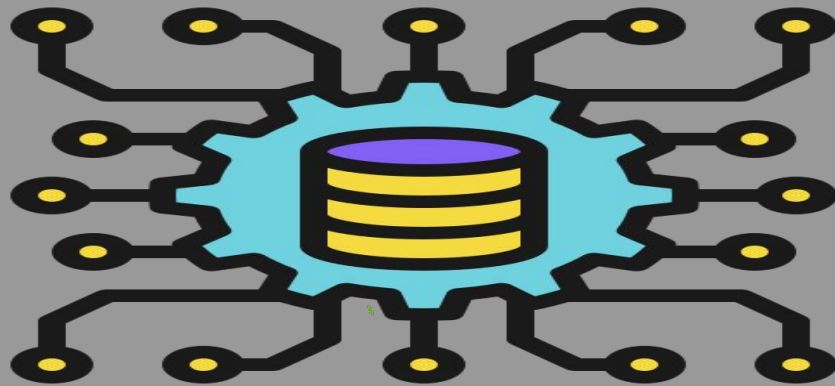




My Bug Bounty Hunting

Reconnaissance v1.0



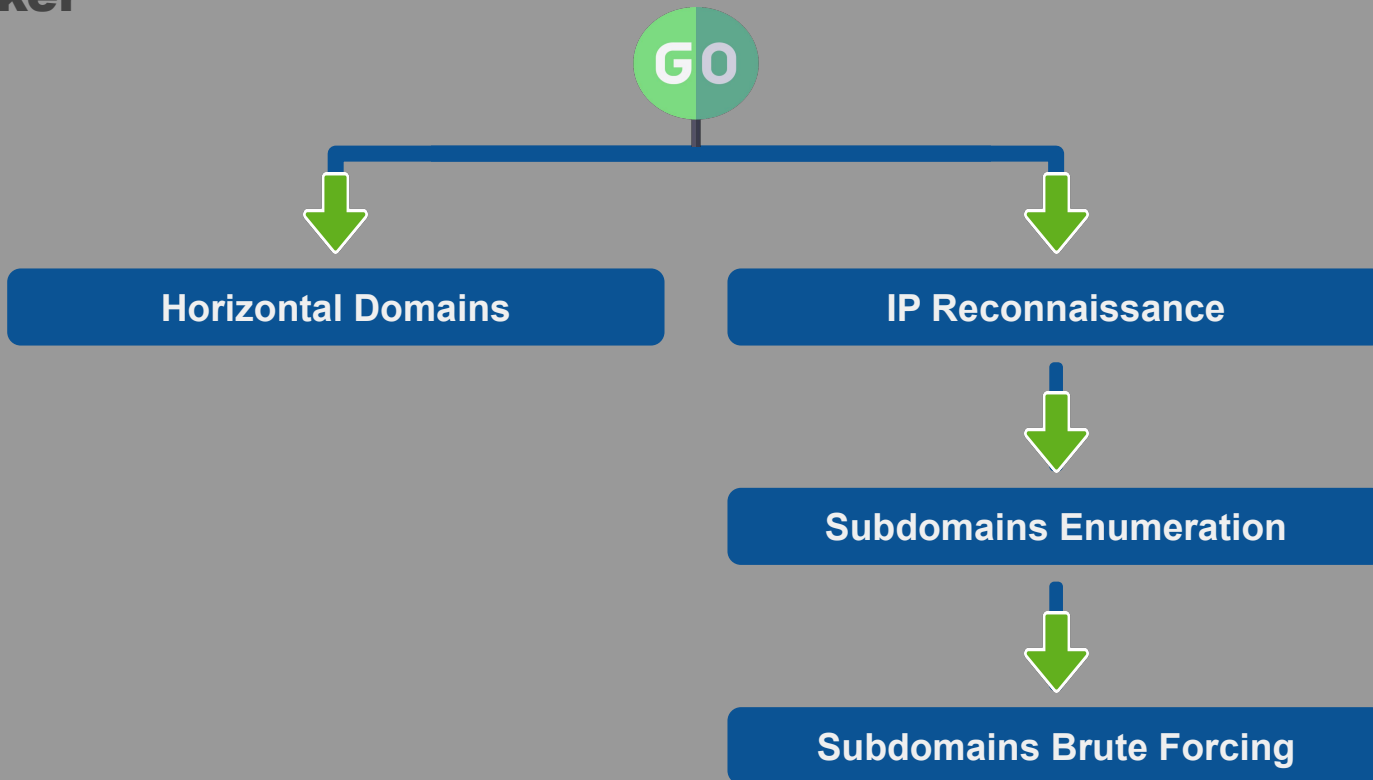
Mahmoud M. Awali

 **@0xAwali**



attacker

Reconnaissance Workflow





attacker

IP Reconnaissance

CIDRs Enumeration



IPs Alive



Full AND Top PORTs Scanning



masscan -P443



SAN From Certificates



Reverse DNS Lookup



Third Level Domains





attacker

My Methodology

Collect Information About Your Target By Using **whois** Command Line e.g.

```
root@mine:~#whois company.com
```

- Registrant Organization
- Registrant Email


Expand the scope by one click with bgp.he.net

The first step of any bug bounty or pentest is to get the [scope](#).














You can do it literally by one click with bgp.he.net.

Look for [company's IP subnets](#) using company name, IP, IPv6, domains.

Get a list of [IP/IPv6 subnets](#) that are ready for [nmap/masscan](#) scanning.



The screenshot shows the Hurricane Electric Internet Services BGP Toolkit interface. The search bar contains 'twitter' and the search button is clicked. The search results are displayed in a table with two columns: 'Result' and 'Description'. The results list various IP ranges and their corresponding descriptions, including 'Twitter International Company' and 'Twitter Inc.'.

Result	Description
twitter	
AS8945	Twitter International Company 
AS54888	Twitter Inc. 
AS35995	Twitter Inc. 
AS13414	Twitter Inc. 
8.25.196.0/24	TWITTER 
8.25.196.0/23	TWITTER 
8.25.195.0/24	TWITTER 
8.25.194.0/23	TWITTER 
69.195.191.0/24	Twitter Inc. 
69.195.190.0/24	Twitter Inc. 
69.195.189.0/24	Twitter Inc. 
69.195.188.0/24	Twitter Inc. 
69.195.187.0/24	Twitter Inc. 
69.195.186.0/24	Twitter Inc. 



attacker

My Methodology

Use Tools e.g. Asnlookup OR Service host.io To Get List Of CIDR

```
root@mine:~#python3 asnlookup.py -o Organization
```

```
" -o Organization " Name Of Organization To Enumerate
```



attacker

My Methodology

Use Tools e.g. **dnsx** To Get Subdomains From Reverse DNS Lookups

```
root@mine:~#cat cidrs.txt | ./mapcidr -silent -o out.txt  
root@mine:~#cat out.txt | sort -u | tee -a ips.txt  
root@mine:~#cat ips.txt | dnsx -r resolvers.txt -ptr -silent -resp-only | tee -a subdomains.txt
```

" -r resovers.txt " Input File Of IPs DNS Resolver

" -ptr " Query PTR Record

" -silent " Silent Mode To Show Only Results

" -resp-only " Display Only Response Data



attacker

My Methodology

Use Tools e.g. **hakrevdns** To Get Subdomains From Reverse DNS Lookups

```
root@mine:~#cat reverseDNS.sh
#!/usr/bin/env bash
for i in `cat cidrs.txt`
do
    prips $i | ./hakrevdns -d -r 1.1.1.1 | tee -a subdomains.txt
done
root@mine:~#./reverseDNS.sh
```

" prips I.P.v.4/cidr " Print All Of The IP Addresses

" -r 1.1.1.1 " IP Of The DNS Resolver



attacker

My Methodology

Use Tools **e.g. masscan** To Scan What IPs Have HTTPS Certificate

```
root@mine:~#masscan -iL ips.txt --source-port 53 --http-user-agent "Mozilla/5.0 (X11; Linux  
x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/77" -p 443 -oL httpservice443alive.txt
```

" -iL input.txt " Reading Input File

" --source-port 53 " Use A Custom Source Port

" --http-user-agent "Mozilla" " Custom User Agent

" -oL httpservice443alive.txt " Output File



attacker

My Methodology

Use Tools **e.g. SANextract** To Get Subject Alternative Names

```
root@mine:~#cat httpservice443alive.txt | ./SANextract -timeout 30s | tee -a subdomains.txt
```



attacker

Extract Third Level Domains From List Of Subdomains



Mine

```
#!/usr/bin/env python3
import os
import sys
import argparse
parser = argparse.ArgumentParser()
parser.add_argument( "-f","--file",help="file that contains list of subdomains" )
parser.parse_args()
args = parser.parse_args()
if args.file:
    if os.path.isfile(args.file):
        list_of_subdomains = open( args.file, 'r' )
        file_of_subdomains = list_of_subdomains.read().split("\n")
        list_of_subdomains.close()
    else:
        parser.error( '%s file not found' % args.file )
for subdomain in file_of_subdomains :
    try :
        if subdomain.count(".") == 2 :
            print(subdomain)
        else :
            third_level_domain = subdomain.split(".")[3] + '.' + subdomain.split(".")[2] + '.' + subdomain.split(".")[1]
            print(third_level_domain)
    except :
        sys.exit()
```

Steps to produce :-

- 1 - Open Your Terminal
- 2 - Write This Command

```
python3 thirdlevel.py subdomains.txt
```



attacker

My Methodology

Try To Discover What IPs Are Up By Using Tools e.g. [nmap](#) To Minimize Time Of Full PORTs Scanning

```
root@mine:~#nmap -sP -PE -PP -PS21,22,23,25,80,113,31339 -PA80,113,443,10042 --source-port 53 -T4 -iL ips.txt
```

" -sP " Skip Port Scan

" -PE " Send An ICMP echo Request

" -PP " Send Timestamp And Address Mask

" --source-port 53 " Use A Custom Source Port

" -PSport -PAanotherport " Sends An Empty TCP Packet With The SYN OR TCP ACK Flag Set

" -iL file.txt " Reading The Input From file.txt

" -T4 " Aggressive Mode Speeds Scans



attacker

My Methodology

Full PORTs Scanning With SYN Scan

```
root@mine:~#nmap -sSV --version-intensity 9 --min-parallelism 64 --min-hostgroup 16 --max-hostgroup 64 --max-retries 3 --min-rate 175 --max-rate 300 -Pn -n --source-port 53 --mtu 24 --data-length 25 -T4 --script-args http.useragent="Mozilla/5.0" --max-scan-delay 10 -p- -iL ips.txt -oA output
```

" -sSV --version-intensity 9 " SYN Scan AND Services Detection Scan

" --min-parallelism 64 " 64 Parallel Tasks

" --max-retries 3 " Number Of Retry Probing Port

" --min-hostgroup 16 --max-hostgroup 64 " Scan Minimum 16 AND Maximum 64 Hosts At One Time

" --mtu 24 " Specific MTU To The Packet

" --data-length 25 " Append Random Data



attacker

My Methodology

Full PORTs Scanning With SYN Scan II

```
root@mine:~#nmap -sSV --version-intensity 9 -PN -n --max-rtt-timeout 1000ms --min-parallelism 1000  
--max-retries 3 --source-port 53 --mtu 24 --data-length 25 --script-args http.useragent="Mozilla/5.0"  
-p- -iL ips.txt -oA output
```

" -sSV --version-intensity 9 " SYN Scan AND Services Detection Scan

" --min-parallelism 1000 " 1000 Parallel Tasks

" --max-retries 3 " Number Of Retry Probing Port

" --max-rtt-timeout 1000ms " Wait 1000 ms Before Timing Out

" --mtu 24 " Specific MTU To The Packet

" --data-length 25 " Append Random Data



attacker

My Methodology

Top 3674 Ports Scanning With SYN Scan

```
root@mine:~#nmap -sSV --version-intensity 9 --min-parallelism 64 --min-hostgroup 16 -T4 --mtu 24  
--max-hostgroup 64 --max-retries 3 --min-rate 175 --max-rate 300 -Pn -n --source-port 53 --data-length 25  
--script-args http.useragent="Mozilla/5.0" --max-scan-delay 10 --top-ports 3674 -iL ips.txt -oA output
```

" --min-rate 175 --max-rate 300 " Nmap will Send Rate Above 175 AND Less 300 packets In Second

" -Pn " Treat All Hosts Are Up

" -n " Never Do DNS Resolution

" --script-args http.useragent="Mozilla/5.0" " Change Default User Agent Header To Mozilla/5.0

" --max-scan-delay 10 " Time Between Probes

" -oA output " Output Files nmap , xml And gnmap



attacker

Subdomains Enumeration

Validation And Filter Wildcard



amass



findomain



subfinder



assetfinder



crtsh



Github Subdomains



attacker

My Methodology

Try To Valid The Third Level Subdomains List By Using **dig** AND Your Mind



Tweet

Steps to produce :-

- 1 - Open Your Terminal
- 2 - Write This Command
`dig nonexistent.api.comapny.com`
- 3 - If It Is Not Resolvable " NXDOMAIN " That Means :-
 - `api.company.com` Is Valid Subdomain
 - There Are Other Subdomains Under `api.comapny.com`
e.g. `exist.api.comapny.com`



attacker

My Methodology

Try To Filter The Wildcard Third Level Subdomains List By Using **dig** AND Your Mind



Blog

Steps to produce :-

- 1 - Open Your Terminal
- 2 - Write This Command
`dig nonexistent.api.comapny.com`
`dig *.api.company.com'`
- 3 - If The Two Are Resolvable To Same IP That Means :-
 - `api.company.com` is Wildcard Subdomain
- 4 - Delete This Subdomain From Third Level Subdomains List



attacker

My Methodology

Use **Subfinder** To Enumerate List Of Third Level Subdomains

```
root@mine:~#subfinder -dL list-of-thirdlevel.txt -config config.yaml -all -timeout 90 -silent -o out
```

" -dL list-of-thirdlevel.txt " List Of Third Level Subdomains e.g. corp.company.com

" -config config.yaml " Configuration File

" -all " Use All Sources For Enumeration

" -timeout 90 " Wait 90 Second Before Timing Out

" -silent " Show Only Subdomains In Output

" -o out " File To Write Output



attacker

My Methodology

Use **findomain** To Enumerate List Of Third Level Subdomains

```
root@mine:~#findomain --config file.ini --ua list.txt -f list-of-thirdlevel.txt --http-timeout 90 --quiet -o output.txt
```

" --config file.ini --ua list.txt " Configuration File And File Containing User Agents Values

" -f list-of-thirdlevel.txt " List Of Third Level Subdomains e.g. corp.company.com

" --http-timeout 90 " Wait 90 Second Before Timing Out

" --quiet " Show Only Subdomains In Output

" -o output.txt " File To Write Output



attacker

My Methodology

Use **amass** To Enumerate List Of Third Level Subdomains

```
root@mine:~#amass enum -passive -config file.ini -df list-of-thirdlevel.txt -timeout 90 -silent -o output.txt
```

" enum -passive " Passively Searching For Subdomains

" -config file.ini " Configuration File

" -timeout 90 " Wait 90 Second Before Timing Out

" -df list-of-thirdlevel.txt " List Of Third Level Subdomains e.g. corp.company.com

" -silent " Show Only Subdomains In Output

" -o output.txt " File To Write Output



attacker

My Methodology

Use **assetfinder** To Enumerate List Of Specific Fourth Level Subdomains

```
root@mine:~#cat analysis-output.txt | assetfinder --subs-only | tee -a subdomains.txt
```

" --subs-only " Display Only Subdomains Of Search Domain



attacker

My Methodology

Use **crtsh** To Enumerate List Of Specific Fourth Level Subdomains

```
root@mine:~#cat crtsh-fourthlevel-enumeration.sh
#!/usr/bin/env bash
for i in `cat analysis-output.txt`
do
    crtsh -o -q $i | tee -a subdomains.txt
done
root@mine:~#./crtsh-fourthlevel-enumeration.sh
```

" -o " Display Only Subdomains

" -q " Specific Third Level Domain To Search



attacker

My Methodology

Use **github-subdomains.py** To Enumerate List Of Specific Fourth Level Subdomains

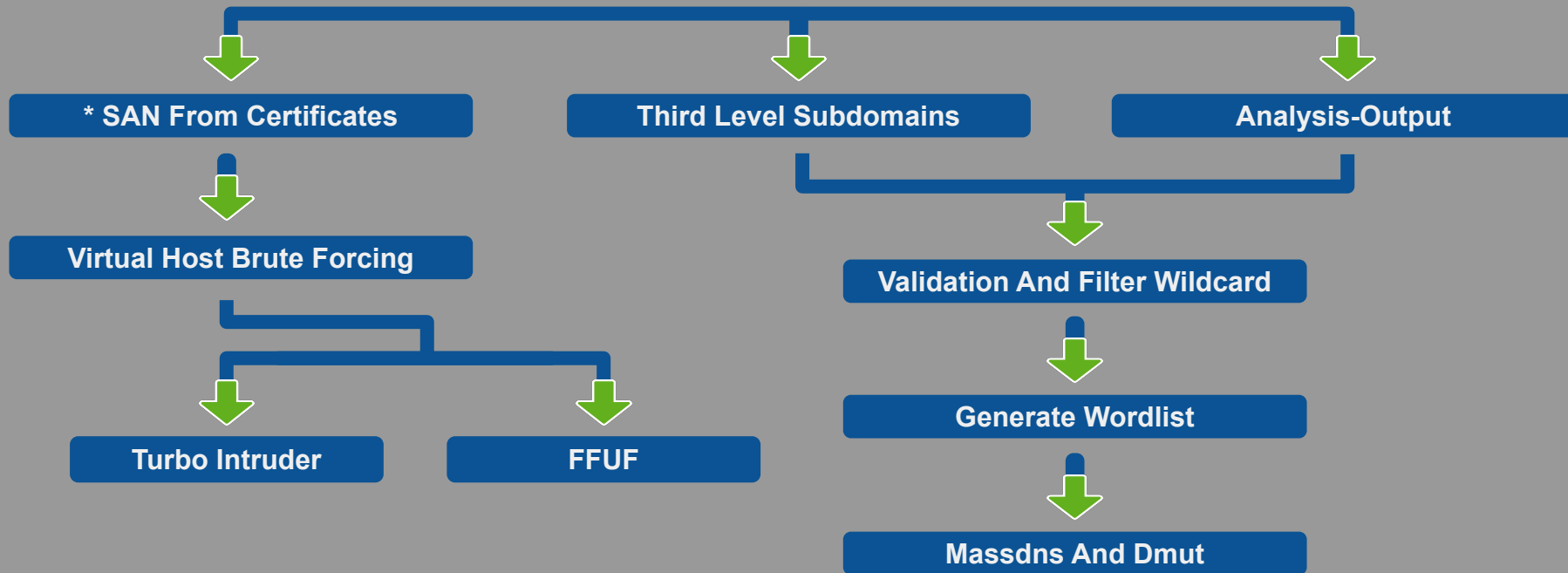
```
root@mine:~#cat github-fourthlevel-enumeration.sh
#!/usr/bin/env bash
for i in `cat analysis-output.txt`
do
    python3 github-subdomains.py -d $i | tee -a subdomains.txt
done
root@mine:~#./github-fourthlevel-enumeration.sh
```

" -d " Fourth Level Subdomain For Search



attacker

Subdomains Brute Forcing





attacker

Analysis Output Of IPv4-Reconnaissance AND Subdomains-Enumeration



Mine

```
root@mine:~#cat list-of-subdomains.txt  
api.corp.company.com  
api.test.company.com  
dev.api.company.com  
test.api.company.com  
ini.api.company.com
```

Steps to produce :-

- 1 - Open list-of-subdomains.txt With Text Editor
- 2 - Grep A Specific Pattern e.g.
api.FUZZ.comapny.com
FUZZ.api.comapny.com



attacker

Extract Third Level Domains From Output Of Subdomains Enumeration



Mine

```
#!/usr/bin/env python3
import os
import sys
import argparse
parser = argparse.ArgumentParser()
parser.add_argument( "-f","--file",help="file that contains list of subdomains" )
parser.parse_args()
args = parser.parse_args()
if args.file:
    if os.path.isfile(args.file):
        list_of_subdomains = open( args.file, 'r' )
        file_of_subdomains = list_of_subdomains.read().split("\n")
        list_of_subdomains.close()
    else:
        parser.error( '%s file not found' % args.file )
for subdomain in file_of_subdomains :
    try :
        if subdomain.count(".") == 2 :
            print(subdomain)
        else :
            third_level_domain = subdomain.split(".")[3] + '.' + subdomain.split(".")[2] + '.' + subdomain.split(".")[1]
            print(third_level_domain)
    except :
        sys.exit()
```

Steps to produce :-

- 1 - Open Your Terminal
- 2 - Write This Command

python3 thirdlevel.py output.txt



attacker

My Methodology

Try To Valid The Third Level Subdomains List By Using **dig** AND Your Mind



Tweet

Steps to produce :-

- 1 - Open Your Terminal
- 2 - Write This Command
`dig nonexistent.api.comapny.com`
- 3 - If It Is Not Resolvable " NXDOMAIN " That Means :-
 - `api.company.com` Is Valid Subdomain
 - There Are Other Subdomains Under `api.comapny.com`
e.g. `exist.api.comapny.com`



attacker

My Methodology

Try To Filter The Wildcard Third Level Subdomains List By Using **dig** AND Your Mind



Blog

Steps to produce :-

- 1 - Open Your Terminal
- 2 - Write This Command
`dig nonexistent.api.comapny.com`
`dig *.api.company.com'`
- 3 - If The Two Are Resolvable To Same IP That Means :-
 - `api.company.com` is Wildcard Subdomain
- 4 - Delete This Subdomain From Third Level Subdomains List



attacker

My Methodology

Generate Your Wordlist e.g.

```
cat Subdomains.txt | sed 's/\./\n/g' | sort -u | tee -a words.txt
```



Tweet

BUG BOUNTY TIP

Target based wordlist

- Take all your subdomains for your target
- Split at the "."s and "-"s, then **un**i**q** them
- Add these to your wordlists

Use the custom list to bruteforce everything from **hidden subdomains** to **weak credentials**!



@Rhynorater

www.intigriti.com





attacker

Generate Wordlist Of Subdomains To Resolve Based On Analysis Output



Mine

```
root@mine:~#cat genearet-wordlist.sh
#!/usr/bin/env bash
for i in `cat analysis-output..txt`
do
    for j in `cat words.txt`
    do
        echo "$i" | sed "s/FUZZ/$j/g" | tee -a $i.txt
    done
done
root@mine:~#./genearet-wordlist.sh
```

Steps to produce :-

- 1 - Open Your Terminal
- 2 - Write This Command
`./generate-wordlist.sh`



attacker

Generate Wordlist Of Subdomains To Resolve Based On Words Of Domains



Mine

```
root@mine:~#cat genearet-wordlist.sh
#!/usr/bin/env bash
for i in `cat thirdlevel-subdomains-list.txt`
do
    for j in `cat words.txt`
    do
        echo "$j.$i" | tee -a $i.txt
    done
done
root@mine:~#./genearet-wordlist.sh
```

Steps to produce :-

- 1 - Open Your Terminal
- 2 - Write This Command
`./generate-wordlist.sh`



attacker

My Methodology

Use Tools e.g. **subgen** To Generate Wordlist Based On Names Of Subdomains

```
root@mine:~#cat genearet-wordlist.sh
#!/usr/bin/env bash
for i in `cat thirdlevel-subdomains-list.txt`
do
    cat common-words-e.g.-corp-test-dev-internal-etc.txt | subgen -d "$i" | tee -a $i.txt
done
root@mine:~#./genearet-wordlist.sh
```

" -d "api.company.com" " Name Of Third Level Subdomains To Generate Similar Pattern



attacker

My Methodology

Use Tools **e.g. dnsvalidator** OR **fresh.py** With **massdns** To Resolve These Subdomains To IPs

```
root@mine:~#cat resolve.sh
#!/usr/bin/env bash
dnsvalidator -tL https://public-dns.info/nameservers.txt -threads 20 -o resolvers.txt
python3 fresh.py -o resolvers.txt
./bin/massdns -r resolvers.txt -t A -o S -w output.txt subdomains-to-resolve.txt
root@mine:~#./resolve.sh
```

" -r resolvers.txt " List Of DNS Servers IPs

" -o S -w output.txt " Normal Output Into Text File



attacker

My Methodology

Use Tools **e.g. dmut** To Perform Permutations And Mutations etc And Brute Force The Result

```
root@mine:~#dmut --workers 100 -d common-words.txt --dns-retries 5 --dnsFile resolvers.txt  
--dns-errorLimit 50 --dns-timeout 3000 --show-stats --output results.txt
```

" -d words.txt " Common Words To Permute etc

" --dns-retries 5 " Try 5 Times In Failed Queries

" --dns-errorLimit 50 " 50 Errors To Disable DNS

" --dns-timeout 3000 " Wait 3 Second To Time Out



attacker

My Methodology

Use Tools **e.g. SANextract** To Get * In Subject Alternative Names

```
root@mine:~#cat httpservice443alive.txt | ./SANextract -timeout 30s -json | tee -a vhosts.json  
root@mine:~#cat vhosts.json | grep "" | tee -a virtual-host-scanning.json
```



attacker

My Methodology

FUZZ The Host Header e.g. Host: FUZZ.company.com To Get Internal Hosts By Using FFUF OR Turbo Intruder

-  Slides
-  Tweet

GET / HTTP/1.1

Host: FUZZ.company.com

User-Agent: Mozilla/5.0

Referer: https://previous.com/path

Origin: https://www.company.com



attacker

My Methodology

```
root@mine:~#ffuf -u https://l.P.v.4/ -H "Host: FUZZ.company.com" -w wordlist.txt -c -mc 200  
-fr "Your-Regex" -timeout 30 -s -replay-proxy http://localhost:8080
```

" -url URL " URL Of Target e.g. https://www.company.com

" -H "Host: FUZZ.company.com" " Fuzz Host Header To Get Subdomains Under company.com

" -w wordlist.txt " Path To The Wordlist

" -c " Colorize Output

" -mc 200 " Match 200 OK HTTP status code

" -fr "Regex" " Filter This Pattern

" -timeout 30 " Wait 30 Second Before Timing Out

" -s " Silent Mode

" -replay-proxy http://localhost:8080 " Send Only Unfiltered Requests Through A Replay Proxy



attacker

FUZZ Host Header By Using Turbo Intruder



Mine

```
root@mine:~#cat file-of-turbo-intruder.py
def queueRequests(target, wordlists):
    engine = RequestEngine(endpoint=target.endpoint,
                           concurrentConnections=100,
                           requestsPerConnection=100,
                           pipeline=True
    )

    for word in open("/path/wordlist"):
        engine.queue(target.req, word.rstrip())
def handleResponse(req, interesting):
    if 'HTTP' in req.response:
        table.add(req)
```

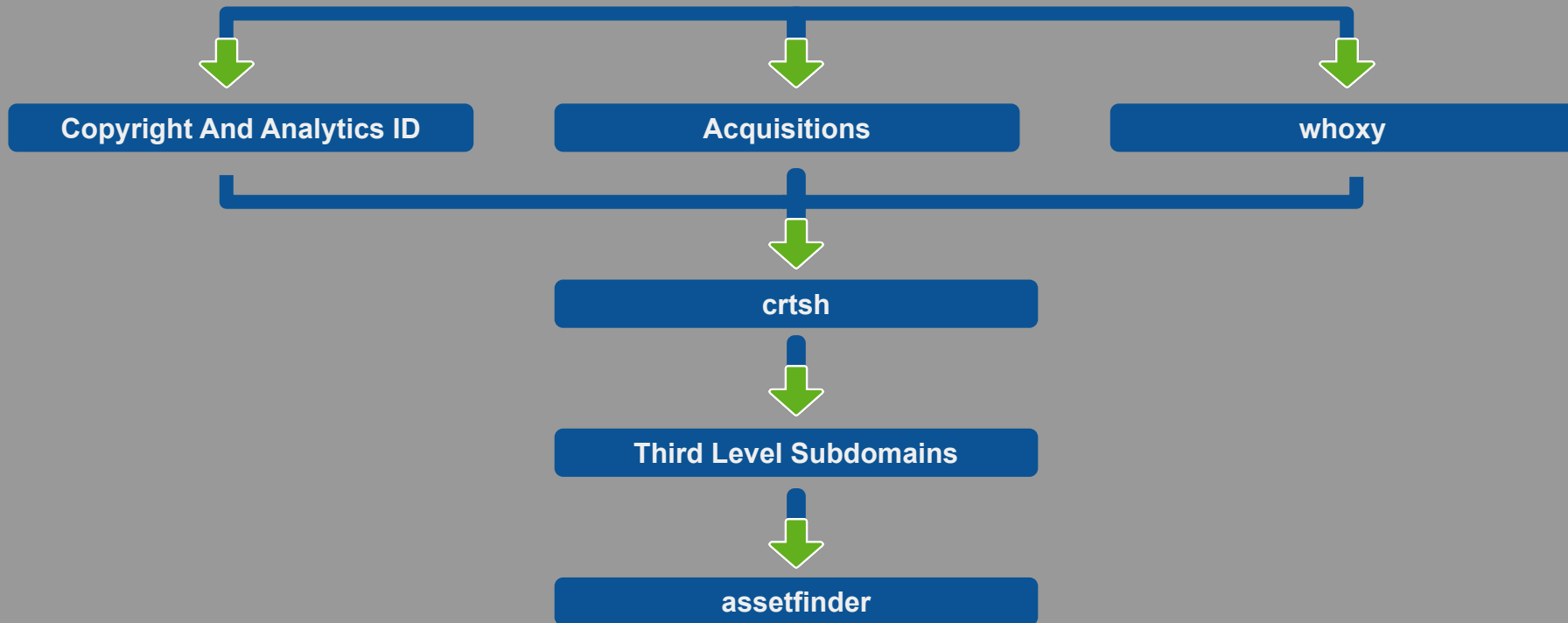
Steps to produce :-

- 1 - Sent Request To Turbo Intruder
- 2 - Add **%s** In Host Header e.g. **Host: %s.company.com**
- 3 - Click Attack
- 4 - Filter Your Result



attacker

Horizontal Domains





attacker

My Methodology

Enumerate Related Domains By Using Registrant Email

```
python3 domain-finder.py --key API-Key -d company.com
```



Tweet



Tweet

BUG BOUNTY TIP

“Want to find more company owned domains? Use whoxy.com to perform reverse whois lookups with the email used to register the main domain”

 [@hacker_](https://twitter.com/hacker_)





attacker

My Methodology

Google



site:acquiredby.co company

Google Search

I'm Feeling Lucky



attacker

My Methodology

Enumerate Related Domains By Using Copyright Mark



Tweet

BUG BOUNTY TIP

“Google the **copyright** footer to get more company domains”
@_zulln

Google “© 2019 Spotify AB”

Spotify Design - Events
<https://spotify.design/events/14-designing-for-tomorrow/> • Vertaal deze pagina
See what's upcoming in Spotify Design's events calendar.

Spotify For Brands
<https://spotifyforbrands.com/> • Vertaal deze pagina
Your audience listens in real-time moments throughout the day. Learn how they stream on Spotify, and how to connect with them in the right context.





attacker

My Methodology

Google



"© 2020 company" "© company 2020"

Google Search

I'm Feeling Lucky



attacker

My Methodology

Try To Enumerate Related Domains By Using Google Analytics ID



Tweet

Steps to produce :-

- 1 - Open Your Terminal
- 2 - Write This Command
`curl -s https://www.company.com | grep 'ga("create"'`
- 3 - Copy Google Analytics ID e.g. UA-*****_**
- 4 - Visit This URL
`site-overview.com/website-report-search/analytics-account-id/ID`
- 5 - Change Google Analytics ID Without UA- To ID



attacker

My Methodology

Reverse Analytics Search

company.com OR UA-11040216

Search Analytics ID or Domain



attacker

My Methodology

Use **crtsh** To Enumerate Subdomains

```
root@mine:~#cat crtsh-subdomains-enumeration.sh
#!/usr/bin/env bash
for i in `cat related-domains-output.txt`
do
    crtsh -o -q $i | tee -a related-subdomains.txt
done
root@mine:~#./crtsh-subdomains-enumeration.sh
```

" -o " Display Only Subdomains

" -q " Specific Related Domain To Search



attacker

Extract Third Level Domains From List Of Subdomains



Mine

```
#!/usr/bin/env python3
import os
import sys
import argparse
parser = argparse.ArgumentParser()
parser.add_argument( "-f","--file",help="file that contains list of subdomains" )
parser.parse_args()
args = parser.parse_args()
if args.file:
    if os.path.isfile(args.file):
        list_of_subdomains = open( args.file, 'r' )
        file_of_subdomains = list_of_subdomains.read().split("\n")
        list_of_subdomains.close()
    else:
        parser.error( "%s file not found" % args.file )
for subdomain in file_of_subdomains :
    try :
        if subdomain.count(".") == 2 :
            print(subdomain)
        else :
            third_level_domain = subdomain.split(".")[3] + '.' + subdomain.split(".")[2] + '.' + subdomain.split(".")[1]
            print(third_level_domain)
    except :
        sys.exit()
```

Steps to produce :-

- 1 - Open Your Terminal
- 2 - Write This Command

```
python3 thirdlevel.py related-subdomains.txt
```




attacker

My Methodology

Use **assetfinder** To Enumerate List Of Third Level Subdomains

```
root@mine:~#cat third-level-subdomains.txt | assetfinder --subs-only | tee -a all-subdomains.txt
```

" --subs-only " Display Only Subdomains Of Search Domain

Thank You

Mahmoud M. Awali

 **@0xAwali**