

Caffeine Monkey

Automated Collection, Detection and Analysis
of Malicious JavaScript

Ben Feinstein, CISSP & Daniel Peck
SecureWorks, Inc.

Malicious Javascript

- Why should you care?
- Malware/Spyware Installation
- Bypass detection

Who'd a thought animated cursors could be so dangerous?

- Developed by Netscape in 1995
- Javascript/Jscript/EMCAScript
- Javascript != DOM
- Blurs the lines between data/code

Feature / functionality bloat

- Blame AJAX
- XMLHttpRequest
- Most security risks occur at edges of functionality

Web 2.0 – Ain't it grand

- Tried using a browser with Javascript turned off lately?
- Ingrained into the web programmer/scripter
- Many popular sites unusable

Is it really dangerous?

- **MoBB #25: Native Function Iterator**
- **MoBB #8: RDS.DataControl URL**
- Gnucitizen.org JavaScript AttackAPI
- **SPI Port Scanning**

Phishing/XSS

- XSS, the seemingly ubiquitous vulnerability that won't go away and keeps getting worse
- Ebay seller rating
- Address bar spoofing

Postmortems

- Superbowl Hack
 - MS06-014
 - MS07-004
- Quicktime Embedded Javascript
- **Adobe PDF XSS**

Obfuscation / evasion techniques

- Whitespace randomization / randomized comments
 - Changes the bytestream 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
 - Specific values for memory corruption/heap spray
 - 0x40000000 can be represent practically infinite ways
- Variable and function name 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

Hunting Malicious Javascript

- Goals
 - Automate the discovery of JavaScript evasion techniques and exploits
 - Automate the discovery of sites hosting malicious content

Pitfalls in Current Techniques

- HoneyClients
 - **Strider HoneyMonkey**
 - **Mitre Honeyclient**
 - HoneyC
- Heavyweight
- Behavioral Analysis

Pitfalls in Current Techniques (cont)

- Human Analysis
 - Time consuming!
 - Error prone
 - Do you trust you `<textarea>` wrapper in 0day conditions?
- There has to be a way to get the best of both worlds

Enter Caffeine Monkey

- Like many ideas, born at local bar
- Central DB for collection and preliminary 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
 - The good people at Michigan for their php script
- Wrapping and logging methods
- Demo

So what did we find?

- Initial Targets
 - Myspace
 - Warez/Serials sites
 - .eduPorno sites
 - StopBadware Sites
- Lots of obfuscated cookies/tracking/etc
- Not perfect, but Myspace runs a cleaner ship than we expected

Good JS vs Bad JS

- Fingerprinting
- How methods are used
- Best show in multiple vs singular purpose

Method Call Graphs

Future of Caffeine Monkey?

- Released today
 - Expand on it and save everyone some time
- Inclusion in proxy?
 - IDS/IPS?
 - Heuristics based addition to signature based platforms?
- Firefox plugin?