CS255: Computer Security

Malware

Chengyu Song 09/27/2022

Lab1: Reverse Engineering

- Goal: understand what the program does and how it works
- Approaches
  - Static: disassembler (objdump, radare2, IDA, Ghidra, Binary Ninja)
  - Dynamic: debugging (gdb, lldb, windbg)
- Why useful?
  - QA: make sure the code is correct
  - Bug fixing: figure out why
  - Malware analysis

Malware

- Malware = **Mal**icious Soft**ware**
  - Virus
  - Worm
  - Botnet
  - Spyware
  - Rootkit
  - Ransomware
  - Crypto miner
  - Keylogger
  - Remote Control
  - etc

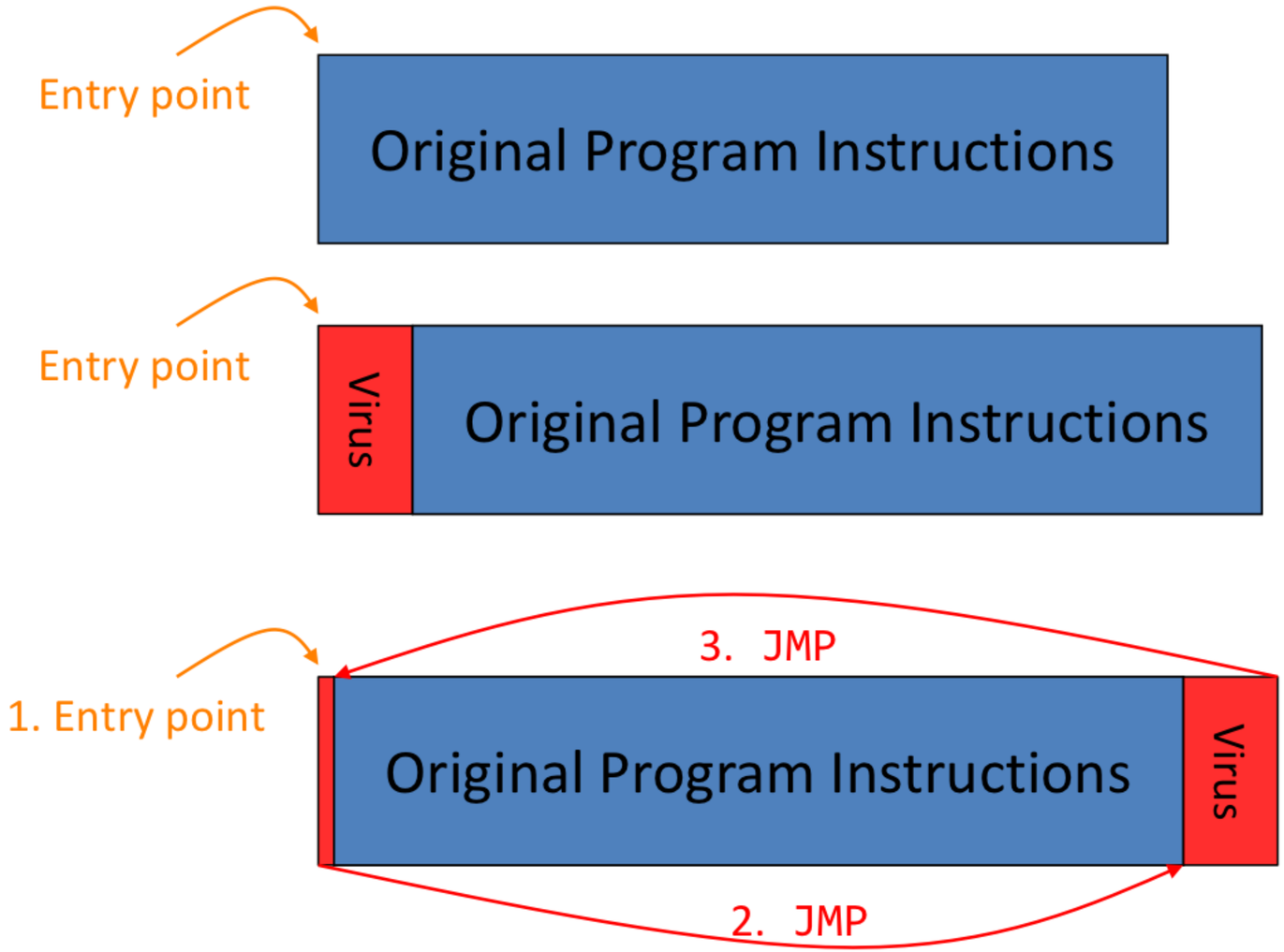
Computer Virus

- Virus = code that replicates
- Originates from a theoretical question
  - **Can a program reproduce itself like organism?**
  - "Theory of self-reproducing automata", *John von Neumann*, 1966
  - Quine: `a='a=%r;print(a%%a)';print(a%a)`
- Like real virus, computer virus
  - **Infect** other programs for replication
  - **Hijack** the normal workflow for activation

Propagation of Virus

- General infection strategy: find some code lying around, alter it to include the virus
  - Executables, boot sectors, script (including embedded)
- Example one: attached USB thumb drive
  - Alter executables it holds to include the virus or **autorun** script
  - So once the drive is attached to another machine, boom
- Example two: email attachment
  - Alters attachment to add a copy of itself

Activation of Virus



Payload

- Besides self-reproducing, what else can the virus do?
  - Pretty much **anything**, payload is decoupled from propagation
    - Only subject to permissions of the infected program
- Examples
  - Brag or exhort (pop up a message)
  - Trash files (just to be nasty) or encrypt them (ransomeware)
  - Damage hardware (e.g., CIH)
  - Keylogging

Computer Worms

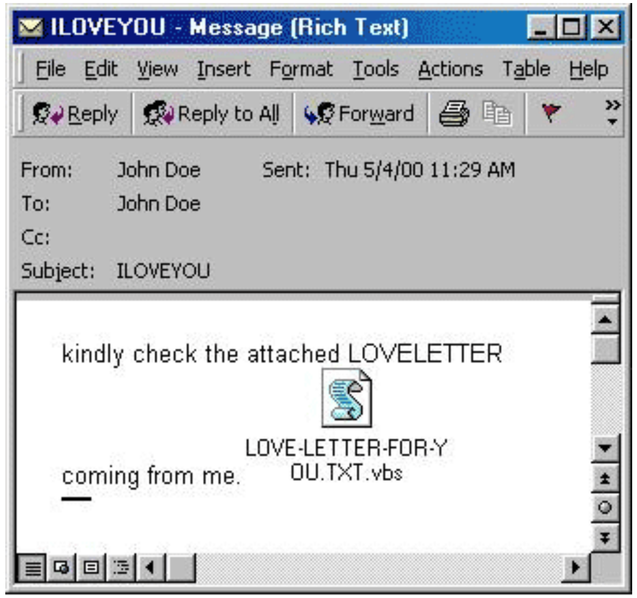
- Worm = malware that **self-propagates**
  - Propagation of virus requires certain type of user interaction
    - Execute program, open file, insert USB disk, etc
  - Worm propagate without user interaction
- How?
  - By exploit **vulnerabilities** of the target system
  - Requires interconnection

Notorious Worms (1)

- Morris (1988): the first worm
  - Scanning the local subnet
  - Exploiting a fingerd **buffer overflow**
  - Exploiting sendmail's DEBUG mode (not a bug!)
  - Infected approximately 6,000 machine
    - 10% of computers connected to the Internet
  - Cost ~ $10 million in downtime and cleanup

Notorious Worms (2)

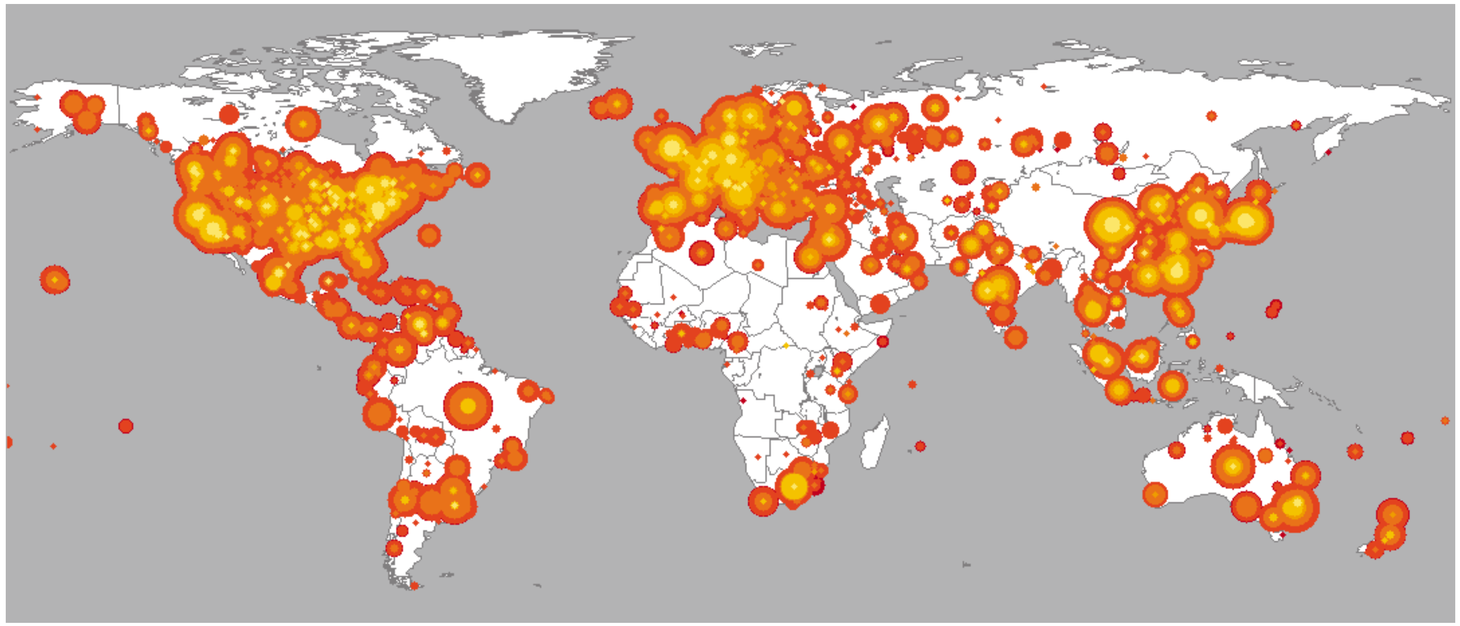
- ILOVEYOU (2000): email worm
  - Propagation through email attachment
  - Scans the contacts and sends an email to everyone
  - Estimated to have caused $5.5–8.7 billion in damages and cost US$15 billion for removal



Un e-mail con el virus ILOVEYOU en todo su esplendor.

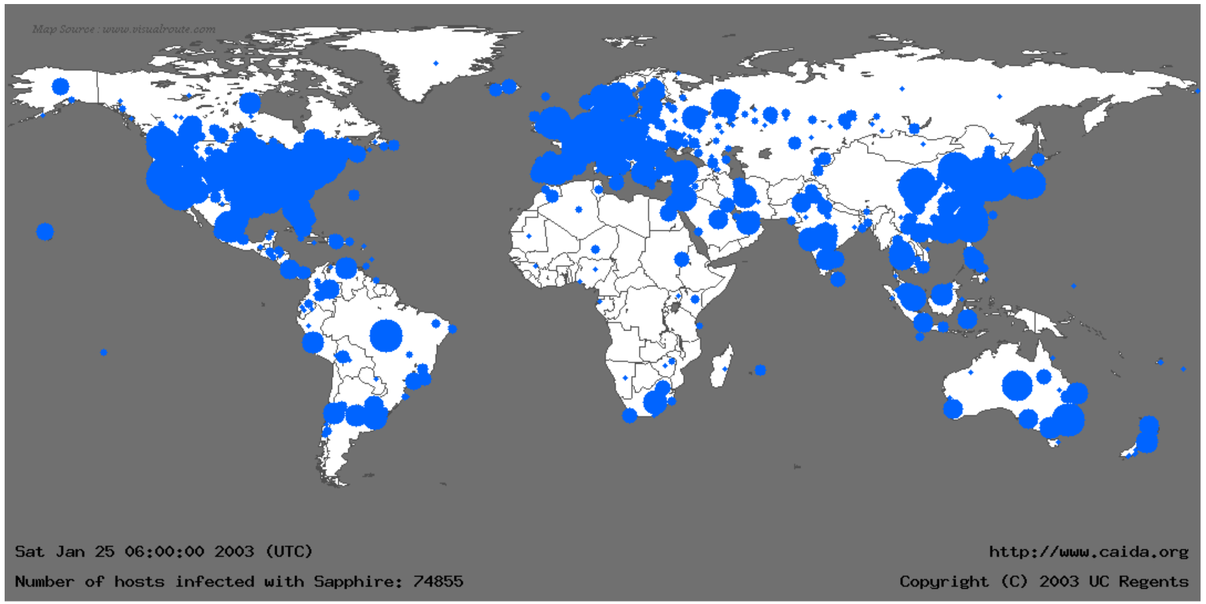
Notorious Worms (3)

- Code Red (2001): fast spreading
  - Exploits buffer overflow vulnerability inside MS IIS
  - Infected more than 359,000 computers in less than 14 hrs



Notorious Worms (4)

- Slammer (2003): fastest ever
  - Exploits buffer overflow vulnerability inside MS SQLServer
  - Infected more than 90 percent of vulnerable hosts within 10 mins



Notorious Worms (5)

- Stuxnet (2010): SCADA
  - Multi-mode spreading
    - Initially spreads via USB (virus-like)
    - Once inside a network, quickly spreads internally using Windows RPC
  - Geographically clustered
    - Iran: 59%; Indonesia: 18%; India: 8%

Notorious Worms (6)

- WannaCry (2017): ransomeware
  - Leaked NSA EternalBlue exploit (Windows SMB)

Notorious Worms (7)

- Mirai Botnet (2016)
  - Infects vulnerable IoT devices (IP cameras and home routers)
    - Common factory default usernames and passwords
  - Used to launch DDoS attacks and mine crypto currency

Botnet

- Botnet = malware that is **remotely controlled** by command and control (C&C) server
  - Collection of compromised hosts (infected in any ways)
  - Platform for many attacks
    - Spam forwarding (70% of all spam)
    - Click fraud / Phishing / Scaware (FakeAV) / Crypto coins
    - Distributed denial-of-service (DDoS)

Spyware

- Spyware = malware that collects your activities
  - Some people don't consider it as real malware (greyware)
    - Google? Facebook?
  - But with advances in machine learning, such activities matters a lot more!

Rootkit/Bootkit

- Rootkit/bootkit = malware that hides other malware
  - Hide the evidence of infection
  - Guarantees persistent
  - Usually executes at very low level (kernel, bootloader, firmware, etc)

Motivations

Click Trajectories: End-to-End Analysis of the Spam Value Chain

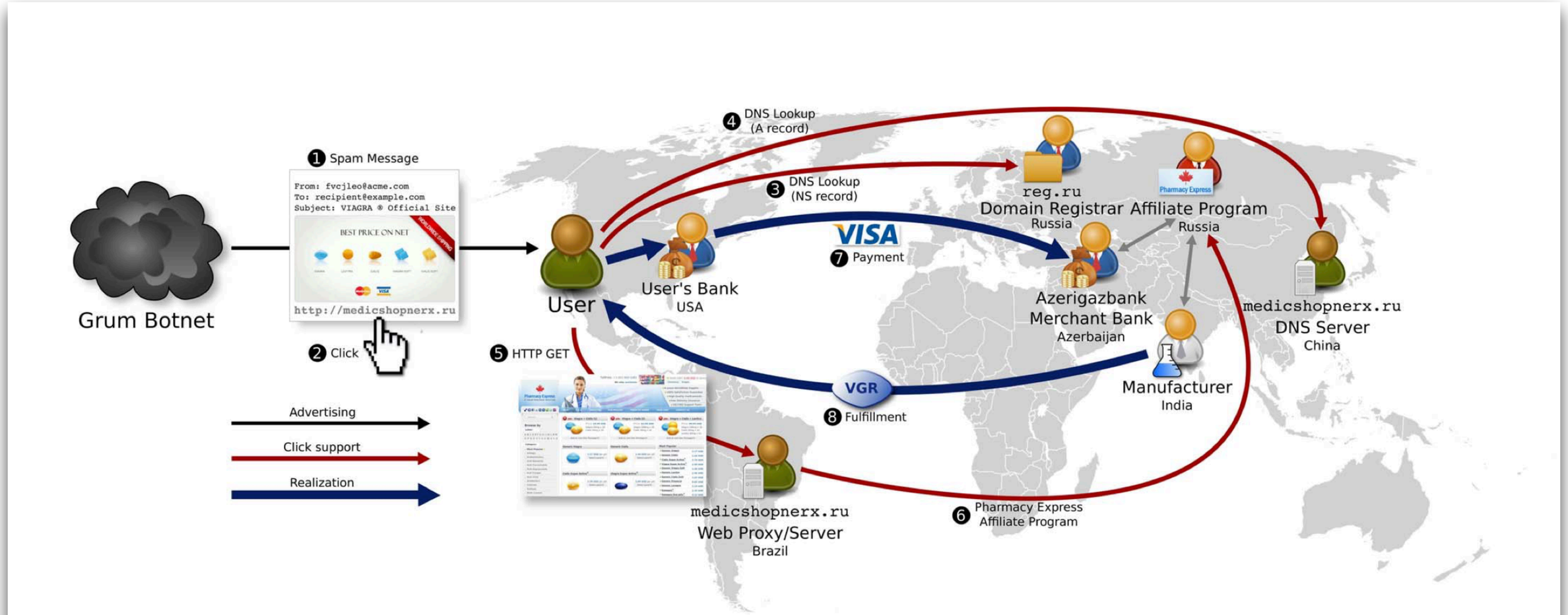


Figure 1: Infrastructure involved in a single URL's value chain, including advertisement, click support and realization steps.

Motivations

Apple’s walled-garden model

- Why malware is much more common on Android platforms than on iOS?
  - They have similar sandbox and permission systems
  - They all have app stores
  - They all have plenty of vulnerabilities

Malware Infection

How malware gets into your system?

- Virus: require human interaction
  - **Do not open suspicious files/attachments**
  - **Do not insert unknown USB/Disk**
  - **Do not insert your thumb drive into unknown computer**
- Worm & drive-by: exploit software vulnerabilities
  - **Patch your system as soon as possible**

Malware Infection (cont.)

How malware gets into your system?

- Trojan horse: disguise as something legitimate
  - **Download software from app store or trusted website**
  - **Do not use pirate software**
  - **Check integrity of the software**
- Social engineering: motivate you to do something dangerous
  - **Think twice**

Malware Detection

How to detect malware?

Idea #1: use signatures

How antivirus software works

- How does our immune system detect viruses? => signature-based detection
- Antivirus: look for bytes corresponding to the malware
  - Where to get the samples?
  - How to make sure each signature is unique/good?
  - Why effective? replicating nature of malware
- Drove development of multi-billion $$ AV industry
  - Limited but necessary

Antivirus

An interesting story ...

virustotal

VirusTotal is a free service that **analyzes suspicious files and URLs** and facilitates the quick detection of viruses, worms, trojans, and all kinds of malware.

SHA256: 71d1723d1269abef2b78d6c46390452058c047bc44949bad8f493446f947c8bc

File name: qvodsetupls27.exe

Detection ratio: 41 / 46

Analysis date: 2013-04-11 11:56:27 UTC (3 days, 10 hours ago)

More details

Analysis | Additional information | Comments | Votes

Antivirus	Result	Update
Agnitum	Trojan.DR.Agent!AmUdZaEHJGw	20130410
AhnLab-V3	Dropper/Win32.Agent	20130410
AntiVir	DR/MicroJoiner.Gen	20130411
Antiy-AVL	-	20130411
Avast	Win32:Microjoin-CD [Trj]	20130411
AVG	Dropper.Tiny.I	20130411
BitDefender	Trojan.Crypt.CG	20130411

The Arm Race

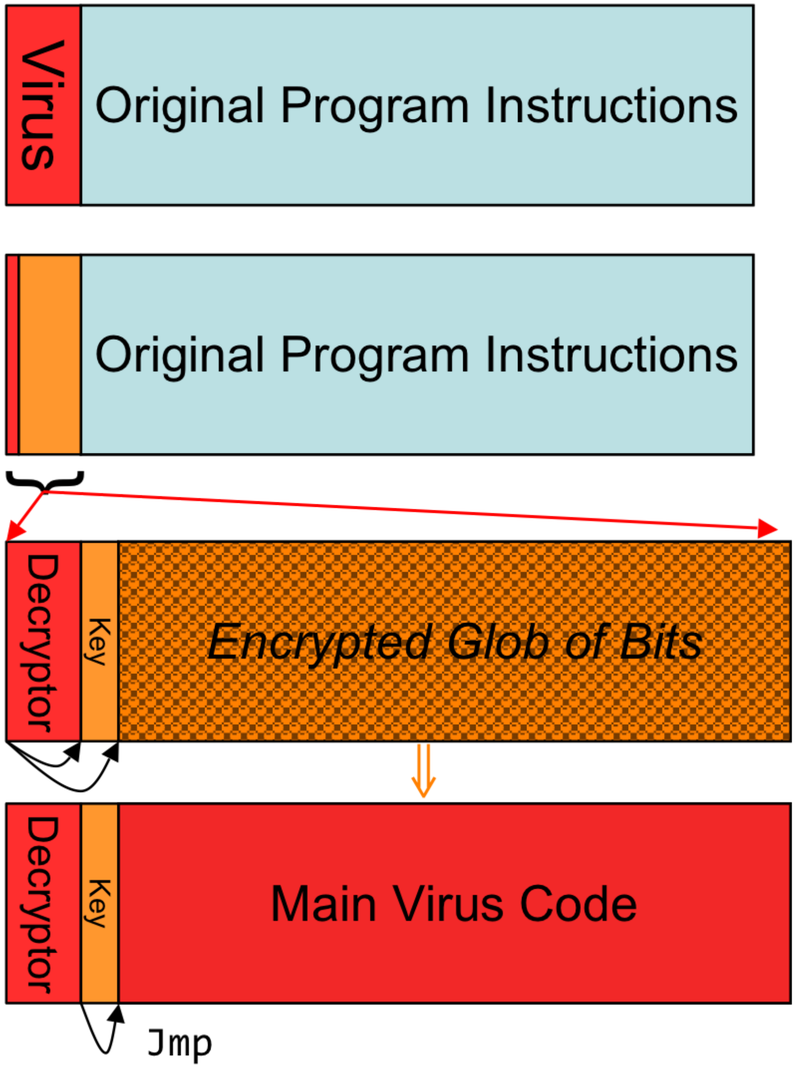
- If you are a virus writer, what would you do to make sure your effort does not get "wasted" by a signature from the AV industry?
- How do viruses evade the detection of our immune system?

Polymorphic Code

Malware fighting back

- Idea: change the appearance of the code every time it propagates
- How? **Encryption**!
  - Encodes the message so that the adversary cannot recover its original content without knowing the secret
- Obfuscation (packing)
  - Weak (but simple/fast) crypto algorithm works fine too
  - Strong crypto algorithm: use random key / initial padding

Unpacking



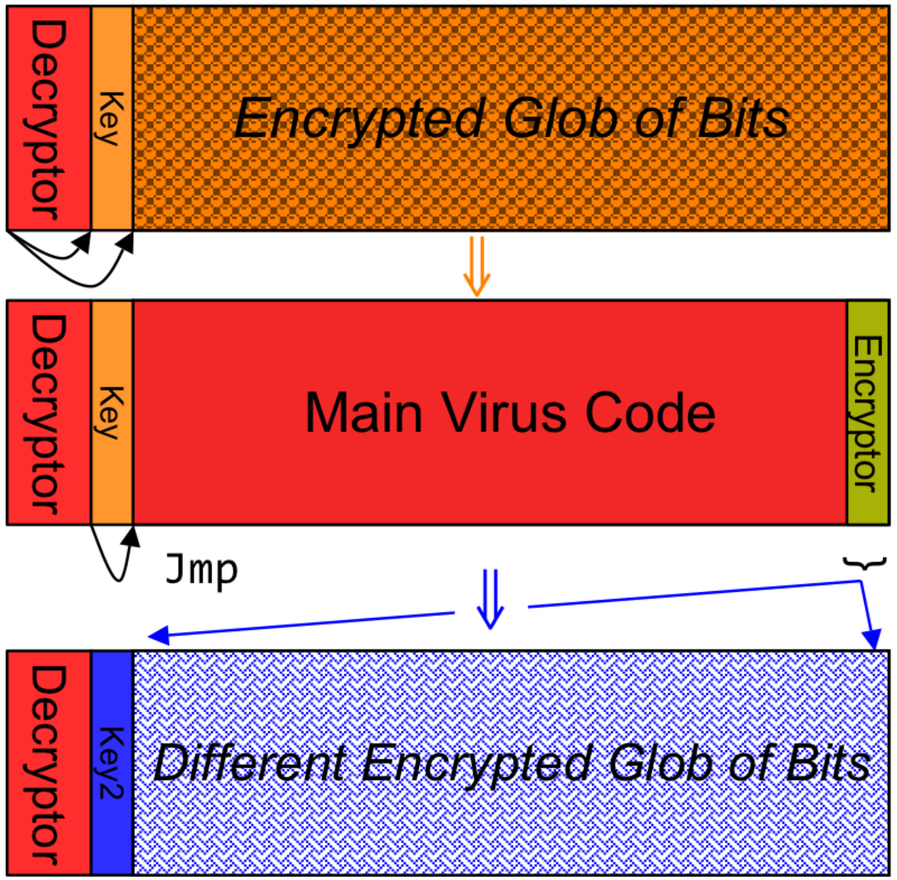
Instead of this …

Virus has *this* initial structure

When executed, decryptor applies key to decrypt the glob …

… and jumps to the decrypted code once stored in memory

Polymorphic Propagation



Once running, virus uses an *encryptor* with a new key to propagate

New virus instance bears little resemblance to original

Detecting Polymorphic Malware

- How would you detect a polymorphic malware?
- Idea #1: detect the unpacker/decryptor
  - False positives: less code to match, legitimate software also use obfuscation to protect IP
- Idea #2: decrypt and detect
  - Speculative runs the software for a while and scan memory
    - But for how long?

The Arm Race

- How to evade auto unpackers or memory scanners?

Metamorphic Code

- Idea: change the **syntax** of the code every time it propagates
- How? **Code rewriter**
  - Renumber registers
  - Change order of conditional code
  - Reorder operations not dependent on one another
  - Replace one low-level algorithm with another
  - Junk dead code
  - etc

Metamorphic Code

Hunting for Metamorphic, Szor & Ferrie, Symantec Corp., Virus Bulletin Conference, 2001

Detecting Metamorphic Malware

- How would you detect a metamorphic malware?
- Idea: focus on **semantics** (behaviors) instead of appearance
  - Create signatures for **malicious behaviors** (e.g., syscall-based)
  - Monitor dynamic behaviors of a process and detect malicious ones

Malicious Behavior Modeling

Effective and Efficient Malware Detection at the End Host

- How to model malicious behaviors?
- How to check malicious behaviors?

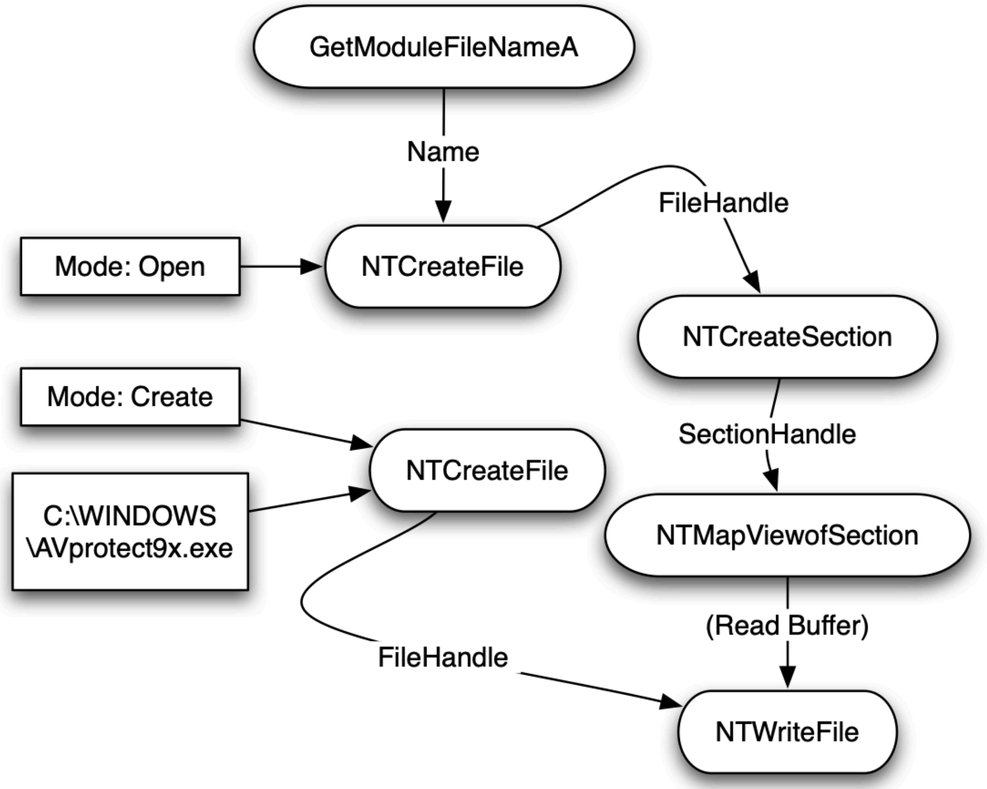


Figure 1: Partial behavior graph for Netsky.

The Arm Race

- Virus-writer countermeasures?
  - Anti dynamic analysis
  - VM/emulator/debugger detection, triggers, env binding, etc
- Metamorphic syscalls
- Rootkits

Summary

Host side detection

- Deciding whether a software is malicious or not in general, is not decidable
  - With theoretical proof (**the halting problem**)
- In practice, signature/black-list based approach has one big limitation
  - **Only detects known malware**
  - VT as an oracle
- What about white list approach, like on iOS
  - Much better but still limited

Notes on ML/DL

Is ML/DL a panacea?

- How does ML/DL work?
  - Features —> **What kind of features are critical to malicious behavior?**
- Later in the class
  - Outside the Closed World: On Using Machine Learning For Network Intrusion Detection (ToT)
  - Practical Evasion of a Learning-Based Classifier

Malware Mitigations

What to do if infected by malware?

Malware Removal

Host side mitigations

- Removal
- Quarantine
- Reinstall
- Persistent malware
  - Rootkit
  - Bootkit
  - Firmware malware

Botnet Take Down

Network level mitigations

- How to communicate with another machine (C&C servers)?
  - IP address => firewall blocking
  - DNS names => DNS sinkhole
  - Domain name generation => algorithm extraction
  - Decentralize (P2P malware) => poison

Ecosystem Take Down

Block the monetization channels

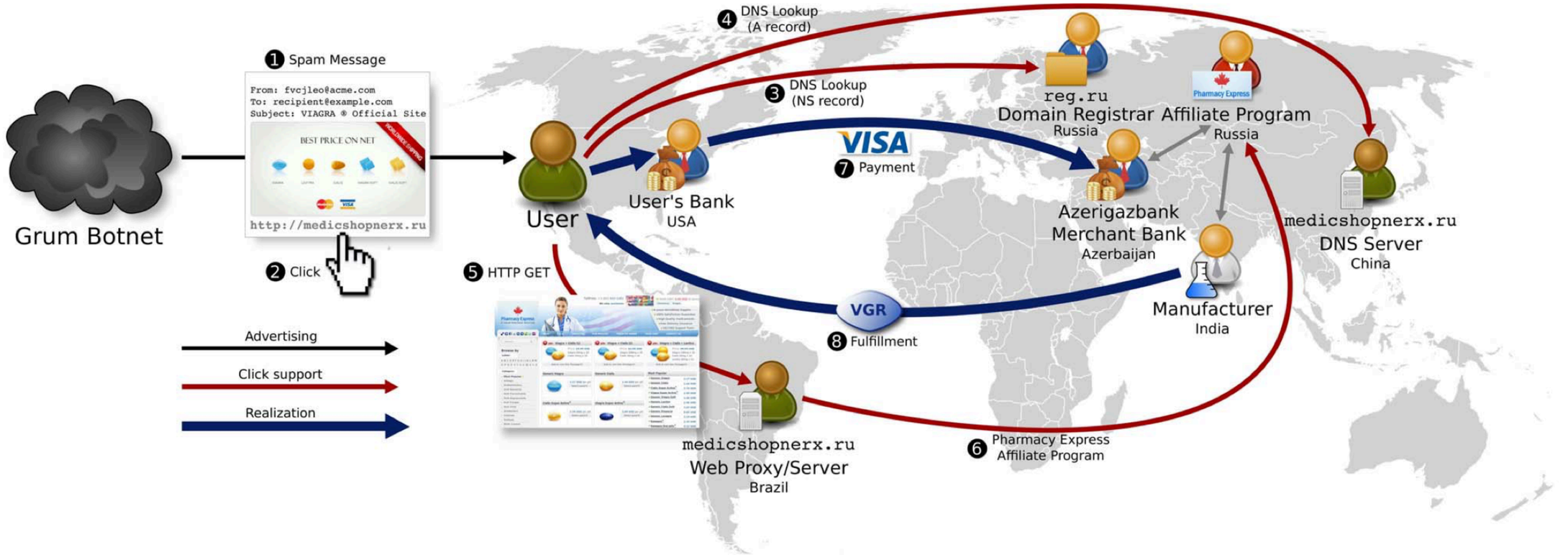
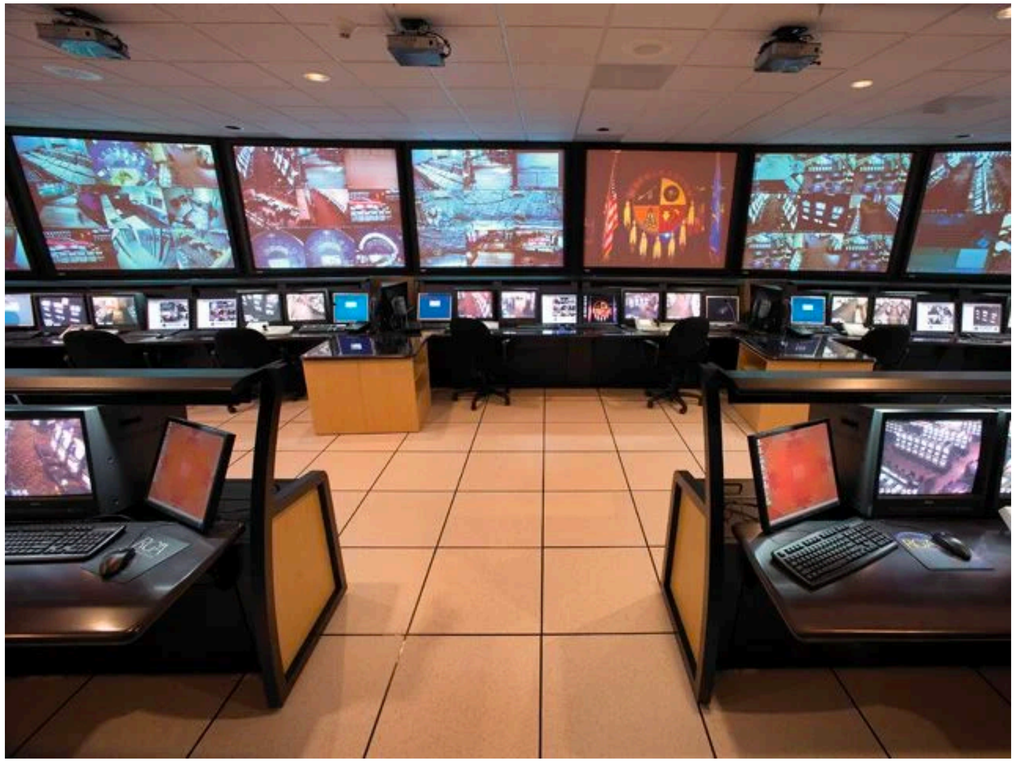


Figure 1: Infrastructure involved in a single URL's value chain, including advertisement, click support and realization steps.

Host Intrusion Detection

Intrusion Detection System

- Constant monitoring: looking for malicious behaviors or policy violations



Intrusion Detection SYSTEM

Major Components

- Monitors: collect data
- Policies/Signatures: define what is normal/malicious
- Policy engine: check if collected data comply policies/match signatures
- Reaction (optional)

Intrusion Detection System

Data sources

- Network IDS
  - Course-grained: checking packet headers
  - Fine-grained: checking payloads (a.k.a., deep packet inspection)
- Host IDS
  - Course-grained: OS level events
  - Fine-grained: program internal events

Signatures/Policies

- Signatures: similar to antivirus
  - Appearance-base signatures
  - Behavior-based signatures
- Policies
  - What’s allowed/not allowed/need to be logged/etc
    - e.g., accessing to sensitive configurations

Anomaly Detection

- Statistical-based anomaly detection: modeling what's normal
  - Usually ML/DL based
- Problem?
  - Unseen inputs/events/samples
  - False positives

Attacking IDS

- How would you do it?

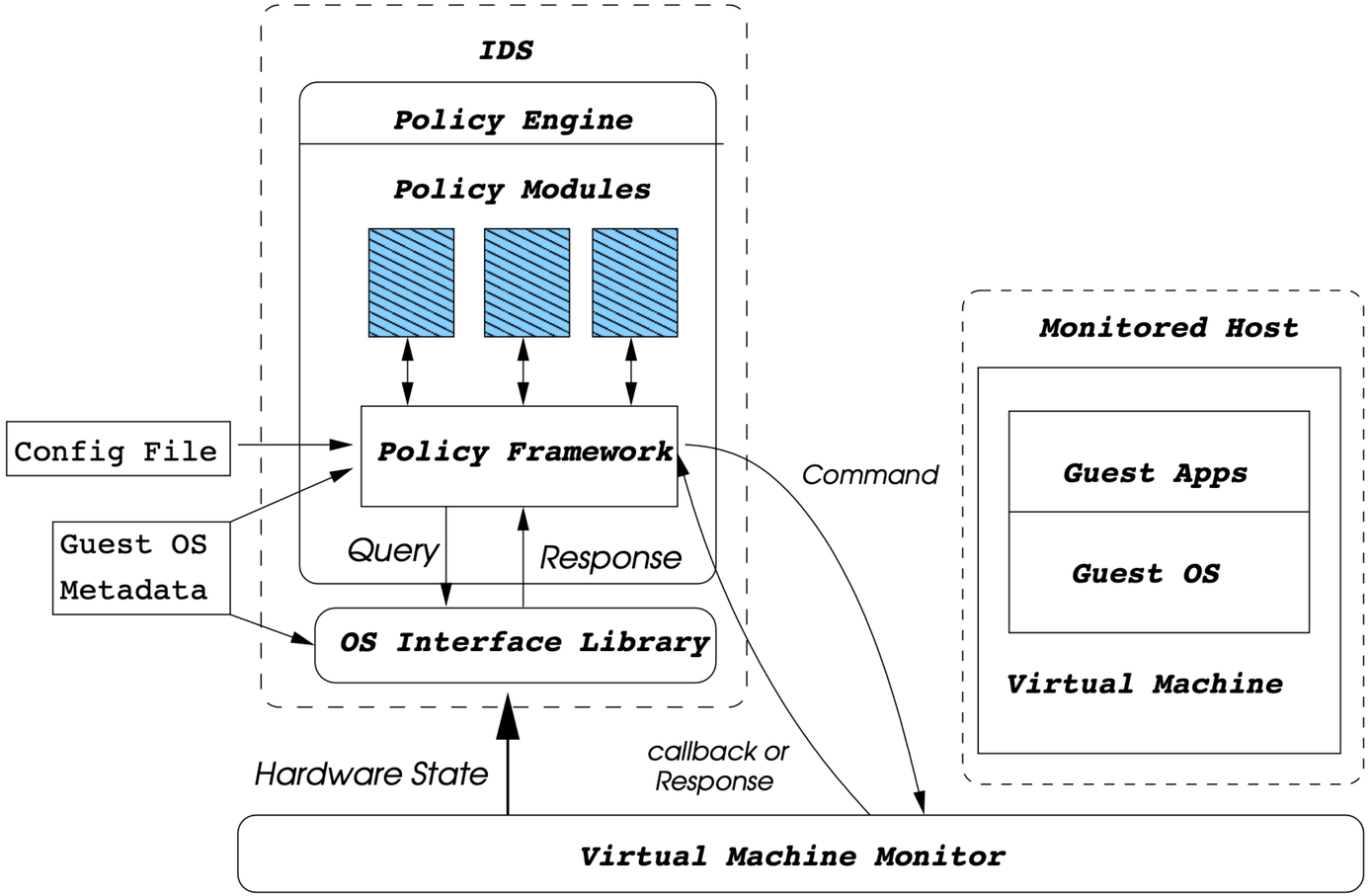
Livewire

A Virtual Machine Introspection Based Architecture for Intrusion Detection

- Motivations: why VMI?
- Challenges?
- Solutions?

Livewire

A Virtual Machine Introspection Based Architecture for Intrusion Detection



Livewire

A Virtual Machine Introspection Based Architecture for Intrusion Detection

- Example Policy Modules
  - Polling (scanning): lie detector, user program integrity, signature, raw socket
  - Event-driven (plant monitors): memory access, NIC access