



# Offsec Recon At Scale and Visualisation

Recon as a Data Challenge

David Lassig

28.Juli 2021



# Inhaltsübersicht

- 1 Wer bin ich?
- 2 Impuls
- 3 Ablauf
- 4 Subdomains
- 5 Portscans
- 6 Schwachstellenscans
- 7 URLs
- 8 Wie WAFs vermeiden?
- 9 Kern und Ausblick



- David Lassig (Hauptmann), Zeitsoldat bis 2022
- Großteil der Dienstzeit im Zentrum Cyber-Operationen (Ethical Hacking und Data Engineering), seit 2019 im CIHBw (ZSwKBw)
- “dienstlich” u.a. derzeit mit Orchestrierung und DevSecOps beschäftigt
- privat Beteiligung an BugBounty-Programmen/Plattformen und Maker-Basteleien (IoT, 3D-Druck)



# Inhaltsübersicht

- 1 Wer bin ich?
- 2 Impuls**
- 3 Ablauf
- 4 Subdomains
- 5 Portscans
- 6 Schwachstellenscans
- 7 URLs
- 8 Wie WAFs vermeiden?
- 9 Kern und Ausblick



# BugBounty und VDP

- großes Wachstum an BugBounty und VDP-Programmen
- “Crowd-Sourced” Security Scanning kann durch kein Tool ersetzt werden
- charakteristisch sind große Scopes mit vielen Zielen





# Große Scopes und viele Daten

## Recon data for Public Bug Bounty Programs

This list consists recon data curated from our DNS dataset, please send us a pull request to expand bug bounty programs [listed here](#). You can also query any domain using Chaos API key, with realtime updates.



new programs

new subdomains

hackerone

bugcrowd

intigriti

self hosted

with rewards

no rewards

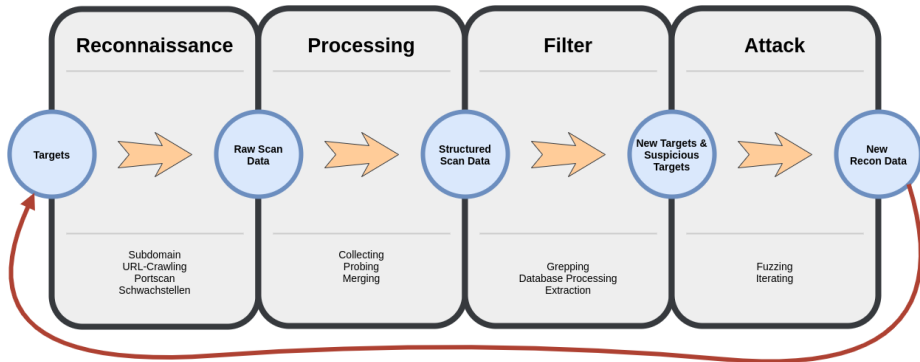
offers swag

503 programs

<input type="checkbox"/>	PROGRAM	OFFERS REWARD	OFFERS SWAG	NO OF SUBDOMAINS ↑	LAST UPDATED
<input type="checkbox"/>	Swisscom	Yes	No	2675524	4 Jul
<input type="checkbox"/>	Telenet	Yes	No	2632384	4 Jul
<input type="checkbox"/>	h Shopify	Yes	No	673386	4 Jul
<input type="checkbox"/>	h Verizon Media	Yes	No	427988  2	4 Jul
<input type="checkbox"/>	Cisco Meraki	Yes	No	380382  69	4 Jul
<input type="checkbox"/>	stanford	Yes	No	300460  2	4 Jul
<input type="checkbox"/>	h Alibaba	Yes	No	230718	4 Jul
<input type="checkbox"/>	h AT&T	Yes	No	217963	4 Jul
<input type="checkbox"/>	Microsoft Online Services	Yes	No	182890  1	4 Jul
<input type="checkbox"/>	h IBM	No	No	119994	4 Jul
<input type="checkbox"/>	h MTN Group	No	No	115439	4 Jul
<input type="checkbox"/>	WP Engine	No	No	99342	4 Jul
<input type="checkbox"/>	Google	Yes	No	81104	4 Jul
<input type="checkbox"/>	h Intuit	No	No	79998	4 Jul
<input type="checkbox"/>	h TTS Bug Bounty	Yes	No	74346  2	4 Jul



# Automatisierung und Datenpipelines





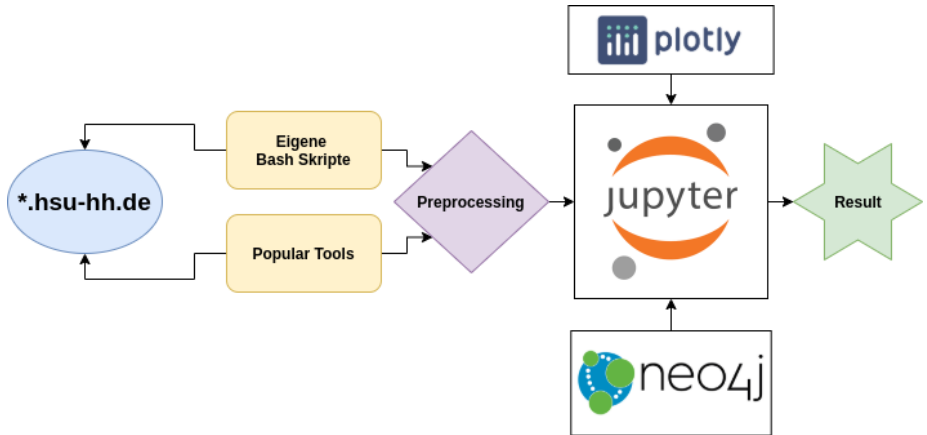
# Inhaltsübersicht

- 1 Wer bin ich?
- 2 Impuls
- 3 Ablauf**
- 4 Subdomains
- 5 Portscans
- 6 Schwachstellenscans
- 7 URLs
- 8 Wie WAFs vermeiden?
- 9 Kern und Ausblick





# Ablauf





# Inhaltsübersicht

- 1 Wer bin ich?
- 2 Impuls
- 3 Ablauf
- 4 Subdomains**
- 5 Portscans
- 6 Schwachstellenscans
- 7 URLs
- 8 Wie WAFs vermeiden?
- 9 Kern und Ausblick



# Subdomain Methodologie - Passiv

- SSL-Zertifikate (Censys, GoogleCT, FacebookCT)
- Öffentliche Database APIs (BinaryEdge, SecurityTrails, Chaos, C99)
- Web Archive (Wayback, ArchiveIt)
- Scraping (Baido, Bing, Yahoo)
- -> <https://github.com/OWASP/Amass>

```
amass enum -passive -d example.com -config path/amass_config.ini
```



# Subdomain Methodologie - Aktiv

- ReverseLookup
- Zonen-Transfer
- Bruteforce mit kontextbezogenen Permutationen
- CIDR und ASN Sweeping (ASN?)
- -> <https://github.com/OWASP/Amass>

```
amass enum -active -d example.com -asn 1234 -cidr 10.0.0.0/20 -config path/amass_config.ini
```



# Subdomain Processing und Visualisierung

```
for domain in $sd_source; do

# passive amass with additional scripts
amass enum -scripts $DIR/amass_exts/ -passive -d $domain -o $sd/passive_subs_amass_$domain -include assetfinder,subfinder,github-

# active amass
amass enum -active -d $domain $ama_asn $ama_cidr -o $sd/active_subs_amass_$domain -ip -config /home/user/tools/configs/amass_conf

# brute-forcing
subbrute.py ~/tools/wordlists/subdomains.txt $domain | filter-resolved > $sd/brute_subs_subbrute_$domain
```

## DEMO

Jupyter: Verarbeiten und Darstellen von amass/subbrute Daten als Graph



# Inhaltsübersicht

- 1 Wer bin ich?
- 2 Impuls
- 3 Ablauf
- 4 Subdomains
- 5 Portscans**
- 6 Schwachstellenscans
- 7 URLs
- 8 Wie WAFs vermeiden?
- 9 Kern und Ausblick



## Alte und neue Wege

- [+] **nmap** ist besonders genau
- [+] **nmap** hat viele Plugins für Fingerprinting
- [-] **nmap** ist sehr langsam und Default-Scans landen in Firewall
- [+] **masscan** ist besonders schnell (gesamtes Internet in 3min)
- [+] **masscan** nutzt eigenen TCP/UDP-Stack
- [-] **masscan** keine Plugins oder sonstiger Komfort
- -> Kombination aus Beidem
  - <https://github.com/capt-meelo/MassMap>



# Portscan Processing und Visualisierung

```
./massmap.sh scope_ips
```

## DEMO

Jupyter: Vearbeiten und Darstellen von massmap/nmap Daten als Graph





# Inhaltsübersicht

- 1 Wer bin ich?
- 2 Impuls
- 3 Ablauf
- 4 Subdomains
- 5 Portscans
- 6 Schwachstellenscans**
- 7 URLs
- 8 Wie WAFs vermeiden?
- 9 Kern und Ausblick



# Einfache Kollaboration und Skalierbarkeit gewinnt

- nikto und nmap-Plugins haben weiterhin Daseinsberechtigung
- jedoch zunehmend CI/CD, Skalierbarkeit und Erweiterbarkeit wichtig
- -> <https://github.com/projectdiscovery/nuclei>

```
id: amazon-mws-secret-token-value

info:
  author: puzzlepeaches
  name: Amazon MWS Secret Token
  severity: medium

requests:
  - method: GET
    path:
      - "{{BaseURL}}"

extractors:
  - type: regex
    part: body
    regex:
      - "amzn\\.mws\\.([0-9a-f]{8})-([0-9a-f]{4})-([0-9a-f]{4})-([0-9a-f]{4})-([0-9a-f]{12})"
```



# Schwachstellenscans Processing und Visualisierung

```
echo "$domain" | nuclei -rl 20 -c 3 -silent \  
-t ~/tools/signatures/nuclei-templates/ \  
-o "$sd"/nuclei_"$domain_fn"
```

## DEMO

Jupyter: Vearbeiten und Filtern von nuclei Scandaten



# Inhaltsübersicht

- 1 Wer bin ich?
- 2 Impuls
- 3 Ablauf
- 4 Subdomains
- 5 Portscans
- 6 Schwachstellenscans
- 7 URLs**
- 8 Wie WAFs vermeiden?
- 9 Kern und Ausblick



# Quellen und Tools

- Alte Versionen von Webseiten
  - WaybackMachine, Archive.org
  - -> <https://github.com/tomnomnom/waybackurls>, <https://github.com/lc/gau>
- Spidering über zugängliche Seiten
  - -> gospider, Burp Spider, scrapy
- Threat Exchanges und URL Crawl Datenbanken
  - Alien Labs Open Threat Exchange
  - Common Crawl
  - -> <https://github.com/lc/gau>

## DEMO

Shell: Einfachheit URL Farming mit gau



# One-Liner KungFu

- <https://github.com/dwisiswant0/awesome-oneliner-bugbounty>

```
gospider -S targets_urls.txt -c 10 -d 5 --blacklist ".(jpg|jpeg|gif|css)" --other-source \
| grep -e "code-200" \
| awk '{print $5}' | grep "=" \
| qsreplace -a \
| dalfox pipe
```

XSS-Test auf URLs mit Parametern

Herausfiltern der URLs mit Parametern

Crawling aller erreichbaren Endpunkte über alle URLs in target\_urls.txt

Permutationen über alle URLs und Parameter

Herausfiltern aller HTTP 200 Codes --> "erreichbar"



# One-Liner KungFu

```
[*] __ Start scan [SID:18][18/210][8.57%] / URL: https://ilias.hsu-hh.de/goto.php?client_id=unibw&targ
[I] Found 3 testing point in DOM base parameter mining
[I] Content-Type is text/html; charset=UTF-8is __
[I] X-Frame-Options is SAMEORIGIN
[*] Finish Scanneries[90.48%][19/210 Tasks][9.05%] Testing "client_id" param and waiting headless
[*] __ Start scan [SID:19][19/210][9.05%] / URL: https://ilias.hsu-hh.de/login.php?lang=en&target=
[I] Found 5 testing point in DOM base parameter mining
[I] Found 54 testing point in Dictionary base paramter mining
[I] Content-Type is text/html; charset=UTF-8
[I] X-Frame-Options is SAMEORIGIN
[I] Reflected cmd[doStandardAuthentication] param => Injected: /inATTR-double(4) .
294 line: rdAuthentication%5D=DalFoxtarget=&lang=de" ><div class="icon none small" aria-la
300 line: rdAuthentication%5D=DalFoxtarget=&lang=en" ><div class="icon none small" aria-la
306 line: rdAuthentication%5D=DalFoxtarget=&lang=fr" ><div class="icon none small" aria-la
312 line: rdAuthentication%5D=DalFoxtarget=&lang=it" ><div class="icon none small" ar
[I] Reflected enddate param => Injected: /inATTR-double(4) .
294 line: e/login.php?enddate=DalFoxtarget=&lang=de" ><div class="icon none small" aria-la
300 line: e/login.php?enddate=DalFoxtarget=&lang=en" ><div class="icon none small" aria-la
306 line: e/login.php?enddate=DalFoxtarget=&lang=fr" ><div class="icon none small" aria-la
312 line: e/login.php?enddate=DalFoxtarget=&lang=it" ><div class="icon none small" ar
[I] Reflected key param => Injected: /inATTR-double(4)
294 line: hh.de/login.php?key=DalFoxtarget=&lang=de" ><div class="icon none small" aria-la
300 line: hh.de/login.php?key=DalFoxtarget=&lang=en" ><div class="icon none small" aria-la
306 line: hh.de/login.php?key=DalFoxtarget=&lang=fr" ><div class="icon none small" aria-la
312 line: hh.de/login.php?key=DalFoxtarget=&lang=it" ><div class="icon none small" ar
[I] Reflected cat param => Injected: /inATTR-double(4) -
294 line: hh.de/login.php?cat=DalFoxtarget=&lang=de" ><div class="icon none small" aria-la
300 line: hh.de/login.php?cat=DalFoxtarget=&lang=en" ><div class="icon none small" aria-la
306 line: hh.de/login.php?cat=DalFoxtarget=&lang=fr" ><div class="icon none small" aria-la
312 line: hh.de/login.php?cat=DalFoxtarget=&lang=it" ><div class="icon none small" ar
[I] Reflected token param => Injected: /inATTR-double(4) -
```



# One-Liner KungFu

## ■ obfuscate the URL

```

132 line: /SID=4780c188-9/SRT=DaIFox/TTL=1/LNG=DU/USERINFO LOGIN" target=" blank">Ben
[V] Triggered XSS Payload (found DOM Object): ="onmouseenter=prompt(1) class=dalfox
36 line: c188-9/TTL=1/LNG=EN/"onmouseenter=prompt(1) class=dalfox ?="><img
59 line: c188-9/TTL=1/LNG=FR/"onmouseenter=prompt(1) class=dalfox ?="><img
[POC][V][GET] [REDACTED]DB=1/LNG=DU/LRSET=1/SET=1/SID=4780c188-9/TTL=1/%22onmouseenter=prompt%281%29%20class=dalfox%20?=[
[*] Finish Scan
[*] Start scan [SID:183][183/210][87.14%%] / URL [REDACTED]/DB=1/LNG=DU/LRSET=1/SET=1/SID=4780c188-9/TTL=1/LNG=EN/NXT
[I] Found 6 testing point in DOM base parameter mining
[I] Found 1 testing point in Dictionary base parameter mining

```





# Spider Processing

```
function_curl_and_save() {
    origin_url=$1
    filename=$(cat /dev/urandom | tr -dc 'a-zA-Z0-9' | fold -w 16 | head -n 1)

    origin=$(echo "$origin_url" | cut -d'|' -f1)
    url=$(echo "$origin_url" | cut -d'|' -f2-)

    if [ ! -z $url ]; then
        curl_resp=$(curl -Ls -m 3 -w "%{http_code}|$origin|$url|${url_effective}|${time_redirect}|${num_redirects}|${size_download}|${url_effective}" -o /dev/null)
        echo $curl_resp | tee -a curl_html_resp
        if [[ $curl_resp = 404* ]]; then
            rm "$html_output_dir"/"$filename"
        fi
    fi
}
```

- Tool zur Filterung sensibler Informationen <https://github.com/tomnomnom/gf>

## DEMO

Shell: Filterung HTML und JS Quellen mit gf



# Spider Processing und Visualisierung

## DEMO

Jupyter: Verarbeitung und Visualisierung von URL-Daten



# Inhaltsübersicht

- 1 Wer bin ich?
- 2 Impuls
- 3 Ablauf
- 4 Subdomains
- 5 Portscans
- 6 Schwachstellenscans
- 7 URLs
- 8 Wie WAFs vermeiden?**
- 9 Kern und Ausblick



# Parallelisierung von Zielen

- viele Ziele gleichzeitig und iterativ betrachten
- <https://github.com/tomnomnom/meg> scannt viele URL-Pfade und viele Hosts iterativ

## DEMO

Shell: Scan mit meg



# Parallelisierung der Angriffsmaschinen

- mit Axiom können schnell und elegant Cloud-Instanzen zu Hacking-Cluster orchestriert werden
  - -> hohe Parallelität mit sehr vielen IP-Adressen
- <https://github.com/pry0cc/axiom>

## DEMO

Shell: Orchestration einer Axiom Fleet und Distributed Scanning <3 <3 <3



# Inhaltsübersicht

- 1 Wer bin ich?
- 2 Impuls
- 3 Ablauf
- 4 Subdomains
- 5 Portscans
- 6 Schwachstellenscans
- 7 URLs
- 8 Wie WAFs vermeiden?
- 9 Kern und Ausblick**



# Kernbotschaft

- BugBounty/VDP kann schnell zu BigData-Problem werden
  - Datenverarbeitung
  - schnellstmögliche Datenfilterung und -visualisierung
- Automatisiere so viel wie möglich
- Skillset + OpenSource > Propriäre Scan-Tools



# Ausblick

- Integration und Automatisierung in einheitliche Tool-Pipeline
- Einsatz von GNNs (Graph Neural Networks)