CIS 447/544: Computer and Network Security

Anys Bacha

Slides from U. Shankar, M. Hicks, K. Du, D. Boneh, N. Zeldovich, A. Rahmati, C. Hosmer

Malware

Malware

- Malicious code that is stored on and runs on a victim's system
- **How does it get to run?**
  - Attacks a user- or network-facing vulnerable service
    - E.g., using techniques you learned the past couple weeks
- Backdoor: Added by a malicious developer
- Social engineering: Trick the user into running/clicking/installing
- Trojan horse: Offer a good service, add in the bad
- An attacker with physical access downloads it & runs it

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

What Can Malware Do?

- Virtually anything, subject only to its permissions
- Brag: “APRIL 1st HA HA HA HA YOU HAVE A VIRUS!”
- Destroy:
  - Delete/mangle files
  - Damage hardware
- Crash the machine, e.g., by over-consuming resources
  - Fork bombing or “rabbits”: while(1) { fork(); }

What Can Malware Do?

- Steal information ("exfiltrate")
- Launch external attacks
  - Spam, click fraud, denial of service attacks
- Ransomware: e.g., by encrypting files
- Rootkits: Hide from user or software-based detection
  - Often by modifying the kernel
  - Man-in-the-middle attacks to sit between UI and reality

When Does it Run?

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

When Does it Run?

- Some attach themselves to other pieces of code
  - Viruses: run when the user initiates something
    - Run a program, open an attachment, boot the machine
    - **Require a host/program (parasitic)**
  - Worms: run while another program is running
    - No user intervention required
    - **No host needed and are stand-alone programs**

The line between these is thin and blurry

Some malware use both styles

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

Viruses

Viruses

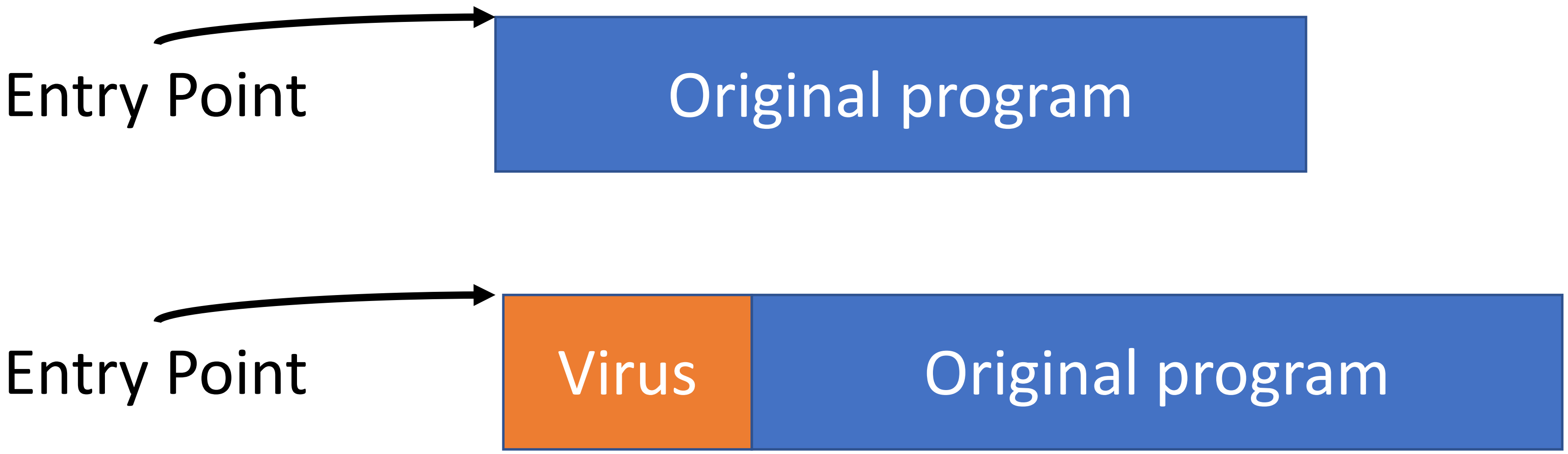
- They are opportunistic: they will eventually be run due to user action

- Two orthogonal aspects define a virus:
  1. How does it propagate?
  2. What else does it do (what is the “payload”)?

- General infection strategy:
  - Alter some existing code to include the virus
  - Share it, and expect users to (unwittingly) re-share

- Viruses have been around since at least the 70s

How Viruses Infect Other Programs



Goal is to take over the entry point

Viruses are classified by what they infect

- Document viruses
  - Implemented within a formatted document
  - Word documents (very rich macros)
  - PDF (Acrobat permits javascript)
  - (This is why you shouldn’t open random attachments)

- Boot sector viruses
  - Boot sector: small disk partition at a fixed location
  - If the disk is used to boot, then the firmware loads the boot sector code into memory and runs it

Viruses are classified by what they infect

- Boot sector viruses (Cont.)
  - What’s supposed to happen: this code loads the OS
  - Similar: AutoRun on music/video disks
  - (Why you shouldn’t plug random USB drives into your computer)

- Memory-resident viruses:
  - “Resident code” stays in memory because it is used so often

A Technological Arms Race

The key is *evasion*

Mechanisms for evasive **propagation**

VS

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

How Viruses Propagate

- First, the virus looks for an **opportunity to run**.
  - Increase chances by attaching malicious code to something a user is likely to run
  - autorun.exe on storage devices
  - Email attachments

- When a virus runs, it looks for an **opportunity to infect** other systems.
  - User plugs in a USB thumb drive: try to overwrite autorun.exe
  - User is sending an email: alter the attachment
  - Viruses can also proactively create emails ("I Love You")

Love Bug Example

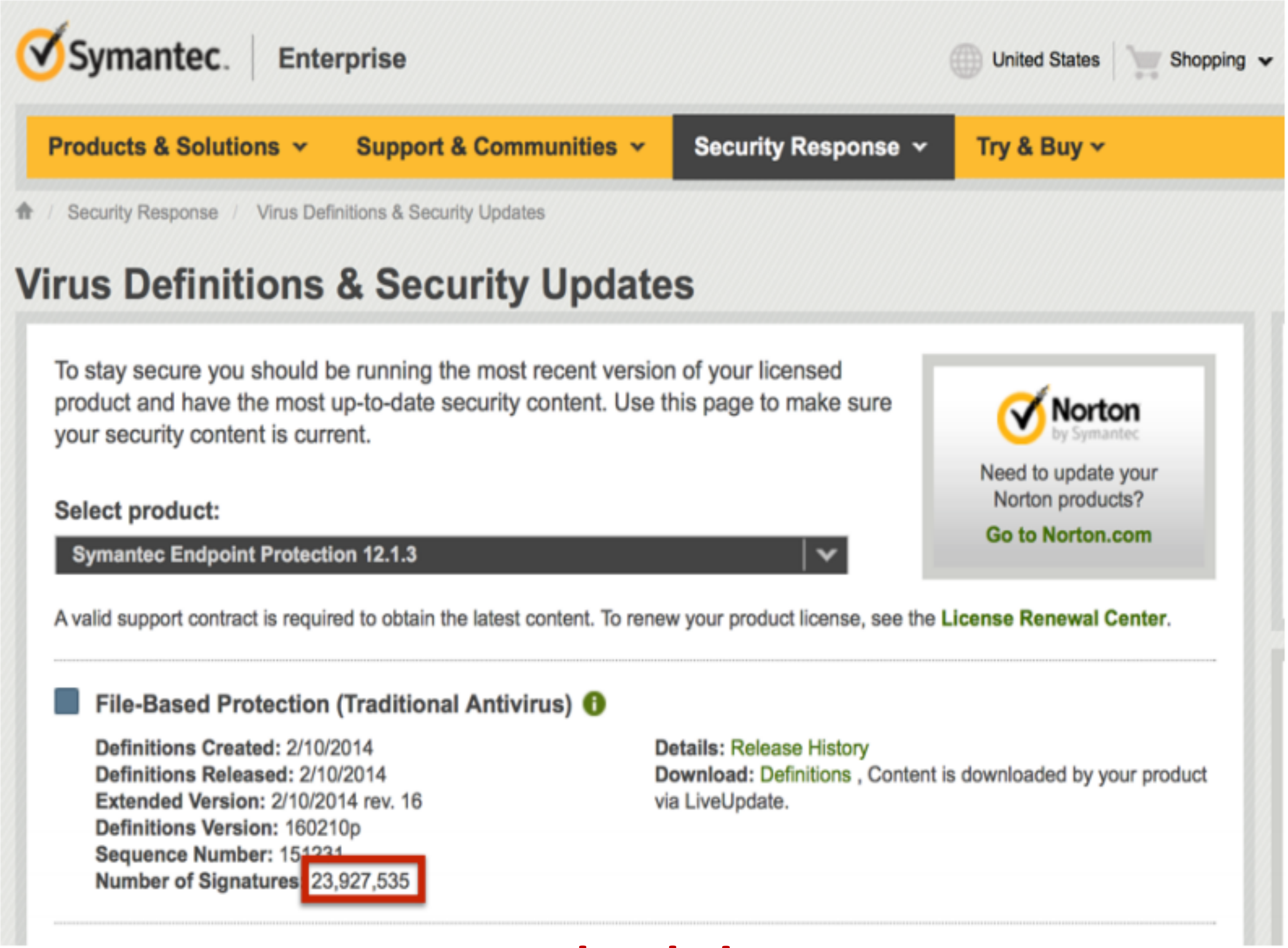
- ***I Love You*** virus is also referred to as the ***Love Bug***
- Attacked millions of Windows PCs
  - Sent as an email with a malicious attachment
- Had a subject line “ILOVEYOU” but included a malicious attachment “LETTER-FOR-YOU.txt.vbs”
  - The latter extension was hidden by default on Windows
- The attachment once executed would infect Office files, images, audio files, and sent a copy of itself to all addresses in the Windows Address Book used by Microsoft Outlook

Detecting Viruses

- **Method 1: Signature-based detection**
    - Look for bytes corresponding to injected virus code
    - Protect other systems by installing a recognizer for a known virus
    - In practice, requires fast scanning algorithms

- This basic approach has driven the multi-billion dollar antivirus market

- #Recognized signatures is a means of marketing and competition
    - But what does that say about how important they are?


Symantec. Enterprise
United States
Shopping
Products & Solutions
Support & Communities
Security Response
Try & Buy
Security Response / Virus Definitions & Security Updates
Virus Definitions & Security Updates
To stay secure you should be running the most recent version of your licensed product and have the most up-to-date security content. Use this page to make sure your security content is current.
Norton by Symantec
Need to update your Norton products?
Go to Norton.com
Select product:
Symantec Endpoint Protection 12.1.3
A valid support contract is required to obtain the latest content. To renew your product license, see the License Renewal Center.
File-Based Protection (Traditional Antivirus)
Definitions Created: 2/10/2014
Definitions Released: 2/10/2014
Extended Version: 2/10/2014 rev. 16
Definitions Version: 160210p
Sequence Number: 151231
Number of Signatures 23,927,535
Details: Release History
Download: Definitions , Content is downloaded by your product via LiveUpdate.

Um…thanks!

FEATURE

Antivirus vendors go beyond signature-based antivirus

Robert Westervelt, News Director

This article can also be found in the Premium Editorial Download **"Information Security magazine: Successful cloud migrations require careful planning."**

Download it now to read this article plus other related content.

Security experts and executives at security vendors are in agreement that signature-based antivirus isn't able to keep up with the explosion of malware. For example, in 2009, Symantec says it wrote about 15,000 antivirus signatures a day; that number has increased to 25,000 antivirus signatures every day.

"Signatures have been dying for quite a while," says Mikko H. Hypponen, chief research officer of Finnish-based antivirus vendor, F-Secure. "The sheer number of malware samples we see every day completely overwhelms our ability to keep up with them."

Security vendors have responded by updating their products with additional capabilities, such as file reputation and heuristics-based engines. They're also making upgrades to keep up with the latest technology trends, such as virtualization and cloud computing.

You Are a Virus Writer

You Are a Virus Writer

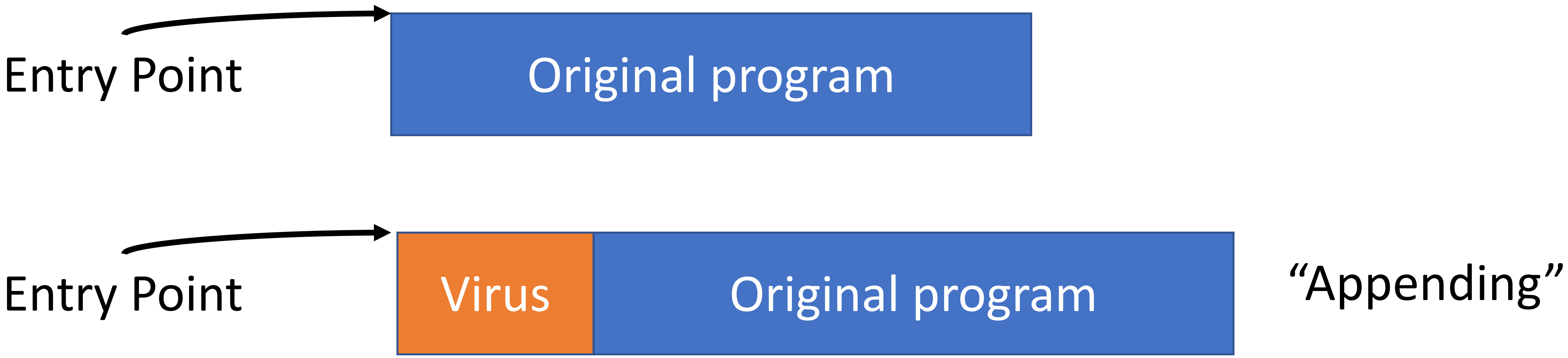
- Your goal is for your virus to spread far and wide
- How do you avoid detection by antivirus software?

You Are a Virus Writer

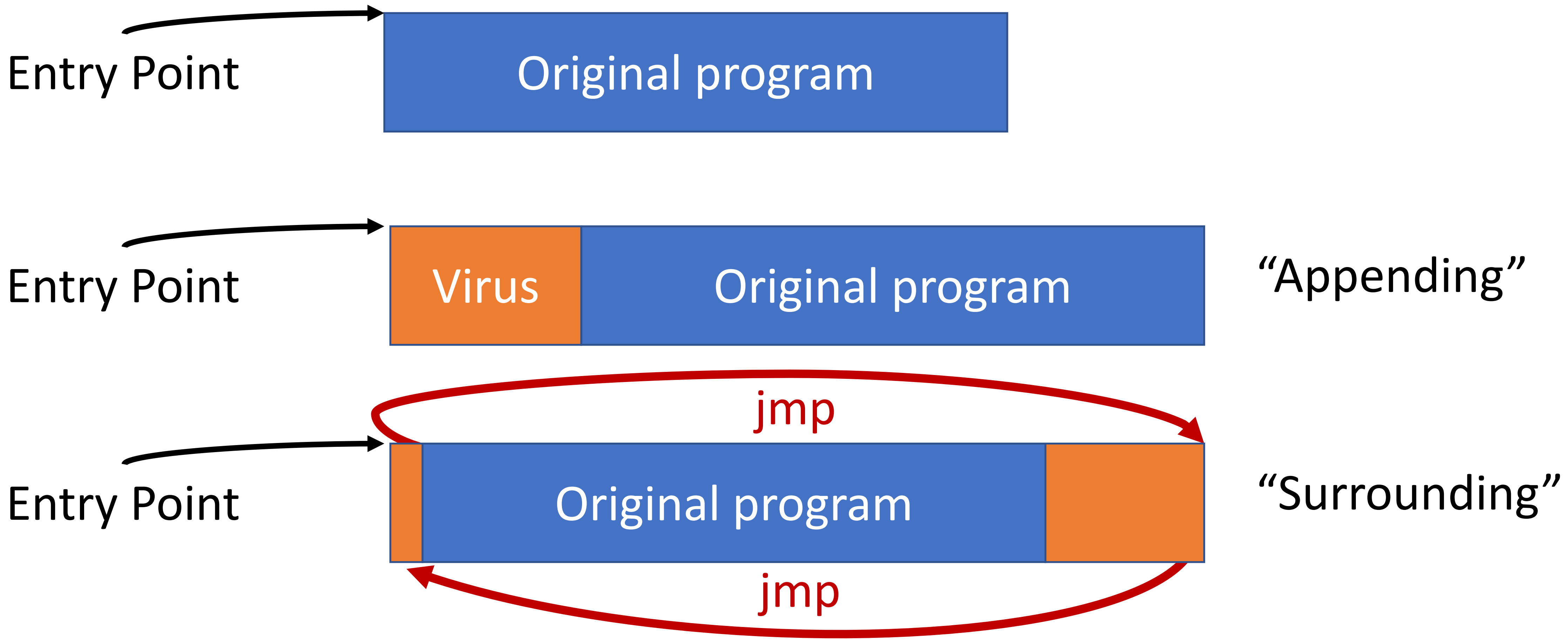
- Your goal is for your virus to spread far and wide

- How do you avoid detection by antivirus software?
  1. **Give them a harder signature to find**

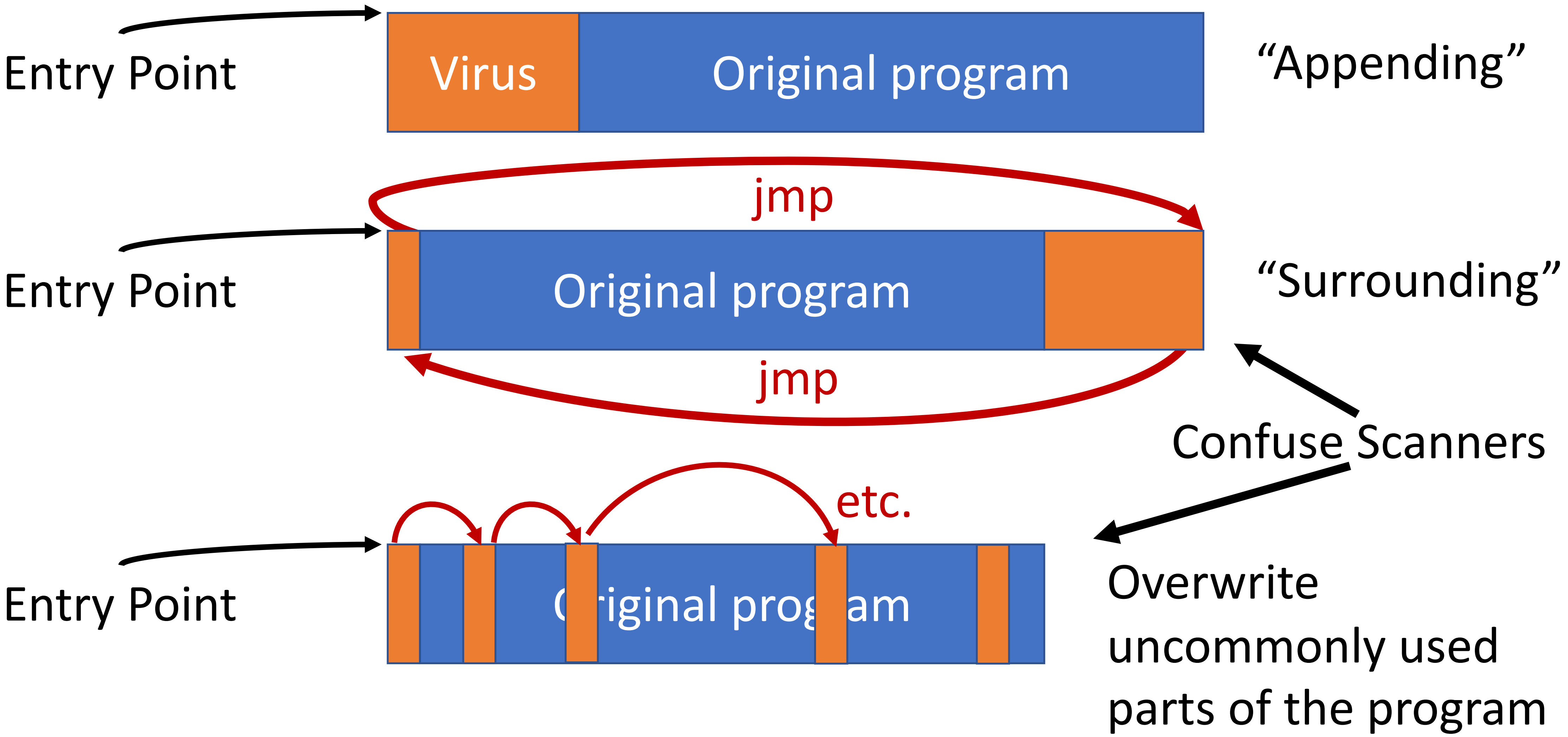
How Viruses Infect Other Programs



How Viruses Infect Other Programs



How Viruses Infect Other Programs



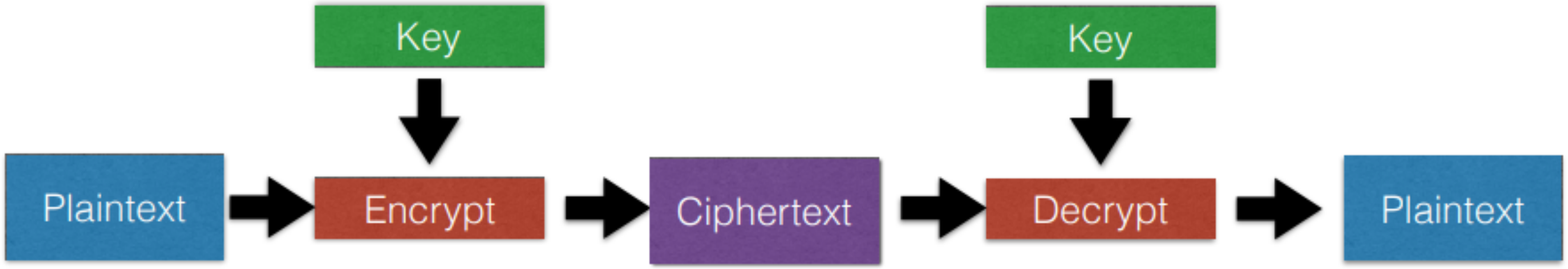
You Are a Virus Writer

- Your goal is for your virus to spread far and wide

- How do you avoid detection by antivirus software?
  1. Give them a harder signature to find
  2. ***Change*** your code so they can't pin down a signature

Goal: every time you inject your code, it should look different

Building Block: Encryption



<u>Symmetric key</u>: both keys are the same
<u>Asymmetric key</u>: different keys

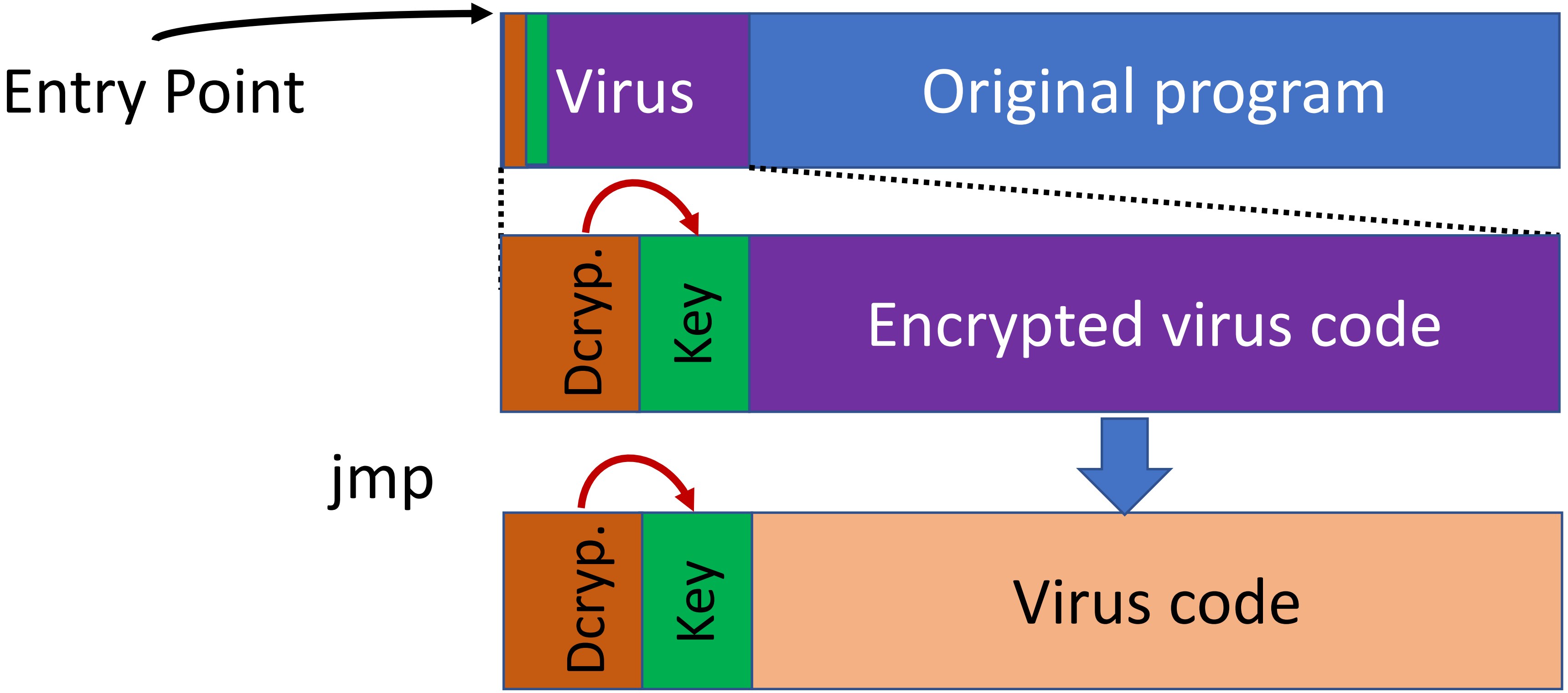
Important property: the ciphertext is ***nondeterministic***
i.e., "Encrypt" has a different output each time

but decrypting always returns the plaintext

Polymorphic Virus

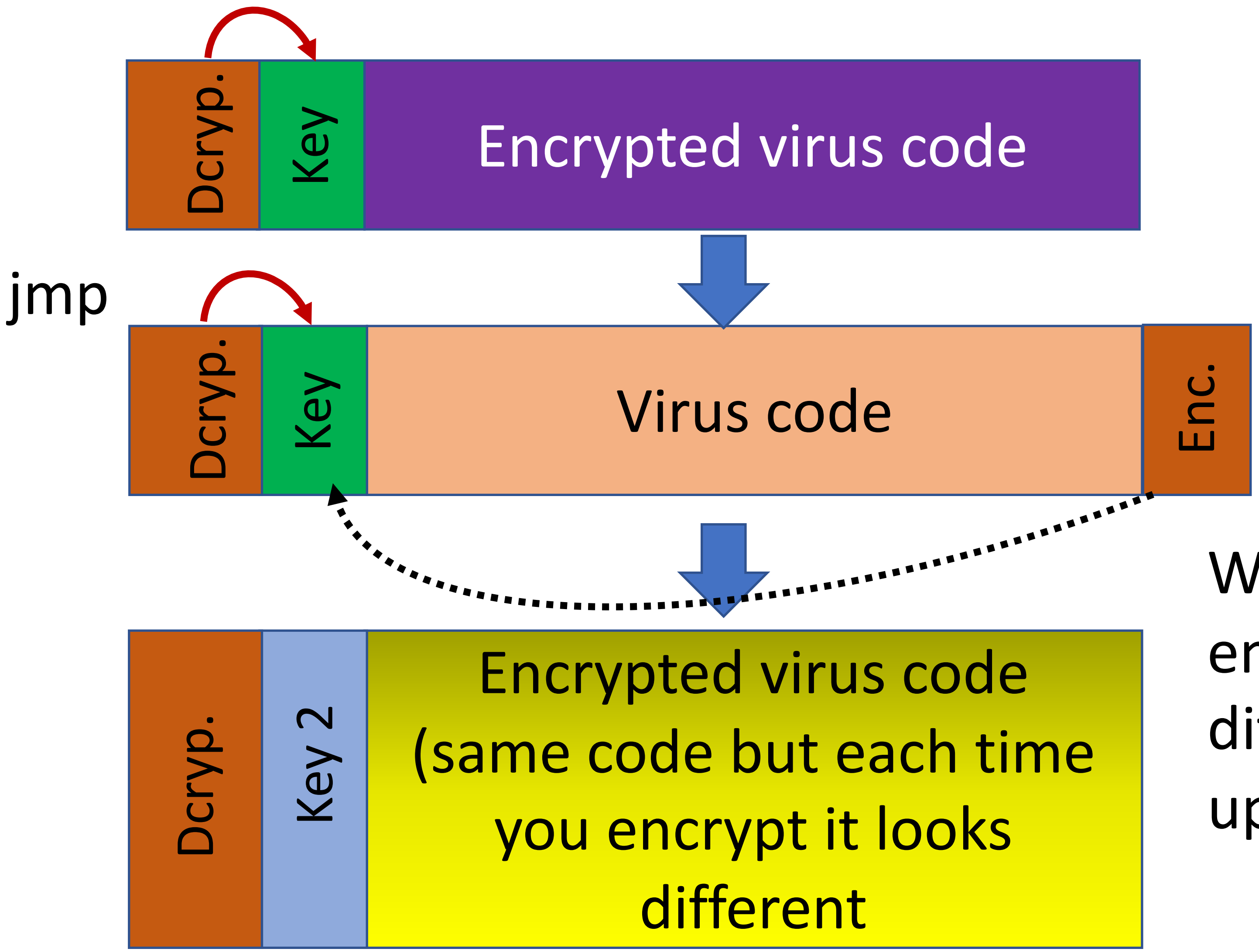
- Polymorphic virus implies “change of appearance”
- Usually includes mutation engines bundled with the virus
  - Common methods include:
    - Encryption (multiple encryption algorithms for further obfuscation)
    - Data appending/data pre-pending

Polymorphic Viruses



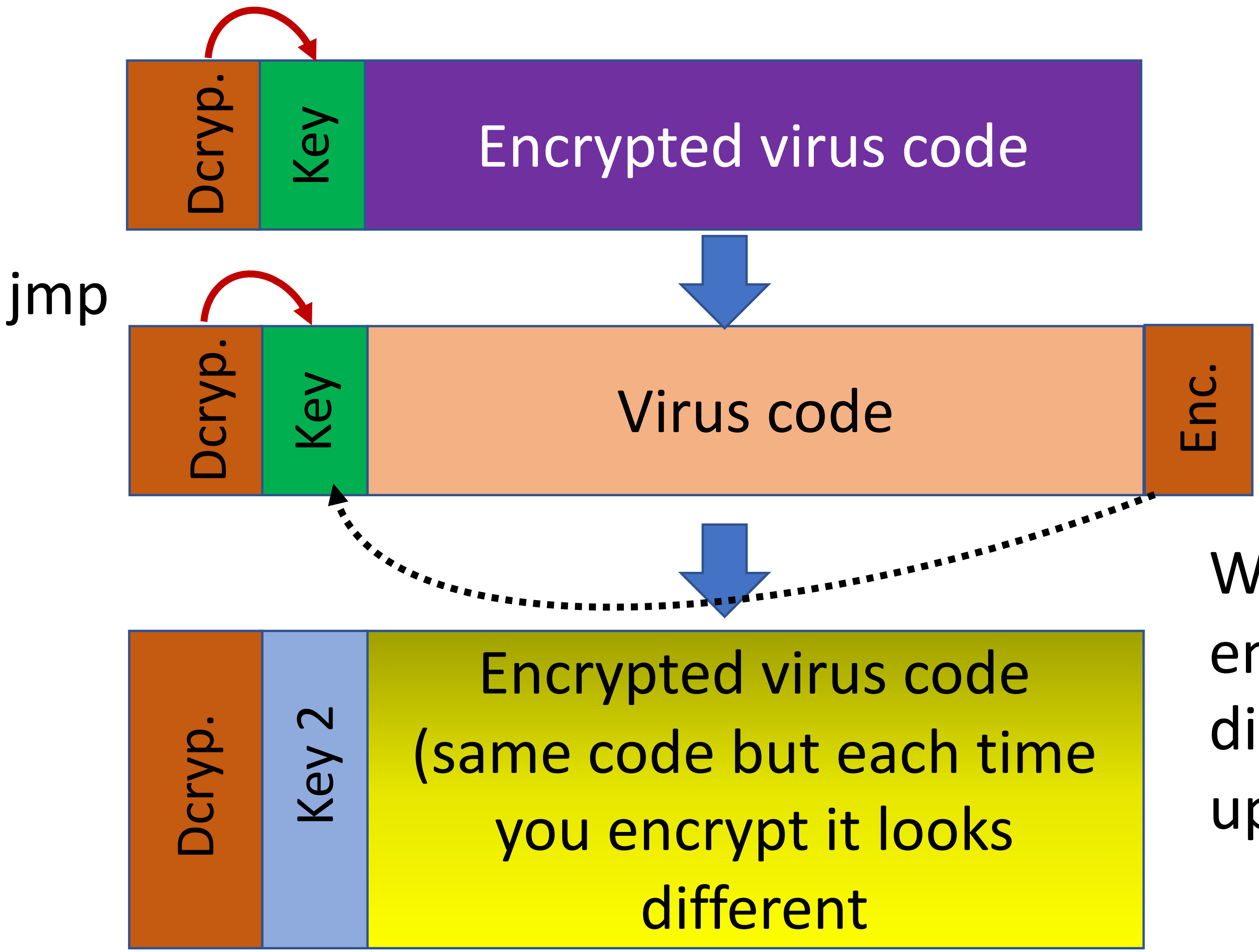
Take over the entry point

Polymorphic Viruses



When used properly, encryption will yield a different, random output upon each invocation

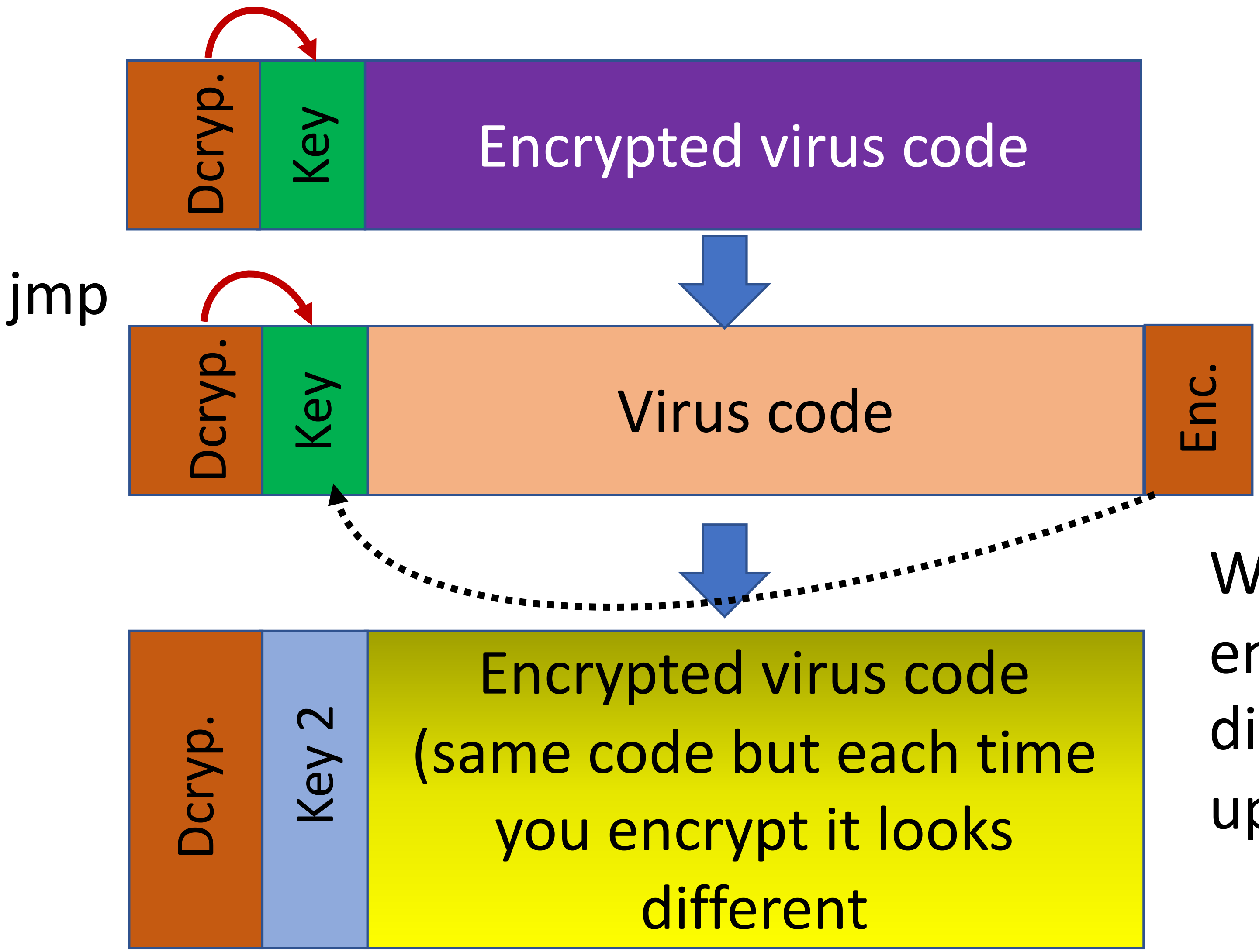
Polymorphic Viruses



Where would you look for a signature?

When used properly, encryption will yield a different, random output upon each invocation

Polymorphic Viruses



Where would you look for a signature?

Start with the decryptor!

When used properly, encryption will yield a different, random output upon each invocation

Polymorphic Viruses: Arms Race

Now you are the antivirus writer: how do you detect?

Polymorphic Viruses: Arms Race

- Idea #1: *Narrow signature* to catch the decrypter
  - Often very small: can result in many false positives
  - Attacker can spread this small code around and jmp

- Idea #2: *Execute* or statically analyze the suspect code to see if it decrypts.
  - How do you distinguish from common “packers” which do something similar (decompression)?
  - How long do you execute the code??

Polymorphic Viruses: Arms Race

- Idea #1: *Narrow signature* to catch the decrypter
  - Often very small: can result in many false positives
  - Attacker can spread this small code around and jmp

- Idea #2: *Execute* or statically analyze the suspect code to see if it decrypts.
  - How do you distinguish from common “packers” which do something similar (decompression)?
  - How long do you execute the code??

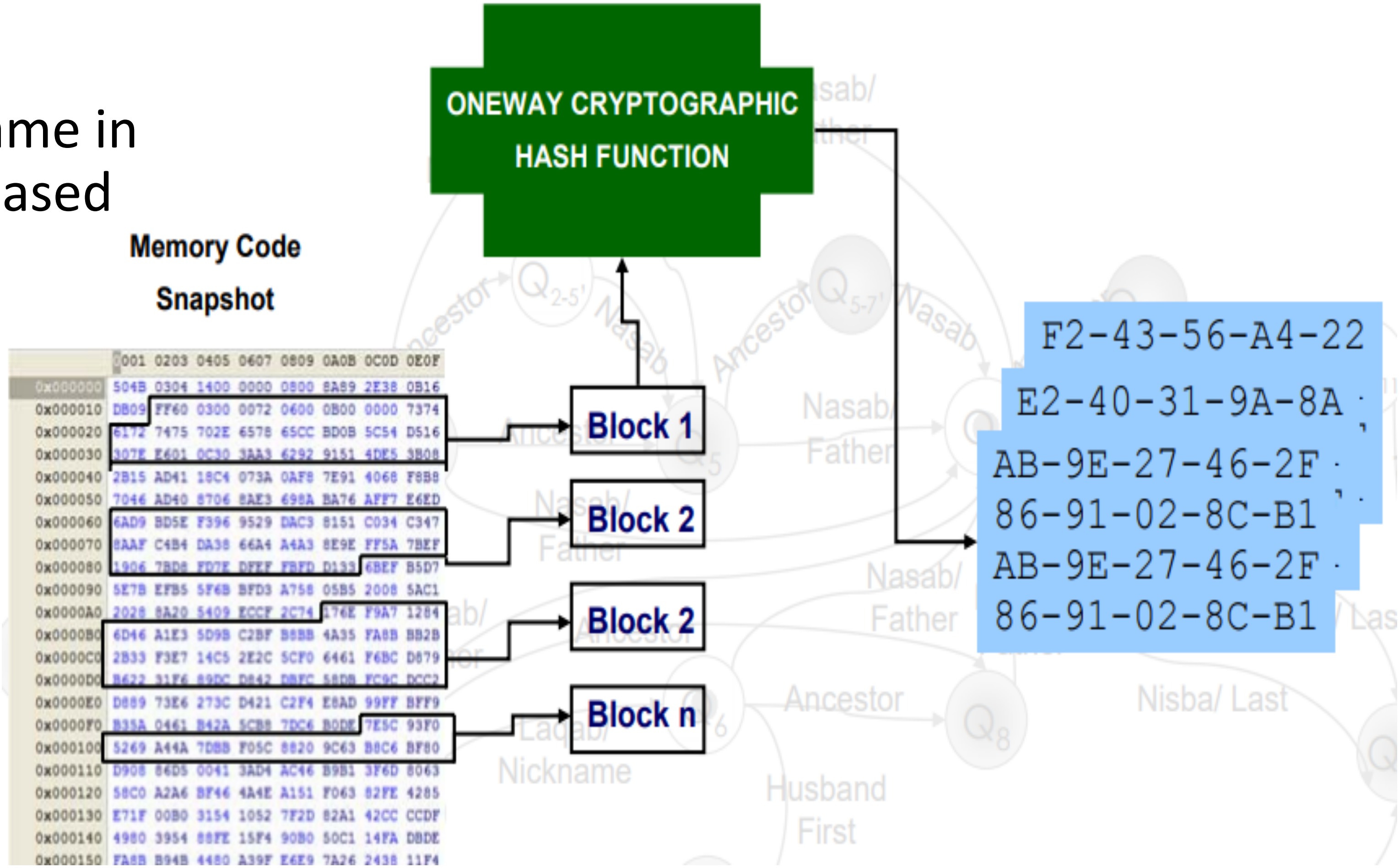
Now you are the virus writer again: how do you evade?

Polymorphic Countermeasures

- Change the decrypter
  - Oligomorphic viruses: change to one of a fixed set of decrypters
  - True polymorphic viruses: can generate an endless number of decrypters
    - e.g., brute force key break
    - Downside: inefficient

Limitations of Polymorphic Virus

- Limitations:
  - The decrypted code is the same in each case making memory based signature detection possible



https://www.blackhat.com/presentations/bh-usa-08/Hosmer/BH_US_08_Hosmer_Polymorphic_Malware.pdf

Metamorphic Code

- Every time the virus propagates, generate a *semantically different* version of the code
  - Higher-level semantics remain the same
  - But the way it does it differs
    - Different machine code instructions
    - Different algorithms to achieve the same thing
    - Different use of registers
    - Swap registers
    - Reorder independent instructions
    - Insert random NOPs
    - Different constants…

Metamorphic Code

- Every time the virus propagates, generate a *semantically different* version of the code
  - Higher-level semantics remain the same
  - But the way it does it differs
    - Different machine code instructions
    - …
- How would you do this?
  - Include a code rewriter with your virus
  - Add a bunch of complex code to throw others off (then just never run it)

```
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

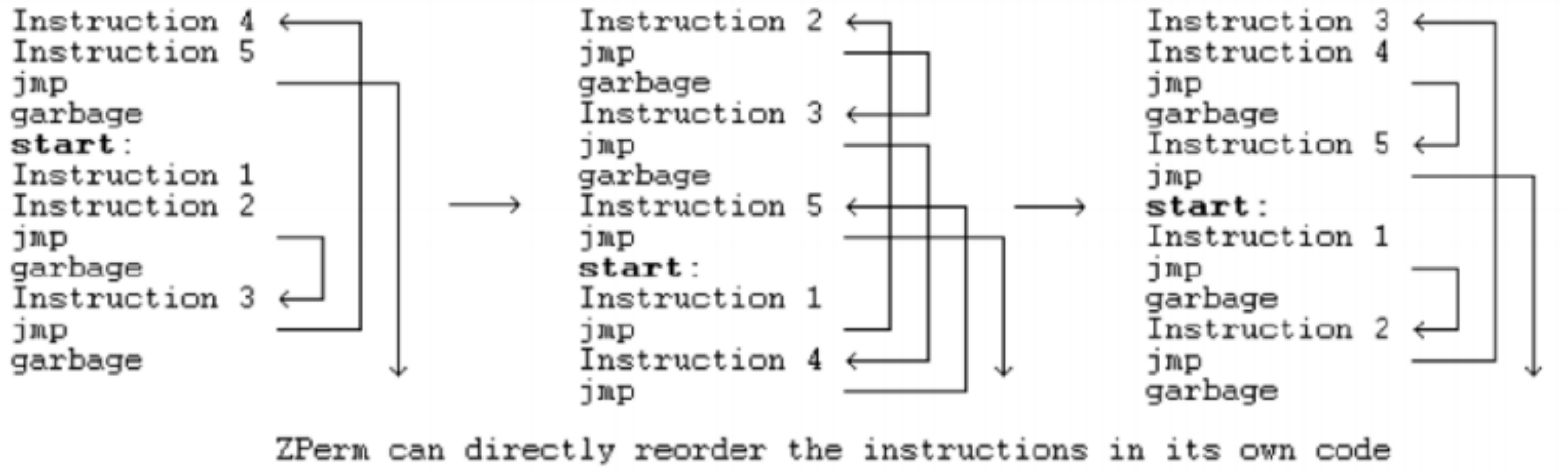


Figure 7 Zperm.A inserts JMP instruction into its code

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") "constant" 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

Detecting Metamorphic Viruses

Detecting Metamorphic Viruses

- Scanning isn't enough: need to analyze execution behavior

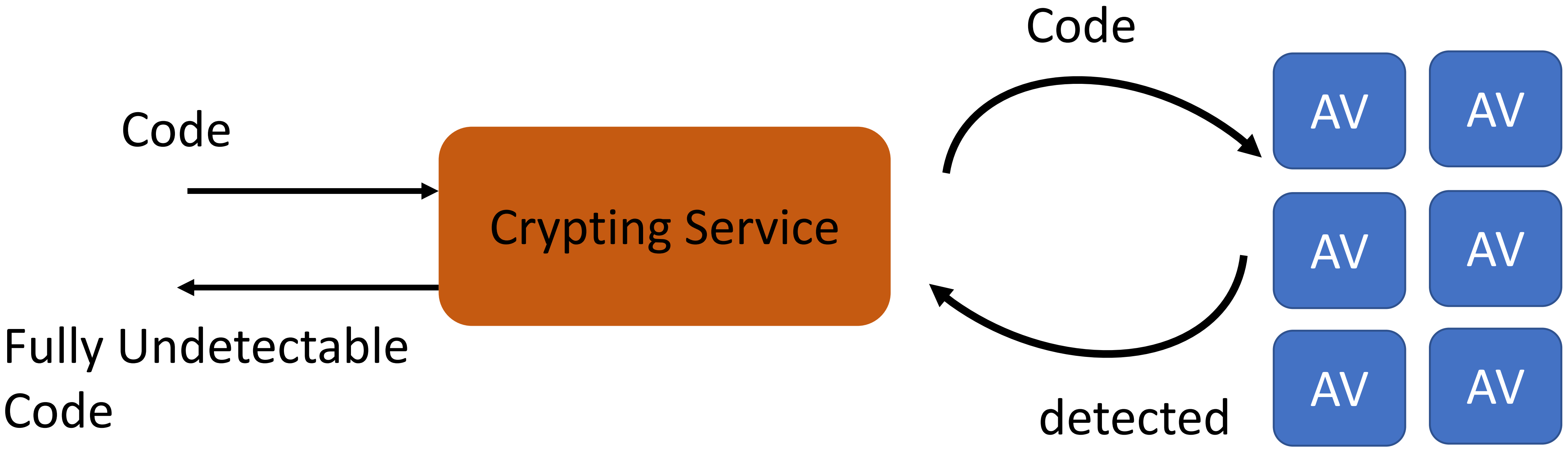
- Two broad stages in practice (both take place in a safe environment, like gdb or a virtual machine)
  1. AV company analyzes new virus to find behavioral signature
  2. AV system at the end host analyzes suspect code to see if it matches the signature

Detecting Metamorphic Viruses

- Countermeasures
  - Have your virus change slowly (hard to create a proper behavioral signature)
  - Detect if you are in a safe execution environment (e.g., gdb) and act differently
- Counter-countermeasures
  - Detect detection and skip those parts
- Counter-counter-counter…. Arms race

Attackers have the upper hand: AV systems hand out signature oracles

Crypting Services

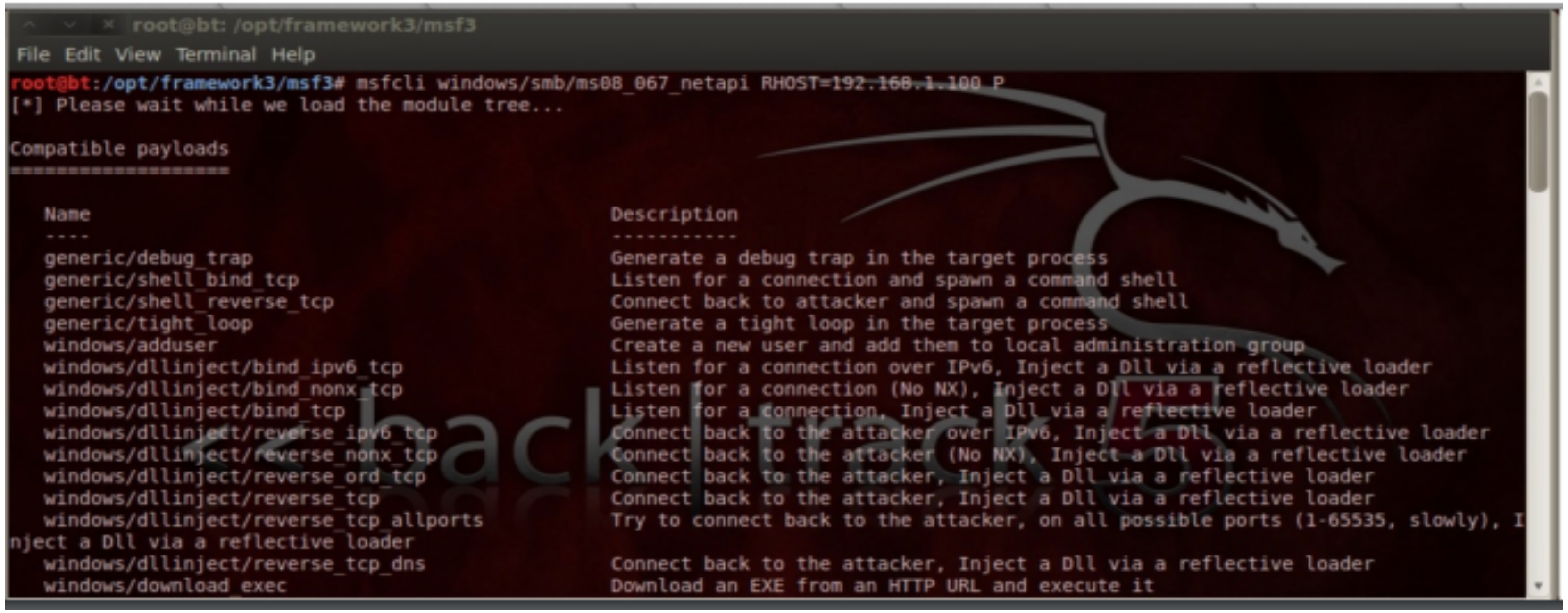


Iteratively obfuscate the code (encrypt + jmp + …)

Until the obfuscated code is “fully undetectable”

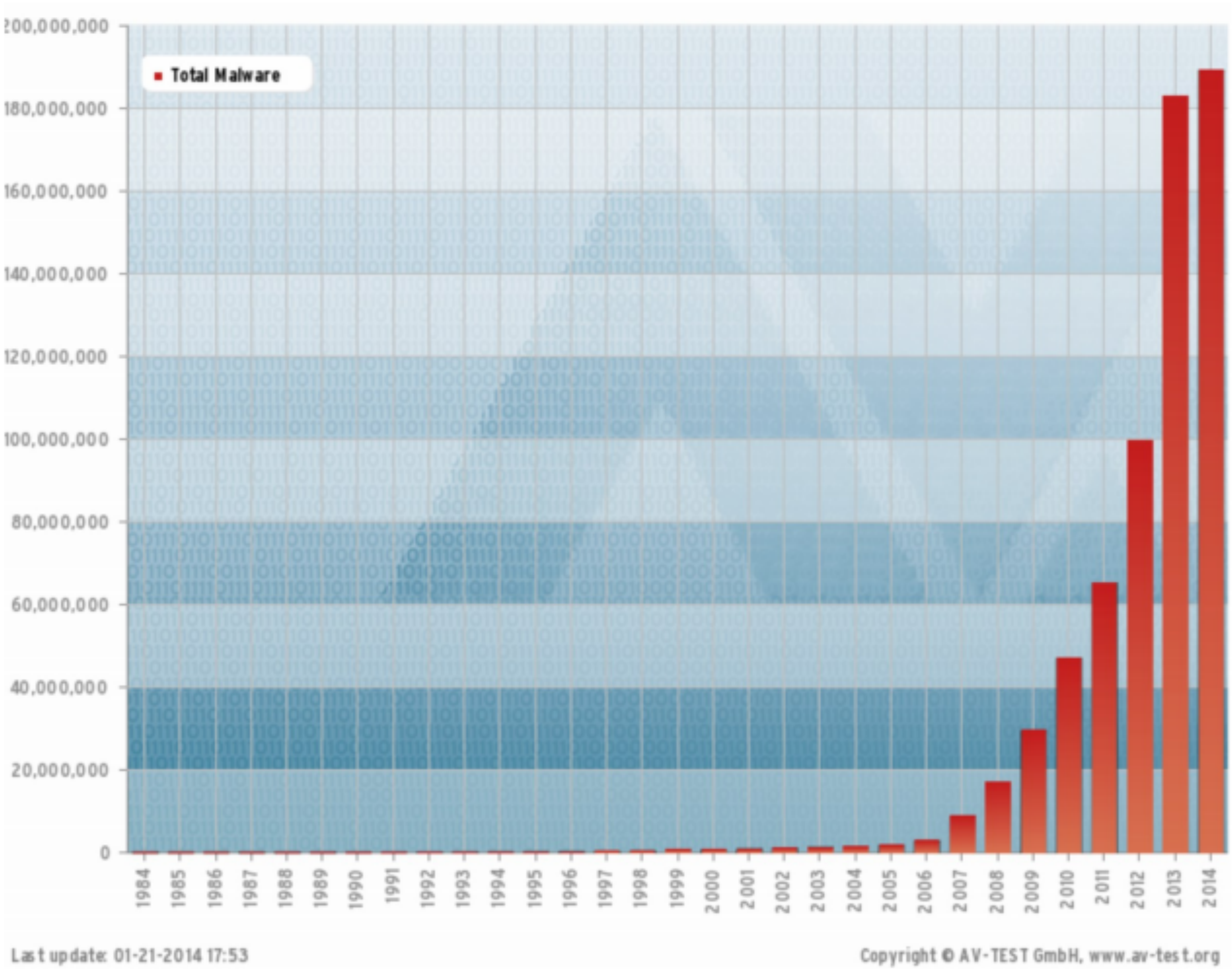
Putting it All Together Sounds Hard

- Creating a virus can be really difficult
- Historically error prone
- But using them is easy: any scriptkiddy can use Metasploit
- Good news: so can any white hat pen tester



So How Much Malware is Out There?

- Polymorphic and metamorphic viruses can make it easy to miscount viruses
- Take numbers with a grain of salt
- Large numbers are in the AV vendors’ best interest
- Previously, most malware was showy
- Now primary goal is frequently to not get noticed



Total Malware
200,000,000
180,000,000
160,000,000
140,000,000
120,000,000
100,000,000
80,000,000
60,000,000
40,000,000
20,000,000
0
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
Last update: 01-21-2014 17:53
Copyright © AV-TEST GmbH, www.av-test.org

Virus Case Studies

Brain

First IBM PC virus (1987)

Propagation

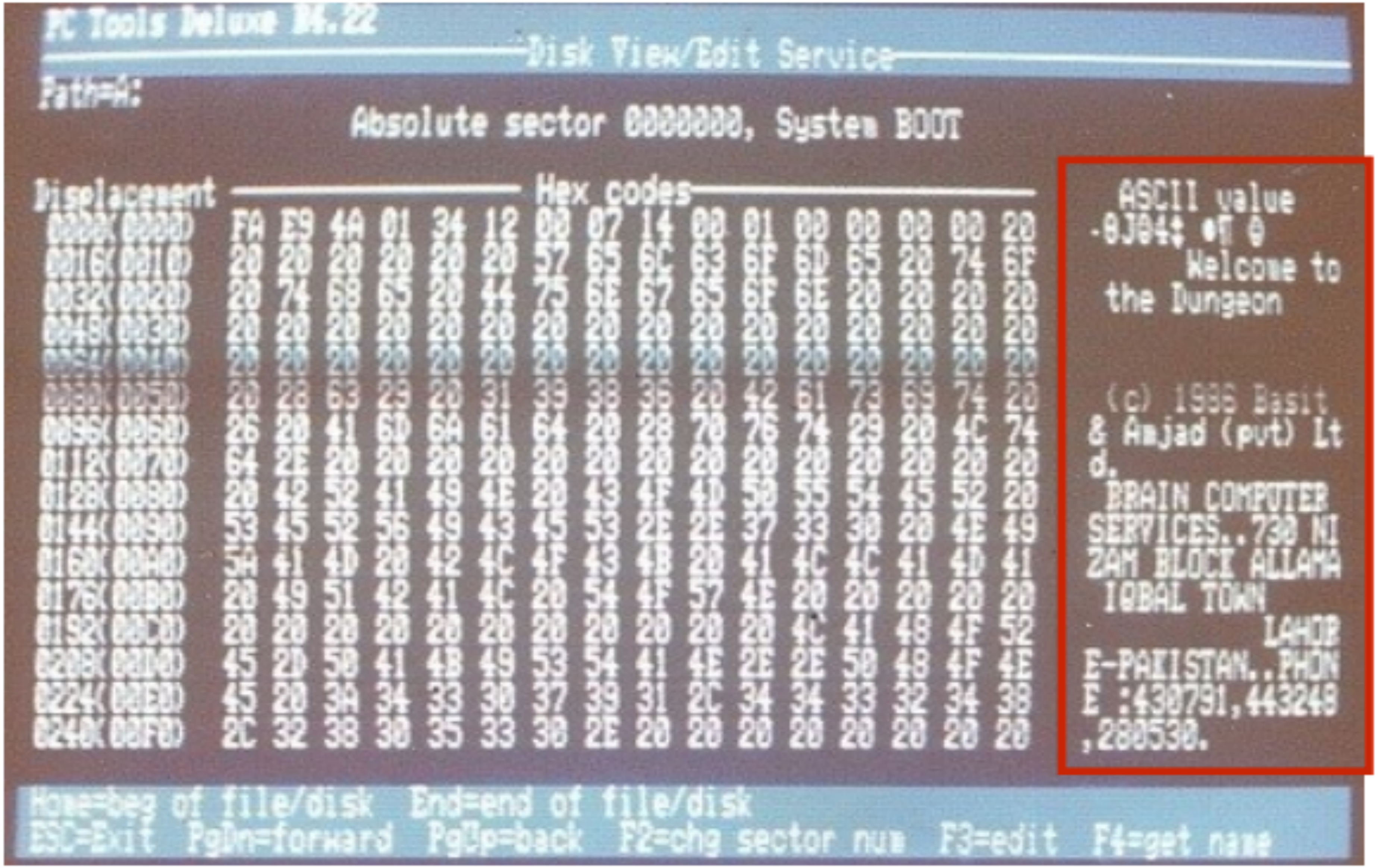
- Copies itself into the boot sector
- Tells the OS that all of the boot sector is “faulty” (so that it won’t list contents to the user)
  - Thus also one of the first examples of a stealth virus
- Intercepts disk read requests for 5.25” floppy drives
  - Sees if the 5th and 6th bytes of the boot sector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

Brain

Payload

- Nothing really; goal was just to spread (to show off?)

- However, it served as the template for future viruses


PC Tools Deluxe R4.22
Disk View/Edit Service
Path=A:
Absolute sector 0000000, System BOOT
Displacement
Hex codes
ASCII value
0000(0000) FA E9 4A 01 34 12 00 07 14 00 01 00 00 00 00 20
0016(0010) 20 20 20 20 20 20 57 65 6C 63 6F 6D 65 20 74 6F
0032(0020) 20 74 68 65 20 44 75 6E 67 65 6F 6E 20 20 20 20
0048(0030) 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0064(0040) 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0080(0050) 20 28 63 29 20 31 39 38 36 20 42 61 73 69 74 20
0096(0060) 26 20 41 6D 6A 61 64 20 28 70 76 74 29 20 4C 74
0112(0070) 64 2E 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0128(0080) 20 42 52 41 49 4E 20 43 4F 4D 50 55 54 45 52 20
0144(0090) 53 45 52 56 49 43 45 53 2E 2E 37 33 30 20 4E 49
0160(00A0) 5A 41 4D 20 42 4C 4F 43 4B 20 41 4C 4C 41 4D 41
0176(00B0) 20 49 51 42 41 4C 20 54 4F 57 4E 20 20 20 20 20
0192(00C0) 20 20 20 20 20 20 20 20 20 20 20 4C 41 48 4F 52
0208(00D0) 45 2D 50 41 4B 49 53 54 41 4E 2E 2E 50 48 4F 4E
0224(00E0) 45 20 3A 34 33 30 37 39 31 2C 34 34 33 32 34 38
0240(00F0) 2C 32 38 30 35 33 30 2E 20 20 20 20 20 20 20 20
Welcome to
the Dungeon
(c) 1986 Basit
& Amjad (pvt) Lt
d.
BRAIN COMPUTER
SERVICES..730 NI
ZAM BLOCK ALLAMA
IQBAL TOWN
LAHOR
E-PAKISTAN..PHON
E :430791,443248
,280530.
Home=beg of file/disk End=end of file/disk
ESC=Exit PgDn=forward PgUp=back F2=chg sector num F3=edit F4=get name

Rootkits

- Recall: a rootkit is malicious code that takes steps to go undiscovered
  - 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… **installed by a CD he had bought.**

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)

Sony XCP rootkit

Detected (2005)

- How it messed up
  - Morally: violated trust
  - Technically: Hid all files that started with “sys”
  - Seriously?: Uninstaller did not actually uninstall;
    - It introduced additional vulnerability instead

Worms

Controlling millions of hosts: How?

- Worm: self-propagates by arranging to have itself immediately executed
  - At which point it creates a new, additional instance of itself
- Typically infects by altering running code
  - No user intervention required
- Like viruses, propagation and payload are orthogonal

Self-propagation

- Goal: spread as quickly as possible
- The key is parallelization
  - Without being triggered by human interaction!

Propagation

1. Targeting: how does the worm find **new prospective victims**?
2. Exploit: how does the worm get code to automatically run?

Morris Worm - 1988

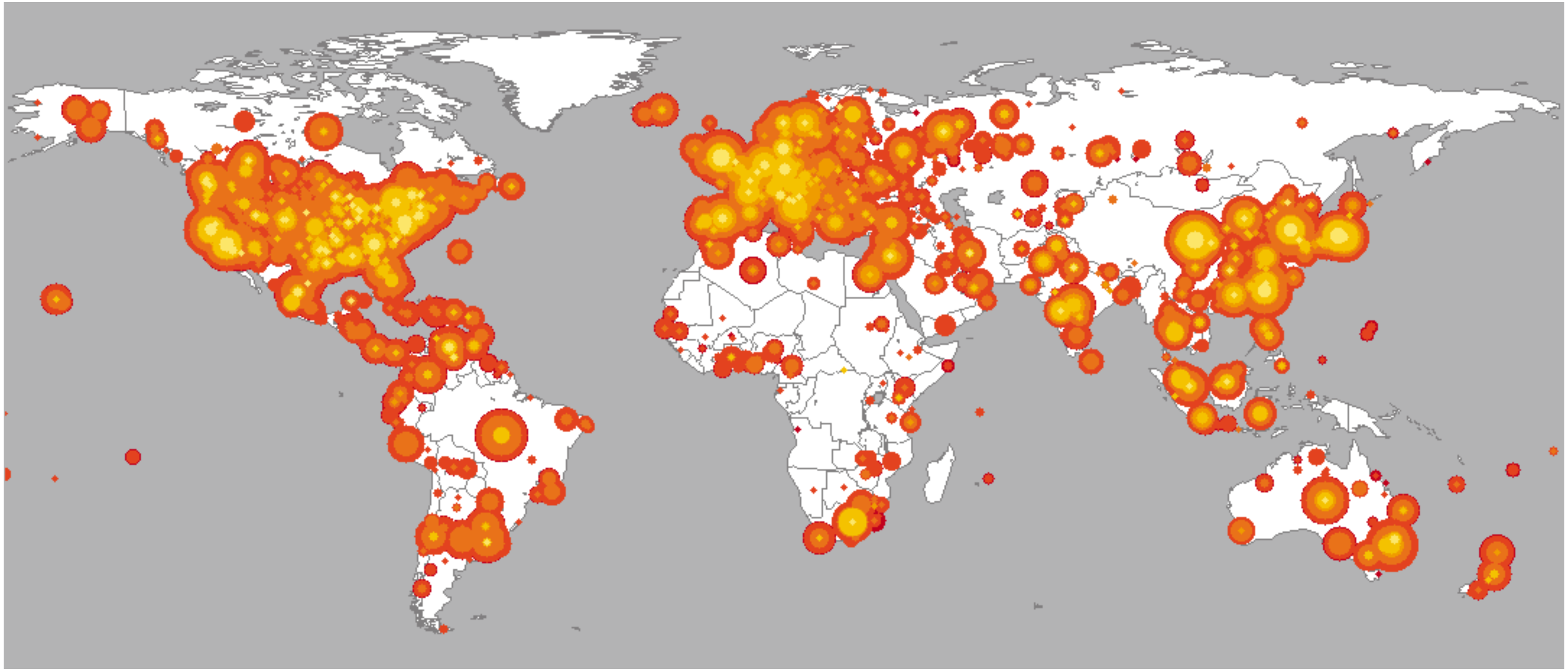
- Variety of attacks
  - Buffer overflow attack against fingerd on VAXes
  - Crack passwords
  - …
- More aggressive than intended
  - 6-10% of all internet hosts infected
- Didn’t check OS: caused Suns running BSD to crash
- End result: $10-100M damages, probation, community service

Code Red - 2001

- Propagation: Exploited an overflow in MS-IIS server
- 300,000 machines infected in 14 hours
- At peak, more than 2000 new infections/minute

- Payload 1: website defacement
  - “HELLO! Welcome to http://www.worm.com! Hacked By Chinese!”

- Payload 2: time bomb
  - Day of month 1-20: Spread
  - Day of month 20+: Attack (flood 198.137.240.91 = whitehouse.gov)



Code Red Propagation

- Spread by randomly scanning the entire 32-bit IP address space
  - Pick a pseudorandom 32-bit number = IP addr
  - Send exploit packet to that address
  - Repeat
- This is a very common worm technique
- If found c:\notworm then do nothing
- Whitehouse.gov changed its IP address
- Made the attack portion useless

Other Examples

- SQL Slammer
- Heartbleed
- Stuxnet
- See previous lecture notes

Malware Summary

- Technological arms race between those who wish to detect and those who wish to evade detection
- Started off innocuously
- Became professional, commoditized
  - Economics, cyber warfare, corporate espionage
- Advanced detection: based on behavior, anomalies
  - Must react to attacker responses

END