Symbiosis Skills And Professional University

Bug Bounty & Web Security Automation: My Internship Journey

Contents of this template

You can delete this slide when you're done editing the presentation

Fonts	To view this template correctly in PowerPoint, download and install the fonts we used
Used and alternative resources	An assortment of graphic resources that are suitable for use in this presentation
Thanks slide	You must keep it so that proper credits for our design are given
Colors	All the colors used in this presentation
Icons and infographic resources	These can be used in the template, and their size and color can be edited
Editable presentation theme	You can edit the master slides easily. For more info, **click here**

For more info:
SLIDESGO | BLOG | FAQs

You can visit our sister projects:
FREEPIK | FLATICON | STORYSET | WEPIK | VIDEVO

Table of contents

01

Introduction

02

Subdomain Enumeration

03

HTTP Probe & Wayback Archive

04

Subdomain Takeover Testing

05

JS & API Key Analysis

06

Port & Service Discovery

Table of contents

07

Vulnerabilities CVE-CVSS

08

Automation Scripting

09

Screenshots & PoC Evidence

10

Outcome & Future Scope

Internship Overview

01

BLOGGERSCON VISION PVT LTD

Project Title:
Security Researcher Intern — Hands-on Vulnerability Discovery & Recon Automation

Organization & Mentorship:
Conducted under BloggersCon Vision Pvt. Ltd., Pune
Guided by CTO Mr. Abhishek Bhaskar and Faculty Mentor Prof. Parul Bhanarkar at SSPU

Duration:
2nd June 2025 to 31st July 2025 — 8 Weeks

Objective:
To gain practical, real-world experience in cybersecurity through structured vulnerability analysis, bug bounty simulation, and automation of reconnaissance workflows. The project focused on web application testing, cloud security evaluation, CVE research, and ethical vulnerability disclosure via VDP platforms.

Understanding Web Vulnerabilities – PortSwigger Labs

Goal:

Master OWASP Top 10 vulnerabilities via hands-on exploitation of API and server-side flaws.

Labs Covered:

API Testing

- Recon & Documentation Analysis
- Endpoint Enumeration & Hidden Parameter Discovery
- Mass Assignment & Server-Side Parameter Pollution
- Exploitation through Automated Tooling
- API Hardening Techniques

Server-Side Exploits

- Path Traversal, Access Control, SSRF
- Authentication Bypass & IDOR
- OS Command Injection
- SQL Injection (Classic, Union-based)
- File Upload Vulnerabilities

Protocol Security & Bug Bounty Fundamentals

03

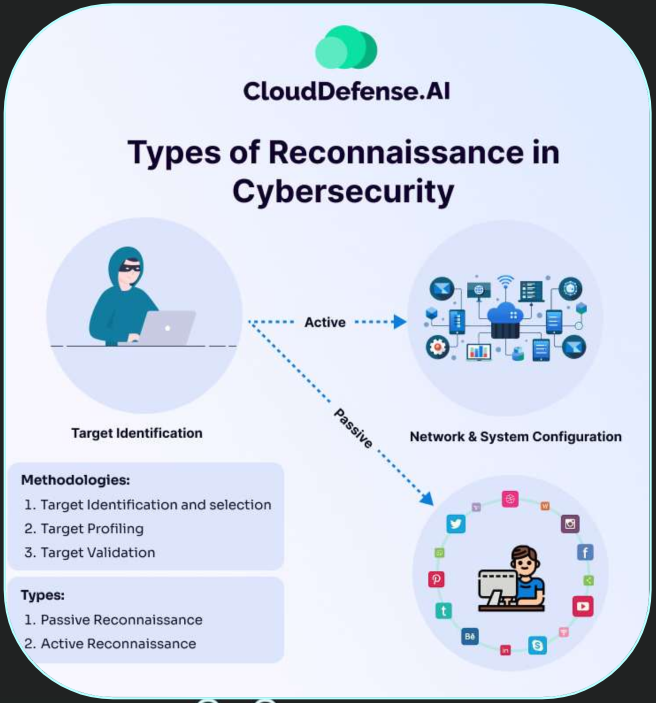
What is an RFC?

- RFC = Request for Comments
- Published by the IETF (Internet Engineering Task Force)
- Defines standards, protocols, and implementation details for the Internet
- Once published, RFCs are never modified—updates come via new RFCs
- First RFC published in April 1969 by Steve Crocker (RFC 1)

RFC #	Topic	Date
791	IPv4	Sep 1981
793	TCP	Sep 1981
1035	DNS Spec	Nov 1987
2616	HTTP/1.1	Jun 1999
5246	TLS v1.2	Aug 2008

Reconnaissance & Manual Vulnerability Testing

04


CloudDefense.AI
Types of Reconnaissance in Cybersecurity
Active
Passive
Target Identification
Network & System Configuration

Methodologies:

1. Target Identification and selection
2. Target Profiling
3. Target Validation

Types:

1. Passive Reconnaissance
2. Active Reconnaissance

📡 Recon Workflows

- Automated subdomain enumeration via **httpx**, **naabu**, **ffuf**, and **nmap**
- ASN & DNS mapping to identify edge assets and exposure
- JavaScript parsing with **LinkFinder**, **SecretFinder** for endpoint/secret discovery
- Infrastructure fingerprinting and tech stack profiling

💣 Manual Vulnerability Testing

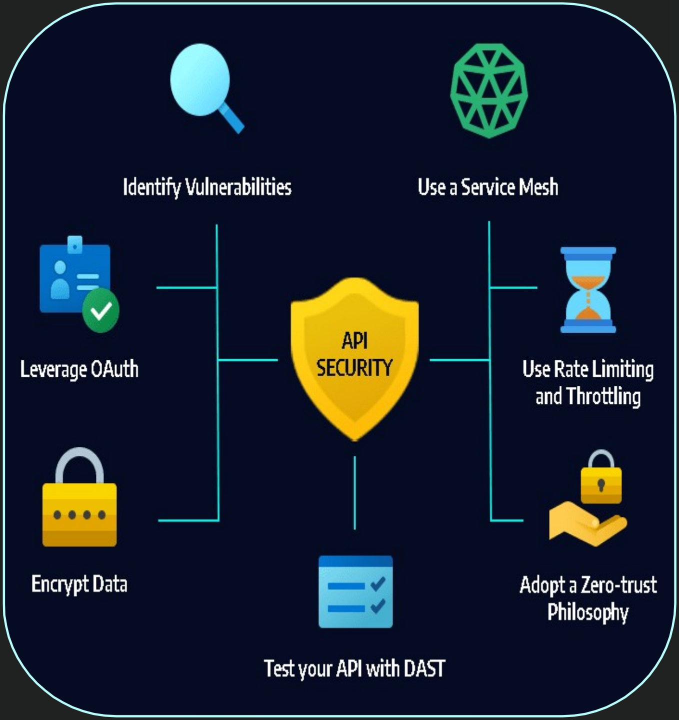
- Reflected XSS validation via crafted payloads and **DOM** inspection
- DOM analysis for insecure functions and user-controlled sinks
- Manual fuzzing of parameters & header vectors for **SSRF**, **IDOR**, etc.
- Browser **DevTools**, Burp Suite, and custom scripts for deep validation

🎯 Key Findings (Coca-Cola Target)

- Validated reflected XSS with proof of impact
- Detected hardcoded API endpoints and potential secrets in JS
- Scoped out vulnerable response flows and legacy redirect

05

API Security & Fuzzing Automation



Identify Vulnerabilities

Use a Service Mesh

Leverage OAuth

API SECURITY

Use Rate Limiting and Throttling

Encrypt Data

Adopt a Zero-trust Philosophy

Test your API with DAST

🧩 Overview

- Mapped API endpoints via JS parsing, DNS recon & ASN enumeration
- Assessed for CORS, auth bypass, verbose errors & rate-limit flaws
- Automated fuzzing with ffuf, qsreplace, nuclei across headers, params, bodies
- Logged reproducible findings for markdown-ready reporting

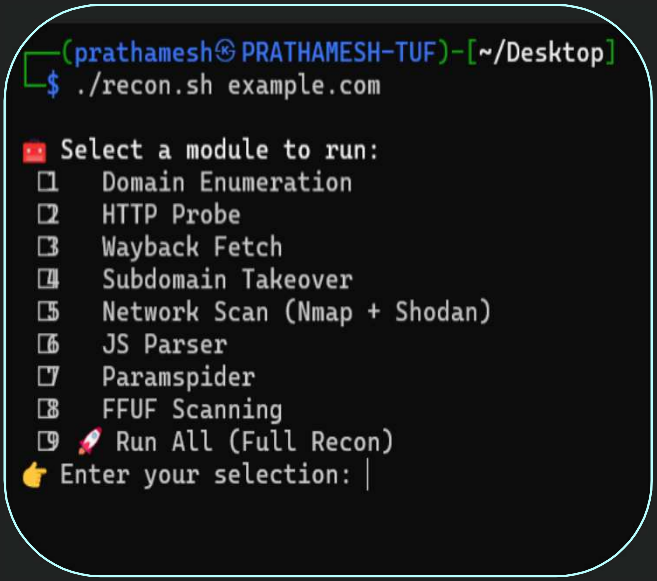
🧪 Real-World Use Case: Coca-Cola Infra

- Discovered reflected input in public-facing search API

```
ffuf -u "https://[redacted]/search?query=FUZZ" \ -w
payloads/xss.txt -t 50 -mc all -o xss_fuzz_results.json
```

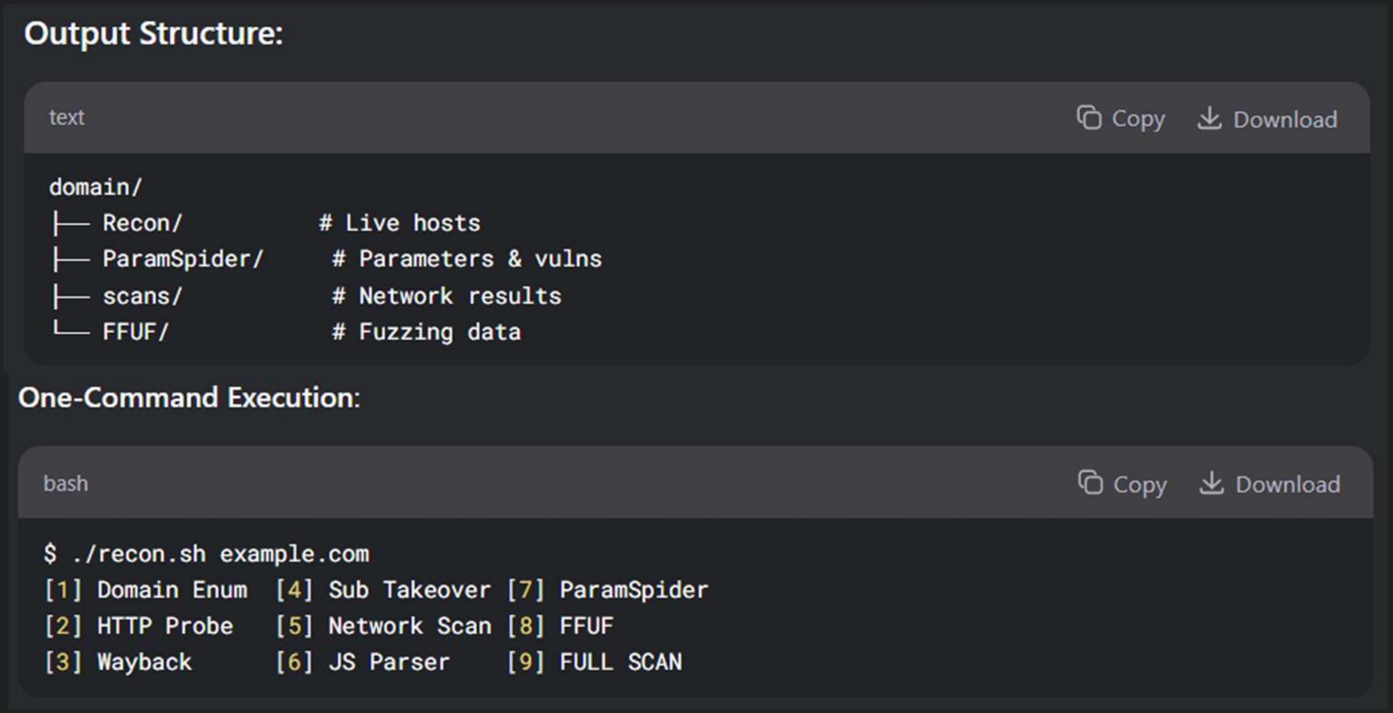
- Validated reflected XSS via manual payload testing in Burp Suite & browser tools
- Analyzed JS for insecure DOM references and hardcoded endpoints
- Documented exploit flow with PoC and responsible disclosure guidance

Recon Automation Toolkit Development



```
┌──(prathamesh㉿PRATHAMESH-TUF)-[~/Desktop]
└─$ ./recon.sh example.com

🧰 Select a module to run:
 1️⃣  Domain Enumeration
 2️⃣  HTTP Probe
 3️⃣  Wayback Fetch
 4️⃣  Subdomain Takeover
 5️⃣  Network Scan (Nmap + Shodan)
 6️⃣  JS Parser
 7️⃣  Paramspider
 8️⃣  FFUF Scanning
 9️⃣ 🚀 Run All (Full Recon)
👉 Enter your selection: 
```



Output Structure:

```
domain/
├── Recon/          # Live hosts
├── ParamSpider/     # Parameters & vulns
├── scans/           # Network results
└── FFUF/            # Fuzzing data
```

One-Command Execution:

```
$ ./recon.sh example.com
[1] Domain Enum  [4] Sub Takeover [7] ParamSpider
[2] HTTP Probe   [5] Network Scan [8] FFUF
[3] Wayback      [6] JS Parser    [9] FULL SCAN
```

JavaScript Recon & Endpoint Discovery

Overview:

- Parsed live JS files sourced via HackerOne Bug Bounty scopes
- Targeted analysis for hidden endpoints and insecure JavaScript functions

Discovered Assets:
Internal API endpoints exposed via JS:

- **/api/internal/data**
- **/auth/debug**

Vulnerable JS Patterns:
DOM manipulation functions:

- **innerHTML**
- **document.write**
- **eval**

Bug Report Reference:

During recon, I discovered a hardcoded OAuth token and vulnerable innerHTML usage in a live JS file linked from a feedback subdomain. The token enabled API-level access without user consent. Submitted responsibly to the concerned VDP via HackerOne. (Mapped to CWE-798: Use of Hard-coded Credentials)

Vulnerability Case Studies & PoCs (CVE Analysis)

06

Tools Used: Documentation, Dorking

PoC Note: Each issue confirmed via isolated endpoint, demonstrated on target test environments, and responsibly disclosed.

CVE ID	Vulnerability Type	Impact Surface	Key Insight
CVE-2025-12940	eval Injection via AWS STS	JS in AWS Response Parsing	Unsafe dynamic execution leading to remote code trigger
CVE-2024-4323	Cloudflare WAF Bypass	HTTP Payload Filtering	Crafted payloads evading security rules & heuristics
CVE-2025-0108	Auth Bypass	Palo Alto Login Workflow	Logic flaw enabling unauthorized access
CVE-2025-37228	Path Traversal	ASP.NET Handler Routing	Arbitrary file access via crafted URI paths
CVE-2025-20162	DHCP DoS	Cisco Network Appliances	Malformed packets exhausting server resources

Real Bug Findings

Critical Risks:
💰 Financial loss • 🔓 Account compromise • 📉 Reputational damage • ⚖️ Regulatory penalties

Exposed API Keys

1. Google Maps API key hardcoded in Coca-Cola Uruguay's JavaScript
2. Unrestricted access to billable services (Autocomplete/Geocoding)
3. *Impact*: Financial loss ($100s/day), data scraping risk

Authentication Bypasses

1. OTP brute-force in Coca-Cola Kombo (7 attempts to compromise)
2. CSRF in Autoklose campaign creation (GET-based state change)
3. *Impact*: Account takeover, financial fraud

Injection Vulnerabilities

1. Blind SQLi (Secondlove.nl), XSS (Coca-Cola Parts, Incode.com)
2. HTML injection (ATO chatbot)
3. *Impact*: Data theft, session hijacking, phishing

Sensitive Data Exposure

1. Constant Contact OAuth keys in public GitHub repos
2. Wayback Machine exposing historical PII endpoints
3. *Impact*: Credential compromise, regulatory violations

Security Misconfigurations

1. Missing rate limiting (Eightfold.ai promo API)
2. Subdomain takeovers (beesy.me)
3. *Impact*: Brute-force attacks, service disruption

Key Takeaways & Future Scope

1. Critical Vulnerabilities & Mitigation

- **JS/API Exposure**: Hardcoded keys enable data breaches & financial abuse (Google Maps API)
- **CRLF/SQLi**: Header injection & blind SQLi require input sanitization & parameterized queries
- **Subdomain Takeovers**: Dangling DNS records lead to phishing/malware distribution

2. Automated Reconnaissance Efficiency

- **Bash scripts** integrate tools (Subfinder/Httpx/Waybackurls) for:

✓ Subdomain enumeration
✓ Live endpoint discovery
✓ Hidden parameter mining (ParamSpider)

- **FFUF/Dirb** for critical path brute-forcing (admin panels/backups)

3. Proactive Security Integration

- **Shift-left testing**: Embed vulnerability scans in CI/CD pipelines
- **Automated monitoring**: Scripts for continuous key/subdomain exposure detection
- **Impact-driven reports**: CVSS scoring + PoC payloads for faster remediation

4. Future Growth Opportunities

- **Cloud/Web3 Expansion**:

✓ Serverless API security & misconfigured S3 buckets
✓ Smart contract auditing & Web3 wallet hijacking prevention

- **AI-Powered Defense**:

✓ ML-based risk prioritization of recon data
✓ Fuzzing frameworks (AFL) for zero-day detection

THANK YOU