The Bug Hunter's Methodology

Application Hacking v1

Disclaimer

Many people can teach:

';alert('xss');//

Less people teach where to look for web bugs.

Herein lies some of my favorite tips, tools, and tricks...

About Me

- Father, husband, hacker, gamer, sometimes streamer.
- Almost two decades of offensive security
- Twitter | YouTube | Twitch

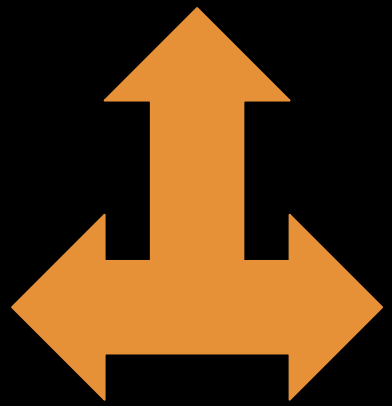
- Currently Playing:

DESTINY 2

PATH OF EXILE

DOTA 2

TBHM v4



Recon

Application Analysis

Agenda

Getting Started

Mental Hurdles

Pre-Manual Testing and Automation

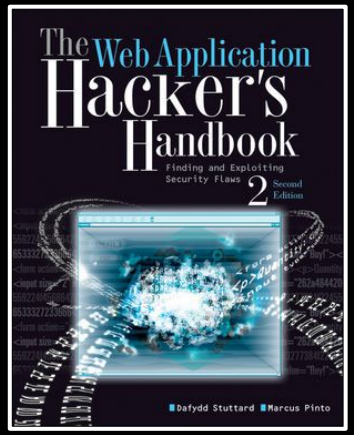
Content Discovery

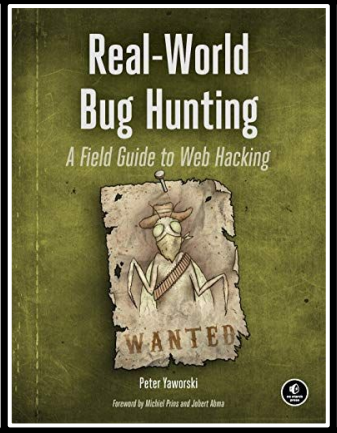
Application Analysis

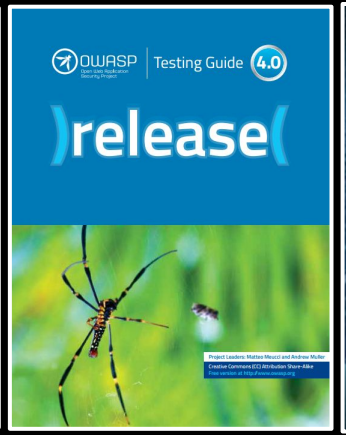
Getting Started

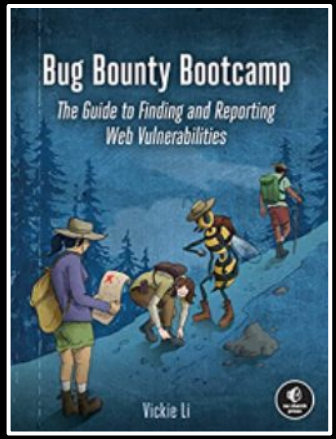
Beginning Your Journey

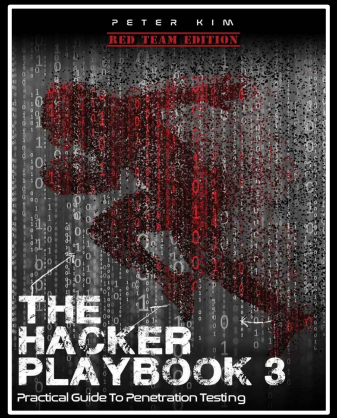
TBHM will attempt to give you tips, tricks, and tools related testing but there are many great wholistic texts available to supplement your app hacking journey.

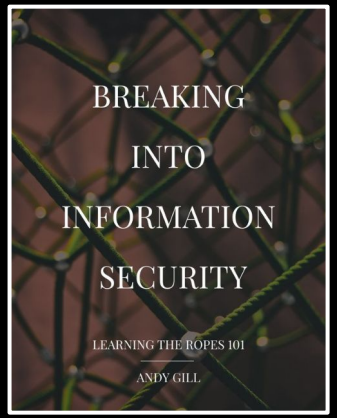


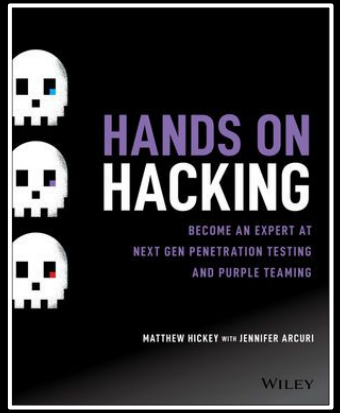


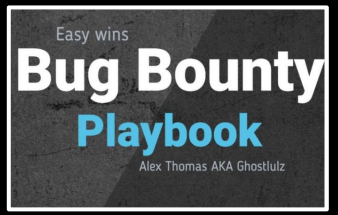


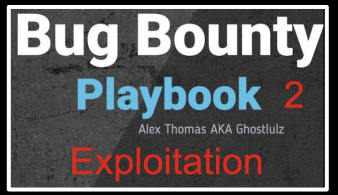












Beginning Your Journey

In addition to written text, in 2022, there are multitudes of free (or cheap) online labs to practice techniques and tools you see in TBHM. I recommend:

PentesterLab

Web Security Academy

HACKTHEBOX

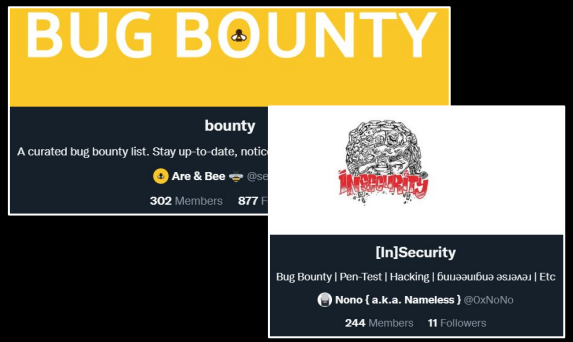
Vulnhub has over 400 crackme's to practice on.

OVWAD has over 180 applications to practice on.

Beginning Your Journey

Lastly, and *relatively* still a new, thing is hacking content creators. The amount of information you can learn from their feeds (Twitter, Twitch, and YouTube) is amazing.

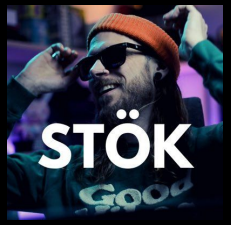
Check out:



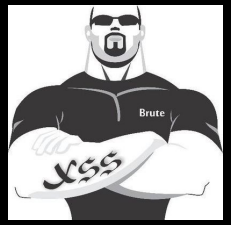
https://twitter.com/i/lists/937662149962461184

https://twitter.com/i/lists/1253517962272743424

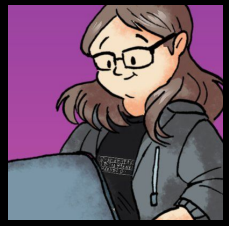

@danielmiessler



@stok



@brutelogic


@InsiderPhD


@infosec_au

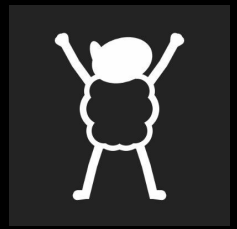

@Farah_Hawaa

@zseano


@hacker_


@hakluke


@albinowax


@tomnomnom

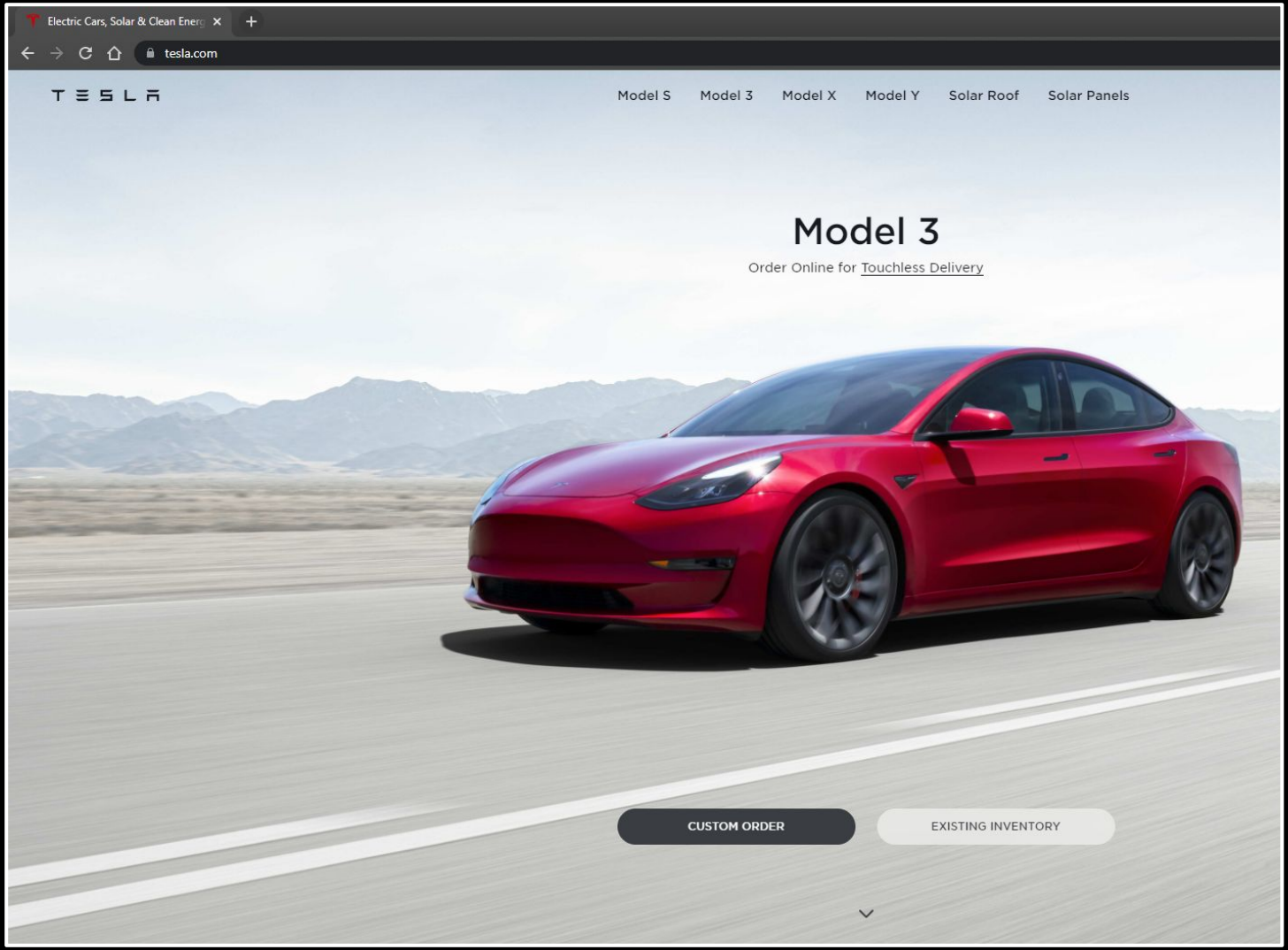

@_JohnHammond


@ippsec

@nahamsec

Mental Hurdles

Client Reputation



GitHub Security - GitHub Bug Bo

bounty.github.com

GitHub Security

Rules Targets Scope Rewards FAQs Submit a vulnerability

GitHub Security Bug Bounty

Software security researchers are increasingly engaging with internet companies to hunt down vulnerabilities. Our bounty program gives a tip of the hat to these researchers and provides rewards of $30,000 or more for critical vulnerabilities.

If you have found a vulnerability, submit it here.

You can find useful information in our rules, scope, targets and FAQ sections.

Happy hacking!

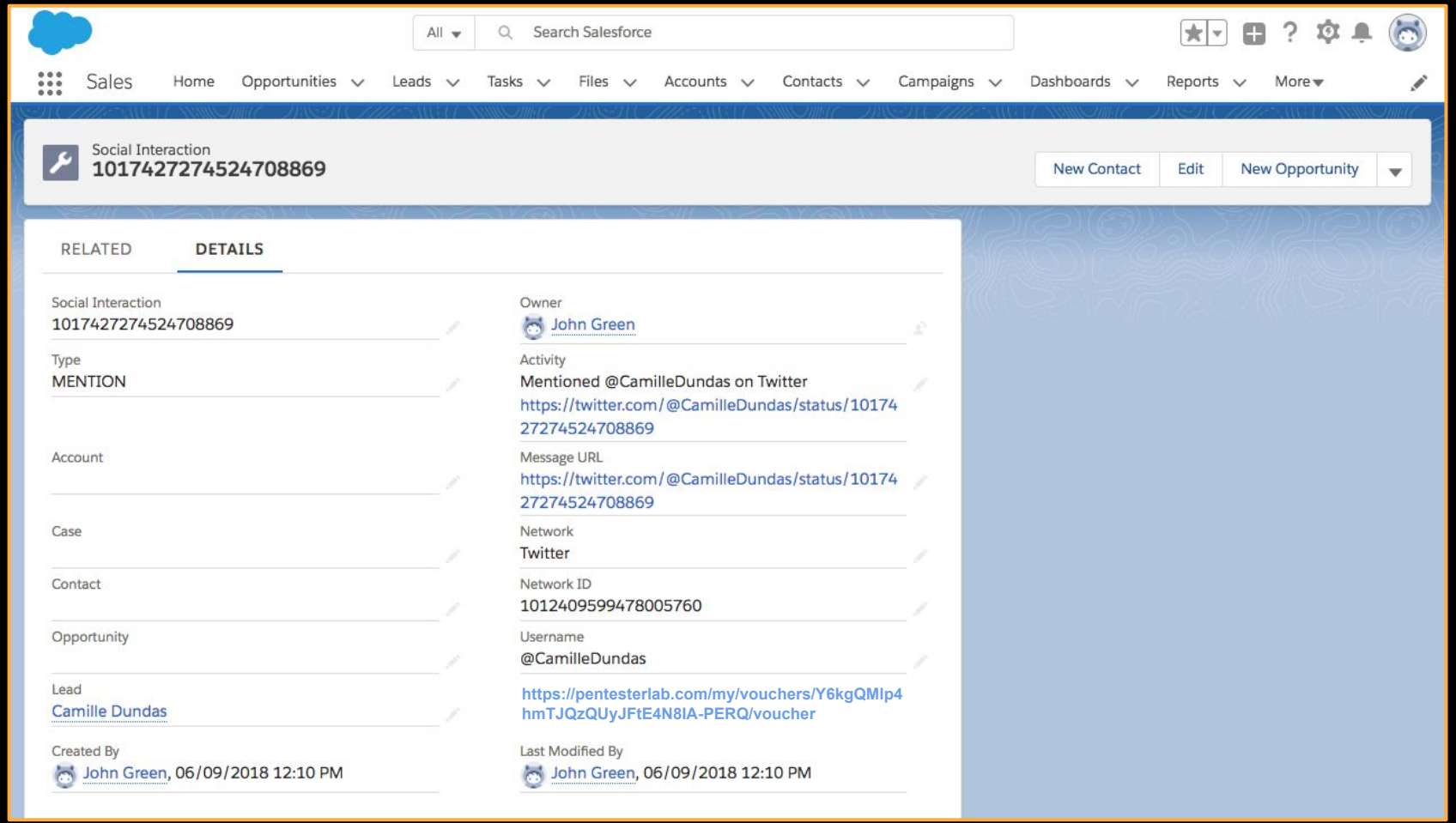
Leaderboard

These are the current top 10 bounty hunters based on total points earned across all targets. For the full list of contributors, check out GitHub's bounty hunters.

1	Aleksandr Dobkin<img src=404 onerror=alert(document.domain)> @adob	30,750 pts	
2	joernchen	28,500 pts	

Pre-Testing

Size

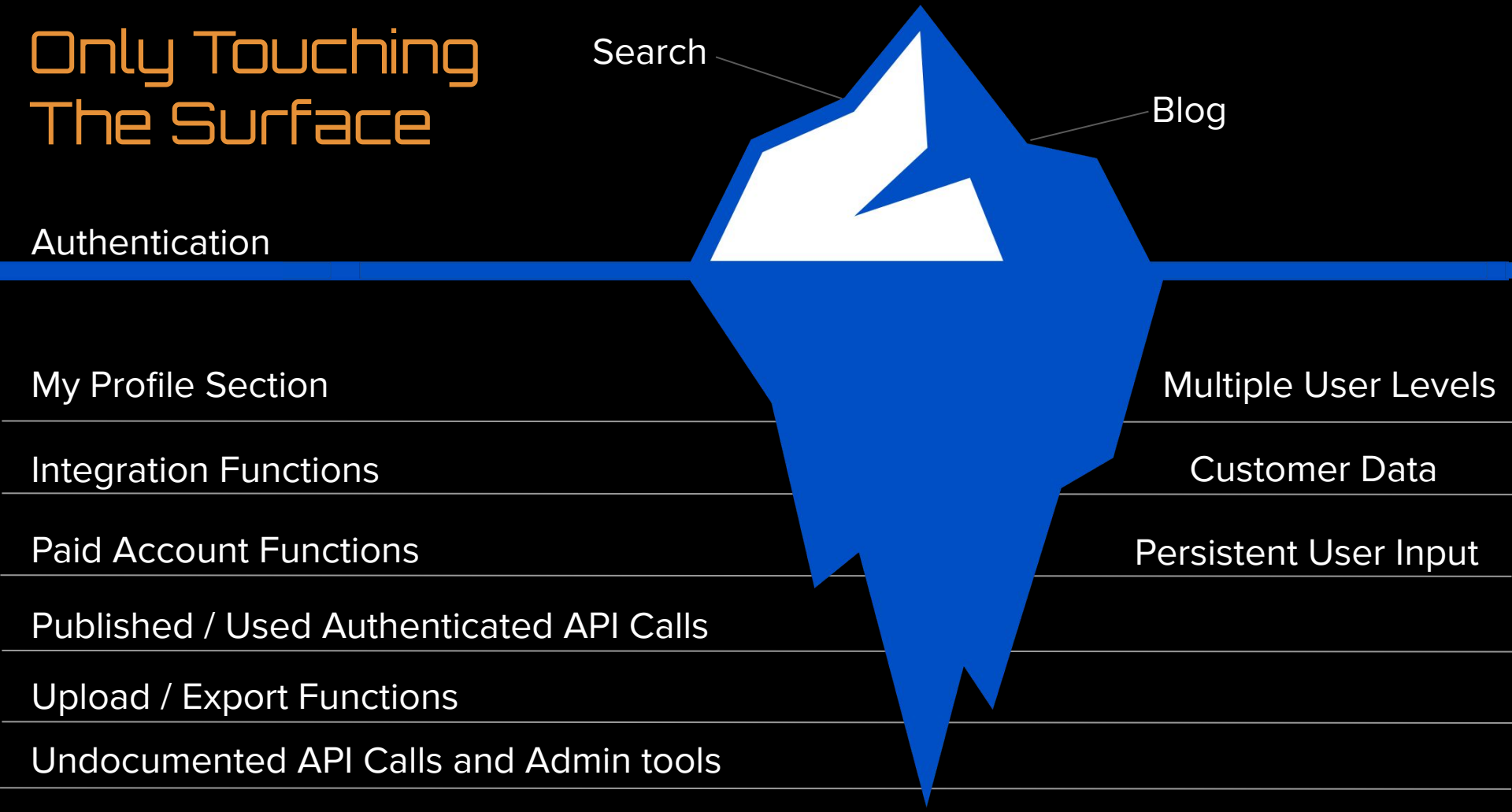

All
Search Salesforce
Sales
Home
Opportunities
Leads
Tasks
Files
Accounts
Contacts
Campaigns
Dashboards
Reports
More
Social Interaction
10174272745247088 69
New Contact
Edit
New Opportunity
RELATED
DETAILS
Social Interaction
10174272745247088 69
Type
MENTION
Account
Case
Contact
Opportunity
Lead
Camille Dundas
Created By
John Green, 06/09/2018 12:10 PM
Owner
John Green
Activity
Mentioned @CamilleDundas on Twitter
https://twitter.com/@CamilleDundas/status/10174 27274524708869
Message URL
https://twitter.com/@CamilleDundas/status/10174 27274524708869
Network
Twitter
Network ID
1012409599478005760
Username
@CamilleDundas
https://pentesterlab.com/my/vouchers/Y6kgQMIp4 hmTJQzQUyJFtE4N8IA-PERQ/voucher
Last Modified By
John Green, 06/09/2018 12:10 PM

Self Hosted

OSS

COTS

Only Touching The Surface



Pre-Manual Testing and Automation

- Tech Profiling
- Finding CVE's and Misconfigs
- Port Scan

Testing Layers

Open Ports and Services {Default creds on services, service level exploits}

Web Hosting Software {Default creds, Web server misconfigurations, web exploits}

Application Framework {...}

Application: Custom Code or COTS {...}

Application Libraries (usually javascript) {...}

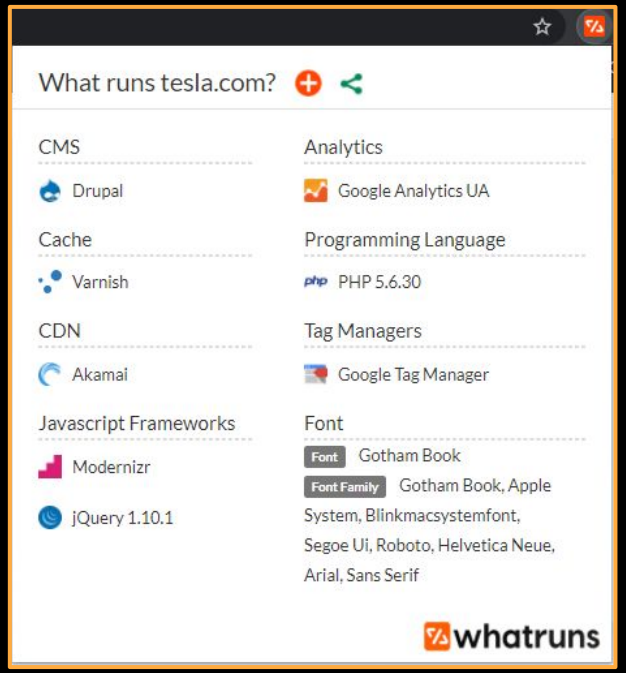
Integrations

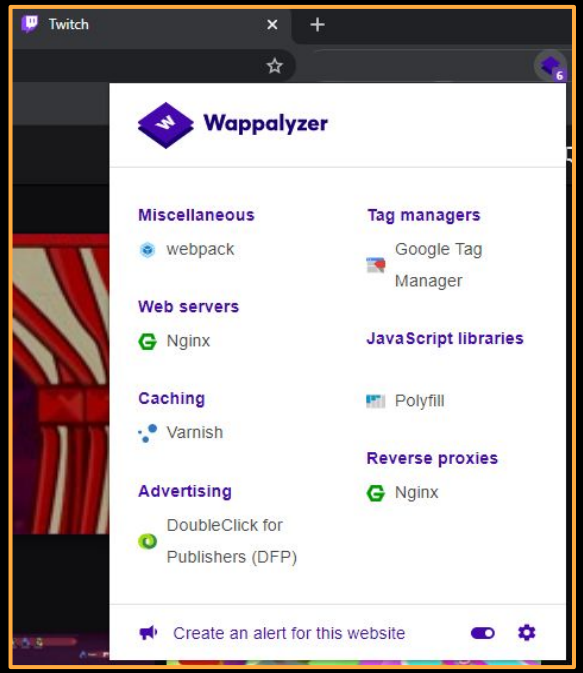
Tech Profiling (Browser Extensions)

You can use several different browser extensions to find information about your target.

whatruns

Wappalyzer





Tech Profiling (webanalyze)

You can also use command line tools (to integrate into your automation) if you wish to.

```
root@TBox4:~# webanalyze -host https://www.twitch.tv -crawl 2
 :: webanalyze        : v1.0
 :: workers           : 4
 :: apps              : apps.json
 :: crawl count       : 2
 :: search subdomains : true

https://www.twitch.tv (0.7s):
    Polyfill,  (JavaScript libraries)
    Nginx,  (Web servers, Reverse proxies)
    Varnish,  (Caching)
```

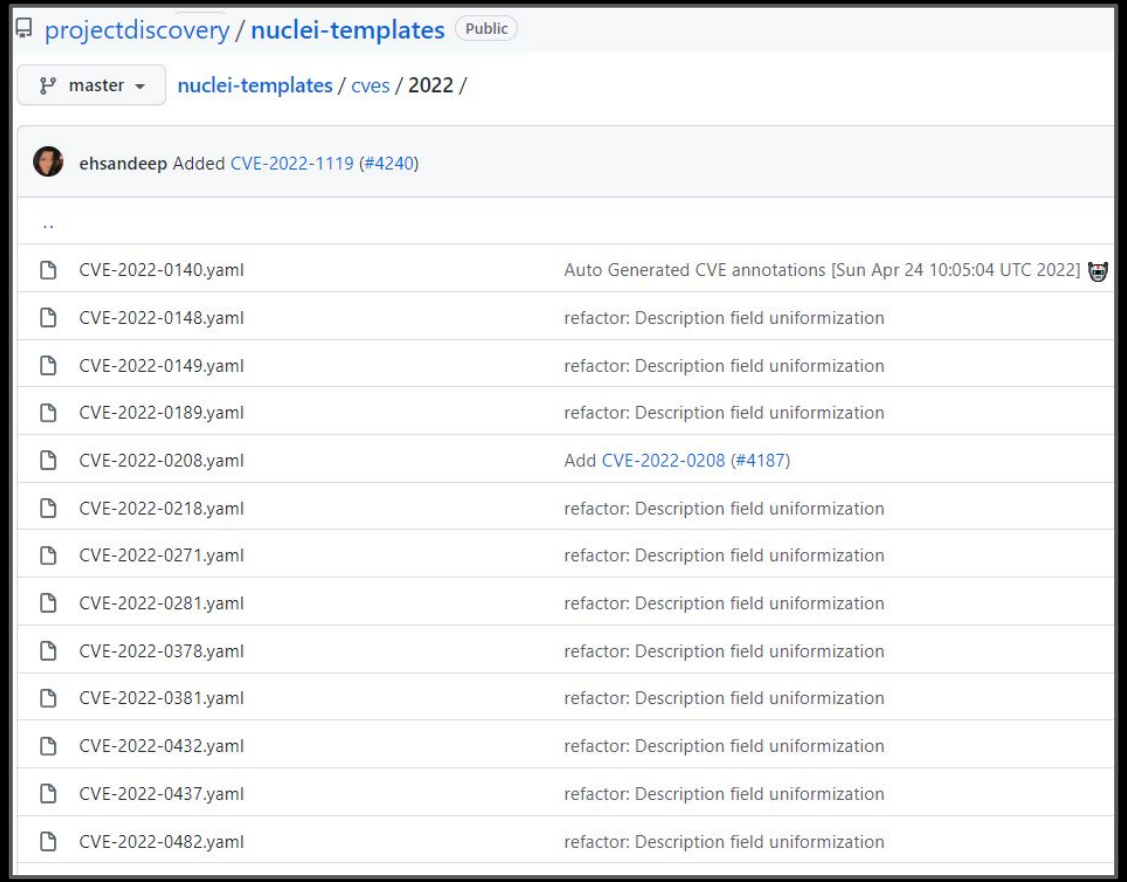
Finding CVE's and Misconfigs

Having completed recon on an assessment or bounty we want to analyze if targets have:

- Known vulnerabilities
- Framework login pages
- Default creds
- And more

related to **non-custom code**.

These are usually platform, application server, framework, CMS, library, and misconfiguration related.



Finding CVE's and Misconfigs (Nuclei)

Nuclei Scanner is a tool by the Project Discovery team.

- 1000+ CVE's
- 100+ Informational detections
- 500+ admin panel detectors
- 1500+ other checks
  - creds/keys
  - 67 subdomain takeover
  - Http form brute force
  - 3428 total templates


```
nuclei -l tesla_httprobe.txt -t brute-force/* -t cves/* -t basic-
detections/* -t dns/* -t files/* -t panels/* -t security-misconfiguration/*
-t subdomain-takeover/* -t technologies/* -t tokens/* -t vulnerabilities/*
```

```
[content-delivery-network:akamai] [http] https://auth.tesla.com/
[content-delivery-network:akamai] [http] http://auth.tesla.com/
[content-delivery-network:akamai] [http] https://3.tesla.com/
[content-delivery-network:akamai] [http] http://3.tesla.com/
[ntlm-directories] [http] http://autodiscover.tesla.com/powershell/
[web-server:ms-iis] [http] http://autodiscover.tesla.com/
[web-server:apache] [http] https://employeefeedback.tesla.com/
[content-delivery-network:akamai] [http] http://employeefeedback.tesla.com/
[web-server:apache] [http] https://feedback.tesla.com/
[content-delivery-network:akamai] [http] http://feedback.tesla.com/
[content-delivery-network:akamai] [http] https://edr.tesla.com/
```

Finding CVE's and Misconfigs (Others)

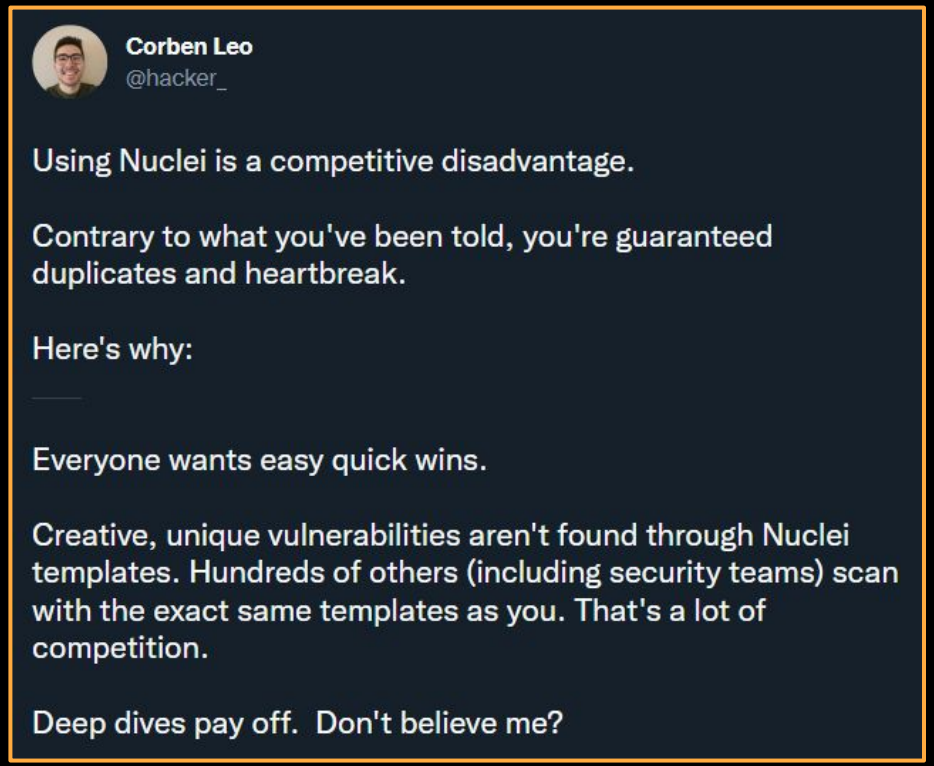
Several other scanning tools exist in this realm. None are completely comprehensive so you end up having to run multiple tools or "port" checks to your preferred tool (they are all still more app-centric than something like Nessus):

```
"github") && StringSearch("response", "There isn't a GitHub Pages site here.")
"github") && StringSearch("response", "For root URLs (like http://example.com/) you must pr
"heroku") && StringSearch("response", "There's nothing here")
"heroku") && StringSearch("response", "No such app")
"tumblr.com") && StringSearch("response", "There's nothing here.")
"tumblr.com") && StringSearch("response", "Whatever you were looking for doesn't currently
"myshopify.com") && StringSearch("response", "Only one step left!")
"pageserve.co") && StringSearch("response", "You've Discovered A Missing Link. Our Apologie
"tictail.com") && StringSearch("response", "Building a brand of your own?")
"createsend.com") && StringSearch("response", "Trying to access your account?")
"cargocollective.com") && StringSearch("response", "404 Not Found")
"pantheon.io") && StringSearch("response", "404 error unknown site")
"fly.io") && StringSearch("response", "404 Not Found")
"uptimerobot.com") && StringSearch("response", "page not found")
"strikinglydns.com") && StringSearch("response", "page not found")
"tilda.cc") && StringSearch("response", "Please renew your subscription")
"azurewebsites.net") && StringSearch("response", "404 Web Site not found")
"NoSuchBucket")
"The specified bucket does not exist")
"smartling.com") && StringSearch("response", "Domain is not configured")
"acquia.com") && StringSearch("response", "If you are an Acquia Cloud customer and expect t
"fastly.net") && StringSearch("response", "Please check that this domain has been added to
"fastly.net") && StringSearch("response", "Fastly error: unknown domain:")
"pantheonsite.io") && StringSearch("response", "The gods are wise")
```

(Jaeles Subdomain takeover signatures)

Gofingerprint by Tanner Barnes	23 (panel checks)
Sn1per by @xer0dayz	102 active checks 26 passive checks
Intrigue Core by jcran	50 panel checks (mapped with metasploit modules too)
Vulners (Burp Ext)	292 Technology profiles (panels and library checks)
Jaeles Scanner by j3ssi3jjj	28 Panel Checks 37 CVE's 7 injection testers (LFI, XSS, SQLi, ++) 44 Sub Takeover
retire.js	83 Vulnerabilities related to javascript libs and frameworks

Finding CVE's and Misconfigs (tips)



Corben Leo
@hacker_

Using Nuclei is a competitive disadvantage.

Contrary to what you've been told, you're guaranteed duplicates and heartbreak.

Here's why:

Everyone wants easy quick wins.

Creative, unique vulnerabilities aren't found through Nuclei templates. Hundreds of others (including security teams) scan with the exact same templates as you. That's a lot of competition.

Deep dives pay off. Don't believe me?

Corben is right, but in a certain context.

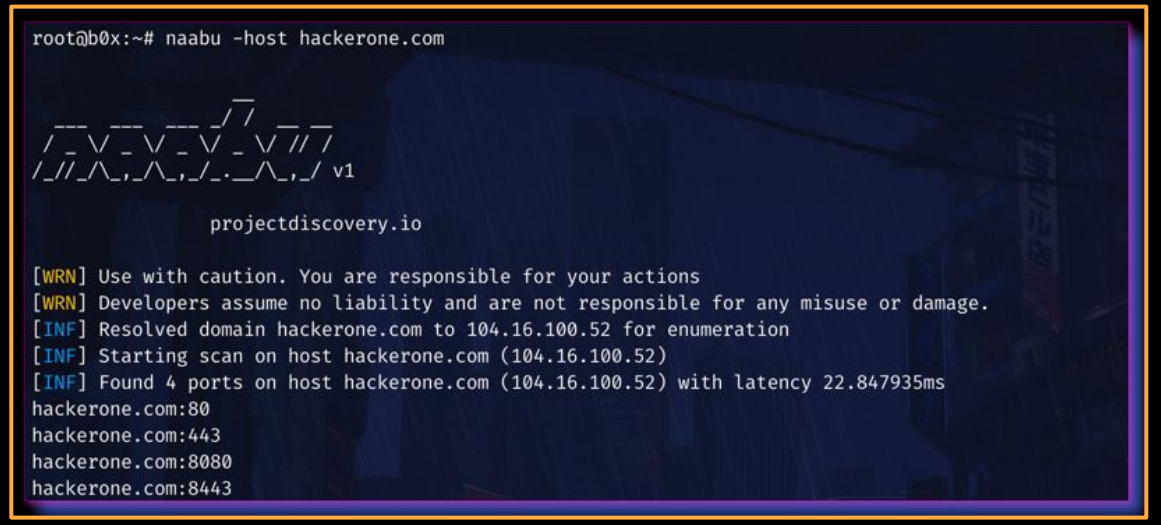
If you are testing a main site or highly known target, then yes, scanning has probably been done already.

However, if you are testing targets that are “fresh” or found via recon and you have a feeling not many testers have seen them before, it can absolutely find great things.

In addition, making templates for new vulns or content discovery techniques in Nuclei is SUPER easy.

Port Scanning (tips)

- Fast And Simple SYN/CONNECT probe based scanning
- Optimized for ease of use and lightweight on resources
- Automatic IP deduplication for port scan
- NMAP integration for service discovery
- Multiple input support - STDIN/HOST/IP/CIDR
- Multiple output format support - JSON/TXT/STDOUT



Content Discovery

Content Discovery Tools

Gobuster v3.1.0

Gobuster is a tool used to brute-force:

- URIs (directories and files) in web sites.
- DNS subdomains (with wildcard support).
- Virtual Host names on target web servers.
- Open Amazon S3 buckets

ffuf - Fuzz Faster U Fool

dirsearch

dirsearch - Web path discovery

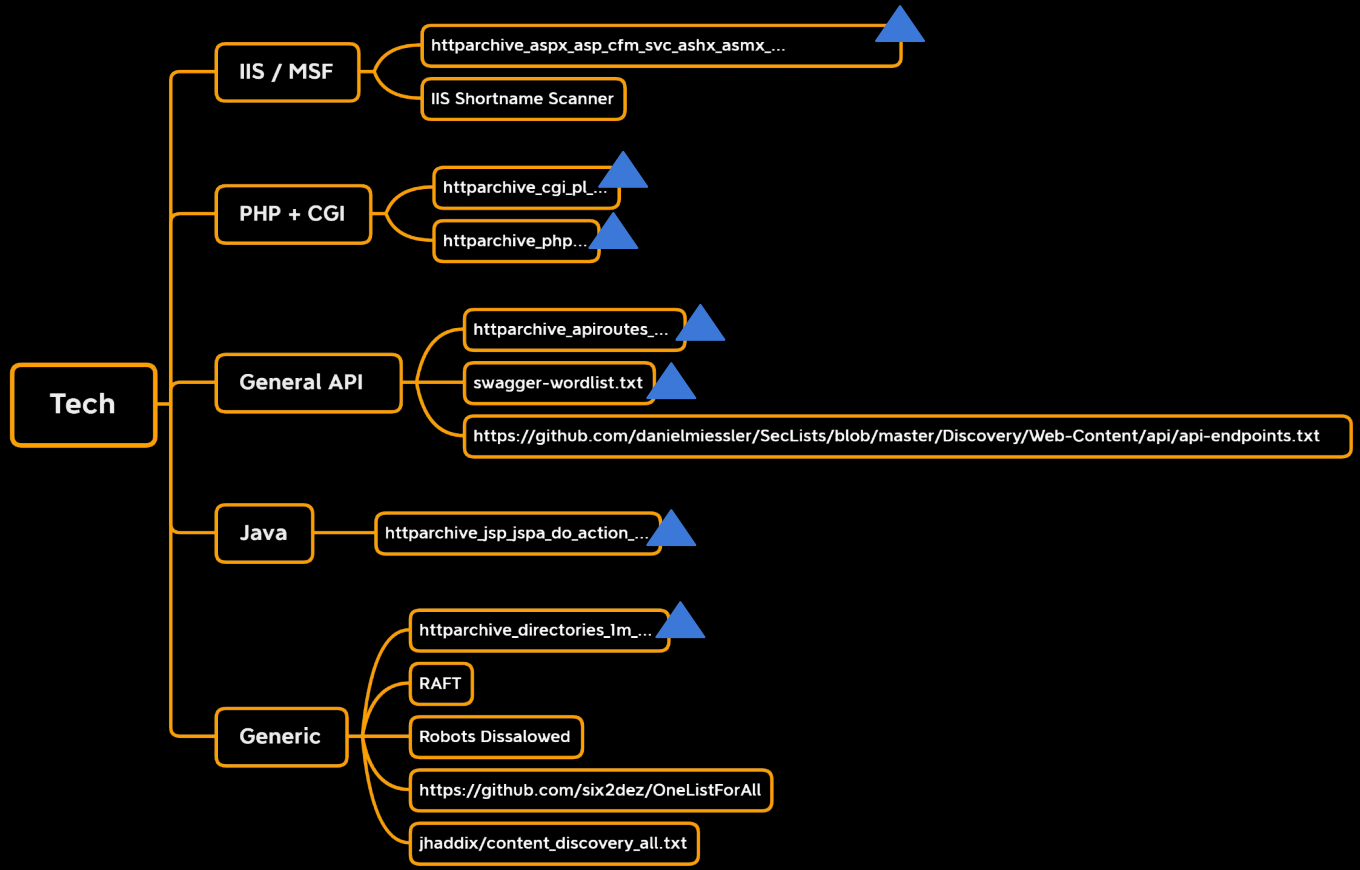
Wfuzz: The Web fuzzer

pypi v3.1.0 | license GPLv2 | python 3.4 | 3.5 | 3.6 | 3.7 | 3.8 | codecov 71%

Wfuzz provides a framework to automate web applications security assessments and could help you to secure your web applications by finding and exploiting web application vulnerabilities.

Content Discovery Lists

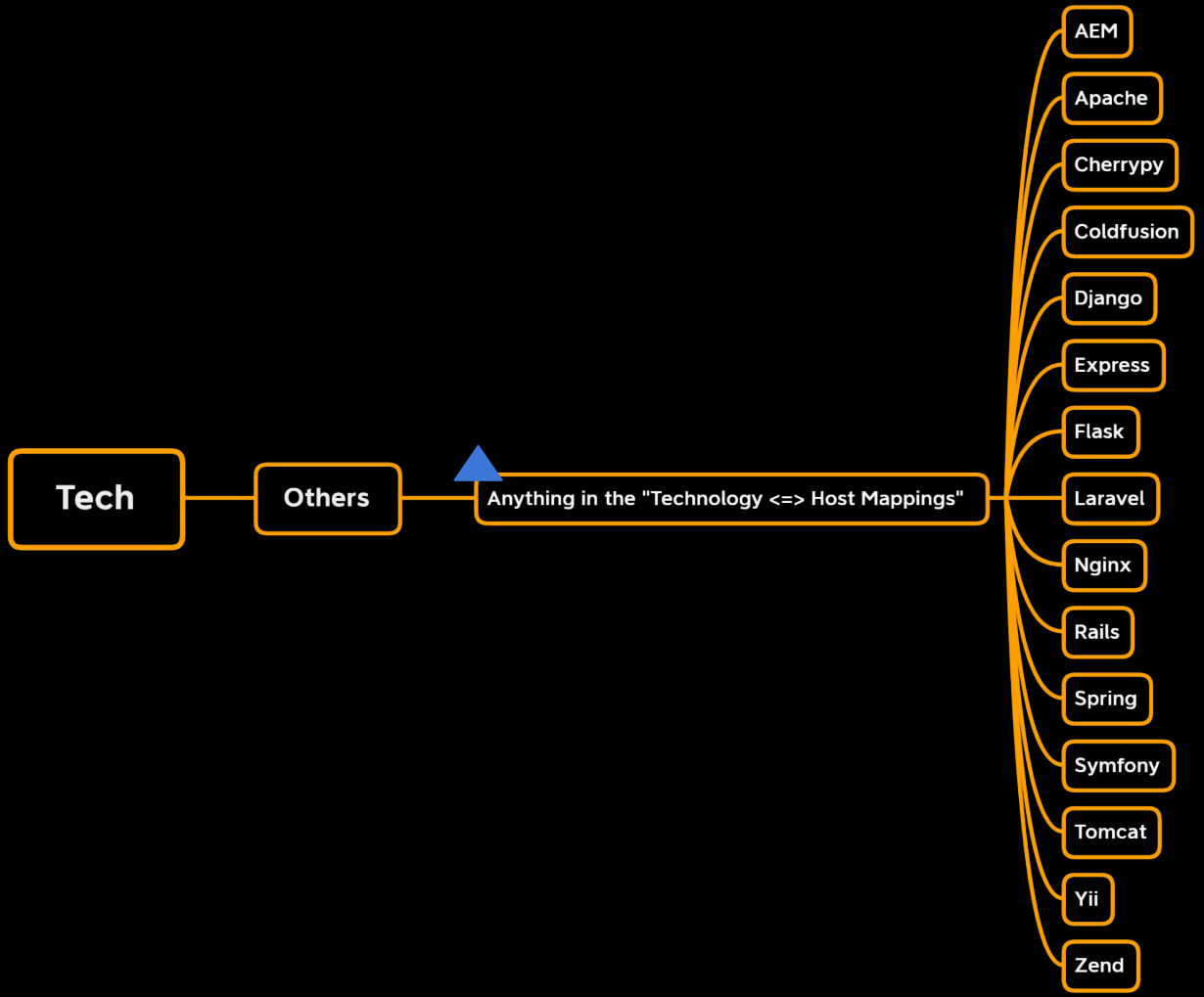
(Technology)



wordlists.assetnote.io

Content Discovery Lists

(Technology)



Content Discovery (Technology tips)

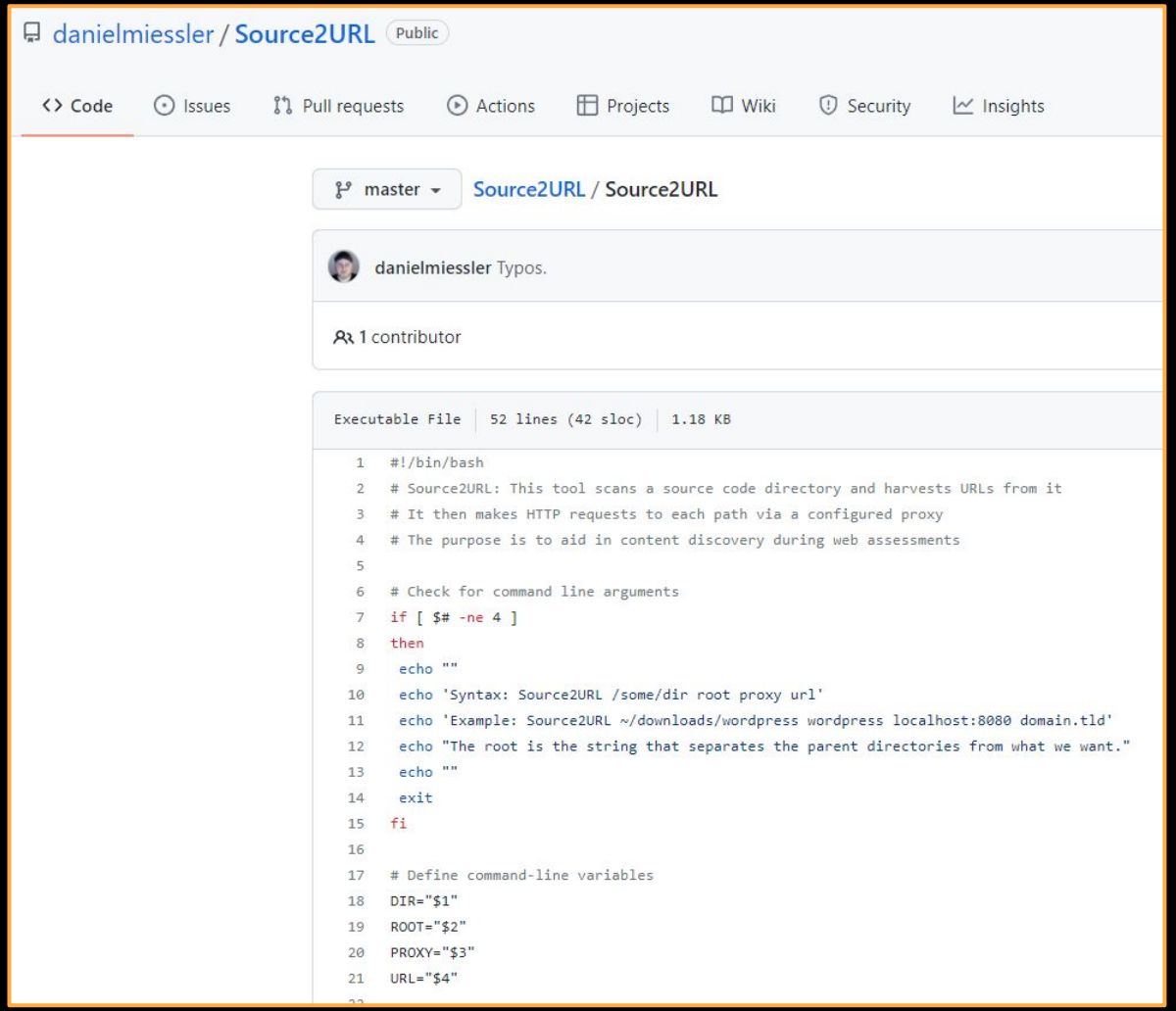
Pay special attention to:

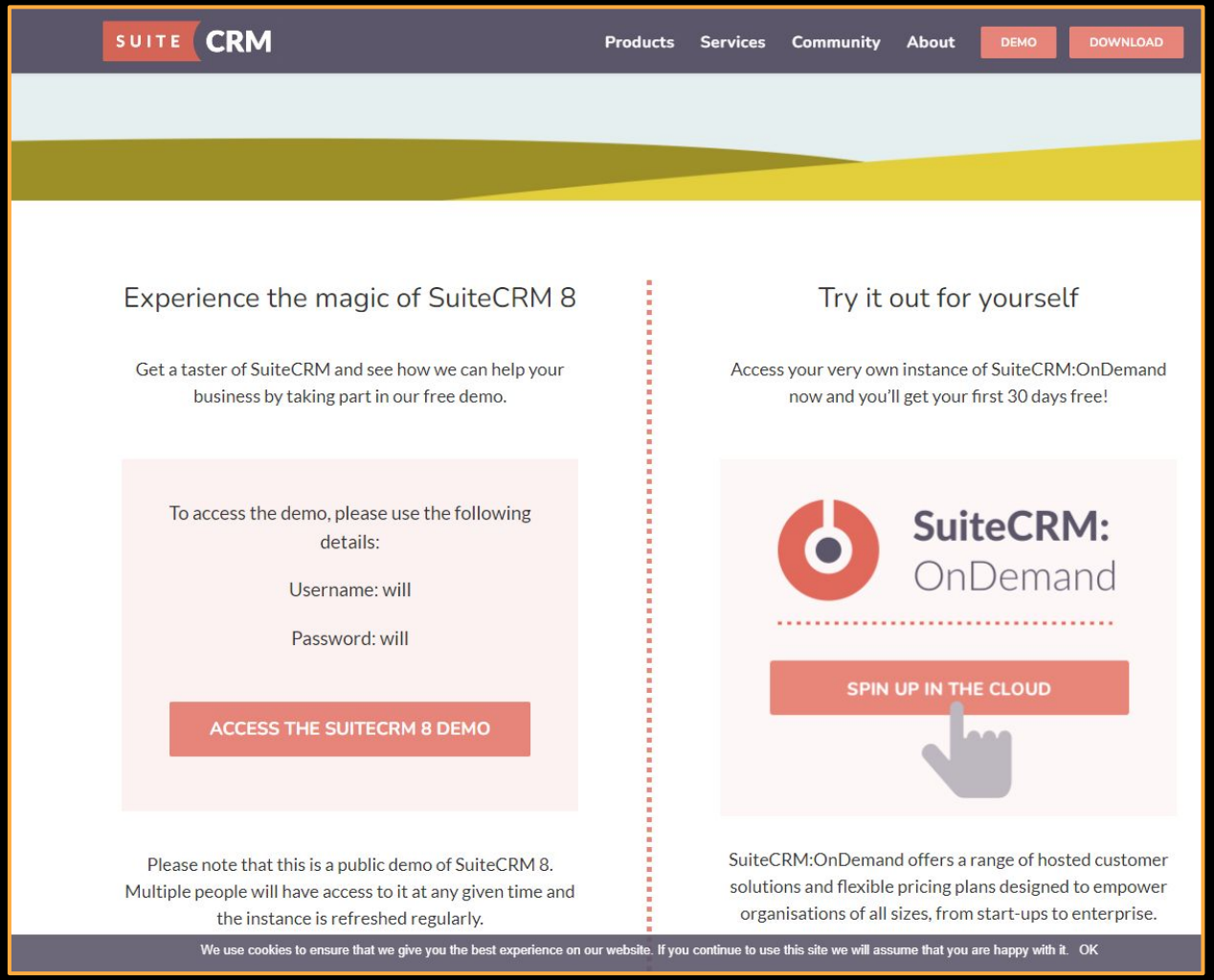
- Config files for DB connections
- Where the admin login and routes/endpoints are

Appsettings.json in a ASP.NET app

Content Discovery Lists

(OSS)



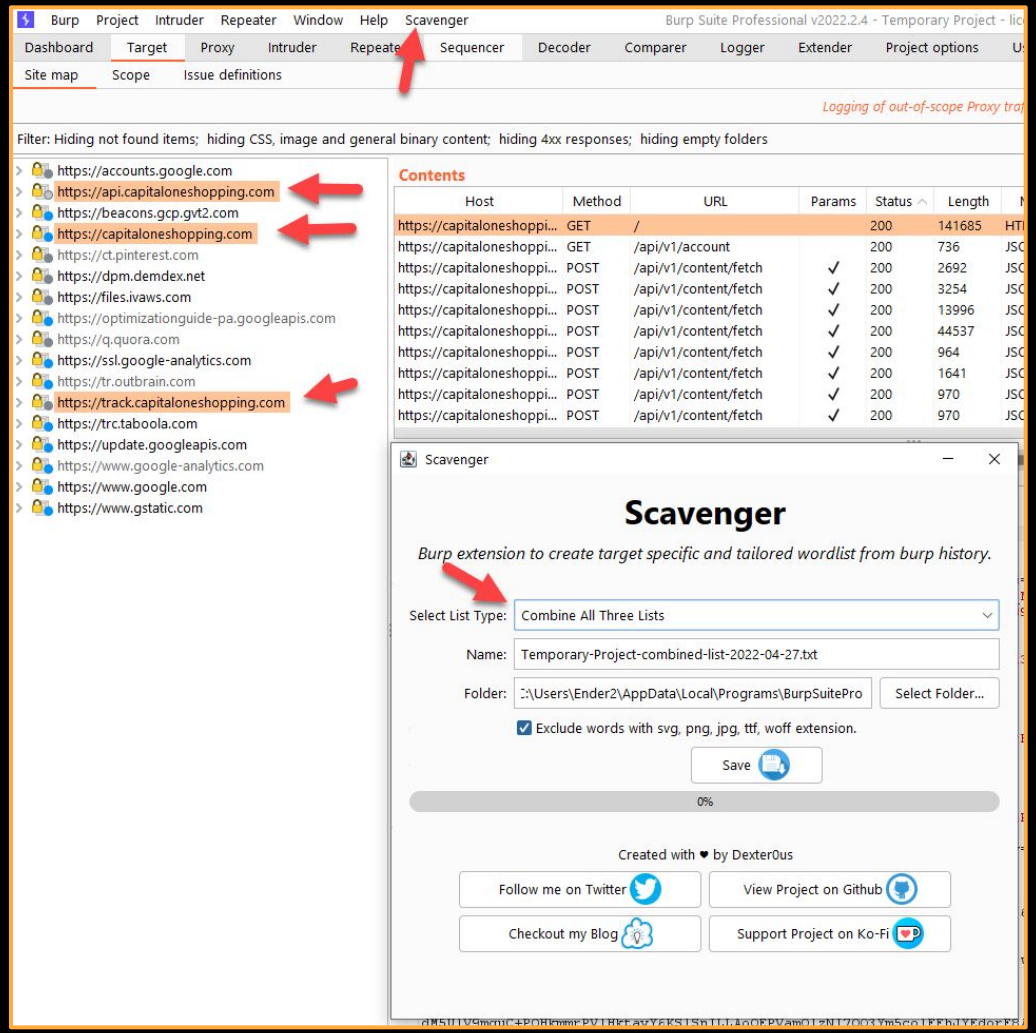


Content Discovery Lists

(COTS / Paid)

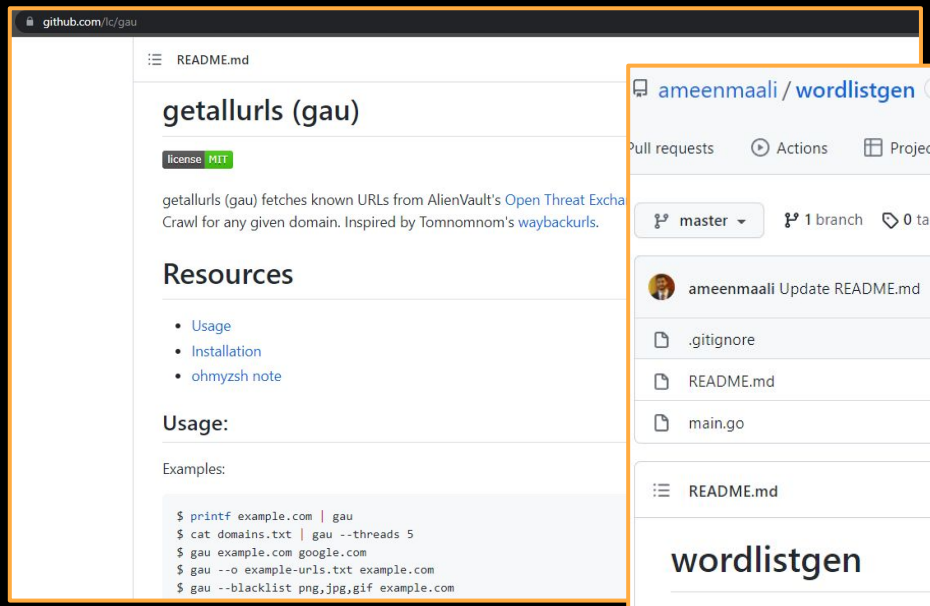
Content Discovery Lists (Custom)

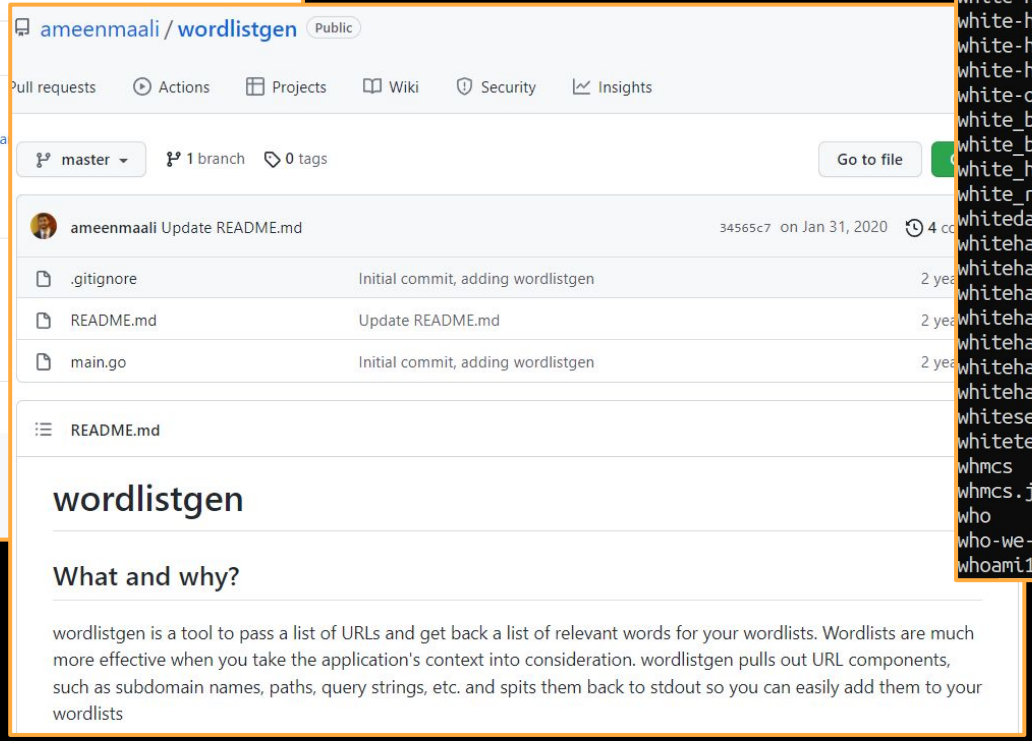
github.com/0xDexter0us/Scavenger



Content Discovery Lists (Historical)

echo bugcrowd.com | gau | wordlistgen | sort -u






```
whats-in-my-hacking
whats-the-buzz-octo
whatsabugworth-upda
whatsabugworth__1_.
when-humans-and-aut
when-to-reward-a-bu
when-to-reward-a-bu
whereissecure
whhackersbr
whit3h4t
white-house-cyberse
white-house-cyberse
white-house-cyberse
white-house-takes-a
white-outreach-logo
white_bg.jpg
white_bg1.jpg
white_hat
white_rabz
whitedaemon
whitehat007
whitehat3086
whitehat32
whitehat_sec
whitehathckr
whitehatsmile
whitehattushu
whitesector
whitetester
whmcs
whmcs.jpg
who
who-we-are
whoami13
```

Shoutout:

github.com/michael1026/trashcompactor

Content Discovery (Tip - Recursion)

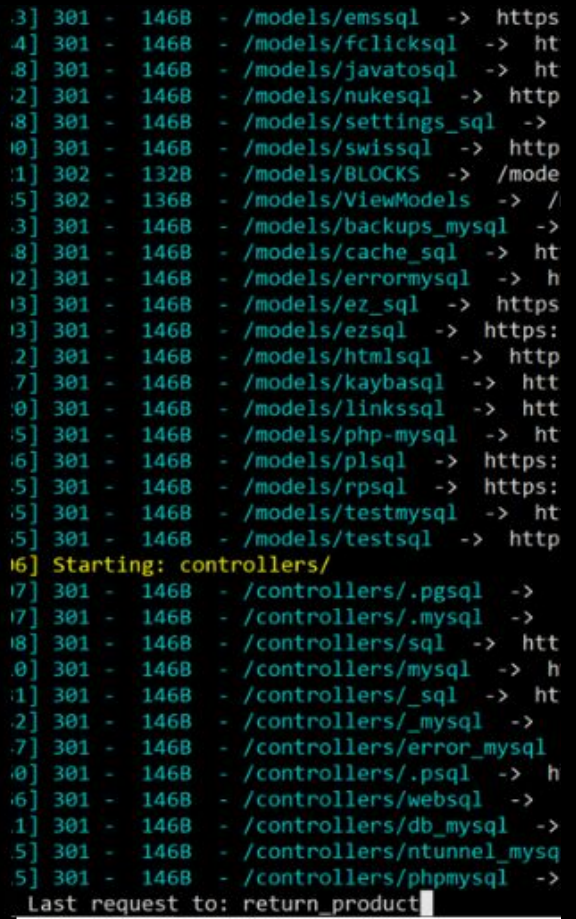
Bounty Tip

02

"Content discovery" is trying to guess sensitive paths and files that might exist in the application but are not linked anywhere.

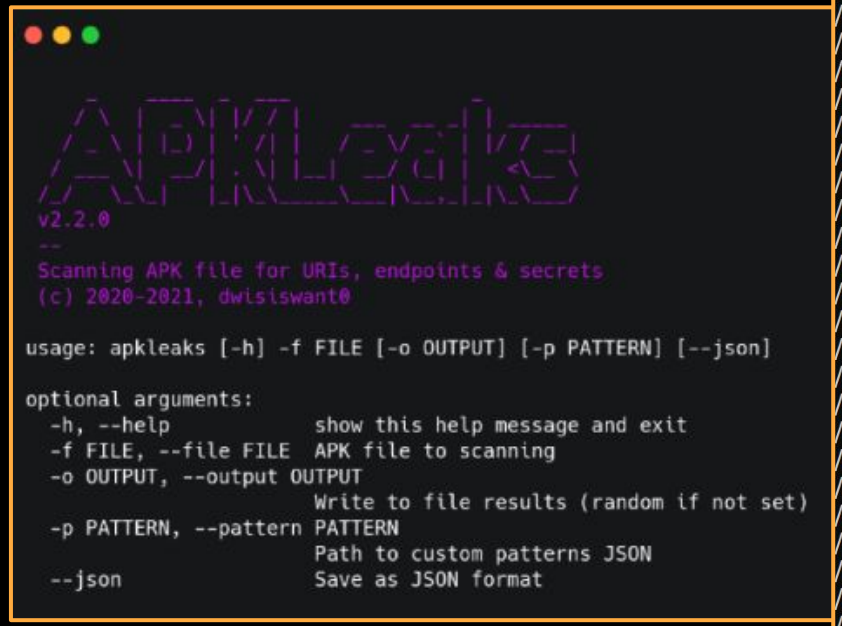
Often when doing this part of web assessment you will run into 401 Not Authorized responses. It is beneficial to recursively brute force that path. Often resources past that path have not had the same access controls applied. In addition, 401 replies should be investigated with waybackmachine (https://archive.org/web/) to see if they ever did not have authentication applied and to garner clues about the application pathing.

https://someapp.com/admin/ 401
https://someapp.com/admin/dashboard/ 401
https://someapp.com/admin/dashboard/members 200



Content Discovery (Tip - Endpoints from mobile)

Often scope is based off of a domain. Many times the mobile API is hanging off the main domain. So any endpoints you get from a mobile app might be in scope.



github.com/dwisiswant0/apkleaks

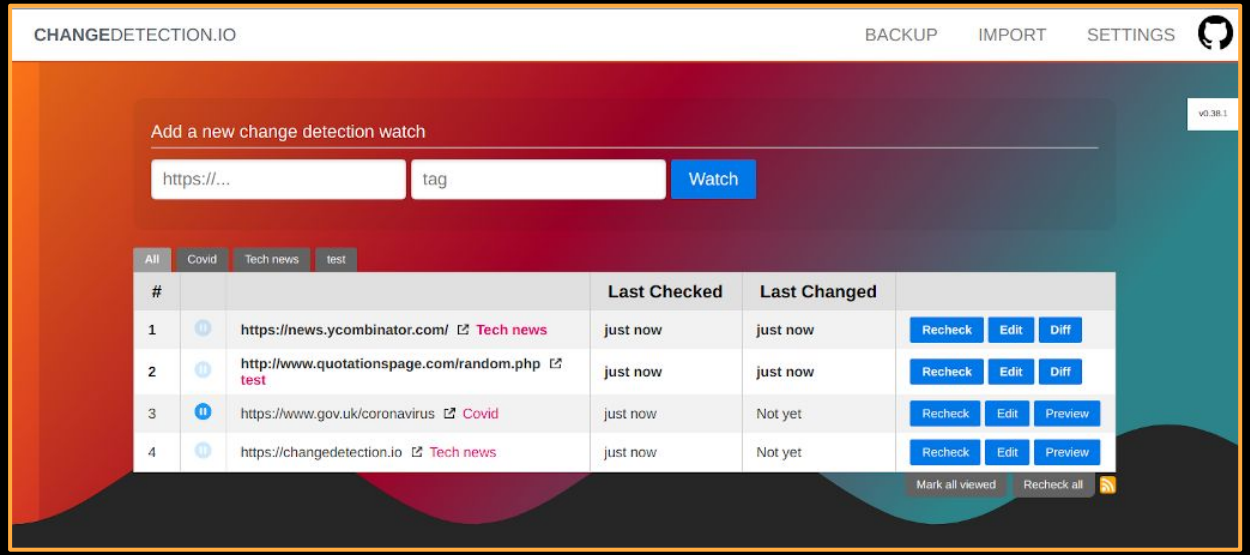
```
/#access_token
/?error=access_denied
/rt/mobile/action-execution-log
/rt/risk/verifyidentity
/rt/external-rewards/create-link/
/rt/external-rewards/delete-link/
/rt/external-rewards/get-account-linking-sc
/rt/external-rewards/get-celebration-screer
/rt/external-rewards/get-program-details-sc
/rt/external-rewards/get-programs/
/rt/finprod/finprod-rewards-eligibility/eli
/rt/finprod/finprod-rewards-eligibility/eli
/rt/gifting/get-gift-details
/rt/gifting/get-landing-page
/rt/gifting/get-purchase-page
/rt/gifting/get-purchased-gifts
/rt/gifting/get-redemption-page
/rt/gifting/purchase-gift-card
/rt/gifting/redeem
/rt/gifting/send-gift-email
/rt/riders/log-hub-user-interaction
/rt/communications/get-unsubscriptions
/rt/communications/set-unsubscriptions
/rt/payments-compliance/v1/oe/hydrate
/rt/payments-compliance/v1/uc/submit
/rt/payments-compliance/v2/uc/submit
/rt/mobile-integration-test/{name}
/rt/rewards/get-client-gaming
```

ubereats.com

Content Discovery (Tip - Changes)

Sometimes being the 1st one to a new function on a target is pivotal. There are several ways to stay informed of new things for your target:

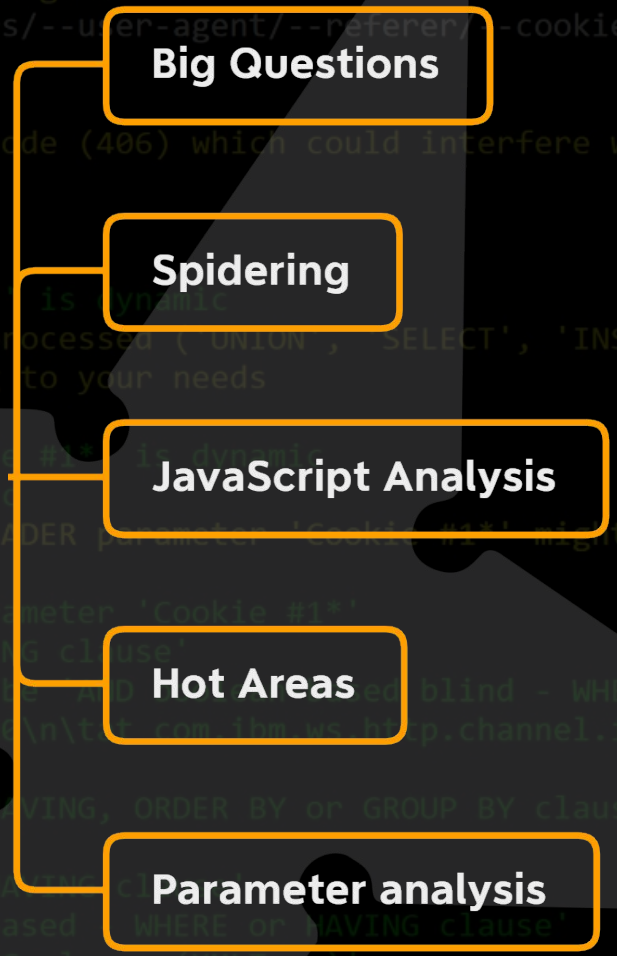
- Targets Newsletter
- Affiliate Programs
- Googling Conference Talks
- Monitoring the domain for code changes



github.com/dgtlmoon/changedetection.io

Ty patrik <3

Application Analysis



The Big 7 Questions (#1)

How does the app pass data?

resource?parameter=value&param2=value

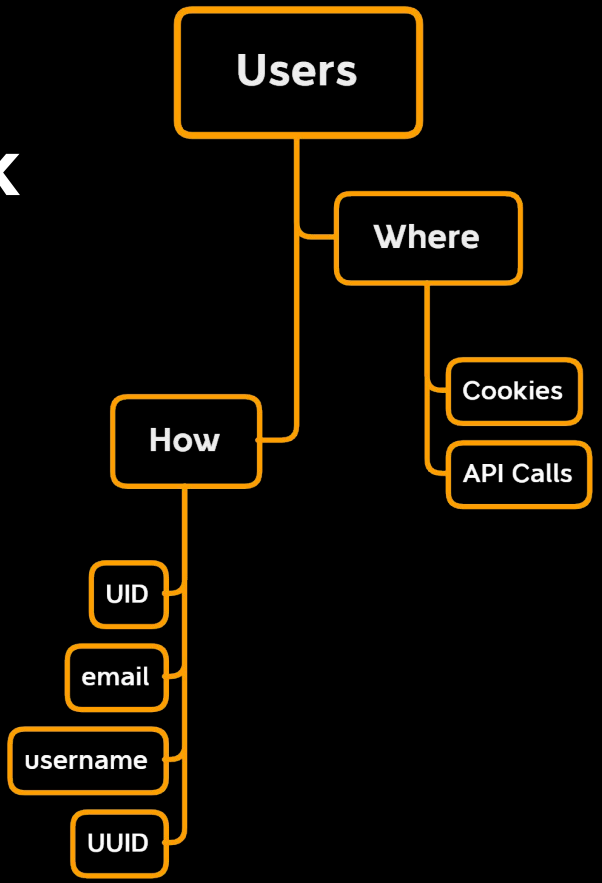
Method -> /route/resource/sub-resource/...

Understanding this will be the cornerstone of how you test for vast categories of bugs. The bugs will be there, but if you're not familiar with where to inject your payloads, you will fail.

The Big 7 Questions (#2)

How/where does the app talk about users?

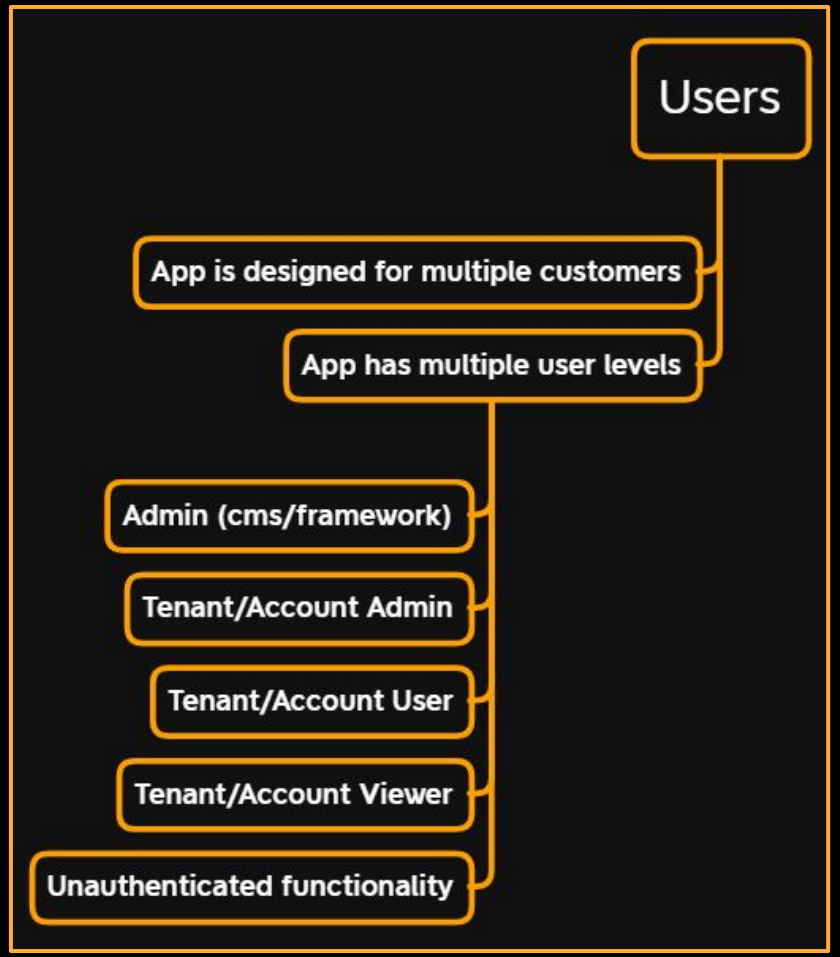
Understanding how users (yourself and other users) are referenced and where in the application is pivotal to finding several bug classes, most specifically Access, Authorization, Logic, and Information Disclosure bugs.



The Big Questions (#3)

Does the site have multi-tenancy or user levels?

This will also dictate how we test for authorization and access bugs.

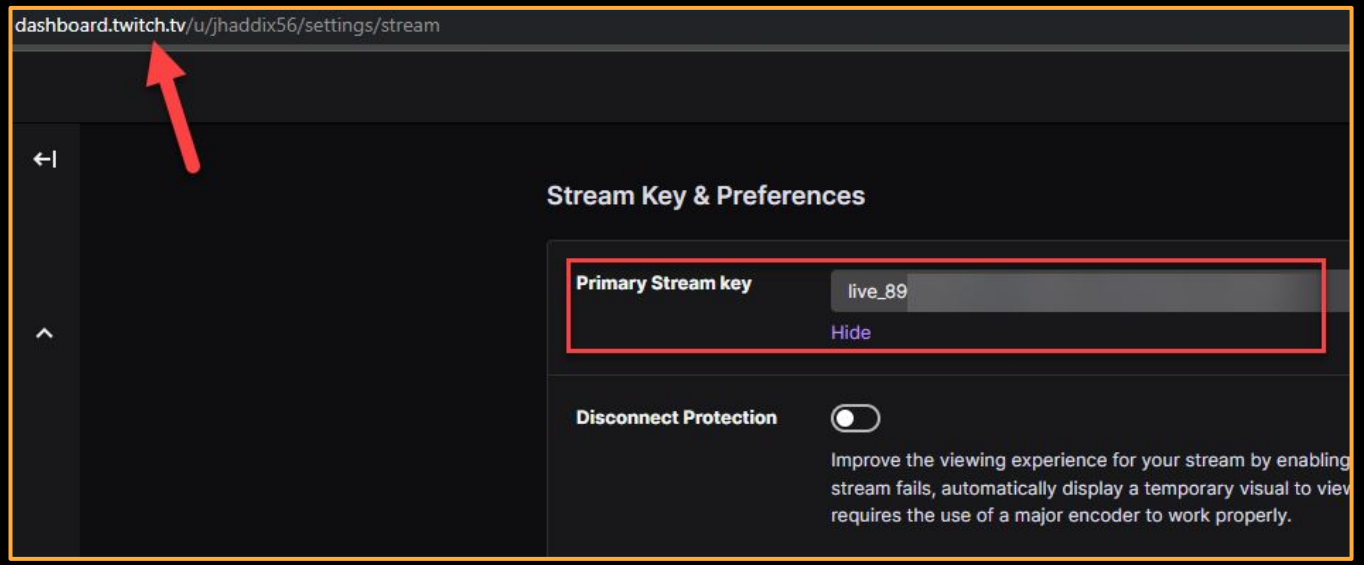


The Big Questions (#4)

Does the site have a unique threat model?

If the application houses more than the standard PII data, it's easy to forget to target that data in your testing.

Examples: API keys, application data for doxing.

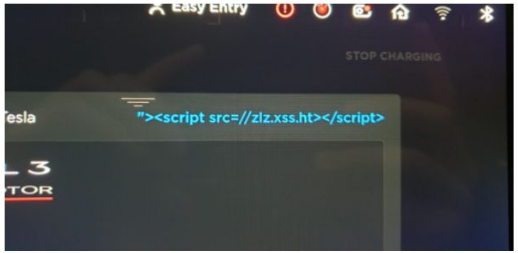


The Big Questions (#5)

Has there been past security research & vulns?

https://samcurry.net/cracking-my-windshield-and-earning-10000-on-the-tesla-bug-bounty-program/

After spending more time messing with the input I saw that the allowed content length for the input was very long. I decided to name the Tesla my XSS hunter payload and continued toying around with the other functionalities on the car.



My idea for setting this name was that it may show up on some internal Tesla website for vehicle management or possibly from a functionality within my

The DOM DOOM XSS

We had to invent the DOM DOOM XSS. After some googling it seems like it has n been done before, but feel free to enlighten us. We gladly give credit where credit due.

The technical aspect of this is pretty straightforward. We hosted a web version of DOOM at a domain we own (feel free to try it out) and went on to find a DOM XS Tesla. The DOM XSS was soon thereafter found at forums.tesla.com *(it should be n is a self xss, meaning very limited potential impact).*

https://labs.detectify.com/2017/07/27/how-we-invented-the-tesla-dom-doom-xss/

bugcrowd.com/tesla/crowdstream?filter=disclosures

Tesla

Accelerating the world's transition to sustainable energy

$100 – $15,000 per vulnerability · Partial safe harbor

Submit report · Do you like this program?

Program details · Announcements 1 · CrowdStream · Hall of Fame

All submissions · Disclosed reports

Disclosed report: postMessage XSS in Tesla Payment page
By TheTime · Program Tesla · Reward $500 · Priority P3
Disclosed on 22 Feb 2022

Disclosed report: testdrive form user data info leak via [3rdparty] exploit
By 1hack0 · Program Tesla · Priority P1
Disclosed on 23 Aug 2021

Disclosed report: Password change does not invalidate API keys
By MatthiasU · Program Tesla · Reward $500 · Priority P3
Disclosed on 22 Mar 2021

Disclosed report: config files with vpn pre-shared-key and other credentials
By phfboi · Program Tesla · Reward $10,000 · Priority P1
Disclosed on 26 Feb 2021

Disclosed report: Authentication Bypass through HTTP Request Smuggling
By riramar · Program Tesla · Priority P3
Disclosed on 10 Feb 2021

CrowdStream › postMessage XSS in Tesla Payment page

postMessage XSS in Tesla Payment page

Disclosed by TheTime

Program	Disclosed date	Reward
Tesla	22 Feb 2022 2 months ago	$500

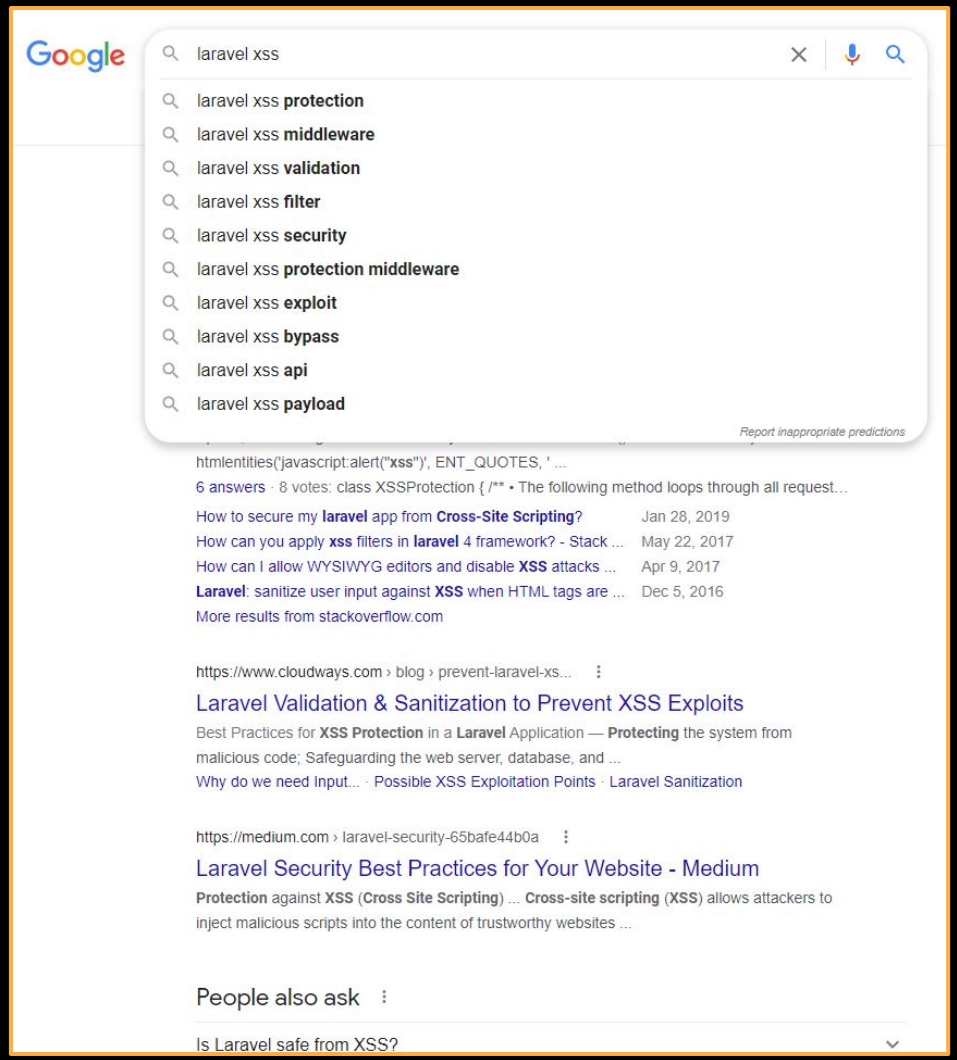
Summary by TheTime

postMessage XSS in Tesla's payment pages

Report details

Submitted	25 May 2021 16:20:10 UTC
Target Location	*.tesla.com
Target category	Website Testing
VRT	Cross-Site Scripting (XSS) > Reflected > Non-Self
Priority	P3
Bug URL	https://www.tesla.com/ro_ro/model3/design#payment

https://bugcrowd.com/disclosures/aac249ea-fe92-4b43-98e9-dda021c0ff4d/postmessage-xss-in-tesla-payment-page



The Big Questions (#6)

How does the app handle:

XSS ? : Google the web framework to understand (or test yourself on a custom application) how it protects from XSS or what output encoding options it uses.

CSRF ?

Code Injection ? (SQL, Template, ++)

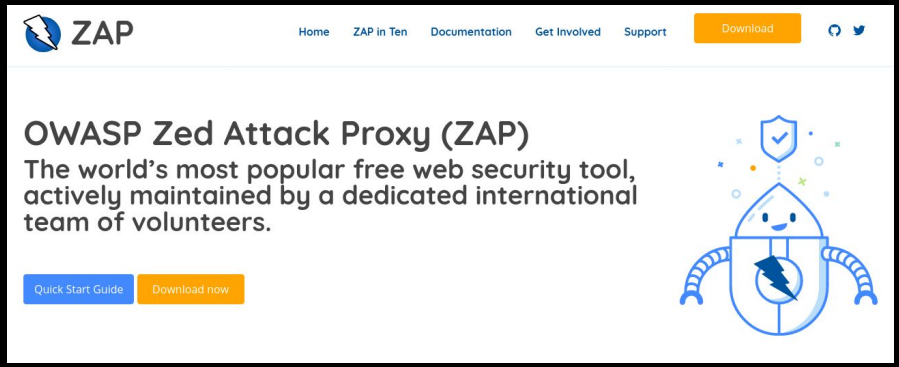

[05:06:24] [INFO] loading tamper script 'xforwardedfor'
[05:06:24] [WARNING] using too many tamper scripts is usually not a good idea
custom injection marking character ('*') found in option '--headers/--user-agent/--referer/--cookie'
. Do you want to process it? [Y/n/q] Y
[05:06:28] [INFO] testing connection to the target URL
[05:06:29] [WARNING] the web server responded with an HTTP error code (406) which could interfere wi
th the results of the tests
[05:06:29] [INFO] testing if the target URL is stable
[05:06:29] [INFO] target URL is stable
[05:06:29] [INFO] testing if (custom) HEADER parameter 'Cookie #1*' is dynamic
[05:06:29] [WARNING] currently only couple of keywords are being processed ('UNION', 'SELECT', 'INSE
RT', 'UPDATE', 'FROM', 'WHERE'). You can set it manually according to your needs
[05:06:30] [WARNING] reflective value(s) found and filtering out
[05:06:30] [INFO] confirming that (cust
[05:06:31] [INFO] (custom) HEADER param
[05:06:31] [WARNING] heuristic (basic) test shows that (custom) HEADER parameter 'Cookie #1*' might
not be injectable
[05:06:34] [INFO] testing for SQL injection on (custom) HEADER parameter 'Cookie #1*'
[05:06:34] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[05:06:39] [INFO] (custom) HEADER parameter 'Cookie #1*' seems to be 'AND boolean-based blind - WHER
E or HAVING clause' injectable (with --string="\\xa0\\xa0\\xa0\\xa0\n\tat com.ibm.ws.http.channel.in
bound.impl.HttpInboundLink.ready(HttpInboundLink.java:287)")
[05:06:48] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause
'
[05:06:48] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[05:06:49] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause'
[05:06:49] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[05:06:49] [INFO] testing 'MySQL >= 5.0 error-based - Parameter replace'

Spidering

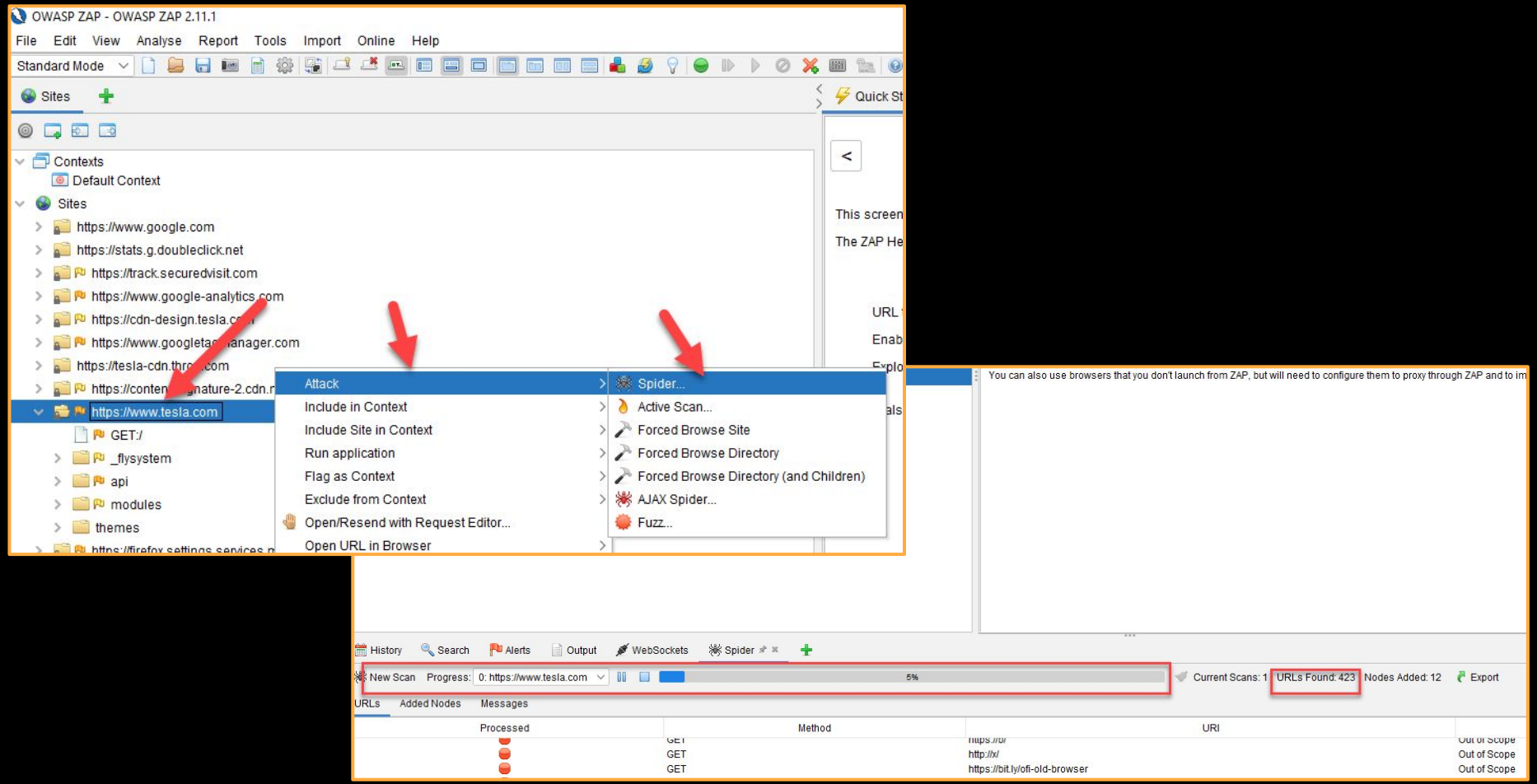
Spidering (Zap & Burp)

To ensure code coverage (finding all possible endpoints and parameters) you must spider the site.

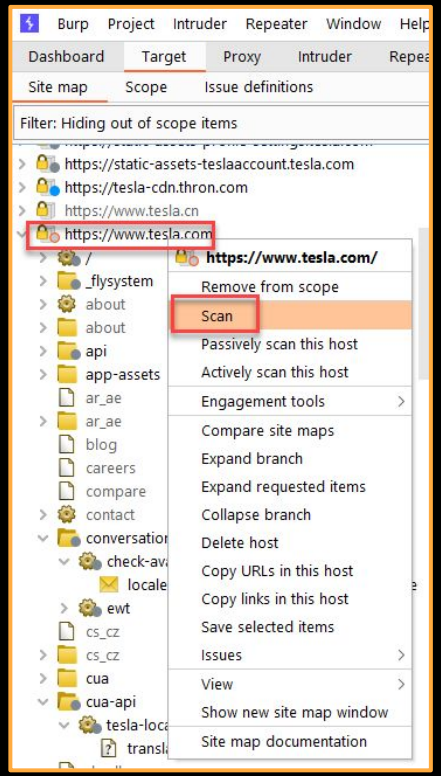
The most common way to do this is with and interceptions proxy. Which you should have been using to answer some of the Big Questions earlier.

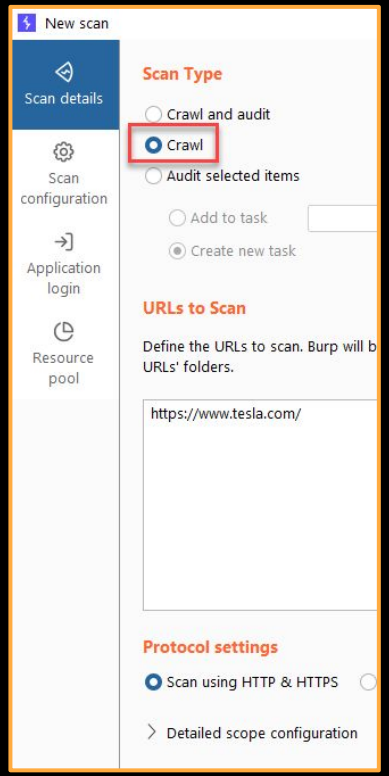


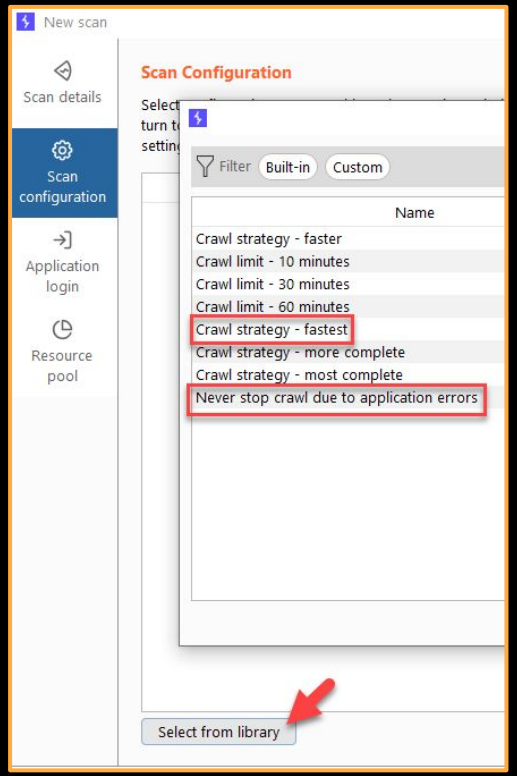
Spidering (Zap)

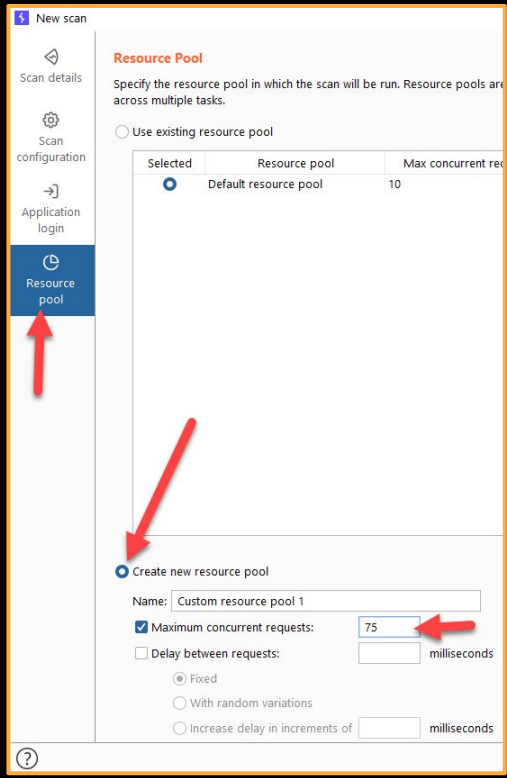


Spidering (Burp)





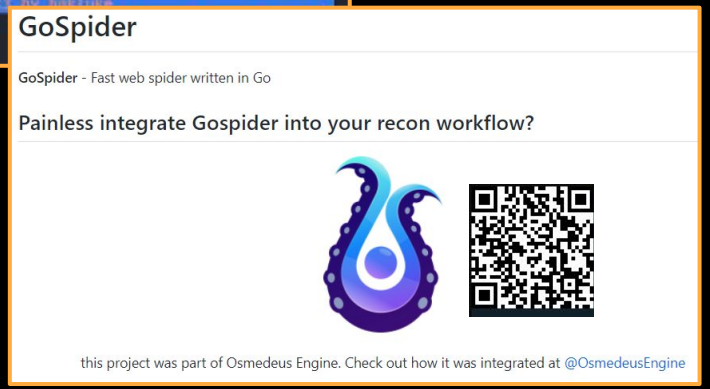




Spidering (on the command line)

GoSpider and hakcrawler remain my “go-to” tools for command line spidering ast they parse robots.txt files, js files, and more.

```
root@arcanum:~# gospider -s https://www.hackerone.com --depth 1
[url] - [code-200] - https://www.hackerone.com
[robots] - https://www.hackerone.com/core/*.css$
[robots] - https://www.hackerone.com/core/*.css?
[robots] - https://www.hackerone.com/core/*.js$
[robots] - https://www.hackerone.com/core/*.js?
[robots] - https://www.hackerone.com/core/*.gif
[robots] - https://www.hackerone.com/core/*.jpg
[robots] - https://www.hackerone.com/core/*.jpeg
[robots] - https://www.hackerone.com/core/*.png
[robots] - https://www.hackerone.com/core/*.svg
[robots] - https://www.hackerone.com/profiles/*.css$
[robots] - https://www.hackerone.com/profiles/*.css?
[robots] - https://www.hackerone.com/profiles/*.js$
[robots] - https://www.hackerone.com/profiles/*.js?
[robots] - https://www.hackerone.com/profiles/*.gif
[robots] - https://www.hackerone.com/profiles/*.jpg
[robots] - https://www.hackerone.com/profiles/*.jpeg
[robots] - https://www.hackerone.com/profiles/*.png
[robots] - https://www.hackerone.com/profiles/*.svg
[robots] - https://www.hackerone.com/core/
[robots] - https://www.hackerone.com/profiles/
[robots] - https://www.hackerone.com/README.txt
[robots] - https://www.hackerone.com/web.config
[robots] - https://www.hackerone.com/admin/
[robots] - https://www.hackerone.com/comment/reply/
[robots] - https://www.hackerone.com/filter/tips
[robots] - https://www.hackerone.com/node/add/
[robots] - https://www.hackerone.com/search/
[robots] - https://www.hackerone.com/user/register
[robots] - https://www.hackerone.com/user/password
[robots] - https://www.hackerone.com/user/login
[robots] - https://www.hackerone.com/user/logout
```




JavaScript

JavaScript Parsing (on the command line)

Linkfinder was the OG but now is integrated to many of the command line spiders on the previous slide. One thing to consider is that we still want to parse inline JavaScript (js NOT in .js files).

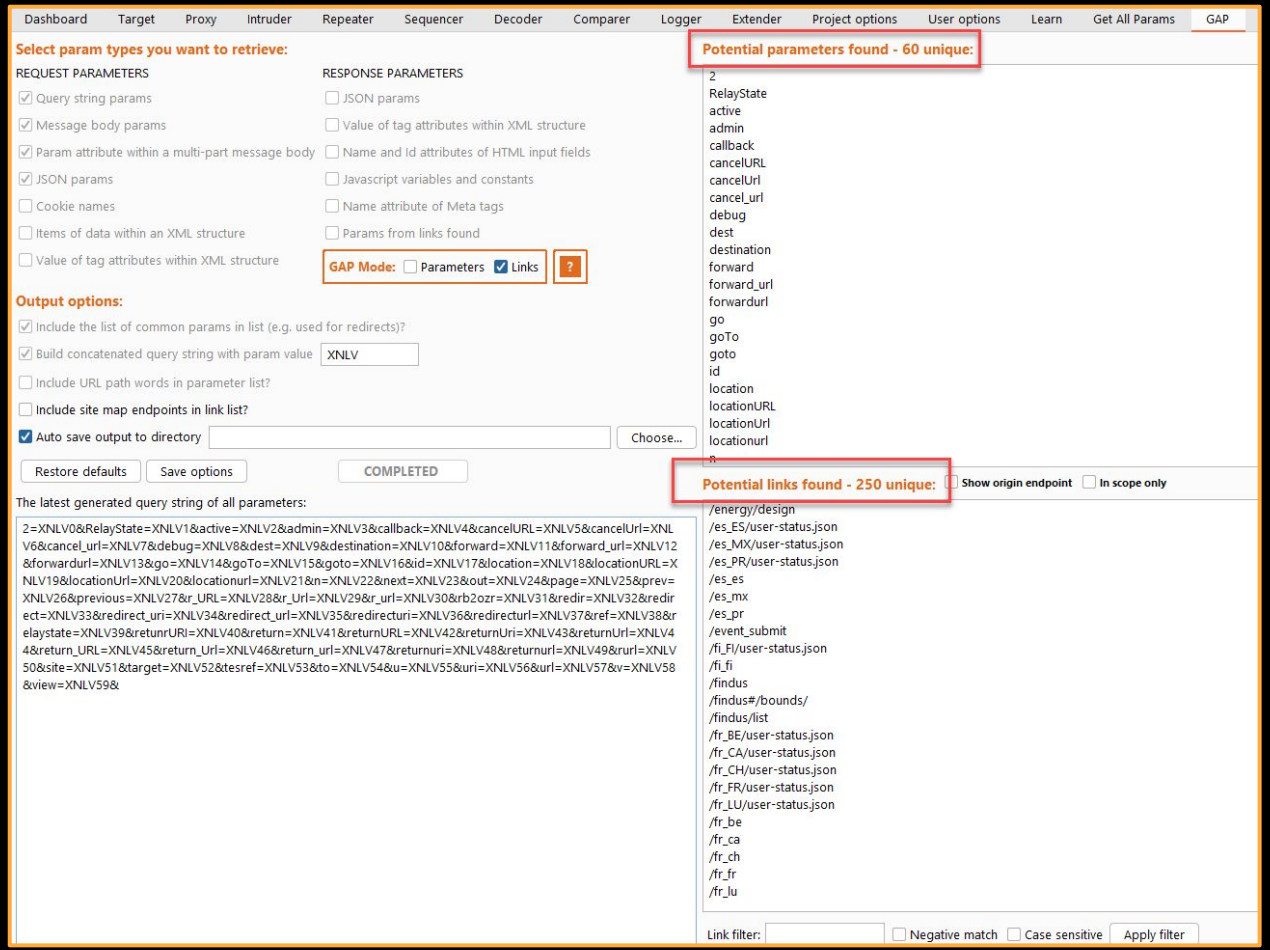
A new tool I like for this is xnLinkFinder by @Xnl-h4ck3r

```
root@arcanum:~/xnLinkFinder# python3 xnLinkFinder.py -i tesla.com -v -d 2 -sp https://www.tesla.com

                                                o          o    o--o          o
                                                |    o     | /  |    o        |
                                      \ / o-o   |      o-o  OO   0-o    o-o   o-0 o-o o-o
                                       o  |  | |    | |  | | \  |    | |  | |  | |-' |
                                      / \ o  o 0---o| o  o o  o o    | o  o  o-o o-o o
                                                    |                |
                                                    ' by @Xnl-h4ck3r '             v0.2


User-Agent Group: desktop
Processing URL:
Response 200: tesla.com

Proccessing URL's, depth 2:
Response 200: https://www.tesla.com/inventory/new/m3
Response 200: https://www.tesla.com/inventory/new/ms
Response 200: https://www.tesla.com/
Response 200: https://www.tesla.com/inventory/new/my
Response 200: https://www.tesla.com/energy/design
Response 200: https://www.tesla.com/modelx/design
Response 200: https://www.tesla.com/en_sg
Response 200: https://www.tesla.com/en_gb
Response 200: https://www.tesla.com/en_ca
Response 200: https://www.tesla.com/en_au
Response 200: https://www.tesla.com/ko_kr
Response 200: https://www.tesla.com/_flysystem/s3/js/js_D4R1163OMFi6ZBU-DfUImCQ0Dc-qCbe6bcNczb5zEts.js
```

JavaScript Parsing (in Burp)

In addition @Xnl-h4ck3r created a Burp extension called GAP which finds BOTH parameters and paths.

https://github.com/xnl-h4ck3r/burp-extensions



JavaScript Parsing (tip)

Minified or obfuscated Javascript still needs to be assessed manually. In some cases https://beautifier.io can do some work, but only in small number of cases.

Stay tuned to @matsuuu_ in this space...

Matsu
@matsuuu_

Replying to @matsuuu_ @joohoi and 3 others

A small sneak peek. It ain't much yet but it's honest work

Got traversing through the AST working today so that I can statically analyze what variables are being used in the function calls

As it's walking the AST, it's a recursion hell waiting for implementation so bear with me

Matsu @matsuuu_ · Apr 2

I'm utilizing the TypeScript Language Services as it comes with a lot of tools to manage the AST

1 3

Matsu
@matsuuu_

Replying to @matsuuu_ @dee_see and 4 others

Oh and something that's not visible from the pic is that it already crawls sites recursively, resolving imports from es modules.

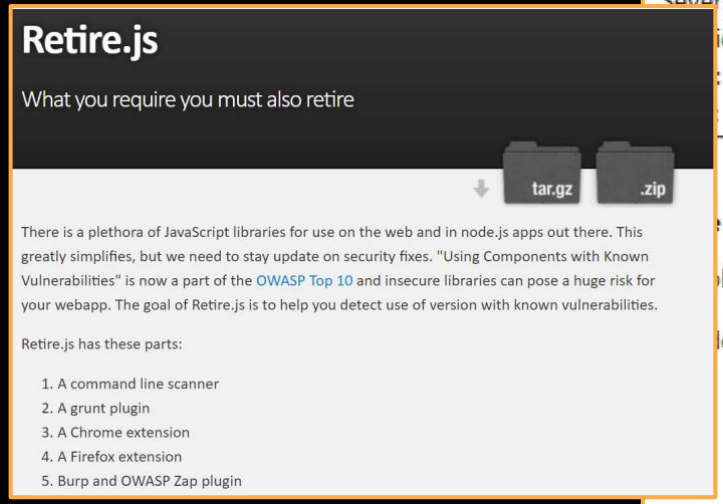
So if app.js imports foo.js and it imports bar.js, they all get crawled already.

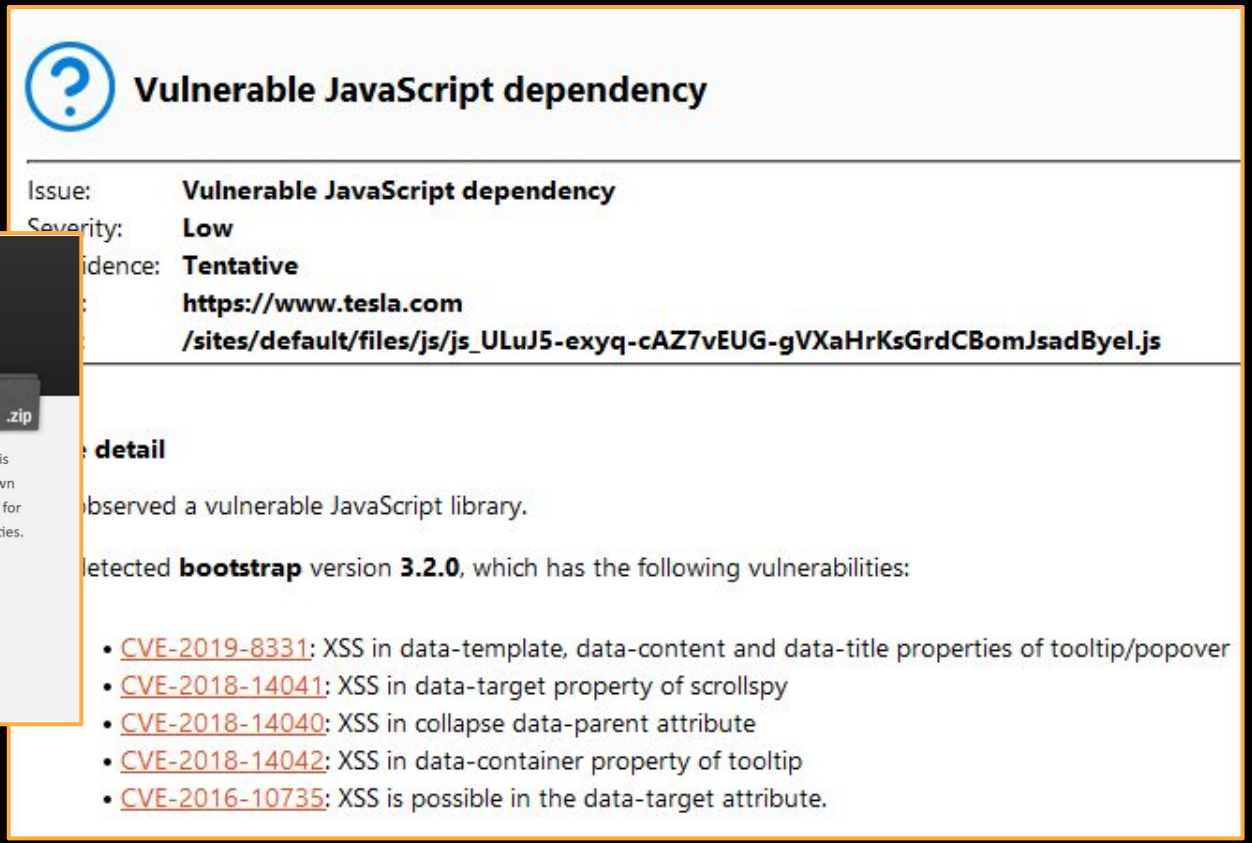
The hardest part is next and that is to make it all flow infinitely

```
matsu 【=◆◡◆=】 Thunderstorm:~/Projects/lurker/lib$ bat test/test-mini.js
       File: test/test-mini.js
       Size: 258 B
   1   function multipleEscaped(id, name) {
   2 ~     fetch(`http://localhost:3001/${id}/foo/${name}/bar`);
   3   }
   4
   5   const userId = 123;
   6
   7   multipleEscaped(userId, "Foobar");
   8
   9   multipleEscaped(420, "Blaze It");
  10
  11
  12   const randomString = "baz";
  13   multipleEscaped(userId, randomString);
matsu 【=◆◡◆=】 Thunderstorm:~/Projects/lurker/lib$ npm start

> lurker@1.0.0 start
> tsc && node ./dist/index.js

Fetching and sourceFiling  http://localhost:8000/test/test-mini.js
Fetch function found {
  targetUrl: '`http://localhost:3001/${id}/foo/${name}/bar`',
  variables: { id: [ '123', '420' ], name: [ '"Foobar"', '"Blaze It"', '"baz"' ] },
  options: {},
  dynamic: true,
  sourceFileName: 'http://localhost:8000/test/test-mini.js',
  fetchText: 'fetch(`http://localhost:3001/${id}/foo/${name}/bar`)'
}
{
  fuzzRoute: '`http://localhost:3001/FUZZ/foo/FUZZ/bar`',
  argsUsedBySystem: { id: [ '123', '420' ], name: [ '"Foobar"', '"Blaze It"', '"baz"' ] },
  variablesToFuzzWith: [ '123', '420', '"Foobar"', '"Blaze It"', '"baz"' ]
}
matsu 【=◆◡◆=】 Thunderstorm:~/Projects/lurker/lib$
```

JavaScript (Libs / Dependencies)

The pioneer here was Retire.js but now Burp has a lot of this data built in.





Heat Mapping

Heat Mapping

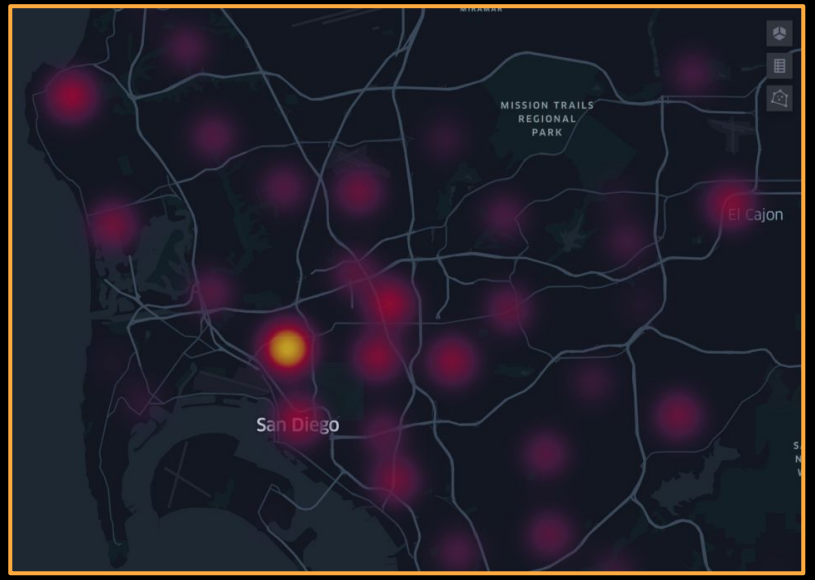
Heat mapping is my own term for:

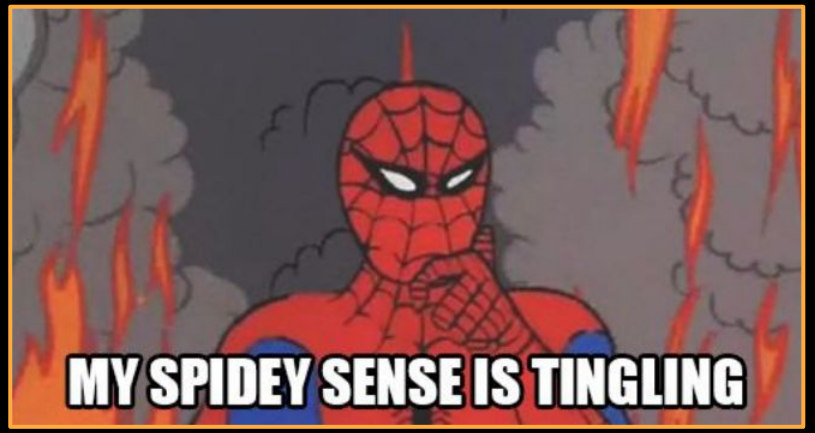
- “Places” inside the application where bad things can normally happen

Or

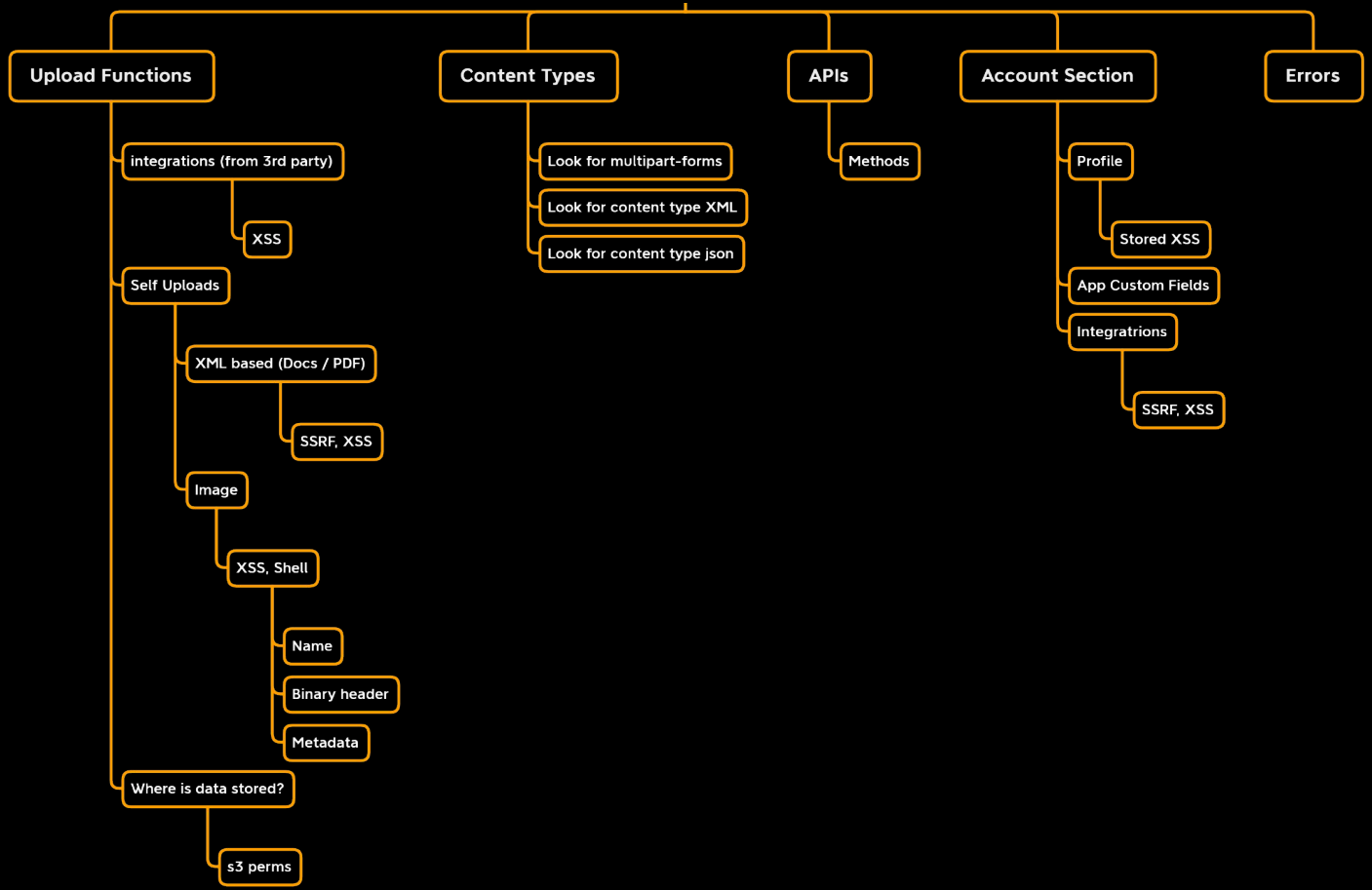
- Things I want to look at, that may indicate interesting places to explore from a hacker PoV

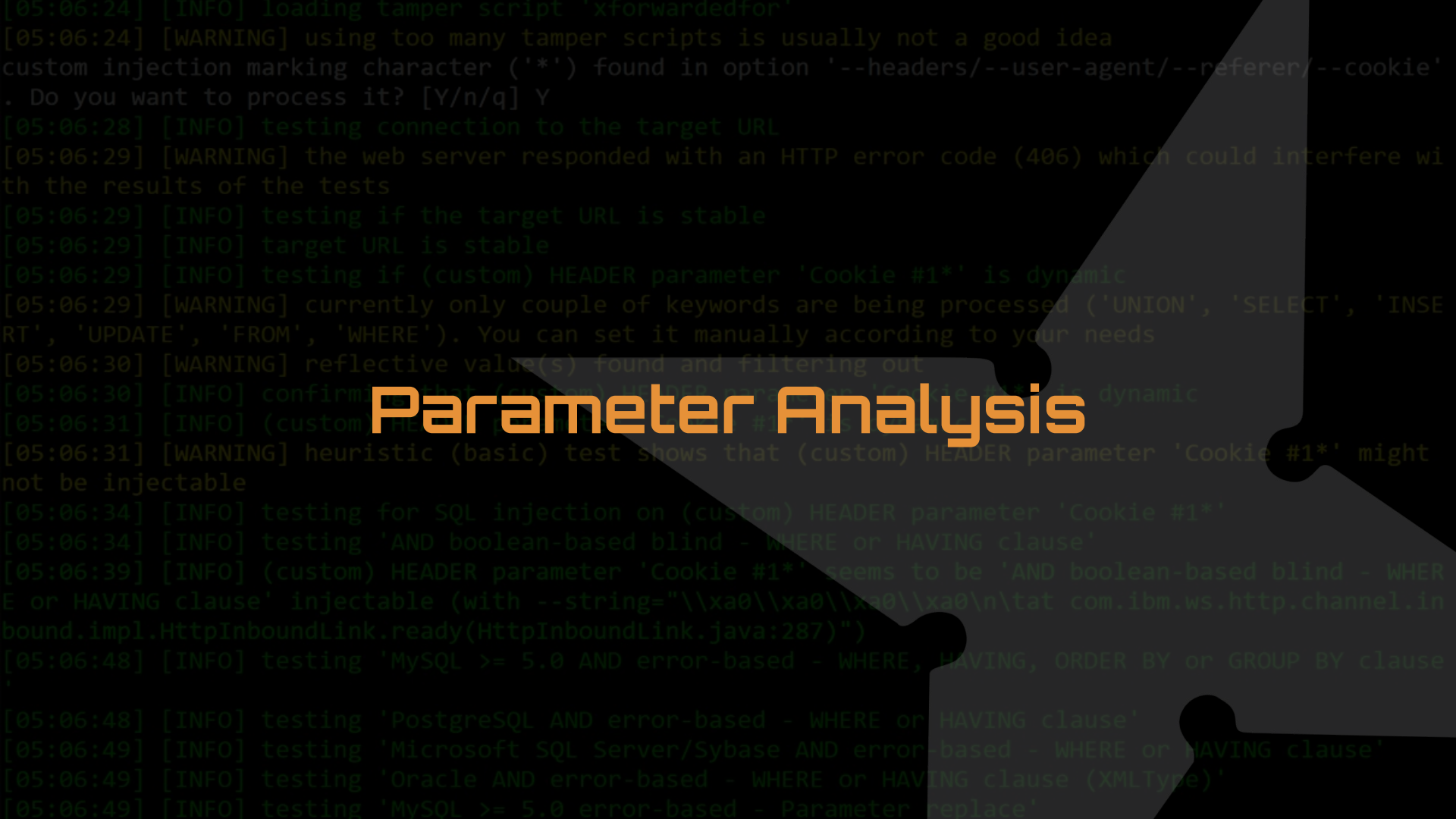
Call it hacker intuition… or whatever.





Heat Mapping Mindmap (WIP)

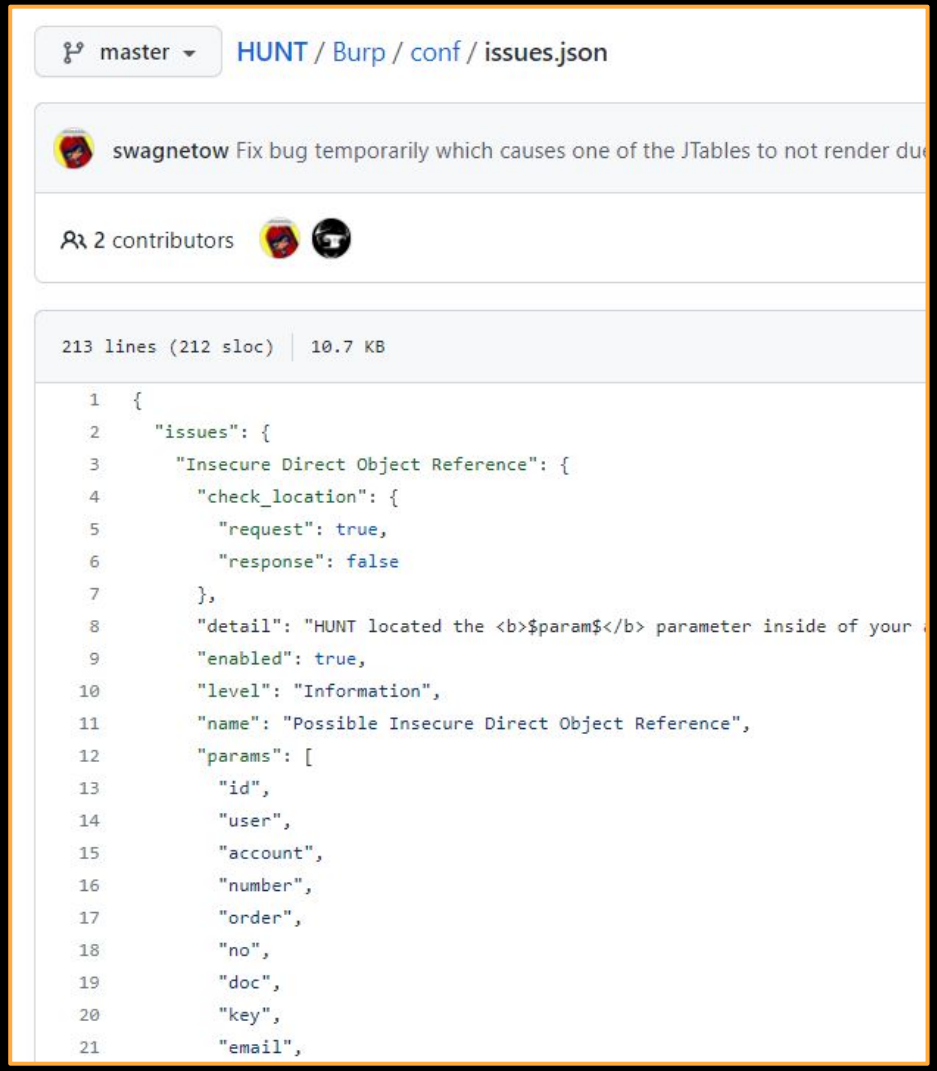



Parameter Analysis

Parameter Analysis

Several years ago I did a project to analyze the top parameters that were vulnerable (statistically) to certain vulnerability classes.

To this day, this remains one of my testing **super-powers**. It gives you semblance of priority in sea of inputs when testing against large applications.



Parameter Analysis

Since HUNT's inception there have been other projects to attempt this with smaller datasets.

- XSSed.com
- Hackerone.com Disclosures

Additionally many of these lists have been ported to pattern files for TomNomNom's GF tool (to grep for).

Simply run GF with a pattern file like the ones from 1nianl33t's Gf-Patterns...

cat urls.txt | gf sqli

Parameter Analysis

TBH, i'm planning on redoing the whole thing soon with revamped lists, sources, and patterns for GF.

Here are the current lists/patterns that exist (many of them are likely duplicates of each other):

https://github.com/bugcrowd/HUNT/blob/master/Burp/conf/issues.json
https://github.com/lutfumertceylan/top25-parameter/tree/master/gf-patterns
https://github.com/1ndianl33t/Gf-Patterns
https://github.com/emadshanab/Gf-Patterns-Collection
https://github.com/mrofisr/gf-patterns
https://github.com/robre/gf-patterns
https://pentesterlab.com/my/vouchers/vIMySX0Kfd5t7-3u0ZPNoOd4G-peSlGp/voucher

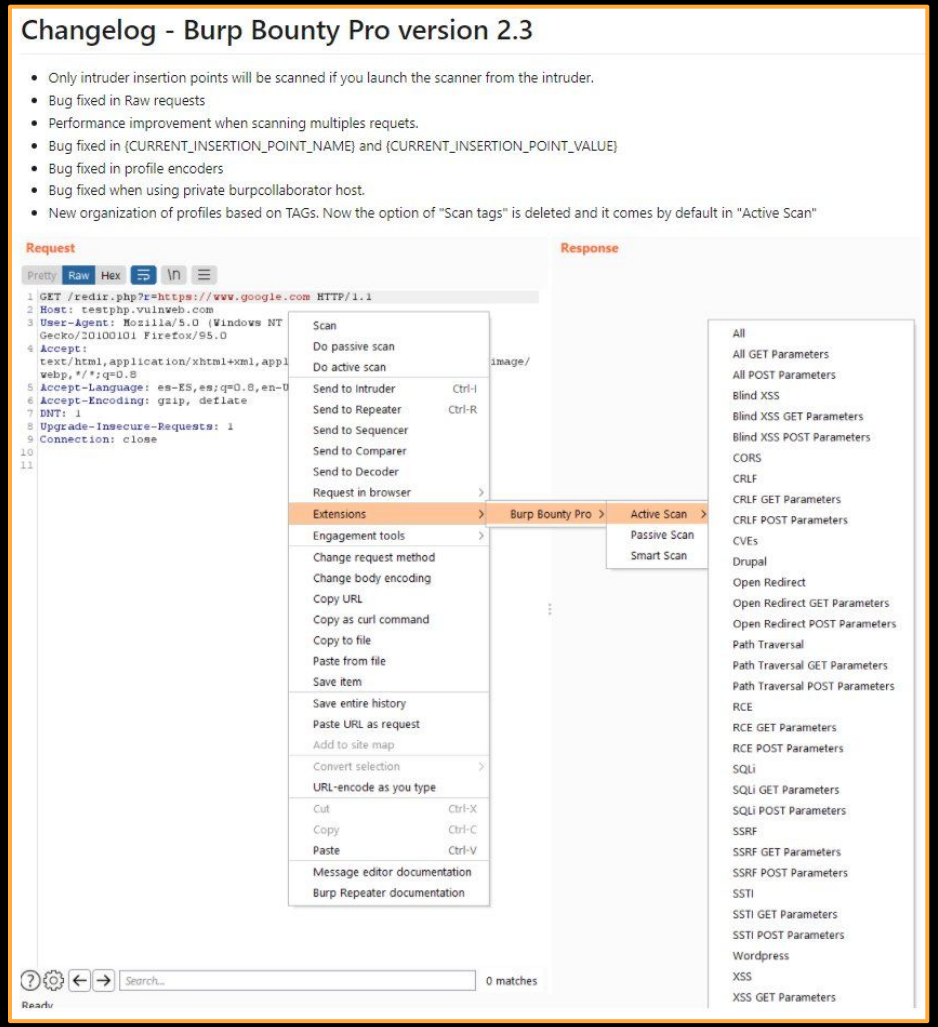
Keep an eye out for:

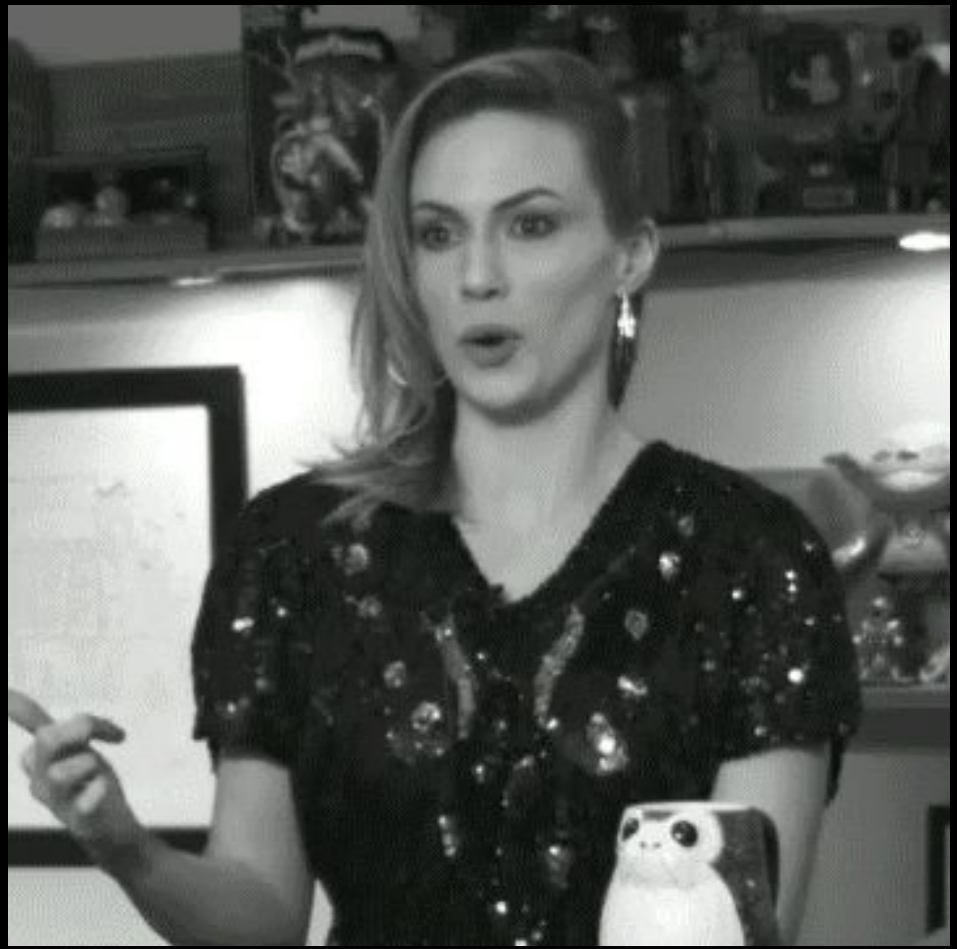
jhaddix/sus_params on GitHub soon.

Parameter Analysis

GF and GF patterns work on the command line, but if you want this in your browser (since HUNT is unmaintained atm) you can look into the **paid** tool “BurpBounty” which has embedded most of the previous sources.

BurpBounty makes it easy to extend Burp with writing your own scan checks easily, and have a library of their own in Pro.





What's Next?

- V1.1
- TBHM App Analysis Workshop
- TBHM Recon v5