Fast & Effective Recon

unvalidor

Recon v1

Hey!
This little checklist is taken from the book Bug Bounty Bootcamp by Vickie Li and a little bit of my own knowledge. I hope it helps you.

- Ways to contact me :
  - t.me/unvalidor
  - x.com/unvalidor
  - github.com/unvalidor

Dedicated to the one who knows. 🌙

---

1. Manual Scrolling

- Scrolling through the target (opening links, pages)
- Gaining access to all properties that users can utilize
- Accessing properties that are not commonly used
- Creating accounts with all permissions
- Observing how the application appears for every user type
- Identifying input points
- Understanding how different users interact with each other
- **Attack surface idea:** all points that an attacker could try to exploit on the target

---

2. Google Dorking

Examples:

1. `site:example.com`
2. `inurl:"course/admin.php"`
3. `intitle:"index of"`
4. `link:"https://wikipedia.org/redos"` → finds all sites that have this link on the page
5. `filetype:pdf`
6. `*` → (e.g., `start * end` to find how to hack `*` using Google)
7. `"how to hack"` → searches for this exact phrase
8. `how to hack` → searches for pages containing the words *how*, *to*, and *hack*
9. `(sqli | sql injection)`
10. `"how to hack" (reddit.com | facebook.com)`
11. `"how to port scanning" -nmap` → excludes results containing *nmap*

Subdomains

- `site:*.example.com`

- Test: `site:example.com inurl:app/kibana`

Finding Buckets

- `site:s3.amazonaws.com company_name`
- `site:example.com ext:php`
- `site:example.com ext:log`
- `site:example.com ext:txt password` → searches for text files potentially containing passwords

---

3. Scope Discovery

- Determine which domains and subdomains are in scope
- Identify IP addresses, domains, and subdomains
- Enumerate assets

---

4. Whois and Reverse Whois

```
whois site.com
```

Reverse Whois:

- https://viewdns.info/reversewhois
- Find domains registered with a specific email, company name, address, phone number, etc.

---

5. IP Address

```
nslookup facebook.com
```

Reverse IP Lookup:

- https://viewdns.info/ (Provides a reverse IP lookup tool)

Finding IP Range:

```
whois <ip_address>          # Check the 'NetRange' field
whois -h whois.cymru.com "157.242.2.20"  # Maps to ASN? Yes
whois -h whois.cymru.com "157.242.2.25"  # Maps to ASN? Yes
whois -h whois.cymru.com "157.242.2.27"  # Maps to ASN? No
```

---

6. Certificate Parsing (Finding More Hosts)

- **Tools:** `crt.sh`, `Censys`, `Cert Spotter`
- **Example:** For `facebook.com`, check the **Subject Alternative Name (SAN)** field in its SSL certificates to find other associated hosts.
- **Query:** https://crt.sh/?q=facebook.com&output=json

7. Subdomain Enumeration

- **Tools:** `Amass`, `Sublist3r`
- **Wordlist: Seclists**
- **Custom Wordlist:** https://github.com/assetnote/commonspeak2/
- **Remove Duplicates:**

```
sort -u f1.txt f2.txt > sorted_unique_subdomains.txt
```

Gobuster DNS Command:

```
gobuster dns -d target_domain -w wordlist.txt
```

- **Pro Tip:** Also test for subdomains of the subdomains you have already discovered (multi-level subdomain enumeration).

---

8. Service Enumeration

- **Active Scanning (Open Ports):**

```
nmap site.com
```

- **Passive Intelligence:**
  Use tools like `Shodan`, `Censys`, and `Project Sonar` to discover services without directly probing the target.`

---

9. Directory Brute Forcing

Common HTTP Status Codes

Status Code	Meaning	Notes
200	OK – File or directory found	Valuable hit
403	Forbidden – Access denied	Try bypass techniques
404	Not Found	No result
301	Moved Permanently – Redirect	Worth noting, follow redirect

Tools & Usage Examples

FFUF (Fast web fuzzer)

```
ffuf -u https://target.com/FUZZ -w /path/to/wordlist.txt -t 50 -mc 200,403
```

- `-u`: Target URL with `FUZZ` placeholder
- `-w`: Wordlist path

- `-t`: Number of threads
- `-mc`: Match HTTP status codes (e.g., 200 OK, 403 Forbidden)

Dirsearch

```
python3 dirsearch.py -u https://target.com -e php,txt,log -w /path/to/wordlist.txt
```

- `-e`: Extensions to try
- `-w`: Wordlist
- `-u`: Target URL

Gobuster

```
gobuster dir -u https://target.com -w /path/to/wordlist.txt -x php,txt,log -t 50 -s 200,403
```

- `-x`: File extensions
- `-s`: Status codes to show
- `-t`: Threads

EyeWitness

```
EyeWitness -f urls.txt --web --no-prompt
```

- `-f`: File containing URLs
- `--web`: Web screenshot mode
- `--no-prompt`: Runs without user interaction

Snapper

```
snapper -i urls.txt -o screenshots/
```

- `-i`: Input file with URLs
- `-o`: Output directory for screenshots

Pro Tips

- Use `SecLists` for high-quality wordlists: `/usr/share/seclists/Discovery/Web-Content/`
- Try extensions like `.bak`, `.zip`, `.old`, `.conf`, `.env`
- Use proxy tools (e.g., Burp Suite) to monitor and replay requests
- Combine with Google Dorking to identify likely endpoints before brute forcing
- Respect rate limits and avoid detection by adjusting thread count and delay

---

10. Spidering

Tools & Practical Usage

Burp Suite – Crawler (Professional Edition)

Purpose: GUI-based spidering with session handling, form parsing, and scope control.

Steps:

1. **Proxy Setup:** Configure your browser to route traffic through Burp.
2. **Login First:** Manually log in to capture session cookies.
3. **Target Tab → Right-click domain → "Spider this host"**
4. **Configure Scope:** Limit to target domain only.
5. **Enable Form Submission:** Burp can auto-fill and submit forms.
6. **Review Results:** Found URLs appear in **Target → Site Map.**

Export URLs:

```
Target → Site Map → Right-click → "Copy URLs" or "Save as CSV"
```

OWASP ZAP – Spider + AJAX Spider

Purpose: Open-source alternative to Burp with both traditional and JavaScript-based crawling.

Usage:

1. Launch ZAP and set target URL.
2. Use **Spider** for static crawling.
3. Use **AJAX Spider** for dynamic JS-heavy sites.
4. Review results under **Sites** tree.

Command-line:

```
zap-cli spider https://target.com
zap-cli ajax-spider https://target.com
```

Hakrawler

Purpose: Fast CLI crawler for automation and CI pipelines.

Usage:

```
echo https://target.com | hakrawler -depth 3 -subs -js -robots -sitemap -threads 10
```

- `-depth`: Crawl depth
- `-subs`: Include subdomains
- `-js`: Parse JavaScript files
- `-robots`, `-sitemap`: Include robots.txt and sitemap.xml

Photon

Purpose: Crawls and extracts URLs, files, secrets, and endpoints.

Usage:

```
python3 photon.py -u https://target.com -t 3 -o output/
```

- `-t`: Threads
- `-o`: Output directory

LinkFinder

Purpose: Extracts endpoints from JavaScript files.

Usage:

```
python3 linkfinder.py -i https://target.com/app.js -o cli
```

Pro Tips

- Always crawl **after authentication** to access protected areas.
- Use **session cookies** or **headers** in CLI tools to mimic logged-in users.
- Export discovered URLs for brute forcing, fuzzing, or vulnerability scanning.
- Combine spidering with **JS analysis** to uncover hidden API endpoints.
- Respect robots.txt—but don't rely on it to hide sensitive paths.

---

11. Third-Party Hosting (Cloud Storage)

Google Dorking for Buckets:

- `site:s3.amazonaws.com company_name`
- `site:amazonaws.com company_name`

Bucket Search Engine:

- `https://buckets.grayhatwarfare.com/` → Search for company, application, or project names.

Discovery Tools:

- `https://github.com/nahamsec/lazys3/`
- `https://github.com/eth0izzle/bucket-stream`

Interacting with Found Buckets (AWS CLI):

```
# Install the AWS CLI
pip install awscli

# List the contents of a bucket
aws s3 ls s3://bucket_name/

# Copy a file to the bucket
aws s3 cp file.txt s3://bucket_name/

# Remove a file from the bucket
aws s3 rm s3://bucket_name/file.txt
```

---

12. GitHub Reconnaissance

Manual Techniques:

- Search for target names (company, project, product) in the GitHub search bar.
- Check the GitHub profiles of company developers.
- Examine code repositories, issues, and commit histories for exposed secrets or sensitive information.

Automated Tools:

- **GitRob:** `https://github.com/michenriksen/gitrob/` (Scans GitHub repositories for sensitive files)
- **truffleHog:** `https://github.com/trufflesecurity/truffleHog/` (Scans Git repositories for high-entropy strings and secrets)

---

13. OSINT (Open-Source Intelligence)

Tools and Resources:

- **PasteHunter:** `https://github.com/kevthehermit/PasteHunter/` (Monitors paste sites for sensitive information)
- **Wayback Machine (Internet Archive):** `https://archive.org/web/` (Views historical versions of websites)

Endpoint Extraction:

- Extract endpoints and URLs from the Wayback Machine:
  - **waybackurls:** `https://github.com/tomnomnom/waybackurls` (A tool to fetch all URLs saved by the Wayback Machine for a domain)

---

14. Tech Stack Fingerprinting

Vulnerability Database:

- **CVE Database:** `https://cve.mitre.org/cve/search_cve_list.html` (Search for Common Vulnerabilities and Exposures)

Fingerprinting Techniques:

- **Service Version Detection:**

```
nmap -sV scanme.nmap.org
```

- **Source Code Analysis:**
  - View page source (`Ctrl+U` in browsers)
  - Search for keywords like: `powered by`, `built with`, `running`

Fingerprinting Tools:

- **Wappalyzer** (Browser extension)
- **BuiltWith:** `https://builtwith.com/` (Online technology profiler)

Note:

Focus on **what is important to the company** and **prioritize those targets.**