**Web Application Security Assessment Report**

Target: Damn Vulnerable Web Application (DVWA)

Prepared by: Nikhil Kumar Yadav  
CIN ID: FIT/AUG25/CS3110  
Date: 21 August 2025

# 1. Executive Summary

This assessment was performed on Damn Vulnerable Web Application (DVWA) to identify and demonstrate common web vulnerabilities using ethical hacking tools. The goals were to detect issues, map them to OWASP Top 10, and document reproducible steps, business impact, and remediation guidance.

- Scope: DVWA (local test environment)  
- Tools Used: Burp Suite (Community), OWASP ZAP, SQLMap, Browser DevTools  
- Vulnerabilities Identified so far:  
 • SQL Injection (Critical)  
 • Cross-Site Scripting (XSS) – Reflected/Stored (High)  
- Overall Risk Rating: High  
- Outcome: Evidence captured (screenshots) and mitigations proposed per finding.

# 2. Methodology

1) Reconnaissance & Crawling (ZAP/Burp) → 2) Parameter discovery & manual probing → 3) Targeted exploitation (SQLMap for SQLi; payloads for XSS) → 4) Evidence collection (screenshots, tool logs) → 5) Risk rating & remediation.

# 3. Findings

## 3.1 SQL Injection — Confirmed (Critical)

• OWASP Category: A03:2021 – Injection  
• Location: DVWA → SQL Injection module (parameter: id)  
• Description: Unsanitized user input is concatenated into SQL queries, enabling extraction of database contents.  
• Impact: Full compromise of user data (usernames, password hashes), potential lateral impact depending on DB privileges.

Evidence:

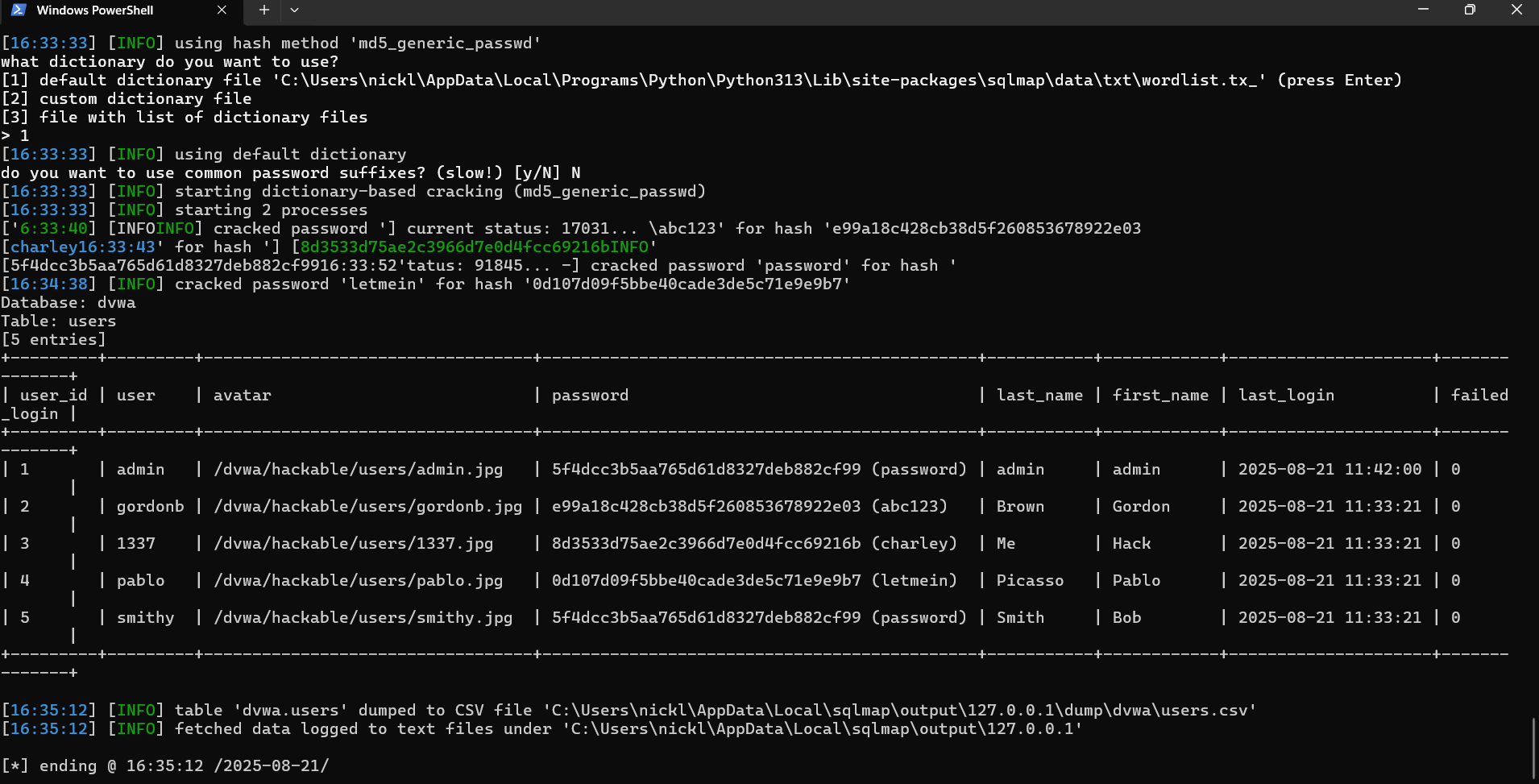


Figure 1: SQLMap dump showing extraction of the 'dvwa.users' table.

Steps to Reproduce:

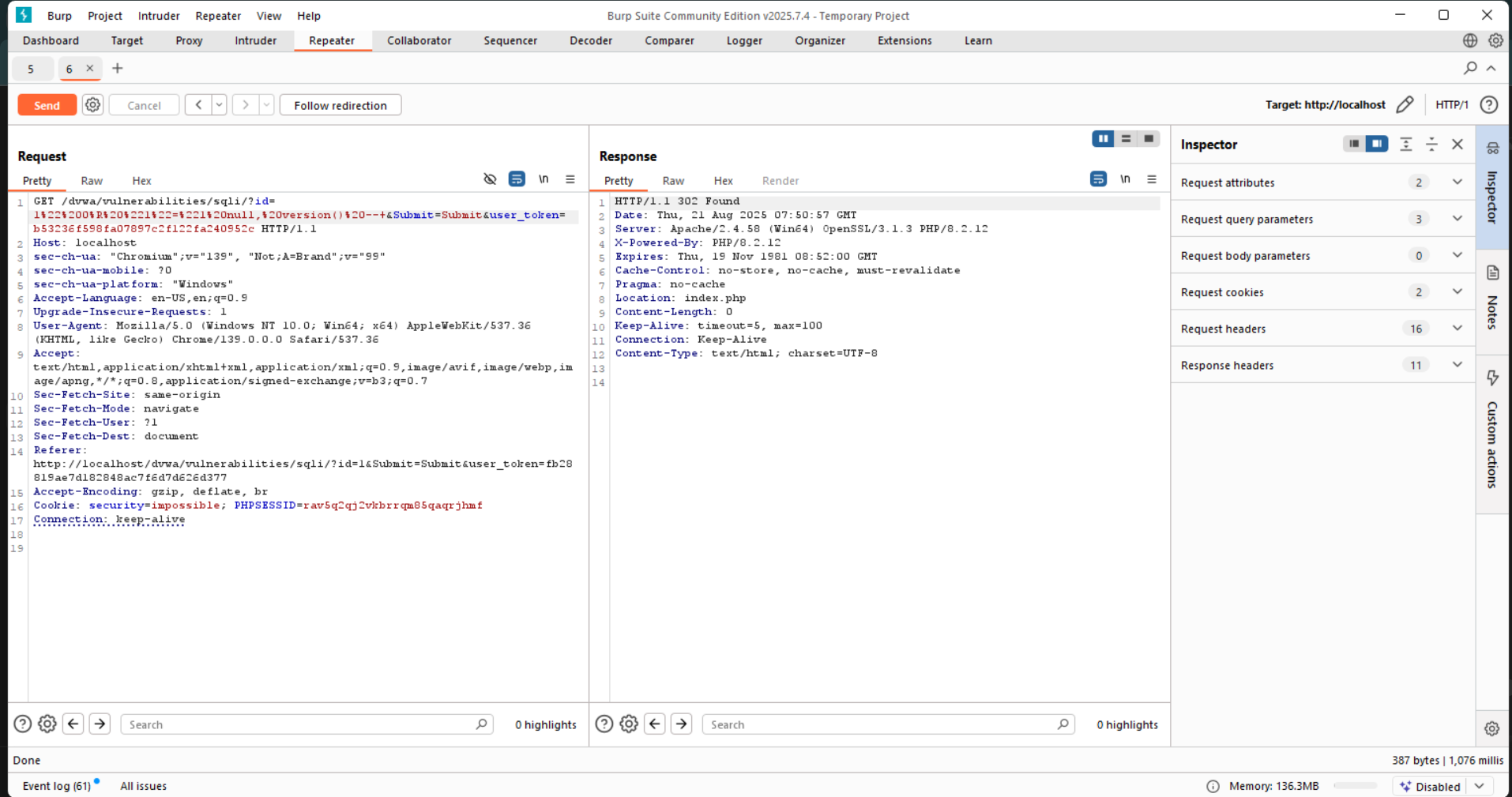
1) Set DVWA security level to Low.  
2) Browse to the SQL Injection module; capture the vulnerable request.  
3) Run SQLMap against the endpoint with the session cookie.  
4) Enumerate DBs and dump the 'dvwa.users' table to confirm exposure.

Remediation: Use parameterized queries (prepared statements), whitelist numeric IDs, enforce least-privilege DB users, rotate exposed credentials, and consider WAF rules for defense-in-depth.

## 3.2 Cross-Site Scripting (XSS) — Confirmed (High)

• OWASP Category: A03:2021 – Injection (XSS)  
• Locations: DVWA → XSS (Reflected) and XSS (Stored)  
• Description: User-controlled input is reflected/stored without proper encoding, allowing script execution in the victim’s browser.  
• Impact: Session hijacking, credential theft, user redirection and phishing, defacement.

Evidence:



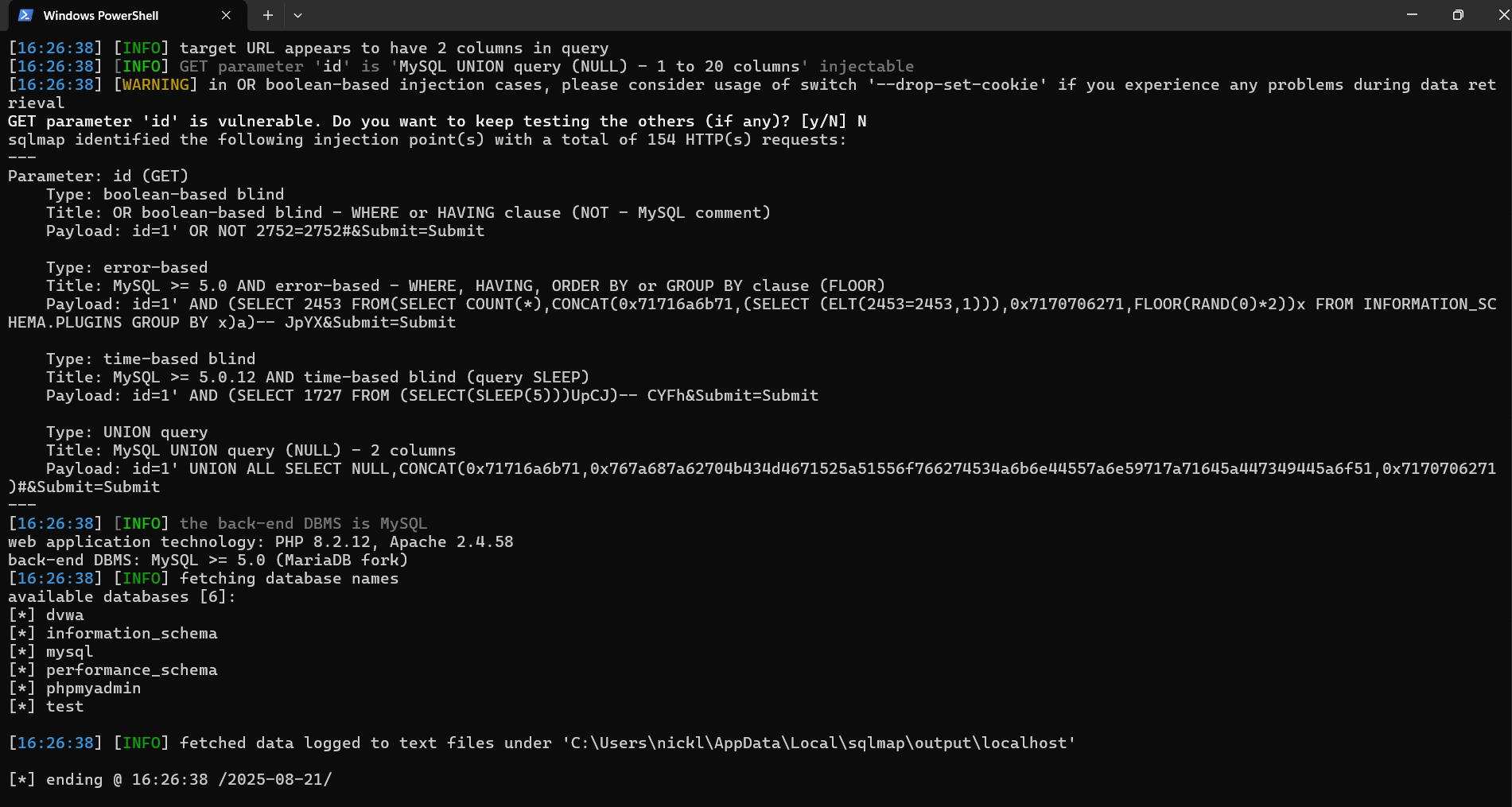


Figure 2–3: XSS payload execution evidence (alert popups / reflected output).

Steps to Reproduce:

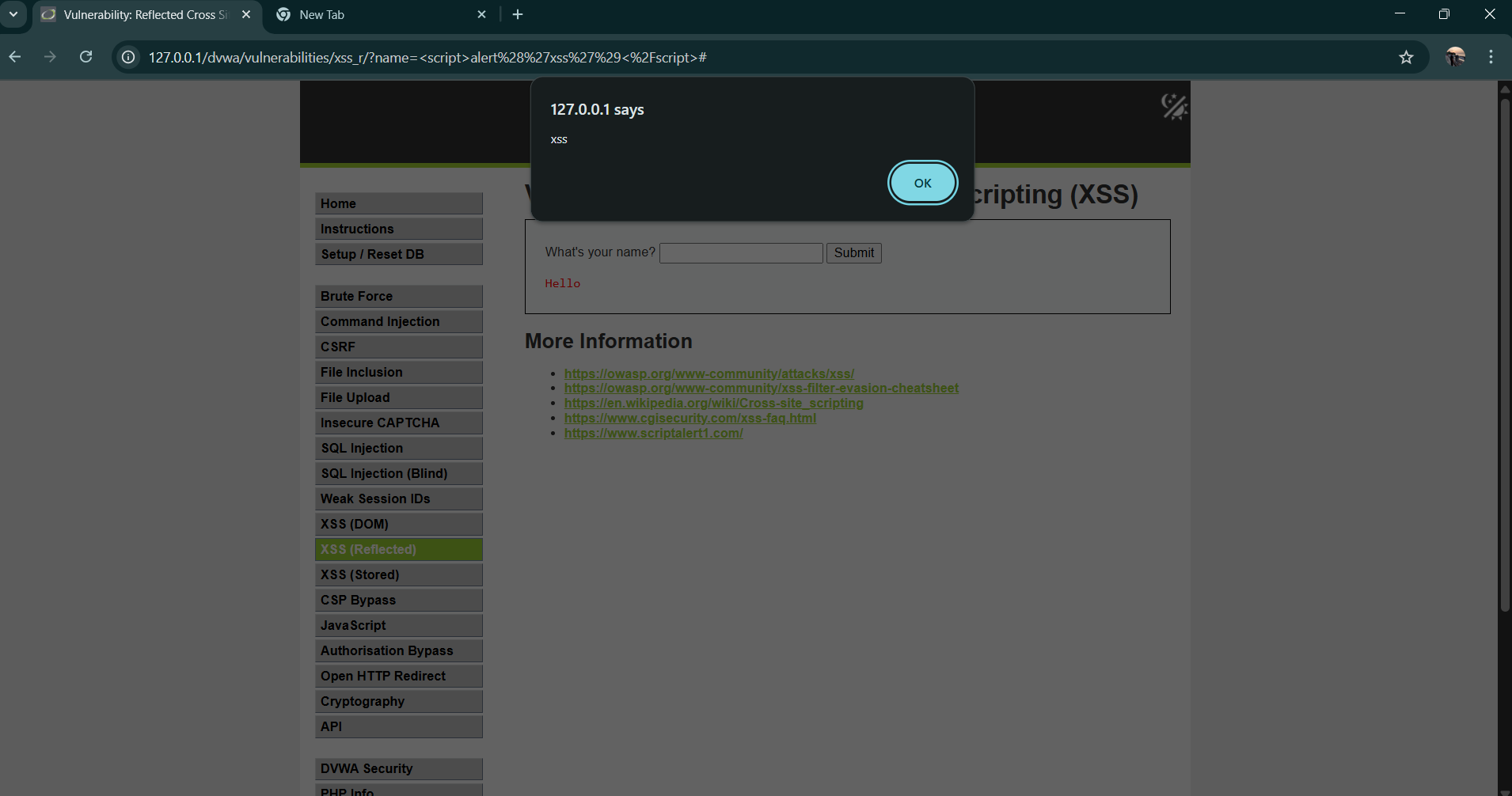
1) Navigate to XSS (Reflected) and submit payload: <script>alert('xss')</script>  
2) For XSS (Stored), post the payload in the message/comment and reload to verify persistence.  
3) Capture screenshots of the executed payloads.

Remediation: Apply context-aware output encoding (HTML/attribute/JS/URL), validate inputs, and enforce a strict Content Security Policy (CSP).

## 3.3 Authentication & Session Management — Observations (Medium/High)

• OWASP Category: A07:2021 – Identification and Authentication Failures  
• Description: Weak authentication and insufficient cookie protections increase the risk of account takeover.  
• Impact: Session hijacking, unauthorized access.

Evidence:



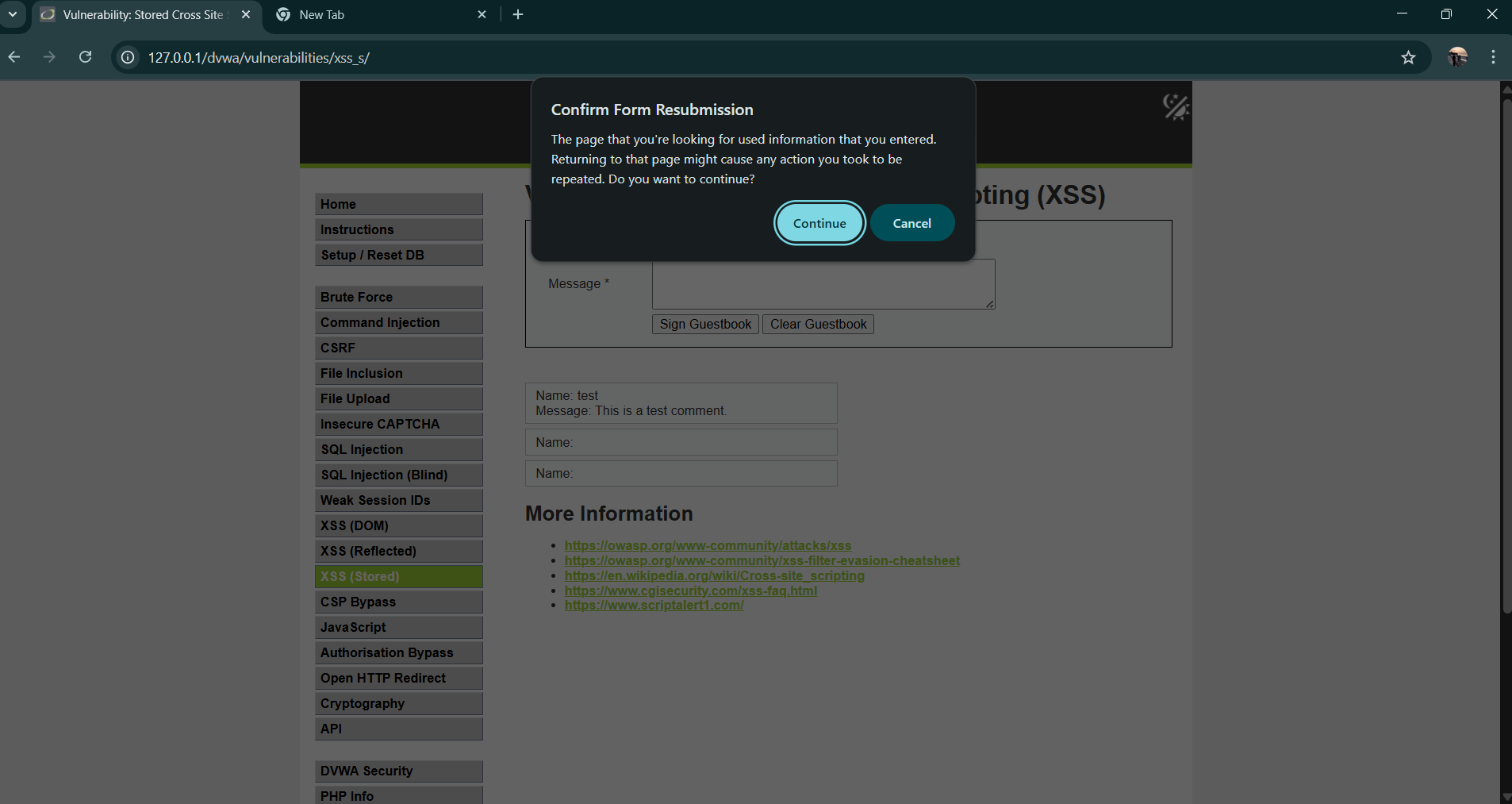


Figure 4–5: Authentication/session evidence (e.g., cookies without Secure/HttpOnly/SameSite).

Suggested Tests: Attempt weak/default passwords, check cookie flags (HttpOnly, Secure, SameSite), verify logout invalidates sessions, and observe rate-limiting/lockout behavior.

Remediation: Enforce strong password policy and lockout, enable MFA, set secure cookie flags, shorten idle timeouts, and invalidate sessions server-side on logout.

# 4. OWASP Top 10 Mapping

|  |  |  |  |
| --- | --- | --- | --- |
| Vulnerability | OWASP Top 10 (2021) | Risk Level | Status |
| SQL Injection | A03: Injection | Critical | ❌ Vulnerable |
| XSS (Reflected/Stored) | A03: Injection (XSS) | High | ❌ Vulnerable |
| Authentication Weaknesses | A07: Identification & Authentication Failures | Medium/High | ⚠️ Findings/Observations |
| CSRF (Not Tested) | A01: Broken Access Control | TBD | ⚠️ Pending |
| Command Injection (Not Tested) | A03: Injection | TBD | ⚠️ Pending |

# 5. OWASP Compliance Checklist

|  |  |
| --- | --- |
| OWASP Top 10 Category | Secure? |
| A01: Broken Access Control | ⚠️ Not Fully Tested |
| A02: Cryptographic Failures | ⚠️ Not Tested |
| A03: Injection | ❌ Found (SQLi, XSS) |
| A04: Insecure Design | ⚠️ Not Tested |
| A05: Security Misconfiguration | ⚠️ Not Tested |
| A06: Vulnerable & Outdated Components | ⚠️ Not Tested |
| A07: Identification & Authentication Failures | ⚠️ Observed Issues |
| A08: Software & Data Integrity Failures | ⚠️ Not Tested |
| A09: Security Logging & Monitoring Failures | ⚠️ Not Tested |
| A10: Server-Side Request Forgery (SSRF) | ⚠️ Not Tested |

# 6. Conclusion

The DVWA assessment confirmed critical Injection risks (SQL Injection and XSS) and observed weaknesses in authentication/session handling. Remediations outlined above should be implemented prior to deployment of any production application with similar patterns. Continued testing across the full OWASP Top 10 is recommended.