==========

[*] General Notes

==========

```
# Burp Scope :
    .*\.example\.com$

# Zap Scope :
    https?://[a-zA-Z0-9_~.-]{1,}.example.com.*
```

```
- in big comp. the subdomains used to host many other services
    - examples: mail.site.com , ftp.site.com , ..
    - this help in security,load, ..

- comp. register domain > to public ip/cidr
- to find info's about the registeration of any domin use [ whois ]
- we could host other subdomains/sites on same server

- In case of crop subdomains if the robots.txt give `Not Found` error
without redirect to login page , it is recommanded to directory Fuzzing.
- We also may login in some cases with our email and paybass cotrol

- cat live-subs.txt | grep -E
"portal|login|admin|signin|dashboard|test|internal|dev|developer|signup|r
egister|panel"


- awk > let us use more than one delimiter
    - Ex:  [ awk -F "AS" '{print $2}'] >> to extract ASN No.
```

BBP_Checklist :

- [ ] Subdomain Enumeration
- [ ] Internet Search Engine Discovery (Shodan, Censys, etc.)
- [ ] Google Dorking
- [ ] Internals Subdomains/virtual hosts
- [ ] IP enumeration
- [ ] Port Scanning

- [ ] Wayback History
- [ ] Technology Fingerprinting
- [ ] Directory Enumeration
- [ ] Web Crawler & URL Discovery
- [ ] Endpoints/Parameter Fuzzing
- [ ] Hardcoded Information in JavaScript
- [ ] Search for API Endpoints
- [ ] Web Apps Pentesting

==========

[*] Domain Enumeration

==========

Acquisitions :

- www.crunchbase.com (Find Acquisitions)

WHOIS :

```
whois example.com
```

```
amass intel -whois -d example.com
```

ASNs :

- Manual:
  - http://bgp.he.net
  - https://bgpview.io
  - https://ipinfo.io

```
# Validate ASN Ownership**
whois AS<number>

# Get IP Prefixes Owned by the ASN
whois -h whois.cymru.com " -v AS<number>"
curl ipinfo.io/AS<number>
```

```
    # Active Scanning of Live Hosts
    masscan -p1-65535 <ip_range> --rate=1000
    naabu -iL ip_ranges.txt -top-ports 1000 -o naabu_common.json
```

```
echo ASXXXXX | asnmap | naabu -p 443 | httpx
```

```
amass intel -active -asn <asn-no>
```

```
echo X.x.x.0/24 | /root/go/bin/tlsx -san -cn -silent -resp-only |
/root/go/bin/dnsx -silent | /root/go/bin/httpx

echo X.X.X.0/24 | /root/go/bin/tlsx -san -cn -silent -resp-only |
/root/go/bin/dnsx -silent | /root/go/bin/httpx -td -sc -fr -title -ip -rl
10 -t 2 -r /root/resolvers.txt -random-agent -o alive-subs-ASN.txt
```

```
- Find a list ASN numbers (Not Accuret) :
    - amass intel -org [company-name]
```

Shodan Dorking :

```
org:"Intigriti"

hostname:test.com

asn:AS1234

http.status:200 org:"Intigriti"

ssl.cert.subject.CN:"intigriti.com"

http.favicon.hash:<favicon_hash>

org:<company> http.component:php

org:<company> http.title:Login,Log in,Register,Signin, Sign in, Sign up

org:<company> http.title:"Index of"
```

```
org:<company> http.status:200,404 -port:80 -port:443 -port:8080 -port:8443

org:<company> product:jenkins
```

Google Dorking :

```
site:   filetype:jsp
site:   filetype:php
site:   filetype:asp

site:*.example.com intitle:"index of /"

site:test.com inurl:file.php?p=

site:   inurl:&

site:   inurl:login  -www

site:   inurl:register  -www

site:   inurl:signup  -www

site:  inurl:admin -www

site:  -www  inurl:api

site:  inurl:v1 -www

site:  inurl:/app -www
```

Github Dorks :

```
- "company" password
- "company" secret
```

```
- "company" credentials
- "company" token
- "company" config
- "company" key
- "company" pass
- "company" login
- "company" ftp
- "company" pwd
- "company" ssh_auth_password
- "company" send_key
- "company" send_keys
-
```

==========

[*] Subdomain Enumeration

==========

Subdomain naming conventions.

- Companies often use **predictable and commonly structured subdomain naming conventions**.

Order of Naming Convention :

1. **Environment** first (dev, test, stage)
2. **Service or product** next (api, app, web)
3. **Region or instance ID** (optional)

Format Examples :

- `dev-api-us.domain.com`
- `staging-web1.domain.com`
- `qa-app-v2.domain.com`

Crt.sh:

```
curl -s https://crt.sh/\?q\=\domain.com\&output\=json | jq -r '.
[].name_value' | grep -Po '(\w+\.\w+\.\w+)$' | tee subdomains.txt
```

Brute-forcing:

```
shuffledns -mode bruteforce -d <domain> -w <wordlist> -r <resolvers> -o
<output>
```

Alteration:

```
# used with large orgs
shuffledns -d <domain> -w <permutations-wordlist> -r <resolvers> -o
<output>
```

ASNs :

- Manual:
  - http://bgp.he.net
  - https://bgpview.io
  - https://ipinfo.io

```
    # Validate ASN Ownership**
     whois AS<number>

    # Get IP Prefixes Owned by the ASN
    whois -h whois.cymru.com " -v AS<number>"
    curl ipinfo.io/AS<number>

    # Active Scanning of Live Hosts
    masscan -p1-65535 <ip_range> --rate=1000
    naabu -iL ip_ranges.txt -top-ports 1000 -o naabu_common.json
```

- Automated:

```
echo ASXXXXX | asnmap | naabu -p 443 | httpx
```

```
amass intel -active -asn <asn-no>
```

```
echo x.x.x.0/24 | /root/go/bin/tlsx -san -cn -silent -resp-only |
/root/go/bin/dnsx -silent | /root/go/bin/httpx
```

Subfinder :

```
/root/go/bin/subfinder -rl 50 -all -r /root/resolvers.txt -d $domain  |
tee -a subs.txt
```

Amass :

```
- Find a list ASN numbers (Not Accuret) :
    - amass intel -org [company-name]
- Subdomains :
    - amass enum -active -d [company-name]
    - amass intel -asn [ASN No.]
    - amass intel -cidr [CIDR Range]
    - amass intel -whois -d [Domain Name] // Use Email regs. to search
for other domains related for that company where those domains contains
other subdomains (we must validats that those domains belong to that
company)
```

```
/root/go/bin/amass enum --passive -r /root/resolvers.txt -d $domain | tee
-a subs.txt

/root/go/bin/amass enum -active -r /root/resolvers.txt -norecursive -
noalts -d $domain | tee -a subs.txt
```

Internals Subdomains/virtual hosts :

- locate internals subdomains [internal-api's , devops , tools , ..]

```
cat subs.txt | xargs -I{} host {} | tee host-out.txt

dirsearch -u -e -H 'X-Forwarded-For: 127.0.0.1'
```

```
curl -H 'Host: 127.0.0.1' url
```

Shodan :

- look for different ASNs from httpx output and use shodan to look for that subnet .
- Ex:

```
    Search > 187.77.90.0/24
    or
    Search > asn:ASN4587
    or
    earch > asn:ASN4587 "title:login"
        // look for apps on those ips
```

Censys :

```
copy page  > \b[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}
nano r-ips.txt
cat r-ips.txt | httpx -sc -mc 200
```

Live Subdomain Validation (DNS Resolution) :

```
/root/go/bin/httpx -td -sc -fr -asn -title -ip -rl 10 -t 2 -r
/root/resolvers.txt -p 8000,8080,8443,443,80,3000,5000 -random-agent -H
"Referer: https ://domain.com" -o alive-subs-HTTPX.txt
-l

/root/go/bin/httpx -l live-subs.txt -p
8080,8443,8000,8888,8081,8181,3306,5432,6379,27017,15672,10000,9090,5900
-threads 80 -title -sc -cl -server -ip -o services-ports.txt
```

```
dnsx -l active_subs.txt -resp -t 10 -o resolved_subs.txt
```

```
dnsx -resp-only -silent
```

IP enumeration

- Internet scanning :
  - https://en.fofa.info/
  - censys.com
  - shodan >> hostname:grab.com
  - HTTPX:

```
cat alive-subs-HTTPX.txt  | grep -oE "\b([0-9]{1,3}\.){3}[0-9]{1,3}\b" | sort -u | tee ips-online.txt
```

```
- Massdns:
```

```
massdns -r /root/resolvers.txt -t A -o S -w massdns-list.out subs.txt

cat massdns-list.out | awk '{print $3}' | sort -u | grep -oE "\b([0-9]{1,3}\.){3}[0-9]{1,3}\b" > ips-online.txt
```

Ports Enumeration/Scanning :

- `masscan`

```
masscan -iL ips-online.txt -p0-65535 --rate 10  --randomize-hosts  -oL masscan.out
```

- `naabu`

```
/root/go/bin/naabu -host x.x.x.x/24

# Scan entire internal subnet and save results
naabu -host 192.168.1.1/24 -o internal_ports.txt
# Scan specific port ranges
naabu -host 192.168.1.1/24 -p 80,443,8000-9000 -o internal_ports.txt
```

```
/root/go/bin/naabu  -c 2 -rate 10 -exclude-ports 25  -top-ports 1000 -r
/root/resolvers.txt -l ips-online.txt -o naabu-ports-scan.txt

/root/go/bin/naabu  -c 2 -rate 10 -top-ports 1000 -r /root/resolvers.txt
 -nmap-cli 'nmap -sV -sC' -o naabu_result.txt

/root/go/bin/naabu -c 2 -rate 10 -exclude-ports 25 -top-ports 1000 -r
/root/resolvers.txt -host X.X.X.X/24 -o naabu-ports-scan.txt
```

- nmap

```
nmap -A -sV -sC -Pn --max-rate 50
nmap -sS -A -PN -p- --script=http-title site.com
nmap --script vuln -sV -sC --top-ports 1000 -T2 -iL
nmap -iL ips.txt -sSV -A -O -Pn -F -oX nmap.xml
nmap -sV --top-ports 1000
nmap --data-length 30
nmap --script ssl-cert -p 443 <subdomain/ip>

nmap -sV -sS -D RND:50 <ip>
```

JavaScript Analysis

```
# Js Links and hardcoded secrets
/root/go/bin/katana -jsl -jc -rl 10 -p 1 -rd 1 -c 2 -u

# query strings parameters
/root/go/bin/katana  -rl 10 -c 1 -f qurl -u

cat f-urls.txt | sort -u | grep -a ".js$" |/root/go/bin/httpx  -mc 200 -r
/root/resolvers.txt  -random-agent -threads 2 -rl 10  | tee  live-js-
urls.txt

/root/go/bin/nuclei  -t /root/nuclei-templates/http/exposures/tokens -rl
10 -bs 2 -c 2 -l live-js-urls.txt -o nuclei_exposures.txt
```

Archive Enumeration

Archive URL Collection

```
gau --subs target.com | anew archived_urls.txt
gau --threads 2 --blacklist ttf,woff,svg,png,jpg,gif,mp4,jpeg
http://url.com/ --o urls.txt

cat domain.txt | uro | grep -E '\.
(xml|json|pdf|sql|txt|zip|tar|gz|tgz|bak|7z|rar|log|cache|secret|db|backu
p|yml|gz|config|csv|yaml|md|md5|exe|dll|bin|ini|bat|sh|tar|deb|rpm|env|dm
g|tmp|crt|pem|key|pub|asc)'
```

```
waybackurls target.com | anew wayback_urls.txt
```

```
/root/go/bin/urlfinder -f
woff,css,png,svg,jpeg,gif,jpg,woff2,swf,ico,tif,tiff,pdf,svg,webp,avif,ap
k  -all -rl 10  -d mars.com -o mars.com.txt
```

```
waymore -i <domain> -mode U -mc 200 --no-subs -oU
waymore -i http://vulnweb.com -mode R --no-subs -oR  waymore-Respons.txt
```

Content Discovery

Feroxbuster

```
feroxbuster -u https://target.com -w /usr/share/wordlists/common.txt -r -
t 20 -o recursive_results.txt

feroxbuster --random-agent -k -E -B --scan-limit 2 -t 1 --rate-limit 10 -
d 2 -w -u
```

FFuF

```
ffuf -u https://target.com/FUZZ -w
/usr/share/wordlists/content_discovery.txt -mc 200,403 -t 1 -p "0.1-0.3"
-r -rate 10 -recursion-depth 3 -o ffuf_results.txt

ffuf -u http://127.0.0.1:42000/api/FUZZ  -t 1 -p "0.1-0.3" -r -rate 10 -w
/root/without_dots.txt
```

Gospider

```
/root/go/bin/gospider -s "http://testphp.vulnweb.com/" -c 2 -d 5 --
blacklist ".(jpg|jpeg|gif|css|tif|tiff|png|ttf|woff|woff2|ico)"

/root/go/bin/gospider -s "http://127.0.0.1:3000/" -c 2 -K -d 5 --
blacklist .(ico|css|svg) -o go-s-out.txt
```

Dirsearch

```
dirsearch --random-agent --max-rate=10 -t 2 -e
php,asp,aspx,jsp,html,zip,jar,sql,log,db,backup,json,gz,rar,conf -u

dirsearch -i 200 -e
php,bak,old,zip,tar.gz,txt,log,conf,json,asp,jsp,aspx,yml,yaml,rar

dirsearch --random-agent --max-rate=10 -t 2 -H 'X-Forwarded-For:
127.0.0.1' -u

dirsearch -e xml,json,sql,db,log,yml,yaml,bak,txt,tar,gz,zip -x
403,404,500,400,502,503,429 --random-agent -u
```

Web Crawler & URL Discovery

Katana

```
katana -jsl -jc -rl 10 -p 1 -rd 1 -c 2 -u

katana -u subdomains-alive.txt -jsl -jc -rl 5 -p 1 -rd 1 -c 2 -d 5  -kf
-fx -ef woff,css,png,svg,jpeg,gif,jpg,woff2 -o  allurls.txt
```

Gospider

```
gospider -s "https://target.com" -d 2 -o gospider_output/

gospider -s "https://target.com" -c 2 -K -d 5 --blacklist .(ico|css|svg)
-o gospider_output/
```

Hakrawler

```
echo "https://target.example.com" | hakrawler -depth 2 -plain -js -out
hakrawler_results.txt
```

Parameter Discovery

Arjun Parameter Discovery

```
arjun -u "https://target.example.com" -m GET,POST --stable -w /burp-
params -o params.json
```

Katana

```
katana -d 5 -rl 10 -c 2  -f qurl -u
```

FFuF Parameter Bruteforce

```
ffuf -u https://target.com/page.php?FUZZ=test -w
/usr/share/wordlists/params.txt -o parameter_results.txt
```

Arjun Parameter Discovery

```
arjun -u "https://target.example.com" -m GET,POST --stable -o params.json

arjun -m GET,POST -d 1 --rate-limit 10 --stable  -u $domain  -oT arjun-
params.txt
```

ParamSpider Web Parameters

```
paramspider.py --domain https://cpcalendars.cartscity.com --exclude
woff,css,js,png,svg,php,jpg --output g.txt
```

Automation Scanning Tools

Nuclei :

```
/root/go/bin/nuclei -t /root/nuclei-templates/http/exposures/tokens -rl
10 -bs 2 -c 2 -l live-js-urls.txt -o nuclei_exposures.txt
```

```
/root/go/bin/nuclei -rl 10 -bs 2 -c 2 -dast -list live-kat-params.txt -o
nuclei_output_vulnerabilities.txt

cat  urls/param-urls.txt | /root/go/bin/gf sqli | /root/go/bin/nuclei -rl
10 -bs 2 -c 2 -dast -tags sqli -o nuclei_sqli.txt

/root/go/bin/nuclei -tags xss,lfi,sqli -u

/root/go/bin/nuclei -rl 10 -bs 2 -c 1 -dast -tags xss,sqli,lfi,cmdi,ssrf
-l

/root/go/bin/nuclei -rl 10 -bs 2 -c 2 -as critical,high,medium -u

# With specific templates
nuclei -u http://target -t misconfiguration/

nuclei -u http://target -t cves
nuclei -u http://target -tags exposure,disclosure,misconfig
nuclei -u http://target -tags default-login
nuclei -u http://target -tags wordpress
```

IIS :

```
shortscan -F -V uri

sns -u $host
```

Cloud Enumeration :

```
cloud_enum -k target.com -b buckets.txt -o cloud_enum_results.txt
```

S3 Bucket Access Test

```
aws s3 ls s3://<bucket_name> --no-sign-request
```

S3 Bucket Content Dump

```
python3 AWSBucketDump.py -b target-bucket -o dumped_data/
```

Dalfox :

```
/root/go/bin/dalfox --waf-evasion  url  -p

/root/go/bin/dalfox  -w 10 --skip-grepping --skip-headless --skip-mining-
all --skip-xss-scanning file param-urls.txt

cat valid-Urls.txt | /root/go/bin/gf xss | sed 's/=.*/=/' | sort -u |
/root/go/bin/dalfox  pipe
```

SQLi :

```
sqlmap --banner --random-agent --batch --level=3 --
tamper="between,randomcase,space2comment" -u

sqlmap -m sqli.txt  --banner --random-agent --batch --level=3 --
tamper="between,randomcase,space2comment"

sqlmap -r req.txt --string "string when true "

sqlmap -r req.txt -p login --level 3 --dbs --batch

sqlmap -u --data="foo=bar&foo2=bar2"

sqlmap -u --dbs --forms --crawl=2
```

Subdomain Takeover :

```
/root/go/bin/subjack -w 404-list.txt -ssl -t 2 -timeout 30 -o 404-
results.txt -v -c

/root/go/pkg/mod/github.com/haccer/subjack@v0.0.0-20201112041112-
49c51e57deab/fingerprints.json
```

==========

[*] After Recon

==========

Based On Status Code :

1. 200 status code subdomains

1. **Service identification** :
    1. port scanning
    2. Wappalyzer
2. **Enumerate Directories and Files :**
   [/admin, /config, /backup.zip, or /db.sql.]
3. **Enumerate endpoints, parameters**
4. **Questions to ask** :
    1. How does the app handle special chars <>"/
    2. How does the site reference a user
    3. Are there multiple user roles
5. **App Feature analysis :**
    - File Handling [File Upload, File Inclusion]
    - Broken **Access Control**
    - Broken **User Authentication** [OAuth / SSO / 2FA]
    - Injection (SQL, No-SQL, Command)
    - Security Miss-configuration
    - Business logic flaws
    - Other

---

2. 404 status code subdomains

1. Test for Subdomain Takeover [Subjack or Subzy]
2. Enumerate Hidden Directories and Files [custom 404 pages]

---

3. 403 status code subdomains

1. Enumerate Directory and File Access (e.g., /admin, /backup.zip)
2. Check for HTTP Header-Based Restrictions
   [Test bypasses by modifying headers like :]
   1. User-Agent

2. Referer
3. Origin
3. Test for Method-Based Access
4. Look for Path Traversal Vulnerabilities
5. Bypass Using Alternate Encoding

---

4. 5xx status code subdomains

1. **500 Internal Server Error — Debug Disclosure**

- Send unexpected input or malformed requests:

```
curl -X POST -d 'username=admin&password=\'' http://target.com/login
```

Finding:

- The server returns:
  `HTTP/1.1 500 Internal Server Error X-Powered-By: Express Error: SyntaxError: Unexpected token '`

  Impact:
- Stack traces or error messages may leak:
  - Internal file paths
  - Programming language
  - Frameworks in use
  - SQL syntax hints

  These hints can assist in SQLi, **RCE**, or **LFI exploitation**

2. **502 Bad Gateway — Misconfigured Reverse Proxy**

Fuzz uncommon ports or send unexpected Host headers:

```
curl -H "Host: internal.target.com" http://public.target.com
```

Finding:

- The response is a `502 Bad Gateway`
- Or you may find that some internal subdomains yield 502 while others 200

  Impact:

- Reveals reverse proxy paths
- May indicate **internal services** are exposed
- Can help in **SSRF** or **port enumeration**

3. **503 Service Unavailable — DoS Potential**

Send high numbers of concurrent requests or use slow HTTP techniques:

```
slowloris -dns target.com -port 80 -timeout 40
```

Finding:

- Target returns:
  `HTTP/1.1 503 Service Unavailable Retry-After: 120`

Impact:

- Application fails under moderate load
- Can lead to **Denial of Service (DoS)** vulnerability report

4. **504 Gateway Timeout — SSRF/Backend Timeout**

Try internal service URLs (SSRF testing):

```
curl "http://target.com/api/fetch?url=http://169.254.169.254/latest/meta-data/"
```

Finding:

- Response: `504 Gateway Timeout`
- Suggests access to internal services was attempted but timed out
  Impact:
- May confirm **SSRF capability**
- Indicates that **internal services** (e.g., AWS metadata) are **reachable**

Heat Mapping :

- Look For those **Bugs** in those Web Apps **Functionality** :

1. **Content Types** :
   - Look for multipart-forms [Shell, injections, ++]
   - Look for content type XML [XXE]

- Look for content type json [API vulns]

2. **APIs** :
   [Broken Auth ,BAC , mass assignment ,SQLi , SSRF , File Upload, Data Exposure ,Command / Template Injection]
3. **Account Section** :
   - Profile [stord XSS]
   - App Custom Fields [stord XSS, SSTI]
   - Integrations [SSRF, XSS]
4. **Paths or URLs passed as values** : [SSRF, Redirs]
5. **Uploads Functions** :
   - Self Uploads
     - XML Based (Docs/PDF) [SSRF,XSS,XXE]
     - Image [XSS,Shell] {name, Binary_header , Metadata}
   - Where is data stored? [s3 perms]
6. **File download** :
   [Path traversal, info disclosure]
7. **Search bars** :
   [Reflected XSS, sql injection]

==========