Viruses & Worms

Thanks to Prof. Vern Paxson for these slides

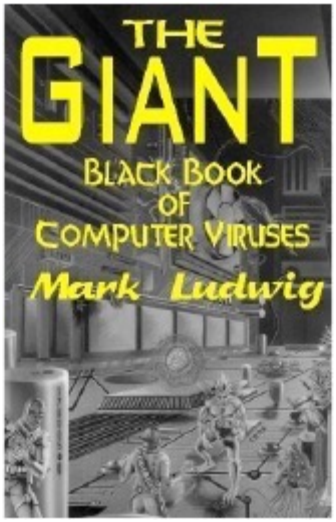
Malware That Propagates

- Virus = code that propagates (replicates) across systems by arranging to have itself eventually executed
  - Generally infects by altering stored code

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

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

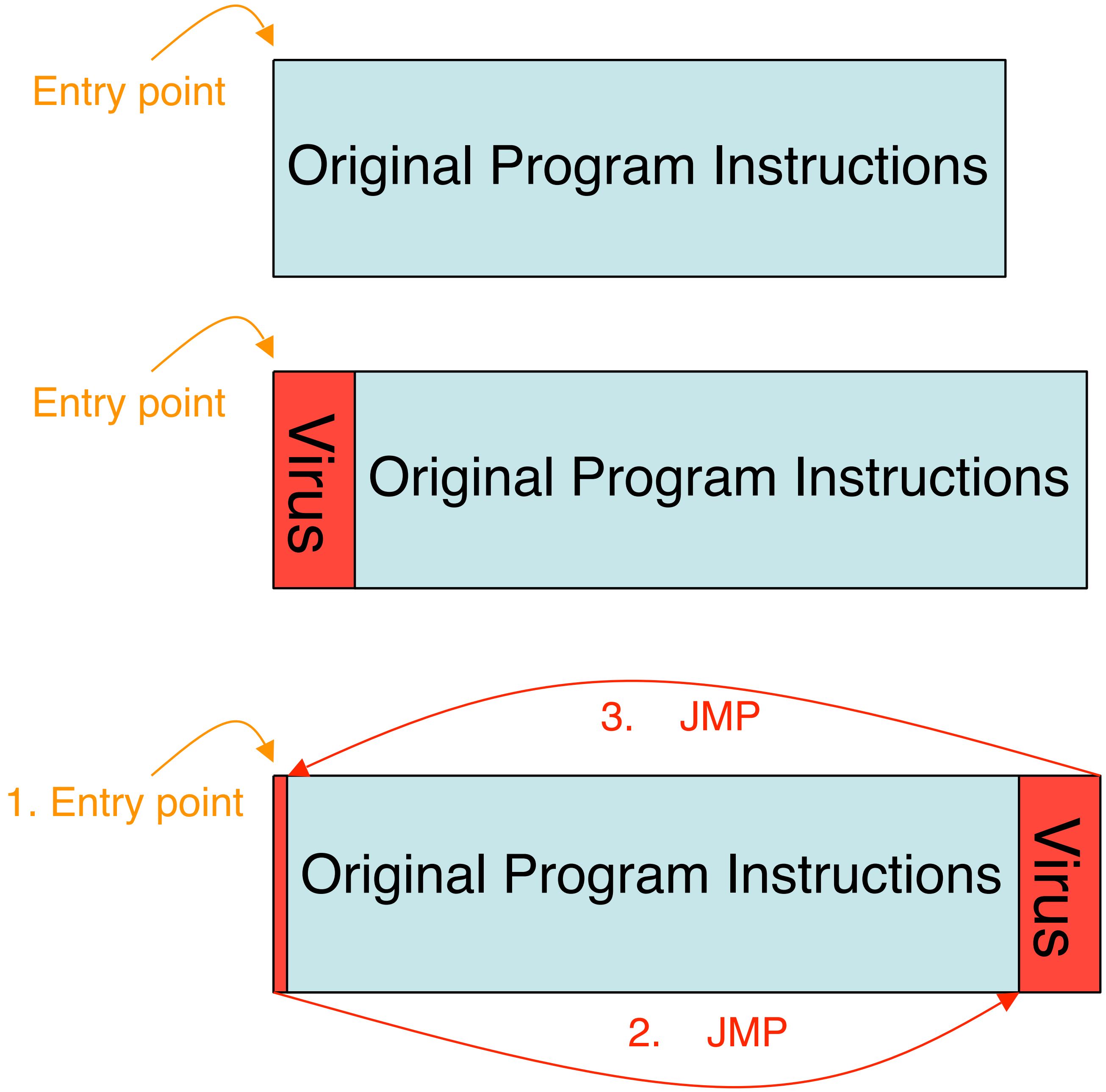


Propagation

- 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

***autorun** is handy here!*

- 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")


Entry point
Original Program Instructions
Entry point
Virus
Original Program Instructions
3. JMP
1. Entry point
Original Program Instructions
Virus
2. JMP

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
- …

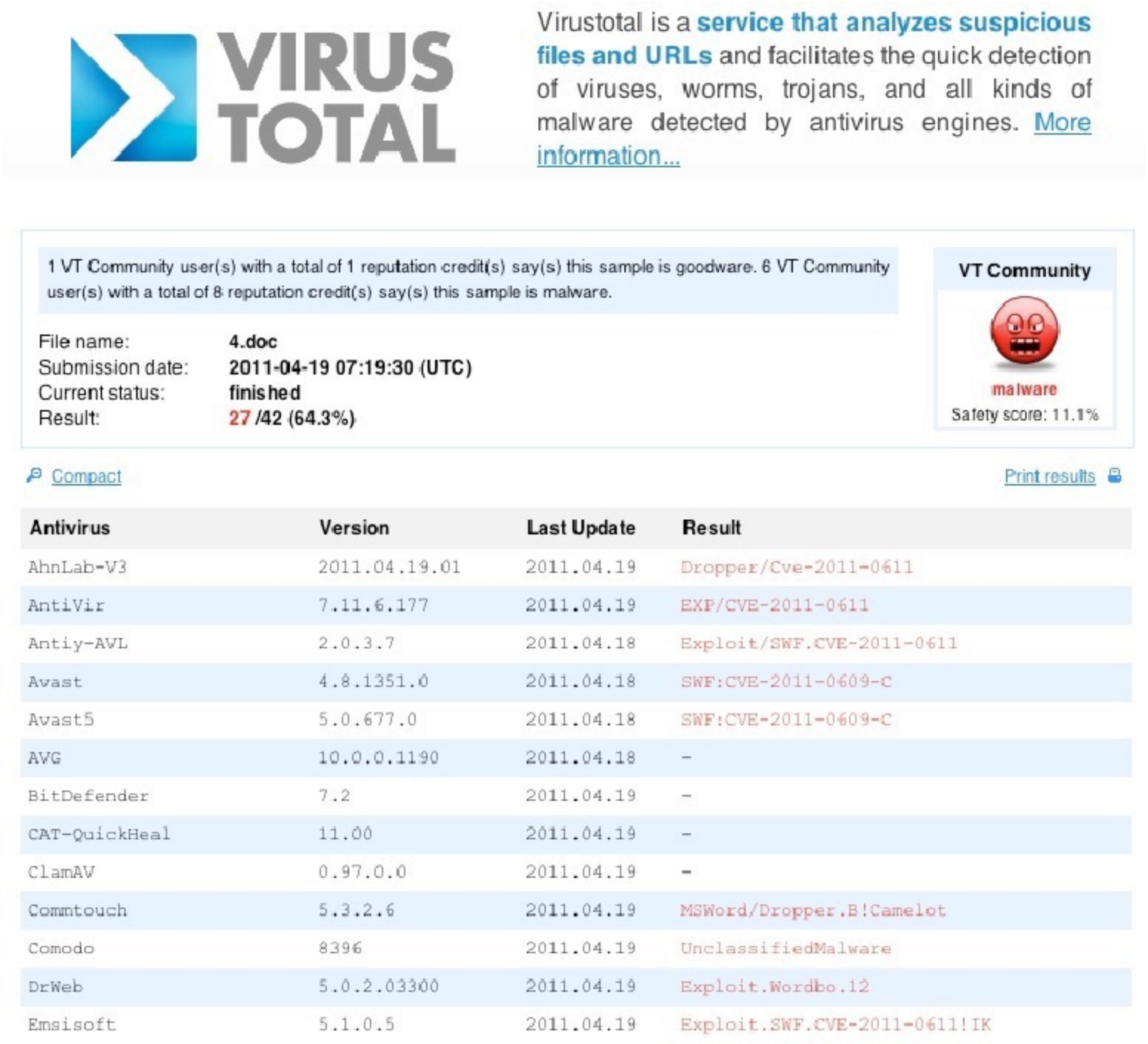
Many variants are possible, and of course can combine techniques

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 (glib) 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%

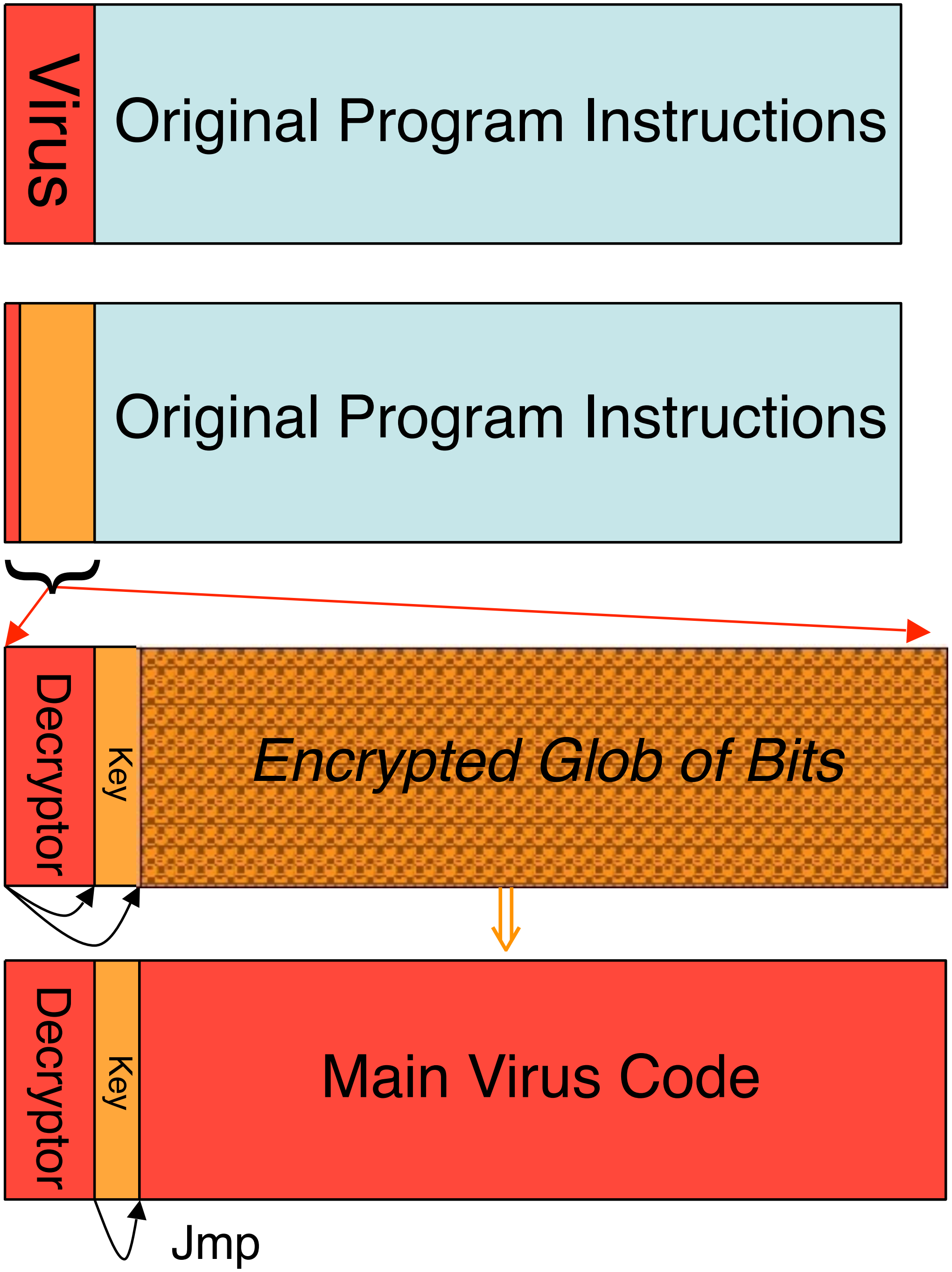
Antivirus	Version	Last Update	Result
AhnLab-V3	2011.04.19.01	2011.04.19	Dropper/Cve-2011-0611
AntiVir	7.11.6.177	2011.04.19	EXP/CVE-2011-0611
Antiy-AVL	2.0.3.7	2011.04.18	Exploit/SWF.CVE-2011-0611
Avast	4.8.1351.0	2011.04.18	SWF:CVE-2011-0609-C
Avast5	5.0.677.0	2011.04.18	SWF:CVE-2011-0609-C
AVG	10.0.0.1190	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



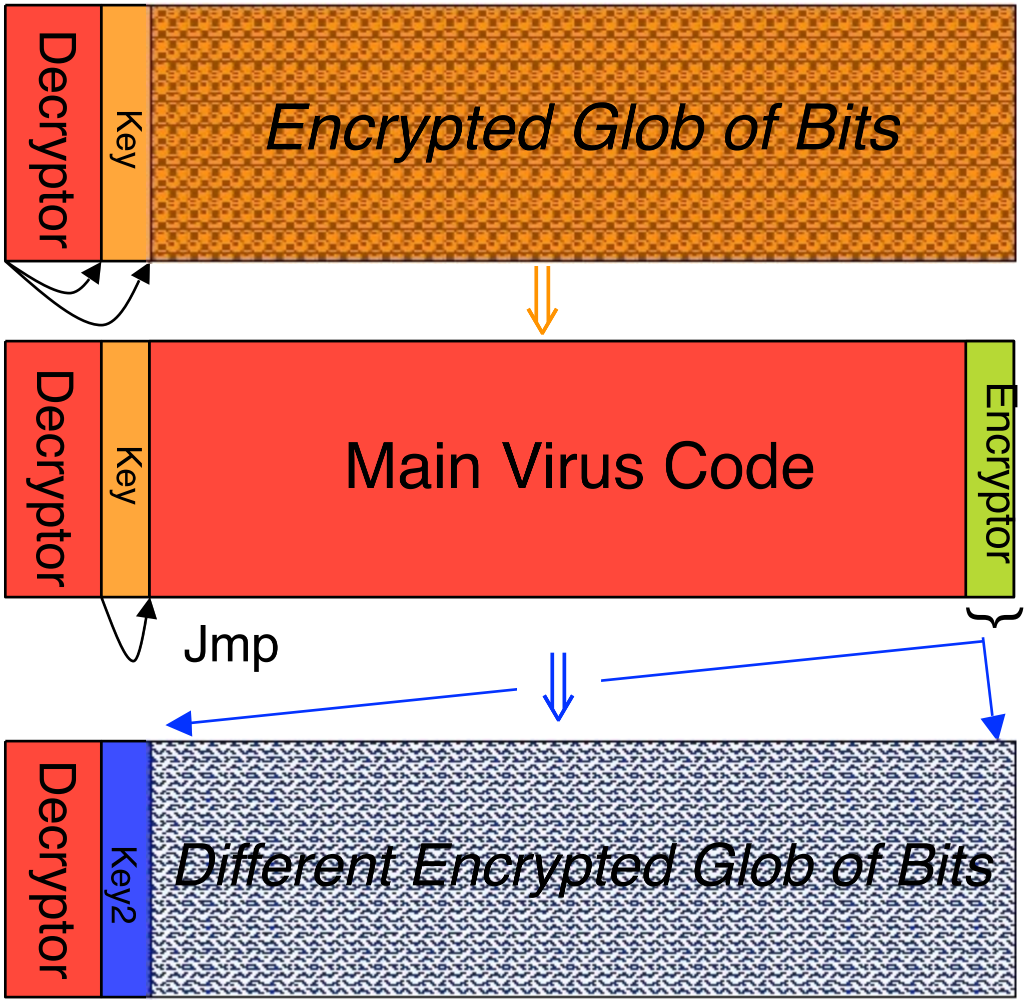
Instead of this …

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

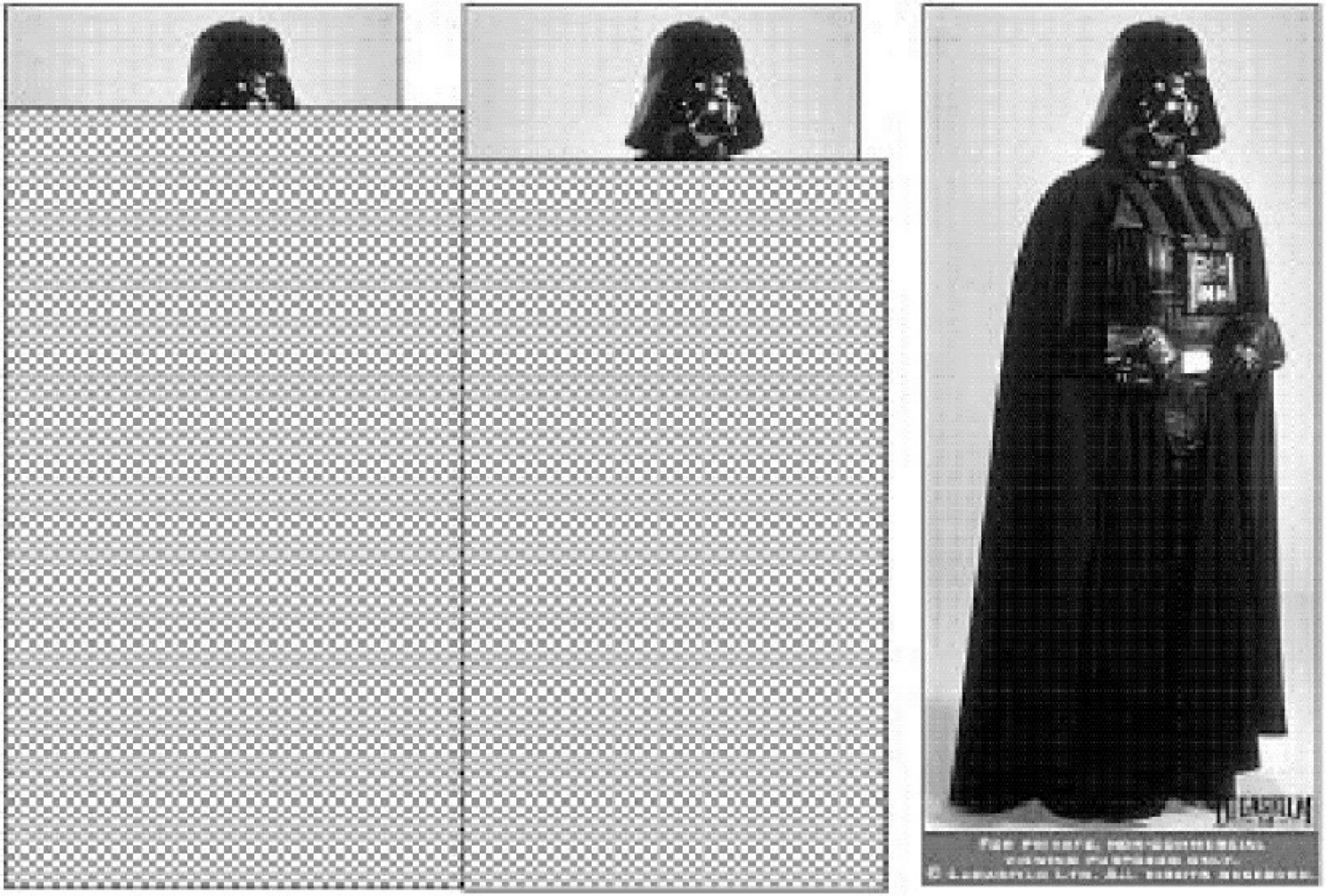
- Given polymorphism, how might we then detect viruses?
- Idea #1: use narrow sig. that targets decryptor
  - Issues?
    - Less code to match against ⇒ more false positives
    - Virus writer spreads decryptor across existing code
- Idea #2: execute (or statically analyze) suspect code to see if it 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 or has no important effect!

Polymorphic Code In Action



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

Metamorphic Code In Action

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

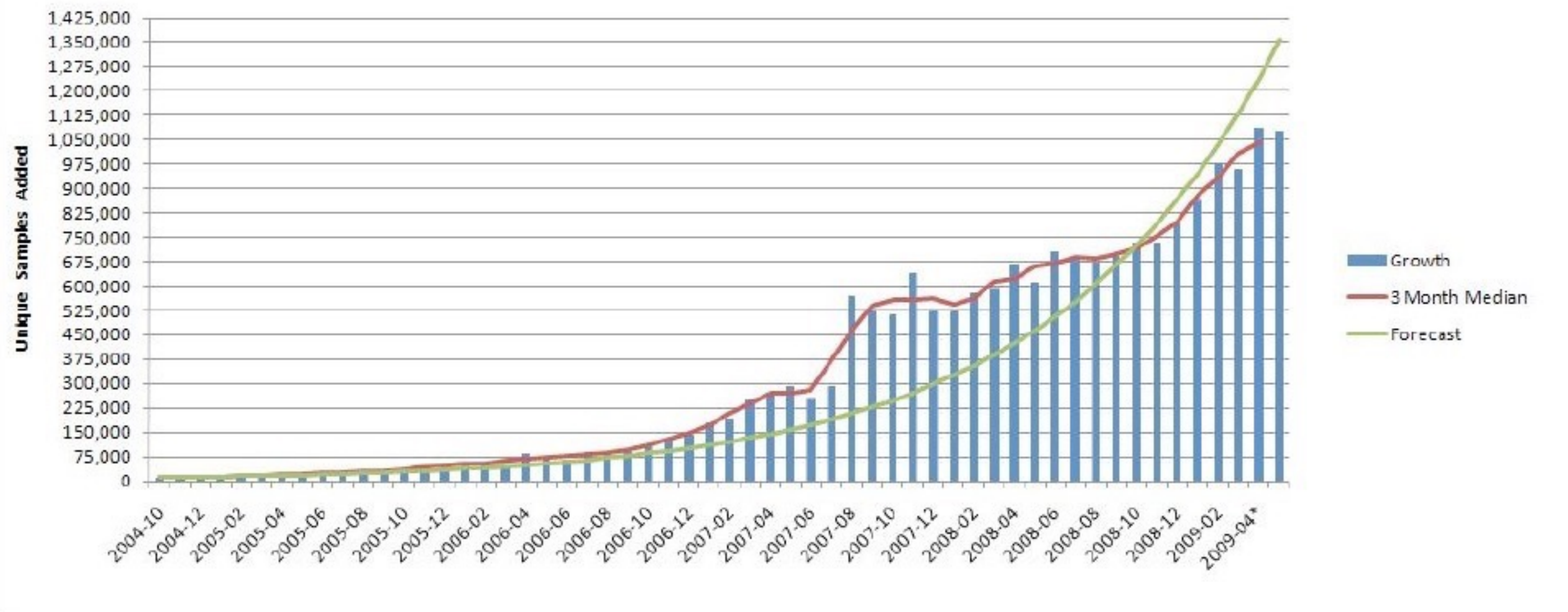
Detecting Metamorphic Viruses?

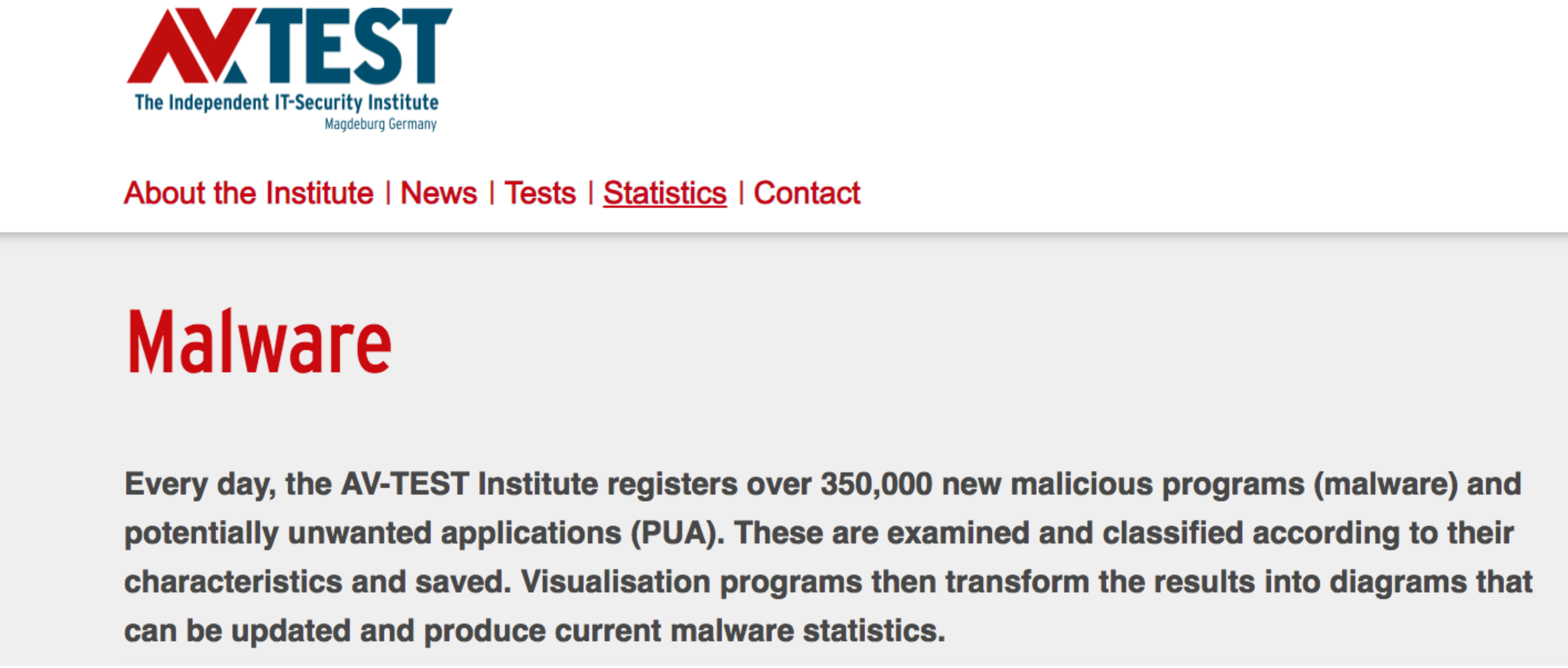
- Need to analyze execution behavior
  - Shift from syntax (*appearance* of instructions) to semantics (*effect* of instructions)
- Two stages: (1) AV company analyzes new virus to find behaviorial signature, (2) AV software on end system analyzes suspect code to test for match to signature
- What countermeasures 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 malcode 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





AV-TEST

The Independent IT-Security Institute

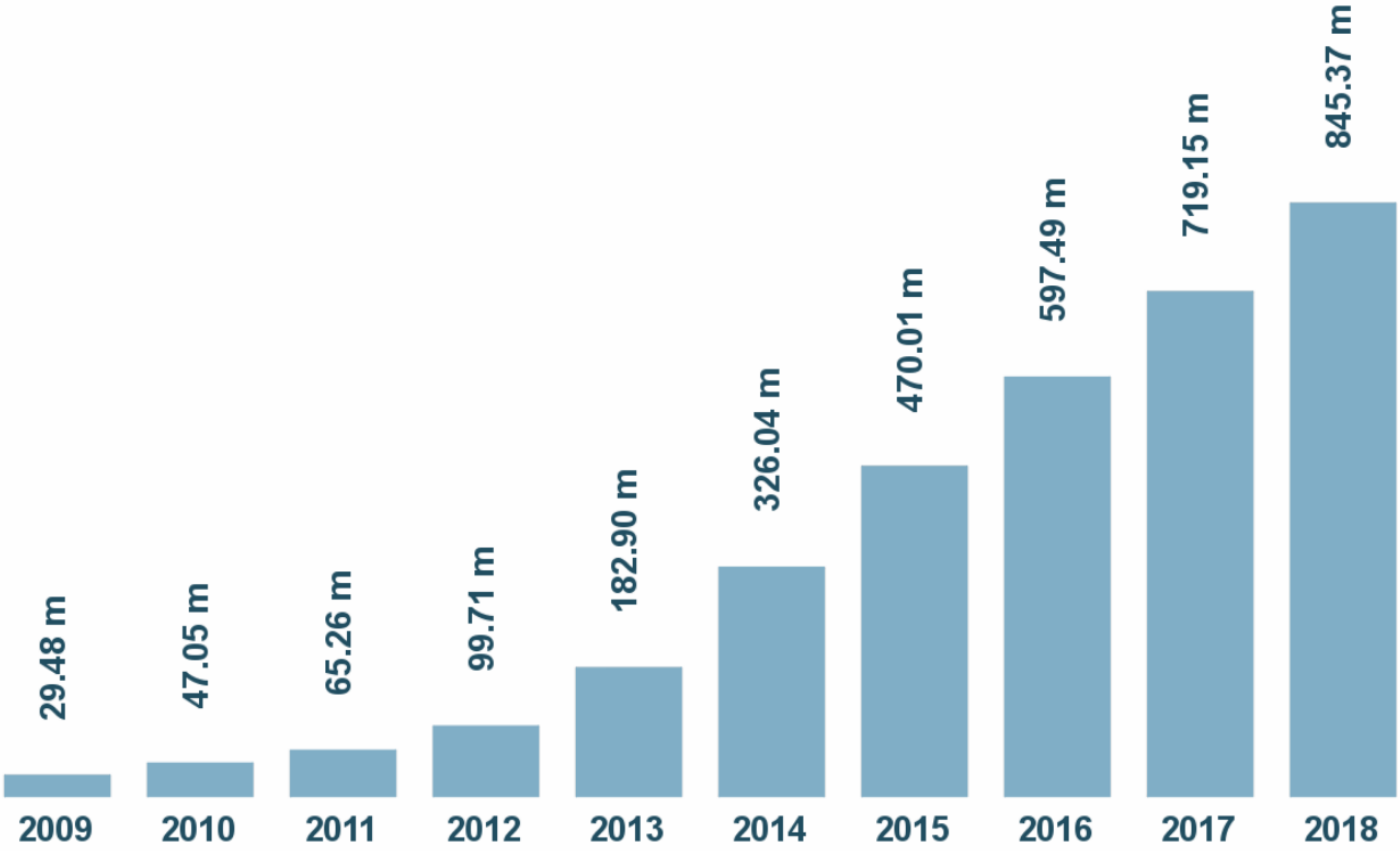
Magdeburg Germany

Malware

Every day, the AV-TEST Institute registers over 350,000 new malicious programs (malware) and potentially unwanted applications (PUA). These are examined and classified according to their characteristics and saved. Visualisation programs then transform the results into diagrams that can be updated and produce current malware statistics.

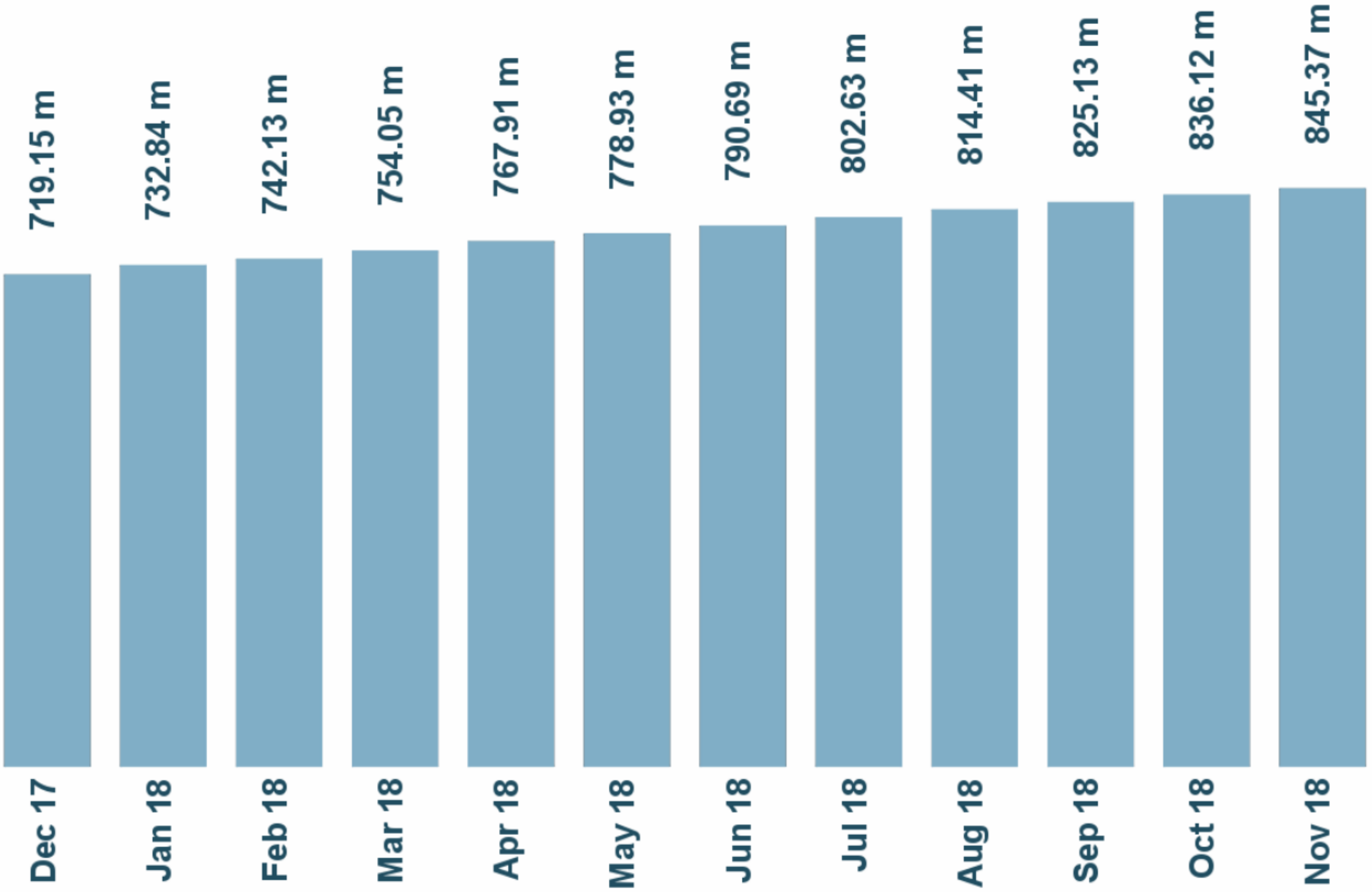
Total malware

AV-Test.org malware statistics



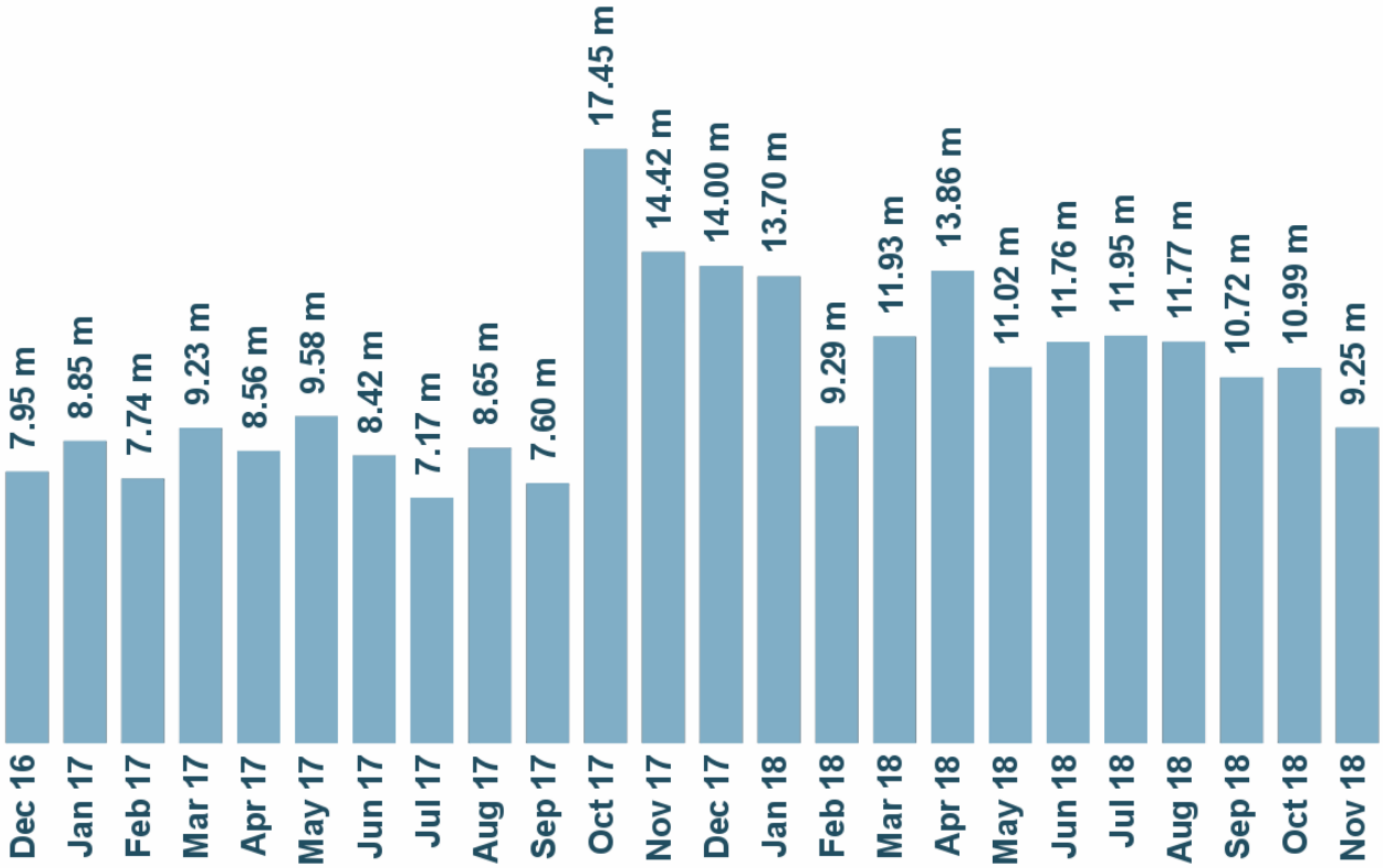
AV-Test.org malware statistics

Total malware



New malware

AV-Test.org malware statistics



Infection Cleanup

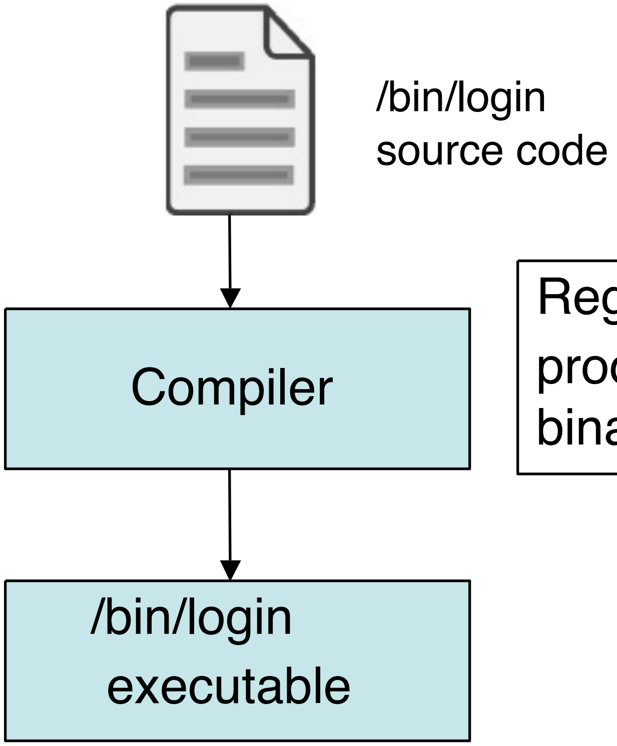
- Once malware detected on a system, how do we get rid of it?
- May require restoring/repairing many files
  - This is part of what AV companies sell: per-specimen disinfection procedures
- What about if malware executed with adminstrator privileges?
  - "*nuke the entire site from orbit. It's the only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

The Perils of Rebuilding From Source

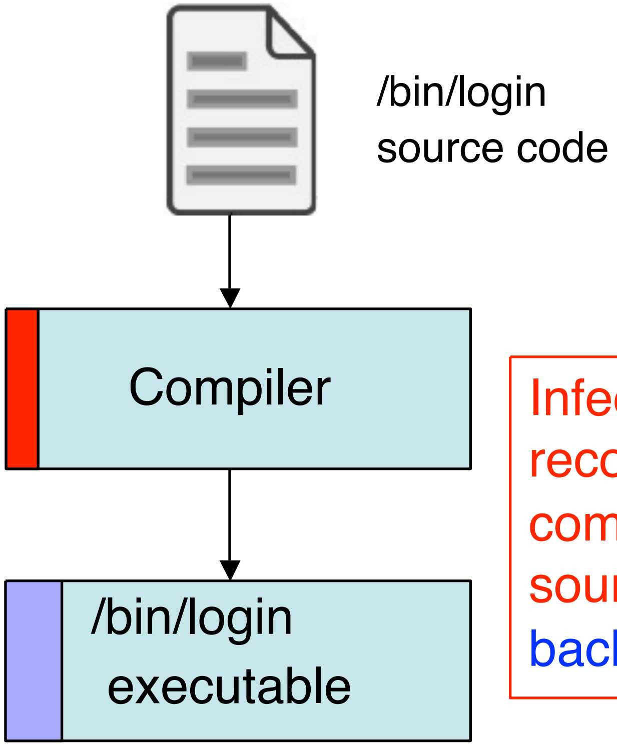
- If we have complete source code for system, we could rebuild from that instead, right?

- Suppose forensic analysis shows that virus introduced a backdoor in /bin/login executable
  - (Note: this threat isn't specific to viruses; applies to any malware)

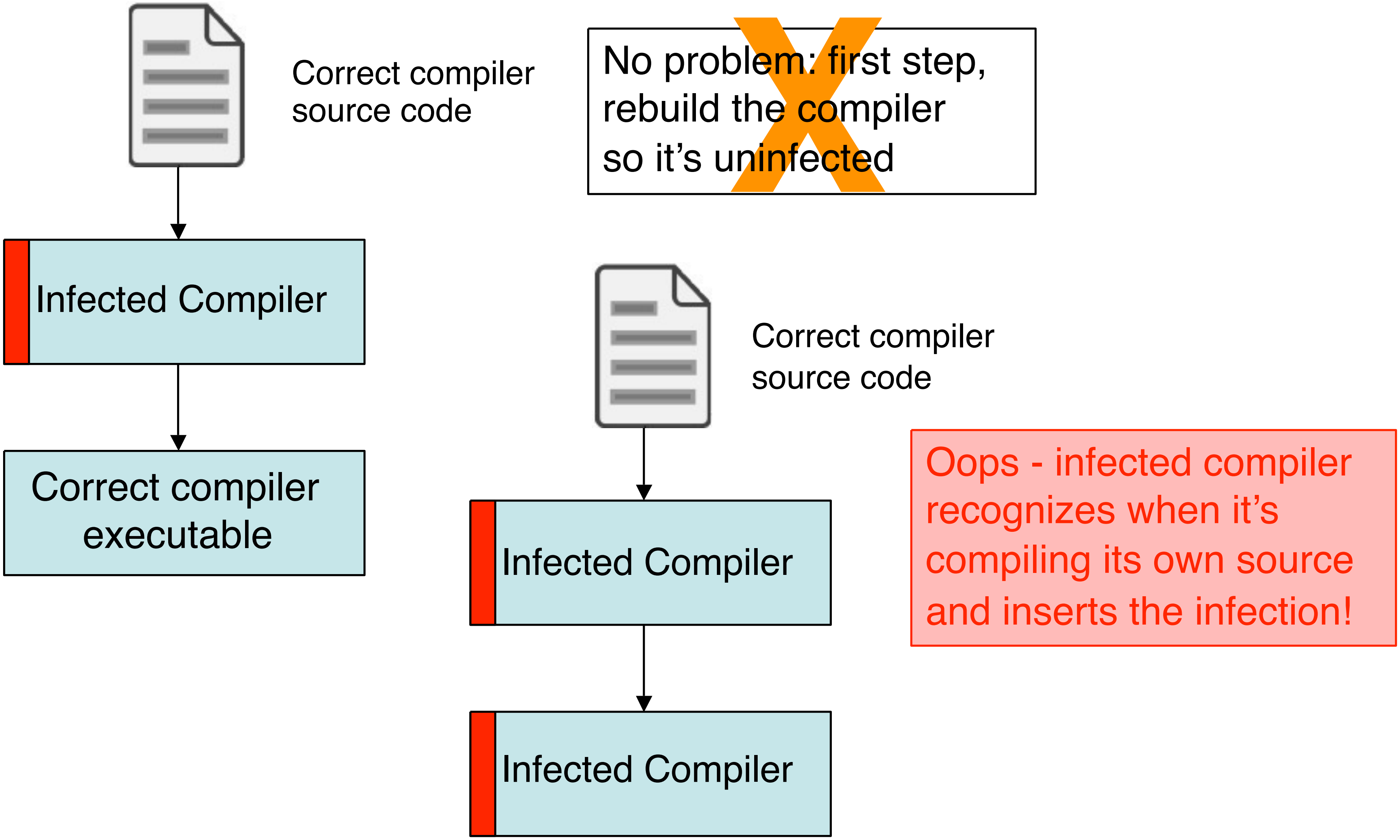
- Cleanup procedure: rebuild /bin/login from source …



Regular compilation process of building login binary from source code



Infected **compiler** recognizes when it’s compiling /bin/login source and inserts extra back door when seen


Correct compiler source code
Infected Compiler
Correct compiler executable
Correct compiler source code
Infected Compiler
Infected Compiler

~~No problem: first step, rebuild the compiler so it’s uninfected~~

Oops - infected compiler recognizes when it’s compiling its own source and inserts the infection!

No amount of careful source-code scrutiny can prevent this problem.

And if the *hardware* has a back door …

Reflections on Trusting Trust
Turing-Award Lecture, Ken Thompson, 1983

Botnets

- 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 / project 1 / 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 of C&C Messages

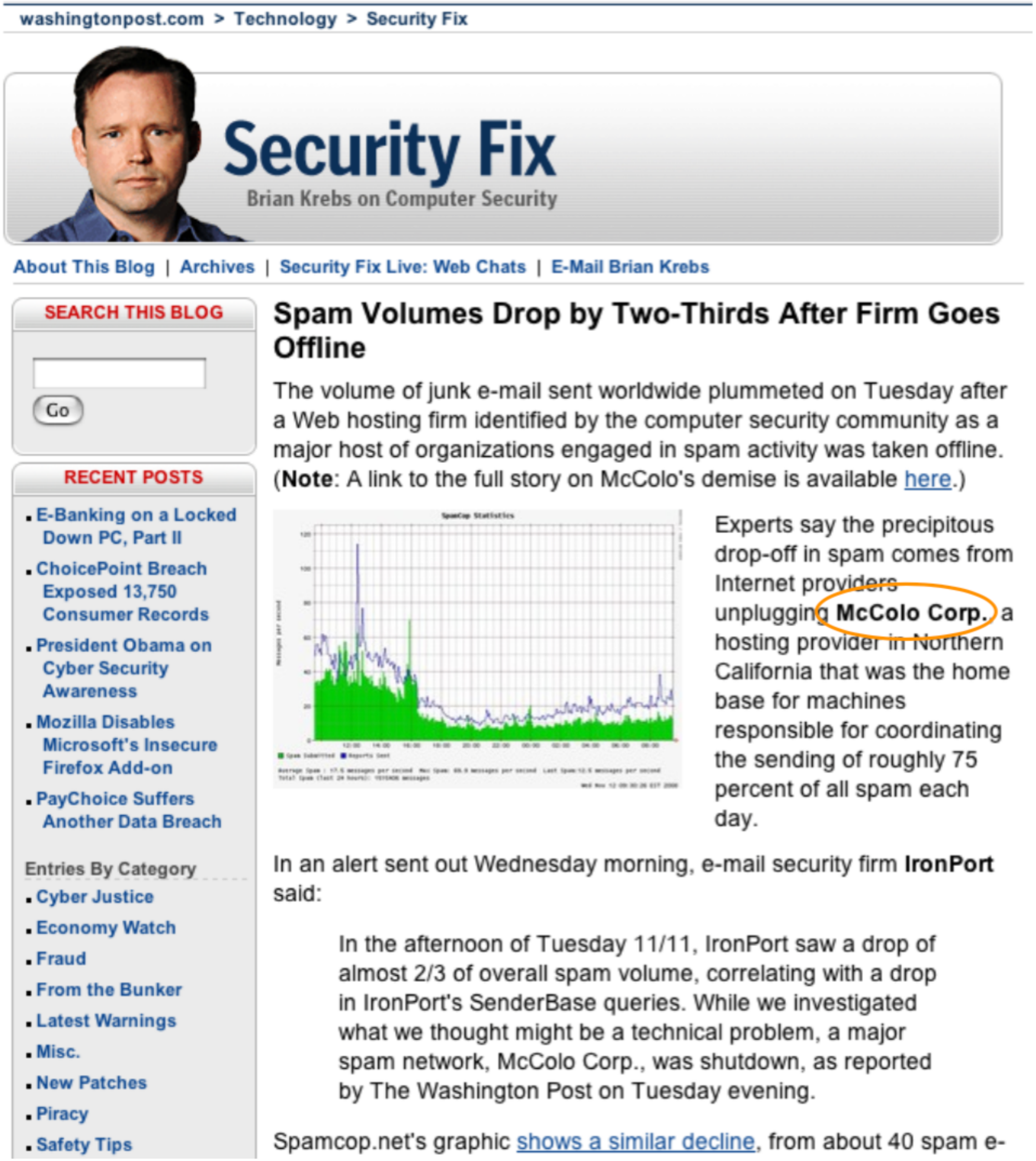
1. Activation (report from bot to botmaster)
2. Email address harvests
3. Spamming instructions
4. Delivery reports
5. Denial-Of-Service instructions
6. Sniffed passwords report

From the “Storm” botnet circa 2008

Fighting Bots / Botnets

- How can we defend against bots / botnets?

- Defense #1: prevent the initial bot infection
  - Equivalent to preventing malware infections in general …. HARD
- Defense #2: Take down the C&C master server
  - Find its IP address, get associated ISP to pull plug



washingtonpost.com > Technology > Security Fix

Security Fix

Brian Krebs on Computer Security

About This Blog | Archives | Security Fix Live: Web Chats | E-Mail Brian Krebs

SEARCH THIS BLOG

Go

RECENT POSTS

- E-Banking on a Locked Down PC, Part II
- ChoicePoint Breach Exposed 13,750 Consumer Records
- President Obama on Cyber Security Awareness
- Mozilla Disables Microsoft's Insecure Firefox Add-on
- PayChoice Suffers Another Data Breach

Entries By Category

- Cyber Justice
- Economy Watch
- Fraud
- From the Bunker
- Latest Warnings
- Misc.
- New Patches
- Piracy
- Safety Tips

Spam Volumes Drop by Two-Thirds After Firm Goes Offline

The volume of junk e-mail sent worldwide plummeted on Tuesday after a Web hosting firm identified by the computer security community as a major host of organizations engaged in spam activity was taken offline. (**Note**: A link to the full story on McColo's demise is available here.)

Experts say the precipitous drop-off in spam comes from Internet providers unplugging **McColo Corp.**, a hosting provider in Northern California that was the home base for machines responsible for coordinating the sending of roughly 75 percent of all spam each day.

In an alert sent out Wednesday morning, e-mail security firm **IronPort** said:

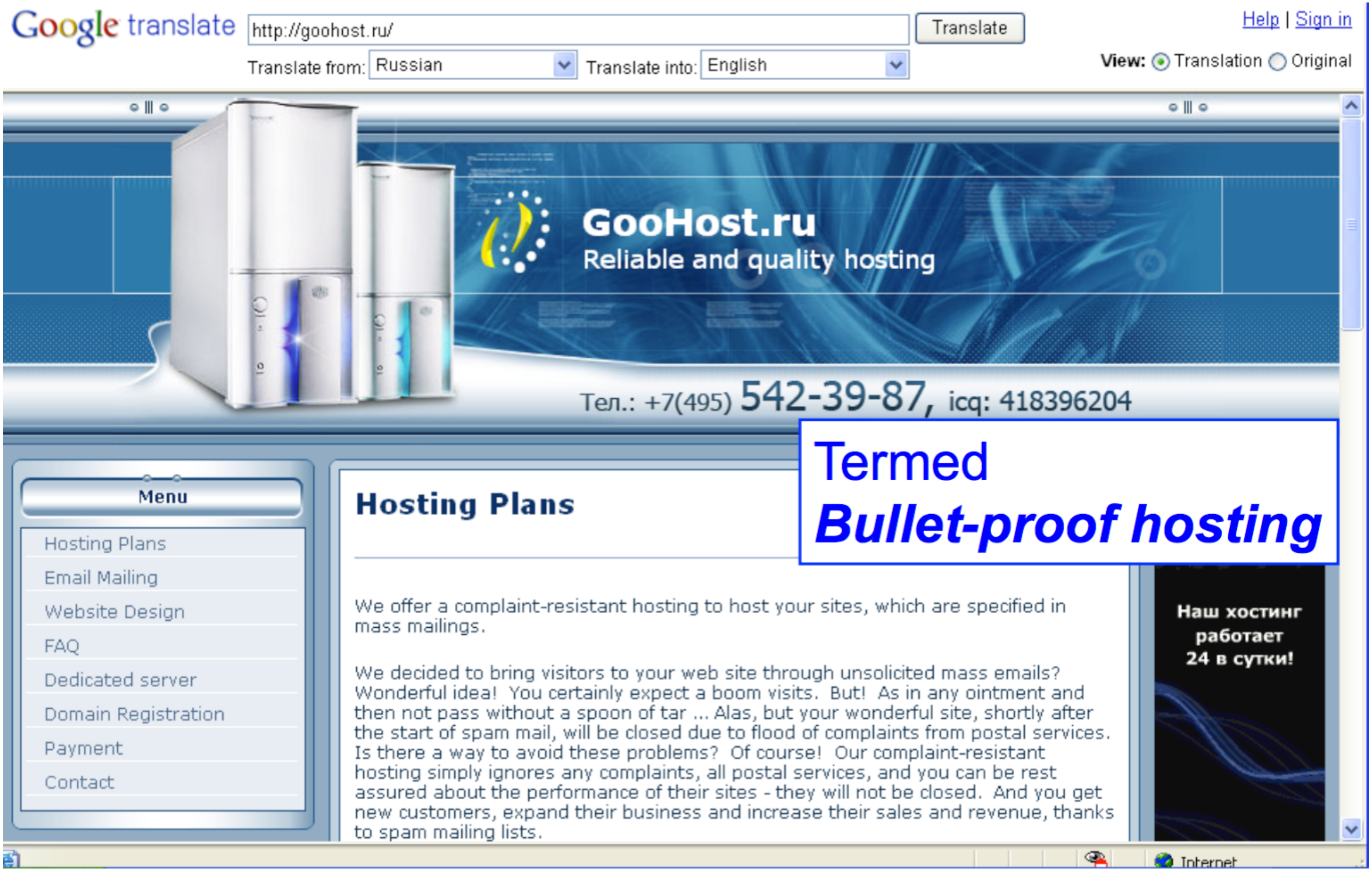
> In the afternoon of Tuesday 11/11, IronPort saw a drop of almost 2/3 of overall spam volume, correlating with a drop in IronPort's SenderBase queries. While we investigated what we thought might be a technical problem, a major spam network, McColo Corp., was shutdown, as reported by The Washington Post on Tuesday evening.

Spamcop.net's graphic shows a similar decline, from about 40 spam e-

Fighting Bots / Botnets

- How can we defend against bots / botnets?

- Defense #1: prevent the initial bot infection
  - Equivalent to preventing malware infections in general …. HARD
- Defense #2: Take down the C&C master server
  - Find its IP address, get associated ISP to pull plug
- Botmaster countermeasures?
  - Counter #1: keep moving around the master server
    - Bots resolve a domain name to find it (e.g. `c-and-c.evil.com`)
    - Rapidly alter address associated w/ name ("fast flux")
  - Counter #2: buy off the ISP …



Google translate http://goohost.ru/ Translate

Translate from: Russian Translate into: English

Help | Sign in

View: ● Translation ○ Original

GooHost.ru

Reliable and quality hosting

Тел.: +7(495) 542-39-87, icq: 418396204

Termed

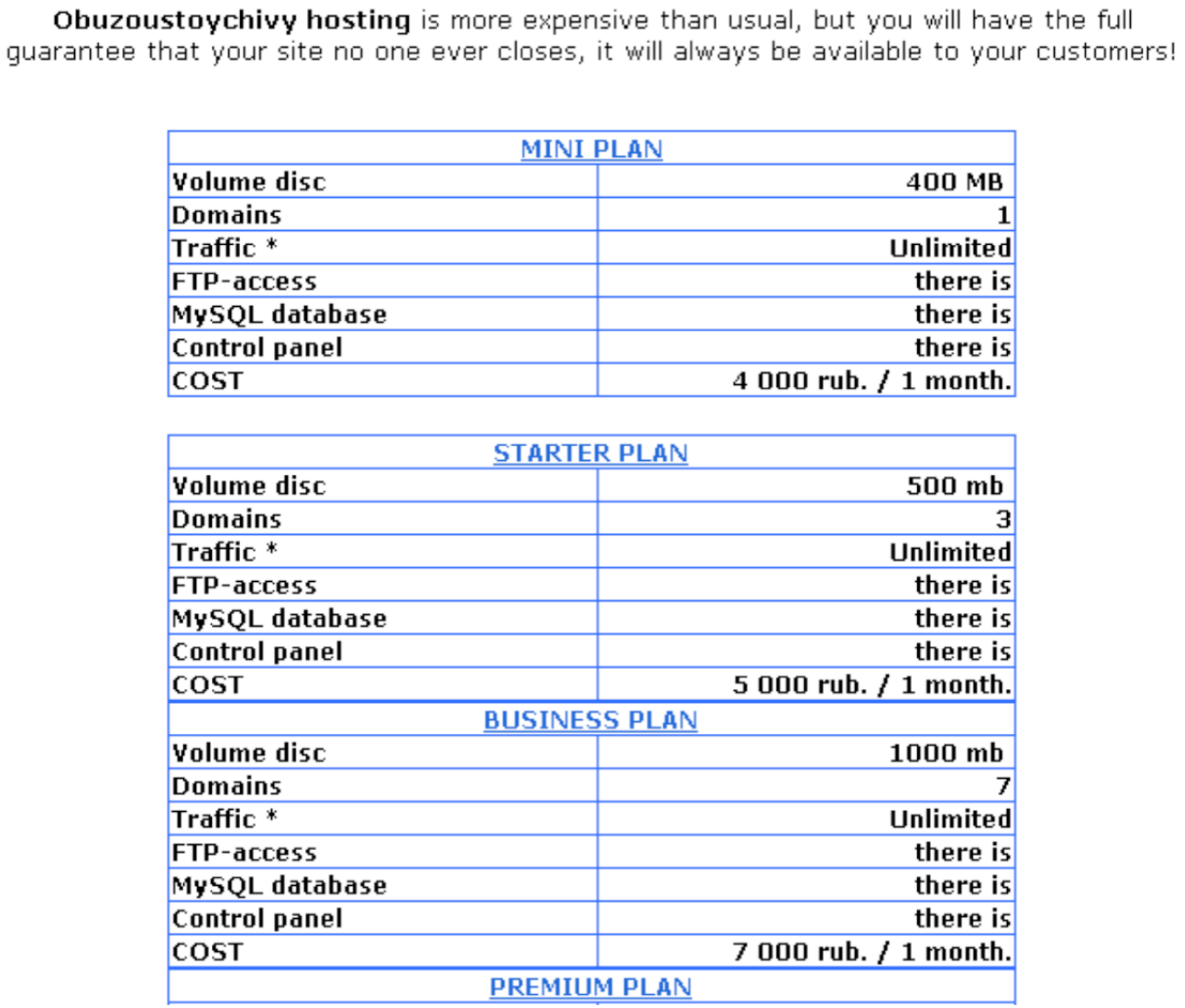
Bullet-proof hosting

Hosting Plans

We offer a complaint-resistant hosting to host your sites, which are specified in mass mailings.

We decided to bring visitors to your web site through unsolicited mass emails? Wonderful idea! You certainly expect a boom visits. But! As in any ointment and then not pass without a spoon of tar ... Alas, but your wonderful site, shortly after the start of spam mail, will be closed due to flood of complaints from postal services. Is there a way to avoid these problems? Of course! Our complaint-resistant hosting simply ignores any complaints, all postal services, and you can be rest assured about the performance of their sites - they will not be closed. And you get new customers, expand their business and increase their sales and revenue, thanks to spam mailing lists.

Наш хостинг работает 24 в сутки!



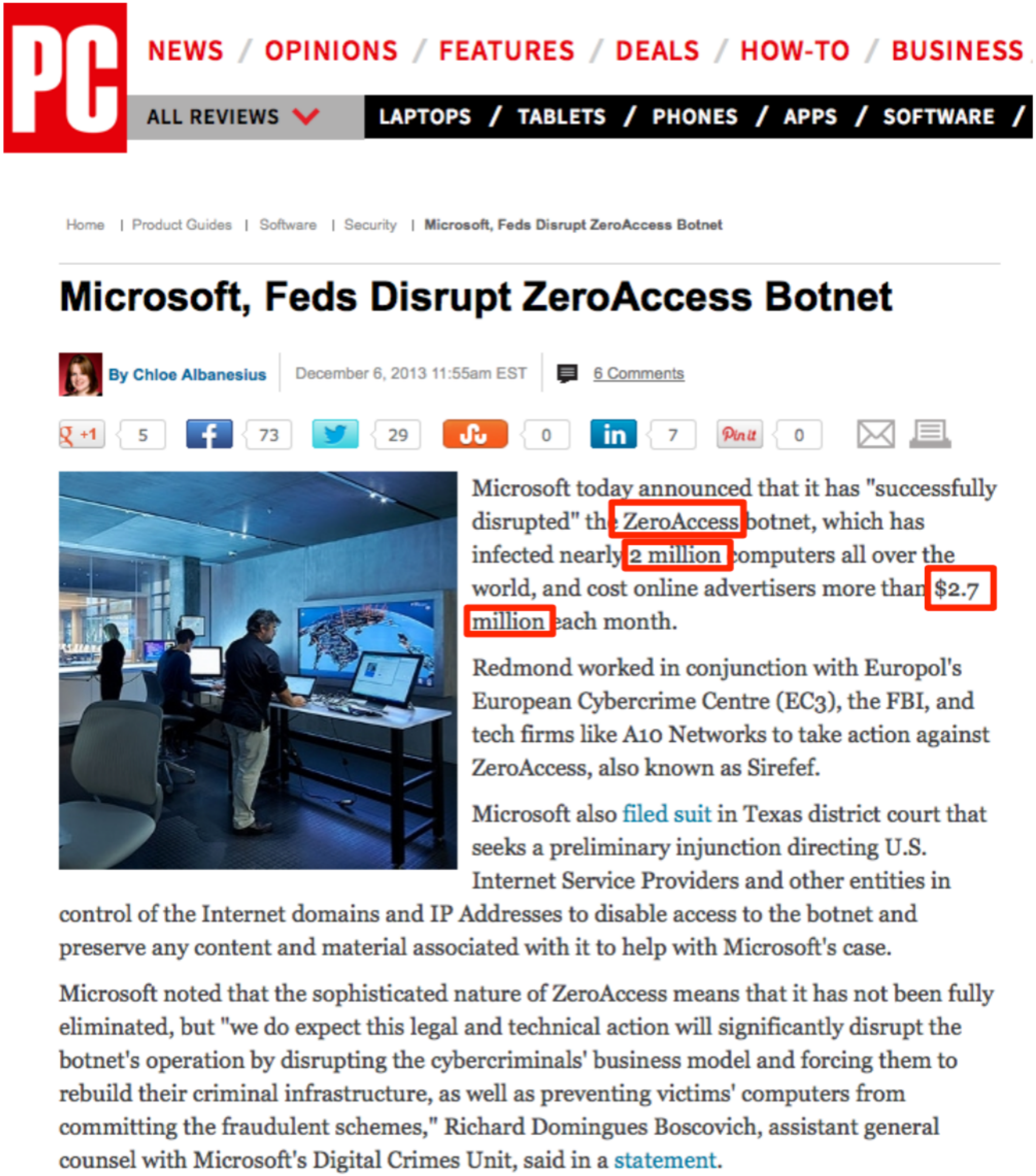
Obuzoustoychivy hosting is more expensive than usual, but you will have the full guarantee that your site no one ever closes, it will always be available to your customers!

MINI PLAN	
Volume disc	400 MB
Domains	1
Traffic *	Unlimited
FTP-access	there is
MySQL database	there is
Control panel	there is
COST	4 000 rub. / 1 month.

STARTER PLAN	
Volume disc	500 mb
Domains	3
Traffic *	Unlimited
FTP-access	there is
MySQL database	there is
Control panel	there is
COST	5 000 rub. / 1 month.
BUSINESS PLAN	
Volume disc	1000 mb
Domains	7
Traffic *	Unlimited
FTP-access	there is
MySQL database	there is
Control panel	there is
COST	7 000 rub. / 1 month.
PREMIUM PLAN	

Fighting Bots / Botnets, con't

- Defense #3: Legal action
  - Use law enforcement to seize the domain names and IP addresses used for C&C
  - This is what's currently often used, often to good effect …


PC
NEWS / OPINIONS / FEATURES / DEALS / HOW-TO / BUSINESS
ALL REVIEWS
LAPTOPS / TABLETS / PHONES / APPS / SOFTWARE /
Home | Product Guides | Software | Security | Microsoft, Feds Disrupt ZeroAccess Botnet
+1 5
73
29
0
7
Pin it 0

Microsoft, Feds Disrupt ZeroAccess Botnet

By Chloe Albanesius | December 6, 2013 11:55am EST | 6 Comments

Microsoft today announced that it has "successfully disrupted" the ZeroAccess botnet, which has infected nearly 2 million computers all over the world, and cost online advertisers more than $2.7 million each month.

Redmond worked in conjunction with Europol's European Cybercrime Centre (EC3), the FBI, and tech firms like A10 Networks to take action against ZeroAccess, also known as Sirefef.

Microsoft also filed suit in Texas district court that seeks a preliminary injunction directing U.S. Internet Service Providers and other entities in control of the Internet domains and IP Addresses to disable access to the botnet and preserve any content and material associated with it to help with Microsoft's case.

Microsoft noted that the sophisticated nature of ZeroAccess means that it has not been fully eliminated, but "we do expect this legal and technical action will significantly disrupt the botnet's operation by disrupting the cybercriminals' business model and forcing them to rebuild their criminal infrastructure, as well as preventing victims' computers from committing the fraudulent schemes," Richard Domingues Boscovich, assistant general counsel with Microsoft's Digital Crimes Unit, said in a statement.

Fighting Bots / Botnets, con't

- Defense #3: Legal action
  - Use law enforcement to seize the domain name and IP addresses used for C&C
  - Botmaster counter-measure?
  - Each day (say), bots generate large list of possible domain names using a Domain Generation Algorithm
    - Large = 50K, in some cases
  - Bots then try a random subset looking for a C&C server
    - Server **cryptographically signs** its replies, so bot can't be duped
    - Attacker just needs to hang on to a small portion of names to retain control over botnet
- This is becoming state-of-the-art …
- Counter-counter measure?
  - Behavioral signature: look for hosts that make a lot of failed DNS lookups (research)

Addressing The Botnet Problem

- What are our prospects for securing the Internet from the threat of botnets? What angles can we pursue?
- Angle #1: detection/cleanup
  - Detecting infection of individual bots hard as it's the *defend-against-general-malware* problem
  - Detecting bot doing C&C likely a losing battle as attackers improve their sneakiness & crypto
  - Cleanup today lacks oomph:
    - **Who's responsible?** … and do they **care**? (*externalities*)
    - Landscape could greatly change with different model of liability
- Angle #2: go after the C&C systems / botmasters
  - Difficult due to ease of Internet anonymity & complexities of international law
    - But: a number of recent successes in this regard
    - Including some via peer pressure rather than law enforcement (McColo)

Addressing The Problem, con’t

- Angle #3: prevention
  - Bots require installing new executables or modifying existing ones
  - Perhaps via infection …
    - … or perhaps just via user being fooled / imprudent
- In general, preventing malware infection is *hard.* Really hard
- What if we were able to provably secure 99% of all desktops!
  - (Good luck with that)
  - Is this good enough? Are we now safe?
  - No!
  - This is an asymmetric problem
    - Defenders must defend everything
    - Attackers need only a handful of targets

Addressing The Problem, con’t

- Better models?

- We could lock down systems so OS prohibits user from changing configuration
  - Sacrifices flexibility
  - How does this work for home users?
  - => Mobile (Android/iOS). Did this solve the problem?

- Or: structure OS/browser using Privilege Separation
  - Does this solve the problem?
  - Depends on how granular the privileges are … and how secure the privileged components are

Worms

Large-Scale Malware

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

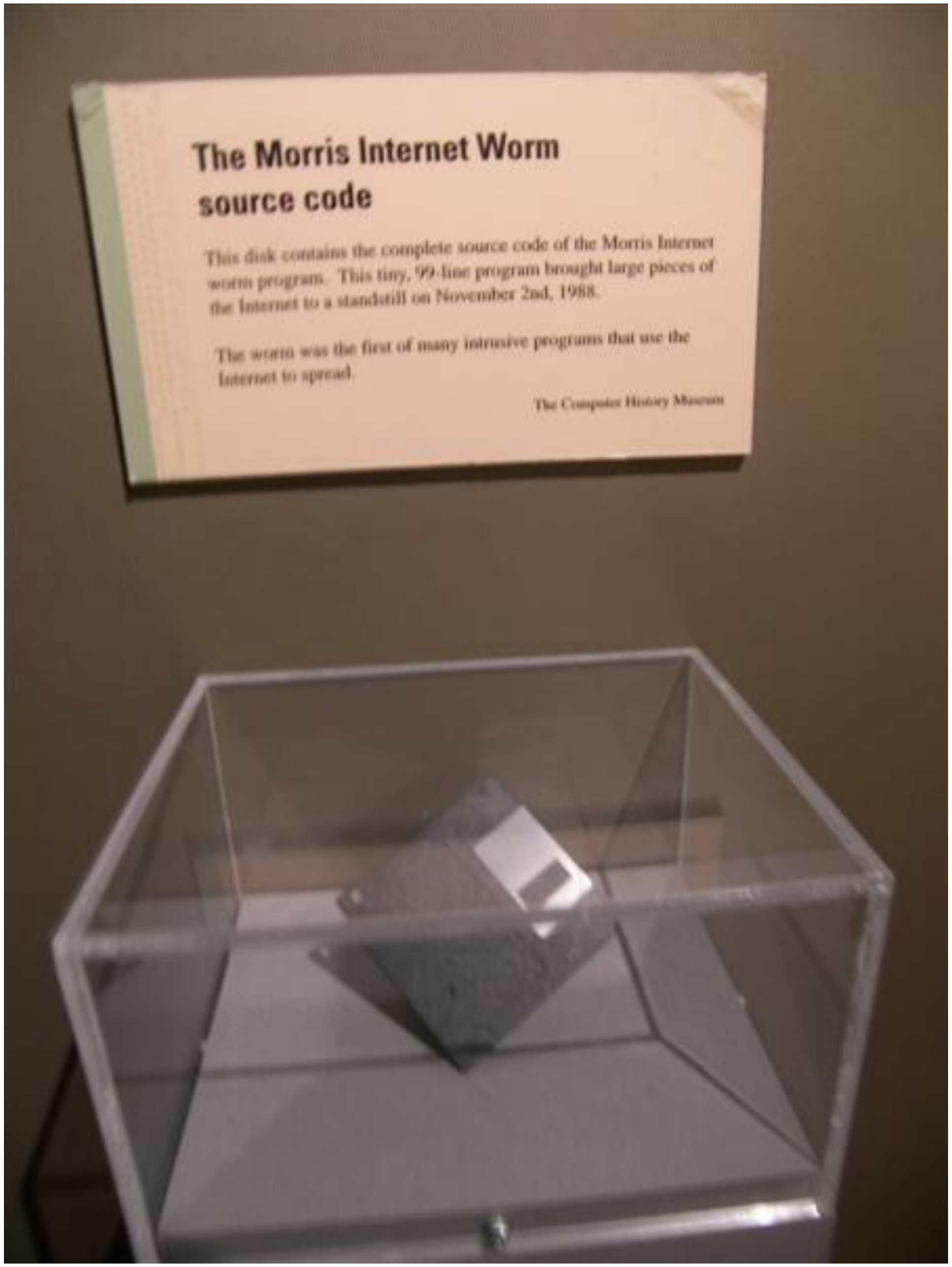
- Botnet = set of compromised machines (“bots”) under a common command-and-control (C&C)
  - Attacker might use a worm to get the bots, or other techniques; orthogonal to bot’s use in botnet

The Problem of Worms

- Virus = code that propagates (replicates) across systems by arranging to be eventually executed
  - Generally infects by altering *stored code*
- Worm = code that self-propagates/replicates across systems by arranging to have itself immediately executed
  - Generally infects by altering or initiating *running code*
  - No user intervention required
- Like with viruses, for worms we can separate out propagation from payload
- Propagation includes notions of *targeting* & *exploit*
  - How does the worm **find** new prospective victims?
  - How does worm get code to **automatically run**?

Studying Worms

- *Internet-scale events*
  - Surprising dynamics / emergent behavior
  - Hard problem of attribution (who launched it)
- Modeling propagation mathematically
- Evolution / ecosystem
  - Shifting perspectives on nature of problem
  - *Remanence*
- “Better” worms
- Thinking about defenses
  - Including “white worms”
- Mostly illustrated from a historical perspective …
  - Details/dates/names for the most part not important
    - Other than **Morris Worm**, **Code Red**, and **Slammer**



The Morris Internet Worm source code

This disk contains the complete source code of the Morris Internet worm program. This tiny, 99-line program brought large pieces of the Internet to a standstill on November 2nd, 1988.

The worm was the first of many intrusive programs that use the Internet to spread.

The Computer History Museum

The Arrival of Internet Worms

- Worms date to Nov 2, 1988 - the *Morris Worm*
- ***Way*** ahead of its time
- Employed whole suite of tricks to infect systems …
  - *Multiple* buffer overflows
  - Guessable passwords
  - "Debug" configuration option that provided shell access
  - Common user accounts across multiple machines
- … and of tricks to find victims
  - Scan local subnet
  - Machines listed in system's network config
  - Look through user files for mention of remote hosts

Arrival of Internet Worms, con't

- Modern Era began Jul 13, 2001 with release of initial version of Code Red
- Exploited known buffer overflow in Microsoft IIS Web servers
  - *On by default* in many systems
  - Vulnerability & fix announced previous month
- Payload part 1: web site defacement
  - **`HELLO! Welcome to http://www.worm.com! Hacked By Chinese!`**
  - Only done if language setting = English

Code Red of Jul 13 2001, con't

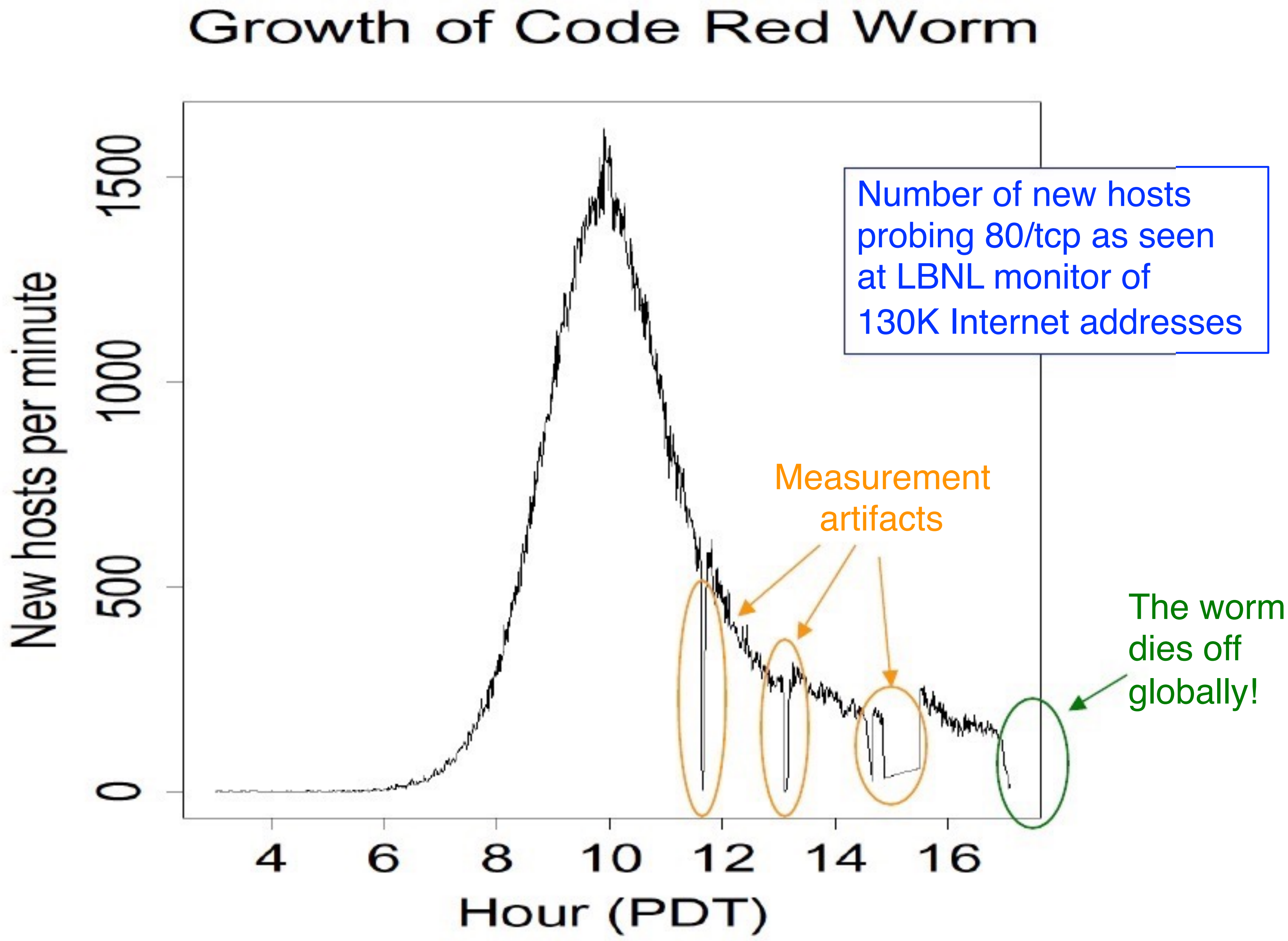
- Payload part 2: check day-of-the-month and …
  - … 1st through 20th of each month: spread
  - … 20th through end of each month: attack
    - Flooding attack against 198.137.240.91 …
    - … i.e., *www.whitehouse.gov*
- Spread: via *random scanning* of 32-bit IP address space
  - Generate pseudo-random 32-bit number; try connecting to it; if successful, try infecting it; repeat
  - Very common (but not fundamental) worm technique
- Each instance used same random number seed
  - How well does the worm spread?

Linear growth rate

Code Red, con’t

- Revision released July 19, 2001.
- White House responds to threat of flooding attack by changing the address of *www.whitehouse.gov*
- Causes Code Red to die for date ≥ 20th of the month due to failure of TCP connection to establish.
  - Author didn’t carefully test their code - buggy!

- But: this time random number generator correctly seeded. Bingo!



Number of new hosts probing 80/tcp as seen at LBNL monitor of 130K Internet addresses

Measurement artifacts

The worm dies off globally!

Modeling Worm Spread

- Worm-spread often well described as *infectious epidemic*
  - Classic SI model: homogeneous random contacts
    - SI = Susceptible-Infectible
- Model parameters:
  - N: population size
  - $S(t)$: susceptible hosts at time t.
  - $I(t)$: infected hosts at time t.
  - β: *contact rate*
    - How many population members each infected host communicates with per unit time

$$\boxed{\begin{array}{c} N = S(t) + I(t) \\ S(0) = I(0) = N/2 \end{array}}$$

- Auxiliary parameters reflecting the relative proportion of infected/susceptible hosts
  - $s(t) = S(t)/N \qquad i(t) = I(t)/N \qquad s(t) + i(t) = 1$

Computing How An Epidemic Progresses

- In continuous time:

Increase in # infectibles per unit time

$$\frac{dI}{dt} = \beta \cdot I \cdot \frac{S}{N}$$

Proportion of contacts expected to succeed

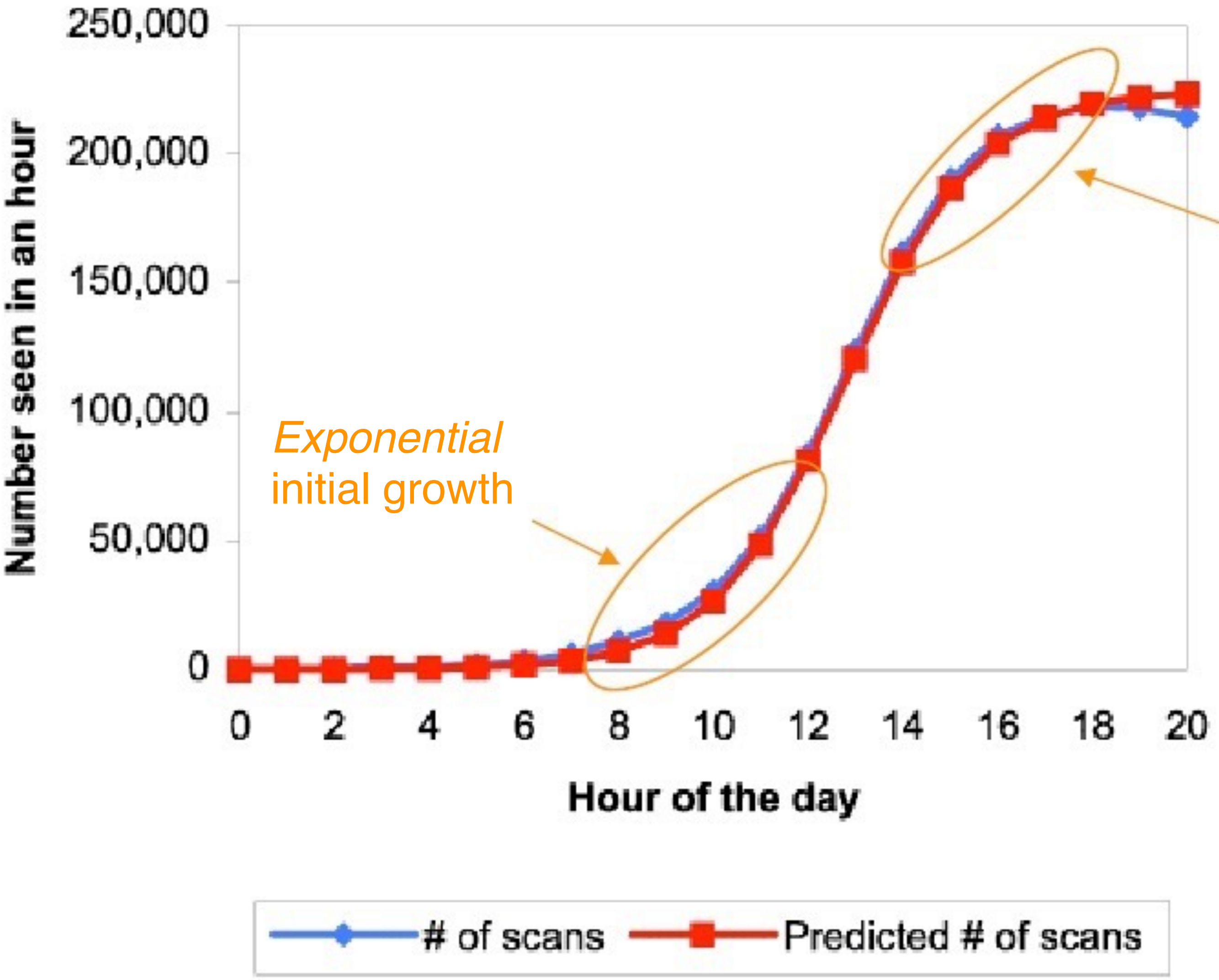
Total attempted contacts per unit time

- Rewriting by using i(t) = I(t)/N, S = N - I:

$$\frac{di}{dt} = \beta i(1-i) \quad \Rightarrow \quad i(t) = \frac{e^{\beta t}}{1+e^{\beta t}}$$

Fraction infected grows as a *logistic*

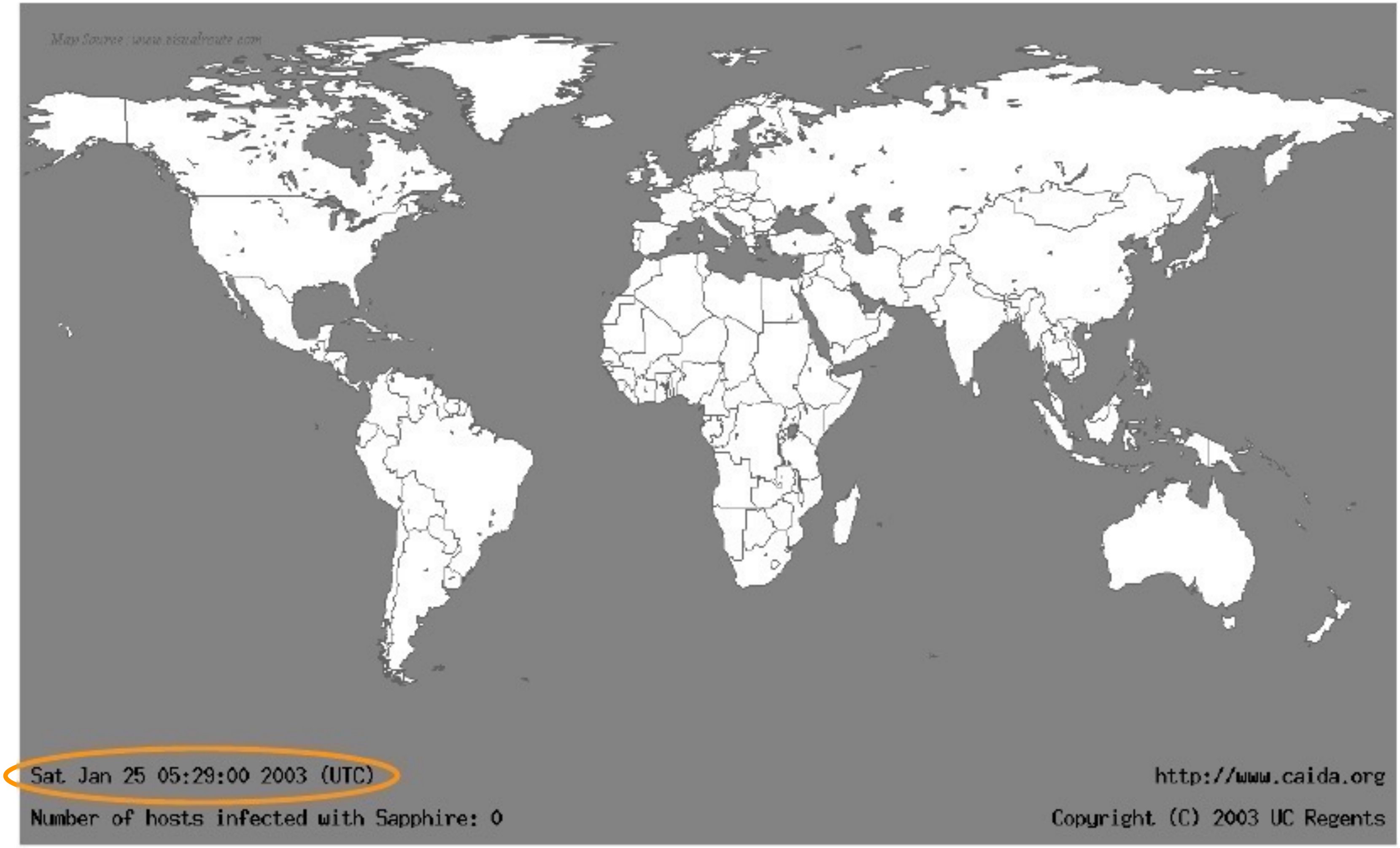
Fitting the Model to Code Red



Exponential initial growth

Growth slows as it becomes harder to find new victims!

Life Just Before Slammer



Map Source : www.visualroute.com

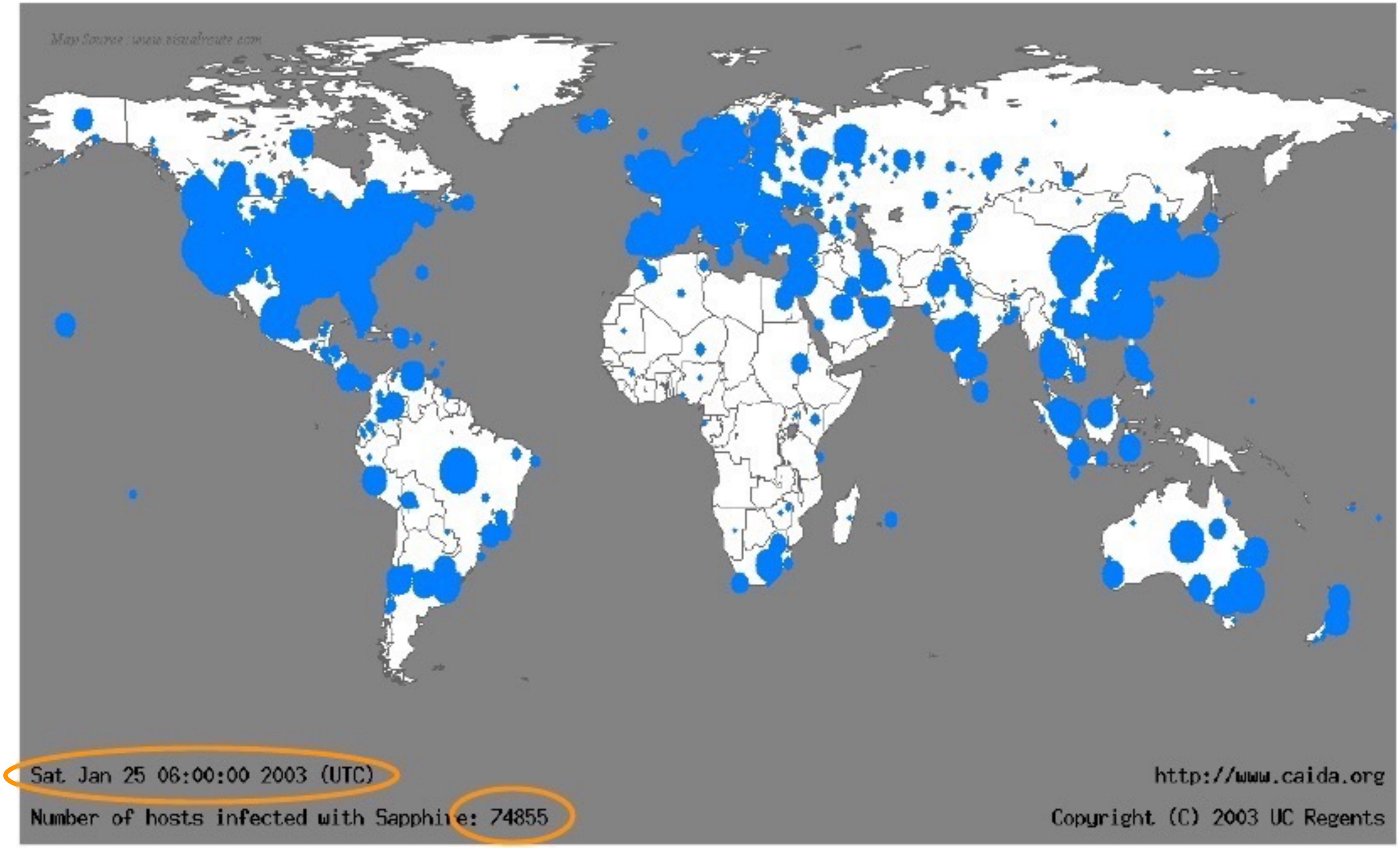
Sat Jan 25 05:29:00 2003 (UTC)

Number of hosts infected with Sapphire: 0

http://www.caida.org

Copyright (C) 2003 UC Regents

Life Just After Slammer


Map Source : www.visualroute.com
Sat Jan 25 06:00:00 2003 (UTC)
Number of hosts infected with Sapphire: 74855
http://www.caida.org
Copyright (C) 2003 UC Regents

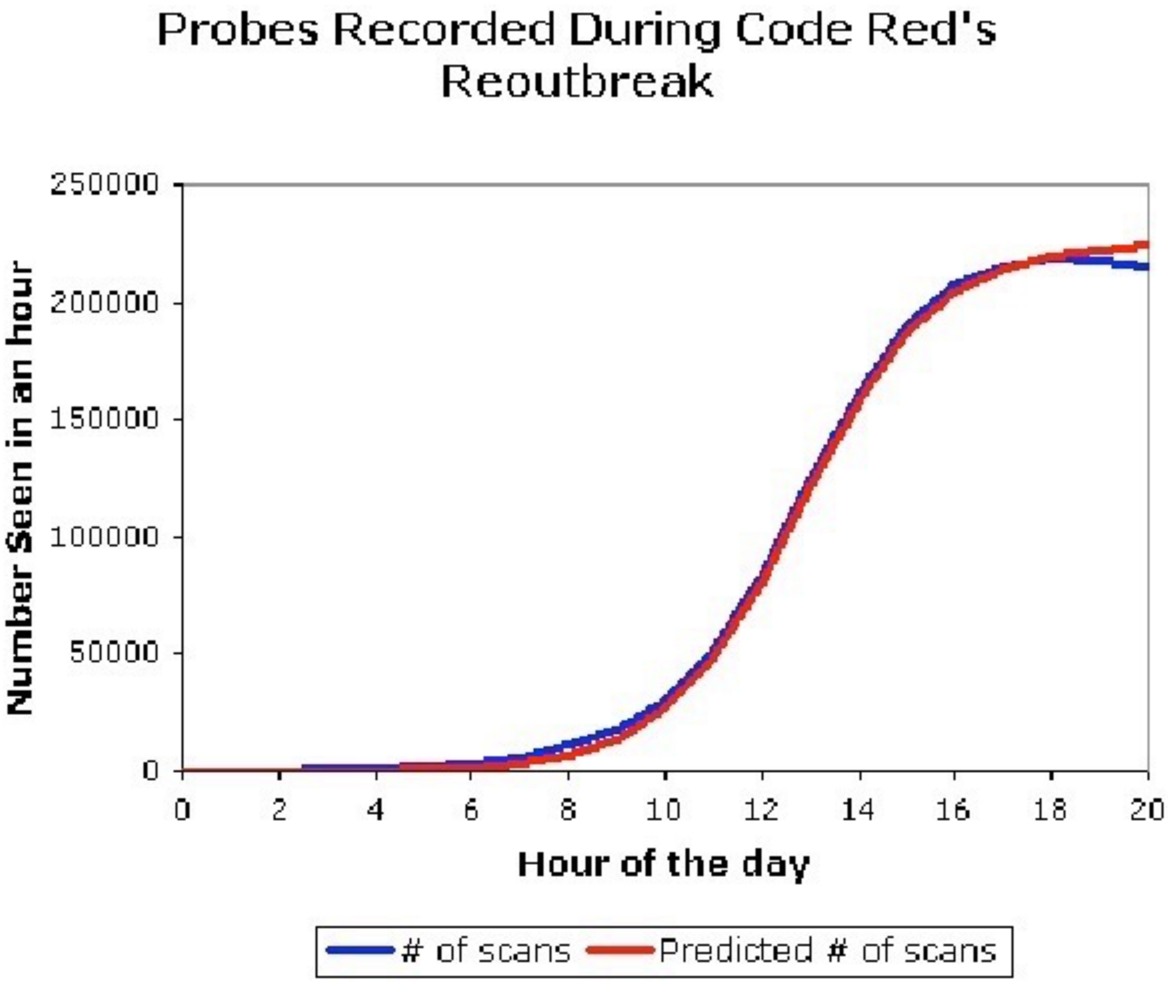
Going Fast: *Slammer*

- Slammer exploited connectionless UDP service, rather than connection-oriented TCP
- *Entire worm* fit in a single packet!

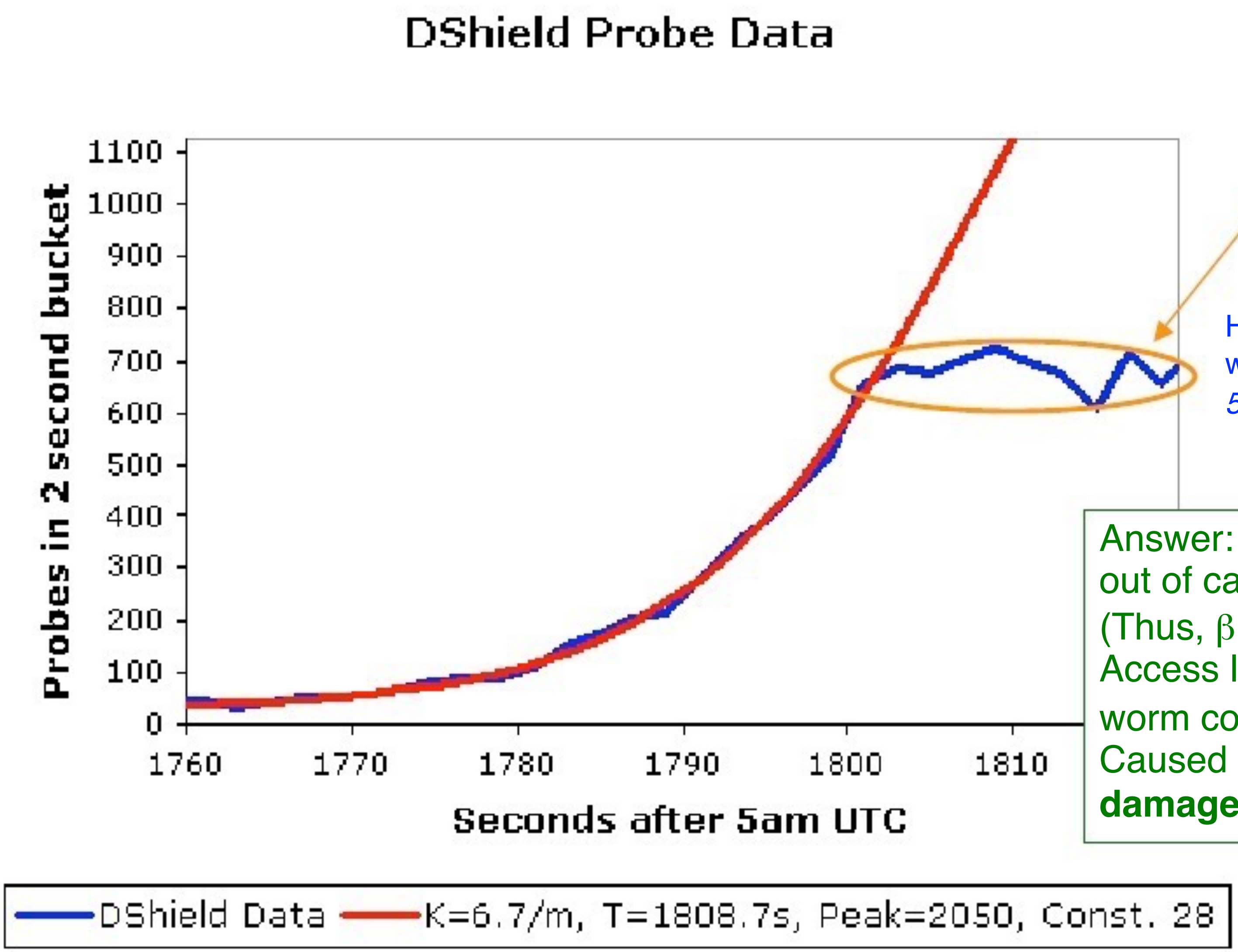
⇒ When scanning, worm could “fire and forget” *Stateless!*

- Worm infected 75,000+ hosts in 10 minutes (despite broken random number generator).
- At its peak, **doubled every <u>8.5 seconds</u>**

The Usual Logistic Growth



Slammer's Growth

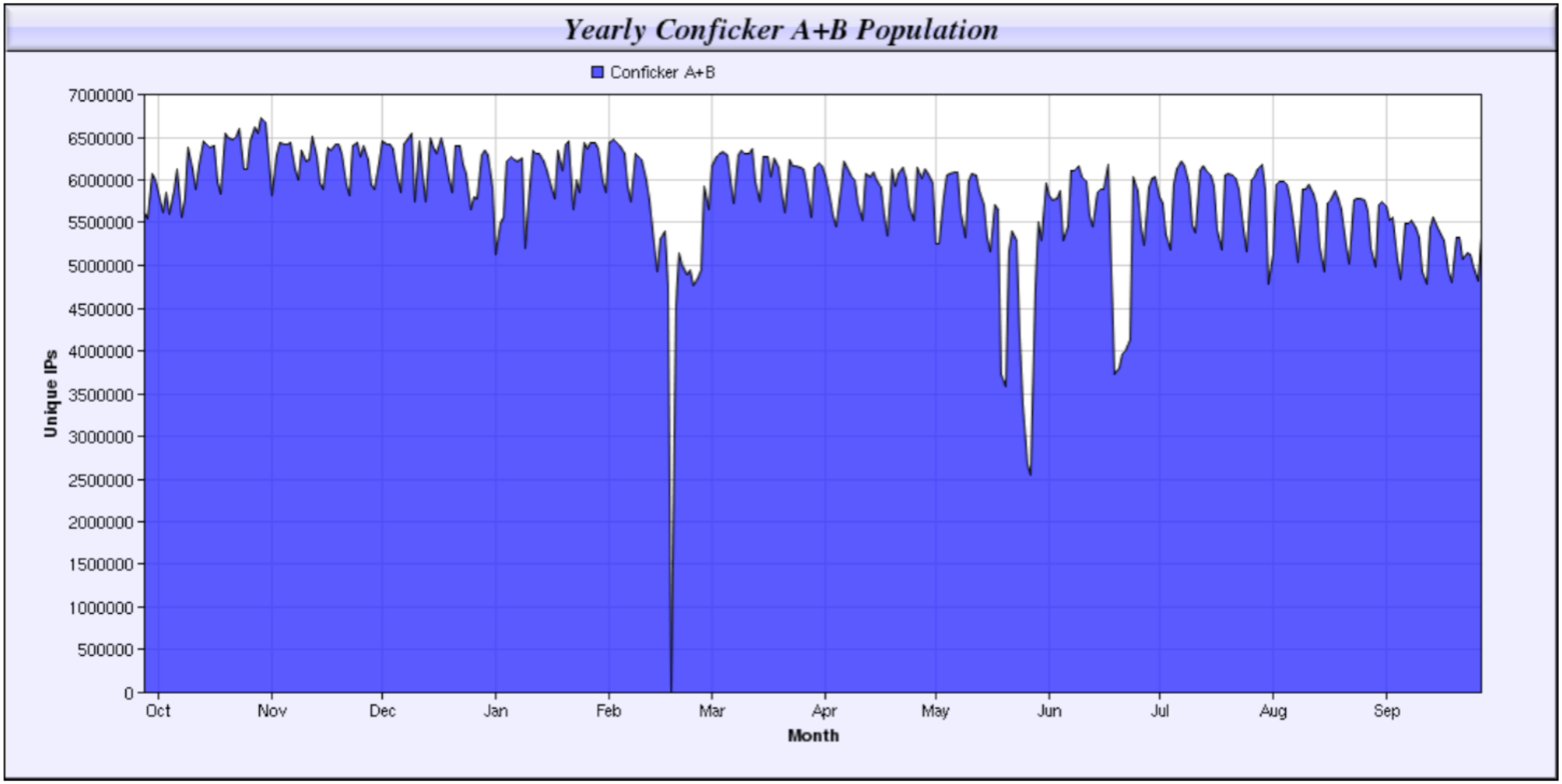


What could have caused growth to deviate from the model?

Hint: at this point the worm is generating *55,000,000 scans/sec*

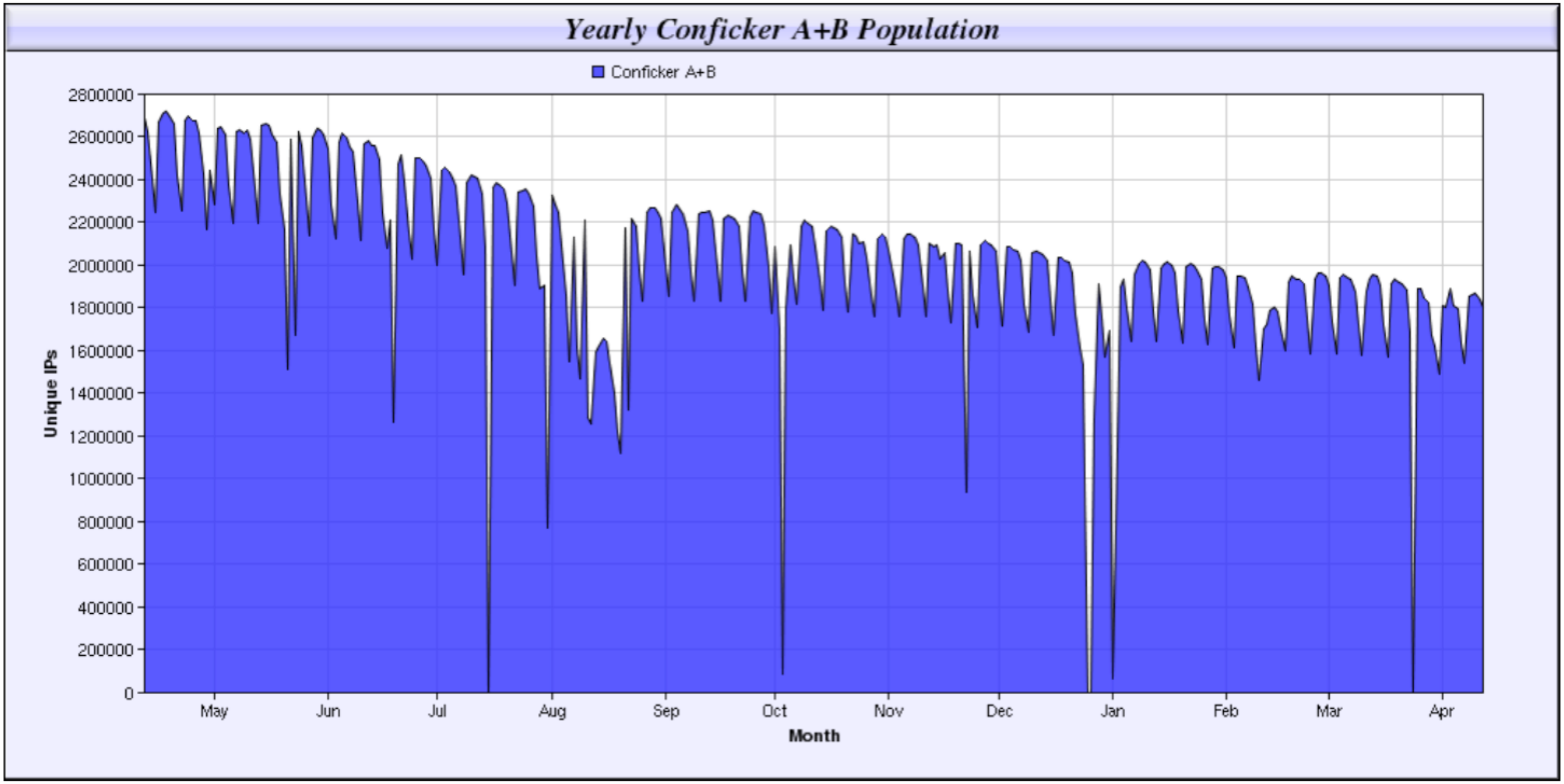
Answer: the Internet ran out of carrying capacity! (Thus, β decreased.) Access links used by worm completely clogged. Caused **major collateral damage**.

Big Worms: Conficker



2009 - 2010

Big Worms: Conficker



2012 - 2013

Stuxnet

- Discovered July 2010. (Released: Mar 2010?)
- Multi-mode spreading:
  - Initially spreads via USB (virus-like)
  - Once inside a network, quickly spreads internally using Windows RPC
- Kill switch: programmed to die June 24, 2012
  - Targeted *SCADA* systems
  - Used for industrial control systems, like manufacturing, power plants
- Symantec: infections geographically clustered
  - Iran: 59%; Indonesia: 18%; India: 8%

Stuxnet, con’t

- Used four *Zero Days*
  - Unprecedented expense on the part of the author
- “Rootkit” for hiding infection based on installing Windows drivers with valid digital signatures
  - Attacker stole private keys for certificates from two companies in Taiwan
- Payload: do nothing …
  - … unless attached to particular models of frequency converter drives operating at 807-1210Hz
  - … like those made in Iran (and Finland) …
  - … and used to operate centrifuges for producing enriched Uranium for nuclear weapons

Stuxnet, con’t

- Payload: do nothing …
  - … unless attached to particular models of frequency converter drives operating at 807-1210Hz
  - … like those made in Iran (and Finland) …
  - … and used to operate centrifuges for producing enriched Uranium for nuclear weapons
- For these, worm would slowly increase drive frequency to 1410Hz …
  - … enough to cause centrifuge to **fly apart** …
  - … while sending out fake readings from control system indicating everything was okay …
- … and then drop it back to normal range

Israel Tests on Worm Called Crucial in Iran Nuclear Delay

By WILLIAM J. BROAD, JOHN MARKOFF and DAVID E. SANGER

Published: January 15, 2011

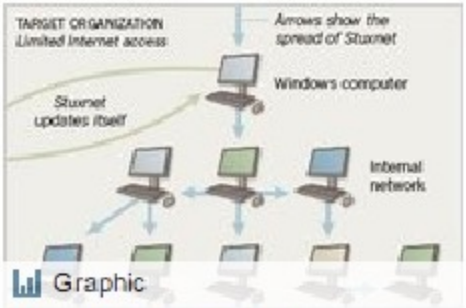
This article is by ***William J. Broad, John Markoff*** *and* ***David E. Sanger.***

Enlarge This Image

Nicholas Roberts for The New York Times

Ralph Langner, an independent computer security expert, solved Stuxnet.

Multimedia



How Stuxnet Spreads

The Dimona complex in the Negev desert is famous as the heavily guarded heart of Israel's never-acknowledged nuclear arms program, where neat rows of factories make atomic fuel for the arsenal.

Over the past two years, according to intelligence and military experts familiar with its operations, Dimona has taken on a new, equally secret role — as a critical testing ground in a joint American and Israeli effort to undermine Iran's efforts to make a bomb of its own.

Behind Dimona's barbed wire, the experts say, Israel has spun nuclear centrifuges virtually identical to Iran's at Natanz, where Iranian scientists are struggling to enrich uranium. They say Dimona tested the effectiveness of the Stuxnet computer worm, a destructive program that appears to have wiped out roughly a fifth of Iran's nuclear

Worm Take-Aways

- Potentially enormous reach/damage

  ⇒ *Weapon*
- Hard to get right
- Emergent behavior / surprising dynamics
- Institutional antibodies
- Remanence: worms stick around
  - E.g. Nimda & Slammer still seen in 2011!
- *Propagation faster than human response*
- What about fighting a worm using a worm?
  - "White worm" spreads to disinfect/patch
  - Experience shows: likely not to behave predictably!
  - Additional issues: legality, collateral damage, target worm having already patched so white worm can't access victim

Summary

- Malware = malicious code that runs on a victim's system
  - Infection can occur in a variety of ways
- Some malware propagates automatically
  - Viruses
  - Worms
- Botnet = set of compromised machines
  - Botnets are a modern, persistent, and very real threat
  - Extremely hard problem

Closing Thought…

- As long as criminals can continue to monetize malware, the malware threat is likely to remain