

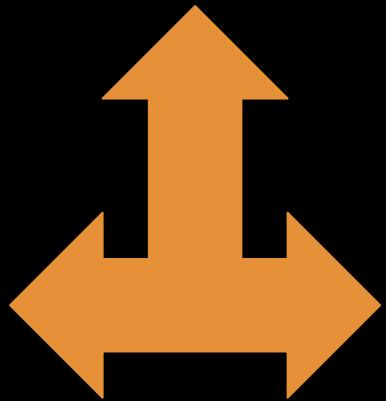
```
[05:06:24] [INFO] loading tamper script 'xforwardedfor'  
[05:06:24] [WARNING] using too many tamper scripts is usually not a good idea  
custom injection marking character ('*') found in option '--headers/--user-agent/--referer/--cookie'  
. Do you want to process it? [Y/n/q] Y  
[05:06:28] [INFO] 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] (custom) HEADER parameter 'Cookie #1*' is dynamic  
[05:06:31] [INFO] heuristic (basic) test s (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\x0\x0\x0\x0\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'
```

The Bug Hunter's Methodology v4.01

Recon

TBHM v4

Recon



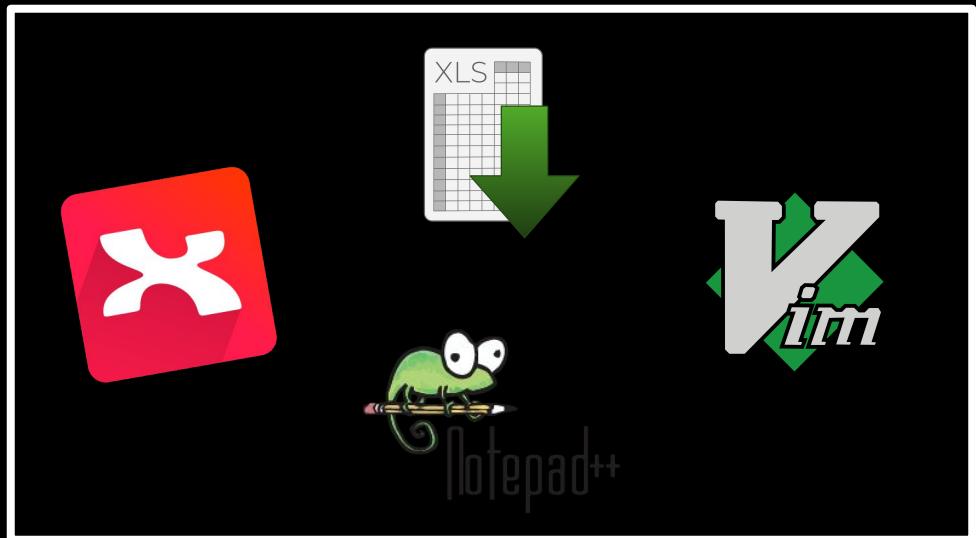
Application
Analysis

About Me

- Father, husband, hacker, gamer, sometimes streamer.
 - 28th on BC leaderboard
 - [Twitter](#) | [YouTube](#) | [Twitch](#)
-
- Currently Playing:



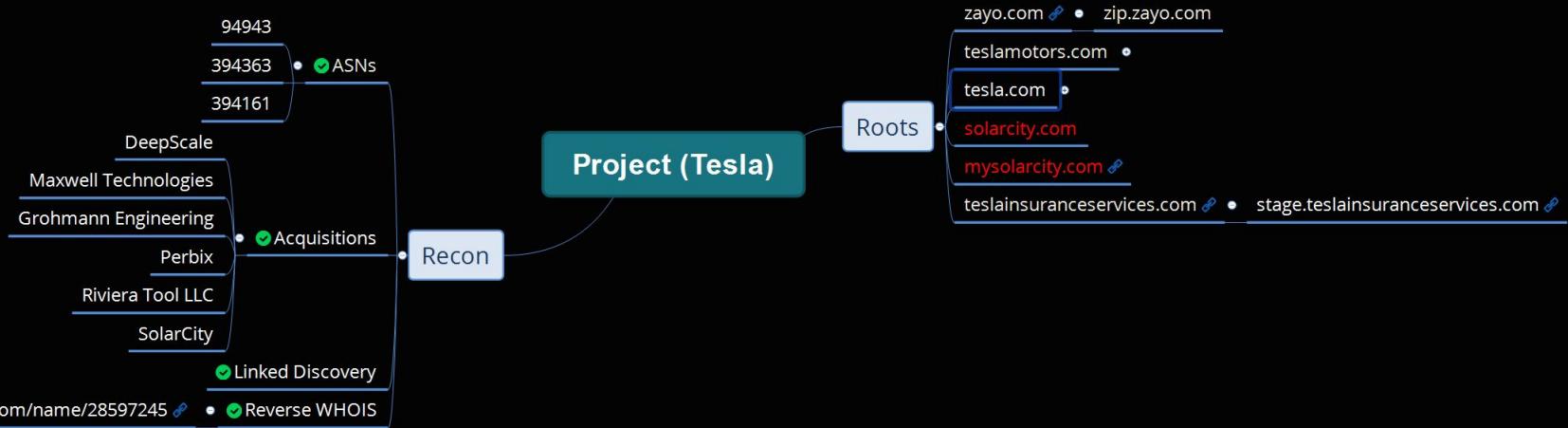
Project Tracking

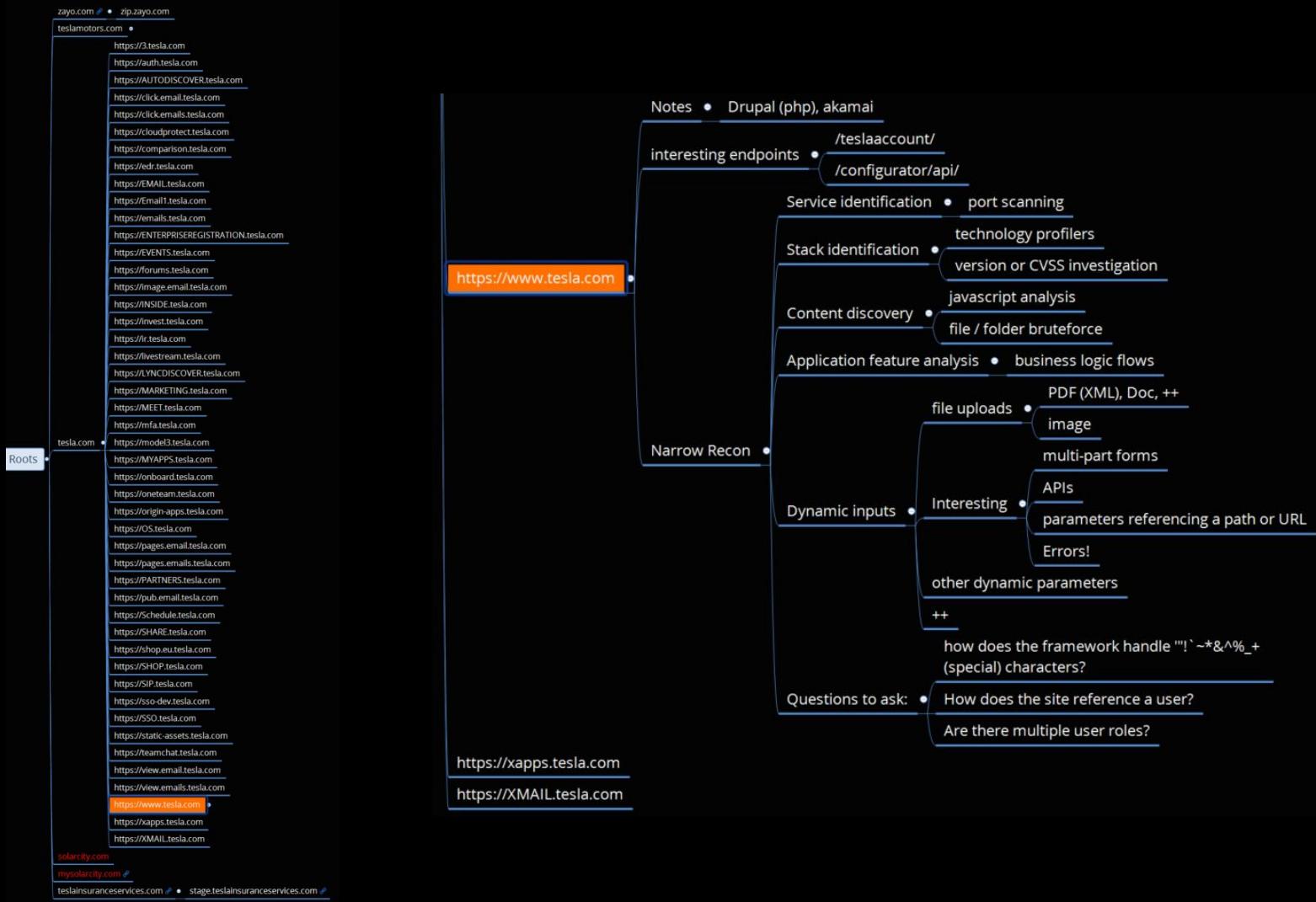


In many parts of the workshop we will need to keep track of site-hierarchy, tools output, interesting notes, etc.

I use mindmaps with **Xmind** but the same effect can be achieved through a lot of different programs.

Mindmaps allow me to visualize large scope bug hunting targets and also allow me to break up methodology for in-depth bug hunting as well.





Mission

Wide Recon is the art of discovering as many assets related to a target as possible.
Make sure your scope permits testing these sites.

Scope
Domains

Acquisitions

ASN
Enumeration

Other

Reverse
WHOIS

Subdomain
Enumeration

Port
Analysis

Vulns:
Subdomain Takeover
Buckets
Github leaks

Automation/Helper:
Interlace
Screenshoting
Frameworks

```
[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] (custom) HEADER parameter 'Cookie #1*' is dynamic
[05:06:31] [INFO] (custom) HEADER parameter 'Cookie #1*' is dynamic
[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\x0\x0\x0\x0\x0\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'
```

Finding Seeds/Roots

Scope Domains (Bugcrowd)

bugcrowd.com/tesla

Vehicle software updates or assistance, coordinate with goodwill Tesla when more information is required or if you have questions or concerns about your vehicle's software or service on the Researcher registration page.

vehicle by over-the-air update, offering assistance at a service center to restore the vehicle's software using our standard service tools, or other actions we deem appropriate. Tesla has complete discretion as to the software or other assistance that will be provided and it may be only for a limited number of times. Tesla's support does not extend to any out-of-pocket expenses (e.g. towing) incurred by you. Tesla reserves the right to limit the number of service requests per pre-approved, good-faith researcher and unregister a research-registered vehicle at any time.

- Tesla considers that a pre-approved, good-faith security researcher who complies with this policy to access a computer on a research-registered vehicle has not accessed a computer without authorization or exceeded authorized access under the Computer Fraud and Abuse Act ("CFAA").
- Tesla will not bring a copyright infringement claim under the Digital Millennium Copyright Act ("DMCA") against a pre-approved, good-faith security researcher who circumvents security mechanism, so long as the researcher does not access any other code or binaries.
- Tesla will not consider software changes, as a result of good-faith security research performed by a good-faith security researcher, to a security-registered vehicle to void the vehicle warranty of the security-registered vehicle, notwithstanding that any damage to the car resulting from any software modifications will not be covered by Tesla under the vehicle warranty.

Targets

In scope

Target name	Type
A hardware product that you own or are authorized to test against (Vehicle/PowerWall/etc.)	Hardware
*.tesla.com	Website
*.tesla.cn	Website
*.teslamotors.com	Website
*.tesla.services	Website
Any host verified to be owned by Tesla Motors Inc. (domains/IP space/etc.)	Website
Official Tesla Android apps	Android
Official Tesla iOS apps	iOS

Scope Domains (HackerOne)

SIGN IN | SIGN UP

hackerone

FOR BUSINESS FOR HACKERS HACKTIVITY COMPANY TRY HACKERONE



Verizon Media
Build Brands Members Love

<https://www.verizonmedia.com> · @verizonmedia

Reports resolved: 5997 Assets in scope: 58 Average bounty: \$394-\$500

Submit report

Bug Bounty Program
Launched on Feb 2014
Managed by HackerOne
Bounty splitting enabled ⓘ

Policy Hacktivity Thanks Updates (24)

Policy

Welcome to Verizon Media

With brands like Yahoo, HuffPost and TechCrunch, Verizon Media helps people stay informed and entertained, communicate and transact, while creating new ways for advertisers and partners to connect. From XR experiences to advertising and content technology, Verizon Media is an incubator of innovation and is revolutionizing the next generation of content creation in a 5G world.



Paranoids

Response Efficiency

8 hrs
Average time to first response

2 days
Average time to triage

2 months
Average time to bounty

95% of reports

If you've found a vulnerability that affects an asset belonging to Verizon Media, but is not included as in scope on any of the Verizon Media programs, please report it to this program.

Acquisitions (Crunchbase)

The screenshot shows the Crunchbase search interface. A red arrow points from the search bar to the results page, which displays two sections: 'ORGANIZATIONS' and 'PEOPLE'. The 'ORGANIZATIONS' section lists 'Twitch', 'Twitchy', and 'Twitchtime'. The 'PEOPLE' section lists 'Robert W. Twitchell Jr.', 'Chu Boi', and 'Brooke Van Dusen'. Both sections have a 'SHOW MORE' button at the bottom.

https://www.crunchbase.com

← Back

Crunchbase Pro

SEARCH

Companies

People

Investors

Funding Rounds

Acquisitions

Schools

Events

Hubs

My Searches

My Lists

Marketplace

Add New Profile

Results

twitch

ORGANIZATIONS

- Twitch — Twitch is social video platform for gamers where more than 100 million gather every month to broadcast...
- Twitchy — Twitchy is a ground-breaking social media curation site powered by a kinetic staff of social media ju...
- Twitchtime — Web & App Development | Digital Marketing Company

SHOW MORE ▾

PEOPLE

- Robert W. Twitchell Jr — Bob has an extensive background in the wireless industry. He holds over 100 granted ...
- Chu Boi — ChuBoi a Twitch streamer that plays FIFA and a bunch of other games.
- Brooke Van Dusen — Brooke is Twitch's Director of Game Developer Success, where he works with game studio...

SHOW MORE ▾

EVENTS

We want to continue to gather seed/root domains. Acquisitions are often a new way to expand our available assets if they are in scope. We can investigate a company's acquisition on sites like <https://crunchbase.com>, wikipedia, and Google.

Acquisitions (Crunchbase)

Overview

Acquired by  Amazon Number of Acquisitions 4

Twitch
Twitch is social video platform for gamers where more than 100 million gather every month to broadcast, watch and talk about video games.
San Francisco, California, United States

Categories
Headquarters Regions
Sub-Organization of
Founded Date
Founders
Operating Status
Funding Status
Last Funding Type
Number of Employees
Also Known As
Legal Name

Social Media, Video, Video Games, Video Streaming
San Francisco Bay Area, West Coast, Western US
 Amazon
2007
Emmett Shear, Justin Kan, Kyle Vogt
Active
M&A
Series C
251-500
twitch.tv, Twitch Interactive
Twitch Interactive, Inc.

IPO Status
Company Type

Private
For Profit

Website www.twitch.tv/
Facebook [View on Facebook](#)
LinkedIn [View on LinkedIn](#)
Twitter [View on Twitter](#)

Twitch is social video for gamers. It is a video platform and community for gamers where more than 100 million gather every month to broadcast, watch and talk about video games. Twitch's video platform is the backbone of both live and on-demand distribution for the entire video game ecosystem. This includes game developers, publishers, media...

[Read More](#)

Acquisitions

Number of Acquisitions 4

Twitch has acquired 4 organizations. Their most recent acquisition was Revlo on Dec 13, 2018.

Which types of acquisition does this organization make most frequently? 

Acquired Organization Name	Announced Date	Price	Transaction Name
 Revlo	Dec 13, 2018	—	 Revlo acquired by Twitch
 ClipMine	Aug 17, 2017	—	 ClipMine acquired by Twitch
 Curse	Aug 16, 2016	—	 Curse acquired by Twitch
 GoodGame	Dec 9, 2014	—	 GoodGame acquired by Twitch

Here we can possibly drill down into old domains related to Revlo, ClipMine, Curse, and GoodGame.

Remember to do some Googling on these acquisitions to see if they are still owned by the parent company. Many times acquisitions will split back out or get sold to another company.

ASN Enumeration (bgp.he.net)

Autonomous System Numbers are given to large enough networks. These ASN's will help us track down some semblance of an entity's IT infrastructure. The most reliable way to get these is manually through Hurricane Electric's free-form search:

- <http://bgp.he.net>

Because of the advent of cloud infrastructure, ASNs aren't always a complete picture of a network. Rogue assets could exist on cloud environments like AWS and Azure. Here we can see several IP ranges.

The screenshot shows the Hurricane Electric BGP Toolkit interface. A red box highlights the search bar containing the query "twitch". Red arrows point from this bar to the search results table. The table has two columns: "Result" and "Description". The "Result" column lists various IP ranges and ASN numbers, while the "Description" column lists the organization name "Twitch Interactive Inc." followed by small American flag icons. The results include:

Result	Description
twitch	Twitch Interactive Inc.
AS46489	Twitch Interactive Inc.
AS397153	Twitch Interactive Inc.
99.181.96.0/19	Twitch Interactive Inc.
99.181.80.0/21	Twitch Interactive Inc.
99.181.64.0/20	Twitch Interactive Inc.
52.223.240.0/20	Twitch Interactive Inc.
52.223.224.0/20	Twitch Interactive Inc.
52.223.208.0/21	Twitch Interactive Inc.
52.223.192.0/20	Twitch Interactive Inc.
45.113.128.0/22	TWITCH INTERACTIVE, INC.
2a01:62e0:f001:b3::/64	Twitch Interactive, Inc.
2a01:62e0:f001:b2::/64	Twitch Interactive, Inc.

ASN Enumeration (cmd line)

```
root@TBox4:~/tools/Asnlookup# python asnlookup.py -o tesla

ASNLLOOKUP
Author: Yassine Aboukir (@yassineaboukir)

[*] User's license key is valid!
[*] IP addresses owned by tesla are the following (IPv4 or IPv6):
[*] IPv4 addresses saved to:
/root/tools/Asnlookup/output/tesla_ipv4.txt

192.95.64.0/24
199.120.48.0/24
199.120.49.0/24
199.66.10.0/24
199.66.11.0/24
199.66.9.0/24
205.234.11.0/24
209.133.79.0/24
213.19.141.0/24
8.21.14.0/24
8.45.124.0/24

https://twitter.com/yassineaboukir
```

```
root@j3ssie ► /tmp/demo # echo 'tesla' | metabigor net --org -v
[0000] INFO Metabigor beta v1.1 by @j3ssiejjj
[0000] INFO Store log file to: /tmp/metabigor.log
[0000] INFO [*] Starting get IP Info for Organization: tesla
[0000] INFO Get data from: http://asnlookup.com/api/lookup?org=tesla
[0000] INFO Get data from: https://bgp.he.net/search?search%5Bsearch%5D=tesla&commit=Search
[0001] INFO 192.95.64.0/24
[0001] INFO 199.120.48.0/24
[0001] INFO 199.120.49.0/24
[0001] INFO 199.66.10.0/24
[0001] INFO 199.66.11.0/24
[0001] INFO 199.66.9.0/24
[0001] INFO 205.234.11.0/24
[0001] INFO 209.133.79.0/24
[0001] INFO 213.19.141.0/24
[0001] INFO 8.21.14.0/24
[0001] INFO 8.45.124.0/24

https://twitter.com/j3ssiejjj
```

Some automation is available to get ASNs. One such tool is the ‘net’ switch of “[Metabigor](#)” by j3ssiejjj which will fetch ASN data from a keyword from bgp.he.net and asnlookup.com

Another is “[ASNLLookup](#)” by Yassine Aboukir which utilizes the maxmind.com dataset.

One problem with cmd line enumeration is that you could return records from another org on accident that contains the keyword ‘tesla’.

ASN Enumeration (with Amass)

For discovering more seed domains we want to scan the whole ASN with a port scanner and return any root domains we see in SSL certificates, etc.

We can do this with Amass intel

[Amass](#) is written by Jeff Foley and the Amass team.

Result	
twitch	
AS46489	Twitch Interactive Inc.

```
amass 3.7.1 from Jeff Foley (caffix) installed
root@TBox4:~/tools/Asnlookup# amass intel -asn 46489
justin.tv
ttvnw.net
twitch.tv
twitchcon.com
socialcam.com
```

Reverse WHOIS (with Whoxy.com)

Every website has some registration info on file with the registrars. Two key pieces of data we can use are Organization name and any emails in the WHOIS data. To do this you need access to a large WHOIS database. WHOXY.com is one such database.

You can use whoxy.com in this fashion, after you register and your free API key:

- <http://api.whoxy.com/?key=APIkeyHERE&reverse=whois&name=Twitch+Hostmaster>

Careful with reverse whois data as it is the least high fidelity source of new root/seed domains. It might include many parked domains or redirects to out of scope assets.

Who owned [twitch.tv](#) in the past? (4 records)

19 NOV 2012

Owner: WhoisGuard WhoisGuard Protected () ([14,045 domains](#))
Geolocation: Los Angeles, CA, United States ([120 million domains](#) from **United States** for **\$3,500**)
Nameservers: [asia1.akam.net](#), [asia9.akam.net](#), [eur2.akam.net](#), [ns1-167.akam.net](#)
Status: CLIENTXFERPROHIBITED

26 MAR 2015

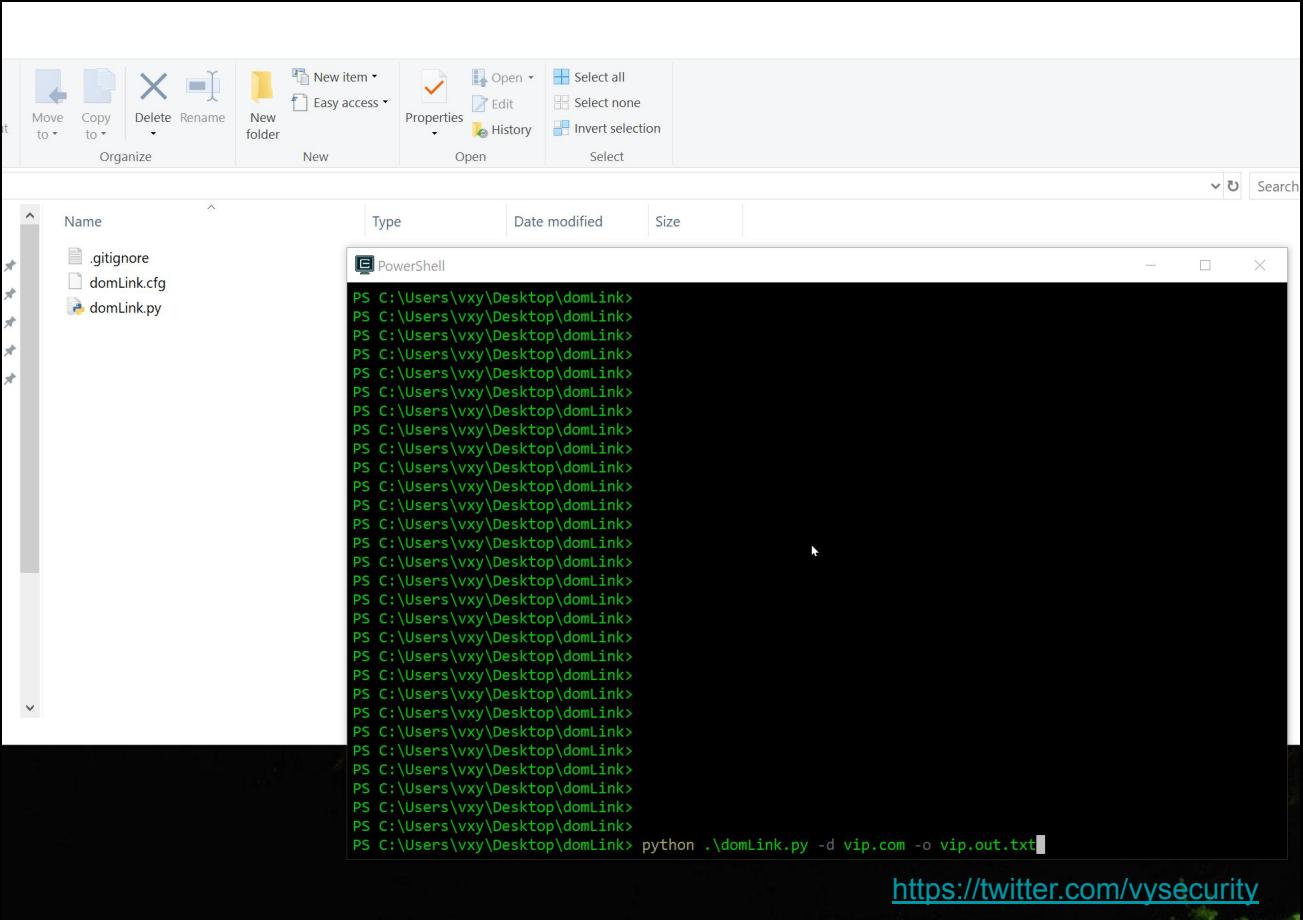
Owner: DOMAIN MASTER ([60,341 domains](#)) **UPDATED**
Company: JUSTIN.TV ([20 domains](#))
Geolocation: SAN FRANCISCO, CA, United States ([120 million domains](#) from **United States** for **\$3,500**)
Email: domainmaster@justin.tv ([20 domains](#))
Nameservers: [a1.verisigndns.com](#), [a2.verisigndns.com](#), [a3.verisigndns.com](#), [ns1.p18.dynect.net](#)
Status: clientTransferProhibited

22 JUN 2015

Owner: Twitch Hostmaster ([574 domains](#)) **UPDATED**
Company: Twitch Interactive, Inc. ([575 domains](#))
Geolocation: San Francisco, CA, United States ([120 million domains](#) from **United States** for **\$3,500**)
Email: hostmaster@amazon.com ([96,849 domains](#))
Nameservers: [a1.verisigndns.com](#), [a2.verisigndns.com](#), [a3.verisigndns.com](#), [ns1.p18.dynect.net](#)
Status: clientDeleteProhibited, clientTransferProhibited, clientUpdateProhibited

Reverse WHOIS (with DOMLink)

[DOMLink](#) is a tool written by Vincent Yiu (@vysecurity) which will recursively query the WHOXY WHOIS API. It will start by querying our targets WHOIS record, then analyze all the data and look for other records which contain the organization name or are registered to emails in the record. It does this recursively until it finds no more records of match.



The screenshot shows a Windows File Explorer window with the following details:

- Toolbar:** Move to, Copy to, Delete, Rename, New folder, New item, Easy access, Open, Select all, Open History, Select, Invert selection.
- File List:** .gitignore, domLink.cfg, domLink.py
- Terminal Window:** PowerShell session showing repeated PS C:\Users\vxv\Desktop\domLink> commands, followed by the command PS C:\Users\vxv\Desktop\domLink> python .\domLink.py -d vip.com -o vip.out.txt.

Ad/Analytics Relationships (builtwith.com)

You can also glean related domains and subdomains by looking at a target's ad/analytics tracker codes. Many sites use the same codes across all their domains. Google analytics and New Relic codes are the most common. We can look at these "relationships" via a site called BuiltWith. Builtwith also has a Chrome and Firefox extension to do this on the fly.

- https://builtwith.com/relationships/twitch_tv

BuiltWith is also a tool we'll use to profile the technology stack of a target in later slides.

The screenshot shows the BuiltWith Pro Dashboard for the Twitch.TV Technology Profile. The top navigation bar includes links for Reports, Tools, Plans & Pricing, Customers, Account, Help, and a Logout button. A red arrow points to the 'Logout' button. Below the navigation is a breadcrumb trail: Home / twitch.tv Technology Profile / twitch.tv Historical Website Relationship Profile. The main content area features a large 'TWITCH.TV' logo. Below it are tabs for Technology Profile, Detailed Technology Profile, Meta Data Profile, Relationship Profile (which is selected), and Redirect Profile. The Relationship Profile tab displays a table titled 'TWITCH.TV Tag History' with columns for Type, ID, First Detected, and Last Detected. Two specific rows are highlighted with red arrows: 'UA-XXXXX' (First Detected: Jan 2017, Last Detected: Jul 2018) and 'UA-23719667' (First Detected: Nov 2011, Last Detected: Mar 2018). To the right of the table is a sidebar titled 'TWITCH.TV Connected Websites' listing various websites like 4egaming.com, ahikocake.com, alacon01.com, alt-fx.com, astrogaming.co.uk, avalonstart.tv, b0eh.com, bafe.com, bliiny.tv, bit.ly, boothebun.net, brettdoesgaming.com, and bytem33.com. A red arrow points to the 'Relationship' tab in the top right corner of the sidebar. At the bottom of the page is a section titled 'Tag History & Relationships' showing a network diagram where various domains are connected through tracking codes. A red arrow points to the bottom left of this diagram.

Ad/Analytics Relationships (getrelationship.py)

Want to do it on the command line?

[@M4ll0k](#) has you covered!

<https://raw.githubusercontent.com/m4ll0k/Bug-Bounty-Toolz/master/getrelationship.py>

```
m4ll0k@m4ll0k Desktop % echo "uber.com" | python3 getrelationship.py
m.21north.in
999yese.com
amerencsr.com
a-pple.es
avs.auto
chewy.com
chromadex.com
daftaruber.com
deck.gl
driveuber.co.nz
durangodelivery.com
edrsilver.com
fastenal.com
internetubeatbr.com
ishutter.in
jackcooperlogistics.com
jacksonvillewatertaxi.com
jump.com
kitchensofubereats.com
laureate.net
lingnei.cn
lingnei.com
```

Google-Fu

You can Google the:

- Copyright text
- Terms of service text
- Privacy policy text

from a main target to glean related hosts on Google.

The screenshot shows a Google search results page. At the top, the Google logo is followed by the search query "© 2019 Twitch Interactive, Inc." inurl:twitch. Below the search bar, there are tabs for All, News, Images, Shopping, Maps, More, Settings, and Tools. The "All" tab is selected. The search results indicate "About 5,480 results (0.49 seconds)".

The first result is a link to the App Store for "Twitch: Live Game Streaming on the App Store - iTunes - Apple". The link is <https://itunes.apple.com/us/app/twitch-live-game-streaming/id460177396?mt=8>. It includes a star rating of 4.8 from 452,159 reviews, a "Free" price, and categories like iOS, Entertainment, and more. A note mentions "Tobacco, or Drug Use or References. Infrequent/Mild Horror/Fear Themes. Infrequent/Mild Profanity or Crude Humor. Copyright: © 2019 Twitch Interactive, Inc."

The second result is a link to "Build Twitch Extensions | Twitch Developers" at <https://dev.twitch.tv/build/>. It describes how Twitch Extensions enable users to create live apps that interact with the stream, such as mini-chats or interactive experiences.

A "People also ask" section is visible on the right, listing the following questions:

- Can you embed twitch streams?
- How do I watch a livestream on twitch?
- What is a twitch extension?
- How do I enable extensions on twitch?

Below these, there are two more links:

- "Embedding Twitch | Twitch Developers" at <https://dev.twitch.tv/docs/embed/>. It discusses embedding Twitch on a website.
- "Twitch Extensions | Twitch Developers" at <https://dev.twitch.tv/extensions/>. It describes creating live apps that interact with the stream.

Shodan

Shodan is a tool that continuously spiders infrastructure on the internet. It is much more verbose than regular spiders. It captures response data, cert data, stack profiling data, and more. It requires registration.

Example:

<https://www.shodan.io/search?query=twitch.tv>

We can glean a valuable question: is twitch.amazon.eu relevant to our testing?

The screenshot shows the Shodan search interface with the query "twitch.tv" entered in the search bar. The results page displays the following information:

- TOTAL RESULTS:** 182
- TOP COUNTRIES:** United States (82), Germany (32), France (13), United Kingdom (11), Canada (10)
- TOP SERVICES:** HTTPS (76), HTTP (28), 27/015 (27), Minecraft (24), 27/016 (14)
- TOP ORGANIZATIONS:** Amazon.com (29), OVH SAS (8), marris GmbH (7), OVH Hosting (6), Fasty (6)
- TOP OPERATING SYSTEMS:** Linux 3.x (1)
- TOP PRODUCTS:** nginx (49), Minecraft (24), Apache httpd (23), ARK: Survival Evolved (10), Counter-Strike: Global Offensive (6)

Below the search results, several specific host entries are shown with red arrows pointing to them:

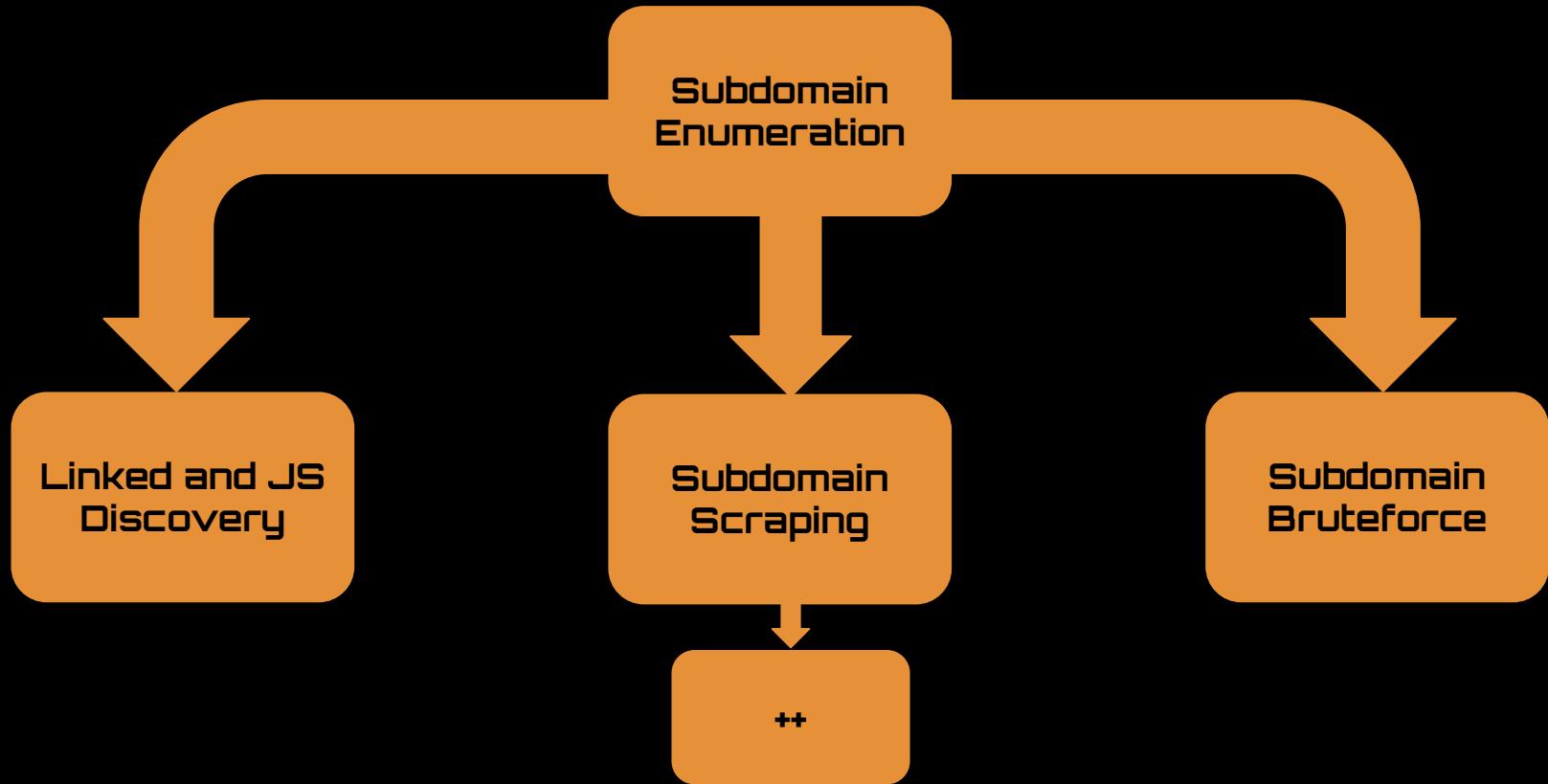
- 151.101.62.167** (Fasty):
 - SSL Certificate details (Issued By: GlobalSign, Common Name: twitch.map.fastly.net, Organization: Fasty, Inc.)
 - Supported SSL Versions: TLSv1.2
- 66.20.9.192** (Rackspace hosting):
 - SSL Certificate details (Issued By: Thawte TLS RSA CA, Common Name: www.geissele.com)
 - Supported SSL Versions: TLSv1.2
- 185.187.187.109** (Mesut Tunga trading as TUNGA Bilgi Teknolojileri V):
 - Counter-Strike: Global Offensive Server details (Name: Twitch.tv/bilbrain | Canlı, Players: 0/32, Operating System: Linux, Map: aim_reeline, Version: 1.36.7.3)
- 62.94.217.163** (Amazon Data Services Ireland Limited):
 - SSL Certificate details (Issued By: Amazon, Common Name: twitch.amazon.eu)
 - Supported SSL Versions: TLSv1, TLSv1.1, TLSv1.2

```
[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] (custom) HEADER parameter 'Cookie #1*' is dynamic
[05:06:31] [INFO] (custom) HEADER parameter 'Cookie #1*' is dynamic
[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\x0\x0\x0\x0\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'
```

Finding Subdomains

Subdomain Enumeration



```
[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] reflectivevalue(s) found and filtering out
[05:06:30] [INFO] confirming that (custom) HEADER parameter 'Cookie #1*' is dynamic
[05:06:31] [INFO] (custom) HEADER parameter 'Cookie #1*' is dynamic
[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 - (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\x0\x0\x0\x0\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'
```

Linked and JS Discovery

Linked Discovery (with Burp Suite Pro)

Another way to widen our scope is to examine all the links of our main target. We can do this using Burp Suite Pro.

We can visit a seed/root and recursively spider all the links for a term with regex, examining those links... and their links, and so on... until we have found all sites that could be in our scope.

This is a **hybrid** technique that will find **both** roots/seeds and subdomains.

1. Turn off passive scanning
2. Set forms auto to submit (if you're feeling frisky)
3. Set scope to advanced control and use “keyword” of target name (not a normal FQDN)
4. Walk+browse main site, then spider all hosts recursively!
5. Profit

Linked Discovery (with Burp Suite Pro)

Burp after requesting one site:

The screenshot shows a browser window displaying a Twitch stream for 'SethDrumsTV'. The stream title is 'Playing Drums & Enjoying Life [PG]'. Below the stream, there's a message from a viewer named 'shadow_atrium' saying 'just subscribed!'. The Burp Suite Pro interface is overlaid on the right side of the browser. The 'Target' tab is selected, showing a list of URLs that have been captured or are being monitored. These URLs include various Twitch-related domains and services like Amazon, Google, and GitHub.

Burp Suite Pro interface details:

- Target tab is selected.
- Captured URLs (partial list):
 - http://a
 - https://aax-eu.amazon-adsystem.com
 - http://a.amazon-adsystem.com
 - https://adservice.google.com
 - https://api.twitch.tv
 - https://apiservices.krnd.net
 - https://app.twitch.tv
 - https://beacon.krnd.net
 - https://blog.twitch.tv
 - http://c.amazon-adsystem.com
 - https://clients1.google.com
 - https://clips-
 - https://clips-alpha.twitch.tv
 - https://clips-beta.twitch.tv
 - https://clips-staging.twitch.tv
 - https://clips.twitch.tv
 - https://cm.g.doubleclick.net
 - https://connector.krnd.net
 - https://consumer.krnd.net
 - http://csgtatic.com
 - https://csgtatic.com
 - https://cvt.twitch.tv
 - https://d202tv0y9u9t.cloudfront.net
 - https://d3aqphizm0tq0.cloudfront.net
 - http://dai.google.com
 - https://dai.google.com
 - https://dev.twitch.tv
 - https://dpf-creative-service.twitch.tv
 - https://dpm.demdex.net
 - https://edge.quantserve.com
 - https://facebook.github.io
 - http://fb.me
 - http://feross.org
 - https://fp-keys-twitch-licensekeyserver.com
 - https://get.truex.com
 - https://git-aws.intel Justin.tv
 - http://github.com
 - https://github.com
 - https://ggt.twitch.tv
 - https://grsmto.github.io
 - https://help.twitch.tv
 - https://admixs.com
 - https://id-dev.twitch.tv
 - https://twitch.tv
 - https://idsync.rldn.com
 - https://imasdk.googleapis.com
 - https://inspector.twitch.tv
 - https://irc-ws.chat.twitch.tv
 - https://js.recurly.com
 - https://link.krnd.net
 - http://link.twitch.tv
 - https://link.twitch.tv
 - https://m.twitch.tv
 - https://match.adsvr.org
 - https://match.prod.bdr.org

Target Scope

Define the in-scope targets for your current work. This configuration affects the behavior of tools throughout the suite. The easiest way to configure scope is to browse to your target and use the context menus in the site map to include or exclude URL paths.

Use advanced scope control

Include in scope

- Add
- Edit
- Remove
- Paste URL
- Load ...

Exclude from scope

- Add
- Edit
- Remove
- Paste URL
- Load ...

Add URL to include in scope

Specify a regular expression to match each URL component, or leave blank to match any item. An IP range can be specified instead of a hostname.

Protocol: Any

Host or IP range: twitch

Port: Enter regex or leave blank

File: Enter regex or leave blank

Paste URL

OK Cancel

Burp Suite Professional v1.7.37 - Temporary Project - licensed to Jason Haddix [2 user license]

Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts

Site map Scope

Filter: Hiding not found items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders

Filter by request type

- Show only in-scope items
- Show only requested items
- Show only parameterized requests
- Hide not-found items

Filter by MIME type

- HTML
- Other text
- Script
- Images
- XML
- Flash
- Other binary
- CSS

Filter by status code

- 2xx [success]
- 3xx [redirection]
- 4xx [request error]
- 5xx [server error]

Folders

- Hide empty folders

Filter by search term

Filter by file extension

Filter by annotation

Show all Hide all Revert changes

Request Response

Raw Params Headers Hex

```
GET /_chromneveatab-serviceworker.js HTTP/1.1
Host: www.google.com
Connection: close
Pragma: no-cache
Cache-Control: no-cache
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/71.0.3578.90
Safari/537.36
Accept: /*
Service-Worker: script
X-Client-Data:
CjctyQEpbdAQCjEtskBCMdygElqEPKAQi/pSoBCOyngEI4qjKARiumMoBGPolygE=
Referer: https://www.google.com/_chromneveatab-serviceworker.js
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

Type a search term 0 matches

Linked Discovery (with Burp Suite Pro)

The screenshot shows the Burp Suite Pro interface. At the top, there's a navigation bar with tabs like Target, Proxy, Spider, Scanner, Intruder, Repeater, Sequencer, Decoder, Comparer, Extender, Project options, User options, and Alerts. Below that is a sub-navigation bar with Site map and Scope.

The main area displays a "Site map" showing a large number of URLs under the "https://twitch." domain, such as https://api.twitch.tv, https://app.twitch.tv, https://blog.twitch.tv, etc. A red arrow points from the text "Seems good..." down towards this list.

To the right of the site map, there's a "Contents" table with columns for Host, Method, URL, Params, Status, Length, and MIME type. The table is currently empty.

At the bottom of the interface, there are tabs for Request and Response, and buttons for Raw and Hex.

Seems good... let's try spidering all these too...

Linked Discovery (with Burp Suite Pro)

After the 1st spider run we've now discovered a ton of linked URLs that belong to our project.

Not only subdomains, but **NEW** seeds/roots (twtchapp.net, ext-twitch.tv, twitchsvc.net).

We can also now spider these new hosts and repeat until we have Burp Spider fatigue.

The screenshot shows the Burp Suite Pro interface. At the top, there's a navigation bar with tabs like Target, Proxy, Spider, Scanner, Intruder, Repeater, Sequencer, Decoder, Comparer, Extender, Project options, User options, and Alerts. Below that is a Site map tab. The main area has a 'Filter' dropdown set to 'Hiding out of scope and not found items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders'. On the left, a tree view lists many URLs under the root 'https://affiliate.twitch.tv', with several highlighted in orange and one specifically labeled 'https://polylill.twitchsvc.net' highlighted with a red arrow. To the right is a 'Contents' table with columns: Host, Method, URL, Params, Status, Length, MIME type, and Title. The table lists various Twitch-related URLs. Below the table are tabs for Request, Response, Raw, Params, Headers, and Hex. A detailed request message is shown in the Request tab. On the far right, there's a sidebar titled 'Issues' with a list of findings and a cartoon character at the bottom.

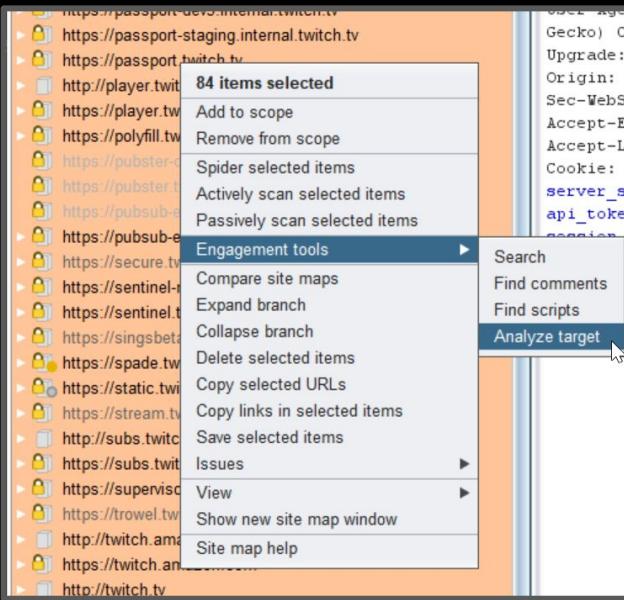
Host	Method	URL	Params	Status	Length	MIME type	Title
https://irc-ws.chat.t...	GET	/		101	129		
https://pubsub-edge...	GET	/v1		101	129		
https://help.twitch.t...	GET	/		200	112313	HTML	Twitch Portal
https://im.twitch.t...	GET	/		200	725837	HTML	All Games - Twitch
https://player.twitch.t...	GET	/		200	1524	HTML	Twitch
https://warcraftontwit...	GET	/		200	1192	HTML	
https://www.twitch.t...	GET	/		200	80196	HTML	Twitch
https://www.twitch.t...	GET	/well-known/assetli...		200	1255	JSON	

Linked Discovery (with Burp Suite Pro)

Now that we have this data, how do we export it?

Clumsily =(

- 1) Select all hosts in the site tree
- 2) In PRO ONLY right click the selected hosts
- 3) Go to “Engagement Tools” -> “Analyze target”
- 4) Save report as an html file
- 5) Copy the hosts from the “Target” section



Target analysis

Targets

- https://affiliate.twitch.tv/
- https://api.twitch.tv/
- https://app.twitch.tv/
- https://badges.twitch.tv/
- https://bits.twitch.tv/
- http://blog.twitch.tv/
- https://blog.twitch.tv/
- https://bttsqiy6dnv05acplp5vy0mfigrh3z.ext-twitch.tv/
- https://client-event-reporter-darklaunch.twitch.tv/
- https://client-event-reporter.twitch.tv/
- https://clips-alpha.twitch.tv/
- https://clips-beta.twitch.tv/
- https://clips-staging.twitch.tv/
- https://clips.twitch.tv/
- https://countess.twitch.tv/
- https://cvp.twitch.tv/
- https://d4uvfd04uq6raoenvj7m86gdk16v.ext-twitch.tv/
- http://dev.twitch.tv/
- https://dev.twitch.tv/
- https://dfp-creative-service.twitch.tv/
- http://discuss.dev.twitch.tv/
- https://discuss.dev.twitch.tv/
- https://download.twitch.tv/
- https://embed.twitch.tv/
- https://gql.twitch.tv/
- http://help.twitch.tv/
- https://help.twitch.tv/
- https://id-dev.twitch.tv/
- https://id.twitch.tv/
- https://inspector.twitch.tv/
- https://irc-ws.chat.twitch.tv/
- https://jira.twitch.com/
- https://launcher.twitch.tv/
- http://link.twitch.tv/
- https://link.twitch.tv/
- https://m.twitch.tv/
- https://mobile.twitch.tv/
- http://music.twitch.tv/
- https://music.twitch.tv/
- https://notifications-v1.twitchapp.net/
- https://passport-dev1.internal.twitch.tv/
- https://passport-dev2.internal.twitch.tv/
- https://passport-dev3.internal.twitch.tv/
- https://passport-staging.internal.twitch.tv/
- https://passport.twitch.tv/
- http://player.twitch.tv/

Linked Discovery (with GoSpider or hakrawler)

Linked discovery really just counts on using a spider recursively.

One of the most extensible spiders for general automation is [GoSpider](#) written by [i3ssiejjj](#) which can be used for many things and supports parsing js very well.

In addition [hakrawler](#) by [hakluke](#) has many parsing strategies that interest bug hunters.

```
root@IBox4:~/tools# gospider -s https://www.twitch.tv
[robots] - https://www.twitch.tv/
[robots] - https://www.twitch.tv/directory
[robots] - https://www.twitch.tv/directory/all
[robots] - https://www.twitch.tv/directory/*
[robots] - https://www.twitch.tv/videos/week
[robots] - https://www.twitch.tv/.well-known/assetlinks.json
[robots] - https://www.twitch.tv/admin/*
[robots] - https://www.twitch.tv/email-unsubscribe/
[robots] - https://www.twitch.tv/login$
[robots] - https://www.twitch.tv/message/*
[robots] - https://www.twitch.tv/signup$
[robots] - https://www.twitch.tv/user/*
[subdomains] - api.twitch.tv
[subdomains] - passport.twitch.tv
[subdomains] - gql.twitch.tv
[subdomains] - cvp.twitch.tv
[subdomains] - irc-ws.chat.twitch.tv
[subdomains] - pubsub-edge.twitch.tv
[subdomains] - spade.twitch.tv
[subdomains] - www.twitch.tv
[url] - [code-200] - https://www.twitch.tv
[javascript] - https://polyfill.twitchsvc.net/
.prototype.includes,default,fetch,Intl,~locale
[javascript] - https://static.twitchcdn.net/co
[javascript] - https://static.twitchcdn.net/as
[javascript] - https://static.twitchcdn.net/as
[url] - [code-200] - https://www.twitch.tv/.we
[subdomains] - m.twitch.tv
[url] - [code-200] - https://www.twitch.tv/
[url] - [code-200] - https://www.twitch.tv/dir
[url] - [code-200] - https://www.twitch.tv/mes
```



```
[robots] http://bugcrowd.com/*preview
[sitemap] https://bugcrowd.com/
[sitemap] https://bugcrowd.com/contact/
[sitemap] https://bugcrowd.com/faq/
[sitemap] https://bugcrowd.com/leaderboard/
[sitemap] https://bugcrowd.com/list-of-bug-bounty-programs/
[sitemap] https://bugcrowd.com/press/
[sitemap] https://bugcrowd.com/pricing/
[sitemap] https://bugcrowd.com/privacy/
[sitemap] https://bugcrowd.com/term/
[sitemap] https://bugcrowd.com/resources/responsible-disclosure-program/
[sitemap] https://bugcrowd.com/resources/why-care-about-web-security/
[sitemap] https://bugcrowd.com/resources/what-is-a-bug-bounty/
[sitemap] https://bugcrowd.com/stories/november/
[sitemap] https://bugcrowd.com/stories/riskto/
[sitemap] https://bugcrowd.com/stories/tagged/
[sitemap] https://bugcrowd.com/tour/
[sitemap] https://bugcrowd.com/tour/platform/
[sitemap] https://bugcrowd.com/tour/crowd/
[sitemap] https://bugcrowd.com/customers/programs/new
[sitemap] https://bugcrowd.com/portal/
[sitemap] https://bugcrowd.com/portal/user/sign_in
[sitemap] https://bugcrowd.com/portal/user/sign_up/
[url] https://bugcrowd.com/user/sign_in
[subdomain] bugcrowd.com
[url] https://tracker.bugcrowd.com/user/sign_in
[subdomain] tracker.bugcrowd.com
[url] https://www.bugcrowd.com/
[subdomain] www.bugcrowd.com
[url] https://www.bugcrowd.com/products/how-it-works/
[url] https://www.bugcrowd.com/products/how-it-works/the-bugcrowd-difference/
[url] https://www.bugcrowd.com/products/platform/
[url] https://www.bugcrowd.com/products/platform/integrations/
[url] https://www.bugcrowd.com/products/platform/vulnerability-rating-taxonomy/
[url] https://www.bugcrowd.com/products/attack-surface-management/
[url] https://www.bugcrowd.com/products/bug-bounty/
[url] https://www.bugcrowd.com/products/vulnerability-disclosure/
[url] https://www.bugcrowd.com/products/next-gen-pen-test/
[url] https://www.bugcrowd.com/products/bug-bash/
[url] https://www.bugcrowd.com/resources/ceritols-nearity-one-report
```

Subdomain Enumeration (with SubDomainizer)

[SubDomainizer](#) by Neeraj Edwards is a tool with three purposes in analyzing javascript. It will take a page and run

1. Find subdomains referenced in js files
2. Find cloud services referenced in js files
3. Use the **Shannon Entropy** formula to find potentially sensitive items in js files

It will take a single page and scan for js files to analyze.

```
key": "BGzteaQY0qrTAPN8EuuowBVG67pHwyZo879XZkC7cUV2QP4qQf-92Pmm9ty0uriJdiKnMDDRi28F5HQK6uSk0vM"  
Key="6Ld65QcTAAAAMBbAE8dkJq4Wi4CsJy7flvKhYqX"
```

If just looking for subdomains [subscraper](#) by Cillian-Collins might be better because it has recursion.

```
Finding Subdomains and cloud data of given domain in all Javascript files...
Searching completed...
Got all the important data.

~~~~~RESULTS~~~~~

Some cloud services url's found. They might be interesting, Here are the URLs:

d2v0zitv8y9u9t.cloudfront.net
d3aqoihi2n8ty8.cloudfront.net

Successfully got all the subdomains...

Total Subdomains: 35
static.twitchcdn.net
api.twitch.tv
app.twitch.tv
blog.twitch.tv
canary.twitch.tv
irc-ws.chat.twitch.tv
client-event-reporter.twitch.tv
client-event-reporter-darklaunch.twitch.tv
clips.twitch.tv
cvp.twitch.tv
cvp-test.twitch.tv
dev.twitch.tv
glass.twitch.tv
gql.twitch.tv
help.twitch.tv
id.twitch.tv
id-dev.twitch.tv
passport-dev1.internal.twitch.tv
passport-dev2.internal.twitch.tv
passport-dev3.internal.twitch.tv
passport-staging.internal.twitch.tv
music.twitch.tv
passport.twitch.tv
player.twitch.tv
player-core-web.twitch.tv
pubster.twitch.tv
pubster-darklaunch.twitch.tv
pubsub-edge.twitch.tv
pubsub-edge-darklaunch.twitch.tv
rc.twitch.tv
release.twitch.tv
spade.twitch.tv
tmi.twitch.tv
vizima.twitch.tv
www.twitch.tv
```



```
[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] (custom) HEADER parameter 'Cookie #1*' is dynamic
[05:06:31] [INFO] (custom) HEADER parameter 'Cookie #1*' is dynamic
[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\x0\x0\x0\x0\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'
```

Subdomain Scraping

Subdomain Scraping Sources

Search Sources

The next set of tools scrape domain information from all sorts of projects that expose databases of URLs or domains.

New sources are coming out all the time so the tools must evolve constantly.

This is only a small list of sources. Many more exist.

Infrastructure Sources



Certificate Sources



Security Sources



Subdomain Scraping Example (Google)

1. site:twitch.tv -www.twitch.tv
2. site:twitch.tv -www.twitch.tv -watch.twitch.tv
3. site:twitch.tv -www.twitch.tv -watch.twitch.tv -dev.twitch.tv
4. ...

The screenshot shows a Google search results page with the following details:

- Search Query:** site:twitch.tv -www.twitch.tv -watch.twitch.tv -dev.twitch.tv
- Results Count:** About 28,500,000 results (4.88 seconds)
- First Result:**
 - Title:** TwitchOrlando (@TwitchOrlando) | Twitter
 - Link:** link.twitch.tv/TwitchOrlando
 - Description:** Orlando area community of streamers & viewers of @Twitch that meet up & network. ... Community @TwitchCon impromptu meetup Today! ... If you have moved away from Florida, you are still welcome to come by.
- Second Result:**
 - Title:** Amazon.com: Atari 80 Classic Games in One [Download]: Video Games
 - Link:** link.twitch.tv/AtariClassicsDeal
 - Description:** Collection includes both arcade and Atari 2600 titles. ... Atari 80 Classic Games in One provides a host of classic arcade games, including popular titles such as Asteroids, Centipede and Millipede. This

Subdomain Scraping (Amass)

For scraping subdomain data there are two industry leading tools at the moment; Amass and Subfinder. They parse all the “sources” referenced in the previous slide, and more.

Amass has the most sources, extensible output, bruteforcing, permutation scanning, and a ton of other modes to do additional analysis of attack surfaces.

```
root@Test2:~/tools/amass# amass -d twitch.tv
twitch.tv
passport-external.aws.twitch.tv
gql.twitch.tv
pubsub-edge.twitch.tv
pubsub-edge.chat.twitch.tv
passport.twitch.tv
www.twitch.tv
m.twitch.tv
irc-ws-edge.chat.twitch.tv
irc-ws.chat.twitch.tv
app.twitch.tv
download.twitch.tv
discuss.dev.twitch.tv
invite.twitch.tv
join.twitch.tv
edgecast.hls.twitch.tv has a static DNS wildcard
blog.twitch.tv
polls.twitch.tv
th.blog.twitch.tv
link.twitch.tv
servers.twitch.tv
cis.blog.twitch.tv
graphql.prod.us-west2.twitch.tv
it.blog.twitch.tv
rc.twitch.tv
de.blog.twitch.tv
tr.blog.twitch.tv
release.twitch.tv
nl.blog.twitch.tv
canary.twitch.tv
ccu.event-engineering.twitch.tv
pong.prod.us-west2.twitch.tv
jp.blog.twitch.tv
event-panel.event-engineering.twitch.tv
assets.help.twitch.tv
uploads-regional.twitch.tv
websub-test-proxy.twitch.tv
```

Subdomain Scraping (Amass)

Amass also correlates these scraped domains to ASNs and lists what network ranges they appeared in.

Useful.

If a new ASN is discovered you can feed it back to amass intel.

Average DNS names processed: 49/sec		
OWASP Amass v2.9.0		
https://github.com/OWASP/Amass		
439 names discovered - alt: 77, dns: 3, cert: 105, archive: 24, scrape: 218, api: 12		
ASN: 54113 - FASTLY - Fastly	2	Subdomain Name(s)
151.101.64.0/22	2	Subdomain Name(s)
151.101.0.0/22	2	Subdomain Name(s)
151.101.188.0/22	52	Subdomain Name(s)
151.101.40.0/22	1	Subdomain Name(s)
151.101.244.0/22	1	Subdomain Name(s)
151.101.192.0/22	2	Subdomain Name(s)
151.101.128.0/22	2	Subdomain Name(s)
ASN: 16509 - AMAZON-02 - Amazon.com, Inc., US		
52.24.0.0/14	35	Subdomain Name(s)
52.84.48.0/23	4	Subdomain Name(s)
13.35.8.0/23	4	Subdomain Name(s)
34.208.0.0/12	15	Subdomain Name(s)
35.160.0.0/13	16	Subdomain Name(s)
54.254.128.0/17	1	Subdomain Name(s)
54.186.0.0/15	9	Subdomain Name(s)
52.18.0.0/15	2	Subdomain Name(s)
52.36.0.0/14	6	Subdomain Name(s)
35.178.0.0/15	1	Subdomain Name(s)
2600:9000:20d::/48	54	Subdomain Name(s)
52.40.0.0/14	5	Subdomain Name(s)
54.192.144.0/22	12	Subdomain Name(s)
52.220.0.0/15	1	Subdomain Name(s)
54.171.0.0/16	2	Subdomain Name(s)
3.8.0.0/14	1	Subdomain Name(s)
54.192.12.0/22	55	Subdomain Name(s)
2600:9000:2001::/48	26	Subdomain Name(s)
54.148.0.0/15	10	Subdomain Name(s)
2600:9000:204b::/48	8	Subdomain Name(s)
52.10.0.0/15	3	Subdomain Name(s)
2600:9000:201d::/48	8	Subdomain Name(s)
52.84.44.0/22	1	Subdomain Name(s)
52.88.0.0/15	7	Subdomain Name(s)
2600:9000:2145::/48	8	Subdomain Name(s)
184.169.128.0/17	1	Subdomain Name(s)
52.208.0.0/13	3	Subdomain Name(s)
52.32.0.0/14	11	Subdomain Name(s)
13.35.124.0/22	44	Subdomain Name(s)
50.112.128.0/19	3	Subdomain Name(s)
52.52.0.0/15	3	Subdomain Name(s)
54.215.128.0/18	2	Subdomain Name(s)
54.68.0.0/15	2	Subdomain Name(s)
54.191.0.0/16	3	Subdomain Name(s)
54.70.0.0/15	2	Subdomain Name(s)
54.200.0.0/15	6	Subdomain Name(s)
50.18.0.0/18	1	Subdomain Name(s)
ASN: 0 - Private Networks		
10.0.0.0/8	66	Subdomain Name(s)
ASN: 46489 - JUSTINTV - Twitch Interactive Inc., US		
52.223.240.0/20	21	Subdomain Name(s)
192.16.64.0/21	32	Subdomain Name(s)
99.181.64.0/20	2	Subdomain Name(s)
52.223.224.0/20	27	Subdomain Name(s)
45.113.128.0/22	10	Subdomain Name(s)
192.108.239.0/24	4	Subdomain Name(s)
52.223.208.0/21	28	Subdomain Name(s)
199.9.248.0/21	164	Subdomain Name(s)
185.42.204.0/22	15	Subdomain Name(s)

ASN: 46489 - JUSTINTV - Twitch Interactive Inc., US		
52.223.240.0/20	21	Subdomain Name(s)
192.16.64.0/21	32	Subdomain Name(s)
99.181.64.0/20	2	Subdomain Name(s)
52.223.224.0/20	27	Subdomain Name(s)
45.113.128.0/22	10	Subdomain Name(s)
192.108.239.0/24	4	Subdomain Name(s)
52.223.208.0/21	28	Subdomain Name(s)
199.9.248.0/21	164	Subdomain Name(s)
185.42.204.0/22	15	Subdomain Name(s)
ASN: 40341 - Q9-AS-CAL2 - Q9 Networks Inc., CA		
162.219.8.0/21	1	Subdomain Name(s)
ASN: 14618 - AMAZON-AES - Amazon.com, Inc., US		
54.84.0.0/15	1	Subdomain Name(s)
52.72.0.0/15	1	Subdomain Name(s)
52.0.0.0/15	71	Subdomain Name(s)
2600:1f18::/33	1	Subdomain Name(s)
52.2.0.0/15	2	Subdomain Name(s)
18.204.0.0/14	3	Subdomain Name(s)
52.200.0.0/13	4	Subdomain Name(s)
52.4.0.0/14	137	Subdomain Name(s)
52.20.0.0/14	1	Subdomain Name(s)
ASN: 395224 - BITLY-AS - Bitly Inc, US		
67.199.248.0/24	4	Subdomain Name(s)
ASN: 15169 - GOOGLE - Google LLC, US		
35.185.0.0/19	1	Subdomain Name(s)
ASN: 22606 - EXACT-7 - ExactTarget, Inc., US		
13.111.18.0/24	5	Subdomain Name(s)
13.111.19.0/24	1	Subdomain Name(s)
13.111.20.0/24	1	Subdomain Name(s)
13.111.97.0/24	1	Subdomain Name(s)
ASN: 11377 - SENDGRID - SendGrid, Inc., US		
167.89.64.0/19	1	Subdomain Name(s)
ASN: 38895 - AMAZON-AS-AP Amazon.com Tech Telecom, JP		
2600:9000:20c7::/48	8	Subdomain Name(s)

Subdomain Scraping (Subfinder v2)

[Subfinder](#) is another best in breed tool originally written by ice3man and Michael Skelton.

It is now maintained by a larger group, called projectdiscovery.io

It incorporates multiple sources, has extensible output, and more.

```
root@b0x:~# subfinder -d hackerone.com -v
____ _[ ]_ / [ ]_ _ _[ ]_ ___ _ _ 
(_\| | ' \_ \| | ' \| _ / _ ) ' \| 
/_\_, \| . / \| \| \| _ \| _ \| v2

projectdiscovery.io

[WRN] Use with caution. You are responsible for your actions
[WRN] Developers assume no liability and are not responsible for any misuse or damage.
[WRN] By using subfinder, you also agree to the terms of the APIs used.

[INF] Enumerating subdomains for hackerone.com
[sitedossier] www.hackerone.com
[virustotal] api.hackerone.com
[virustotal] support.hackerone.com
[virustotal] docs.hackerone.com
[virustotal] mta-sts.hackerone.com
[virustotal] mta-sts.forwarding.hackerone.com
[virustotal] a.ns.hackerone.com
[virustotal] b.ns.hackerone.com
[virustotal] links.hackerone.com
[virustotal] info.hackerone.com
[archiveis] hackerone.com
[securitytrails] email.gh-mail.hackerone.com
[securitytrails] mta-sts.managed.hackerone.com
[securitytrails] web-seo-content-for-business.theflyingkick.websitedesignresource.api.hackerone.com
[passivetotal] cf-ssl5349-protected-cover-photos-user-content.hackerone.com
[passivetotal] o1.email.hackerone.com
[passivetotal] go.hackerone.com
[passivetotal] cf-ssl5349-protected-profile-photos-user-content.hackerone.com
[passivetotal] o3.email.hackerone.com
[passivetotal] profile-photos-user-content.hackerone.com
[passivetotal] cf-ssl41462-protected-profile-photos-user-content.hackerone.com
[passivetotal] cover-photos-user-content.hackerone.com
[passivetotal] staging.hackerone.com
[passivetotal] cf-ssl41462-protected-cover-photos-user-content.hackerone.com
```

Subdomain Scraping (github-subdomains.py)

```
root@TBox4:~/tools/github-search# python3 github-subdomains.py -t ".twitch.tv" -d twitch.tv > twitch.tv
root@TBox4:~/tools/github-search# cat twitch.tv | wc -l
1828
root@TBox4:~/tools/github-search# cat twitch.tv | grep -v ".tmi"
player.twitch.tv
www.twitch.tv
api.twitch.tv
blog.twitch.tv
passport.twitch.tv
m.twitch.tv
rechat.twitch.tv
go.twitch.tv
clips.twitch.tv
dev.twitch.tv
irc.chat.twitch.tv
id.twitch.tv
link.twitch.tv
discuss.dev.twitch.tv
usher.twitch.tv
secure.twitch.tv
live.twitch.tv
irc-ws.chat.twitch.tv
```

A new addition to my subdomain enumeration is scraping Github. Github-subdomains.py is a script written by [Gwendal Le Coguic](#) as part of his EPIC Github enumeration repo called “[github-search](#)” It will query the Github API for references to a root and pull out subdomains.

Note: the Github API returns somewhat random results and is rate limited. In my automation I run 5 iterations of this script. Four of them with a six second sleep in between them, and the last with a 10 second sleep, to get some consistency.



Subdomain Scraping (shosubgo)

Another valuable bespoke scraping technique is gathering subdomains from Shodan.

[Shosubgo](#) is a Go script written by [inc0qbyt3](#) which is effective and reliable for this method.

```
root@TBox4:~/tools/shosubgo# go run main.go -d twitch.tv -s
*.canary.twitch.tv
*.rc.twitch.tv
*.release.twitch.tv
api-a.chat.twitch.tv
api.globetrotter.external.twitch.tv
app.twitch.tv
assets.help.twitch.tv
badges.twitch.tv
ccu.event-engineering.twitch.tv
chatdepot.twitch.tv
click.twitch.tv
click-staging.twitch.tv
clips-canary.twitch.tv
clock.event-engineering.twitch.tv
costream.twitch.tv
download-staging.sings.twitch.tv
es-es.twitch.tv
event-panel.event-engineering.twitch.tv
gap-v4-cubic.event-engineering.twitch.tv
go.twitch.tv
id.twitch.tv
id-canary.twitch.tv
im.twitch.tv
ingest.twitch.tv
irc-ws.chat.twitch.tv
irc.chat.twitch.tv
ja.twitch.tv
lv.twitch.tv
metadata.twitch.tv
moonlight.twitch.tv
owl.internal.us-west-2.twitch.tv
passport.twitch.tv
pong.twitch.tv
production.helix-origin.twitch.tv
pubsub-edge.twitch.tv
pubsub-edge.chat.twitch.tv
spectre.twitch.tv
staging.helix-origin.twitch.tv
staging.passport.twitch.tv
stg.moonlight.twitch.tv
tmi.twitch.tv
ttv-redirect.m7g.twitch.tv
us-west-2.uploads-regional.twitch.tv
vpn-anycast-2.twitch.tv
```



Subdomain Scraping (Cloud Ranges)

A highly valuable technique is to monitor whole cloud ranges of AWS, GCP, and Azure for SSL sites, and parse their certificates to match your target.

Doing this is cumbersome on your own but possible with something like masscan. [Daehee Park outlines it here.](#)

Luckily [Sam Erb](#) did a wonderful [defcon talk](#) on this and created a service which scans every week.

Some bash scripting required ;)

```
root@TBox4:~/tools/sslScrape# curl 'https://tls.bufferover.run/dns?q=.twitch.tv' 2>/dev/null | jq .Results
*.canary.twitch.tv
*.chat.twitch.tv
*.dev.us-west2.twitch.tv
*.go.twitch.tv
*.m.twitch.tv
*.rc.twitch.tv
*.release.twitch.tv
*.tv.twitch.tv
*.twitch.tv
[
]
app.twitch.tv
canary-extensions-worker.sings.twitch.tv
canary-extensions.sings.twitch.tv
click-staging.twitch.tv
click.marketing.twitch.tv
click.twitch.tv
cloud.marketing.twitch.tv
countess.twitch.tv
dev-extensions-worker.sings.twitch.tv
dev-extensions.sings.twitch.tv
dev.twitch.tv
dev.us-west2.twitch.tv
discuss.dev.twitch.tv
dmca.twitch.tv
download.twitch.tv
extensions-worker.sings.twitch.tv
extensions.sings.twitch.tv
i18n.m7g.twitch.tv
id.twitch.tv
image.marketing.twitch.tv
invite.twitch.tv
join.twitch.tv
lvs-prod.twitch.tv
m.twitch.tv
metadata.twitch.tv
pages.marketing.twitch.tv
passport.twitch.tv
polls.twitch.tv
pong-staging.twitch.tv
```



```
[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 file(s) found and filtering out
[05:06:30] [INFO] confirming that (custom) HEADER parameter 'Cookie #1*' is dynamic
[05:06:31] [INFO] (custom) HEADER parameter 'Cookie #1*' is dynamic
[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 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\x0\x0\x0\x0\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'
```

Subdomain Bruteforce

Subdomain Bruteforce

At this point we move into guessing for live subdomains.

If we try and resolve

`this totally does not exist.company.com` we will *usually* not get a record.

So we can use a large list of common subdomain names and just try and resolve them analyzing if they succeed.

The problem in this method is that only using one DNS server to do this will take forever. Some tools have come out that are both threaded and use multiple DNS resolvers simultaneously. This speeds up this process significantly. **Massdns** by [@blechschmidt](#) pioneered this idea.

Amass (8 revolvers by default) does this with the `-rf` flag.

Subdomain Bruting (Amass)

Amass offers bruteforcing via the “enum” tool using the “brute” switch.

- `amass enum -brute -d twitch.tv -src`

It has a built in list but you can specify your own lists.

You can also specify any number of resolvers

- `amass enum -brute -d twitch.tv -rf
resolvers.txt -w bruteforce.list`

Doing this with amass also gives us the opportunity to resolve the found domains.

I haven't checked out [aisdnsbrute](#) yet but i've heard it's fast.

```
Starting DNS queries for brute forcing
[Crush]          costream.twitch.tv
[Crush]          tr.blog.twitch.tv
[Crush]          www.cis.blog.twitch.tv
[Crush]          kr.blog.twitch.tv
[Crush]          affiliate.twitch.tv
[Crush]          fr.blog.twitch.tv
[Crush]          clock.event-engineering.twitch.tv
[Crush]          rivals.twitch.tv
[Brute Forcing]  staging.passport.twitch.tv
[ThreatCrowd]    view.sfmarketing.twitch.tv
[Sublist3rAPI]   mta2.sfmarketing.twitch.tv
[ThreatCrowd]    click.sfmarketing.twitch.tv
[Sublist3rAPI]   mta.marketing.twitch.tv
[Sublist3rAPI]   mta.sfmarketing.twitch.tv
[Sublist3rAPI]   mta.e.community.twitch.tv
[Crush]          pages.marketing.twitch.tv
[Crush]          cloud.marketing.twitch.tv
[Crush]          click.marketing.twitch.tv
[Crush]          view.marketing.twitch.tv
[ThreatCrowd]    sfmarketing.twitch.tv
[Crush]          link.twitch.tv
[Crush]          watch.twitch.tv
[AlienVault]     www.link.twitch.tv
Average DNS queries performed: 3173/sec
```

Subdomain Bruteforce (shuffleDNS)

```
root@b0x:~# shuffledns -d hackerone.com -w words.txt -r resolvers-excellent.txt
```

```
____ / / _ _ / / / / _ _ / / _  
(_-< _ \ \ / / / / / -) / - / - \(_-  
/ / / / / \_, / / / / \_, / / / / \_, / v1
```

projectdiscovery.io

```
[WRN] Use with caution. You are responsible for your actions  
[WRN] Developers assume no liability and are not responsible for any misuse or damage.  
[INF] Started generating bruteforce permutation  
[INF] Generating permutations took 46.728µs  
[INF] Creating temporary massdns output file: /tmp/shuffledns474707985/bpb1p67dd9u0me2rqs0  
[INF] Executing massdns on hackerone.com  
[INF] Massdns execution took 165.709341ms  
[INF] Parsing output and removing wildcards  
[INF] Finished enumeration, started writing output  
support.hackerone.com  
b.ns.hackerone.com  
api.hackerone.com  
www.hackerone.com  
docs.hackerone.com  
mta-sts.managed.hackerone.com  
mta-sts.forwarding.hackerone.com  
mta-sts.hackerone.com  
a.ns.hackerone.com  
[INF] Finished resolving. Hack the Planet!
```

If you like separating the work to different tools or prefer the massdns core you can use [shuffleDNS](#) by the ProjectDiscovery team.

Subdomain Brutting Lists

A multi resolver, threaded subdomain bruter is only as good as it's wordlist.

There are two trains of thought here:

- Tailored wordlists
- Massive wordlists

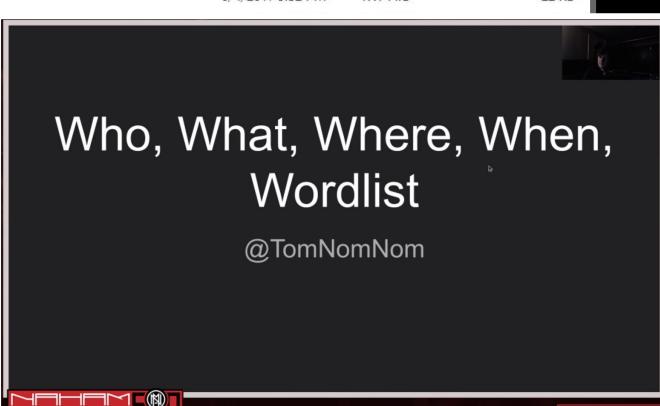
Both have advantages.

My [all.txt](#) file is still what i use on a regular basis. It combines 7 years of DNS bruteforce lists into one.

But you can make customized wordlists with something like what TomNomNom [talked on](#) yesterday.

I also think there was a tool for this recently...

 bluto_lots-of-spinach.txt	6/4/2017 9:42 PM	TXT File	1,946 KB
 deepmagic.com_top50kprefixes.txt	8/20/2015 2:58 PM	TXT File	592 KB
 deepmagic.com_top500prefixes.txt	8/20/2015 2:58 PM	TXT File	4 KB
 dns_raft-large-words-lowercase.txt	6/4/2017 9:56 PM	TXT File	920 KB
 dns_top_100000_RobotsDissalowed.txt	6/4/2017 10:23 PM	TXT File	1,578 KB
 dnscan_subdomains.txt	7/31/2016 3:00 PM	TXT File	5 KB
 dnscan_subdomains-100.txt	7/31/2016 3:00 PM	TXT File	1 KB
 dnscan_subdomains-500.txt	7/31/2016 3:00 PM	TXT File	3 KB
 dnscan_subdomains-1000.txt	7/31/2016 3:00 PM	TXT File	6 KB
 dnscan_subdomains-10000.txt	7/31/2016 3:00 PM	TXT File	62 KB
 dnscan_subdomains-uk-500.txt	7/31/2016 3:00 PM	TXT File	4 KB
 dnscan_subdomains-uk-1000.txt	7/31/2016 3:00 PM	TXT File	7 KB
 dnscan_suffixes.txt	7/31/2016 3:00 PM	TXT File	36 KB
 dnscan_tlds.txt	7/31/2016 3:00 PM	TXT File	9 KB
 dnsenum_dns.txt	6/4/2017 9:24 PM	TXT File	15 KB
 dnspop_bitquark_20160227_subdomains_popular_1000.txt	3/10/2016 3:47 PM	TXT File	5 KB
 dnspop_bitquark_20160227_subdomains_popular_10000.txt	3/10/2016 3:47 PM	TXT File	92 KB
 dnspop_bitquark_20160227_subdomains_popular_100000.txt	3/10/2016 3:47 PM	TXT File	1,393 KB
 dnsrecon_meatsploit_standard_namelist.txt	3/10/2016 3:47 PM	TXT File	11,371 KB
 dnsrecon_subdomains-top1mil-5000.txt	5/19/2017 3:06 AM	TXT File	12 KB
 dnsrecon_subdomains-top1mil-20000.txt	1/16/2017 6:03 PM	TXT File	33 KB
 dnsrecon_subdomains-top1mil-110000.txt	1/16/2017 6:03 PM	TXT File	146 KB
 ethicalhack3r_subdomains.txt	1/16/2017 6:03 PM	TXT File	1,092 KB
 fierce_hostlist.txt	6/4/2017 9:30 PM	TXT File	6 KB
 hostilebruteforcer.txt	8/20/2015 2:58 PM	TXT File	15 KB
 knock_wordlist.txt	6/4/2017 9:32 PM	TXT File	22 KB
 master.txt			
 nmap_vhosts-default.list.txt			
 recon-ng_hostnames.txt			
 reverseraider_fast.list.txt			
 reverseraider_services.list.txt			
 reverseraider_word.list.txt			
 sorted_knock_dnsrecon_fierce_recon			
 subbrute_names.txt			



Subdomain Bruteforce Lists

New lists for subdomain bruteforce are relatively the same nowadays, but the 1st team to really iterate on this is the [AssetNote](#) team with their commonspeak data collection.

The all.txt file includes commonspeak v1 data but there is also a second version of commonspeak data out:

- <https://github.com/assetnote/commonspeak2>

Commonspeak2



Commonspeak2 leverages publicly available datasets from Google BigQuery to generate content discovery and subdomain wordlists.

As these datasets are updated on a regular basis, the wordlists generated via Commonspeak2 reflect the current technologies used on the web.

By using the Golang client for BigQuery, we can stream the data and process it very quickly. The future of this project will revolve around improving the quality of wordlists generated by creating automated filters and substitution functions.

Let's turn creating wordlists from a manual task, into a reproducible and reliable science with BigQuery.

I just want the wordlists...

We will update the [commonspeak2-wordlists](#) repo with any wordlists generated the Commonspeak2 tool.

More infrastructure will be developed to deliver wordlists continuously and this section will be updated in the future.

Alteration Scanning

When bruteforcing or gathering subdomains via scraping you may come across a naming pattern in these subdomains.

Even though you may not have found it yet, there may be other targets that conform to naming conventions.

In addition, sometimes targets are not explicitly protected across naming conventions.

The first tool to attempt to recognize these patterns and brute-force for some of them was [altdns](#) written by [Naffy](#) and [Shubs](#).

Now Amass contains logic to check for these “permutations”. Amass includes this analysis in a default run. Some personal experience cited on the next page.

dev.company.com

dev1.company.com

dev2.company.com

dev-1.company.com

Dev-2.company.com

Dev.1.company.com

Dev.2.company.com

Alteration Scanning (WAF Bypass)

Jason Haddix
@Jhaddix

WAF had me down on www.\$target.com ='(

too bad they missed ww2.\$target.com !

sqli in progress...

#OMGSOMANYTABLESToEXFIL

8:14 PM - 16 Feb 2018

6 Retweets 104 Likes

Jason Haddix
@Jhaddix

Security testing against Akamai? look for
[origin-sub.domain.com](#) or
[origin.sub.domain.com](#) , bypass the filtering
by going to the source.

12:06 PM - 13 Sep 2017

43 Retweets 95 Likes

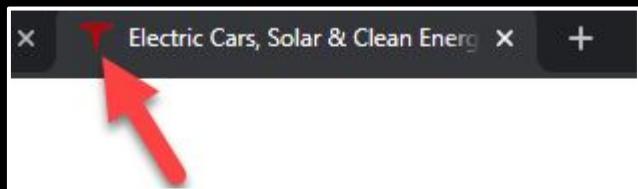
```
[05:06:24] [INFO] loading tamper script 'xforwardedfor'  
[05:06:24] [WARNING] using too many tamper scripts is usually not a good idea  
custom injection marking character ('*') found in option '--headers/--user-agent/--referer/--cookie'  
. Do you want to process it? [Y/n/q] Y  
[05:06:28] [INFO] 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 (custom) HEADER parameter 'Cookie #1*' is dynamic  
[05:06:31] [INFO] (custom) HEADER parameter 'Cookie #1*' is dynamic  
[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\x0\x0\x0\x0\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'
```

Other

Favicon Analysis (favfreak)

A more fringe technique of discovering a brands assets is taking their favicon and hashing it. You can then take this hash and search shodan for it. You can also scan IP ranges and cloud blocks to find assets with the same hash.

In addition you can make hashes of commonly used admin portals or framework logins. This method is useful when an org has modified URL paths and maybe your scanners are only looking for a specific path. If the org has changed it all you won't find it, but org's rarely change the favicon for the frameworks.
Check out [FavFreak](#) by Devansh Batham
([0xAsm0d3us](#))



```
shodan search org:"Target"  
http.favicon.hash:116323821 --fields  
ip str,port --separator " " | awk '{print  
$1":">$2}'
```

```
levi@Asm0d3us-Hackb0x:~/Desktop/proc$ cat urls.txt | python3 favfreak.py -o output
```

FFIVFREAH

- Coded with <3 by Devansh Batham

```
[ERR] Not Fetched 'http://accounts.  
[INFO] Fetched 'http://apiproxy-mov  
[ERR] Not Fetched 'http://abossmoni  
[ERR] Not Fetched 'https://apiproxy  
[ERR] Not Fetched 'https://android-  
[ERR] Not Fetched 'https://appcdn.f  
[ERR] Not Fetched 'https://accounts.  
[ERR] Not Fetched 'http://apiproxy.  
[INFO] Fetched 'https://credit-card.  
[INFO] Fetched 'https://apiproxy-mov
```

[FingerPrint Based Detection Results] -

```
[spring-boot] 116323821 - count : 2  
[big-IP] 878647854 - count : 2  
[slack-instance] 99395752 - count : 1
```

Port Analysis (masscan)

Most hacker education would have you use nmap here, but [masscan](#) by Robert Graham is much faster for general “finding-open-ports-on-TCP”. Chaining masscan’s output to then be nmap’ed can save a lot of time.

Masscan achieves this speed with a re-written TCP/IP stack, true multi-threading, and is written in C.

Sample syntax for scanning a list of IPs:

- masscan -p1-65535 -iL \$ipFile --max-rate 1800 -oG \$outPutFile.log

A full syntax guide of masscan (authored by Daniel Miessler) can be found here:
<https://danielmiessler.com/study/masscan/>

```
root@test2:~# host twitch.tv
twitch.tv has address 151.101.194.167
twitch.tv has address 151.101.2.167
twitch.tv has address 151.101.130.167
twitch.tv has address 151.101.66.167
twitch.tv mail is handled by 30 alt2.aspmx.l.google.com.
twitch.tv mail is handled by 50 aspmx3.googlemail.com.
twitch.tv mail is handled by 10 aspmx.l.google.com.
twitch.tv mail is handled by 40 aspmx2.googlemail.com.
twitch.tv mail is handled by 20 alt1.aspmx.l.google.com.
root@Test2:~# masscan -p1-65535 151.101.194.167 --max-rate 1800

Starting masscan 1.0.3 (http://bit.ly/14GZzcT) at 2019-01-18 06:34:05 GMT
-- forced options: -sS -Pn -n --randomize-hosts -v --send-eth
Initiating SYN Stealth Scan
Scanning 1 hosts [65535 ports/host]
Discovered open port 80/tcp on 151.101.194.167
Discovered open port 443/tcp on 151.101.194.167
```

Port Analysis (dnmasscan)

One limitation of masscan is that it only scans IP addresses. Y

you can write you own simple converter script or you can use something like [dnmasscan](#) by [@rastating](#)

```
root:~# cat dns.log
github.com
=====
140.82.118.4

rastating.github.io
=====
185.199.111.153
185.199.109.153
185.199.108.153
185.199.110.153

google.com
=====
172.217.169.14
```

```
root:~# dnmasscan example.txt dns.log -p80,443 -oG masscan.log
[*] Resolving domains...
[*] Saved resolved addresses to dns.log
[*] Launching masscan...
-----
Starting masscan 1.0.6 (http://bit.ly/14GZzcT) at 2019-09-17 22:56:48 GMT
-- forced options: -sS -Pn -n --randomize-hosts -v --send-eth
Initiating SYN Stealth Scan
Scanning 6 hosts [2 ports/host]
```



```
root:~# cat masscan.log
# Masscan 1.0.6 scan initiated Tue Sep 17 22:56:48 2019
# Ports scanned: TCP(1;80-80,) UDP(0;) SCTP(0;) PROTOCOLS(0;)
Timestamp: 1568761008 Host: 185.199.111.153 () Ports: 80/open/tcp//http/
Timestamp: 1568761008 Host: 185.199.109.153 () Ports: 443/open/tcp//https/
Timestamp: 1568761008 Host: 185.199.110.153 () Ports: 443/open/tcp//https/
Timestamp: 1568761008 Host: 185.199.108.153 () Ports: 443/open/tcp//https/
Timestamp: 1568761008 Host: 185.199.111.153 () Ports: 443/open/tcp//https/
Timestamp: 1568761008 Host: 172.217.169.14 () Ports: 443/open/tcp//https/
Timestamp: 1568761008 Host: 172.217.169.14 () Ports: 80/open/tcp//http/
Timestamp: 1568761008 Host: 185.199.108.153 () Ports: 80/open/tcp//http/
Timestamp: 1568761008 Host: 140.82.118.4 () Ports: 80/open/tcp//http/
Timestamp: 1568761008 Host: 140.82.118.4 () Ports: 443/open/tcp//https/
Timestamp: 1568761008 Host: 185.199.109.153 () Ports: 80/open/tcp//http/
Timestamp: 1568761008 Host: 185.199.110.153 () Ports: 80/open/tcp//http/
# Masscan done at Tue Sep 17 22:56:59 2019
```

Service Scanning (brutespray)

When we get this service/port information we can feed it to nmap to get a OG outputfile.

We can then scan the remote administration protocols for default passwords with a tool called [Burtespray](#) by [@x90skysn3k](#) which takes the nmap OG file format.

```
Loading File: \  
Welcome to interactive mode!  
  
WARNING: Leaving an option blank will leave it empty and refer to default  
  
Available services to brute-force:  
Service: ftp on port 21 with 9 hosts  
Service: smtp on port 25 with 8 hosts  
Service: smtp on port 587 with 1 hosts  
Service: ssh on port 22 with 8 hosts  
Service: telnet on port 23 with 1 hosts  
Service: mysql on port 3306 with 1 hosts  
  
Enter services you want to brute - default all (ssh,ftp,etc): ftp,ssh,telnet  
Enter the number of parallel threads (default is 2): 5  
Enter the number of parallel hosts to scan per service (default is 1): 10  
Would you like to specify a wordlist? (y/n): y  
Enter a userlist you would like to use:  
Enter a passlist you would like to use: /usr/share/wordlists/  
/usr/share/wordlists/dirb                   /usr/share/wordlists/fern-wifi                   /usr/share/wordlists/sqlmap.txt  
/usr/share/wordlists/dirbuster           /usr/share/wordlists/metasploit                   /usr/share/wordlists/wfuzz  
/usr/share/wordlists/dnsmap.txt           /usr/share/wordlists/nmap.lst  
/usr/share/wordlists/fasttrack.txt       /usr/share/wordlists/rockyou.txt.gz  
Enter a passlist you would like to use: /usr/share/wordlists/metasploit/password.lst  
Would you specify a single username or password (y/n): y  
Enter a username: admin  
  
Starting to brute, please make sure to use the right amount of threads(-t) and parallel hosts(-T)... \  
Brute-Forcing...  
Medusa v2.2 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks <jmk@foofus.net>
```



Github Dorking (manual)

Many organizations quickly grow in their engineering teams. Sooner or later a new developer, intern, contractor, or other staff will leak source code online, usually through a public Github.com repo that they mistakenly thought they had set private.

Enjoy my quick github dork collection:

<https://gist.github.com/jhaddix/1fb7ab2409ab579178d2a79959909b33>

** Helps if your console supports clickable hyperlinks

The repo mentioned earlier by [Gwendal Le Coguic](#) called “[github-search](#)” has some automated github tools for this as well.

Also check out [@th3q3ntelman’s](#) full module on [Github and Sensitive data Exposure](#).

```
root@Test2:~# bash Gdorkslinks.sh twitch.tv
***** Github Dork Links (must be logged in) *****
password
https://github.com/search?q=%22twitch.tv%22+password&type=Code
https://github.com/search?q=%22twitch%22+password&type=Code
npmrc _auth
https://github.com/search?q=%22twitch.tv%22+npmrc%20_auth&type=Code
https://github.com/search?q=%22twitch%22+npmrc%20_auth&type=Code
dockercfg
https://github.com/search?q=%22twitch.tv%22+dockercfg&type=Code
https://github.com/search?q=%22twitch%22+dockercfg&type=Code
.pem private
https://github.com/search?q=%22twitch.tv%22+pem%20private&type=Code
https://github.com/search?q=%22twitch%22+extension:pem%20private&type=Code
id_rsa
https://github.com/search?q=%22twitch.tv%22+id_rsa&type=Code
https://github.com/search?q=%22twitch%22+id_rsa&type=Code
aws_access_key_id
https://github.com/search?q=%22twitch.tv%22+aws_access_key_id&type=Code
https://github.com/search?q=%22twitch%22+aws_access_key_id&type=Code
s3cfg
https://github.com/search?q=%22twitch.tv%22+s3cfg&type=Code
https://github.com/search?q=%22twitch%22+s3cfg&type=Code
htpasswd
https://github.com/search?q=%22twitch.tv%22+htpasswd&type=Code
https://github.com/search?q=%22twitch%22+htpasswd&type=Code
git-credentials
https://github.com/search?q=%22twitch.tv%22+git-credentials&type=Code
https://github.com/search?q=%22twitch%22+git-credentials&type=Code
bashrc password
https://github.com/search?q=%22twitch.tv%22+bashrc%20password&type=Code
https://github.com/search?q=%22twitch%22+bashrc%20password&type=Code
sshd_config
https://github.com/search?q=%22twitch.tv%22+sshd_config&type=Code
```



GitHub Recon and
Sensitive Data Exposure

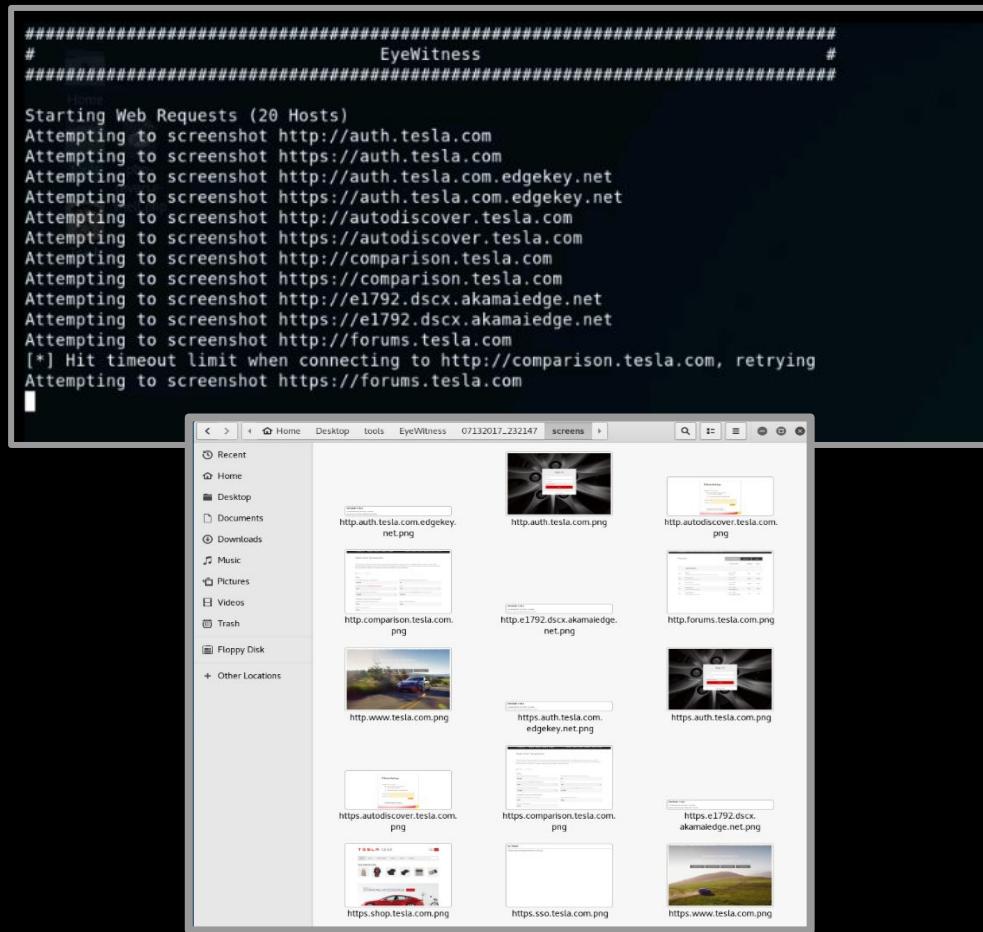


Screenshotting (Eyewitness, Aquatone, httpscreenshot)

At this point we have a lot of attack surface. We can feed possible domains to a tool and attempt to screenshot the results. This will allow us to “eye-ball” things that might be interesting.

There are many tools for this. [Aquatone](#) is a wider recon framework that does this, [HTTPScreenshot](#), and [Eyewitness](#). I use Eyewitness because it will prepend both the http and https protocol for each domain we have observed. I’m not highly tied to this tool though, find one that works for you.

Also, check out [WitnessMe](#)



Subdomain takeover (can i take over xyz)

“Subdomain takeover vulnerabilities occur when a subdomain (subdomain.example.com) is pointing to a service (e.g. GitHub pages, Heroku, etc.) that has been removed or deleted. This allows an attacker to set up a page on the service that was being used and point their page to that subdomain. For example, if subdomain.example.com was pointing to a GitHub page and the user decided to delete their GitHub page, an attacker can now create a GitHub page, add a CNAME file containing subdomain.example.com, and claim subdomain.example.com.”

A great resource for subdomain takeover is [Ed Overflow’s](#) repo [can-i-take-over-xyz](#)

The screenshot shows a GitHub repository page for 'EdOverflow / can-i-take-over-xyz'. The repository has 4 issues and 1 pull request. The main content discusses subdomain takeover, specifically mentioning AWS S3 buckets. It includes a section titled 'AWS S3' with a note that the answer is 'Yes' (indicated by a green checkmark). The text explains that if a domain has a CNAME record for *.s3.amazonaws.com and returns a NoSuchBucket error, it can be taken over by creating a bucket with that name and uploading a file.

EdOverflow / can-i-take-over-xyz

Code Issues 4 Pull requests 1 Projects 0 Wiki Insights

"Can I take over XYZ?" — a list of services and how to claim (sub)domains with dangling DNS records.

AWS S3

Answer: Yes ✓

If a domain has a CNAME record for *.s3.amazonaws.com and is returning NoSuchBucket, then all you need to do is to create a bucket with that name. You will need an AWS account, however, you can use the free tier which is more than enough for a PoC. You can then upload a simple txt file at a random path as a proof of concept.

Subdomain takeover (SubOver & nuclei)

To find subdomain takeovers we can use a few tools.

SubOver is a discontinued stand alone tool by [Ice3man](#) and has since been incorporated to [Project Discovery's nuclei scanner](#).

Nuclei is part of a larger scanning framework but boasts the most takeover checks of any tool i've seen.

```
ice3man@TheDaemon: ~/SubOver
ice3man@TheDaemon:~/SubOver (master)$ ./subover -l targets -v
SubOver v.1.1          Nizamul Rana (@Ice3man)
=====
[-] Trying dadadd.herokuapp.com with CNAME : us-east-1-a.route.herokuapp.com.
[-] Trying example.herokuapp.com with CNAME : example.herokuapp.com.
[#] Found Valid heroku Service At : dadadd.herokuapp.com
[heroku] Takeover Possible At : dadadd.herokuapp.com
[#] Found Valid heroku Service At : example.herokuapp.com

* [#] Done, Enjoy Your Hunt :-)
ice3man@TheDaemon:~/SubOver (master)$
```

 bauthard Merge pull request #137 from ca3s1m/master ...	Latest commit 1009f27 2 days ago
.github/workflows	Check all branches during syntax linting
basic-detections	Linting refactor to make yamllint happy
cves	Create CVE-2018-1271.yaml
dns	Update syntax
examples	Update syntax
files	added debug-pprof
panels	Sophos firewall detection
security-misconfiguration	Typo in gem name
subdomain-takeover	Statuspage removed
technologies	Add more servers
tokens	Linting refactor to make yamllint happy
vulnerabilities	Update rce-shellshock-user-agent.yaml
README.md	

Templates are the core of [nuclei scanner](#) which power the actual scanning engine. This repository stores and houses various templates for the scanner provided by our team as well as contributed by the community. We hope that you also contribute by sending templates via [pull requests](#) and grow the list.

Please [read this guide](#) to build your own custom 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] reflection value() found and filtering out
[05:06:30] [INFO] confirming 'Cookie #1*' is dynamic
[05:06:31] [INFO] (custom) HEADER parameter 'Cookie #1*' is dynamic
[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\x0\x0\x0\x0\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'
```

The logo consists of the word "Automation" in a bold, sans-serif font, followed by a large, stylized double plus sign ("++") where the top bar is slightly curved.

Extending tools (interlace)

Eventually you will want to make script or recon framework of your own. Quickly you will come up against some problems:

- Not all tools extend to take different sources of input
- Some lack threading.
- Not all can be distributed

You can rewrite a tool yourself to handle these issues but some help does exist here.

[Interlace](#) by Michael Skelton aka [Codingo](#) is an awesome tool than help glue together a recon framework.

Interlace can take these tools and add support for: CIDR input, Glob input, threading, proxying, queued commands, and more.

Hakluke wrote a [great guide on it here](#).

If I were scanning 1000 hosts, that would take too long to write out, so I might have used some bash scripting instead:

```
while read line; do
    nikto --host $line > ./${line}-nikto.txt;
done < targets.txt
```

Problem? It's still single threaded and too slow. Especially if, like me, you are suffering from ridiculously slow internet speeds in Australia. This is a great example of a situation where Interlace excels:

```
luke$ interlace -tL ./targets.txt -threads 5 -c "nikto --host
_target > ./_target_nikto.txt" -v
=====
Interlace v1.2 by Michael Skelton (@codingo)
=====
[13:06:16] [VERBOSE] [nikto --host yahoo.com > ./yahoo.com-nikto.txt]
Added after processing
[13:06:16] [VERBOSE] [nikto --host google.com > ./google.com-
nikto.txt] Added after processing
[13:06:16] [VERBOSE] [nikto --host hackerone.com > ./hackerone.com-
nikto.txt] Added after processing
[13:06:16] [VERBOSE] [nikto --host bugcrowd.com > ./bugcrowd.com-
nikto.txt] Added after processing
[13:06:16] [THREAD] [nikto --host google.com > ./google.com-
nikto.txt] Added to Queue
[13:06:16] [THREAD] [nikto --host hackerone.com > ./hackerone.com-
nikto.txt] Added to Queue
[13:06:16] [THREAD] [nikto --host bugcrowd.com > ./bugcrowd.com-
nikto.txt] Added to Queue
[13:06:16] [THREAD] [nikto --host yahoo.com > ./yahoo.com-nikto.txt]
Added to Queue
```

Extending tools (anything TomNomNom writes)

Tomnomnom has an [extensive repo](#) of tools which are awesome. I highly suggest you check them all out.

The screenshot shows Tom Hudson's GitHub profile page. At the top, there is a large white silhouette of a cat-like character with its arms raised, standing on a black background. Below the profile picture, a yellow button says "Taking it easy". The user's name is Tom Hudson and his GitHub handle is tomnomnom. There are two buttons: "Sponsor" with a heart icon and "Follow". A footer note says "Open source tool maker, trainer, talker".

Overview Repositories 95 Projects 0 Stars 220 Followers 3.2k Following 121

Popular repositories

- gron**
Make JSON greppable!
Go 6.5k ⭐ 138
- httpprobe**
Take a list of domains and probe for working HTTP and HTTPS servers
Go 826 ⭐ 177
- assetfinder**
Find domains and subdomains related to a given domain
Go 702 ⭐ 135
- waybackurls**
Fetch all the URLs that the Wayback Machine knows about for a domain
Go 686 ⭐ 113
- hacks**
A collection of hacks and one-off scripts
Go 650 ⭐ 203
- meg**
Fetch many paths for many hosts - without killing the hosts
Go 614 ⭐ 126

Frameworks

It could be recon is not really your thing. That's all right.

Several hunters have open sourced their automation at this point and you can choose one that fits you and use it without worrying too much. I usually classify recon frameworks in rough tiers:

C-Tier: automation built around scripting up other tools in bash or python. Step based, no workflow. Few techniques. Little extensibility.

B-Tier: automation writing a few of their own modules. Some GUI or advanced workflow. Medium number of techniques. Runs point-in-time. Flat files.

A-Tier: (maybe) automation writing all their own modules. Has GUI. Runs iteratively. Manages data via db.

S-Tier: automation writing their own modules. Has GUI. Runs iteratively. Manages data via db. Scales across multiple boxes. Sends alerts to user. Uses novel techniques and iterates quickly. ML + AI.

Frameworks

Warning: My scale is mostly subjective and can be missing factors. It is also based off of my rough experience and gut feel.

Frameworks (C-Tier)

<https://github.com/AdmiralGaust/bountyRecon>

<https://github.com/offhourscoding/recon>

<https://github.com/Sambal0x/Recon-tools>

<https://github.com/JoshuaMart/AutoRecon>

<https://github.com/yourbuddy25/Hunter>

https://github.com/venom26/recon/blob/master/ultimate_recon.sh

<https://gist.github.com/dwisiswant0/5f647e3d406b5e984e6d69d353896cd>

135 lines (113 sloc) | 4.48 KB

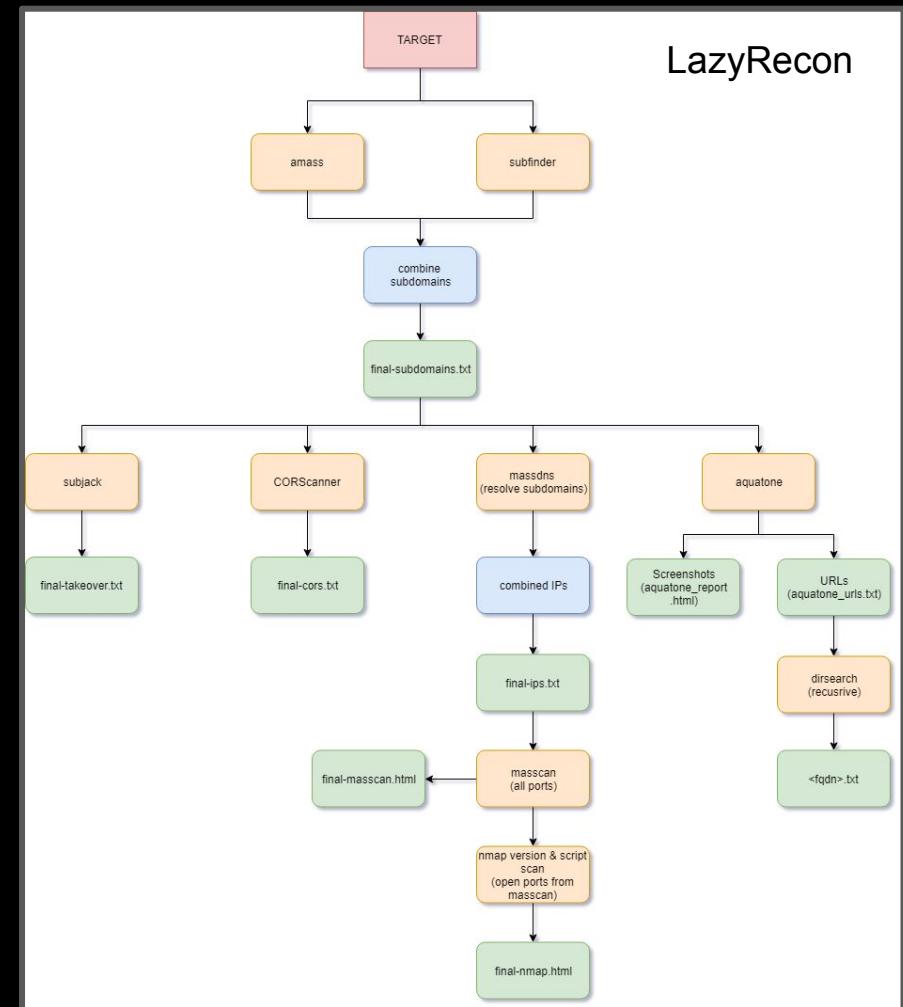
Raw Blame History

ultimate_recon.sh

```
1 #!/bin/bash
2
3 while getopts ":d:" input;do
4     case "$input" in
5         d) domain=${OPTARG}
6             ;;
7         esac
8     done
9 if [ -z "$domain" ]
10 then
11     echo "Please give a domain like \"-d domain.com\""
12     exit 1
13 fi
14
15 sublist3r -d $domain -v -o op.txt
16 subfinder -d $domain -o op.txt
17 assetfinder --subs-only $domain | tee -a op.txt
18 amass enum -passive -d $domain | tee -a op.txt
19 amass enum -active -d $domain -ip | tee -a amass_ips.txt
20 cat amass_ips.txt | awk '{print $1}' | tee -a op.txt
21 cat op.txt | sort -u | tee -a all.txt
22 echo -e "#####Starting Bruteforce#####\n"
23 aitdns -i all.txt -o data_output -w ~/tools/recon/patterns.txt -r -s results_output.txt
24 mv results_output.txt dns_op.txt
25 cat dns_op.txt output.txt
26
27 cat output.txt | sort -u | tee -a all.txt
28 echo "Checking for alive subdomains"
29 cat all.txt | httpprobe | tee -a alive2.txt
30 cat alive2.txt | sort -u | tee -a alive.txt
31
32 ~/tools/massdns/bin/massdns -r ~/tools/massdns/lists/resolvers.txt -q -t A -o S -w massdns.raw all.txt
33 cat massdns.raw | grep -e 'A' | cut -d 'A' -f 2 | tr -d ' ' > massdns.txt
34 cat *.txt | sort -V | uniq > $IP_PATH/final-ips.txt
35 echo -e "${BLUE}[*] Check the list of IP addresses at $IP_PATH/final-ips.txt${RESET}"
36
37 echo "Starting Nuclei"
38 mkdir nuclei_op
39 nuclei -l alive.txt -t "/root/tools/nuclei-templates/cves/*.yaml" -c 60 -o nuclei_op/cves.txt
40 nuclei -l alive.txt -t "/root/tools/nuclei-templates/files/*.yaml" -c 60 -o nuclei_op/files.txt
41 nuclei -l alive.txt -t "/root/tools/nuclei-templates/panels/*.yaml" -c 60 -o nuclei_op/panels.txt
42 nuclei -l alive.txt -t "/root/tools/nuclei-templates/security-misconfiguration/*.yaml" -c 60 -o nuclei_op/security-misconfiguration.txt
43 nuclei -l alive.txt -t "/root/tools/nuclei-templates/technologies/*.yaml" -c 60 -o nuclei_op/technologies.txt
44 nuclei -l alive.txt -t "/root/tools/nuclei-templates/tokens/*.yaml" -c 60 -o nuclei_op/tokens.txt
45 nuclei -l alive.txt -t "/root/tools/nuclei-templates/vulnerabilities/*.yaml" -c 60 -o nuclei_op/vulnerabilities.txt
```

Frameworks (B-Tier)

https://github.com/capt-meelo/LazyRecon	https://github.com/Sreetsec/Sudomy
https://github.com/phspade/Automated-Scanner	https://github.com/devanshbatham/Gorecon
https://github.com/shmilyly/OneForAll	https://github.com/LordNeoStark/tugarecon
https://github.com/SolomonSklash/chomp-scan	https://github.com/s0md3v/photon
https://github.com/TypeError/domained	

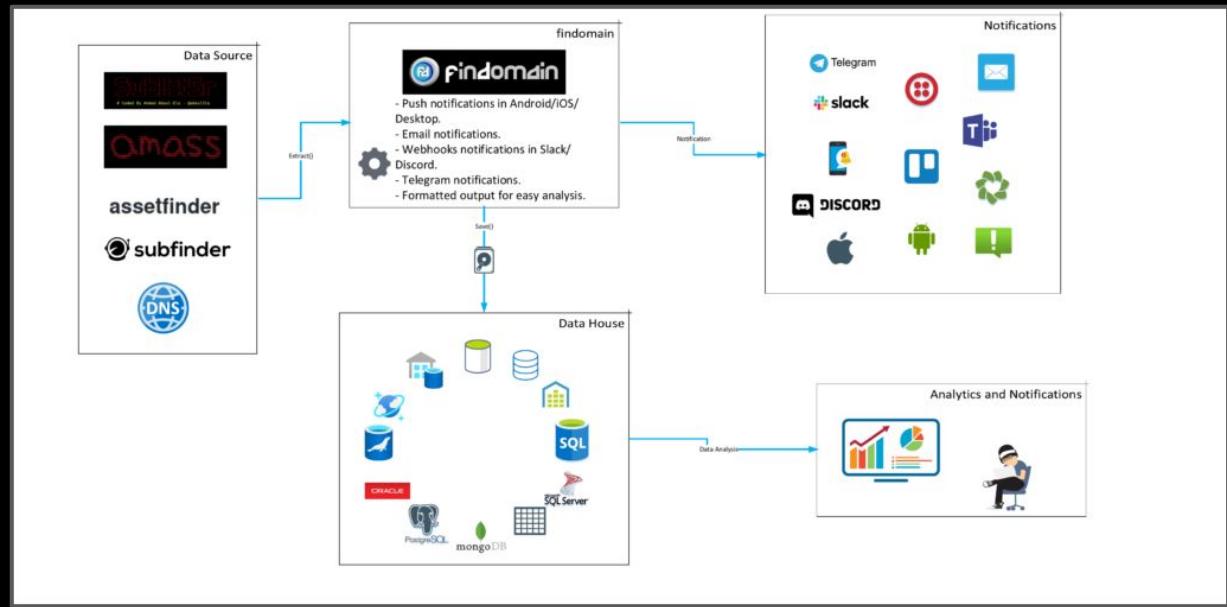


Frameworks (A-Tier)

<https://github.com/Edu4rdSHL/findomain>

<https://github.com/SilverPoision/Rock-ON>

<https://github.com/epi052/recon-pipeline>



Findomain+ alert: 7 new subdomains found for [REDACTED]



monitoring@findomain.app

to [REDACTED]

6:36 AM (13 hours ago)

HOST: utildl1r3vdna.psl.labs, IP: 12.1 [REDACTED], HTTPS: INACTIVE, OPEN PORTS: NULL
HOST: apis.psl.labs, IP: 2.1 [REDACTED], HTTPS: <https://apis.psl.labs>, OPEN PORTS: [80, 443, 3306, 5432]
HOST: sponsoreddata.psl.labs, IP: 2 [REDACTED], HTTPS: <https://sponsoreddata.psl.labs>, OPEN PORTS: [21, 22, 53, 80, 443]
HOST: utildl4r3vdna.psl.labs, IP: 12.1 [REDACTED], HTTPS: INACTIVE, OPEN PORTS: NULL
HOST: ort-developer.psl.labs, IP: 23 [REDACTED], HTTPS: <https://ort-developer.psl.labs>, OPEN PORTS: [80, 443]
HOST: utildl3r3vdna.psl.labs, IP: 12.1 [REDACTED], HTTPS: INACTIVE, OPEN PORTS: NULL
HOST: utildl2r3vdna.psl.labs, IP: 12.1 [REDACTED], HTTPS: INACTIVE, OPEN PORTS: NULL

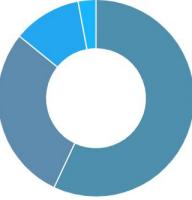
Frameworks (S-Tier)

Intrigue.io

Intrigue Dashboard

Assets Overview

Hosts Applications Domains Certificates Networks



Asset	Endpoints
Hosts	4697
Applications	2376
Domains	926
Certificates	241
Networks	30
Network Services	1214

Individual Assets

Asset	Endpoints
dev.acme.co.uk	ACME EMEA
acme.com.au	ACME APAC
01-us-east.acme.com	ACME US
abc.acme.com	ACME US

Overview Recent Changes Domains (926) Applications (2376) Networks (30) Hosts (4697) Certificates (241) People (2) Services (1214) Products

AssetNote

ASSETNOTE

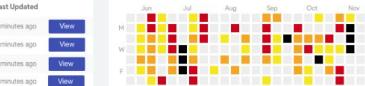
Dashboard

Assets discovered: 5384 Assets discovered today: 83 Vulnerabilities discovered: 644 Vulnerabilities discovered today: 13

Assets That Need Attention

Asset	Group	Exposure Rating	Last Updated	Action
dev.acme.co.uk	ACME EMEA	Critical	5 minutes ago	View
acme.com.au	ACME APAC	Critical	5 minutes ago	View
01-us-east.acme.com	ACME US	High	5 minutes ago	View
abc.acme.com	ACME US	Medium	5 minutes ago	View

Exposure Timeline



Vulnerability and Triage Analysis

Vulnerability Criticality Overview



Breakdown of Vulnerabilities

Type	Count
Input Validation	45
Authentication	35
Ambient/patch Management	30
Logic Flaws	40

Triage Analysis

Category	Count
Unresolved Vulnerabilities	184
Traged Vulnerabilities	363
Ignored Vulnerabilities	77
Average Time To Triage	4 days
Regression Rate	4% of Traged Vulnerabilities

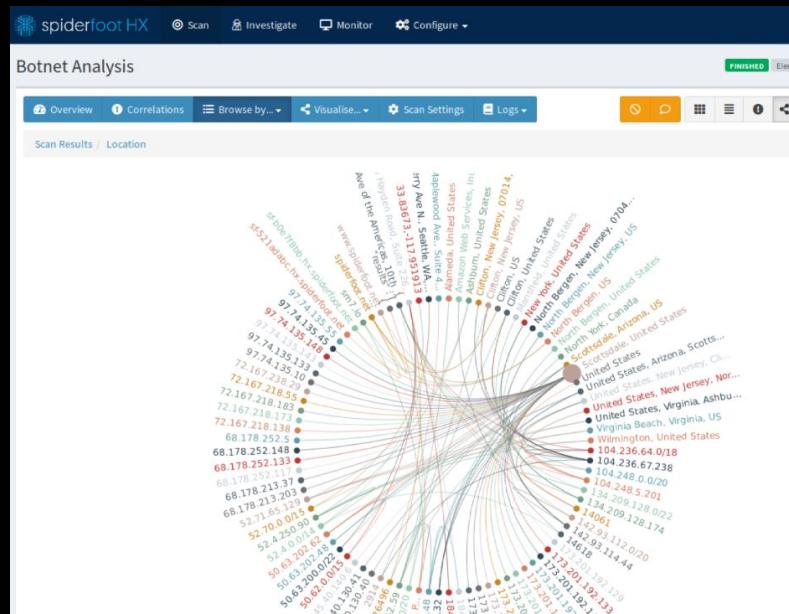
Asset Discovery Feed

Asset	Discovered
01-us-east.staging.acme.com	3 minutes ago
02-us-east.staging.acme.com	3 minutes ago
01-us-east.staging.acme.com	3 minutes ago
acmepromotion.net	5 minutes ago
100bed13C1JU.acme.net	5 minutes ago
dev-0556.acmecloud.com	6 minutes ago
ab.acme.com.au	7 minutes ago

Vulnerability Discovery Feed

Asset	Vulnerability
01-us-east.staging.acme.com	Reflected Cross-Site Scripting
01-us-east.staging.acme.com	Server-Side Request Forgery
02-us-east.staging.acme.com	Reflected Cross-Site Scripting
02-us-east.staging.acme.com	Server-Side Request Forgery
dev.acme.co.uk	Remote Code Execution
dev.acme.co.uk	Information Disclosure
dev.acme.co.uk	Information Disclosure

Frameworks (S-Tier)



Project Discovery Framework (unreleased)



Frameworks (S-Tier)

Osmedeus

Jaeles (scanner)

Show 100 s | entries

Search: curl

RISK	URL	SIGNATURE NAME	DETAILS
critical	https://filter=%2720%etc%pa	uel/pages/select?28%70%72%69%6e%74%28%24%61%3d%27%73%79%73%74%65%6d%27%29%2b%24%61%28%27%cat%	fuelcms-rce-ac4d83246f65b3554126c1239a09293800e864
critical	http://	:8060/cachestart/125116/cacheend/api/client/fluidcv2/javascript/jquery.../.../.../conf/server.xml	zoho-manage-lfi-dd6924382e4a85de0a8faf65a2a17f2ca04dd25
critical	http://201nf	:50/cachestart/125116/cacheend/api/client/fluidcv2/javascript/jquery.../.../.../conf/OpManager/database_params.co	zoho-manage-lfi-bf9839099231cafc34e5463f03dd3beda5bf9f1
critical	http://	:8060/cachestart/125116/cacheend/api/client/fluidcv2/javascript/jquery.../.../.../bin/ssh_host_rsa_key	zoho-manage-lfi-d043c583b13c3fe6875334954be04fa2ac0fb2cb

Showing 1 to 4 of 4 entries (filtered from 5 total entries)

Generated by [Jaeles beta v0.9](#) at 2020-05-16,11:15:14

Jaeles is made with ❤ by [@3ssiejj](#) and it is released under the MIT license.

Become a Backer

written by [j3ssiejj](#) and team



Workspaces Summary

Open	Domain	IP	Technology	Ports
54.			http/nginx	443
54.			ssh/OpenSSH http/Node.js Express framework	22 4801 4901
54.			amqp/RabbitMQ http/Golang net/http server http/nginx esmagent/ ssh/OpenSSH	5672 22 3000 8089 5601 80
54.			ssh/OpenSSH /rpcbind/	22 111
54.			mts-wbt-server/Microsoft Terminal Services http/Apache httpd	80 3389

[HunterSuite.io](#) (unreleased)

Dashboard

No	Value	Technology	Services	Actions
32847	api-testbox.redacted.com	unicorn/19.4.5 Python	N/A	Scan
32850	auth.redacted.com	Nginx NivCMS Bulma	N/A	Scan
32853	autodiscover.redacted.com	Microsoft ASP.NET IIS/10.0 Windows Server	N/A	Scan
32857	cicerone.redacted.com	Exarous Node.js	443 http/Node.js Express framework	Scan
32875	epc.redacted.com	NvCMS Microsoft ASP.NET IIS/8.5 Windows Server	443 https 80 http-proxy/5 DIG-IP load balancer http proxy	Scan
32876	epcapi.redacted.com	Kestrel Microsoft ASP.NET	N/A	Scan
32883	forums.redacted.com	jQuery NivCMS Drupal/7 PHP Akamai Google Tag Manager Underscore.js Google Font API Modernizr	N/A	Scan

DISCOVERY

- Targets
- Assets
- DNS
- Links
- Screen Shots

MONITOR

- Domain
- EndPoint
- Files

VULNERABILITY SCAN

- Wordlists

HunterSuite © 2020 HunterSuite.io

Powered by HunterSuite Engine

Frameworks (S-Tier)

Bountyoffensiveai.com (paid) by @ghostlulz1337

The Bug Bounty Toolkit

Dashboard Screenshots Assets

Airbnb Assets

Search

None Fingerprint

Host	Port	Fingerprints	Service	Attack Tr.
airbnb.com	80	Envoy React 0.muscache.com Varnish 1.1 Nginx Ruby on Rails Ruby nginx	HTTP	<button>Generate</button>
airbnb.com	443	React 0.muscache.com Nginx Envoy Ruby Ruby on Rails Varnish 1.1 test_fingerprint nginx	HTTP	<button>Generate</button>
api-fastly-staging.airbnb.com	80	React 0.muscache.com Ruby on Rails Nginx Ruby Envoy Varnish 1.1 nginx	HTTP	<button>Generate</button>
api-fastly-staging.airbnb.com	443	React 0.muscache.com Ruby on Rails Nginx Ruby Envoy Varnish 1.1 nginx	HTTP	<button>Generate</button>

reNginx by yogeshojha

reNginx

Dashboard

Targets

- Add Target
- List Targets

Scan History

Scan Engine

Notification

3 Total Targets

20,313 Subdomains Discovered

Total Alive Subdomains: 4,065

47,368 Endpoints Discovered

Total Alive endpoints: 6,919

7 Total Scans

Currently Scanning 2

Recent Scans

facebook.com	12 hours ago	Scanning
google.com	12 hours ago	Scanning
hackerone.com	20 hours ago	Successful
yogeshojha.com	20 hours ago	Successful

Frameworks (S-Tier)

Scout (paid) by the secapps.com team

Scout

Search

default last created last updated last executed all scheduled unscheduled displaying 377/377 scouts

nintendo.com d14fa79e-1129-11ea-8d7c-a3e20ce6b851 domains: nintendo.com, nintendo.net, nintendo.de, nintendo.com.br, nintendo.com.eu schedule: never bounty hackerone nintendo	tomtom.com d14fa799-1129-11ea-8d7c-a3e20ce6b851 domains: tomtomgroup.com, teleatlas.com, tomtom.com schedule: never bounty hackerone	roblox.com 854515a0-6849-11ea-87fa-2b666c3b5da8 domains: roblox.com, rbx.com schedule: never bounty hackerone
creditkarma.com d14f80a1-1129-11ea-8d7c-a3e20ce6b851 domains: creditkarma.com, creditkarma.ca schedule: never bounty bugcrowd	usaa.com d14f598e-1129-11ea-8d7c-a3e20ce6b851 domains: usaa.com schedule: never bounty bugcrowd	fanduel.com 4658b550-6845-11ea-8598-1780d6f9f8a9 domains: fanduel.com schedule: never bounty hackerone
airtable.com d14f5966-1129-11ea-8d7c-a3e20ce6b851 domains: airtable.com schedule: never bounty hackerone	kayak.com 516c5940-427a-11ea-93d8-ad5d0e3bd7d6 domains: kayak.com schedule: never	linkedin.com d14f596d-1129-11ea-8d7c-a3e20ce6b851 domains: linkedin.com schedule: never bounty hackerone
paydiant.com d14f5969-1129-11ea-8d7c-a3e20ce6b851 domains: paydiant.com schedule: never bounty hackerone paypal	tesla.com d14fa794-1129-11ea-8d7c-a3e20ce6b851 domains: tesla.com, tesla.cn, teslamotors.com, teslamotors.services schedule: never bounty bugcrowd	getmoneytree.com d14f5977-1129-11ea-8d7c-a3e20ce6b851 domains: getmoneytree.com, moneytree.com schedule: never bounty bugcrowd
asana.com d14fcea3-1129-11ea-8d7c-a3e20ce6b851 domains: asana.com	uipath.com d14f80ca-1129-11ea-8d7c-a3e20ce6b851 domains: uipath.com	verizonwireless.com d14f5993-1129-11ea-8d7c-a3e20ce6b851 domains: verizonwireless.com

Scout

Summary Screenshots Data

Scout

Scout

Impact Solution References Variants (1)

Autocomplete Enabled

Autocomplete was not turned off.

Autocomplete is a HTML tag attribute used to disable the form auto completion mechanism of the browser.

Impact Solution References Variants (10)

Error Disclosure

The application/server disclosed error information within its source code or other files.

Impact Solution References Variants (18)

IP Disclosure

The server or application disclosed internal network information.

Impact

Honorable Mention

Nuclei

1. CHOOSE TEMPLATES

```
root@b0x:~/templates/files# ls
adminer-sql.yml      codeception.yaml  go-expvar.yml      jkstatus.yml  put-rce.yml    server-status-localhost.yml  tomcat-status.yml
akamai-arls.yml      dot-env.yaml     go-prometheus.yml lfi.yml       rails-config.yml   spring-env.yml
application-wadl.yml git-core.yaml   jboss-serverinfo.yml phpinfo.yml  s3-xss.yml    ssrf-files.yml
```

2. RUN AGAINST LIST OF HOSTS

```
root@b0x:~/templates/files# ./nuclei -t dot-env.yaml -l ~/targets.txt
```



projectdiscovery.io

[WRN] Use with caution. You are responsible for your actions

[WRN] Developers assume no liability and are not responsible for any misuse or damage.

[INF] [dot-env] Loaded template Dot Env file (@Ice3man) [high]

[dot-env] http://vulnerable-site.io/.env

[dot-env] http://random-affected-site.net/.env

[dot-env] http://is-this-affected-too.com/.env

[dot-env] http://some-vulnerable-site.com/.env

[dot-env] http://so-many-hosts.com/.env

[dot-env] http://another-vulnerable-site.com/.env

The End