

Malware: Viruses, Worms, & Botnets

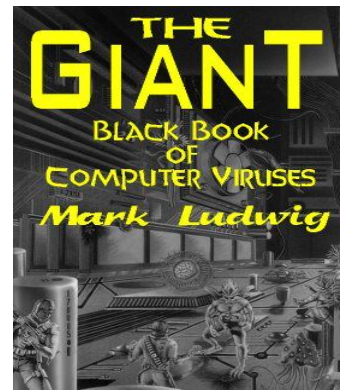
Slide credit: Vern Paxson

Malware That Propagates

- **Virus** = code that **propagates** (replicates) across systems with user intervention
- **Worm** = code that **self-propagates**/replicates across systems without requiring user intervention

The Problem of Viruses

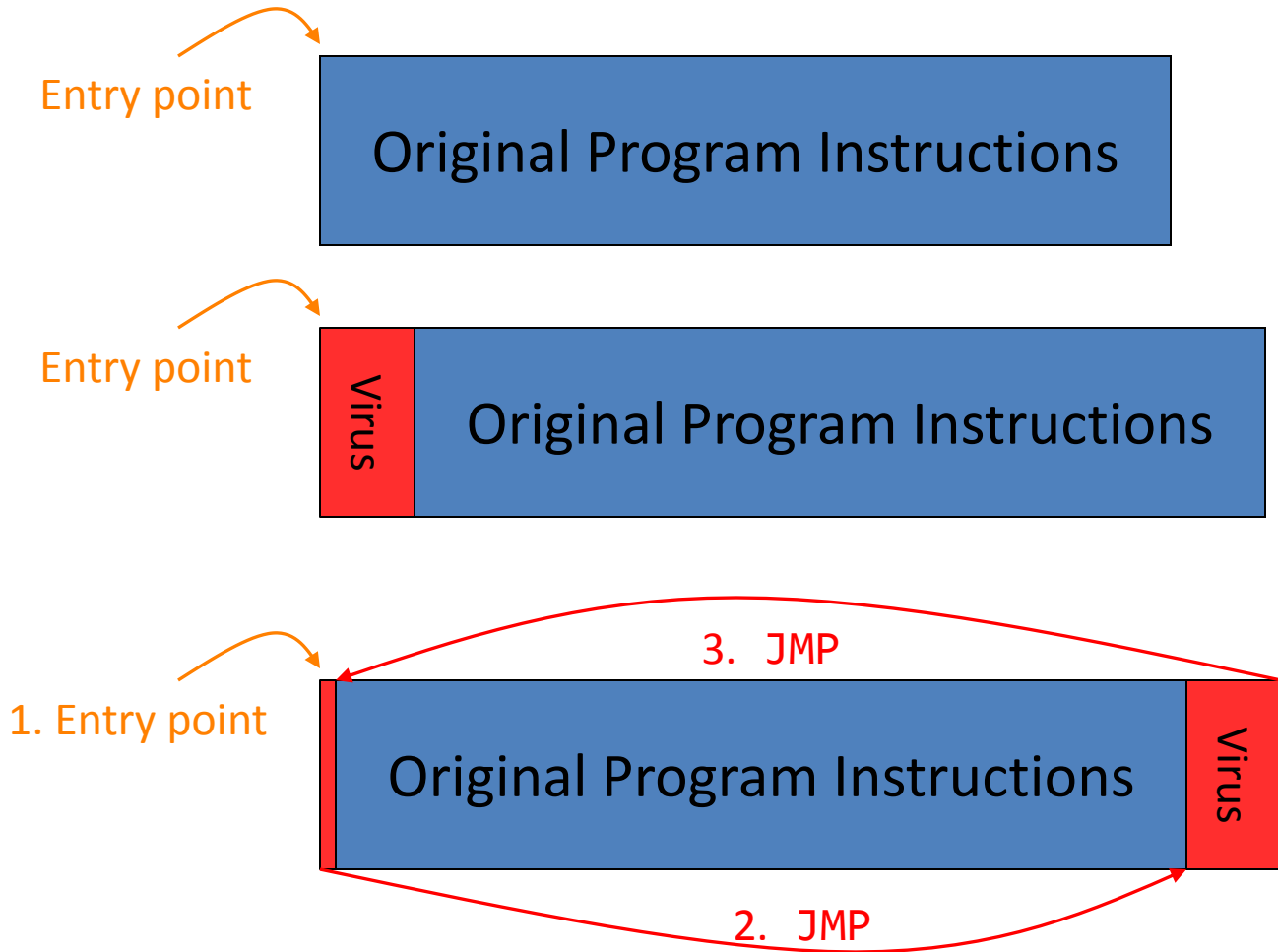
- Virus = code that **replicates**
 - Instances opportunistically create new addl. instances
 - Goal of replication: install code on additional systems
- Opportunistic = code will **eventually** execute
 - Generally due to **user action**
 - Running an app, booting their system, opening an attachment
- Separate notions for a virus: how it **propagates** vs. what else it does when executed (**payload**)
- General infection strategy: find some code lying around, alter it to include the virus
- Have been around for **decades** ...
 - ... resulting **arms race** has heavily influenced evolution of modern malware



Propagation

- When virus runs, it looks for an opportunity to infect additional systems
- One approach: look for USB-attached thumb drive, alter any executables it holds to include the virus
 - Strategy: if drive later attached to another system & altered executable runs, it locates and infects executables on new system's hard drive
- Or: when user sends email w/ attachment, virus **alters attachment** to add a copy of itself
 - Works for attachment types that include **programmability**
 - E.g., Word documents (macros), PDFs (Javascript)
 - Virus can also send out such email proactively, using user's address book + enticing subject (“**I Love You**”)

*autorun is
handy here!*



Original program instructions can be:

- Application the user runs
- Run-time library / routines resident in memory
- Disk blocks used to boot OS
- Autorun file on USB device
- ...

Many variants are possible, and of course can combine techniques

Payload

- Besides propagating, what else can the virus do when executing?
 - Pretty much *anything*
 - Payload is *decoupled* from propagation
 - Only subject to permissions under which it runs
- Examples:
 - Brag or exhort (pop up a message)
 - Trash files (just to be nasty)
 - Damage hardware (!)
 - Keylogging
 - Encrypt files
 - “Ransomware”
- Possibly delayed until condition occurs
 - “time bomb” / “logic bomb”

Detecting Viruses

- Signature-based detection
 - Look for bytes corresponding to injected virus code
 - High utility due to **replicating nature**
 - If you capture a virus V on one system, by its nature the virus will be trying to infect *many other systems*
 - Can protect those other systems by installing recognizer for V
- Drove development of **multi-billion \$\$ AV industry** (AV = “antivirus”)
 - So many **endemic** viruses that detecting well-known ones becomes a “*checklist item*” for security audits
- Using signature-based detection also has de facto utility for **marketing**
 - Companies compete on number of signatures ...
 - ... rather than their quality (harder for customer to assess)



Virustotal is a **service that analyzes suspicious files and URLs** and facilitates the quick detection of viruses, worms, trojans, and all kinds of malware detected by antivirus engines. [More information...](#)

1 VT Community user(s) with a total of 1 reputation credit(s) say(s) this sample is goodware. 6 VT Community user(s) with a total of 8 reputation credit(s) say(s) this sample is malware.

File name: **4.doc**
Submission date: **2011-04-19 07:19:30 (UTC)**
Current status: **finished**
Result: **27 /42 (64.3%)**

VT Community



malware

Safety score: 11.1%

[Compact](#)

[Print results](#)

Antivirus	Version	Last Update	Result
AhnLab-V3	2011.04.19.01	2011.04.19	Dropper/Cve-2011-0611
AntiVir	7.11.6.177	2011.04.19	EXP/CVE-2011-0611
Antiy-AVL	2.0.3.7	2011.04.18	Exploit/SWF.CVE-2011-0611
Avast	4.8.1351.0	2011.04.18	SWF:CVE-2011-0609-C
Avast5	5.0.677.0	2011.04.18	SWF:CVE-2011-0609-C
AVG	10.0.0.1190	2011.04.18	-
BitDefender	7.2	2011.04.19	-
CAT-QuickHeal	11.00	2011.04.19	-
ClamAV	0.97.0.0	2011.04.19	-
Commtouch	5.3.2.6	2011.04.19	MSWord/Dropper.B!Camelot
Comodo	8396	2011.04.19	UnclassifiedMalware
DrWeb	5.0.2.03300	2011.04.19	Exploit.Wordbo.12
Emsisoft	5.1.0.5	2011.04.19	Exploit.SWF.CVE-2011-0611!IK

Virus Writer / AV Arms Race

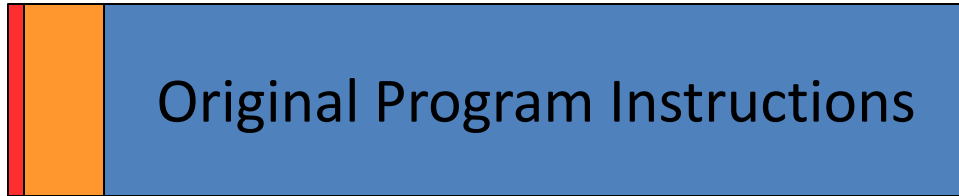
- If you are a virus writer and your beautiful new creations don't get very far because each time you write one, the AV companies quickly push out a signature for it
 - *What are you going to do?*
- Need to keep **changing** your viruses ...
 - ... or at least changing their appearance!
- Writing new viruses by hand takes a lot of effort
- How can you **mechanize** the creation of new instances of your viruses ...
 - ... such that whenever your virus propagates, what it injects as a copy of itself **looks different?**

Polymorphic Code

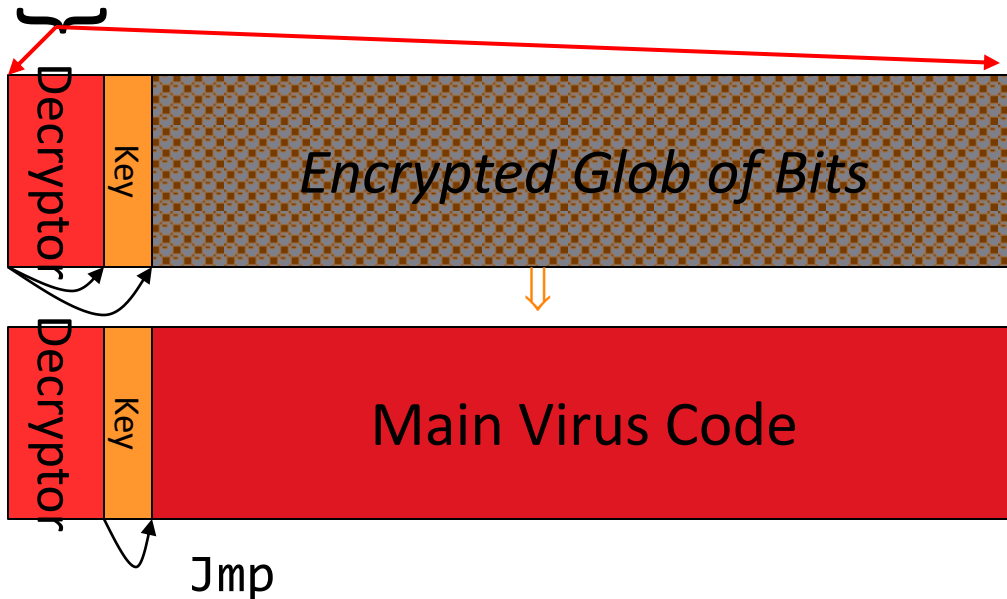
- We've already seen technology for creating a representation of some data that appears completely unrelated to the original data: **encryption!**
- Idea: every time your virus propagates, it inserts a **newly encrypted copy** of itself
 - Clearly, encryption needs to vary
 - Either by using a different key each time
 - Or by including some random initial padding (like an IV)
 - Note: weak (but simple/fast) crypto algorithm works fine
 - No need for truly strong encryption, just **obfuscation**
- When injected code runs, it decrypts itself to obtain the original functionality



Instead of this ...

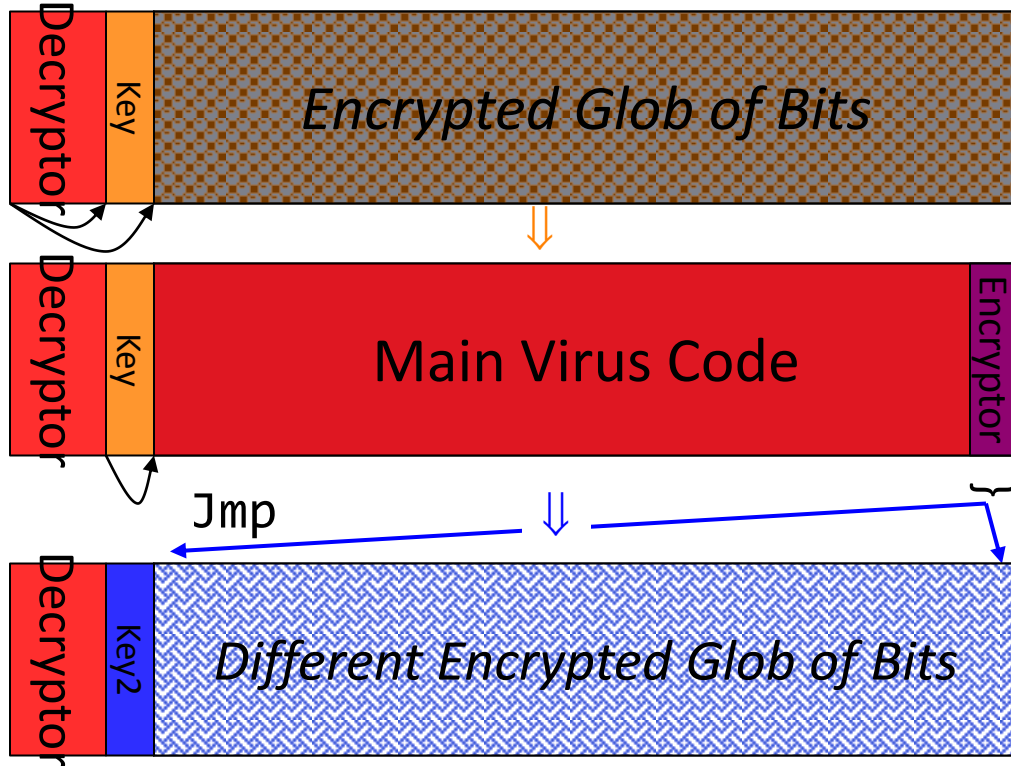


Virus has *this* **initial** structure



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

Arms Race: Polymorphic Code

- Given polymorphism, how might we then detect viruses?
- Idea #1: use narrow sig. that targets decryptor
 - Issues?
 - Less code to match against \Rightarrow more **false positives**
 - Virus writer spreads decryptor across existing code
- Idea #2: execute (or statically analyze) suspect code to see if it decrypts!
 - Issues?
 - Legitimate “*packers*” perform similar operations (decompression)
 - How long do you let the new code execute?
 - If decryptor only acts after lengthy legit execution, difficult to spot
- Virus-writer countermeasures?

Metamorphic Code

- Idea: every time the virus propagates, generate *semantically different* version of it!
 - Different semantics only at immediate level of execution; higher-level semantics remain same
- How could you do this?
- Include with the virus a **code rewriter**:
 - Inspects its own code, generates random variant, e.g.:
 - Renumber registers
 - Change order of conditional code
 - Reorder operations not dependent on one another
 - Replace one low-level algorithm with another
 - Remove some do-nothing **padding** and replace with different do-nothing padding
 - Can be very complex, legit code ... if it's never called!

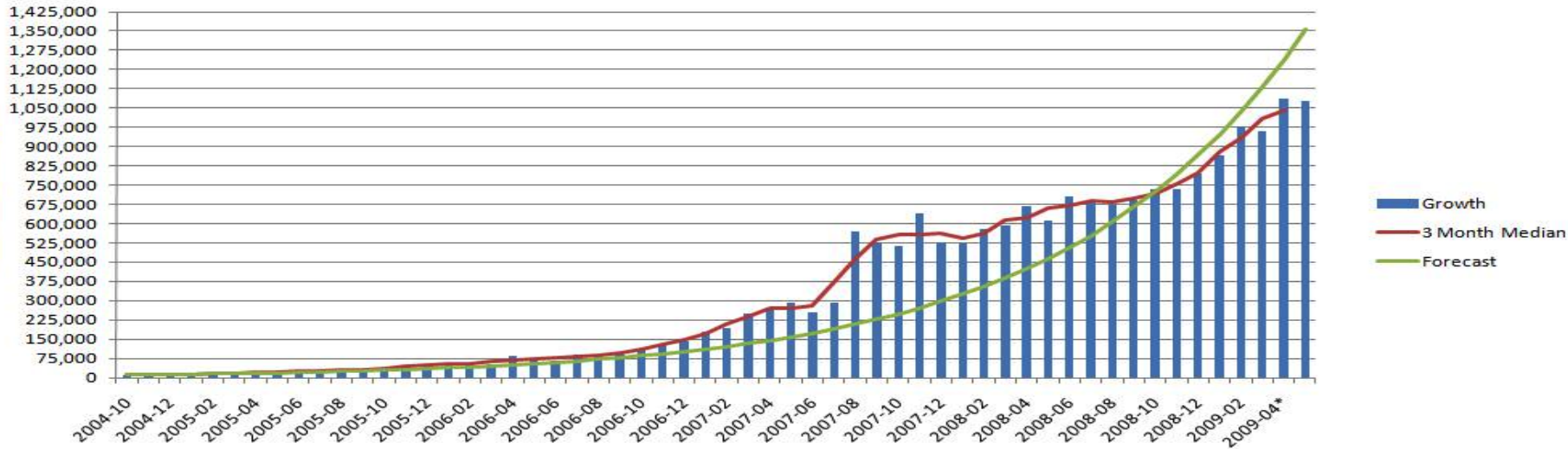
Detecting Metamorphic Viruses?

- Need to analyze execution **behavior**
 - Shift from **syntax** (*appearance* of instructions) to **semantics** (*effect* of instructions)
- Two stages: (1) AV company analyzes new virus to find behavioral signature, (2) AV software on end system analyzes suspect code to test for match to signature
- What countermeasures will the virus writer take?
 - **Delay analysis** by taking a long time to manifest behavior
 - Long time = await particular condition, or even simply clock time
 - Detect that execution occurs in an **analyzed environment** and if so behave differently
 - E.g., test whether running inside a debugger, or in a Virtual Machine
- Counter-countermeasure?
 - AV analysis looks for these tactics and skips over them
- Note: attacker has edge as AV products supply an *oracle*

How Much Malware Is Out There?

- A final consideration re polymorphism and metamorphism: presence can lead to **mis-counting** a single virus outbreak as instead reflecting 1000s of *seemingly different* viruses
 - Thus **take care** in interpreting vendor **statistics** on malware varieties
 - (Also note: public perception that many varieties exist is *in the vendors' own interest*)

New Unique Samples Added to AV-Test.org's Malware Collection



Infection Cleanup

- Once malware detected on a system, how do we get **rid** of it?
- May require restoring/repairing many files
 - This is part of what AV companies sell: per-specimen disinfection procedures
- What about if malware executed with **administrator privileges**?
 - “*nuke the entire site from orbit. It's the only way to be sure*”
 - i.e., **rebuild** system from **original media + data backups** - Aliens
- If we have complete source code for system, we could rebuild from that instead, right?