SQL Injection Penetration Test Report: DVWA

Target: Damn Vulnerable Web Application (DVWA)

1. Executive Summary

For Non-Technical Management

This report summarizes the findings of an active penetration test focused on **SQL Injection (SQLi)** vulnerabilities. The assessment confirmed that poor database query construction allows attackers to bypass security controls and access unauthorized information.

Key Finding: The target application, when set to low and medium security levels, was successfully exploited using common SQLi payloads, resulting in the unauthorized extraction of user data and successful login without credentials.

Overall Risk: CRITICAL. SQL Injection is one of the most severe web vulnerabilities, often leading directly to complete database compromise, sensitive data exposure (e.g., user records, proprietary business information), and potentially full system takeover.

Immediate Recommendation: The development team must immediately adopt modern, secure coding practices, specifically implementing **Parameterized Queries (Prepared Statements)** across all database interactions.

2. Scope and Authorization

Target: The SQL Injection and login functionality within a controlled testing environment (DVWA).
Authorization: Explicitly authorized for destructive and exploitative testing.
Security Focus: Testing the effectiveness of input validation and query construction at Low and Medium security levels.

3. Assessment Objectives

1. **Exploit Low Security:** Achieve unauthorized access (authentication bypass) with minimal effort.
2. **Exploit Medium Security:** Bypass basic server-side filtering to demonstrate insufficient defense.
3. **Prove Data Leakage:** Successfully extract data from the database, confirming full compromise.
4. **Parameter Tampering:** Identify and exploit any other vulnerable GET/POST parameters.

4. Key Finding Summary: Data Leakage and RCE Risk

SQL Injection is the single most critical finding. A successful exploit allows the attacker to fundamentally change the application's logic by manipulating the database query.

Critical Implication: The attacker gains read/write access to the entire database, leading to **Data Confidentiality** (theft of all records) and **Integrity** (alteration or deletion of records) compromise. If the database user has high privileges, this can escalate to **Remote Code Execution (RCE)** on the server.

5. Detailed Findings and Vulnerabilities

ID	Finding/Exploit Method	Security Level	Severity	Simple Explanation
S-01	**Trivial Authentication Bypass**	Low	**CRITICAL**	Application uses simple string concatenation in the login query, allowing login with ' OR '1'='1 payload.
S-02	**Bypassed Input Filter**	Medium	**HIGH**	The application attempts basic filtering (blacklisting), which was defeated by using mixed case or specialized encoding, proving the filter is ineffective.
S-03	**Union-Based Data Extraction**	Low/Medium	**CRITICAL**	Successful use of UNION SELECT

				statement to exfiltrate other users' data, proving complete read access to the database.

6. Risk Assessment Matrix

The risk of SQL Injection is categorized as Critical due to the direct impact on data integrity and confidentiality.

Finding ID	Likelihood	Impact	Overall Risk	Justification
S-01, S-03	High	Critical	**CRITICAL**	Very easy to execute, leading to full database compromise and data theft.
S-02	Medium	High	**HIGH**	Requires slightly more skill but results in the same critical database access.

7. High-Level Remediation Strategy

The solution requires a fundamental shift in how the application interacts with its database.

Priority	Recommendation	Action to Be Taken
1. CRITICAL	**Implement Parameterized Queries**	All database calls **must** use Prepared Statements (e.g., PDO in PHP) to separate the query structure from

		user input. This makes SQLi impossible.
2. HIGH	**Strict Input Validation**	Implement a "whitelisting" approach, ensuring all user input strictly matches expected formats (e.g., only numbers for an ID field).
3. MEDIUM	**Least Privilege Principle**	The application's database user account should only have permissions necessary for its function (e.g., NO DELETE or DROP table permissions).

8. Proposed Remediation Timeline

Priority	Finding	Proposed Target Completion	Responsible Team
P-1 (Critical)	S-01, S-03 (Prepared Statements)	**Immediately / 2 Weeks**	Development Team Lead
P-2 (High)	S-02 (Input Validation)	4 Weeks	Development Team
P-3 (Medium)	Server Configuration / Least Privilege	6 Weeks	DevOps / Database Admin

9. Distribution and Confidentiality

This report details sensitive security flaws discovered during active exploitation.

- **Distribution:** Restricted to authorized management and security personnel only.
- **Confidentiality:** This document is classified as **"Strictly Confidential"** and must not be shared externally.

10. Appendix: Technical Component Details

The full technical details, including the specific SQL payloads used, the step-by-step process for bypassing the medium security filter, and raw command screenshots, are provided in the companion report: **Technical_Report.md** and the uploaded evidence file **Technical Report.pdf**.