Bad Cocktail
Spear Phishing + Application Hacks

OWASP Chicago
August 2008
Spear Phishing Is A Problem

• > 15,000 corporate victims in 15 months
• Victim Losses have exceeded $100,000
• Recent Victims
  – Salesforce.com
  – Critical infrastructure at large energy company

Sources: iDefense Labs, Washington Post
Why Does Spear Phishing Succeed?

• People are “click happy”
• Phishing attacks have gotten more sophisticated – use of legitimate sites with application security flaws
• Reactive Anti-Phishing technologies aren’t good enough....may never be!
People Are “Click Happy”

EMAIL RESPONSES

Scenario (6):

- Entered Data, 0%
- Clicked Link, 24.22%
- No Response, 75.78%
How To Use Legitimate Sites

Take advantage of:

• Cross Site Scripting
• Insecure URL redirection
• Session Fixation
• Insecure ActiveX Controls
The XSS Phishing Mail

• More realistic phishing attack because it uses the actual site (often even over HTTPS)

• XSS Tricks
  – Expect HEX encoding of attack parameters
    • “<script>” = “%3C%73%63%72%69%70%74%3E”
  – Short attack parameter that links to a remote “.js” file for more javascript or an “iframe” tag that loads remote HTML form
XSS Emails in the Real World

- Charter One Bank (Citizens Financial Group)
  – March 2005

https://www.charterone.com/pf/?ygtkt=%61%53%33%87%64%38%80%87%76%23%66%59%44%95%16%28%88%6%19%85%91%20...
Cross Site Scripting Not Dead Yet

**Citibank's critical cross-site scripting vulnerabilities**
*Written by Dimitris Pagkalos*
*Saturday, 16 August 2008*

DaiMon and mox have discovered two critical XSS flaws on Citibank's website.

*read more...*

**Justin.tv non-malicious cross-site scripting worm**
*Written by Dimitris Pagkalos*
*Tuesday, 8 July 2008*

x2Fusion from TheDefaced.org security team, recently contacted us in regards to a serious XSS vulnerability on the popular lifecasting website Justin.tv.

*read more...*

**ICANN and IANA domains hijacked by Turkish crackers**
*Written by Marcelo “Vympel” Almeida and Kevin Fernandez*
*Thursday, 26 June 2008*

The ICANN and IANA websites were defaced earlier today by a Turkish group called "NetDevilz". ICANN is responsible for the global coordination of the Internet’s system of unique identifiers. These include domain names, as well as the addresses used in a variety of Internet protocols.

*read more...*

**HSBC web sites are open to critical XSS attacks. Warning to customers!**
*Written by Dimitris Pagkalos*
*Saturday, 21 June 2008*
URL redirection

• Used to mask where the link is really taking you

• Often comes in one of two ways
  – 3rd party trust (known vendor, popular search site)
  – Or misconfiguration on your site
URL 3rd Party Redirection

• Because search engines never lie... right?

• Often used for tracking Ad clicks, many sites will have a way to redirect based off a URL sent in
Homegrown Redirection

- Be careful about how your own redirects are coded


- Again HEX encoding tricks can be used
  - “evil.com” = “%65%76%69%6C%2E%63%6F%6D”
Don’t forget Flash

• Flash Objects can perform their own redirects.
• “eBay Flash-redirect scam”
  – Reported in Aug 2007
  – Attacker creates legitimate auction page but places malicious flash “SWF” file in description
  – When another eBay user views their page, they are redirected to a cloned malicious site which ask them to login
Would you notice a redirect?

• Since you just clicked on a legitimate link, you may expect the page to reload
Insider Phishing Attack

• Some SSL VPNs can be used by an attacker to form believable “internal” phishing sites
• A legitimate link to the mail server maybe:

Insider Phishing Attack

• If a phisher knows your SSL VPN page and vendor (support page? search email lists?) then linking back out to a site on the internet is often supported.


• Think users know internal IP address from routable addresses?
Next Level: CSRF->DNS->Phish

• This attack as been described at “drive-by pharming” and seen in the wild in Jan 2008 targeting Mexican banking sites

• Complex Attack in 3 Steps
  – 1) Use a CSRF attack against home router to reconfigure DNS settings
    https://192.168.1.1/apply.cgi?submit_button=Submit&action=Apply&block_wan=1&block_loopbacks=0&dns1=6.6.6.6
Next Level: CSRF->DNS->Phish

• Complex Attack in 3 Steps (continued)
  – 2) Attacker hosts DNS server at “6.6.6.6” and returns malicious DNS responses for known banking sites.
  – 3) Malicious response point to fake cloned site. The URL matches the legitimate site, however DNS gave out the wrong IP address

• Attacker can just wait for victim to surf to their trusted site, or send an email with a real link
Drive-By Pharming

• Sneaky, but difficult to execute
  – Must trick users into visiting site hosting CSRF attack
  – Victim's router IP must be know, must be vulnerable to CSRF, often must be logged in
  – HTTPS request will trigger invalid certificate responses
A Report From The Trenches
Symptoms

• “I see a trade executed from my account …10000 shares of a company I haven’t even heard about, were purchased on January 17 (2006) @ 2 pm from my account!” — a client of a well-established brokerage firm in NYC.

• 7 other clients of the same brokerage firm report the same issue — in January 2006.
Investigation

• Was the brokerage firm hacked?
• Was it the end user who was hacked?
• We had dates and times of the trade executions as a clue.
Investigation

• Our team began reviewing the brokerage firm’s online trading application for clues
  – Network logs
  – Web server logs
  – Security mechanisms of the application
• We asked to duplicate the victim’s hard drive and review it for indicators of compromise.
Web Server Logs

- Requested IIS logs for January 17, 2006 from all the (load balanced) servers.

- Combined the log files into one common repository = 1 GB

- Microsoft’s Log Parser to the rescue
Microsoft LogParser

Parsed out all requests to execute.asp using Microsoft Log Parser:

LogParser -o:csv "select * INTO execute.csv from *.log where cs-uri-stem like '/execute.asp%'"
# Can You Find The Smoking Gun?

<table>
<thead>
<tr>
<th>c-ip</th>
<th>cs-method</th>
<th>cs-uri-stem</th>
<th>cs-uri-query</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>172.16.22.33</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=90198e1525e4b03797f833ff4320af39</td>
<td>200</td>
</tr>
<tr>
<td>172.16.54.33</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=3840943093874b3484c3839de9340494</td>
<td>200</td>
</tr>
<tr>
<td>172.16.22.33</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=90198e1525e4b03797f833ff4320af39</td>
<td>200</td>
</tr>
<tr>
<td>172.16.87.231</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=298230e0393bc09849d839209883993</td>
<td>200</td>
</tr>
<tr>
<td>172.16.22.33</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=90198e1525e4b03797f833ff4320af39</td>
<td>200</td>
</tr>
<tr>
<td>172.16.22.33</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=90198e1525e4b03797f833ff4320af39</td>
<td>200</td>
</tr>
<tr>
<td>172.16.121.3</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=676db87873ab0393898de0398348c89</td>
<td>200</td>
</tr>
<tr>
<td>172.16.41.53</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=3840943093874b3484c3839de9340494</td>
<td>200</td>
</tr>
<tr>
<td>172.16.22.33</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=90198e1525e4b03797f833ff4320af39</td>
<td>200</td>
</tr>
<tr>
<td>172.16.22.33</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=90198e1525e4b03797f833ff4320af39</td>
<td>200</td>
</tr>
</tbody>
</table>
Next Step

Parsed out all requests with the suspicious sessionid

LogParser -o:csv "select * INTO sessionid.csv from *.log where cs-uri-query like '%90198e1525e4b03797f833ff4320af39'"
Can You Find The Smoking Gun?

<table>
<thead>
<tr>
<th>Fields</th>
<th>c-ip</th>
<th>cs-method</th>
<th>cs-uri-stem</th>
<th>cs-uri-query</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
<td>c-ip</td>
<td>cs-method</td>
<td>cs-uri-stem</td>
<td>cs-uri-query</td>
<td></td>
</tr>
<tr>
<td>1:18:15</td>
<td>172.16.22.33</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=90198e1525e4b03797f833ff4320af39</td>
<td>200</td>
</tr>
<tr>
<td>1:23:16</td>
<td>172.16.22.33</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=90198e1525e4b03797f833ff4320af39</td>
<td>200</td>
</tr>
<tr>
<td>1:28:15</td>
<td>172.16.22.33</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=90198e1525e4b03797f833ff4320af39</td>
<td>200</td>
</tr>
<tr>
<td></td>
<td>172.16.22.33</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=90198e1525e4b03797f833ff4320af39</td>
<td></td>
</tr>
<tr>
<td></td>
<td>172.16.22.33</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=90198e1525e4b03797f833ff4320af39</td>
<td></td>
</tr>
<tr>
<td>13:53:15</td>
<td>172.16.22.33</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=90198e1525e4b03797f833ff4320af39</td>
<td>200</td>
</tr>
<tr>
<td>13:58:15</td>
<td>172.16.22.33</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=90198e1525e4b03797f833ff4320af39</td>
<td>200</td>
</tr>
<tr>
<td>14:03:15</td>
<td>172.16.22.33</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=90198e1525e4b03797f833ff4320af39</td>
<td>200</td>
</tr>
<tr>
<td>14:07:54</td>
<td>172.16.14.166</td>
<td>POST</td>
<td>/account.asp</td>
<td>sessionid=90198e1525e4b03797f833ff4320af39</td>
<td>200</td>
</tr>
<tr>
<td>14:08:15</td>
<td>172.16.22.33</td>
<td>POST</td>
<td>/execute.asp</td>
<td>sessionid=90198e1525e4b03797f833ff4320af39</td>
<td>200</td>
</tr>
<tr>
<td>14:10:09</td>
<td>172.16.22.33</td>
<td>POST</td>
<td>/confirm.asp</td>
<td>sessionid=90198e1525e4b03797f833ff4320af39</td>
<td>200</td>
</tr>
</tbody>
</table>
Phishing?

- No indications of key logging trojans, malware, viruses, etc. were found on the victim’s computer.
- Look what we found in the archived .pst file:

```
From:  & customer-service@___com
To:    
Cc:    
Subject: Valued Customer Feedback

At we are always striving to improve the customer’s online experience. We are currently experimenting with a new user interface and have selected a few of our valued customers to provide feedback on it.

We would appreciate it if you, Mr. , would review the changes by logging into your account by clicking here and sending us e-mail telling us what you like and what you don’t about what you see.

We appreciate your participation in this process.
Thank You once again,

Customer Service
```

URL: https://www.xyzbrokerage.com/login.asp?sessionid=90198e1525e4b03797f833ff4320af39
Session Fixation

1. HTTP/1.1 200 OK
   - GET /default.jsp
   - Set Cookie: sessionid=3f67e89u76g89aa7V

2. User supplies valid credentials
   - Login page is rendered to the user

3. Attacker crafts phishing email with the following link embedded in it:
   - http://www.victim.com/login.asp;session=3f67e89u76g89aa7V

4. Victim clicks on the legitimate link

5. User is logged in and sessionid 3f67e89u76g89aa7V is associated with the session

6. Attacker browses to any page with the sessionid appended to the request to gain access
Why Reactive Technologies Fail...

![Graph showing responses over time]

**Responses Over Time**

- **Users**
  - Clicked Link
  - Entered Data

- **Hours**
  - 10:00 to 17:00

Intrepidus Group, Inc. © 2007
Thank You

Rohyt Belani CISSP, CISM
rohyt.belani@intrepidusgroup.com

Mike Zusman, CISSP
mike.zusman@intrepidusgroup.com