Security Assessment Report

Target Application: Damn Vulnerable Web Application (DVWA)
Environment: http://localhost/DVWA
Tools Used: OWASP ZAP, Burp Suite, Nikto
Date of Scan: June 26–27, 2025

---

1. Executive Summary

This report summarizes the results of automated and manual security scans performed against DVWA using OWASP ZAP, Burp Suite, and Nikto. The objective was to identify and validate common web vulnerabilities and map them to the OWASP Top 10 risks. The application was scanned in a controlled lab environment on Kali Linux.

---

2. Key Findings

Risk Level	Number of Issues	Notable Examples
High	1	- SQL Injection via Burp Suite
Medium	4	- Missing CSP Header- Missing Anti-Clickjacking Header- Directory Indexing- Public Git Files
Low	4	- Server Version Disclosure- Missing X-Content-Type-Options- Cookie Without SameSite Attribute- HTTP Method Disclosure

Risk Level	Number of Issues	Notable Examples
Informational	4	- Authentication Detected- User Agent Fuzzing Detected- Redirect to login.php- Git ignore and Docker ignore file exposed

---

3. Vulnerability Details

3.1 SQL Injection via Burp Suite

- **Risk Level:** High
- **OWASP Mapping:** A01:2021 – Injection
- **Page:** /vulnerabilities/sqli/
- **Payload:** ' OR '1'='1
- **Tool:** Burp Suite (Community Edition)
- **Description:** This payload altered the backend SQL query to always return true.
- **Impact:** Bypasses authentication and can dump database content.
- **Remediation:** Use parameterized queries; sanitize input; deploy a WAF.

3.2 Missing Content-Security-Policy Header

- **Risk Level:** Medium
- **OWASP Mapping:** A05:2021 – Security Misconfiguration
- **Tool:** ZAP
- **Description:** The response from the server does not include a Content-Security-Policy (CSP) header.

- **Impact:** May allow execution of malicious scripts (XSS).
- **Remediation:** Implement a strict CSP header.

3.3 Missing X-Frame-Options (Clickjacking)

- **Risk Level:** Medium
- **OWASP Mapping:** A05:2021 – Security Misconfiguration
- **Tool:** Nikto
- **Description:** The site does not prevent embedding in iframes.
- **Impact:** Subject to clickjacking attacks.
- **Remediation:** Add X-Frame-Options: SAMEORIGIN or use Content-Security-Policy: frame-ancestors.

3.4 Directory Indexing and Sensitive File Exposure

- **Risk Level:** Medium
- **OWASP Mapping:** A05:2021 – Security Misconfiguration
- **Tool:** Nikto
- **Description:** Public access to /config/, /database/, /tests/, /docs/, and .git/ directory files.
- **Impact:** May expose configuration, credentials, or repo metadata.
- **Remediation:** Disable directory listing and restrict access to internal folders.

3.5 Server Version Disclosure

- **Risk Level:** Low
- **OWASP Mapping:** A05:2021 – Security Misconfiguration
- **Tool:** ZAP, Nikto

- **Description:** Server reveals its version in HTTP headers.
- **Impact:** Aids attacker in identifying software to exploit.
- **Remediation:** Disable version banners in server configuration.

4. OWASP Top 10 Mapping Table

Vulnerability	OWASP Category
SQL Injection	A01:2021 – Injection
Missing CSP Header	A05:2021 – Security Misconfiguration
Missing X-Frame-Options	A05:2021 – Security Misconfiguration
Cookie Without SameSite	A01:2021 – Broken Access Control
Server Version Disclosure	A05:2021 – Security Misconfiguration
Missing X-Content-Type-Options Header	A05:2021 – Security Misconfiguration
Directory Indexing and Git Exposure	A05:2021 – Security Misconfiguration
HTTP Method Disclosure	A07:2021 – Identification and Authentication Failures

---

5. Supporting Evidence

- ZAP HTML report generated and saved as evidence.
- Burp Suite HTTP history logs and payload screenshots.

- Nikto scan output included as textual appendix.
- Tools used: OWASP ZAP, Burp Suite Community Edition, Nikto

6. Conclusion

This assessment identified multiple high, medium, and low-risk vulnerabilities in DVWA, including a confirmed SQL Injection and misconfigured directories. The findings reflect common web application weaknesses as outlined in the OWASP Top 10.

---

7. Recommendations

- Implement secure HTTP headers as listed above.
- Fix SQL Injection by using parameterized queries.
- Restrict directory access and disable file indexing.
- Conduct regular manual and automated security scans.
- Educate developers on secure coding practices using OWASP standards.

Prepared by: Moses Adjewoda
Date: June 26–27, 2025