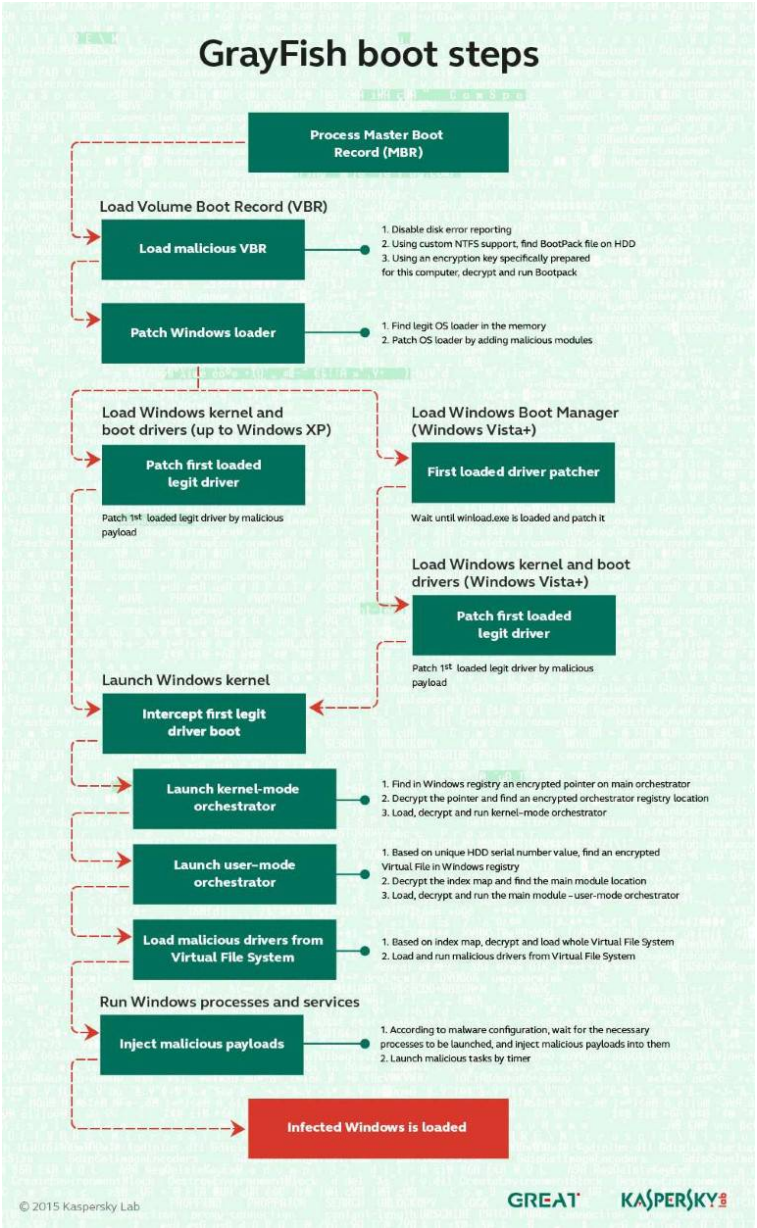
Virtual Machine Based Rootkits



Before infection

After infection

Equation Group Rootkits

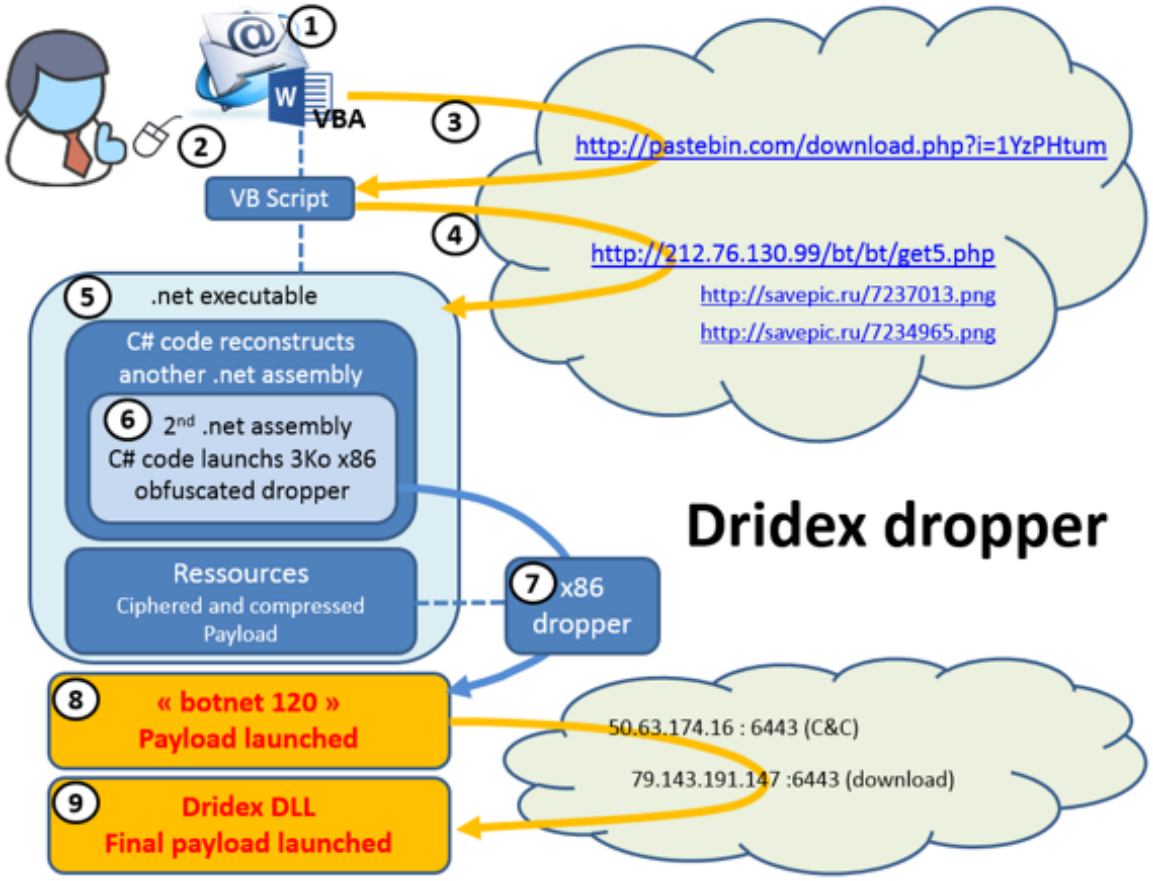


Ransomware


Cryptolocker 2.0
Your personal files are encrypted
Info
Your important files were encrypted on this computer: photos, videos, documents , etc. You can verify this by click on see files and try to open them.
Encryption was produced using unique public key RSA-4096 generated for this computer. To decrypt files, you need to obtain private key.
The single copy of the private key, which will allow you to decrypt the files, is located on a secret server on the Internet; the server will destroy the key within 72 hours after encryption completed. After that, nobody and never will be able to restore files.
To retrieve the private key, you need to pay 0.5 bitcoins.
Your files will be lost without payment on:

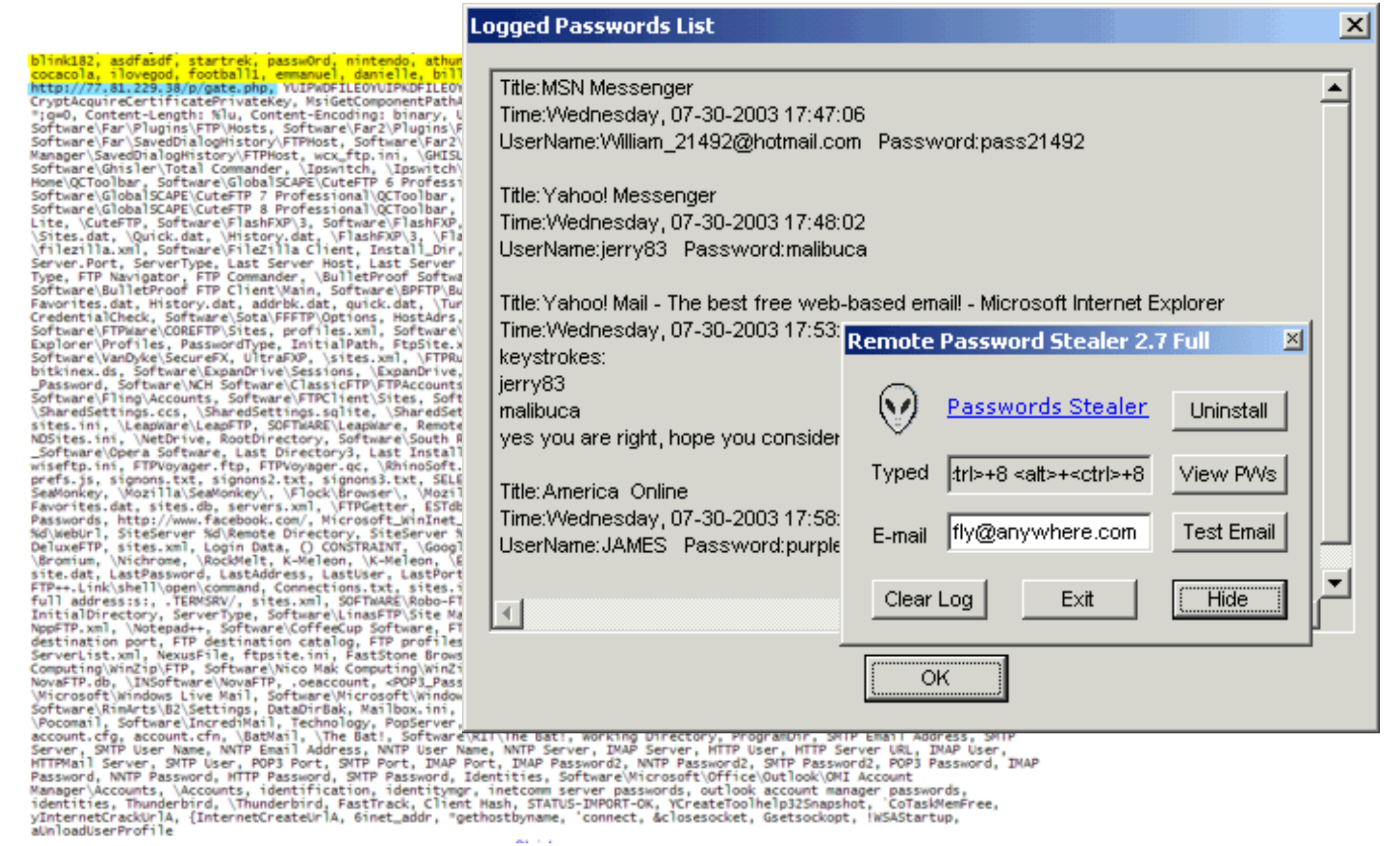
Droppers

Method of exploit and malware are often fully decoupled:



Facilitates black market economy, exploit-as-a-service.

Keylogging and Password Stealing



Botnets (Preview)

- Collection of compromised machines (bots) under (unified) control of an attacker (botmaster)
- Method of compromise decoupled from method of control
  - Launch a worm / virus / drive-by infection / etc.
- Upon infection, new bot "*phones home*" to rendezvous w/ botnet *command-and-control* (**C&C**)
- Lots of ways to architect C&C:
  - Star topology; hierarchical; peer-to-peer
  - Encrypted/stealthy communication
- Botmaster uses C&C to push out commands and updates

Example C&C Messages

1. Activation (report from bot to botmaster)
2. Email address harvests
3. Spamming instructions
4. Delivery reports
5. DDoS instructions
6. *FastFlux* instructions (rapidly changing DNS)
7. HTTP proxy instructions
8. Sniffed passwords report
9. IFRAME injection/report

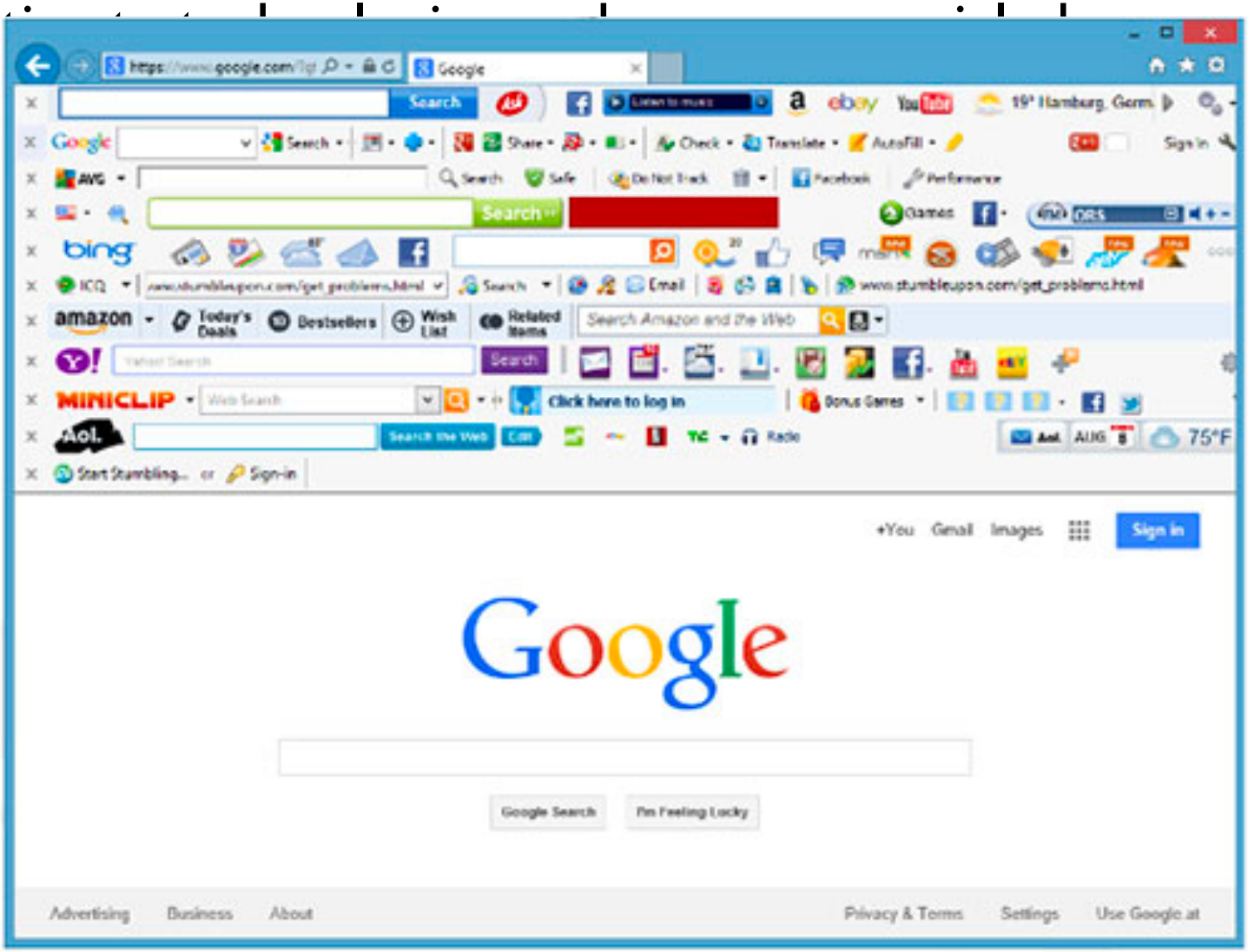
From the "Storm" botnet circa 2008

Potentially Unwanted Programs (PUPs)

- •Legitimate technologies such as commercial, shareware, freeware, or open source products may provide a value or benefit to a user
  - Notification
  - Consent
  - Control
- Distribution
- Installation
- Run-Time Behaviors
- Uninstall

Potentially Unwanted Programs (PUPs)

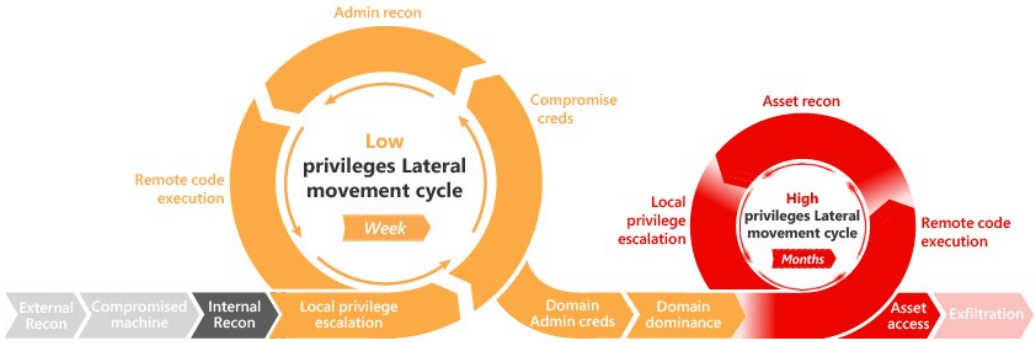
- •Legi…re,
  freev… e or
  bene…
  - Nc…
  - Co…
  - Co…
- Distr…
- Insta…
- Run-…
- Unin…



Adam's take

- Malware is… not always that interesting.
  - Scale is extraordinary, but attacker means are often technically uninteresting
  - Attacker aims are often pretty basic (i.e., $$$)
  - Black market economy analysis is cool, tho…
- *What about the sophisticated attackers?* What are they up to?

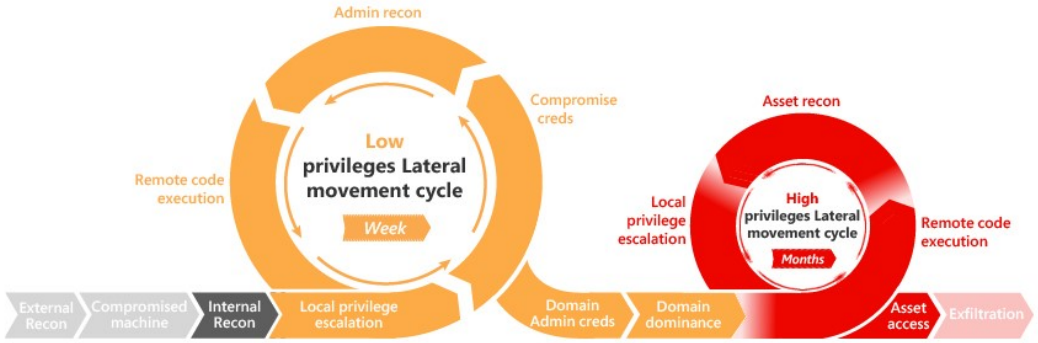
Advanced Persistent Threats



APT 'killchain'

- APT's leverage enhanced 'malware' toolkits in pursuit of larger strategic objectives
- Advanced — variety of intrusion methods, custom malware (many 0-days), attacks tailored to specific target
- Persistent — attacks target over extended period of time, stealthily takes down layers of defense
- Threat — not a matter of 'if,' but 'when'

Advanced Persistent Threats



APT 'killchain'

- Goals (e.g.)
  - Steal IP
  - Compromise security infrastructure
  - Damage physical assets
- Techniques (e.g.)
  - *everything*
- Source
  - Often state sponsored; resources effectively limitless.

MITRE ATT&CK Matrix

APT 'killchain' describes tactics; means are limitless:

Name	Description
Initial Access	The adversary is trying to get into your network.
Execution	The adversary is trying to run malicious code.
Persistence	The adversary is trying to maintain their foothold.
Privilege Escalation	The adversary is trying to gain higher-level permissions.
Defense Evasion	The adversary is trying to avoid being detected.
Credential Access	The adversary is trying to steal account names and passwords.
Discovery	The adversary is trying to figure out your environment.
Lateral Movement	The adversary is trying to move through your environment.
Collection	The adversary is trying to gather data of interest to their goal.
Command and Control	The adversary is trying to communicate with compromised systems to control them.
Exfiltration	The adversary is trying to steal data.
Impact	The adversary is trying to manipulate, interrupt, or destroy your systems and data.

. . .

https://attack.mitre.org/

MITRE ATT&CK Matrix

APT 'killchain' describes tactics; means are limitless:

Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Command and Control	Exfiltration	Impact
Drive-by Compromise	AppleScript	.bash_profile and .bashrc	Access Token Manipulation	Access Token Manipulation	Account Manipulation	Account Discovery	AppleScript	Audio Capture	Commonly Used Port	Automated Exfiltration	Data Destruction
Exploit Public-Facing Application	CMSTP	Accessibility Features	Accessibility Features	Binary Padding	Bash History	Application Window Discovery	Application Deployment Software	Automated Collection	Communication Through Removable Media	Data Compressed	Data Encrypted for Impact
External Remote Services	Command-Line Interface	Account Manipulation	AppCert DLLs	BITS Jobs	Brute Force	Browser Bookmark Discovery	Distributed Component Object Model	Clipboard Data	Connection Proxy	Data Encrypted	Defacement
Hardware Additions	Compiled HTML File	AppCert DLLs	AppInit DLLs	Bypass User Account Control	Credential Dumping	Domain Trust Discovery	Exploitation of Remote Services	Data from Information Repositories	Custom Command and Control Protocol	Data Transfer Size Limits	Disk Content Wipe
Replication Through Removable Media	Control Panel Items	AppInit DLLs	Application Shimming	Clear Command History	Credentials in Files	File and Directory Discovery	Logon Scripts	Data from Local System	Custom Cryptographic Protocol	Exfiltration Over Alternative Protocol	Disk Structure Wipe
Spearphishing Attachment	Dynamic Data Exchange	Application Shimming	Bypass User Account Control	CMSTP	Credentials in Registry	Network Service Scanning	Pass the Hash	Data from Network Shared Drive	Data Encoding	Exfiltration Over Command and Control Channel	Endpoint Denial of Service
Spearphishing Link	Execution through API	Authentication Package	DLL Search Order Hijacking	Code Signing	Exploitation for Credential Access	Network Share Discovery	Pass the Ticket	Data from Removable Media	Data Obfuscation	Exfiltration Over Other Network Medium	Firmware Corruption
Spearphishing via Service	Execution through Module Load	BITS Jobs	Dylib Hijacking	Compile After Delivery	Forced Authentication	Network Sniffing	Remote Desktop Protocol	Data Staged	Domain Fronting	Exfiltration Over Physical Medium	Inhibit System Recovery
Supply Chain Compromise	Exploitation for Client Execution	Bootkit	Exploitation for Privilege Escalation	Compiled HTML File	Hooking	Password Policy Discovery	Remote File Copy	Email Collection	Domain Generation Algorithms	Scheduled Transfer	Network Denial of Service
Trusted Relationship	Graphical User Interface	Browser Extensions	Extra Window Memory Injection	Component Firmware	Input Capture	Peripheral Device Discovery	Remote Services	Input Capture	Fallback Channels		Resource Hijacking
Valid Accounts	InstallUtil	Change Default File Association	File System Permissions Weakness	Component Object Model Hijacking	Input Prompt	Permission Groups Discovery	Replication Through Removable Media	Man in the Browser	Multi-hop Proxy		Runtime Data Manipulation
	Launchctl	Component Firmware	Hooking	Control Panel Items	Kerberoasting	Process Discovery	Shared Webroot	Screen Capture	Multi-Stage Channels		Service Stop
	Local Job Scheduling	Component Object Model Hijacking	Image File Execution Options Injection	DCShadow	Keychain	Query Registry	SSH Hijacking	Video Capture	Multiband Communication		Stored Data Manipulation
	LSASS Driver	Create Account	Launch Daemon	Deobfuscate/Decode Files or Information	LLMNR/NBT-NS Poisoning and Relay	Remote System Discovery	Taint Shared Content		Multilayer Encryption		Transmitted Data Manipulation
	Mshta	DLL Search Order Hijacking	New Service	Disabling Security Tools	Network Sniffing	Security Software Discovery	Third-party Software		Port Knocking		
	PowerShell	Dylib Hijacking	Path Interception	DLL Search Order Hijacking	Password Filter DLL	System Information Discovery	Windows Admin Shares		Remote Access Tools		
	Regsvcs/Regasm	External Remote Services	Plist Modification	DLL Side-Loading	Private Keys	System Network Configuration Discovery	Windows Remote Management		Remote File Copy		
	Regsvr32	File System Permissions Weakness	Port Monitors	Execution Guardrails	Securityd Memory	System Network Connections Discovery			Standard Application Layer Protocol		
	Rundll32	Hidden Files and Directories	Process Injection	Exploitation for Defense Evasion	Two-Factor Authentication Interception	System Owner/User Discovery			Standard Cryptographic Protocol		
	Scheduled Task	Hooking	Scheduled Task	Extra Window Memory Injection		System Service Discovery			Standard Non-Application Layer Protocol		
	Scripting	Hypervisor	Service Registry Permissions Weakness	File Deletion		System Time Discovery			Uncommonly Used Port		
	Service Execution	Image File Execution Options Injection	Setuid and Setgid	File Permissions Modification		Virtualization/Sandbox Evasion			Web Service		

. . .

https://attack.mitre.org/

Stuxnet

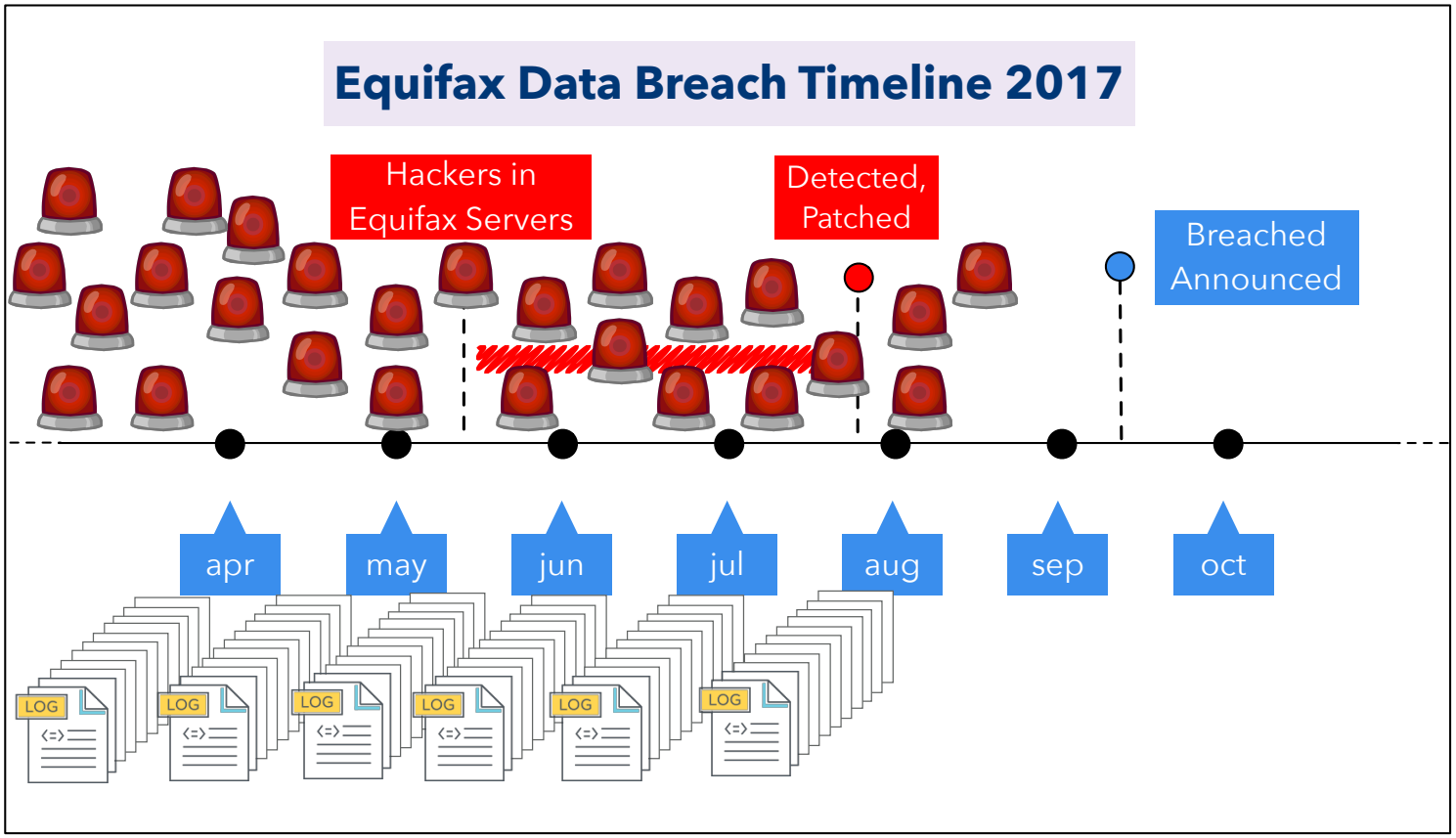
- Initially spread by infected USB flash drives
- Worm spreads quickly through Windows networks
- From Windows machine, attacked specific microcontrollers
- Incorporated 4 0-day exploits!
- Speculation: a US/Israel collaboration to target Iran
  - Reportedly ruined 1/5 of Iran's nuclear centrifuges
- Read more here: http://en.wikipedia.org/wiki/Stuxnet

Stuxnet

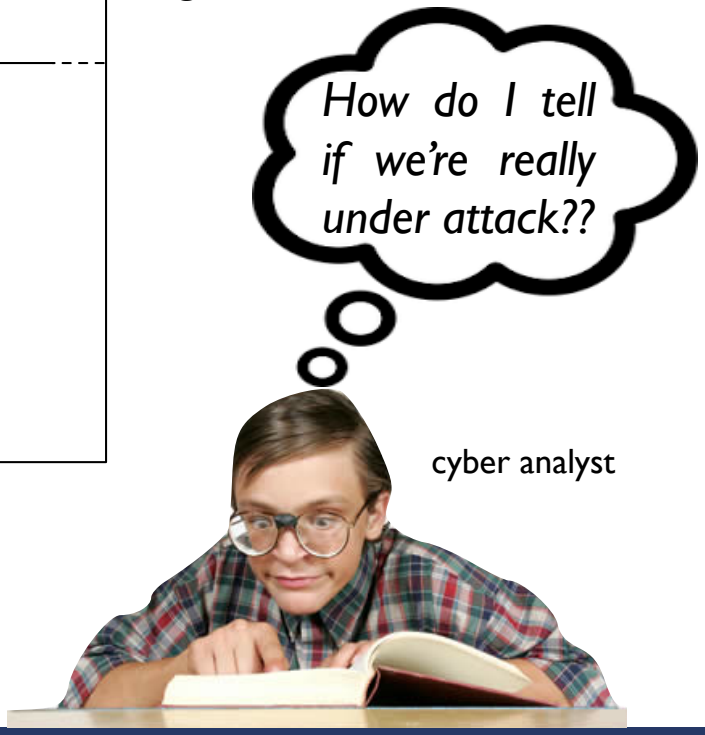
- Initially spread by infected USB flash drives
- Worm spreads quickly through Windows networks
- From Windows machine, attacked specific microcontrollers
- Incorporated 4 0-day exploits!
- Speculation: a US/Israel collaboration to target Iran
  - Reportedly ruined 1/5 of Iran's nuclear centrifuges
- Read more here: http://en.wikipedia.org/wiki/Stuxnet

Advanced Persistent Threats

Q: Why are we so bad at thwarting system intrusions?



- **Tens of thousands of alerts per week!**
- **60-80% are false alarms!**
- **10-40 minutes is required to investigate 1 alert!**
- **More than 70% of alerts are ignored!**



To Learn More…

- Books
  - Stallings and Brown, Chapter 6
  - Pfleeger and Pfleeger, Chapter 3
  - Goodrich and Tamassia, Chapter 4
  - Anderson, Chapter 21
  - Easttom, Chapter 5
- Papers
  - Manufacturing Compromise: The Emergence of Exploit-as-a-Service - Grier
  - Abusing File Processing in Malware Detectors for Fun and Profit - Jana