A More Advanced Recon Automation #1 (Subdomains)

January 18, 2019 003random Bugbounty, Pentesting, recon-serie, Tools

So you want to step up your recon game huh?
Then you are at the right place.

Recon automation can be really use full and if done right, it can save you lots of time. For example think about automating your directory-search tools which will pass its results to your CORS scans etc.

This can also land you some nice vulnerabilities for which you have done nothing, but sit back and relax.

Some fancy charts

Lets first analyze things for a bit. We want to automate what can be automated. The flowchart below describes a possible way of chaining tools in 3 phases. (the link is below the image to enlarge it)

Social

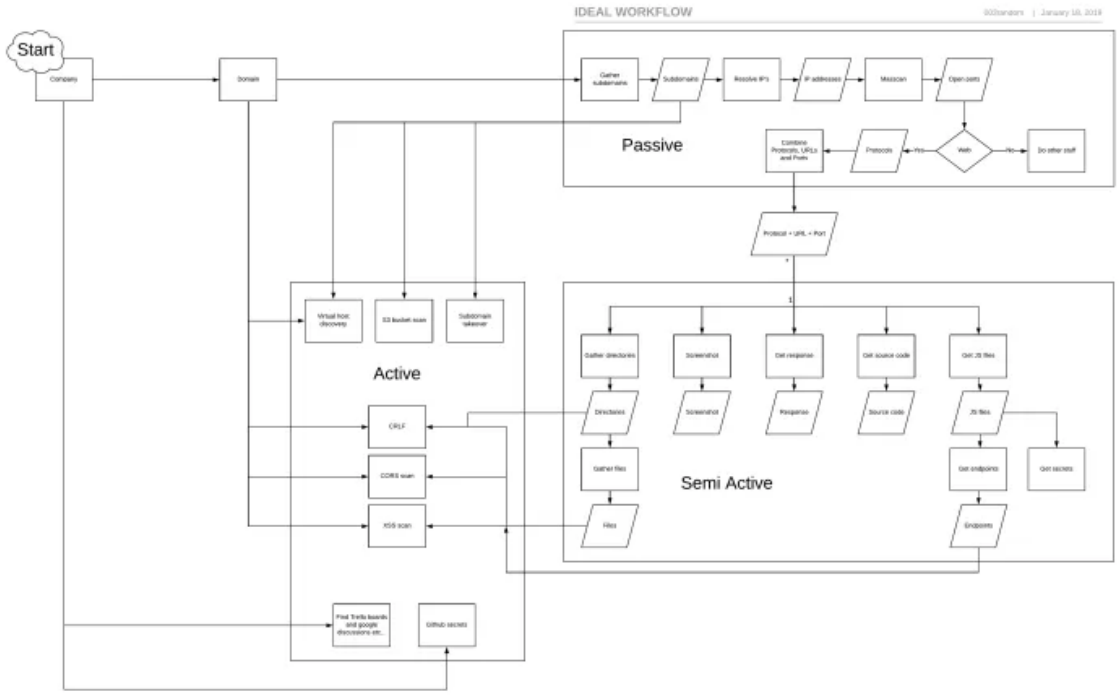
hackerone

github

twitter

Recent Posts

- Expanding your scope (Recon automation #2)
- A More Advanced Recon Automation #1 (Subdomains)
- H1-702 CTF ~ Write-Up
- ICU – Keep An Updated Database Of Your Assets



Workflow.png

We skip the company part for now and move straight to the subdomain gathering.

Subdomain gathering

Tools involved

To get the most results, but still a decent speed, we use these tools:

- Subfinder
- Altdns
- Massdns

We will also use a custom subdomains wordlist and a couple of other small scripts which will be described below.

Share

If you liked this post. Make sure to share it so more people can read it! 🙂

First we bruteforce subdomains with massdns using one or more custom subdomain wordlists.
Then we run subfinder to get subdomains from various sources.
When thats done we run altdns to see if we can get some altered versions of the subdomains.
And finally we add recursion to get subdomains which are more levels deep.

Brute-forcing with massdns

Lets start off with creating our main script for the subdomain gathering.

```
#!/usr/bin/env bash

while getopts ":d:" opt; do
  case $opt in
    d)
      domain=$OPTARG
      ;;
  esac
done

if [[ -z "${domain// }" ]];
then
    echo "[-] No domain supplied. Use -d domain.com"
    echo "[!] Exiting..."
    exit 1
fi
```


To start we use massdns to resolve a list of subdomains we generate with

```
while read -r line
do
```

```
    echo "$line.$2" >> $3
done < $1
```

append_subdomains.sh hosted with ♥ by GitHub view raw

This basically takes in 3 arguments:

1. The subdomains wordlist.
2. The domain for which we generate the subdomains.
3. The output file to save to.

Now we need some subdomain wordlists. There are multiple options you can choose between:

- all.txt, from Jhaddix
- Commonspeaks's subdomains.txt wordlist from Shubs
- Stackoverflow.txt also from Shubs

After you have chosen one or more wordlists, you can create a line in your main script as following:

```
~/scripts/append_subdomains.sh ~/wordlists/commonspeak2-subdomains.txt $domain "wordlist.txt
```

append script hosted with ♥ by GitHub view raw

Next we want to resolve those entries with massdns.

You can do that like this, to save the online subdomains in wordlist-online.txt:

```
massdns -r ~/wordlists/resolvers.txt -q -t A -o S -w "wordlist-online.txt" "wordlist.txt"
```

resolve subdomains with massdns hosted with ♥ by GitHub view raw

Since masscan produces a slightly different output, we want to filter the subdomain from the output with:

```
 awk -F ". " '{print $1}' "wordlist-online.txt" > "wordlist-filtered.txt" && mv "wordlist-fi
```

filter subdomains from masscan output hosted with ❤ by **GitHub** **view raw**

Subfinder

Now that we got the bruteforced subdomains, we go ahead and run some tools to gather subdomains from other sources. A perfect tool for this is subfinder from my friends Ice3man543, Codingo and Mzack9999.

We run subfinder like this:

```
subfinder -d $domain -nW -o "subfinder-online.txt" -rL ~/wordlists/resolvers.txt > /dev/null
```

subfinder command hosted with ❤ by **GitHub** **view raw**

Where resolvers.txt is your text file with resolvers. You can remove that argument as well if you want.

Combining

Now that we have subdomains from various tools, we want to combine those.

Just 'cat' the files into one big file called subdomains.txt

```
cat wordlist-online.txt subfinder-online.txt > subdomains.txt
```

combine output hosted with ❤ by **GitHub** **view raw**

And make the file uniquet

```
sort -u "subdomains.txt" -o "subdomains.txt"
```

sort and uniq subdomains file hosted with ❤ by **GitHub** **view raw**

Altnds

Next on the list is altdns.

Altdns alters the subdomains with a list of given words. For example it can find staging-dev.poc-server.com if you give it dev.poc-server.com.

Using this technique we can discover subdomains others wouldn't have found.
You can run altdns like this (note that we use words.txt from altdns):

```
1  python ~/tools/altdns/altdns.py -i "subdomains.txt"  -o "altdns-wordlist.txt" -w ~/tools/alt
```

altdns command hosted with ♥ by GitHub view raw

Recursion

After you have this list of all existing subdomains. You can start recursion on them to find subdomains that are 1 or 2 levels deep. For example test.staging.dev.poc-server.com.

You can do this by repeating the bruteforcing step, but this time you dont give it poc-server.com, but test.poc-server.com.

Things to remember

There are somethings that are not in this post, and I hope you will find them on your own as well, so you can learn from them.

A couple of examples are:

- There might be a wildcard for subdomains. This means that you would get a huge list of false positives. Below this list I have inserted a code snippet to detect wildcards.
- Giving your own resolvers list to your tools might increase the speed.
- Clean your -online.txt files when you have all output in one big file.
- Give the script some extra parameters like -verbose to toggle the verbose version of masscan, to keep track of the resolved items and how long it will still take.
- FDNS

Wildcard code snippet:

```
1  if [[ "$(dig @1.1.1.1 A,CNAME {test321123,testingforwildcard,plsdontgimmearesult}.$domain +s
2      echo "[!] Possible wildcard detected."
```

```
3  fi
```

subdomains wildcard detection example hosted with ❤ by GitHub view raw

The serie

And that's already it folks.

If you got any questions, try to solve them on your own, and if you still cant answer them, you can shoot me a message.

in #2 we are continuing with the next part of the flowchart.

Thanks for reading!

Credits

SO to Quikke and Europa

Share this:

Related

Expanding your scope (Recon automation #2)
January 31, 2019
In "Bugbounty"

ICU - Keep An Updated Database Of Your Assets
June 15, 2018
In "Tools"

RCE by uploading a web.config
May 22, 2018
In "Bugbounty"

bugbounty recon recon-series tools

Previous Post

Next Post

Leave a Reply

You must be logged in to post a comment.