Last time

We finished up

Buffer overflows

and other memory safety vulnerabilities

By looking at

Overflow Defenses

- Buffer overflow defenses (and workarounds)
- Format string vulnerabilities
- Integer overflow vulnerabilities

This time

Continuing with
Software Security

Getting sick with
Malware

- Types of malware
- How viruses work
- Detecting viruses (and counter-measures)
- Case studies

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
  - Attacker with physical access downloads & runs it

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
  - Attacker with physical access downloads & runs it

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

Malware

What can it 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 (more later this lecture)
  - Crash the machine, e.g., by over-consuming resources
    - Fork bombing or “rabbits”: while(1) { fork(); }
  - 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

Malware

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…"

- Some attach themselves to other pieces of code
  - Viruses: run when the user initiates something
    - Run a program, open an attachment, boot the machine
  - Worms: run while another program is running
    - No user intervention required

Self-propagating malware

- Virus: propagates by arranging to have itself *eventually* executed
  - At which point it creates a new, additional instance of itself
  - Typically infects by altering *stored* code
  - User intervention required

- 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

Self-propagating malware

- Virus: propagates by arranging to have itself *eventually* executed
  - At which point it creates a new, additional instance of itself
  - Typically infects by altering *stored* code
  - User intervention required
- 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

The line between these is thin and blurry
Some malware uses 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

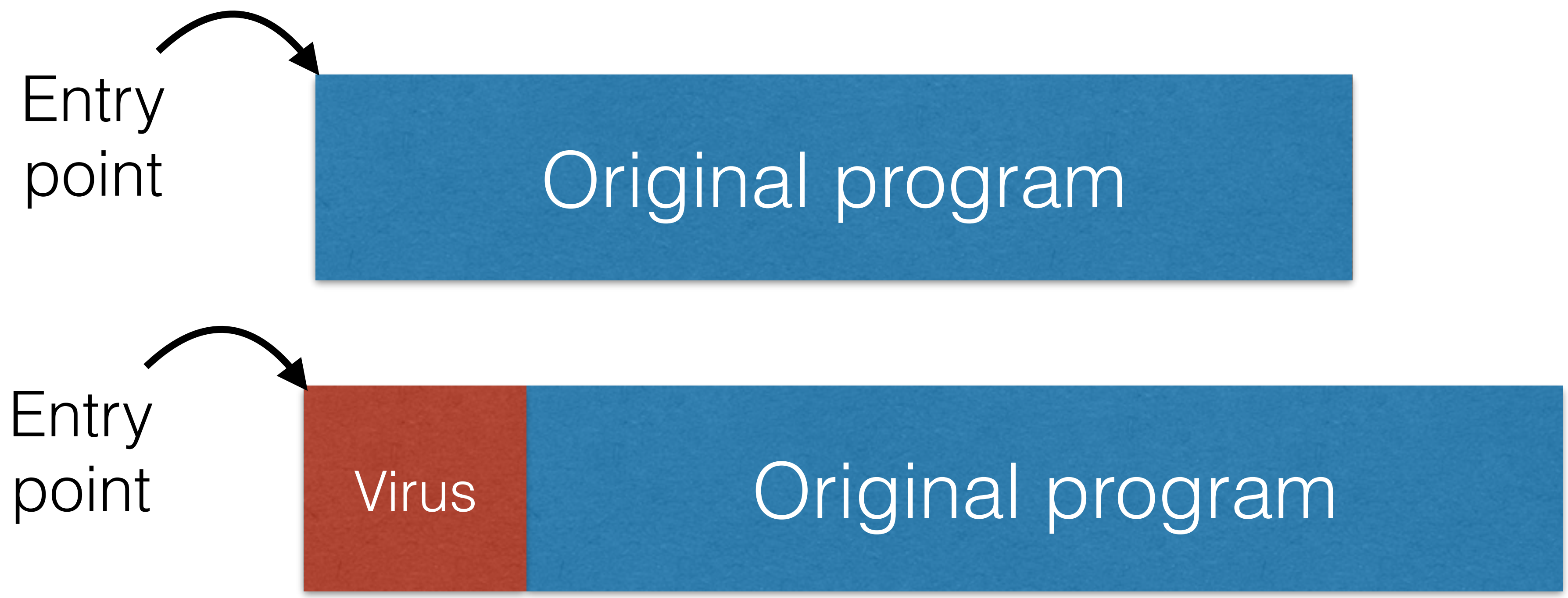
- Two *orthogonal* aspects define a virus:
  1. How does it **propagate**?
  2. What else does it do (what is the "**payload**")?

- 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

How viruses infect other programs



Take over the entry point

Viruses are classified by what they infect

Viruses are classified by what they infect

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

Viruses are classified by what they infect

- Document viruses
  - Implemented within a formatted document
  - Word documents (very rich macros)
  - PDF (Acrobat permits javascript)
  - (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
  - 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)

Viruses are classified by what they infect

- Document viruses
  - Implemented within a formatted document
  - Word documents (very rich macros)
  - PDF (Acrobat permits javascript)
  - (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
  - 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

Viruses have resulted in a technological arms race

The key is ***evasion***

Viruses have resulted in a technological arms race

The key is ***evasion***

Mechanisms for evasive **propagation**

Viruses have resulted in a technological arms race

The key is ***evasion***

Mechanisms for evasive **propagation**

Mechanisms for **detection** and prevention

Viruses have resulted in a technological arms race

The key is ***evasion***

Mechanisms for evasive **propagation**

Mechanisms for **detection** and prevention

Viruses have resulted in a technological arms race

The key is ***evasion***

Mechanisms for evasive **propagation**

Mechanisms for **detection** and prevention

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

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

How viruses propagate

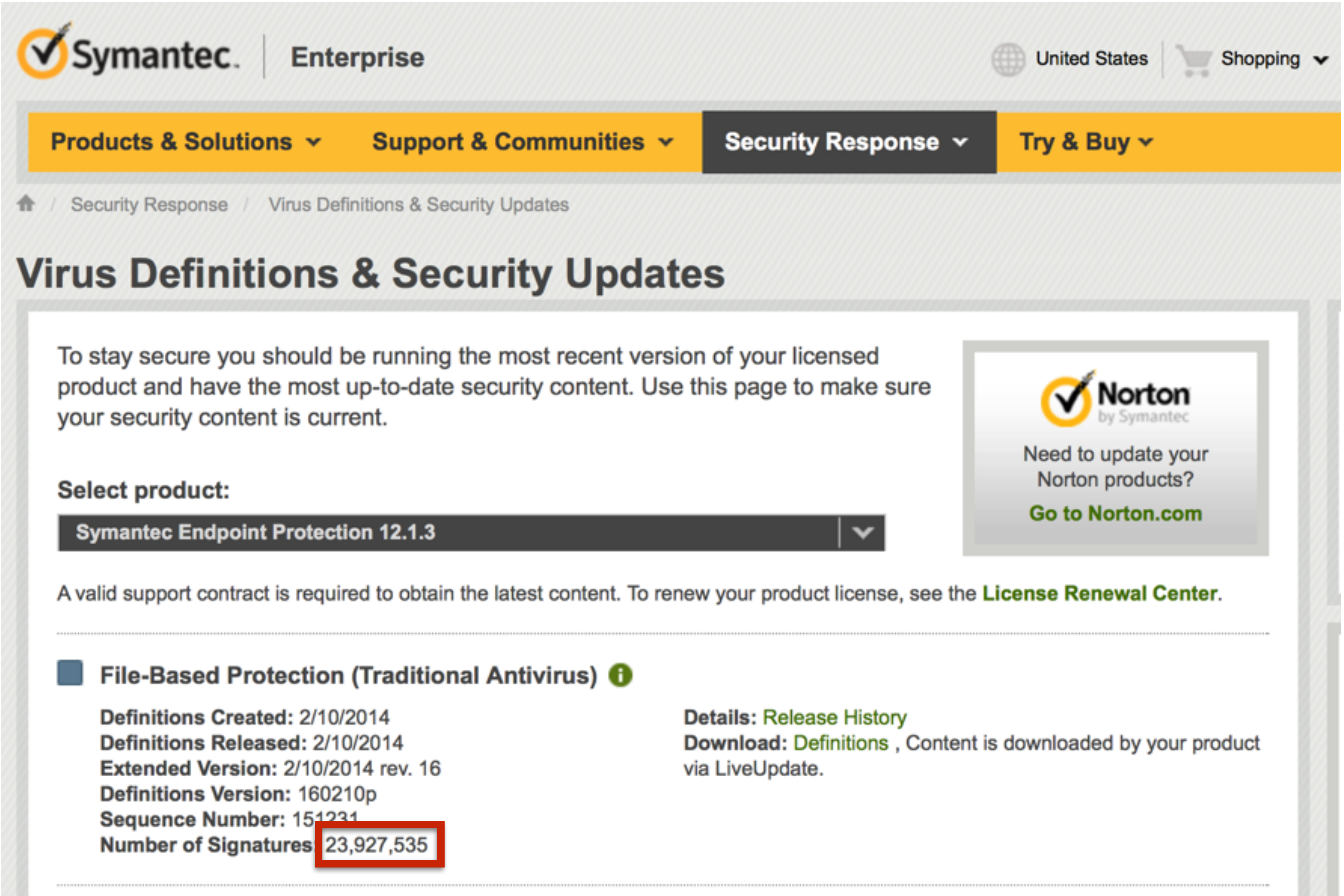
- First, the virus looks for an **opportunity to run**. Increase chances by attaching malicious code to something a user is likely to run
  - autorun.exe on storage devices
  - Email attachments
- When a virus runs, it looks for an **opportunity to infect** other systems.
  - User plugs in a USB thumb drive: try to overwrite autorun.exe
  - User is sending an email: alter the attachment
  - Viruses can also proactively create emails ("I Love You")

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

Share

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

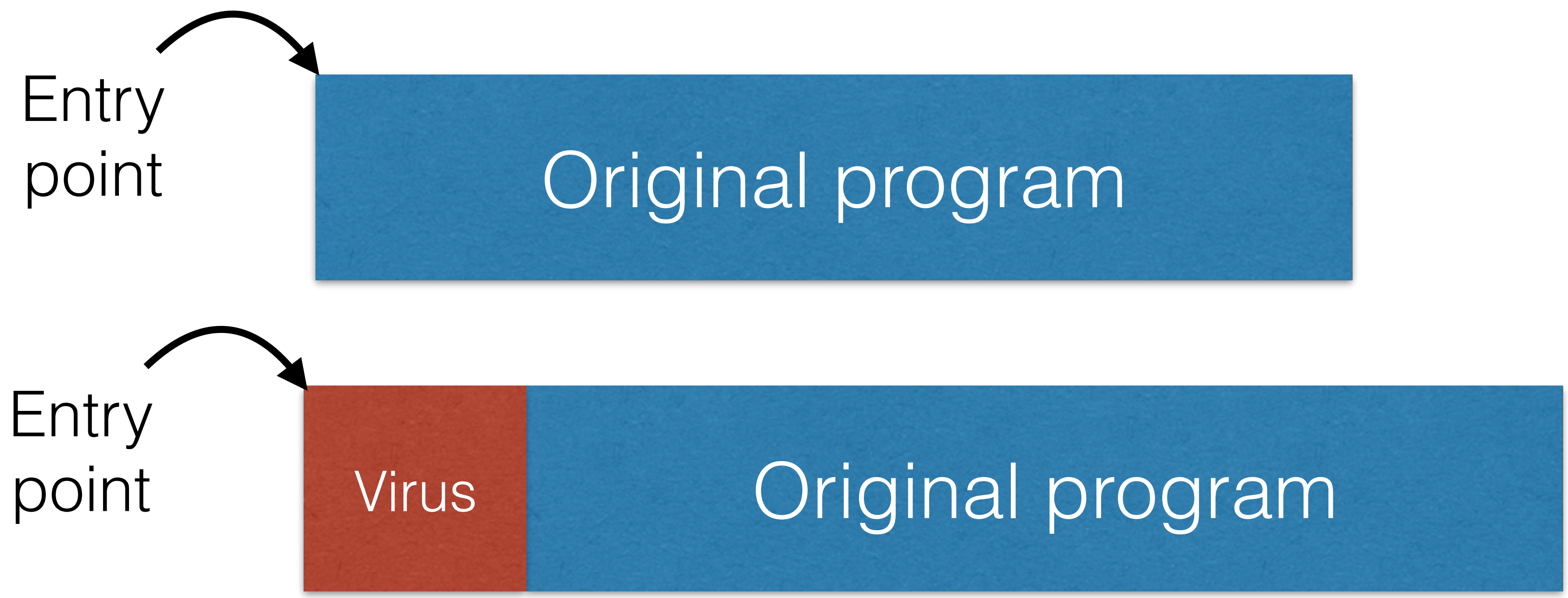
You are a virus writer

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

You are a virus writer

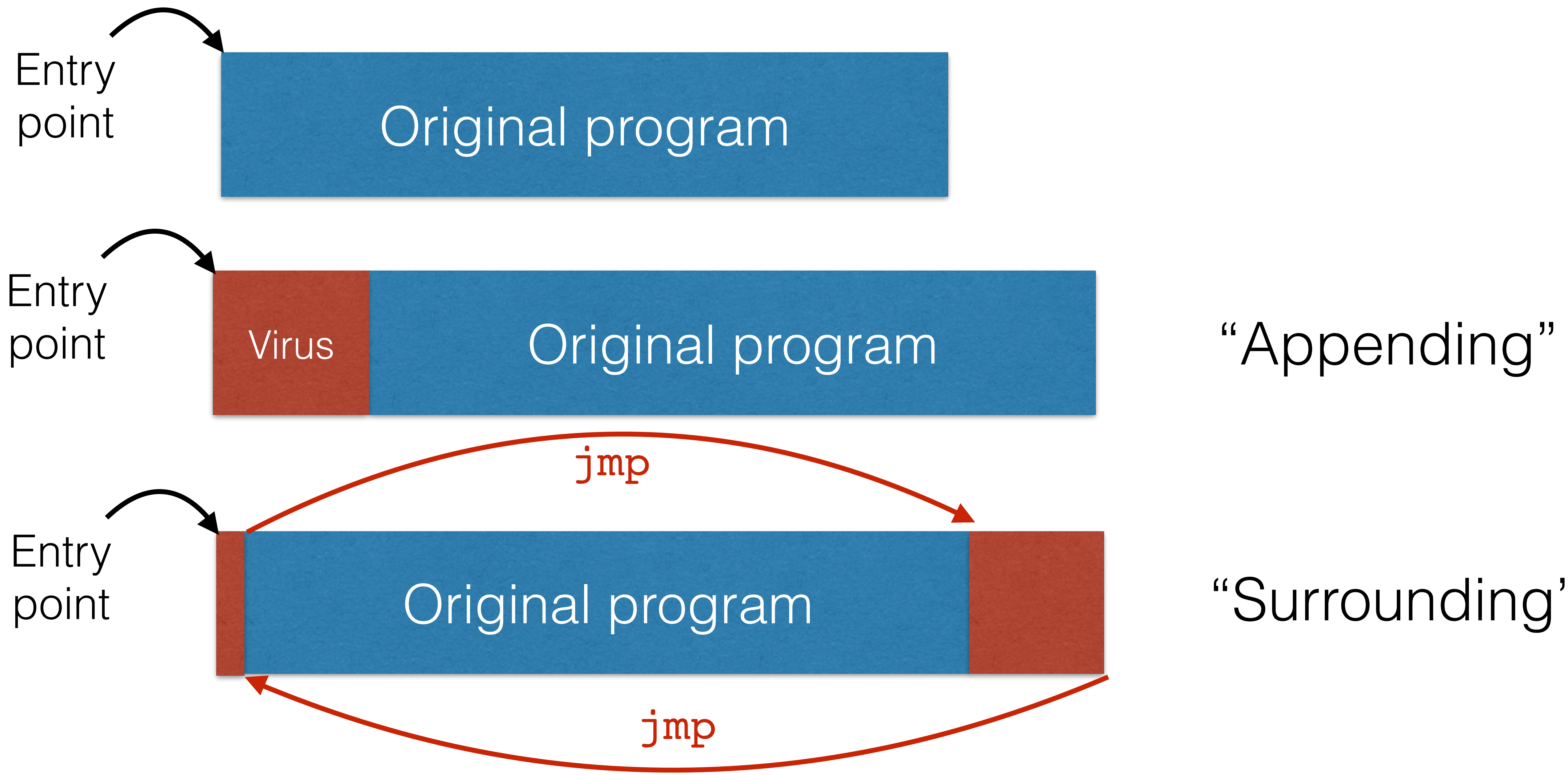
- Your goal is for your virus to spread far and wide
- How do you avoid detection by antivirus software?
  1. **Give them a harder signature to find**

How viruses infect other programs

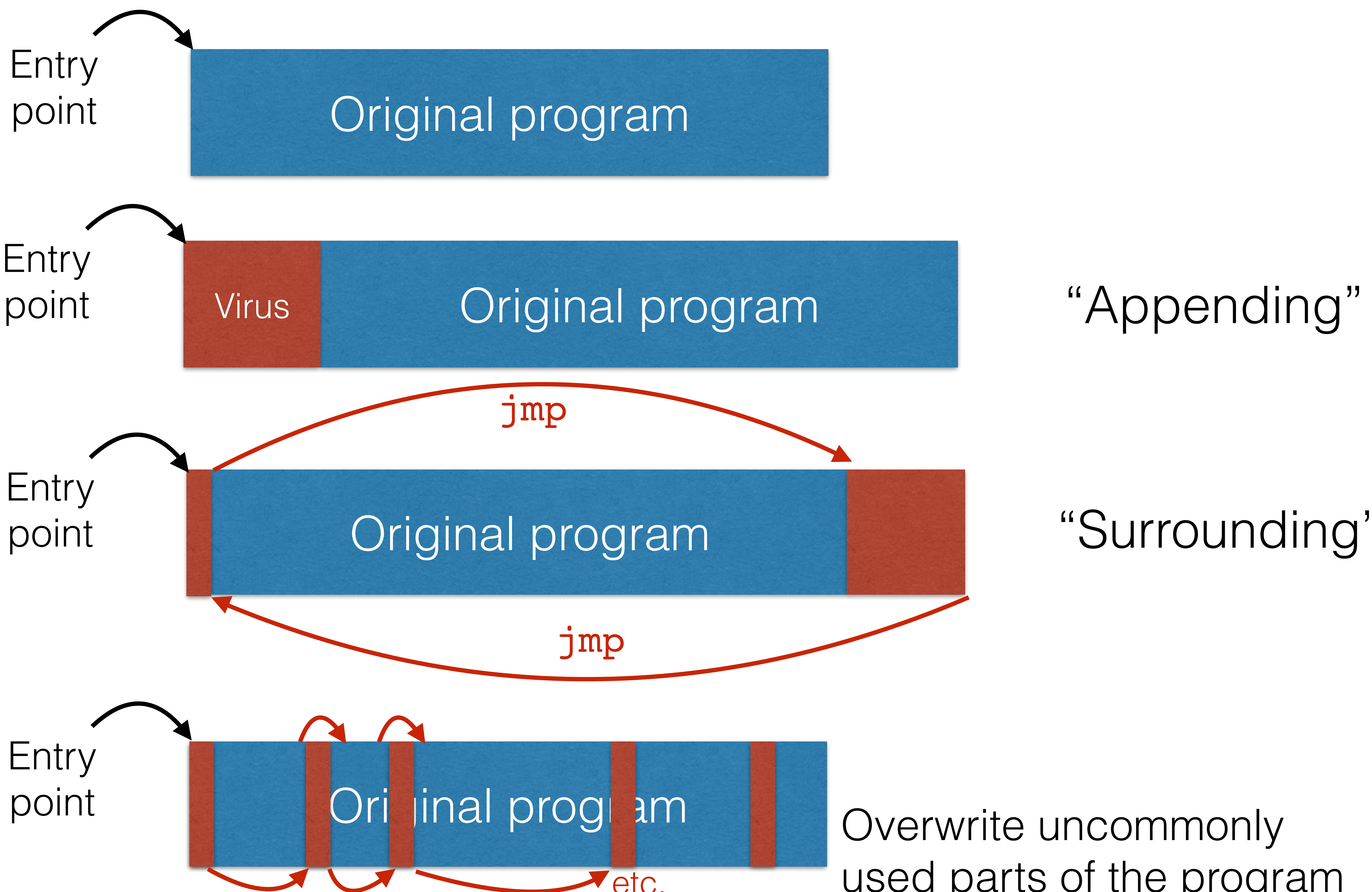


“Appending”

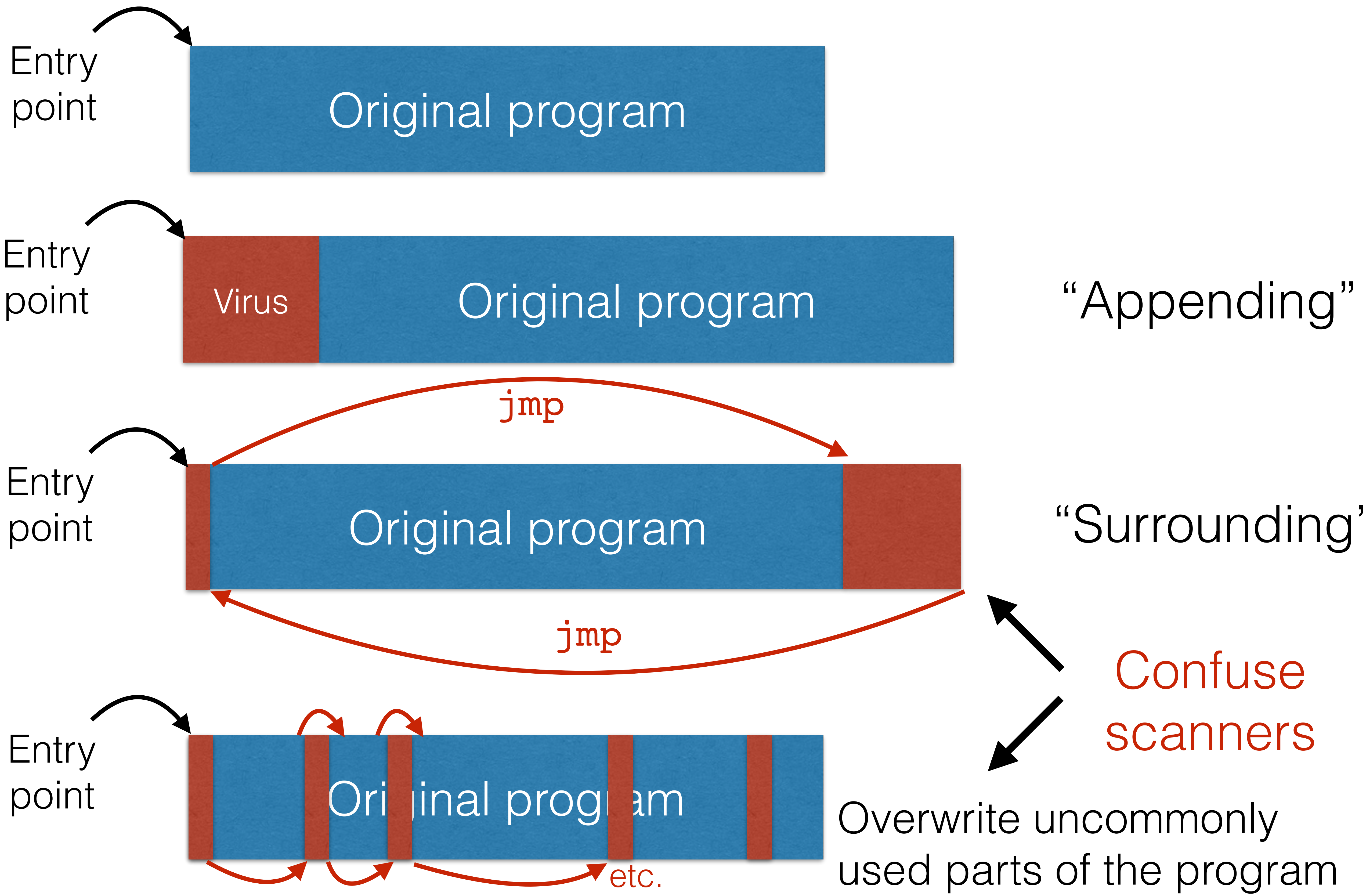
How viruses infect other programs



How viruses infect other programs



How viruses infect other programs



This time

Continuing with
Software Security

Getting sick with
Malware

- Types of malware
- How viruses work
- Detecting viruses (and counter-measures)
- Case studies

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?
    1. Give them a harder signature to find

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

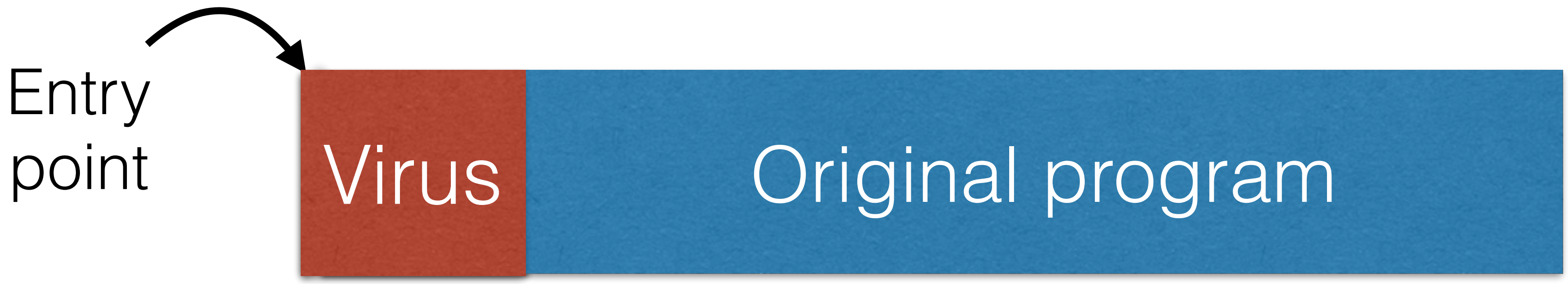
You are a virus writer

- 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

Mechanize code changes:

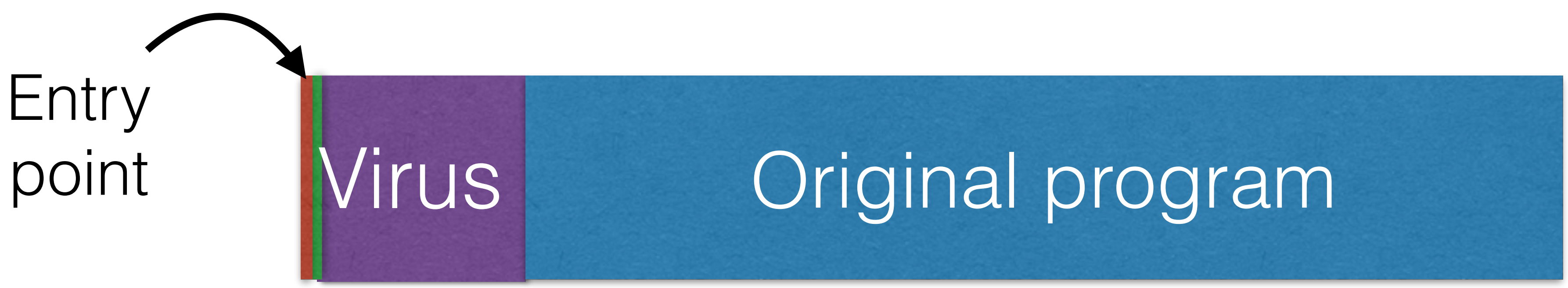
Goal: every time you inject your code, it looks different

Polymorphic viruses



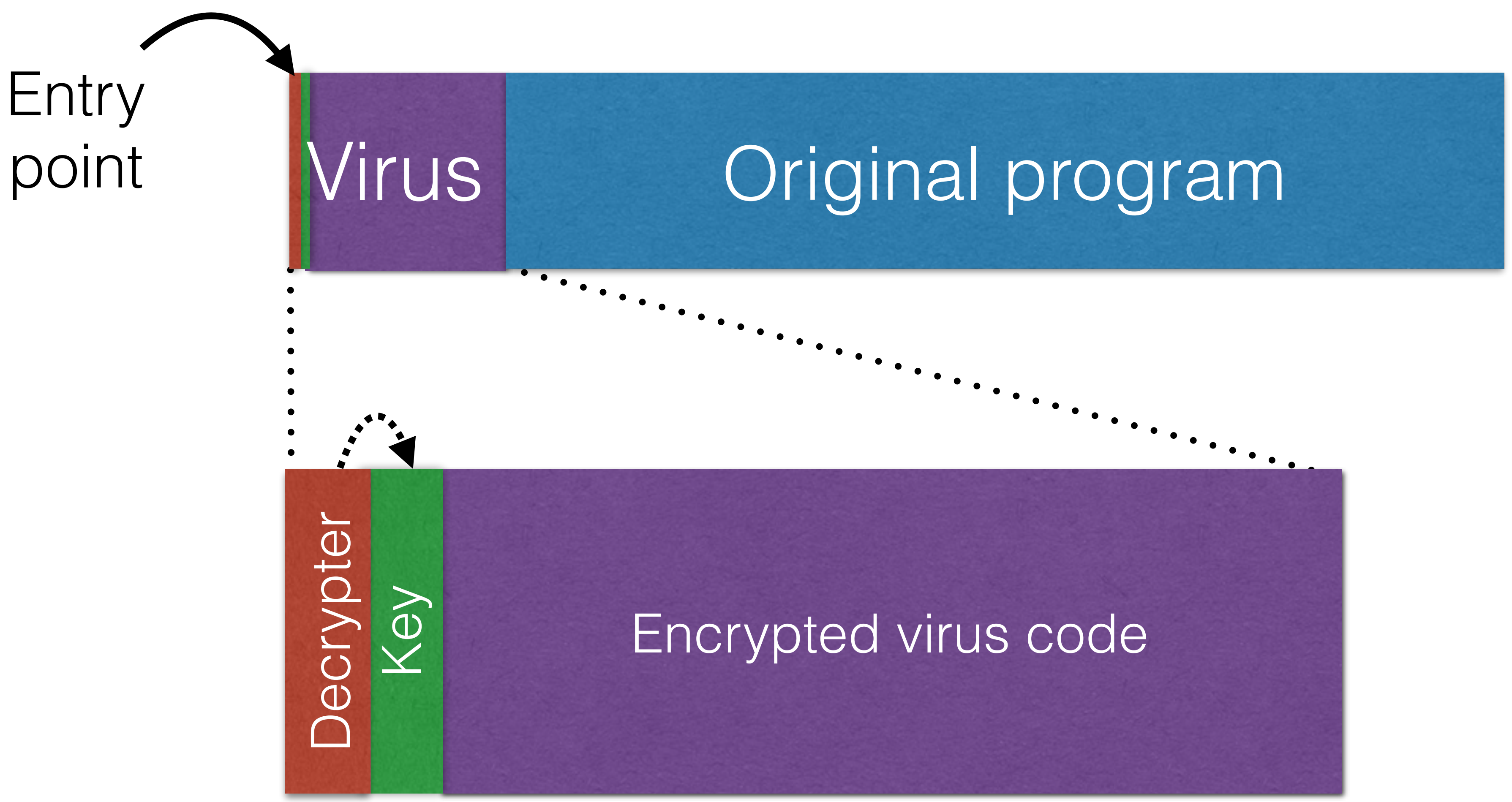
Take over the entry point

Polymorphic viruses



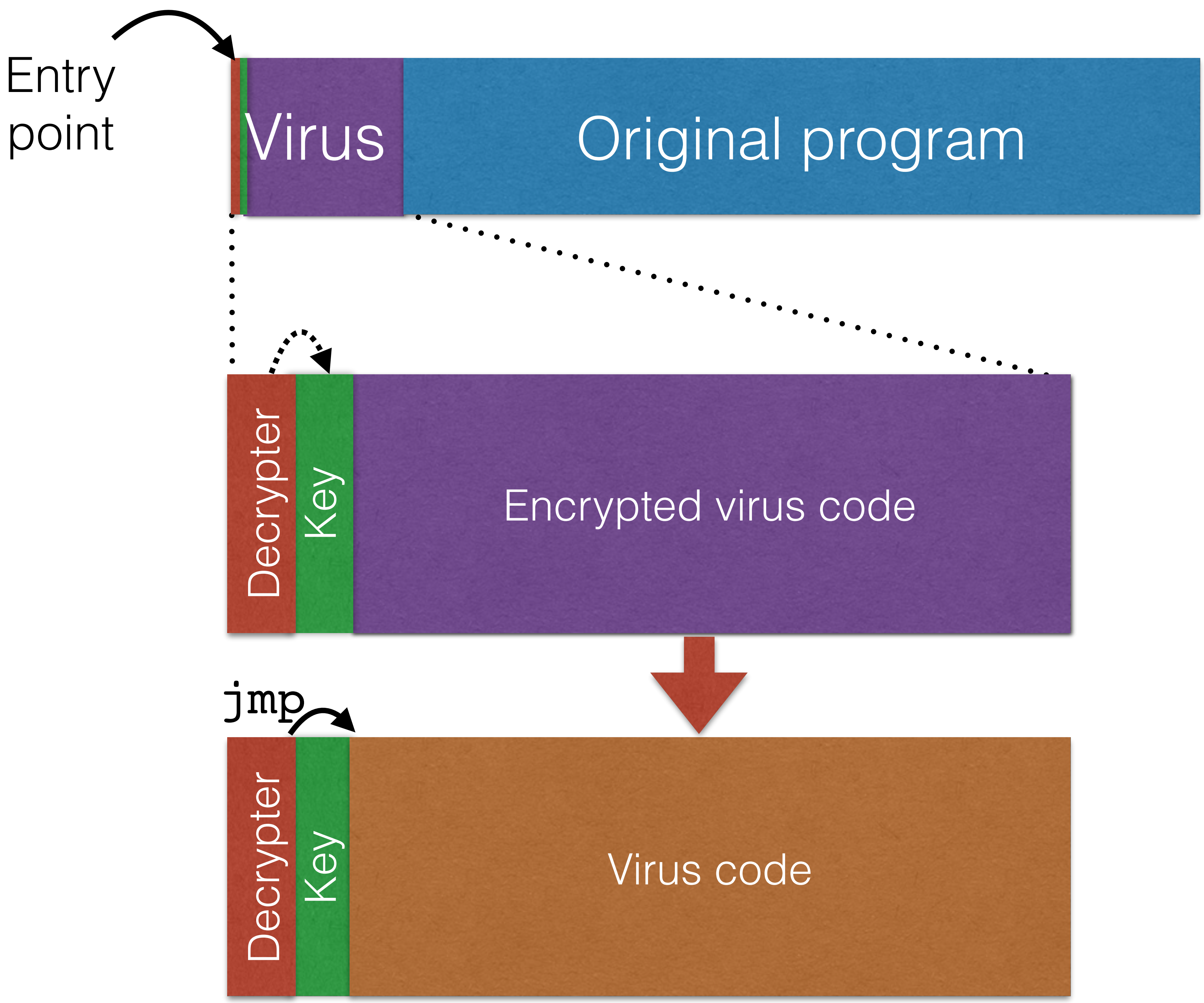
Take over the entry point

Polymorphic viruses



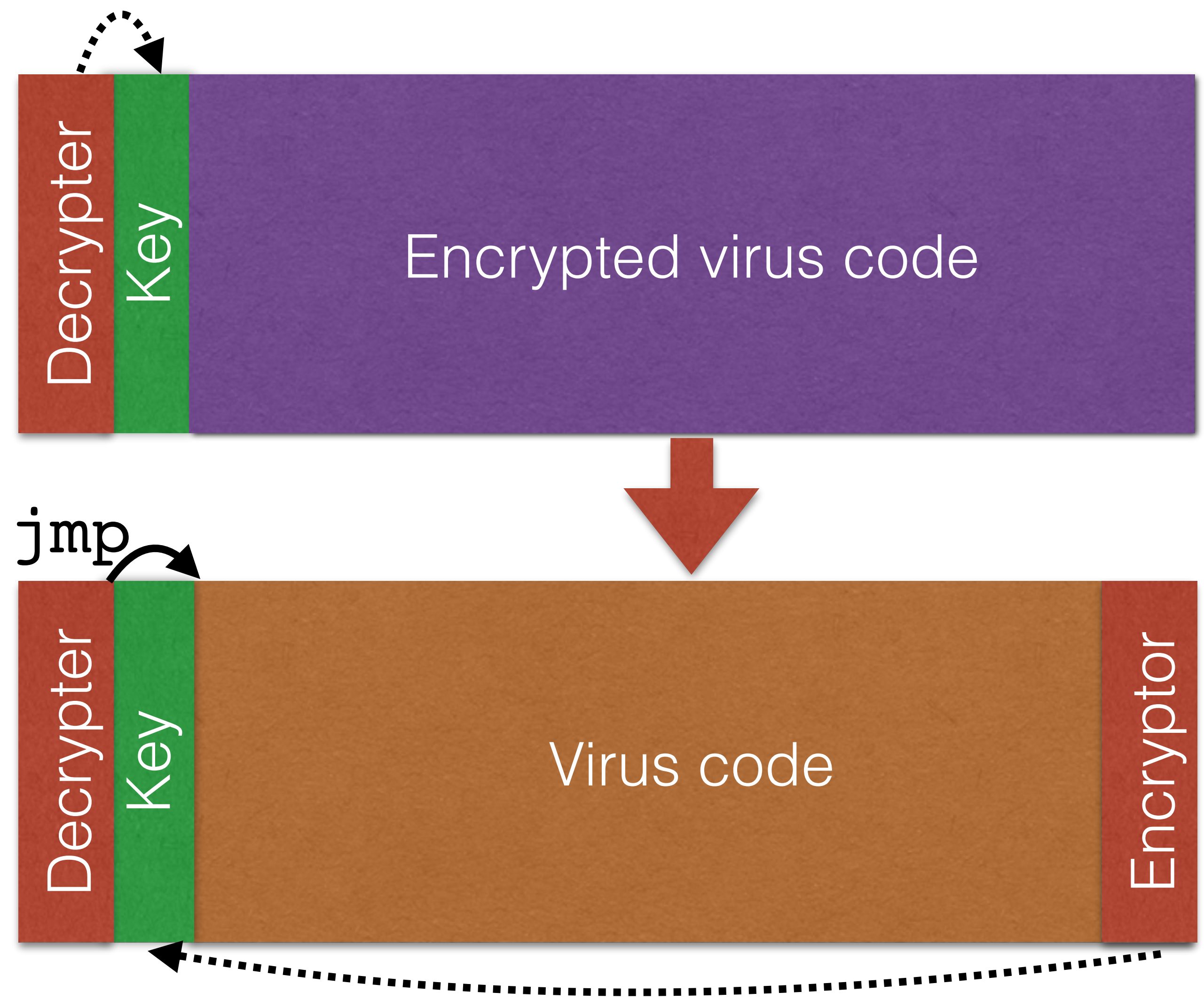
Take over the entry point

Polymorphic viruses

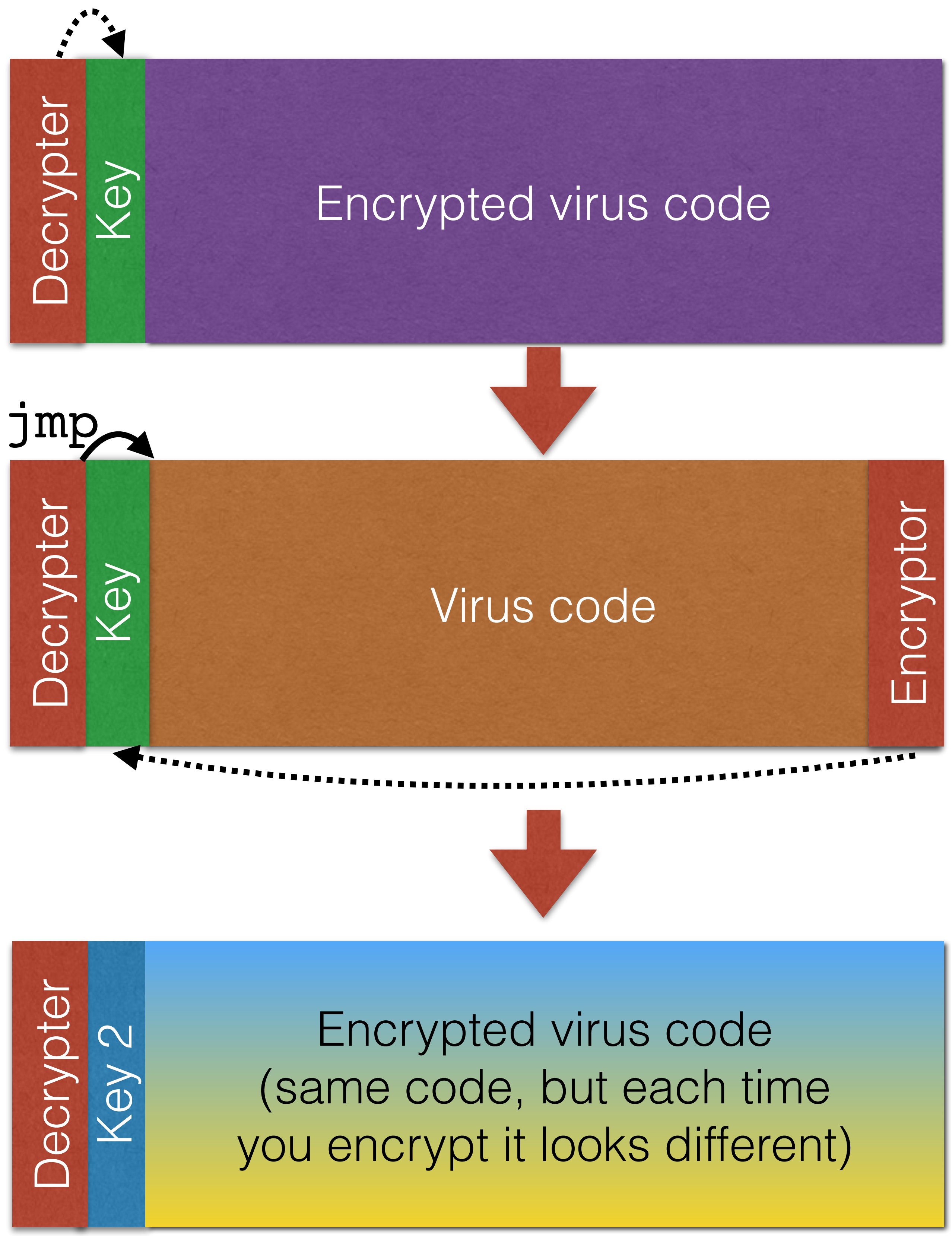


Take over the entry point

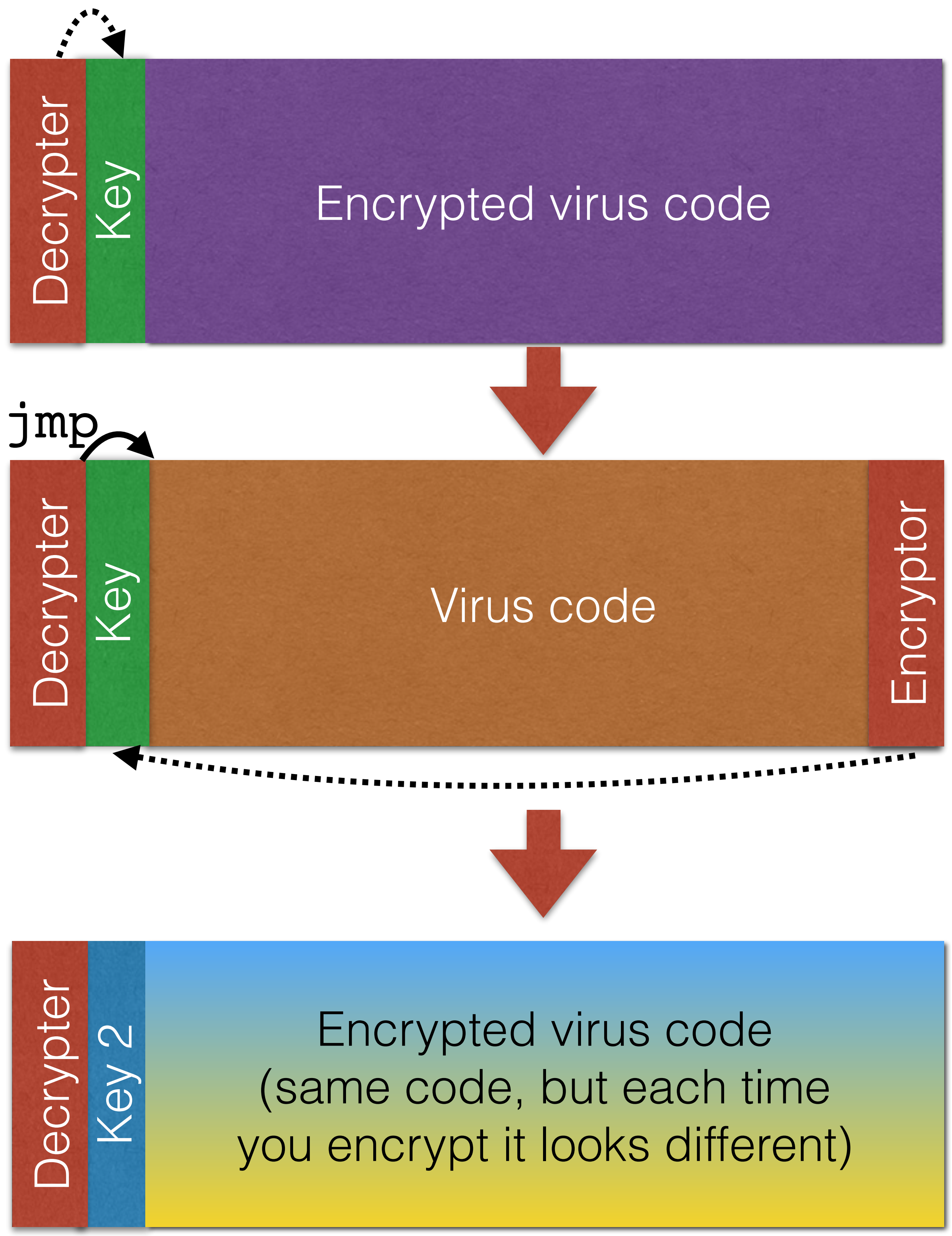
Polymorphic viruses



Polymorphic viruses



Polymorphic viruses

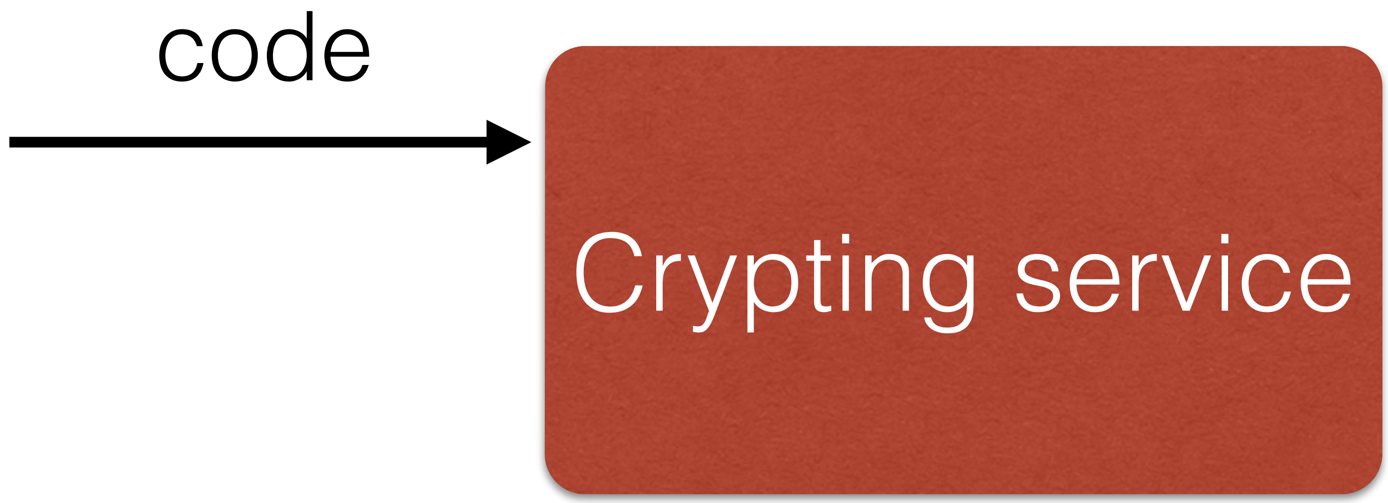


When used properly, encryption will yield a different, random output upon each invocation

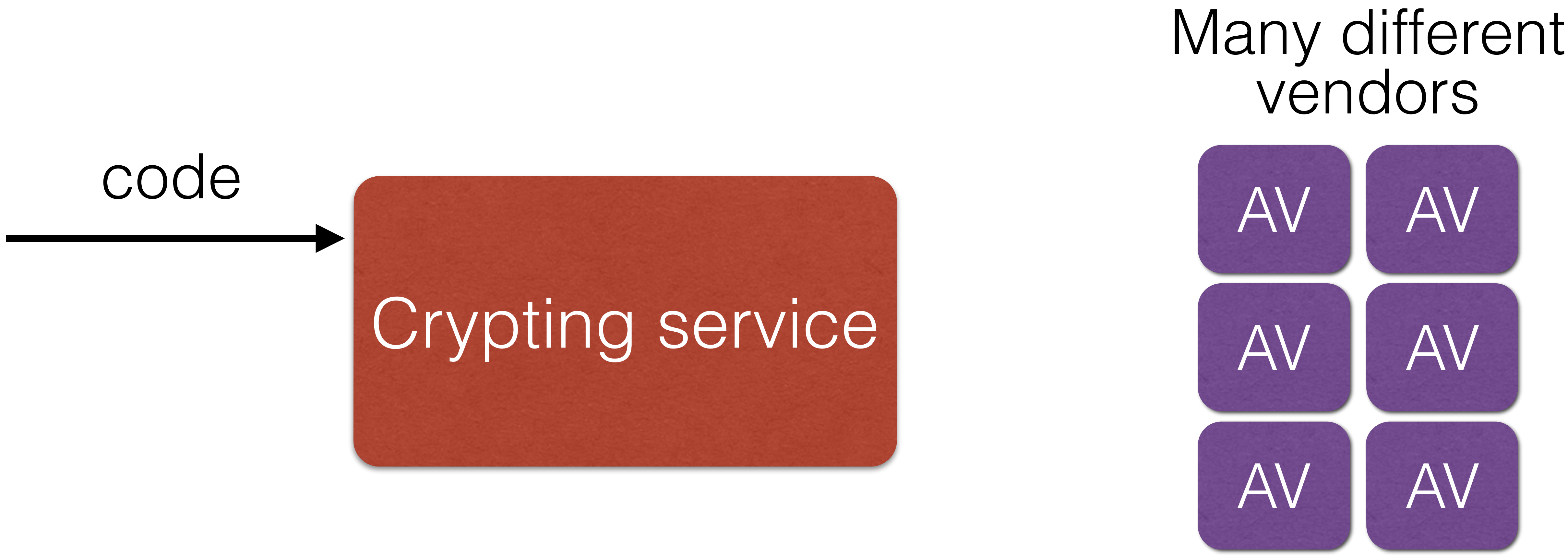
Crypting services

Crypting service

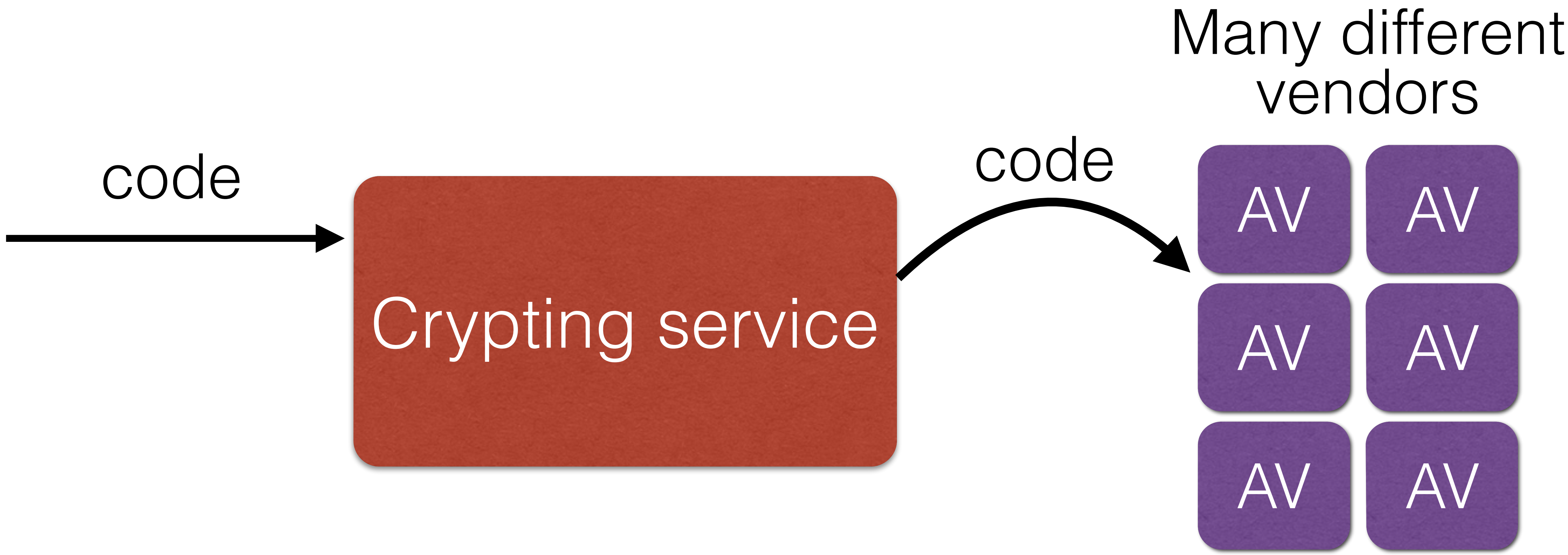
Crypting services



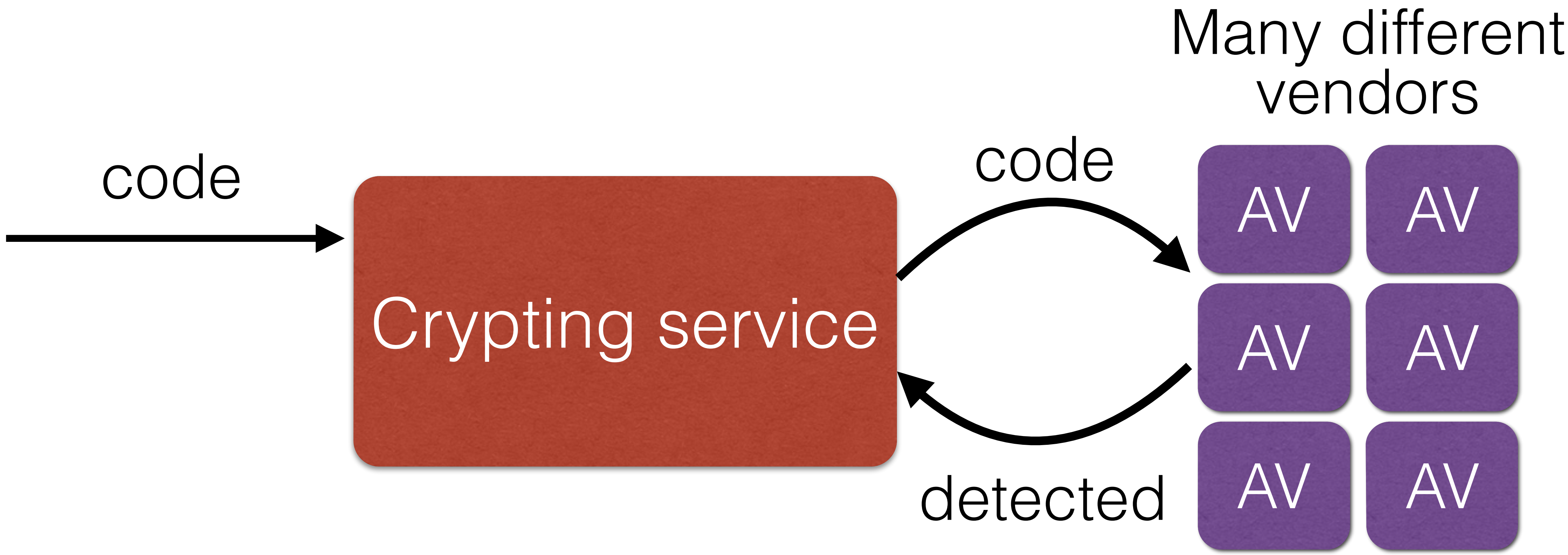
Crypting services



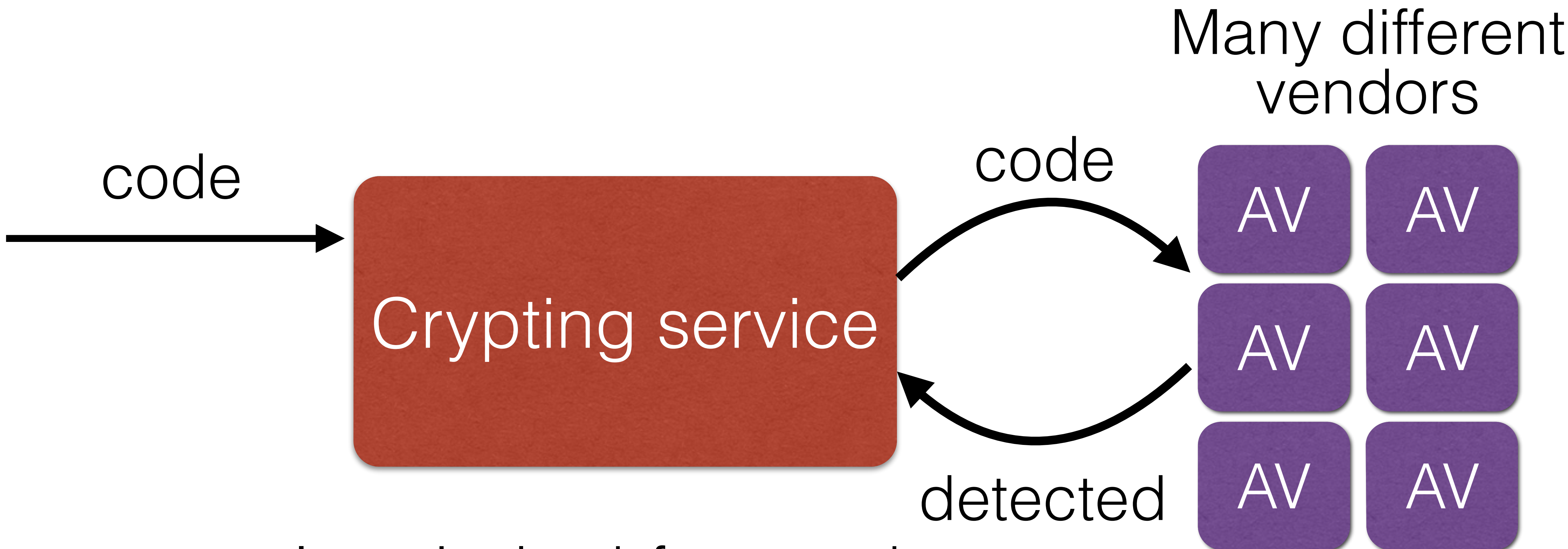
Crypting services



Crypting services



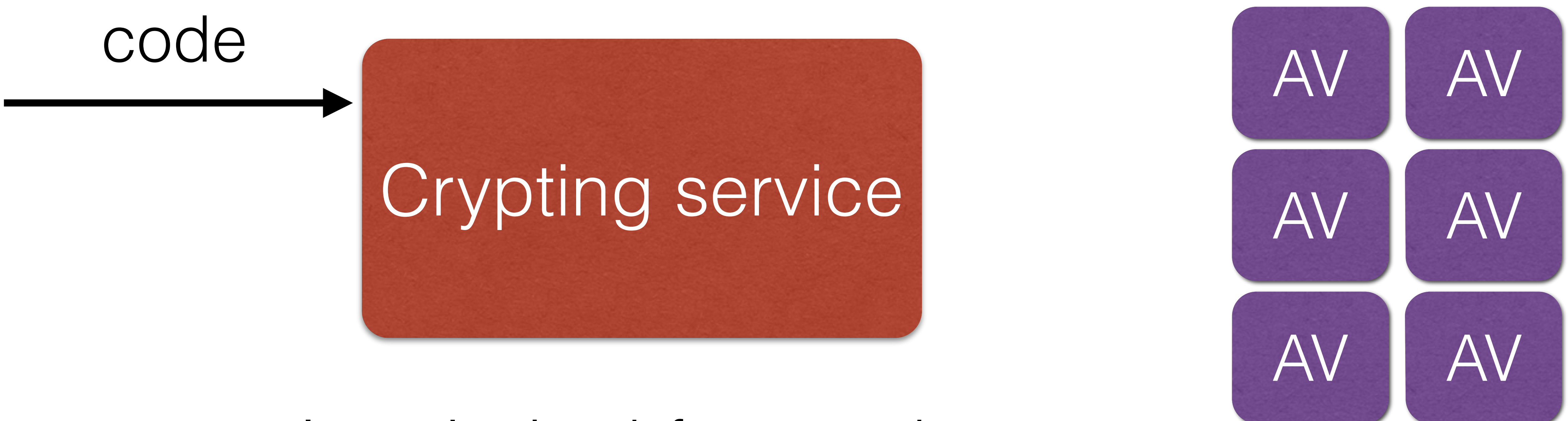
Crypting services



Iteratively obfuscate the code (encrypt + jmp + …)

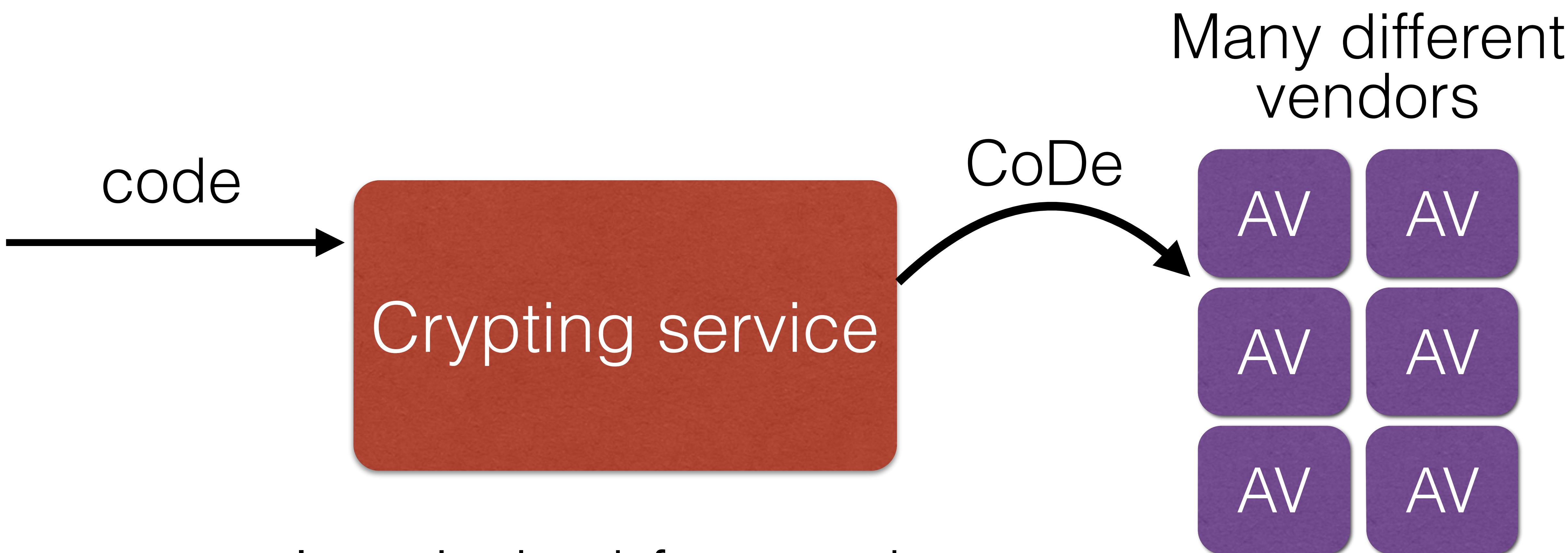
Crypting services

Many different vendors



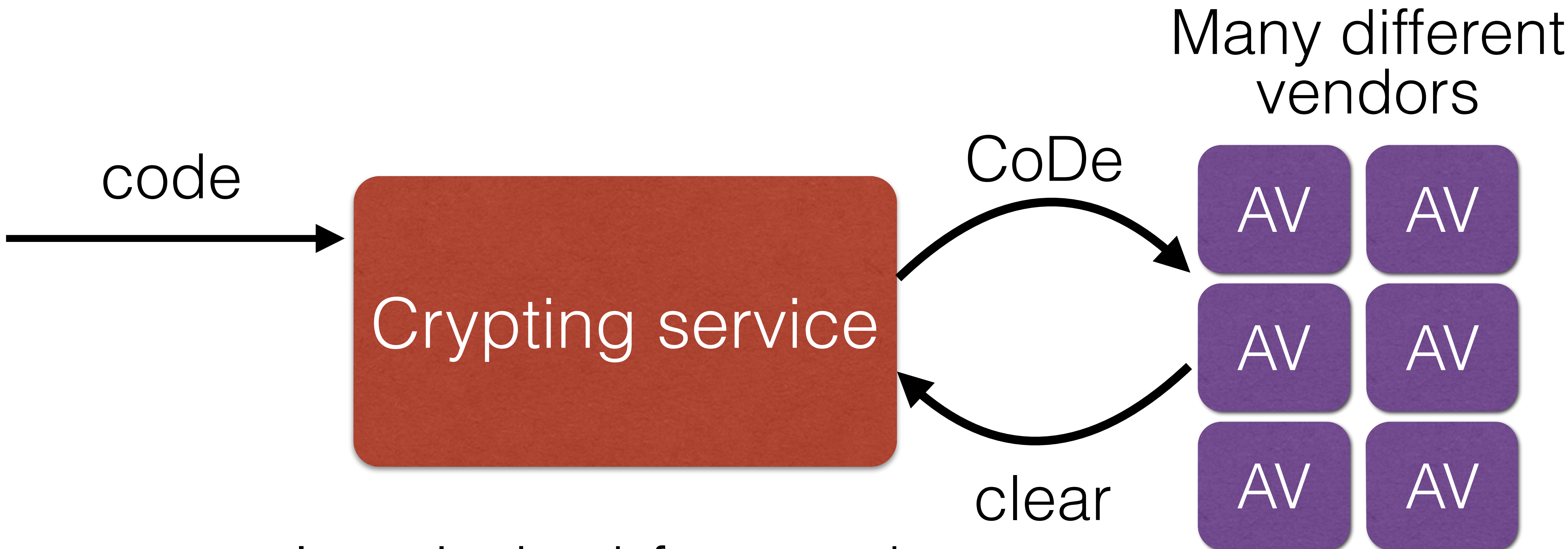
Iteratively obfuscate the code (encrypt + jmp + …)

Crypting services



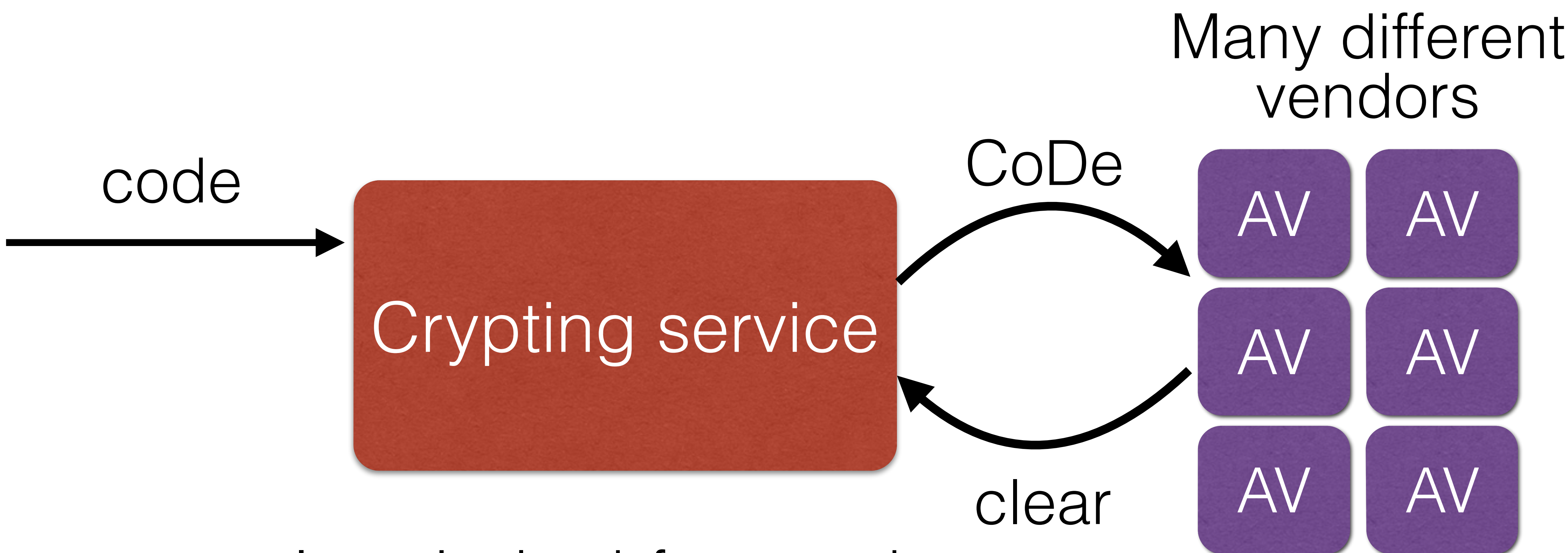
Iteratively obfuscate the code (encrypt + jmp + …)

Crypting services



Iteratively obfuscate the code (encrypt + jmp + …)

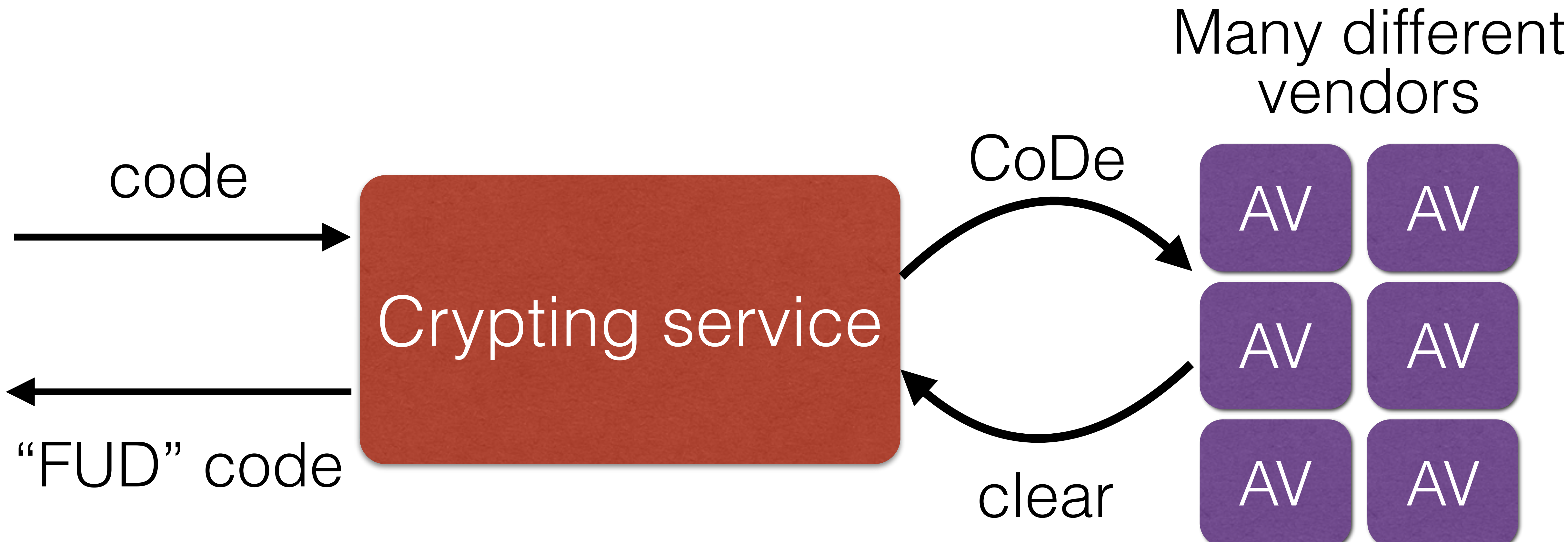
Crypting services



Iteratively obfuscate the code (encrypt + jmp + …)

Until the obfuscated code is “fully undetectable”

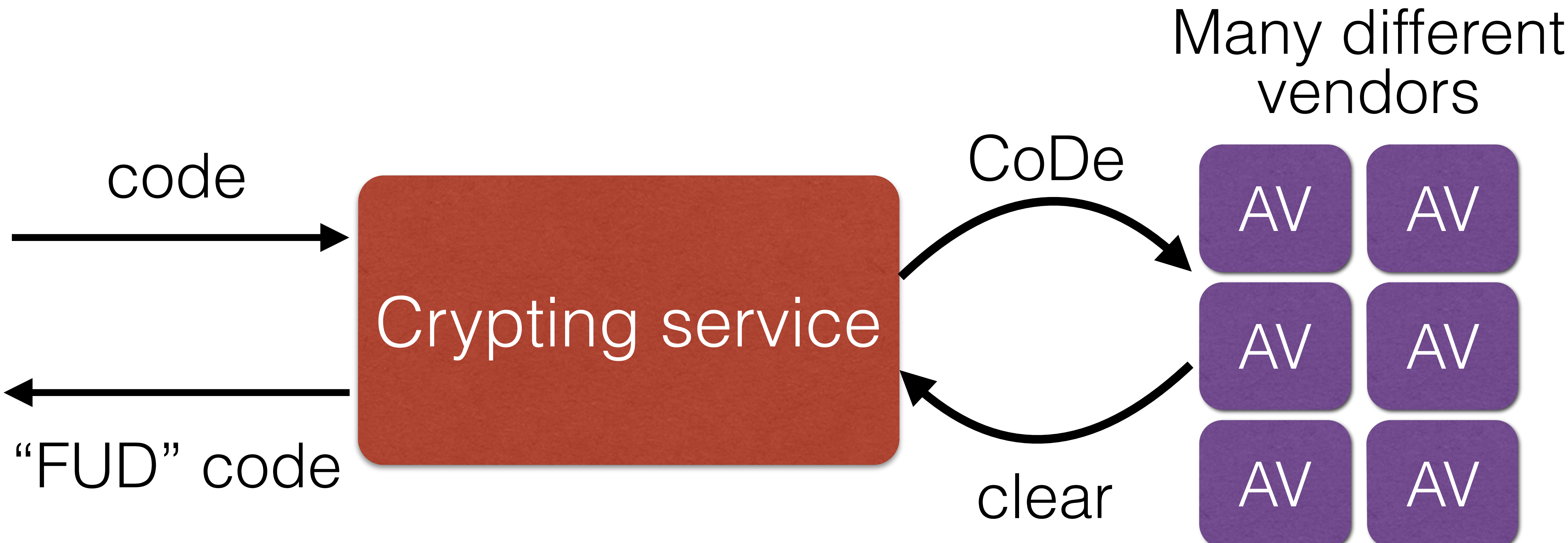
Crypting services



Iteratively obfuscate the code (encrypt + jmp + …)

Until the obfuscated code is “fully undetectable”

Crypting services



Iteratively obfuscate the code (encrypt + jmp + …)

Until the obfuscated code is "fully undetectable"

2013: Web-based crippling services
One charged $20 to "remain undetected for more than 7 days"

Polymorphic viruses: Arms race

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

Polymorphic viruses: Arms race

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

- 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

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

- 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

Metamorphic code

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
        - 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
    - Different algorithms to achieve the same thing
    - Different use of registers
    - Different constants….
- 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]
899C8618110000mov   [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]
89B4BA18110000mov   [edx+edi*4+00001118],esi
```

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

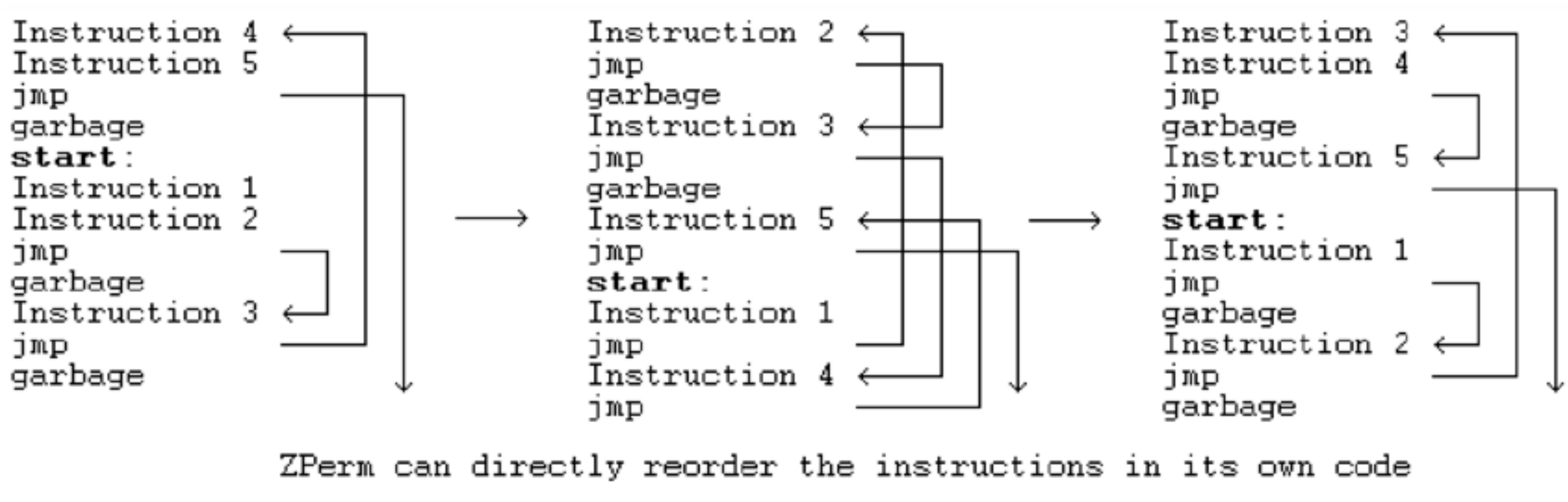
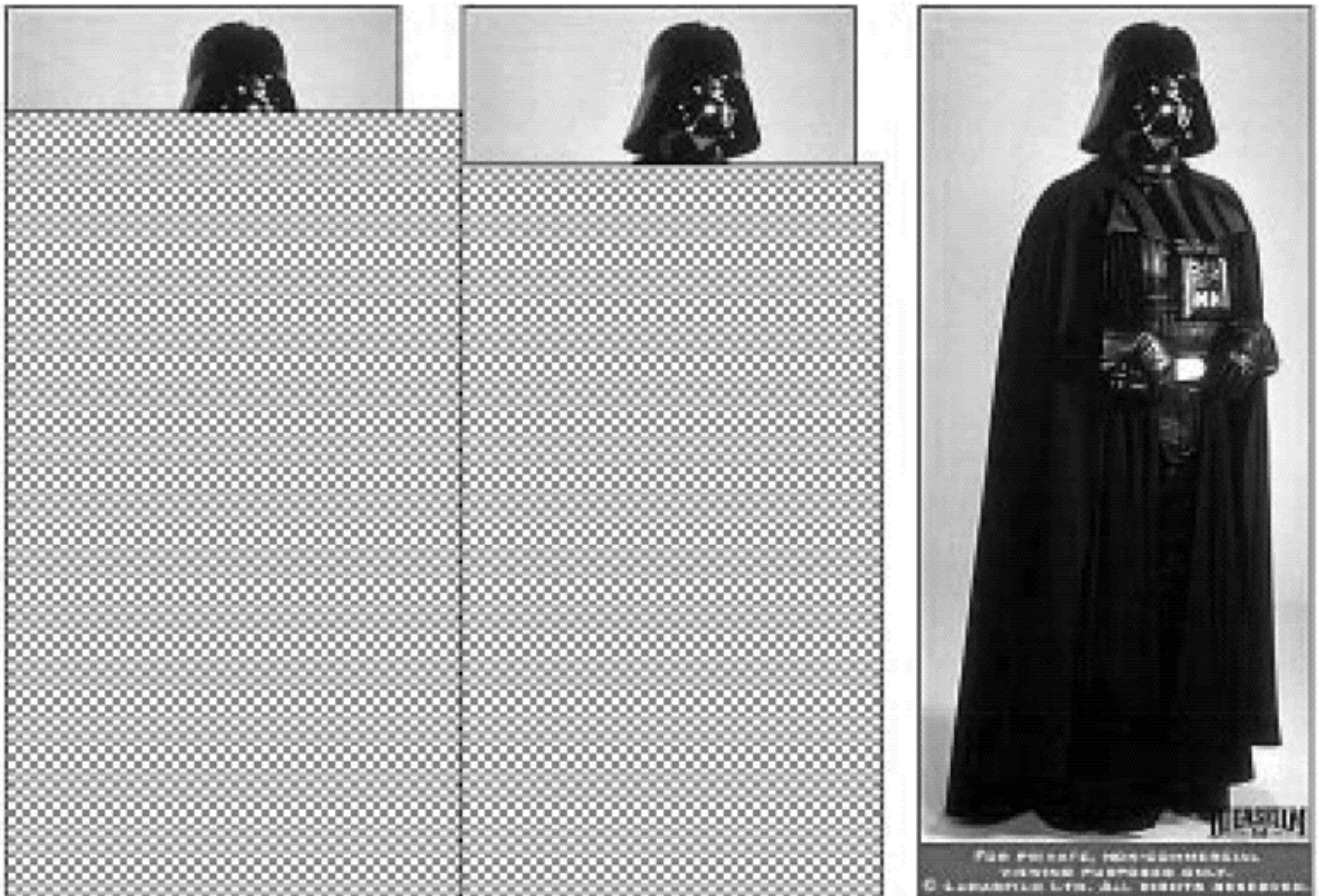


Figure 7. Zperm.A inserts JMP instruction into its code

```
a. An early generation:

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

b. And one of its later generations:

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

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

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

Figure 6: Example of code metamorphosis of Win32/Evol

Polymorphic



When can AV software successfully scan?

Figure 8: A partial or complete snapshot of polymorphic virus during execution cycle

Metamorphic

When can AV software successfully scan?

Figure 10: T-1000 of Terminator 2

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

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 signatures, thus serving as an *oracle*

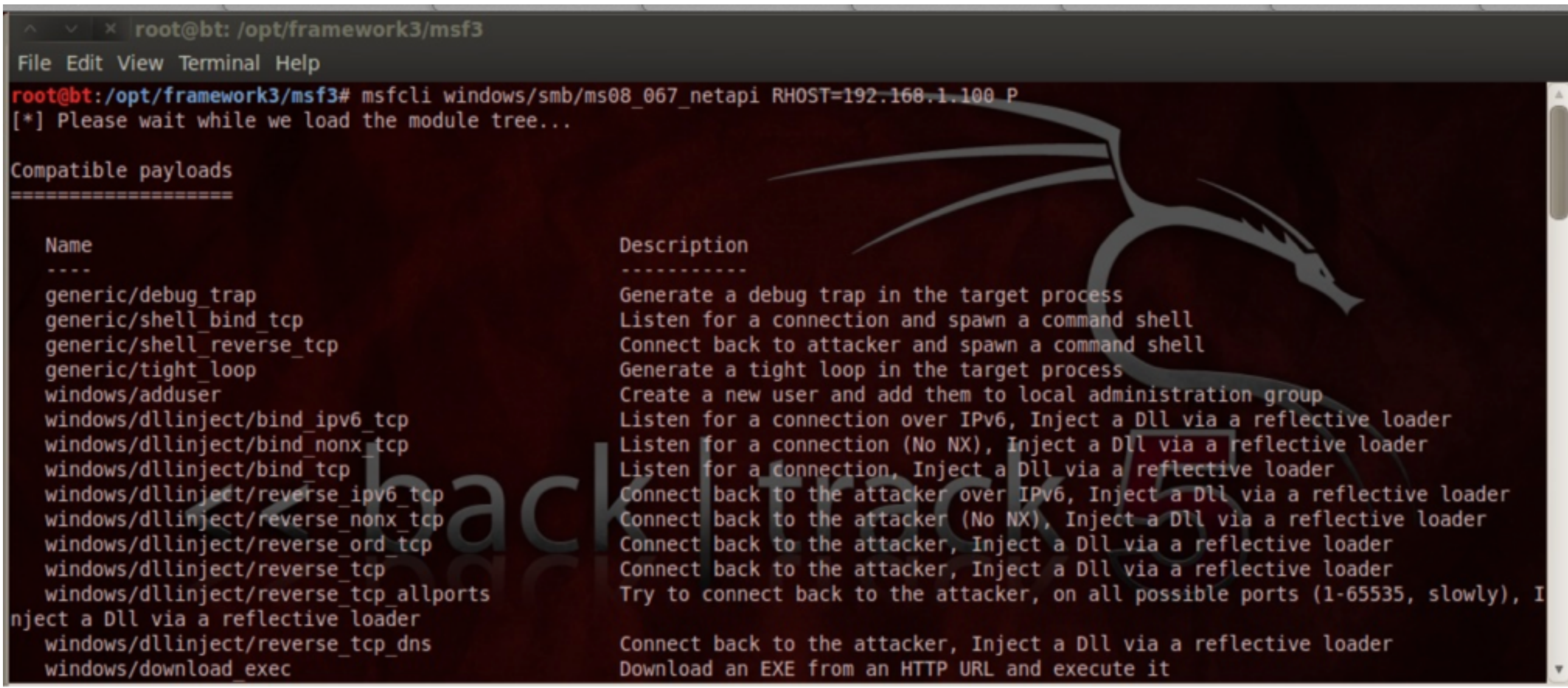
Putting it all together sounds hard

- **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

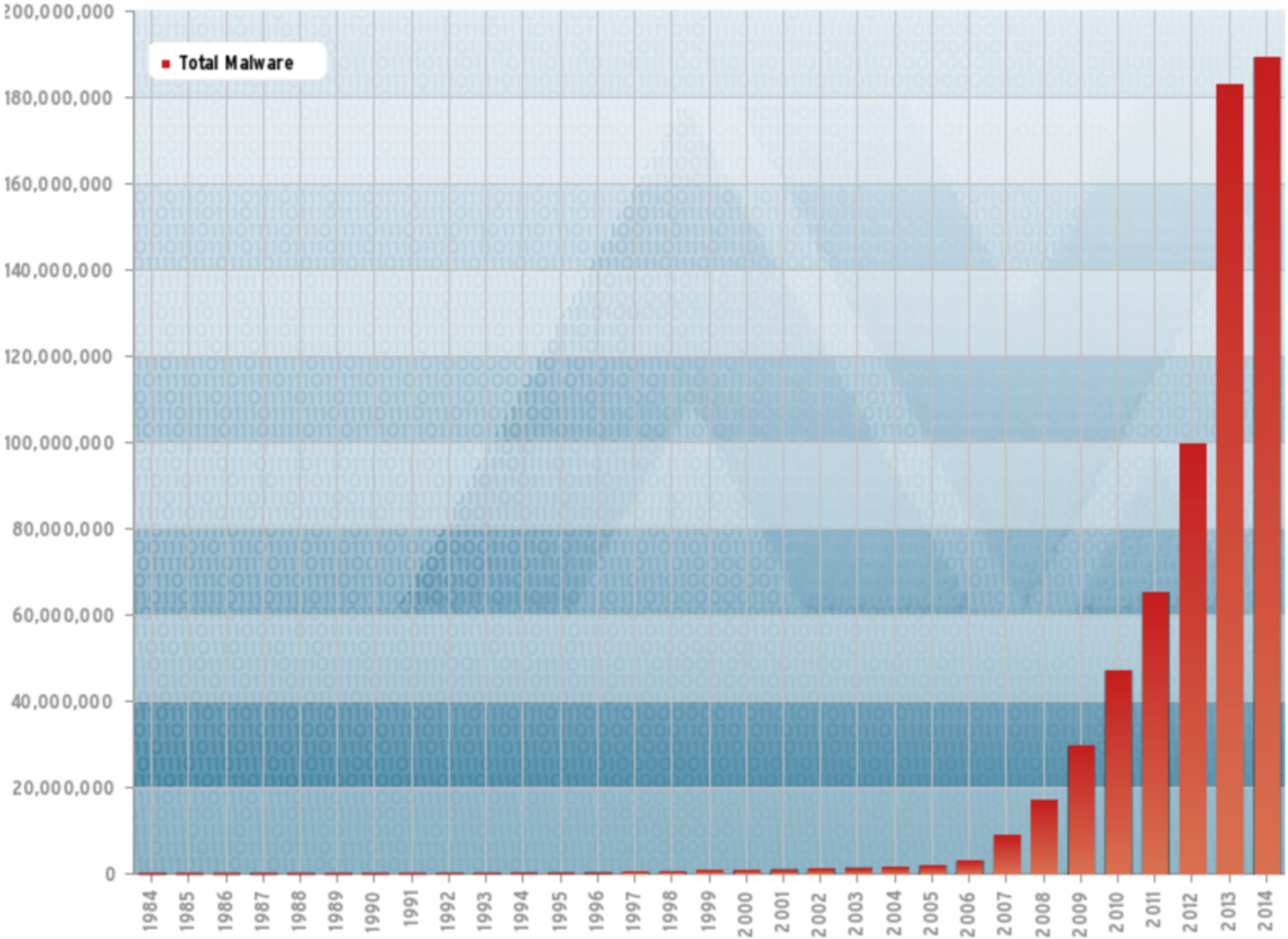
Putting it all together sounds hard

- **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



Last update: 01-21-2014 17:53

How do we clean up an infection?

- Depends what the virus did, but..
- May require restoring / repairing files
  - A service that antivirus companies sell
- What if the virus ran as root?
  - May need to rebuild the entire system
- So what, just recompile it?
  - What if the malware left a backdoor in your compiler?
    - Compile the malware back into the compiler
  - May need to use original media and data backups

Next time

Continuing with
Software Security

Going digging for
Worms
& virus case studies

Optional reading for this lecture:

"Hunting for Metamorphic"

"A History of Computer Viruses — The Famous 'Trio'"

Required reading for next time:

"How to 0wn the Internet in your Spare Time"