Project 2:

Web Application Penetration Testing – Final Report

Intern Name: Jaishri Mahalia
Organization/Institution: Infotact Solution
Supervisor/Mentor:
Submission Date: 07/08/2025

Table of Contents

1. Introduction

Web applications have become integral to modern businesses, providing essential services to customers and organizations alike. However, they are often targeted by cyber attackers due to insecure configurations, weak authentication mechanisms, and poor coding practices.

This project involved performing penetration testing on **Damn Vulnerable Web Application (DVWA)** — an intentionally insecure web application designed for security testing and learning purposes.

The objective was to identify, exploit, and document vulnerabilities based on the **OWASP Top 10** security risks, simulating real-world attack scenarios and recommending effective remediation strategies.

2. Problem Statement

The main objective was to perform a **comprehensive penetration test** on DVWA, focusing on vulnerabilities such as:

- **Broken Authentication**
- **Cross-Site Scripting (XSS)** – Stored and Reflected
- **SQL Injection**
- **Sensitive Data Exposure**

Goals of the assessment:

- Identify existing vulnerabilities in the application.
- Demonstrate possible exploitation techniques.
- Provide practical recommendations to enhance security.

3. Methodology

The following methodology was adopted for the penetration test:

1. **Environment Setup**
    - DVWA launched inside **Kali Linux** using VMware Workstation.
    - Docker container started for DVWA instance.
    - Accessed via: `http://127.8.0.1` or `http://dvwa`.

2. **Tools Used**
    - **Burp Suite** – Intercepting and modifying HTTP requests.
    - **Web Browser** – Accessing and testing DVWA.
    - **Custom Wordlists** – For brute-force authentication testing.
    - **SQL Injection Payloads** – For database exploitation.
3. **Testing Approach**
    - Reconnaissance & Information Gathering.
    - Vulnerability Identification (OWASP Top 10 focus).
    - Exploitation & Proof of Concept (PoC).
    - Documentation & Remediation Planning.

4. Findings & Exploitation Details

4.1 Broken Authentication – Brute Force

Description:
DVWA's login page lacked brute-force protection, allowing repeated login attempts without triggering any lockout.

Steps Performed:

1. Captured login request using Burp Suite.
2. Configured Intruder with **Cluster Bomb** attack type.
3. Supplied username and password wordlists.
4. Identified valid credentials based on HTTP response differences.

Impact:
Unauthorized access to sensitive application areas.

Remediation:

- Implement account lockout after 3–5 failed attempts.
- Enforce strong password policies.
- Integrate CAPTCHA to deter automated attacks.
- Display generic login failure messages.

4.2 Cross-Site Scripting (XSS) – Stored

Description:
Malicious JavaScript payload stored in the application database and executed whenever the vulnerable page was viewed.

Exploit:

```
html
Copy code
<script>alert('Stored XSS')</script>
```

Impact:

- Session hijacking.
- Defacement.
- Redirection to malicious sites.

Remediation:

- Sanitize and validate all user inputs.
- Encode special characters (<, >) before storing.
- Use libraries like **DOMPurify** for HTML sanitization.

4.3 Cross-Site Scripting (XSS) – Reflected

Description:
User input directly reflected on the page without sanitization, enabling one-time execution of malicious scripts.

Exploit:

```
html
Copy code
<img src=x onerror=alert('XSS')>
```

Impact:

- Phishing.
- Credential theft.

Remediation:

- Proper output encoding.
- Input validation to reject script-based payloads.
- Use secure frameworks with built-in XSS protection.

4.4 SQL Injection

Description:
Application vulnerable to SQL Injection via User ID parameter, allowing attackers to bypass authentication and extract database contents.

Exploit:

```sql
%' OR '1'='1
```

Impact:

- Data exfiltration.
- Unauthorized modifications.

Remediation:

- Use prepared statements or parameterized queries.
- Avoid concatenating user input directly into SQL queries.
- Implement strict input validation.

4.5 Sensitive Data Exposure

Findings:

1. **Session Persistence Across Tabs** – Sessions remained valid when the URL was reused in another browser tab.
2. **Exposed Configuration Directory** – Access to `/config/` directory revealed application settings and potential database credentials.

Impact:

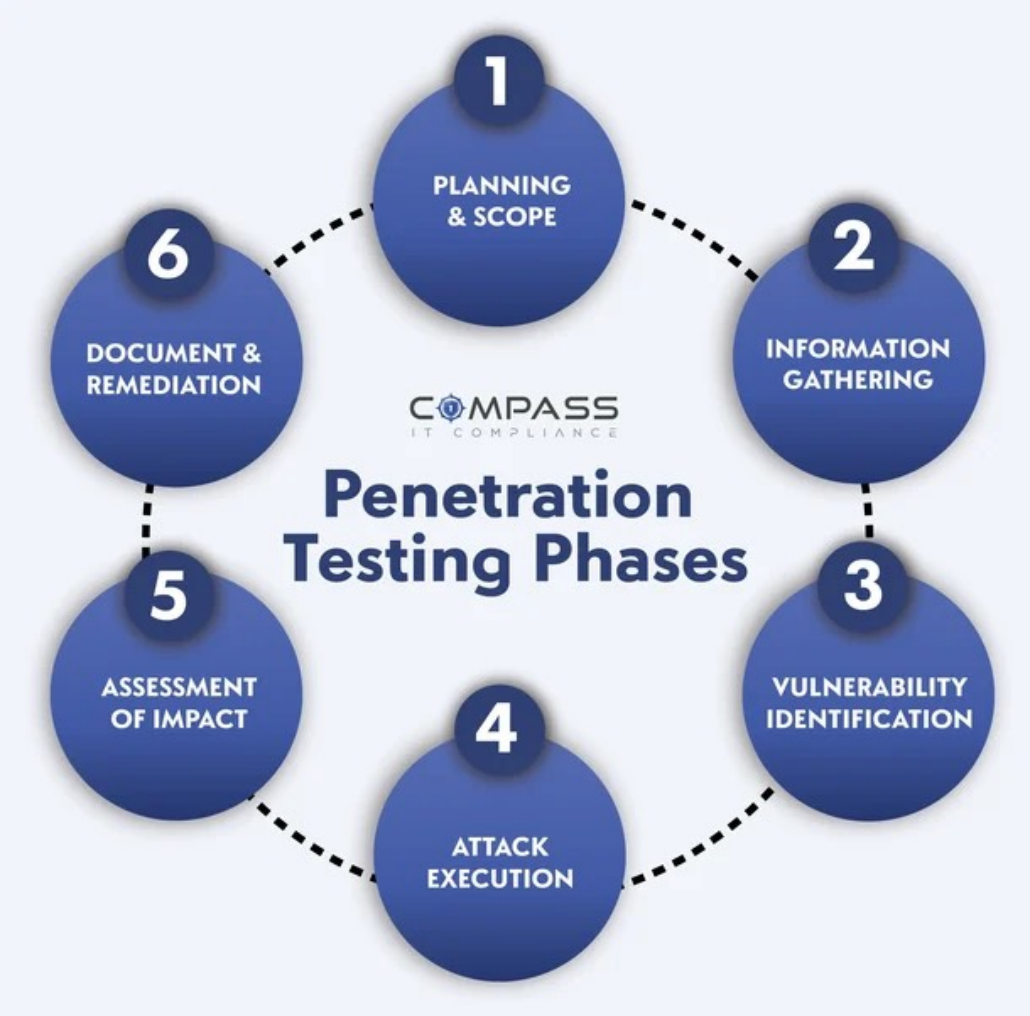
- Unauthorized access to sensitive data.
- Increased risk of targeted attacks.

Remediation:

- Invalidate session tokens on logout or inactivity.
- Restrict direct access to sensitive directories.
- Use **HTTPS** to protect data in transit.

5. Images


Damn Vulnerable Web App (DVWA) - Login - Mozilla Firefox
Damn Vulnerable Web A
192.168.50.40/login.php
DVWA
Username
Password
Login


1
PLANNING & SCOPE
2
INFORMATION GATHERING
3
VULNERABILITY IDENTIFICATION
4
ATTACK EXECUTION
5
ASSESSMENT OF IMPACT
6
DOCUMENT & REMEDIATION
COMPASS
IT COMPLIANCE
Penetration Testing Phases

6. Conclusion

The penetration testing exercise demonstrated that DVWA, being intentionally vulnerable, exposed several high-risk security issues found in real-world applications. Addressing these vulnerabilities requires:

- Strong authentication mechanisms.
- Proper input/output handling.
- Secure coding practices.
- Restricting access to sensitive resources.

Implementing the recommended mitigations will significantly reduce the application's attack surface and enhance its overall security posture.