

TASK 1: WEB APPLICATION SECURITY TESTING

Intern Name: Shahidul Hasan

Track Code: FUTURE_CS_01

Domain: Cyber Security

Internship Provider: Future Interns

Duration: June,2025

Tools Used: DVWA, Burp Suite, Browser, Kali Linux

Submission Type: Task Report

AIM:

To conduct security testing on a vulnerable web application and identify common web vulnerabilities such as SQL Injection, Reflected XSS, and Stored XSS, using ethical penetration testing techniques.

TOOLS USED:

- DVWA (Damn Vulnerable Web Application) — A practice environment for web security
- Burp Suite Community Edition — Intercepting proxy and testing tool
- Browser (Chrome) — To interact with DVWA
- Kali Linux — For running DVWA locally

VULNERABILITIES TESTED:

1. SQL Injection

Test Input: ' OR '1'='1

Module Targeted: DVWA > SQL Injection Testing

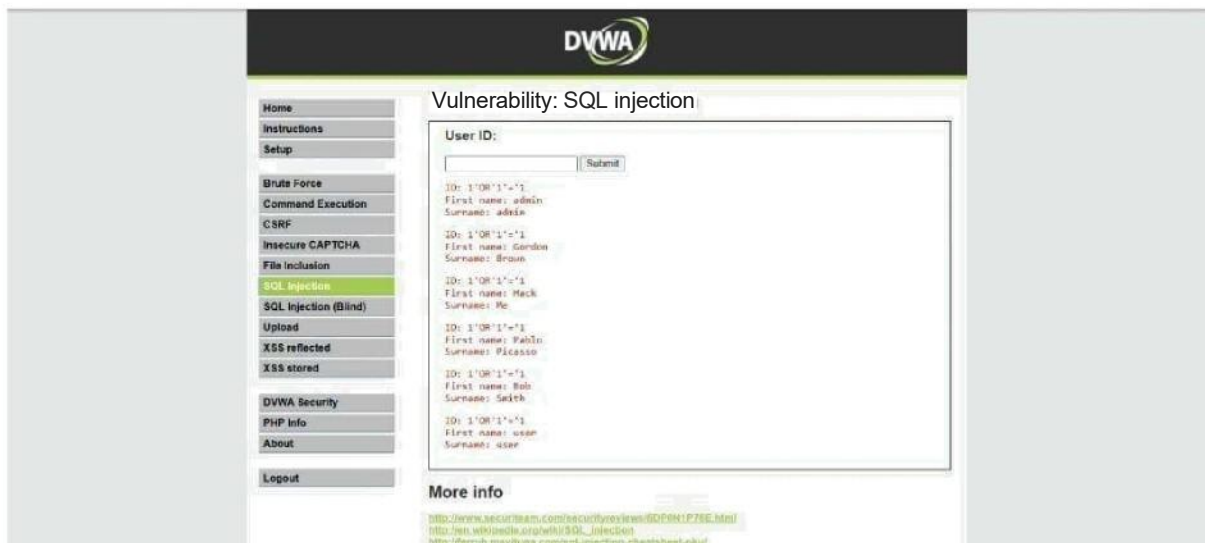
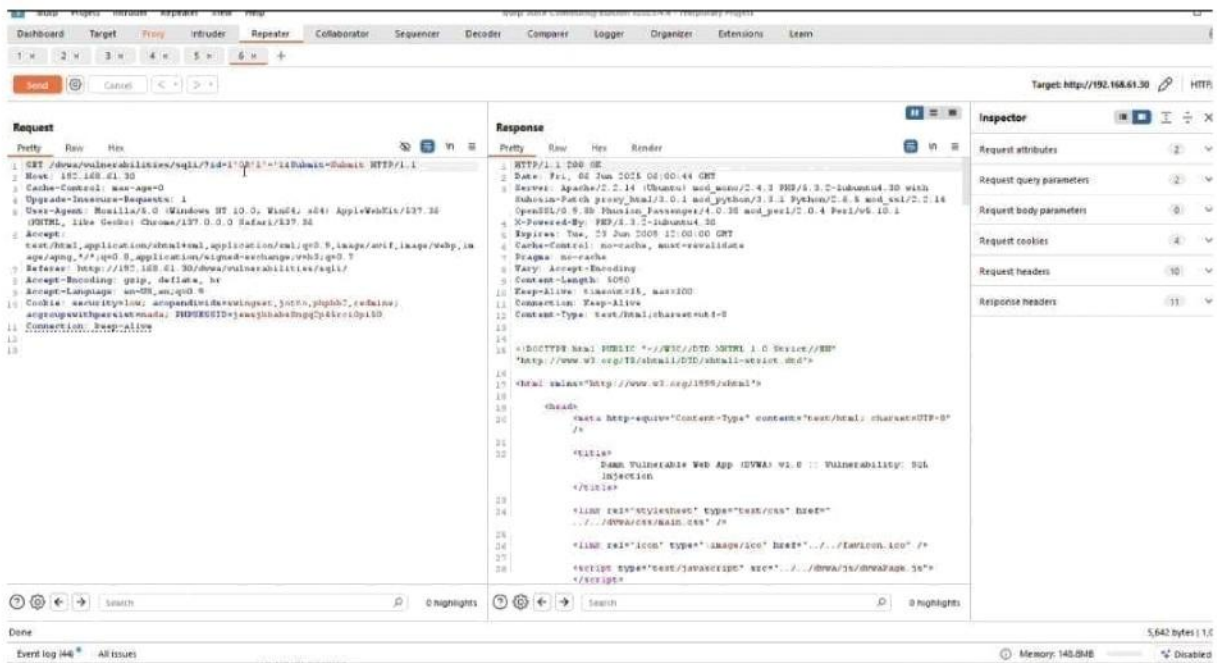
Testing method:

- ✓ Intercepted the request using Burp Suite.
- ✓ Modified the id parameter using the SQL payload above.
- ✓ Server responded with user details, bypassing authentication logic.

Result:

- ✓ The application is vulnerable to **Classic SQL Injection**.

- ✓ Returned unintended user information from the database.



2. Reflected XSS

Test Input: `<script>alert("XSS")</script>`

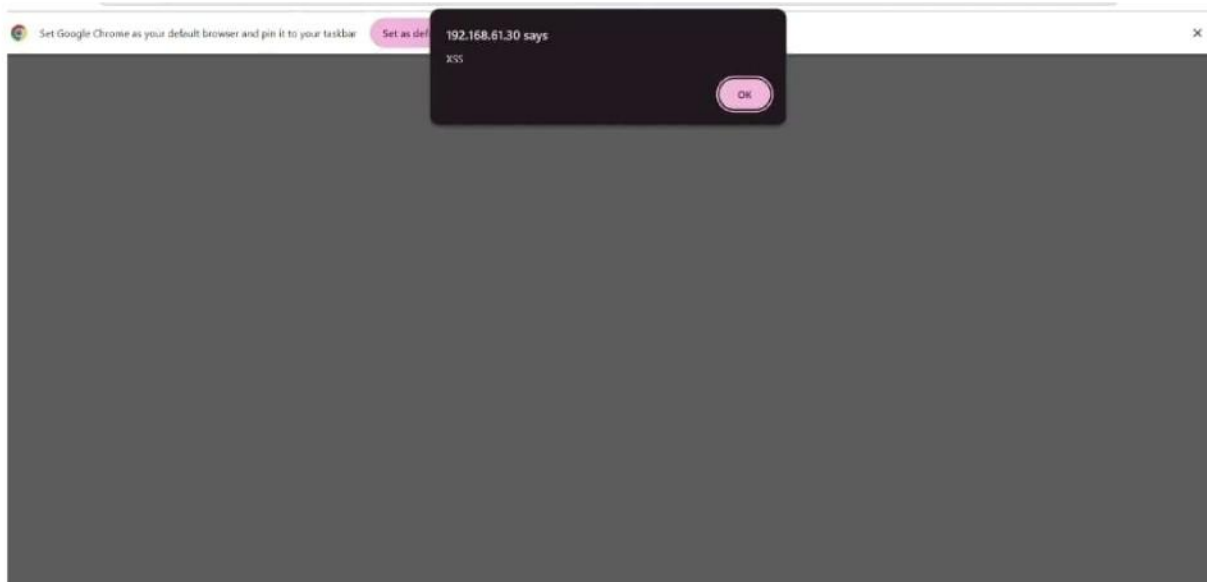
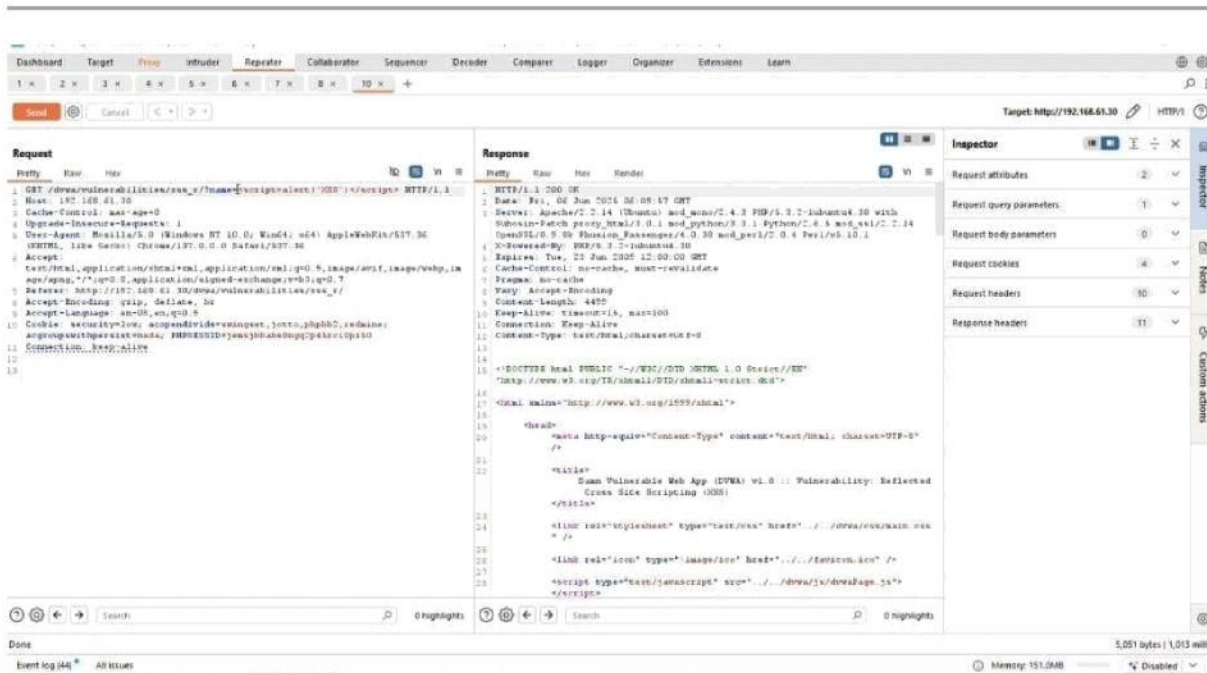
Module Targeted: DVWA > XSS (Reflected)

Testing method:

- ✓ Injected the script in the input field (search/query).
- ✓ Page immediately rendered the input in the response.
- ✓ Alert box appeared in the browser.

Result:

- ✓ The input is reflected without sanitization.
- ✓ Reflected XSS confirmed.



3. Stored XSS

Test Input: `<script>alert("Stored XSS")</script>`

Module Targeted: DVWA > XSS (Stored)

Testing Testing Method:

- ✓ Entered payload into the comment or guestbook form.

Script saved to the backend.

On reloading the page, the alert box triggered automatically.

Result:

DVWA is vulnerable to Stored XSS.

Persistent JavaScript execution is possible.

FINDINGS SUMMARY:

SQL Injection:

The application was vulnerable to classic SQL Injection through unsanitized input fields.

Using a simple payload like ' OR '1'='1, I was able to bypass authentication and retrieve sensitive data.

This indicates that the application does not use parameterized queries or proper input validation.

Reflected XSS (Cross-Site Scripting):

A malicious script entered into a form was immediately reflected and executed in the browser.

The script `<script>alert("XSS")</script>` triggered a popup, proving the vulnerability.

This shows that user input is not properly encoded or sanitized before being displayed.

Stored XSS:

Scripts submitted through the comment section were stored in the database and executed every time the page was reloaded.

The payload persisted across sessions and browsers, affecting all users visiting the page.

This represents a high-severity issue due to its ability to impact multiple users over time.

Security Level:

All tests were conducted with DVWA's security level set to Low, which allows known vulnerabilities to be exploited easily.

It highlights how varying security configurations can change the exposure of the application.

Overall Observation:

The DVWA application effectively demonstrated multiple common web vulnerabilities.

These findings emphasize the importance of secure coding practices, such as input validation, output encoding, and the use of secure development frameworks.

RECOMMENDATIONS:

- ✓ Use prepared statements for all SQL queries.
- ✓ Implement **Content Security Policy (CSP)** headers.
- ✓ Sanitize all input and **encode output** before rendering in HTML.
- ✓ Use **web application firewalls** to detect injection patterns.
- ✓ Validate input length and type server-side.

LEARNING OUTCOME:

Through this task, I gained hands-on experience in:

Identifying and exploiting SQL Injection and XSS vulnerabilities

Using Burp Suite to intercept, manipulate, and replay HTTP requests

Understanding how improper input validation can lead to major risks

Learning remediation techniques to secure modern web applications

Conclusion:

- This task provided valuable hands-on experience in identifying and exploiting common web vulnerabilities using a controlled environment. By working with DVWA and Burp Suite, I was able to successfully perform security assessments targeting SQL Injection, Reflected XSS, and Stored XSS vulnerabilities.
- These tests revealed critical flaws caused by improper input validation and lack of output encoding—issues that are frequently exploited in real-world attacks. The ability to bypass authentication, execute scripts in the browser, and store

malicious code highlighted the significant risks posed by insecure web development practices.

- This exercise not only enhanced my understanding of web application vulnerabilities but also reinforced the importance of adopting secure coding techniques, implementing defense-in-depth strategies, and continuously testing applications for weaknesses throughout the development lifecycle.
- Overall, the task strengthened my practical cybersecurity skills and prepared me to better identify, explain, and remediate real-world web application security issues.