Web Application Security Assessment Report

Application: DVWA (Damn Vulnerable Web Application)

Assessed by: Harsha A

Tools Used: XAMPP, DVWA, phpMyAdmin, Google Chrome, Burp Suite, OWASP ZAP

Date: July 22, 2025

Table of Contents

1. Introduction

This report presents the results of a security assessment performed on the DVWA web application. The goal was to identify and demonstrate common vulnerabilities listed in the OWASP Top 10 using manual testing and automated tools.

2. Scope

Target: http://localhost/dvwa/
Environment: Local (XAMPP + DVWA on Windows 10)
Tools: DVWA, phpMyAdmin, OWASP ZAP, Burp Suite, Chrome Developer Tools
Security Level: Set to Low for testing

3. Methodology

The following approach was used:
- Manual vulnerability testing
- Payload injection (XSS, SQLi)
- Use of automated scanners for recon and validation
- Analysis of DVWA source behavior and database entries

4. Identified Vulnerabilities

4.1 Stored Cross-Site Scripting (XSS)

Vulnerability Type: Stored XSS
Page: XSS (Stored)
Payload Used: <script>alert('XSS')</script>

Steps to Reproduce:
1. Navigate to DVWA → XSS (Stored)
2. Enter payload in the “Message” field
3. Submit the form
4. The script executes as a popup

Impact:
Allows an attacker to execute arbitrary JavaScript in other users’ browsers, leading to session hijacking or defacement.

Mitigation:
- Sanitize and encode all user inputs
- Implement Content Security Policy (CSP)

4.2 SQL Injection

Vulnerability Type: SQL Injection
Page: SQL Injection
Payload Used: 1' OR '1'='1 --

Steps to Reproduce:
1. Navigate to DVWA → SQL Injection
2. Enter the payload in the ID field
3. Submit and observe multiple user records displayed

Impact:
Attackers can bypass authentication or extract sensitive data

Mitigation:
- Use prepared statements (parameterized queries)
- Validate and sanitize all user inputs

5. OWASP Top 10 Mapping

OWASP Category	Vulnerability Detected
A01:2021 – Broken Access Control	❌ Not Tested
A02:2021 – Cryptographic Failures	❌ Not Tested
A03:2021 – Injection	✅ SQL Injection
A05:2021 – Security Misconfiguration	✅ Insecure DVWA setup
A07:2021 – Identification Failures	✅ Weak login form
A08:2021 – Software & Data Integrity	❌ Not Tested
A09:2021 – Security Logging	❌ Not Tested
A10:2021 – Server-Side Request Forgery	❌ Not Tested
A06:2021 – Vulnerable Components	✅ DVWA is intentionally weak
A04:2021 – Insecure Design	✅ Reflected/Stored XSS

6. Mitigation Strategies

- Enforce strong input validation
- Use output encoding (e.g., htmlspecialchars() in PHP)
- Apply least privilege to database users

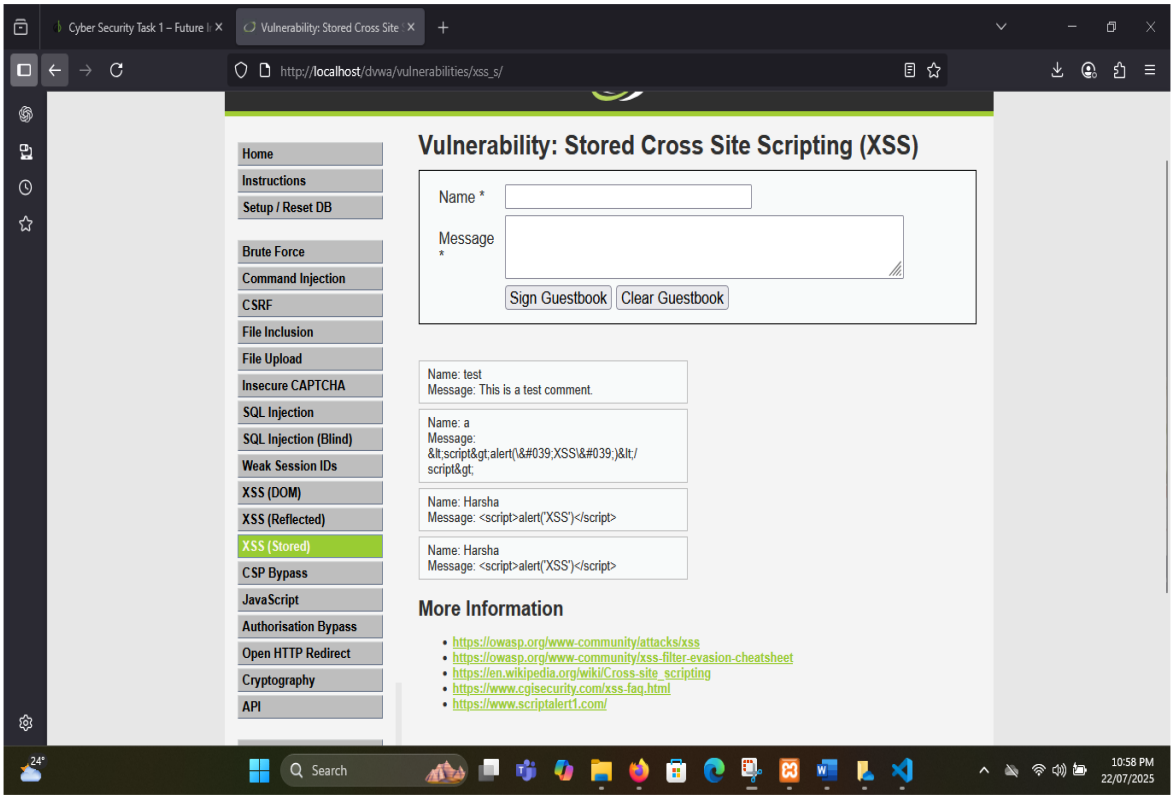
- Sanitize all form inputs and URLs
- Use frameworks that prevent common vulnerabilities

7. Conclusion

The DVWA application successfully demonstrated various common web vulnerabilities. The exercise helped in understanding real-world exploitation methods and their impacts. Fixing such issues is crucial for web application security in production environments.

8. References

- OWASP Top 10 Project – 2021: https://owasp.org/Top10/
- DVWA GitHub: https://github.com/digininja/DVWA
- OWASP Testing Guide


Cyber Security Task 1 – Future
Vulnerability: Stored Cross Site
http://localhost/dvwa/vulnerabilities/xss_s/
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
JavaScript
Authorisation Bypass
Open HTTP Redirect
Cryptography
API
Vulnerability: Stored Cross Site Scripting (XSS)
Name *
Message *
Sign Guestbook
Clear Guestbook
Name: test
Message: This is a test comment.
Name: a
Message: <script>alert(\'XSS\')</script>
Name: Harsha
Message: <script>alert('XSS')</script>
Name: Harsha
Message: <script>alert('XSS')</script>
More Information
https://owasp.org/www-community/attacks/xss
https://owasp.org/www-community/xss-filter-evasion-cheatsheet
https://en.wikipedia.org/wiki/Cross-site_scripting
https://www.cgisecurity.com/xss-faq.html
https://www.scriptalert1.com/
Search
10:58 PM
22/07/2025