Web Application Vulnerability Assessment Report

* Prepared by: Pranjal Jitendra Vishwakarma
* Date: 16th September, 2025.
* Tested Application: DVWA
* Environment: **DVWA** hosted on **Metasploitable 2 VM** (running on VirtualBox, bridged mode).
* Tools used: OWASP ZAP

1. Objective:

The aim of this evaluation was to discover possible security weaknesses in the chosen web application, emulate actual attack scenarios, and suggest countermeasures to enhance the application's security stance.

I chose to host DVWA on **Metasploitable 2 (Linux VM)** rather than XAMPP (on Windows) because it is a pre-built vulnerable environment that closely simulates a real-world server.

1. Methodology:

The approach of this report is based on OWASP Testing Guide and industry standard penetration testing methodology:

1. Reconnaissance: Systematic phase of a cyber-attack where threat actor gather information about a target system to identify the vulnerabilities and plan a successful attack.
2. Vulnerability Scanning: The automated process of identifying security weaknesses and flaws in an organization’s network, systems and applications by using specialized tools to probe for known vulnerabilities before they can be exploited by the attackers.
3. Manual testing: The penetration testing performed by the skilled ethical hackers who use their expertise to stimulate real-world attacks, identify complex or undiscovered vulnerabilities and confirm the exploitability of weaknesses that automated tools might miss.
4. Exploitation (Ethical): Pen testers use ethical hacking skills to not only find network vulnerabilities but also exploit them in stimulated attacks.
5. Reporting: Documented all findings with risk ratings and remediation.
6. OWASP Top 10 Mapping

|  |  |  |
| --- | --- | --- |
| Sr. No. | OWASP Top 10 Category | Fount during Test |
| 1 | Broken Access control | ❌ Not Found |
| 2 | Cryptographic Failure | ❌ Not Found |
| 3 | Injection | ✅ Found |
| 4 | Insecure Design | ⚠️ Minor |
| 5 | Security Misconfiguration | ✅ Found |
| 6 | Vulnerable and Outdated Components | ❌ Not Found |
| 7 | Identification and Authentication Failure | ⚠️ Minor |
| 8 | Software and data integrity Failure | ❌ Not Found |
| 9 | Security logging Monitoring Failure | ❌ Not Found |
| 10 | Server Side Request Forgery | ❌ Not Found |

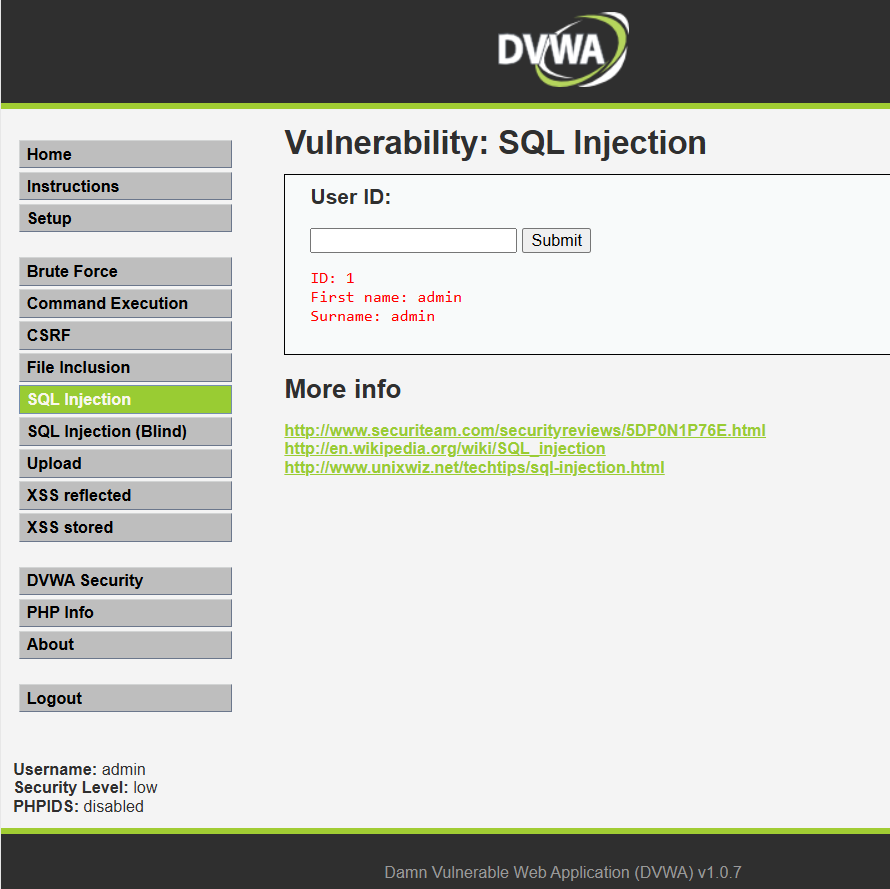
1. Findings And Exploits
2. SQL Injection (High Severity):

SQL injection is a web security vulnerability that allows an attacker to interfere with the queries that an application makes to its database.

**Impact:** The attacker can log in as admin and view or edit the sensitive data.

**Steps to perform:**

1. Navigate to Login page
2. Set level ‘Low’.
3. Enter payload: 1 or any number in username field and any password. DVWA Low directly uses the input in the SQL query without sanitization, so entering 1 fetches the admin record. In DVWA Low **User ID field**, the input is **numeric**, so **quote-based payloads don’t work**.
4. Click Login.



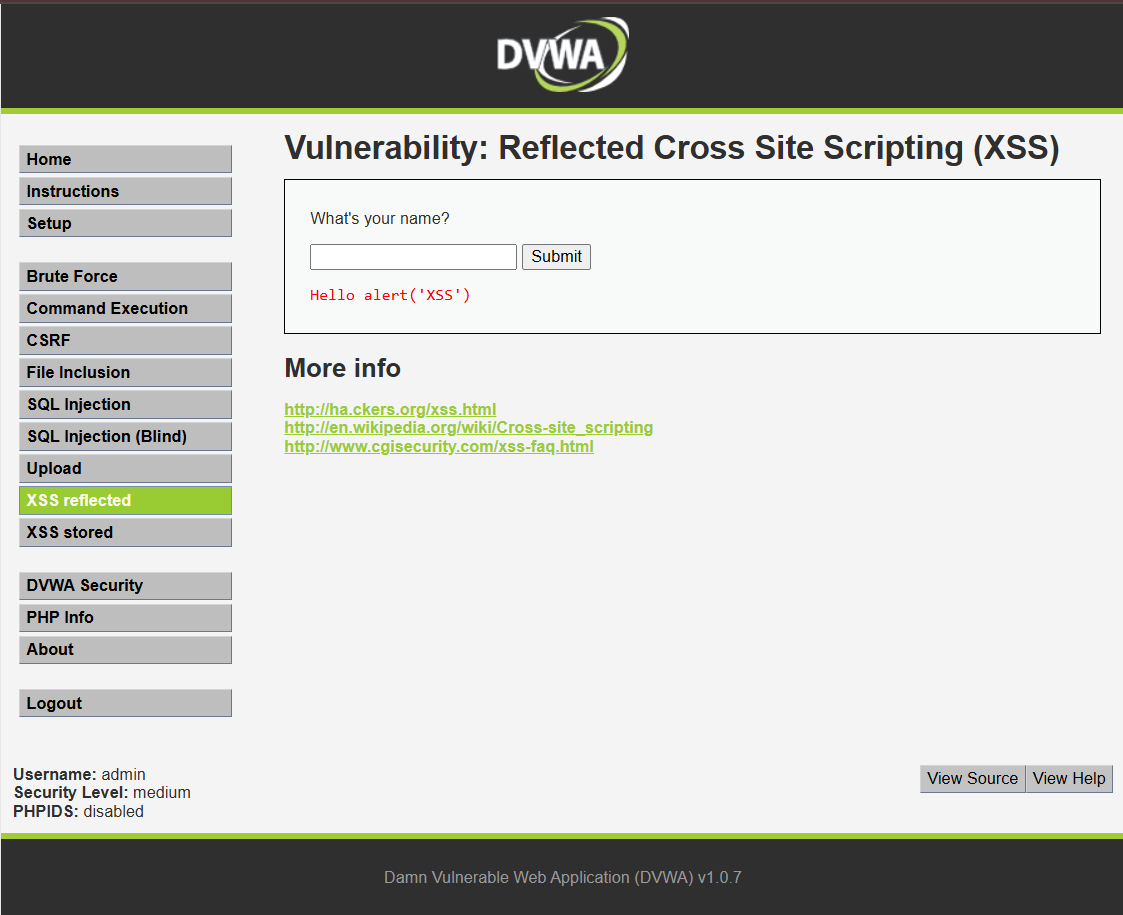
**Mitigation:**

1. Use parameterized queries or prepared statements.
2. Implement input validation and sanitization.
3. Reflected XSS (Medium Severity):

Reflected XSS occurs when a web application **takes user input and immediately reflects it back in the web page without proper validation or sanitization.**

**Steps to perform:**

1. Set level to medium.
2. Navigate to XSS reflected option.
3. Inject payload <script>alert('XSS')</script>
4. Observe pop-up



**Mitigation:**

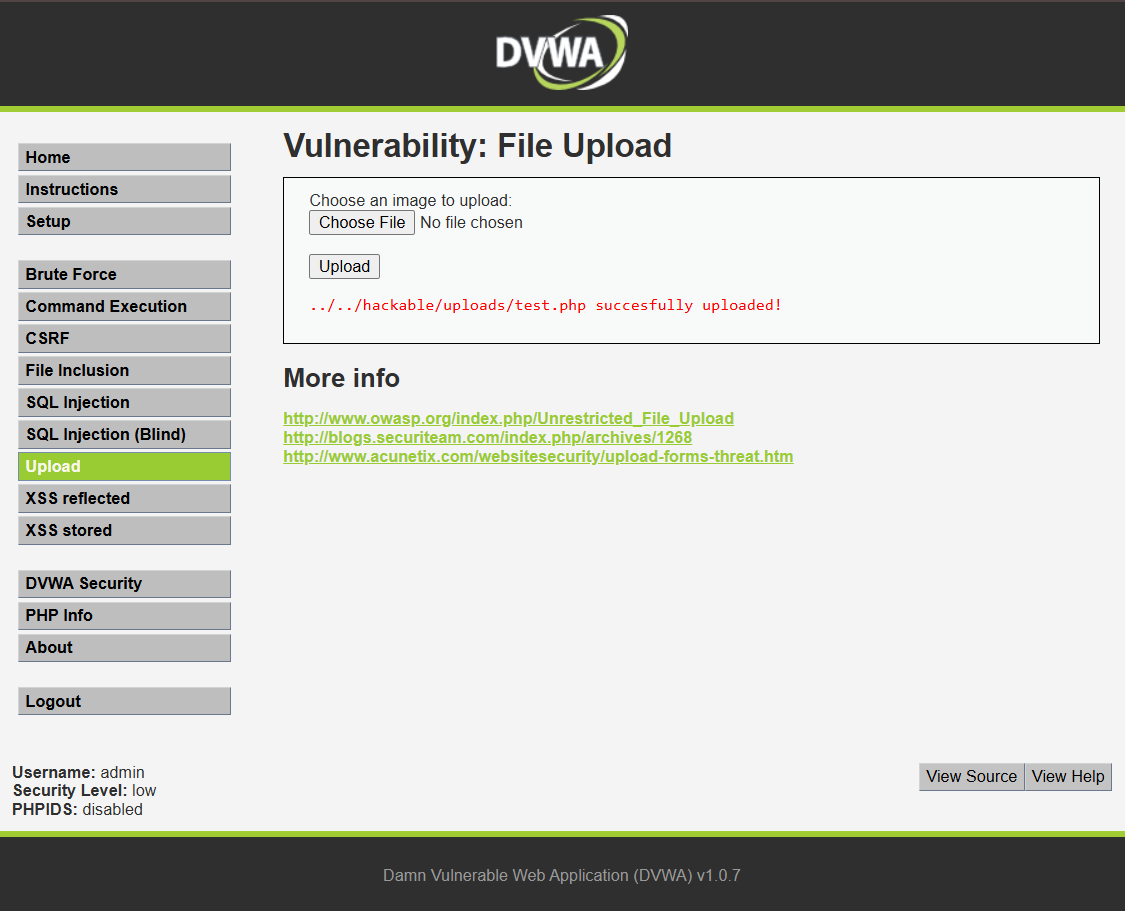
1. Validate and sanitize all user input.
2. Properly encode all output before rendering it in a browser.
3. Implement a Content Security Policy (CSP) to restrict script execution.
4. Security Misconfiguration (Medium Severity):

Directory listing was enabled on /uploads directory.

**Steps to perform:**

1. Set Level to low.
2. Upload a php file. ( Because a PHP file can run code on the server, letting us fully control it if the upload isn’t secure.)

**Impact:** Attackers could access sensitive uploaded files.



**Mitigation:**

1. Disable directory listing in web server configuration.
2. Restrict access to upload folders.
3. Risk Ratings:

|  |  |  |  |
| --- | --- | --- | --- |
| Vulnerability | Security level | Severity | CVSS Score |
| SQL Injection | Low | High | 9.8 |
| Cross site scripting | Medium | Medium | 6.1 |
| Security Misconfiguration | Low | Medium | 5.3 |

## Conclusion

The assessment revealed multiple vulnerabilities including SQL Injection and XSS that could compromise the application if exploited by attackers. Implementing recommended mitigations will significantly improve the overall security posture.