Task-5 – Pentesting, Exploitation & Incident Response

Submitted By : Pritish Borkar
Internship ID : APSPL2518121
Department : Cybersecurity **Organization :** ApexPlanet software Pvt.Ltd
Co-ordinator : Kundan Kumar
Date of Submission : October 30, 2025

1. Executive Summary

This report documents a comprehensive web application security assessment conducted on Damn Vulnerable Web Application (DVWA). The objective was to learn practical cybersecurity skills by performing ethical exploitation, identifying key vulnerabilities, validating their impact, and applying suitable protection measures in a controlled laboratory environment.

Three high-impact vulnerabilities were tested and successfully exploited:

SQL Injection (SQLi) – Allowed extraction of sensitive database contents

Stored Cross-Site Scripting (XSS) – Executed malicious script on victim browsers

Cross-Site Request Forgery (CSRF) – Forced unauthorized password change

In addition to offensive testing, a simulated Incident Response cycle was executed including log analysis, attacker containment, credential rotation, and post-recovery security hardening.

The outcome of this project demonstrates a complete cybersecurity workflow: exploitation → detection → remediation → reporting. This strengthens both attacker-like thinking and defender-level security awareness.

2. Project Objectives

- ✔ Deploy and configure DVWA in a secure isolated environment
- ✔ Identify and exploit major web vulnerabilities
- ✔ Collect and document attack evidence
- ✔ Simulate organizational response to detected attacks
- ✔ Provide mitigation strategies as per best industry standards
- ✔ Produce a professional penetration testing report and demonstration

3. Ethical Considerations

All tests were performed:

On an intentionally vulnerable system (DVWA)

Inside a fully isolated Kali Linux Virtual Machine

With no connection to production or real-world applications

In alignment with cybersecurity ethics and responsible disclosure principles

This ensures 100% safe and legal execution.

4. Technical Environment

- Component Details
- Operating System Kali Linux VM (latest stable)
- Web Server Apache2 on localhost
- Database Server MariaDB
- DVWA URL http://localhost/

- Tools Used Burp Suite, sqlmap, mysql client, iptables, curl, browser developer tools

5. Setup and Configuration

DVWA was downloaded and deployed under /var/www/html/. Apache and MariaDB services were verified running using Linux service commands.

The DVWA configuration file config.inc.php was updated with a custom database user and password. Using http://localhost/setup.php, the initial database creation was executed. Finally, application login was tested using:

Username: admin

Password: password

Security level was set to Low temporarily to allow exploitation demonstrations.

This ensures the environment properly simulates an insecure legacy web application.

☑ 6. Vulnerability Assessment & Manual Exploitation

The assessment follows OWASP methodology focusing on three critical vulnerabilities. Exploitation was performed directly through the browser and supported by automated tools.

6.1 Vulnerability 1 – SQL Injection (Severity: High)

■ Description

SQL Injection occurs when unsanitized user inputs are directly added into SQL queries. Attackers can inject malicious SQL code to manipulate databases, bypass logins, or steal sensitive records.

■ Evidence of Vulnerability

Test Field: ID input parameter on /vulnerabilities/sqli/.

Manual Payloads used:

' – triggered SQL syntax errors

1 OR 1=1 – extracted multiple database rows

These responses confirm that the server executed injected SQL code.

Automated Extraction:

Using sqlmap tool, attackers can extract full database contents including username/password pairs.

Real-world impact:

1. Database breach
2. Identity theft
3. Admin account takeover
4. Complete system compromise

■ Remediation Recommendations

- ☑ Use Prepared Statements and Parameterized Queries
- ☑ Apply strict server-side input validation
- ☑ Escape special characters and integers only in numeric fields
- ☑ Implement Minimum privilege principle for DB user
- ☑ Log and monitor all SQL errors centrally

SQLi remains one of the most dangerous web vulnerabilities according to OWASP Top 10.

6.2 Vulnerability 2 – Stored Cross-Site Scripting (Severity: Medium-High)

■ Description

Stored XSS allows attackers to store malicious JavaScript inside the database. When a victim later views this stored content, the script executes within their browser session.

Tested page: DVWA XSS (Stored) module.

■ Evidence of vulnerability

Payload used:

<script>alert('XSS_TEST')</script>

Result:

Code got stored permanently in the message tabl

Any user revisiting the page saw a popup alert

Script executed with same browser privileges as the victim user

Real-world impact:

1. Cookie and session stealing
2. Auto-actions inside victim browser
3. Phishing redirection or keylogging
4. Complete account takeover without login credentials

■ Remediation Recommendations

- ☑ Sanitize and encode output (HTML encoding of <script>)
- ☑ Apply Content Security Policy (CSP) to control script execution
- ☑ Validate both input and stored database content
- ☑ Secure cookies using HttpOnly & Secure flags

Stored XSS is commonly seen in commenting systems and feedback forms.

6.3 Vulnerability 3 – Cross-Site Request Forgery (Severity: Medium-High)

■ Description

CSRF attacks force logged-in users to perform unauthorized actions without their knowledge — e.g., changing passwords — while authenticated via stored cookies.

Tested page: DVWA CSRF Vulnerability module

■ Exploitation Approach

A custom malicious page csrf_attack.html was placed inside /var/www/html/:

When a logged-in admin simply visits this HTML page,

A hidden form auto-submits and

The admin password changes silently to a new value hacked123

■ Real-world impact:

1. Account compromise of authenticated users
2. Unauthorized money transfers
3. Setting changes without user interaction
4. Very high risk if admin accounts affected

■ Remediation Recommendations

- ☑ Add Anti-CSRF tokens for every state-changing request
- ☑ Validate Origin and Referer headers
- ☑ Mark session cookies as SameSite=Strict
- ☑ Require re-authentication before sensitive actions

CSRF attacks combine deception with technical browser weaknesses.

7. Incident Response Simulation

This segment demonstrates how security teams respond to detected intrusions.

Concept used: IR Lifecycle — Detect → Contain → Eradicate → Recover → Learn

7.1 DETECT – Log Examination

Apache logs were reviewed for suspicious entries:

SQLi payload indicators (' OR 1=1)

Script tags (<script>)

Abnormal POST requests from local attacker

Logs provide undeniable proof of exploitation attempts.

7.2 CONTAIN – Stopping Attacker

Firewall rule example:

sudo iptables -A INPUT -s <attacker-ip> -j DROP

This blocks further communication from suspected attacker origin.

7.3 ERADICATE – Remove malicious components

- Deleted CSRF attack files from server directory
- Database credentials were rotated
- Updated DVWA configuration and cleaned logs
- This removes attacker persistence.

7.4 RECOVER – Restore Security state

1. Raised DVWA Security level to High
2. Re-tested all previously vulnerable modules
3. Continued monitoring after patching
4. System returns to safe operation.

7.5 LESSONS LEARNED

- Proactive improvements include:
- Secure coding culture
- Regular vulnerability assessments
- Continuous monitoring with alerts
- Defense-in-depth layered protection
- This builds long-term cyber resilience.

8. Final Risk Ratings

- Vulnerability Likelihood Impact Final Risk
- SQL Injection Very High Very High Critical
- Stored XSS High High High
- CSRF Medium High High

9. Deliverables Provided

- ✔ DVWA exploitation demonstration
- ✔ Full evidence logs and terminal outputs
- ✔ Report PDF & GitHub repository upload
- ✔ Video narration script & incident response documentation
- ✔ Remediation guidance according to OWASP Top-10

10. Conclusion

This DVWA Task-5 project helped demonstrate real-world exploitation patterns and reinforced secure development practices.

- Key cybersecurity learnings from this assignment:
- Attackers exploit simple validation mistakes with severe consequences
- Lack of proper input handling leads directly to SQL Injection and data breach
- Stored XSS proves the importance of controlling what any user-generated content can do
- CSRF shows that trust in cookies and logged-in sessions can be misused
- Offensive testing helps create better defensive strategies
- Incident Response must be swift, structured, and evidence-driven

The skills practiced in this assignment — exploitation, detection, containment, and remediation — are essential for cybersecurity analysts, penetration testers, SOC analysts, and web security engineers.

In closing, this report highlights a complete cybersecurity lifecycle:

Find → Exploit → Confirm risk → Take defensive action → Strengthen the application

Thank you for reviewing this work. This project reflects my current capabilities in ethical hacking and secure system administration. I look forward to expanding these skills further through future cybersecurity challenges, CTF competitions, and advanced certifications.