Rhombix technologies

Intern in Cyber Security

Syeda fakiha

4th project

Due date 30th September

Vulnerability Assessment and Penetration Testing (VAPT) Report

1. Executive Summary

This report presents the findings of a Vulnerability Assessment and Penetration Testing (VAPT) conducted on a deliberately vulnerable web application (DVWA / OWASP Juice Shop). The objective of this project was to identify potential vulnerabilities, exploit them (in a controlled environment), and provide recommendations to mitigate these issues. The assessment combined automated scanning with manual testing to ensure thorough coverage.

2. Tools and Environment

The following tools and environment were used for this assessment:

DVWA / OWASP Juice Shop installed on the local machine

Kali Linux (attacker machine)

Nmap: Network scanning

Nikto: Web server vulnerability scanning

Burp Suite: Proxy interception and vulnerability testing

OWASP ZAP: Automated vulnerability scanning

Manual testing techniques for SQL Injection, XSS, and Authentication flaws

3. Methodology

The VAPT methodology followed these key stages:

a) Information Gathering

Nmap was used to scan for open ports, services, and versions.

Nikto was used to identify potential web server misconfigurations and outdated components.

b) Vulnerability Scanning

OWASP ZAP and Burp Suite were used to scan the web application for common vulnerabilities.

Automated scans helped in identifying issues such as SQL Injection, XSS, and Broken Authentication.

c) Manual Exploitation

SQL Injection attempts were made using payloads such as ' OR '1'='1.

XSS testing included inserting payloads like <script>alert('XSS')</script>.

Broken Authentication was tested by manipulating cookies and session tokens.

Directory Traversal was tested with payloads like ../../etc/passwd.

d) Documentation

Each identified vulnerability was documented with Title, Description, Risk Level, Steps to Reproduce, Proof of Concept, and Recommendation.

4. Vulnerability Findings

4.1 SQL Injection (Login Page)

Risk Level: High

Description: The login page was found vulnerable to SQL Injection, allowing bypass of authentication.

Steps to Reproduce:

1. Navigate to the login page
2. Enter username: ' OR '1'='1
3. Enter password: anything
4. Successfully logged in as admin

Recommendation: Use parameterized queries and prepared statements. Implement server-side input validation.

4.2 Cross-Site Scripting (XSS) Search Box

Risk Level: Medium

Description: The search functionality reflected unsanitized input back to the browser, enabling execution of JavaScript.

Steps to Reproduce:

1. Go to the search page
2. Enter payload: <script>alert('XSS')</script>
3. Alert box pops up, proving XSS vulnerability.

Recommendation: Sanitize and escape all user input. Use Content Security Policy (CSP).

4.3 Broken Authentication – Session Token

Risk Level: High

Description: Session management was weak. Predictable session tokens allowed attackers to hijack user accounts.

Steps to Reproduce:

1. Log in as a user
2. Capture session token using Burp Suite
3. Reuse the token in another browser to gain unauthorized access.

Recommendation: Use secure random session tokens. Implement session expiration and invalidation after logout.

4.4 Directory Traversal – File Download

Risk Level: Medium

Description: File download functionality allowed traversal outside the intended directory.

Steps to Reproduce:

1. Navigate to the file download page
2. Modify the parameter with. /. /etc/passwd
3. Application returned sensitive system files.

Recommendation: Validate and sanitize file paths. Restrict file access to specific directories.

5. Risk Assessment Summary

The following table summarizes the vulnerabilities discovered:

1. SQL Injection (High Risk)
2. Cross-Site Scripting (XSS) (Medium Risk)
3. Broken Authentication (High Risk)
4. Directory Traversal (Medium Risk)

6. Recommendations

Implement secure coding practices, including prepared statements and parameterized queries.

Enforce strict input validation and sanitization.

Use Web Application Firewalls (WAF) for additional protection.

Regularly update and patch web application software.

Conduct regular security assessments and penetration tests.

7. Conclusion

The VAPT exercise successfully identified critical vulnerabilities in the web application. If exploited by a malicious attacker, these could compromise the confidentiality,

integrity, and availability of the system. By addressing the recommendations provided, the security posture of the application can be significantly improved.