Web Application Vulnerability Assessment Report

Target Application: Damn Vulnerable Web Application (DVWA)

Host OS: Kali Linux

Report Prepared By: Rajatava Ghosh

Date: 11.07.2025

Tools Used: OWASP ZAP, Burp Suite, DVWA

📑 Table of Contents

1. Executive Summary

2. Project Overview

3. Scope of Assessment

4. Testing Methodology

5. Summary of Findings

6. Detailed Vulnerabilities

7. OWASP Top 10 Mapping

8. Skills You’ll Gain

9. Conclusion and Recommendations

# 1. Executive Summary

This report presents the results of a vulnerability assessment conducted on the DVWA application hosted on a local Kali Linux VM. The objective was to simulate attacks using OWASP ZAP and Burp Suite to identify major security flaws, aligned with the OWASP Top 10 framework. The report contains screenshots, impact ratings, and mitigation strategies for real-world vulnerabilities.

# 2. Project Overview

This cybersecurity internship project replicates a penetration testing workflow against a web application. It teaches how attackers exploit common web flaws and how to report them effectively. DVWA serves as the testbed for this engagement.

# 3. Scope of Assessment

Tested Application: Damn Vulnerable Web Application (DVWA)  
Host OS: Kali Linux (Local Virtual Machine)  
Tools Used: OWASP ZAP, Burp Suite  
Test Categories: SQL Injection, XSS, CSRF, Command Injection, Authentication Issues

# 4. Testing Methodology

Phase 1 - Scanning: Conducted automated scans using OWASP ZAP and Burp Suite to find potential vulnerabilities.  
Phase 2 - Exploitation: Validated the issues manually and observed their impact.  
Phase 3 - Reporting: Documented each vulnerability with screenshots, risk levels, and recommended mitigations.

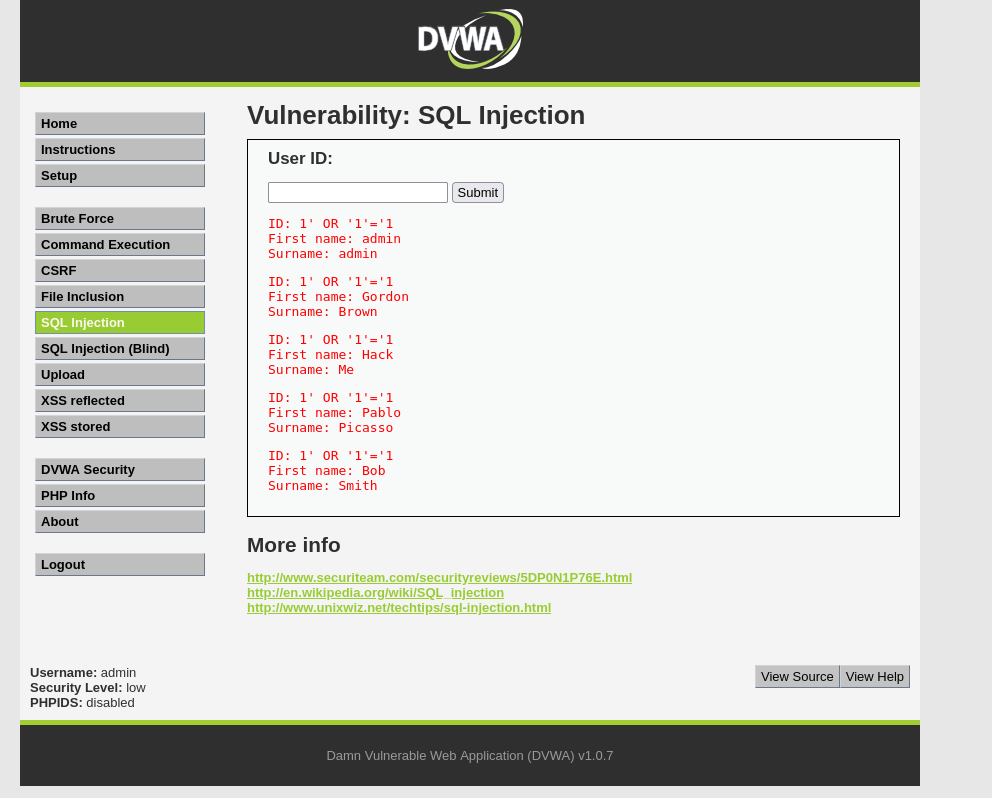
# 5. Summary of Findings

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Vulnerability | Risk Level | Tool Used | OWASP Category |
| 1 | SQL Injection | High | Burp Suite | A1 – Injection |
| 2 | Reflected XSS | Medium | ZAP | A7 – XSS |
| 3 | Stored XSS | High | Manual | A7 – XSS |
| 4 | CSRF | Medium | Manual | A8 – CSRF |
| 5 | Command Injection | High | Burp Suite | A1 – Injection |
| 6 | Cookie Security Issue | Medium | ZAP | A6 – Security Misconfiguration |
| 7 | Missing Anti-CSRF Tokens | Medium | ZAP | A6 – Security Misconfiguration |

# 6. Detailed Vulnerabilities

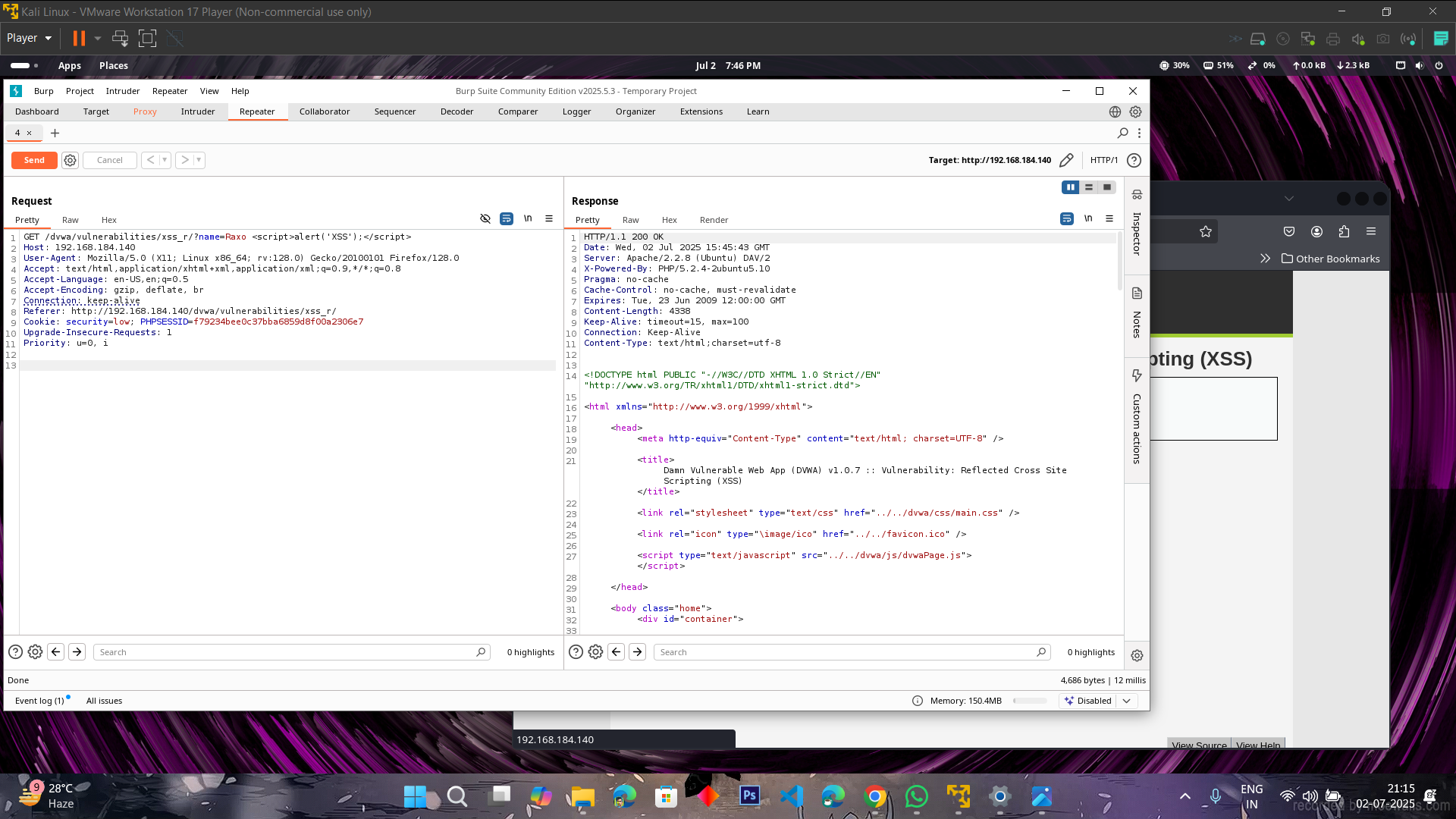
## SQL Injection

Login form input not sanitized, allowing SQL manipulation.  
Risk: High  
Payload: ' OR '1'='1  
Impact: Bypass authentication and dump database.  
Mitigation: Use parameterized queries and input validation.



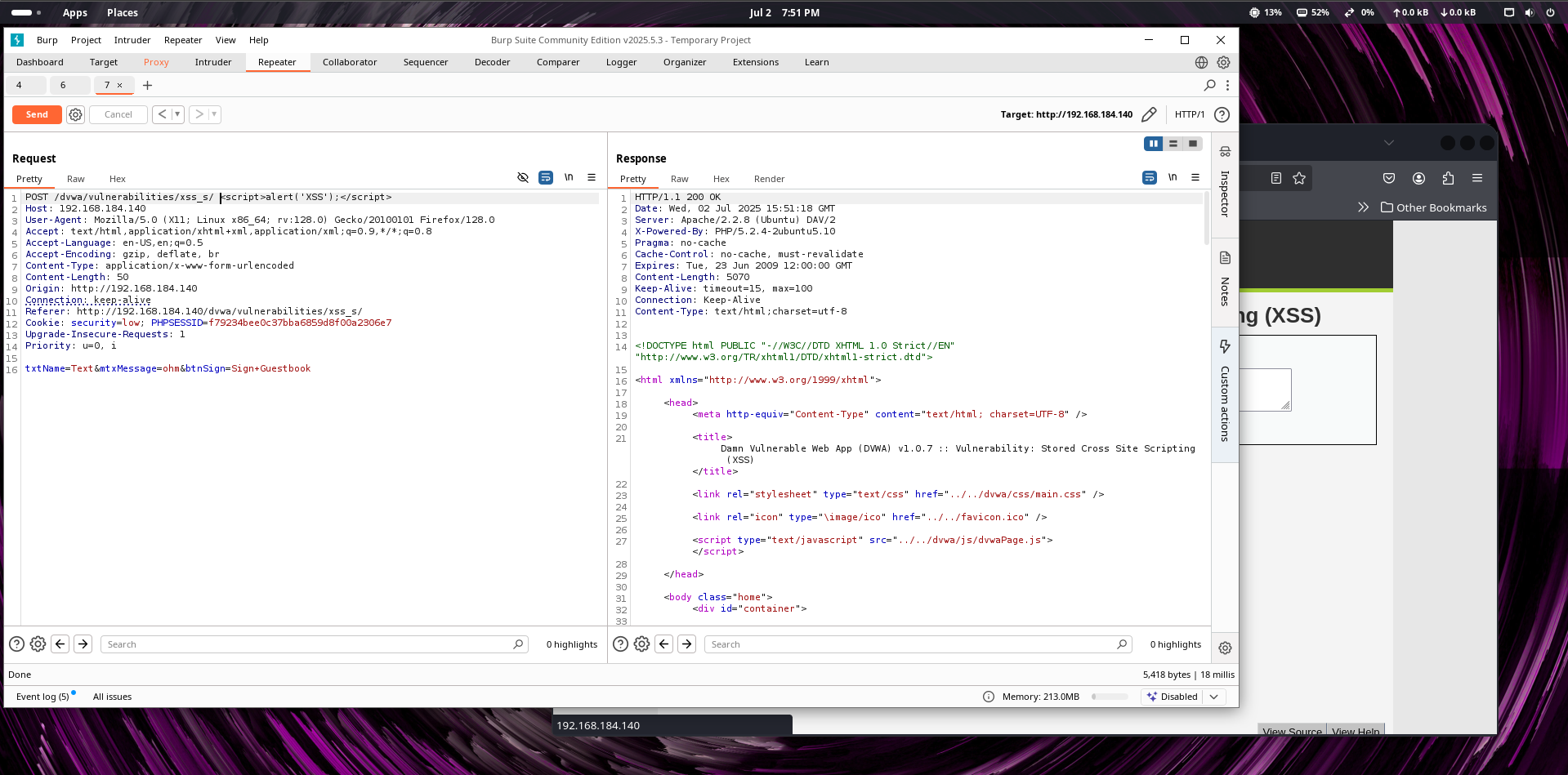
## Reflected XSS

Input reflected in URL without sanitization.  
Risk: Medium  
Impact: Cookie theft, phishing.  
Mitigation: Encode output, use CSP, validate input.



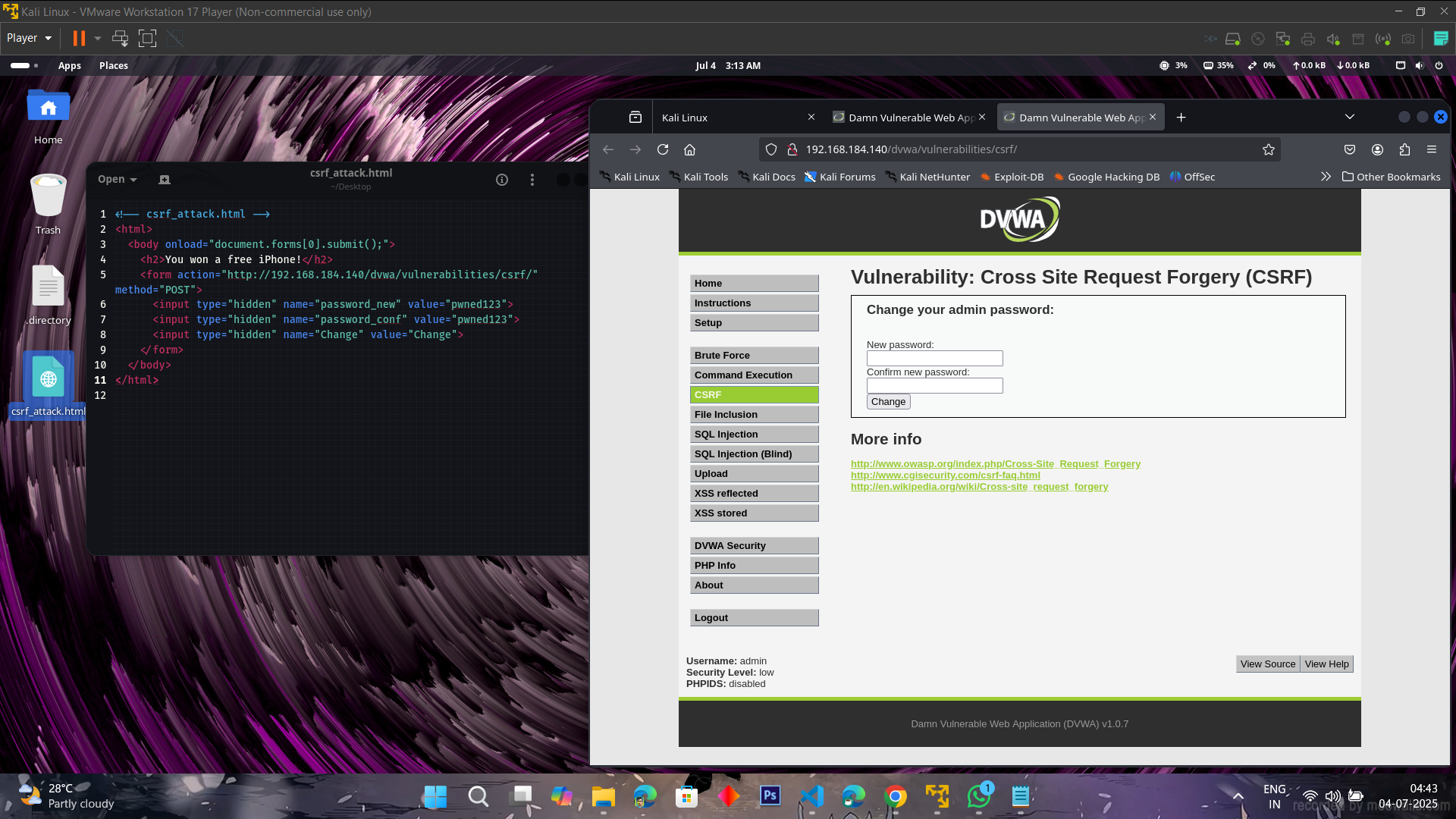
## Stored XSS

Malicious script stored in comment fields.  
Risk: High  
Impact: Persistent attack on other users.  
Mitigation: Sanitize inputs, use secure rendering libraries.



## CSRF

Lack of token validation in sensitive forms.  
Risk: Medium  
Impact: Unauthorized state change (e.g. password reset).  
Mitigation: Use CSRF tokens, SameSite cookies, re-authentication.



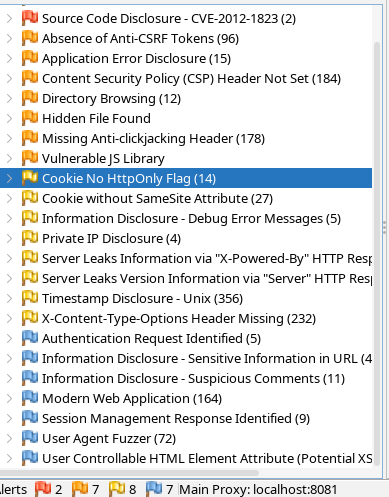
## Command Injection

User input directly executed in system commands.  
Risk: High  
Impact: Remote Code Execution (RCE).  
Mitigation: Sanitize input, avoid unsafe calls, use least privilege.



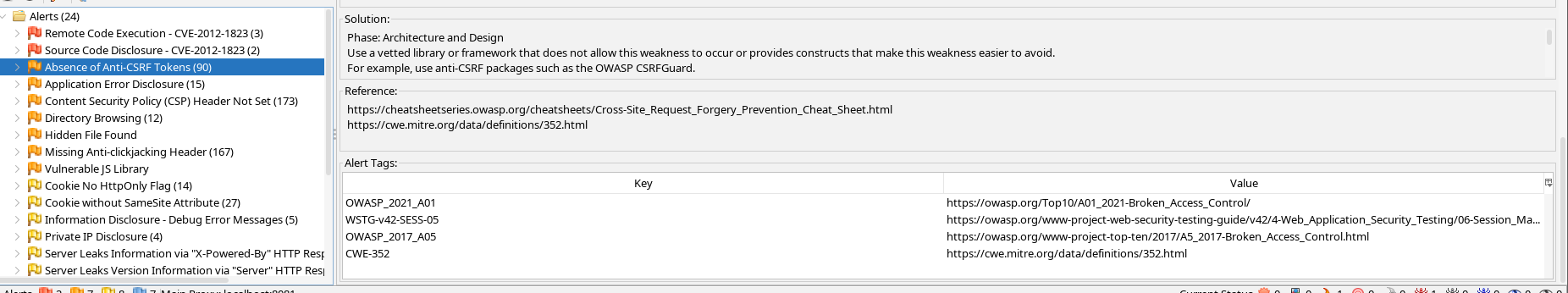
## Cookie Security Issue

Missing HttpOnly and SameSite attributes in cookies.  
Risk: Medium  
Impact: Session hijacking.  
Mitigation: Set secure cookie flags.



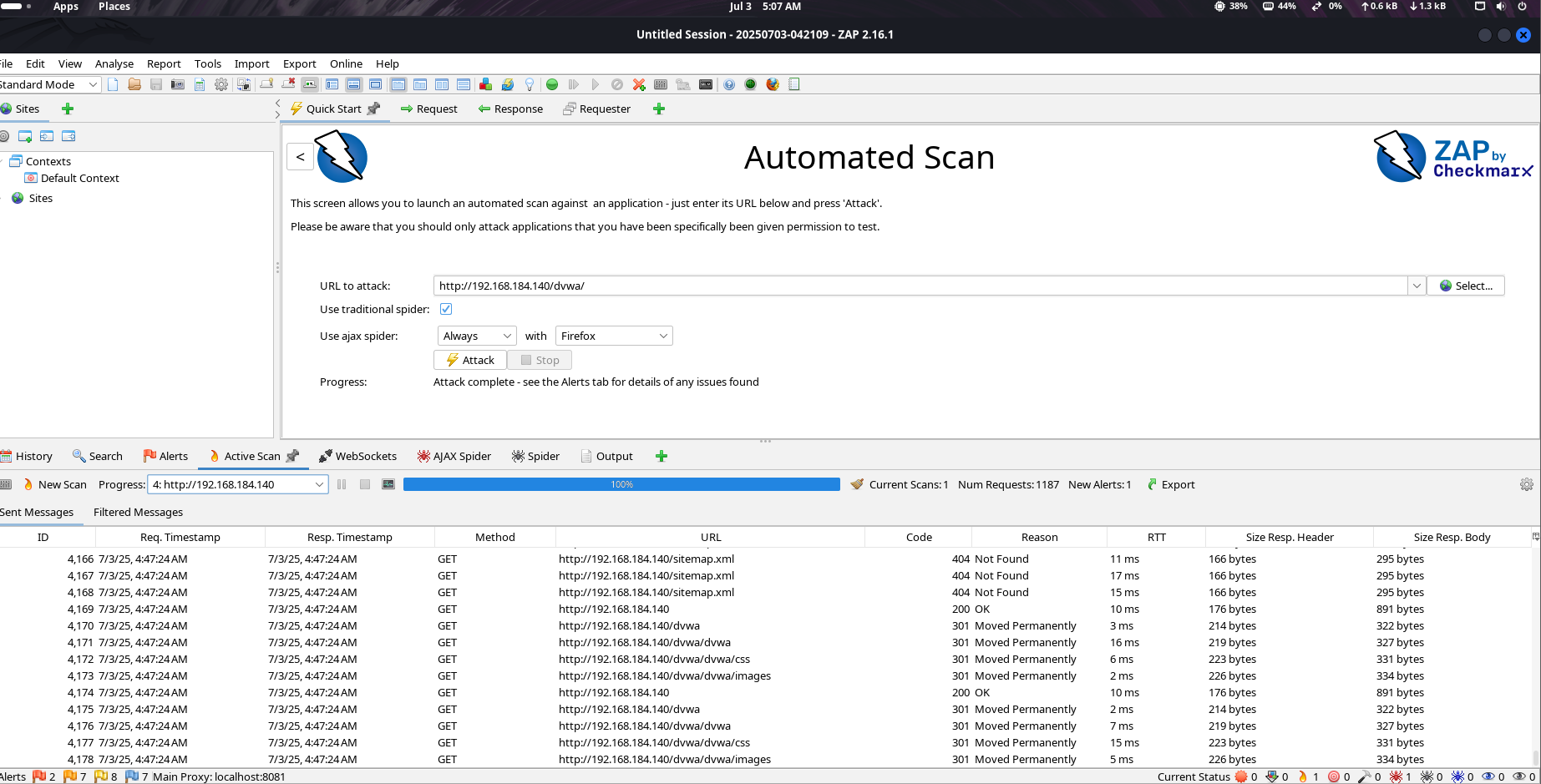
## Missing Anti-CSRF Tokens

Critical forms lack CSRF tokens.  
Risk: Medium  
Impact: Unauthorized requests can be submitted.  
Mitigation: Implement CSRF tokens with frameworks.



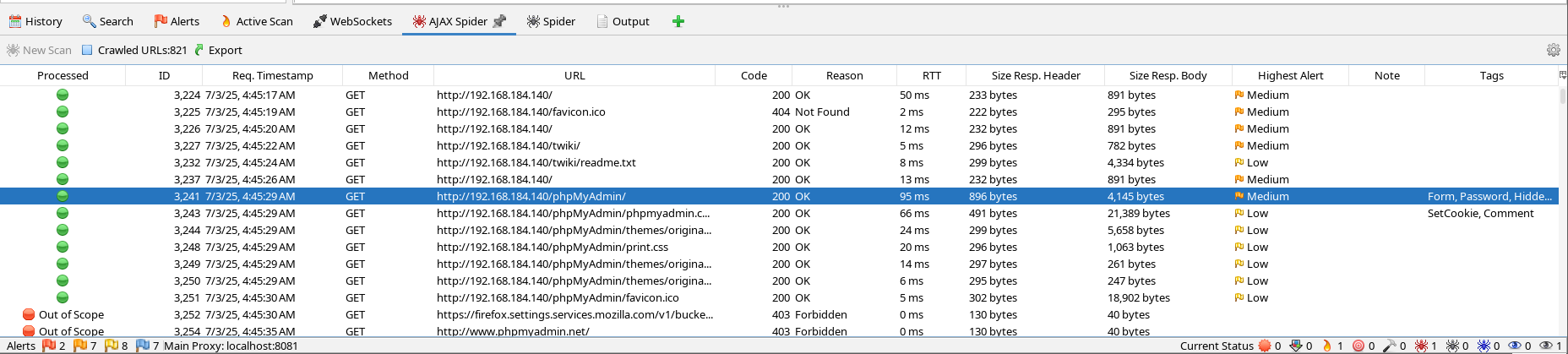
**OWASP ZAP Scan Results**

The scan identified multiple issues including insecure headers and exposed directories. Mitigation: Review and harden web server configurations.



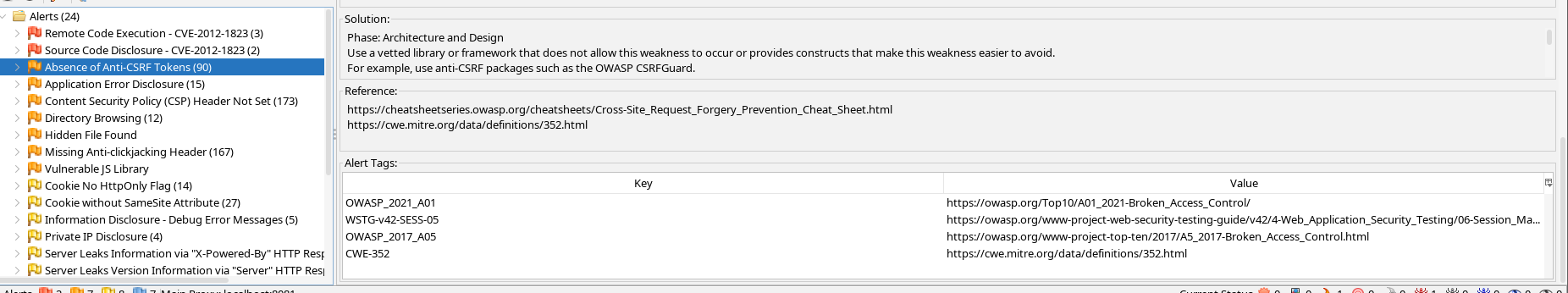
**ZAP AJAX Spider Results**

AJAX spidering revealed hidden routes and files that could be sensitive. Mitigation: Restrict access to admin or internal URLs and monitor endpoint exposure.



**Absence of Anti-CSRF Tokens**

The application lacks anti-CSRF tokens which leaves it vulnerable to cross-site request forgery attacks. Mitigation: Implement CSRF tokens using secure libraries or frameworks.



# 7. OWASP Top 10 Mapping

|  |  |  |
| --- | --- | --- |
| OWASP Category | Detected? | Comments |
| A1 – Injection | ✅ | SQL Injection, Command Injection found |
| A2 – Broken Authentication | ⚠️ | Login mechanism lacks brute-force protection |
| A3 – Sensitive Data Exposure | ❌ | Not tested |
| A4 – XML External Entities (XXE) | ❌ | Not applicable |
| A5 – Broken Access Control | ⚠️ | Potential privilege escalation not tested |
| A6 – Security Misconfiguration | ✅ | Missing headers, cookie flags |
| A7 – Cross-Site Scripting (XSS) | ✅ | Both Reflected and Stored XSS |
| A8 – Insecure Deserialization | ❌ | Not detected |
| A9 – Components with Known Vulnerabilities | ❌ | Not tested |
| A10 – Insufficient Logging & Monitoring | ❌ | Not evaluated |

# 8. Skills You’ll Gain (Detailed Overview)

## 🔓 Web application vulnerability scanning

You will learn how to utilize industry-standard tools such as OWASP ZAP and Burp Suite to discover and analyze web application vulnerabilities. This includes scanning for issues like SQL Injection, Cross-Site Scripting (XSS), and insecure HTTP headers. You'll gain the technical ability to assess web apps just like professional penetration testers.

## 📄 Security documentation and reporting

Gain hands-on experience in writing professional security reports, including vulnerability descriptions, screenshots, impact levels, and remediation steps. Proper documentation is essential for clearly communicating findings to developers, product managers, and security teams.

## ⚔️ Knowledge of OWASP Top 10 threats

You’ll map each identified issue to categories in the OWASP Top 10—such as A01: Broken Access Control, A03: Injection, and A07: Identification and Authentication Failures. This framework is a global standard for prioritizing web application risks.

## 🧠 Basic ethical hacking and penetration testing

Develop a strong foundation in ethical hacking techniques such as manual exploitation, automated scanning, and test-case creation. You'll understand the attacker’s mindset and learn how to ethically discover and report real-world vulnerabilities.

## 🔍 Threat modeling and risk analysis

Learn how to analyze threats, assess potential business impact, and prioritize risks based on likelihood vs. severity. You'll understand how to model an application's attack surface, anticipate high-risk vectors, and communicate the urgency of specific threats effectively.

# 9. Conclusion and Recommendations

The assessment of DVWA uncovered multiple critical vulnerabilities that are representative of real-world risks in production systems. The following actions are recommended:  
  
• Implement secure coding practices (input validation, output encoding)  
• Use frameworks that provide built-in protection against CSRF and XSS  
• Apply security headers and enforce secure cookies  
• Conduct regular security assessments and patch management  
• Educate development teams about the OWASP Top 10 and secure development lifecycle (SDLC)