Bug Hunt 101 — Recon to Report

Name: Muhammad Alameen Abdullahi
Course: Cyber Security Project
Lab Environment: Kali Linux + DVWA
Project Type: Bug Bounty Simulation
Date: 17th January 2026

---

1. Introduction

This project simulates a real-world bug bounty workflow in a controlled lab environment. The objective is to practice reconnaissance, vulnerability discovery, exploitation, and professional reporting using a deliberately vulnerable web application.

All testing was conducted ethically and strictly within scope for educational purposes.

---

2. Environment Setup

2.1 Tools Used

Kali Linux

Damn Vulnerable Web Application (DVWA)

Apache & MySQL

Nmap

Burp Suite Community

SQLMap

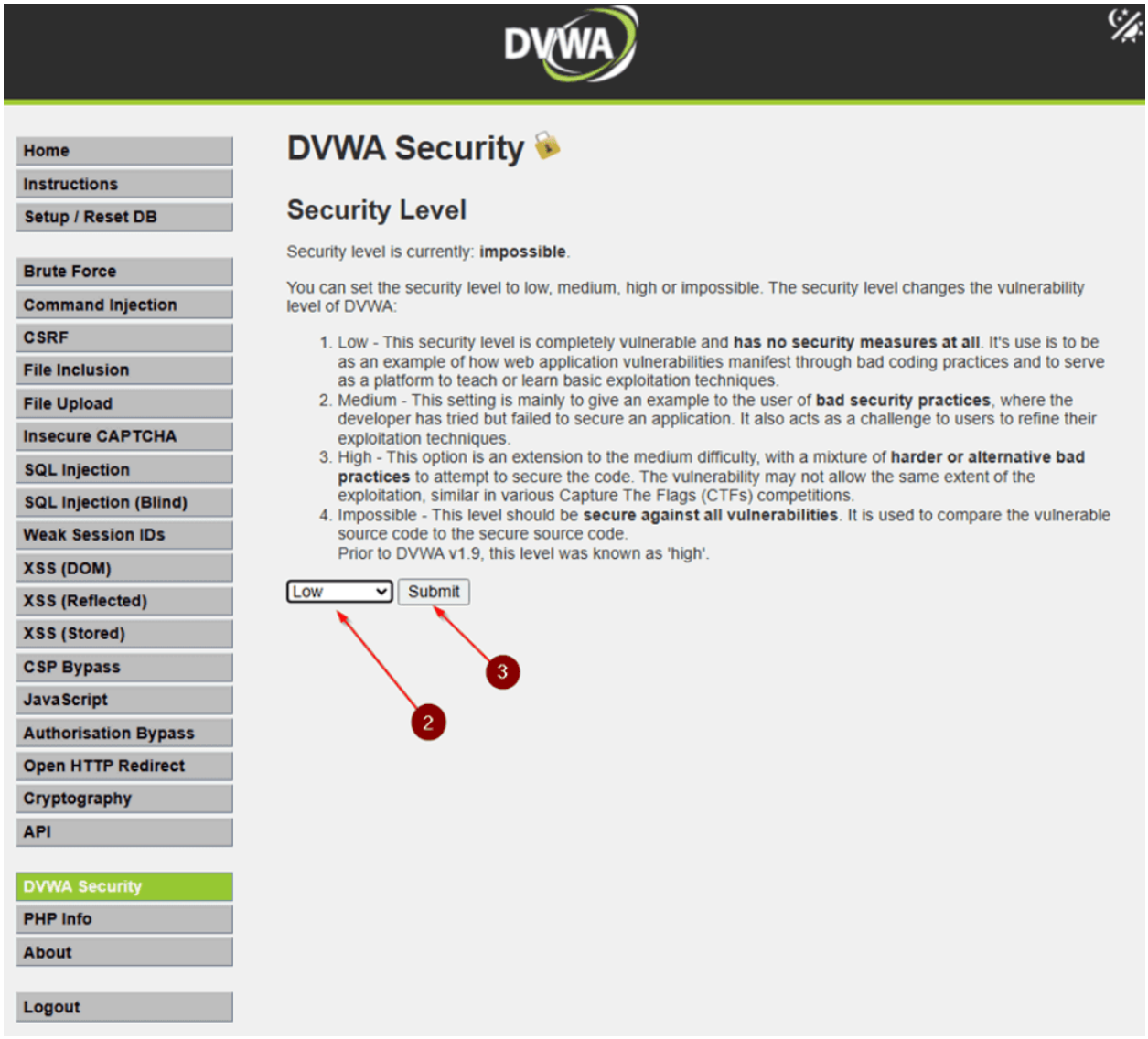
Firefox Browser

2.2 DVWA Configuration

Target URL: http://127.0.0.1/dvwa

Security Level: Low

Database: MySQL



DVWA dashboard showing security level set to Low

---

3. Reconnaissance Phase

3.1 Target Identification

The target application is hosted locally on the Kali machine.

http://127.0.0.1/dvwa

3.2 Port & Service Enumeration

Command used:

nmap -sC -sV 127.0.0.1

Findings:

Port	Service	Description
80	HTTP	Apache Web Server
3306	MySQL	Database Server

```
Starting Nmap 7.01 ( https://nmap.org ) at 2016-10-14 12:33 PDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000013s latency).
Not shown: 995 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
25/tcp   open  smtp
80/tcp   open  http
587/tcp  open  submission
5432/tcp open  postgresql

Nmap done: 1 IP address (1 host up) scanned in 1.63 seconds
```

Nmap scan output on Kali terminal

---

4. Vulnerability Discovery & Exploitation

4.1 Reflected Cross-Site Scripting (XSS)

Location: DVWA → XSS (Reflected)

Payload Used:

<script>alert(document.domain)</script>

Steps to Reproduce:

1. Navigate to DVWA → XSS (Reflected)

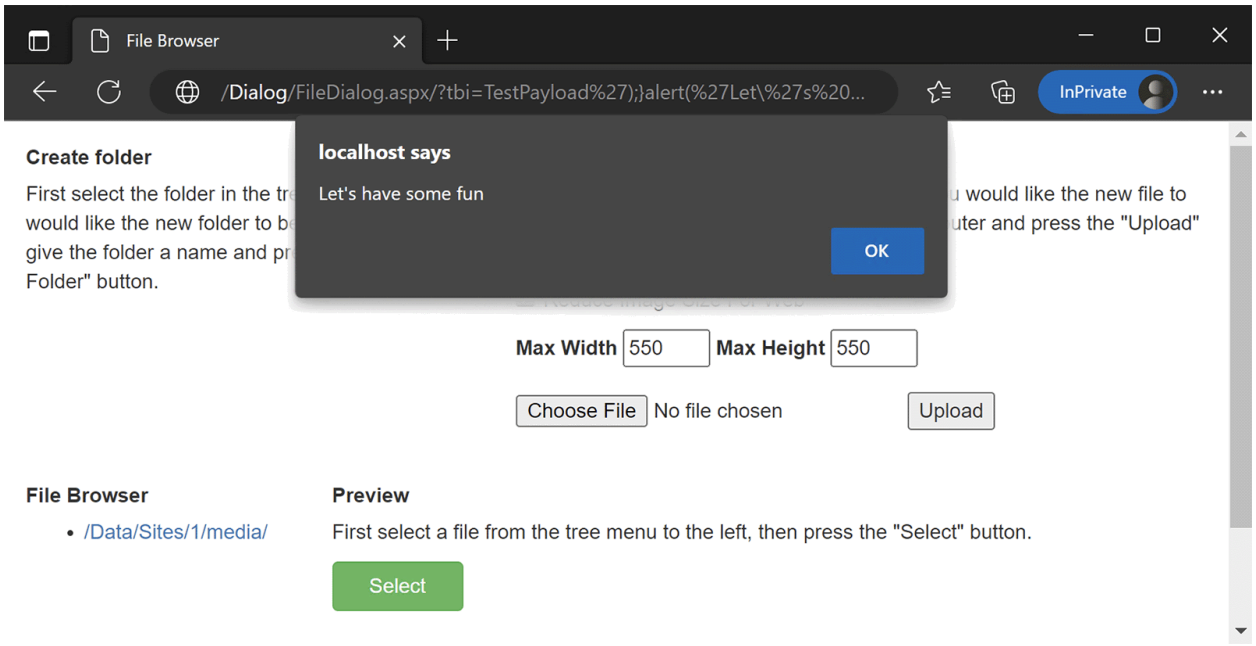
2. Enter the payload into the input field

3. Submit the form

4. JavaScript alert executes

Impact: Allows execution of arbitrary JavaScript in the victim's browser, enabling session hijacking, phishing, and credential theft.

Severity: Medium



Browser showing XSS alert popup

4.2 SQL Injection (SQLi)

Location: DVWA → SQL Injection

Manual Exploitation Payload

' OR 1=1-- -

Result: All user records are displayed, bypassing input validation.

Automated Exploitation Using SQLMap

Command:

```
sqlmap -u "http://127.0.0.1/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" \
--cookie="security=low; PHPSESSID=PASTE_YOUR_SESSION_ID" \
--risk=3 --level=5 --dbs
```

Databases Extracted:

dvwa

information_schema

Impact: Attackers can extract sensitive data, modify database content, or fully compromise the backend system.

Severity: High

SQLMap extracting database names

4.3 Insecure Direct Object Reference (IDOR)

Technique: Manual modification of numeric object identifiers.

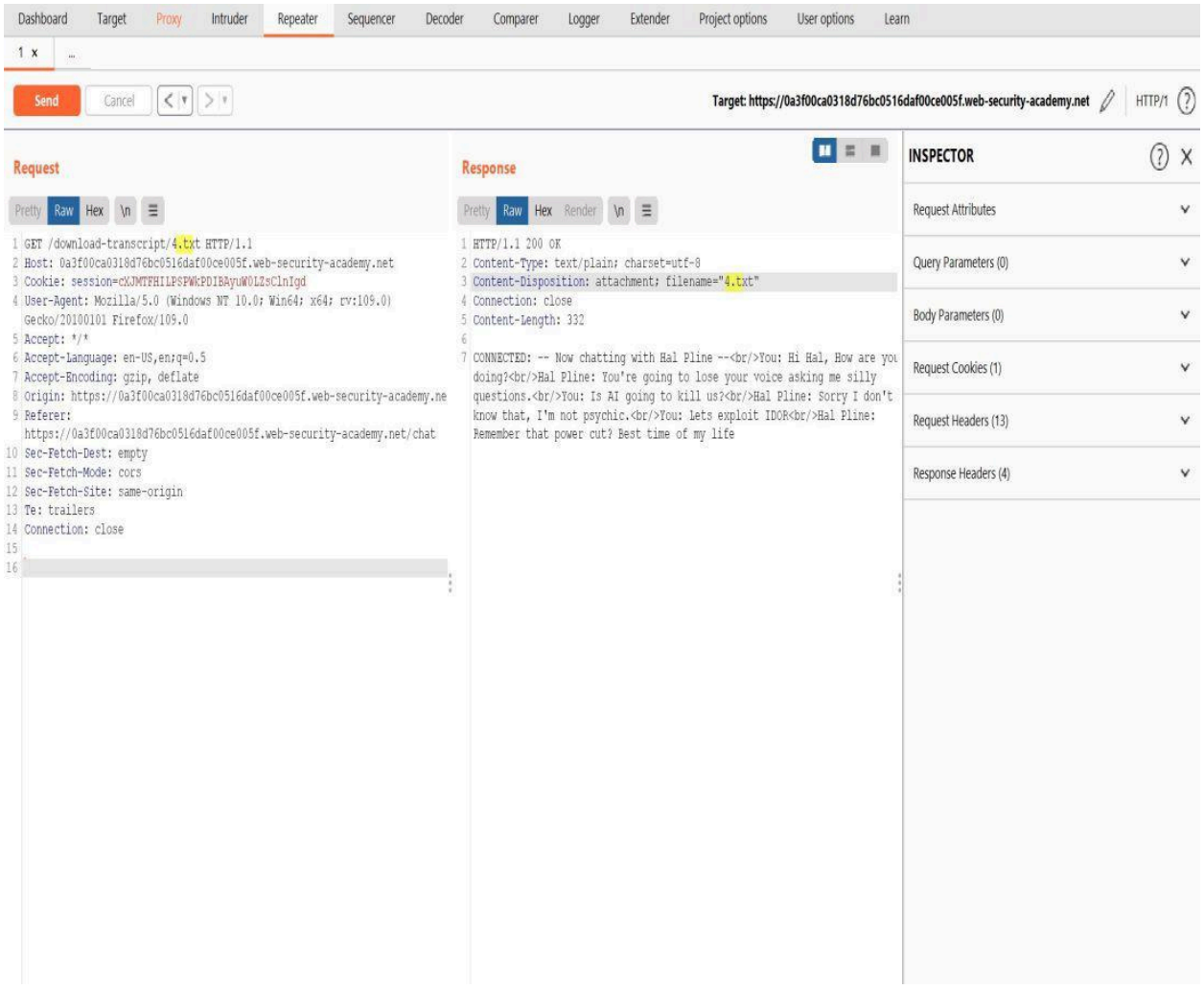
Example:

?id=1 → ?id=2

Result: Unauthorized access to other users' information.

Impact: Violates confidentiality and allows exposure of sensitive user data.

Severity: Medium



Changed user data after ID manipulatio

---

5. Burp Suite Analysis

Burp Suite was used to intercept and analyze HTTP requests.

Steps:

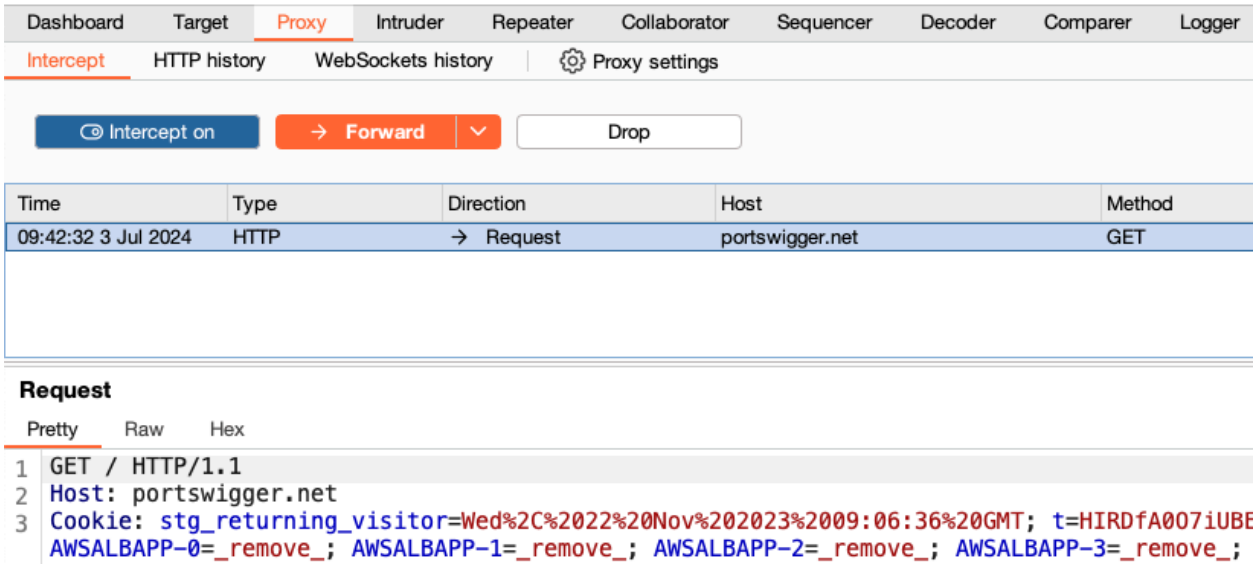
1. Configure browser proxy to Burp

2. Enable Intercept

3. Capture DVWA request

4. Modify parameters and forward request

Impact: Demonstrates how attackers manipulate client-server communication.



Burp Suite Intercept tab showing captured request

---

6. Step-by-Step Kali Linux Execution

6.1 Start Required Services

sudo service apache2 start
sudo service mysql start

Verify DVWA access:

http://127.0.0.1/dvwa

DVWA login page on Kali

6.2 Login Credentials

Username: admin

Password: password

Set DVWA Security Level to Low.

6.3 SQLMap Table Enumeration

```
sqlmap -u "http://127.0.0.1/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" \
--cookie="security=low; PHPSESSID=PASTE_YOUR_SESSION_ID" \
-D dvwa --tables
```

---

7. Summary of Findings

Vulnerability	Severity
Reflected XSS	Medium
SQL Injection	High
IDOR	Medium

---

8. Recommendations

Implement input validation and output encoding

Use prepared statements for all SQL queries

Enforce strict access control checks

Deploy Content Security Policy (CSP)

---

9. Conclusion

This project successfully demonstrated the complete bug bounty lifecycle—from reconnaissance to reporting—using a vulnerable lab environment. The findings reflect common real-world vulnerabilities and emphasize the importance of secure development practices.

---

10. Disclaimer

This project was conducted strictly for educational purposes in a controlled lab environment. No real-world systems were tested or harmed.