



[< Go to the original](#)



Bug Bounty Methodology for Finding Bugs Easily 🐞💰

Welcome, bug bounty hunters! 🕵️ Whether you're just starting out or looking to sharpen your methodology, this guide will help you...



Vipul Sonule

Follow

👤 InfoSec Write-ups androidstudio ~8 min read ·
August 8, 2025 (Updated: August 8, 2025) · Free: No

Welcome, bug bounty hunters! 🕵️ Whether you're just starting out or looking to sharpen your methodology, this guide will help you streamline your bug hunting process and maximize your chances of discovering high-impact vulnerabilities. Let's break down a structured and battle-tested methodology that top hunters use to uncover bugs efficiently. 💡

1. Introduction to Bug Bounties

Bug bounties are programs offered by companies that allow hackers to legally test their applications and systems for vulnerabilities. If you find a valid bug, you get paid. It's a win-win! 🏆

Popular platforms to get started:

- [HackerOne](#)
- [Bugcrowd](#)
- [Synack](#)
- [YesWeHack](#)
- [Intigriti](#)

2. Setting Up Your Environment ⚙️

Before diving in, set up a secure and organized environment:

Tools to Install:

- Burp Suite: [Download](#)
- ZAP Proxy: [Download](#)
- Postman: [Download](#)
- Amass: [GitHub](#)
- Subfinder: [GitHub](#)
- Nuclei: [GitHub](#)
- ffuf: [GitHub](#)
- dirsearch: [GitHub](#)

Copy

```
~/bugbounty/  
├── target1/  
│   ├── recon/  
│   ├── screenshots/  
│   └── notes.md
```

Use a VPN and avoid running tools from your home IP.

3. Choosing a Target 🎯

Pick a target based on:

- Program scope
- Reward structure
- Reputation for responsiveness
- Technology stack (stick to what you know early on)

Use platforms' filters like "Web apps with high payouts and low submission counts."

4. Reconnaissance (Recon) 🔍

Recon is about gathering as much information as possible about your target.

🔍 Subdomain Enumeration:

Use:

Copy

Screenshots: Visual Recon Using Aquatone & httpx

Once you've discovered live hosts, the next step is to **visually understand what these hosts look like**. This isn't just for aesthetics — it helps identify interesting pages like admin panels, login screens, staging environments, and exposed dashboards that could be missed in pure text recon.

Let's dive into two powerful tools for this purpose: **Aquatone** and **httpx**.

♦ What Is Aquatone?

Aquatone is a tool for **domain flyovers**. It takes a list of domains, visits each one in a headless browser, and captures a screenshot.

Installation:

Copy

```
G0111MODULE=on go get -u github.com/michenriksen/aquatone
```

Usage:

You usually combine it with other tools like **Amass**, **Subfinder**, or **Assetfinder**, but here's a basic example:

Copy

```
cat domains.txt | aquatone
```


✅ This combo ensures you only screenshot **live domains**.

📸 Why Screenshots Matter in Bug Bounties

- You may visually spot **juicy targets** like `/admin` , `/dev` , `/phpmyadmin` , or **unprotected internal panels**.
- Screenshot data helps during **report writing**, showing clear proof-of-concept (PoC).
- Often, **error pages**, **directory listings**, or **debugging tools** appear only when viewed in a browser — text-based recon misses these.

🌐 DNS Bruteforcing: Unearth Hidden Subdomains

When you're deep into recon mode, sometimes the usual subdomains aren't enough. That's where DNS bruteforcing comes into play. Think of it as throwing a wide net — you're trying every possible combination of subdomains to see what sticks.

🔧 Tool of Choice: `dnsgen`

`dnsgen` is a powerful tool that helps you **generate permutations** of discovered subdomains — this increases the chances of finding dev, staging, admin, or legacy servers hiding behind unusual names.

🔧 How to Use:


First, install it:

Copy

```
pip install dnsgen
```

Copy

```
cat subdomains.txt | dnsgen - > permutations.txt
```

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. The terminal shows a command being executed: `$ echo 'partner-mail.cyclops.corp.yahoo.com' | dnsgen -`. Below the command, a list of 18 domain permutations is displayed, each on a new line. The domains are variations of the original domain, using different subdomains and TLDs.

```
$ echo 'partner-mail.cyclops.corp.yahoo.com' | dnsgen -  
  
engine.cyclops.corp.yahoo.com  
partner-stage.cyclops.corp.yahoo.com  
partner-mail.cyclopass.corp.yahoo.com  
partner-mail.resetdatacyclops.corp.yahoo.com  
partner-mail.cyclops.corpreregion.yahoo.com  
partner-mail.cyclops.corpv1.yahoo.com  
partner-mail.administratorcyclops.corp.yahoo.com  
reset.cyclops.corp.yahoo.com  
partner-mail.cyclops.chef.corp.yahoo.com  
auth-partner-mail.cyclops.corp.yahoo.com  
partner-mail.cyclops.corpsystem.yahoo.com  
partner-mail.vpn.corp.yahoo.com  
partner-mail.reset.cyclops.corp.yahoo.com  
cyclops-mail.cyclops.corp.yahoo.com  
partner-mail.northamerica.cyclops.corp.yahoo.com  
partner-mail.cyclops.gateway.corp.yahoo.com  
partner-mail.priv.cyclops.corp.yahoo.com
```

Now combine it with [massdns](#) or [puredns](#) to resolve those permutations:

Copy

```
puredns resolve permutations.txt -r resolvers.txt --wildcard-tests
```



Why It Matters:

- These often have **weaker security** or **default creds**.
- They're **not indexed** by search engines or **linked** anywhere — bruteforce is your only way in.

👁️ **Asset Discovery Tools: The Bug Hunter's Radar** 🗺️

Before you can hack it, you have to **find it**. That's where **asset discovery** comes in. Think of it like scanning the entire galaxy for planets 🌍 — only, in our case, we're looking for subdomains, IPs, apps, and forgotten services.

Let's dive into two powerful tools every bug bounty hunter should keep in their arsenal:

♦ **1. Chaos Project by ProjectDiscovery**

The **Chaos Project** is a goldmine maintained by ProjectDiscovery. It contains **regularly updated DNS datasets** for **public bug bounty programs** — all in one place.

💻 How to Use:

Copy

```
# Clone the repo
git clone https://github.com/projectdiscovery/public-bugbounty-pro
# Or use nuclei templates to automate it:
nuclei -t templates/ -target https://chaos.projectdiscovery.io
```

📌 Why Chaos Rocks:

- Easy access to subdomains of hundreds of live bounty programs.

- Integrated easily with tools like **httpx**, **dnsx**, and **nuclei**.

✓ Pro Tip:

Combine Chaos with `httpx` to check which subdomains are live:

Copy

```
cat chaos.txt | httpx -silent -status-code -title
```

♦ 2. crt.sh

`crt.sh` is a **certificate transparency log search engine** that shows you all the SSL/TLS certificates issued for a domain — even for subdomains that were never meant to be public 🙄.

🔍 Why It's Useful:

- Companies often request SSL certs for **internal** subdomains like `staging.domain.com` or `dev-api.domain.com`.
- `crt.sh` logs them — and **you can query them freely**.

🔧 Quick Recon Example:

Search for all subdomains of `example.com` :

Copy

```
https://crt.sh/?q=%25.example.com
```

🧠 Combine With Tools:

5. Endpoint Enumeration 🌐

Find hidden files, folders, and API endpoints.

Tools:

Copy

```
ffuf -u https://example.com/FUZZ -w wordlist.txt -t 50
```

Use wordlists from [SecLists](#).

6. Parameter Discovery 🛠️

Parameters = Potential Bugs! 💣

If you're not paying attention to request parameters, you're missing out on **some of the juiciest bugs** out there. Whether it's an `id` , `page` , `search` , or even something weird like `next_url` — parameters can lead to:

- XSS (Cross-Site Scripting) 🧪
- IDOR (Insecure Direct Object Reference) 🕵️
- SQLi (SQL Injection) 💾
- Open Redirects 🚀
- SSRF (Server-Side Request Forgery) 🌐

So... how do we find them? Let's dive in 📖

Tools for Parameter Discovery

Arjun

Arjun is an amazing tool that automatically detects valid HTTP GET and POST parameters for a given URL by using a huge wordlist.

👉 GitHub: <https://github.com/s0md3v/Arjun>  Command:

Copy

```
python3 arjun.py -u https://example.com/page --get
```

```
Arjun v2.1.5
```

```
[*] Probing the target for stability
[*] Analysing HTTP response for anomalies
[*] Analysing HTTP response for potential parameter names
[*] Logicforcing the URL endpoint
[✓] name: url, factor: redirection
[✓] name: filter, factor: http code
[✓] name: page, factor: http body
```

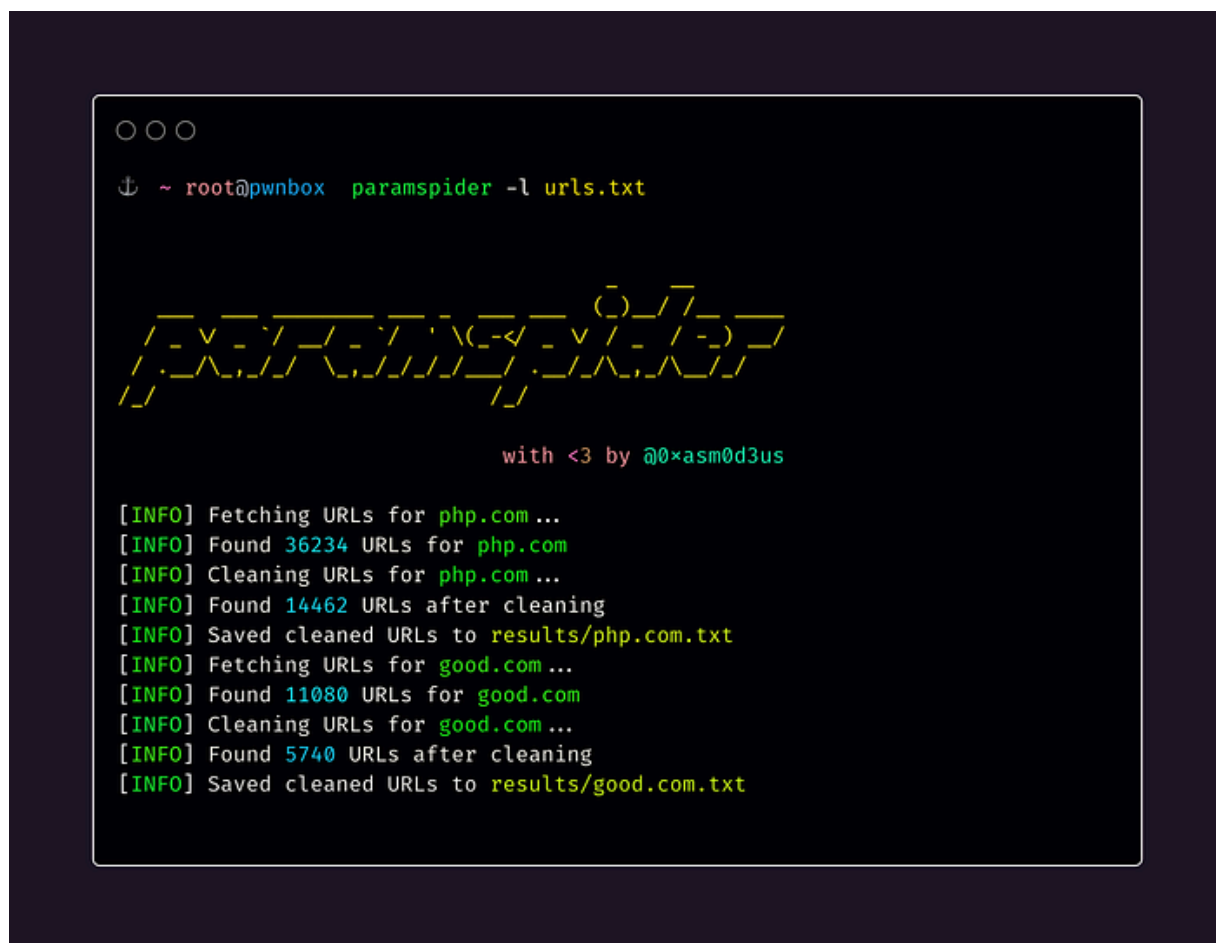
Arjun digs deep and can find parameters others often miss.

ParamSpider

ParamSpider is a tool that scrapes known URLs and endpoints from **Wayback Machine**, **Common Crawl**, and similar archives — revealing hidden or forgotten parameters.

👉 GitHub: <https://github.com/devanshbatham/ParamSpider> 
Command:

Copy



Once you find potential parameters, start testing them manually or automate them with **XSS scanners** and **fuzzers**

7. Vulnerability Discovery

Alright — recon is done, endpoints are mapped, and now the real fun begins: **finding the actual bugs**. This phase is like playing hide-and-seek with a system that *doesn't want* to be found. 😊 Here's how seasoned bug hunters go about it:

7.1 Start With Common Vulnerabilities

Before going all *Matrix mode*, start by checking for common vulnerabilities — especially low-hanging fruits that many developers

Freedium

- **XSS (Cross-Site Scripting)**: Inject `<script>alert(1)</script>` into parameters, headers, or even obscure areas like PDF previewers or 404 error messages. 👉 Use: [XSS Hunter](#) or [Dalfox](#)
- **IDOR (Insecure Direct Object Reference)**: Change resource IDs like `user_id=123` to `user_id=124` and see if you get access to someone else's data. 👉 Use: Burp Intruder or a script with [ffuf](#)
- **CSRF (Cross-Site Request Forgery)**: Try crafting requests that act on behalf of the user. If the action succeeds without a CSRF token, that's a win.
- **Broken Access Control**: Remove JWT tokens, change roles, or access admin panels via direct paths (`/admin` , `/superuser` , etc).

7.2 Understand Application Logic

Sometimes, bugs hide in the **logic** rather than code. These are trickier but high in impact:

- **Price Manipulation**: Change the price of an item in a shopping cart during checkout.
- **Coupon Abuse**: Try reusing or stacking discount codes beyond the intended use.
- **Workflow Bypass**: Skip steps in a multi-step process (like email verification or OTP).
- **Timing-Based Bugs**: Submit a form multiple times quickly to see if rate-limiting breaks (think ticket systems or payment duplication).

7.3 Think Like a Malicious Hacker

Most security issues arise because developers *didn't think like attackers*. Your job is to be malicious *ethically*. 😈

- Ask: *What if I wasn't supposed to be here?*

~~ASK: Can I act as another user without permission.~~

- Ask: *Can I delay, flood, or confuse the app by spamming requests?*

Use tools like:

- Burp Suite
- ZAP
- Postman for fuzzing APIs
- httpx and nuclei for automation

7.4 Train Your Eyes & Mind

The more bugs you find, the better your instinct gets. Bug hunting becomes less about tools and more about experience.

- Keep reading write-ups on HackerOne Hacktivity, Bugcrowd Disclosures, and Medium blogs.
- Watch live hacking streams and CTF replays to learn new tricks.
- Join forums like r/bugbounty, Infosec Writeups, and Discord groups.

7.5 Document Everything

Even if the bug isn't critical, take notes. You'll be surprised how often you'll revisit a weak finding later and turn it into a P1!

Create a Google Sheet or use Notion to track:

- URLs & parameters tested
- Payloads that worked/didn't work
- Screenshots or Burp Suite exports
- Response codes and behavior changes

8. Manual Testing Techniques

Manual testing = real impact.

Check for:

- Logic flaws
- Race conditions
- Rate limiting bypass
- Session management issues

Example:

Check password reset token reuse, or escalate roles via poorly validated APIs.

Use Burp Suite to intercept and replay requests.

9. Reporting the Bug

A good report increases payout odds!

Structure:

- Summary
- Steps to Reproduce
- Impact
- Proof of Concept
- Suggested Fix

Report Writing Tip:

Use clear, concise language. Include screenshots, video proof, and curl commands.

10. Post-Report Reflection 🏆

After submitting:

- Track your reports.
- Learn from accepted or rejected bugs.
- Note interesting behaviors for future use.

Keep a bug journal!

11. Tools of the Trade 🛠️

Here's a quick list of top tools:

Tool Use
Burp Suite Interception & analysis
Nuclei Vulnerability scanning
Subfinder Subdomain discovery
ffuf Directory fuzzing
Arjun Parameter discovery
Dalfox XSS finding
httpx Probing URLs
Waybackurls Discover old endpoints

12. Bonus Tips & Tricks 💡

- 🧠 Focus on logic bugs — fewer people check these.
- 🔄 Repeat testing after new deployments.
- 🧱 Test WAF evasion techniques.
- 🔍 Always test for IDORs in APIs.
- 🐞 Try command injection on uploaders.

13. Recommended Resources 📖

Blogs & Guides:

- [Bug Bounty Notes](#)
- [Hacker101](#)
- [PayloadAllTheThings](#)

Twitter/X Accounts to Follow:

- @ngalog (XSS master)
- @TomNomNom (Recon tools)
- @arkadiyt (Waybackurls)
- @Hacker0x01

YouTube:

- [STÖK](#)
- [NahamSec](#)

👋 Stay Connected

If you enjoyed this guide and want more practical tutorials, recon checklists, and hacker strategies, stay in touch:

- 📧 FREE Newsletter: thehackerslog.substack.com
- 📷 Twitter (X): @VipulSonule
- 👤 LinkedIn: [Vipul Sonule](#)

