Cybersecurity Project Report

Understanding Web Fuzzing, HTTP Requests, and Burp Suite

Name: Abhay Raj

College: LBS College of Engineering, Kasaragod

Submission Date: 20 June 2025

1. Introduction

This project explores the use of Burp Suite and web fuzzing to understand how data flows between a browser and server in web applications. By analyzing GET and POST requests and performing input fuzzing, I simulated the basic approach of ethical hackers in testing web applications.

2. Website Tested

- DVWA (Damn Vulnerable Web Application)
- Hosted locally using XAMPP
- URL: http://localhost/dvwa
- Purpose: To safely practice and understand web application vulnerabilities

3. GET and POST Request Analysis

- **POST Request (Captured via Burp Suite):**

```
POST /dvwa/login.php HTTP/1.1
Host: localhost
Content-Type: application/x-www-form-urlencoded

username=test&password=123&Login=Login&user_token=...
```

- **GET Request (Example from Navigation):**

```
GET /dvwa/vulnerabilities/fi/?page=include.php HTTP/1.1
Host: localhost
```

4. Tools Used

Tool	Purpose
1. Burp Suite	Intercept and modify HTTP requests
2. DVWA	Vulnerable web app for testing
3. Firefox / Burp Browser	Sending traffic through Burp Proxy
4. XAMPP	Local server for hosting DVWA

5. Fuzzing with Burp Suite Intruder

I performed fuzzing by sending multiple inputs to the username field of the login request using Burp Suite Intruder.

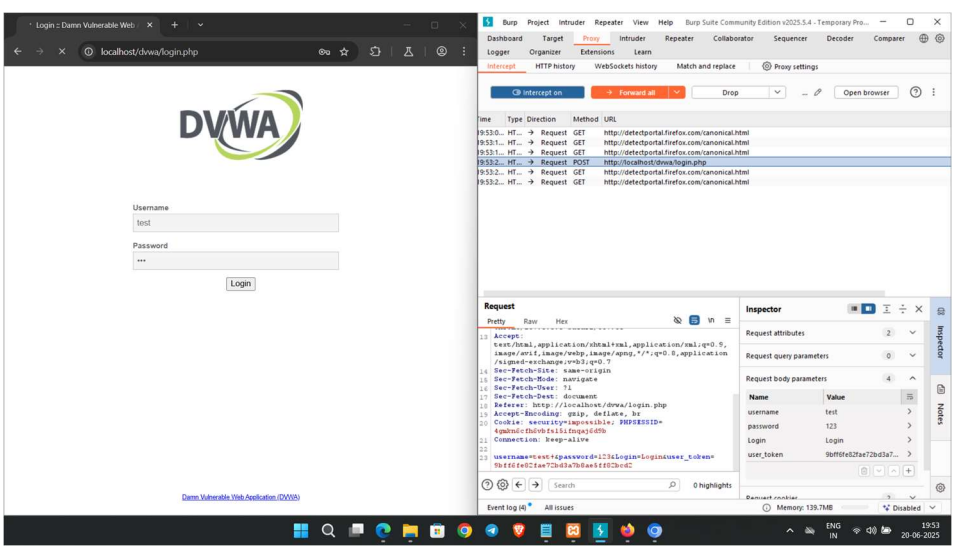
- Steps Taken:

- Captured the POST request using Burp
- Highlighted the username field value (test) and marked it as a payload position
- Entered the following payloads in Burp's Simple List:

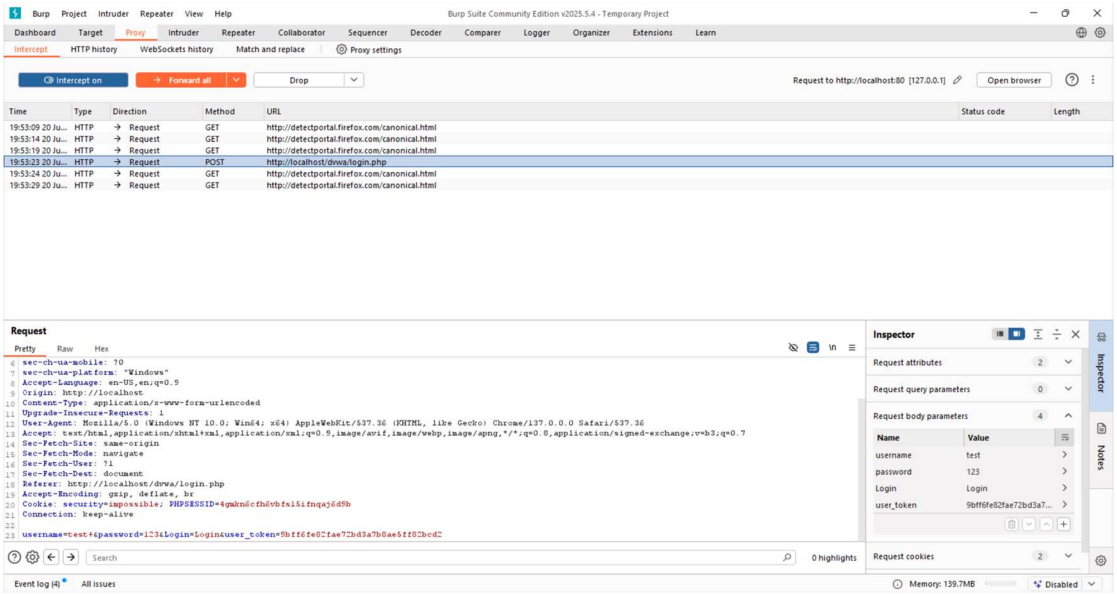
admin
root
' OR 1=1 --
<script>alert(1)</script>
test

6. Screenshots

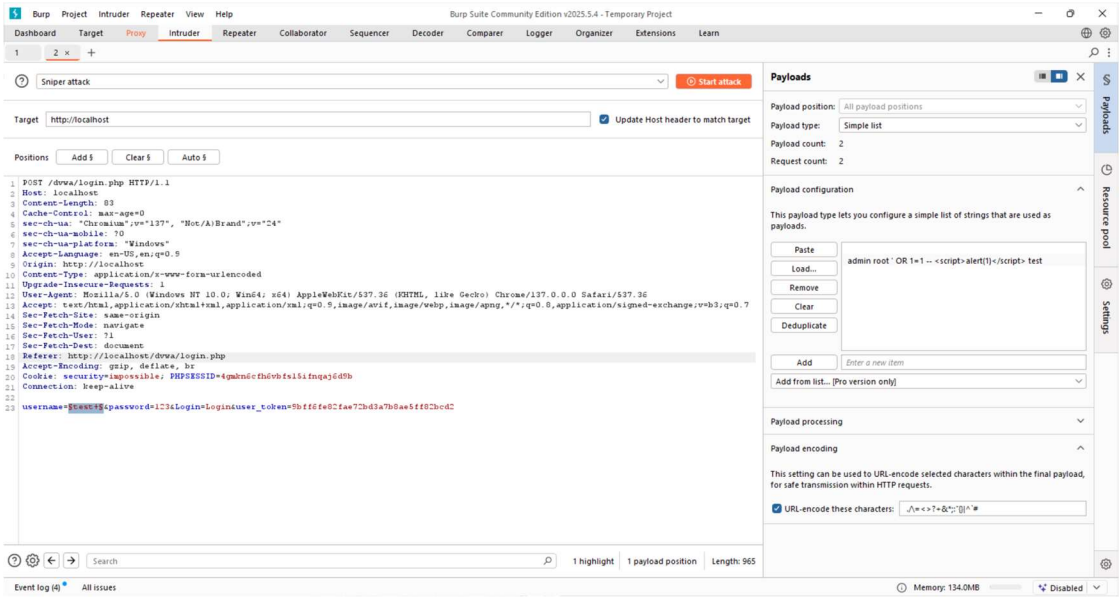
Screenshot 1: **DVWA Login Page with Burp Intercepting POST Request**
Shows the DVWA login form with Burp Suite capturing the login request to /login.php.



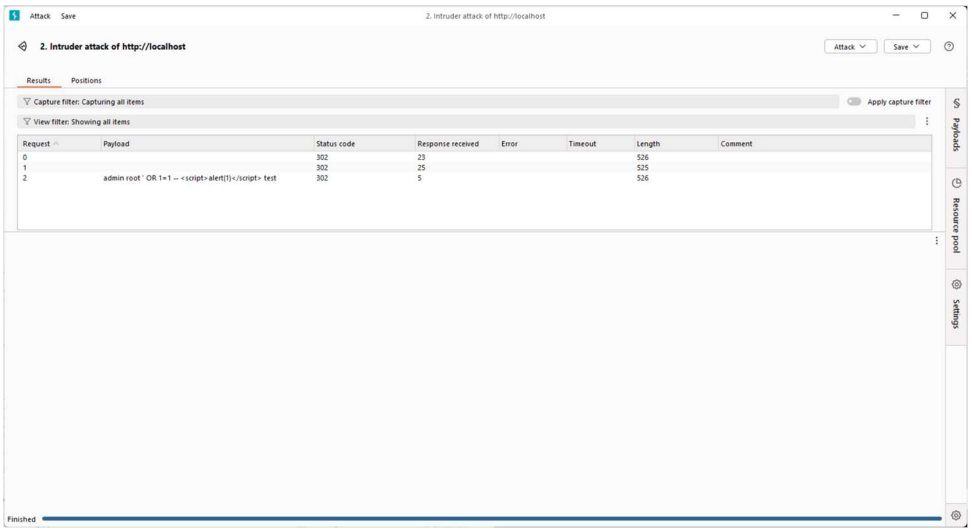
Screenshot 2: **Burp Suite – Intercepted POST Request to Login**
Displays the raw intercepted request with username=test and request body parameters.



Screenshot 3: **Burp Suite Intruder – Payload Position and Payload List**
Shows the configured position on the username field and the list of fuzzing inputs.



Screenshot 4: **Intruder Attack Results**
Displays the results of each payload sent via Intruder, showing status and length differences.



7. Observations

The server responded differently to each payload in terms of length and response time. Although no successful login or bypass occurred, the behavior confirmed that fuzzing can reveal how a server handles various input types. Some responses showed faster rejection, while others took longer to process — indicating backend handling differences.

8. Conclusion

This project gave me hands-on experience with HTTP request analysis and web fuzzing using Burp Suite. I learned how input fields can be tested for unexpected behavior using automated tools. The exercise improved my understanding of web security fundamentals and how attackers and testers approach application testing from a data flow perspective. Even though no major bug was exploited, the focus remained on ethical learning, structured testing, and effective tool usage.