

Web Application Security Testing Report

Intern Name: K. Baby Tejaswi

Project Title: Web Application Security Testing

Tool Used: Damn Vulnerable Web Application (DVWA)

Track id: FUTURE_CS_01

1. Introduction

- **Objective:** Assess DVWA for vulnerabilities, document exploitation methods, and recommend mitigations.
- **Scope:** Focused on Brute Force, SQL Injection (normal & blind), and XSS (DOM-based, Reflected, Stored).
- **Tools Used:** Burp Suite, SQL Map, Foxy Proxy, OWASP ZAP, Kali Linux VM.

2. Methodology

1. **Environment Setup:** DVWA deployed on Kali Linux VM, database configured, and application accessed locally.
2. **Attack Simulation:** Conducted targeted attacks for Brute Force, SQL Injection, and XSS.
3. **Vulnerability Verification:** Confirmed vulnerabilities by analyzing responses and system behavior.
4. **Mapping & Mitigation:** Correlated findings with OWASP Top 10 and proposed remediation strategies.

3. Key Findings

3.1 Brute Force Attack

- **Observation:** Weak authentication allowed automated credential guessing via Burp Suite Intruder.
- **Impact:** Unauthorized access to accounts.

3.2 SQL Injection

- **Normal SQL Injection:** Payload ' OR '1'='1 successfully bypassed authentication.
- **Blind SQL Injection:** SQLMap used to enumerate databases, extract users table, and crack hashed credentials.
- **Impact:** Full database compromise possible.


3.3 Cross-Site Scripting (XSS)

- **DOM-based XSS:** Unsanitized DOM manipulation executed arbitrary JavaScript (alert(document.cookie)).

- **Reflected XSS:** Application echoed unvalidated input, enabling client-side code execution.
- **Stored XSS:** Malicious scripts persisted in guestbook entries and executed in other users' browsers.
- **Impact:** Session hijacking, data theft, and user impersonation.

4. OWASP Top 10 Mapping

OWASP Category	Finding in DVWA
A1 – Injection	SQL Injection vulnerabilities detected.
A2 – Broken Authentication	Brute force attacks possible without restrictions.
A5 – Broken Access Control	Functions accessible without proper checks.
A6 – Security Misconfiguration	Default insecure configurations observed.
A7 – Cross-Site Scripting	DOM, Reflected, and Stored XSS confirmed.
A9 – Components with Known Vulns	DVWA uses intentionally outdated components.
A10 – Insufficient Logging/Monitoring	No alerts or logs captured during attacks.



- Home
- Instructions
- Setup / Reset DB
- Brute Force**
- Command Injection
- CSRF
- File Inclusion
- File Upload
- Insecure CAPTCHA
- SQL Injection
- SQL Injection (Blind)
- Weak Session IDs
- XSS (DOM)
- XSS (Reflected)
- XSS (Stored)
- CSP Bypass
- JavaScript

Vulnerability: Brute Force

Login

Username:

Password:

Login

Username and/or password incorrect.

More Information

- https://owasp.org/www-community/attacks/Brute_force_attack
- <https://www.symantec.com/connect/articles/password-crackers-ensuring-security-your-password>
- <https://www.g0tmi1k.com/brute-force-attack-web-forms>

a)Brute force


Capture filter: Capturing all items

View filter: Showing all items

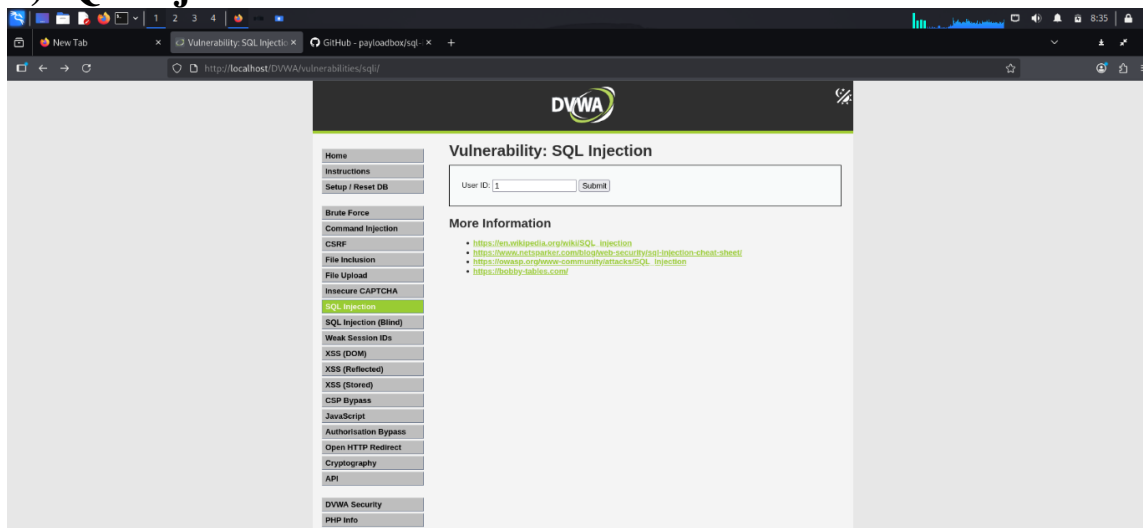
Request	Payload 1	Payload 2	Status code	Response received	Error	Timeout	Length	Comment
11	admin	password	200	6			5033	
0			200	4			5030	
2	cyber	admin	200	2			5030	
4	12345	admin	200	5			5030	
7	cyber	passwe	200	3			5030	
9	12345	passwe	200	4			5030	
13	passwoerf	password	200	2			5030	
15	aksa	password	200	1			5030	
17	cyber	3243425	200	3			5030	

request Response

pretty Raw Hex Render



b)SQL injection

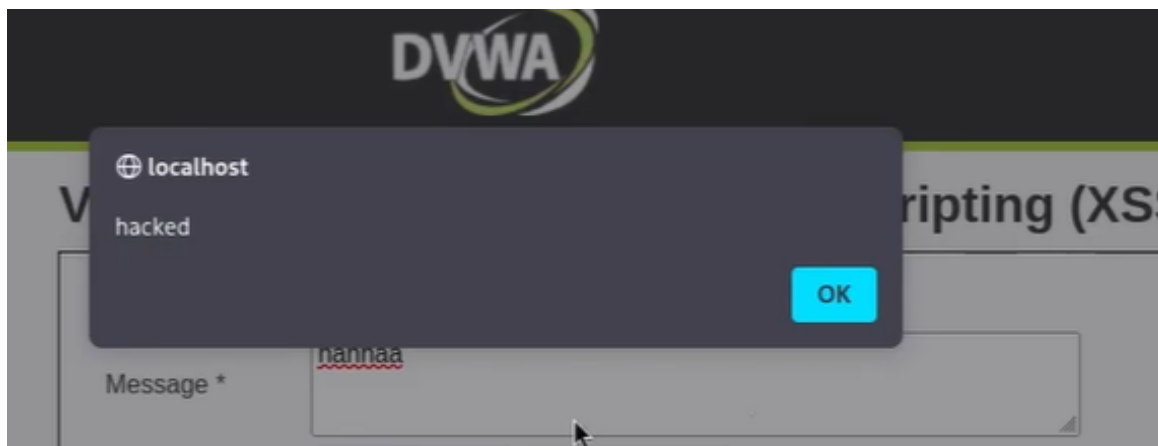
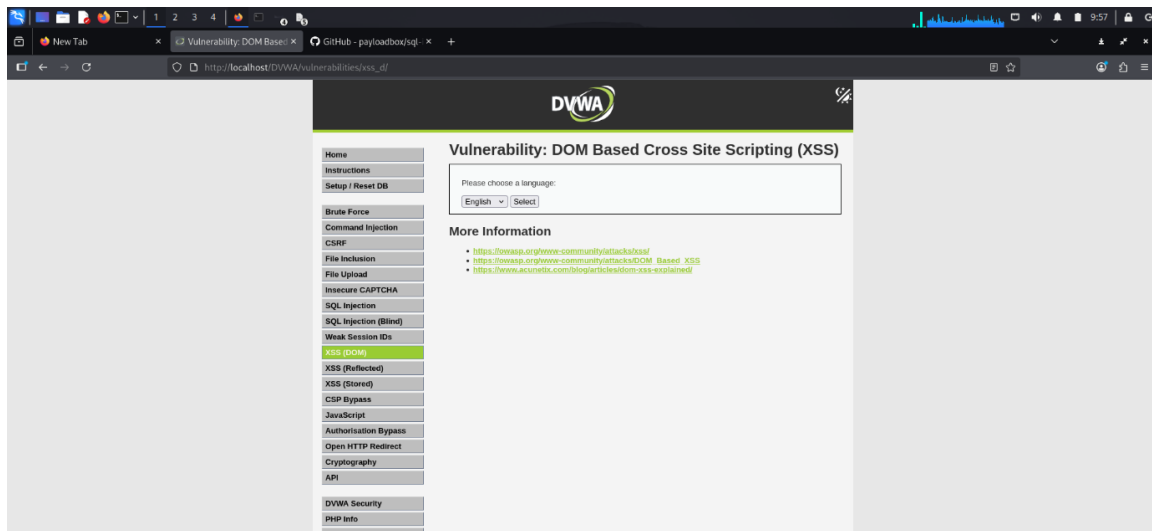


```
Database: dvwa
Table: users
[5 entries]
+-----+-----+
| user_id | password |
+-----+-----+
| 1       | 21232f297a57a5a743894a0e4a801fc3 (admin) |
| 2       | e99a18c428cb38d5f260853678922e03 (abc123) |
| 3       | 8d3533d75ae2c3966d7e0d4fcc69216b (charley) |
| 4       | 0d107d09f5bbe40cade3de5c71e9e9b7 (letmein) |
| 5       | 5f4dcc3b5aa765d61d8327deb882cf99 (password) |
+-----+-----+
```

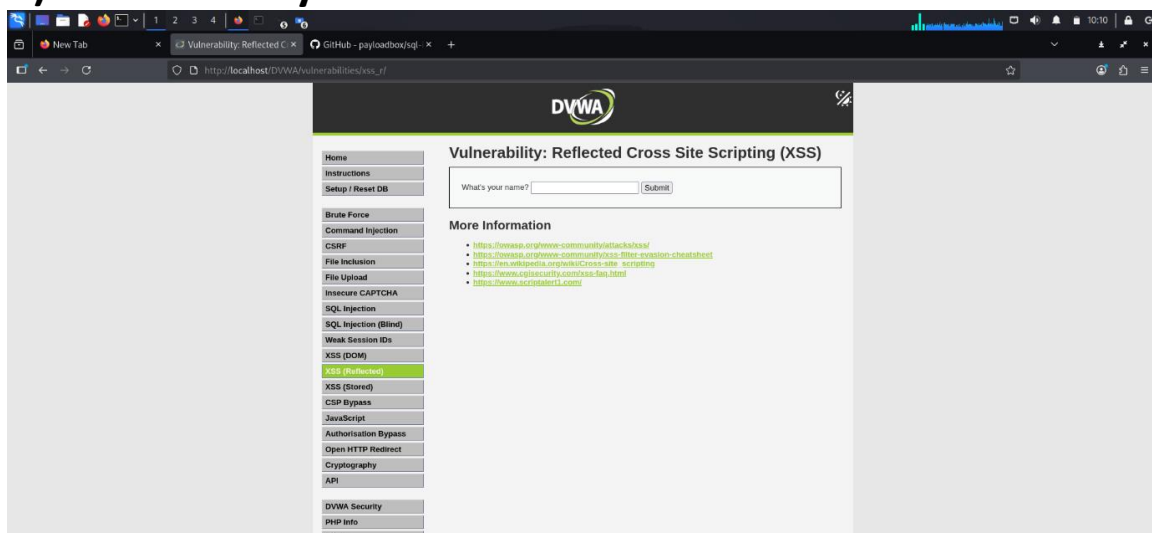
```
[06:43:32] [INFO] table 'dvwa.users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/localhost/dump/dvwa/users.csv'
[06:43:32] [WARNING] HTTP error codes detected during run:
404 (Not Found) - 683 times
[06:43:32] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/localhost'

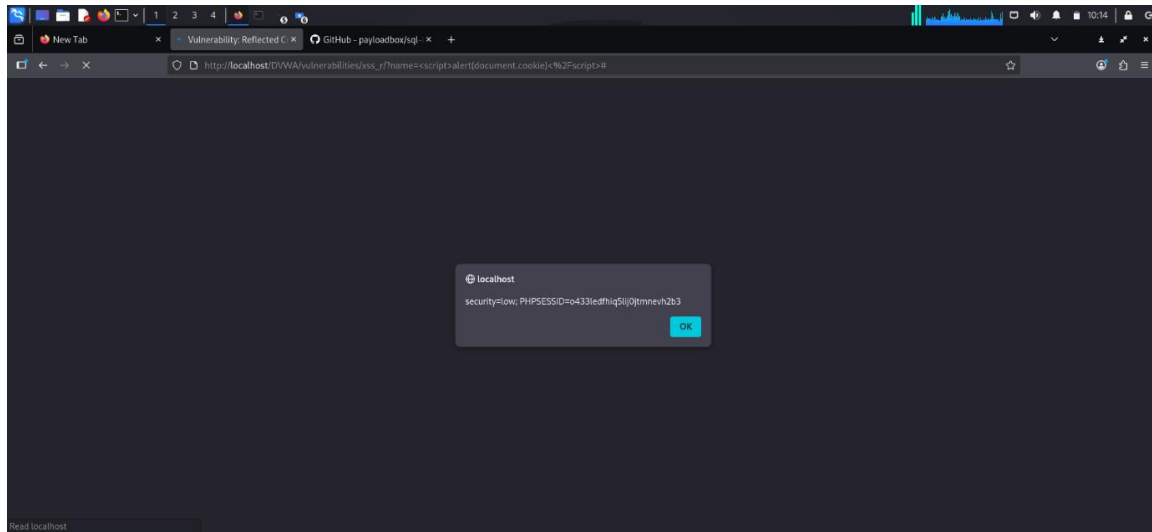
[*] ending @ 06:43:32 /2025-06-20/
```

a) Vulnerability: DOM Based XSS

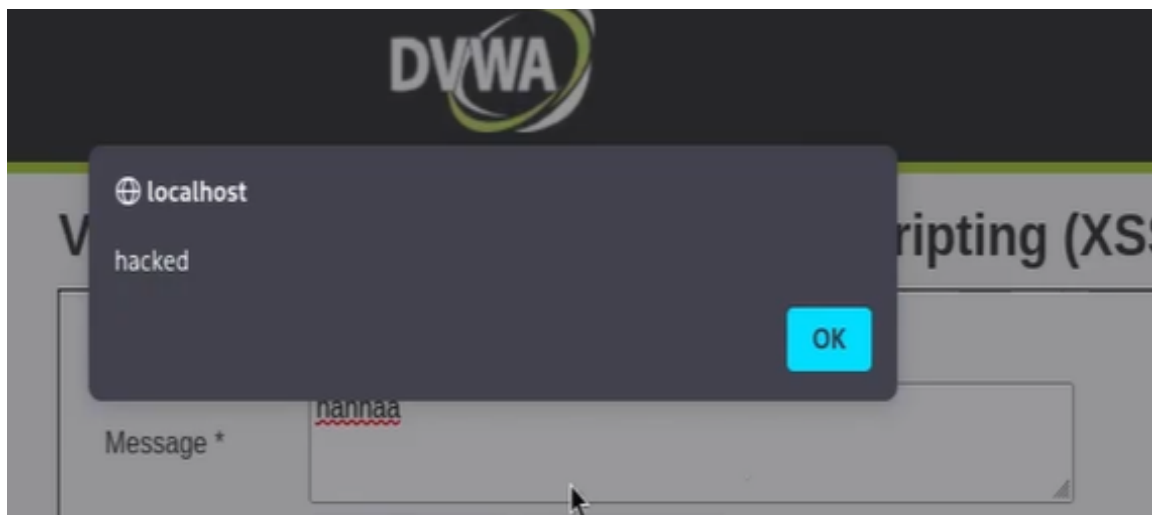
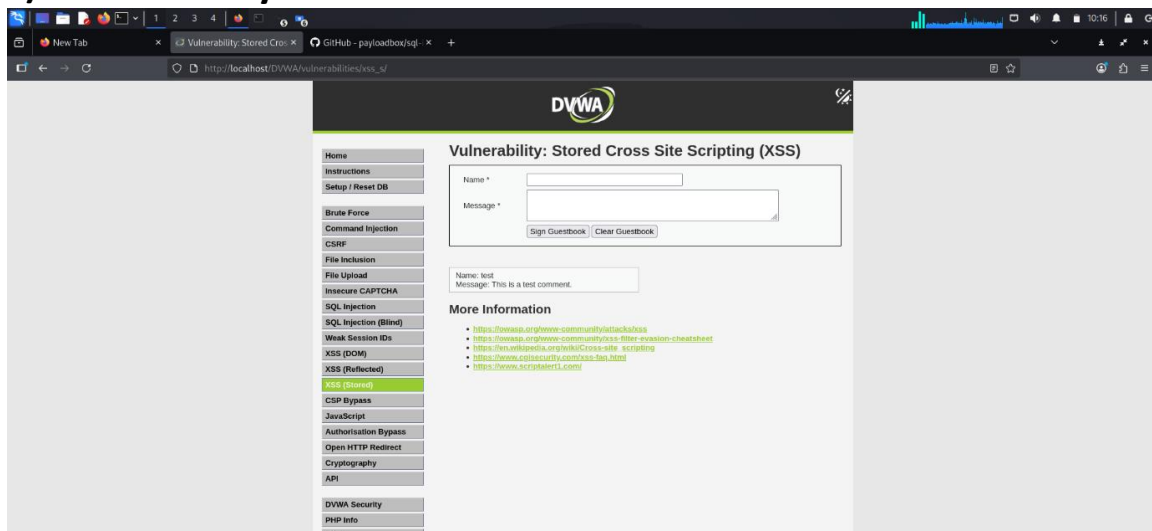


b) Vulnerability: Reflected XSS





c) Vulnerability: Stored XSS



Mitigation Strategies

Brute Force

- **Implement account lockout after failed login attempts:** Lock a user's account temporarily after several unsuccessful login attempts to deter automated guessing attacks.
- **Use CAPTCHA to prevent automated attacks:** Add CAPTCHA challenges on login forms to distinguish human users from bots.
- **Enforce strong password policies:** Require passwords with sufficient length and complexity to make brute-force attempts less effective.

SQL Injection

- **Use prepared statements and parameterized queries:** Always separate SQL code from user input so that attackers cannot inject malicious SQL commands.
- **Validate and sanitize inputs thoroughly:** Check and clean all user-provided data to ensure it meets expected formats before processing.
- **Employ Web Application Firewalls (WAF):** Use a WAF to detect and block suspicious SQL injection patterns before they reach your application.

Cross-Site Scripting (XSS)

- **Apply output encoding:** Convert special characters in user-supplied data into safe representations before displaying them in the browser to prevent script execution.
- **Implement Content Security Policies (CSP):** Set a CSP header to restrict where scripts can be loaded from, reducing the impact of injected code.
- **Validate and sanitize all user inputs:** Rigorously check and clean user input to ensure that harmful scripts are removed or escaped before processing.

5. Conclusion

The DVWA assessment confirmed critical vulnerabilities that reflect real-world security challenges. While DVWA is intentionally insecure, the findings emphasize the importance of:

- Secure coding practices (input validation, parameterized queries).
- Stronger authentication mechanisms (lockouts, CAPTCHAs, password policies).
- Defense-in-depth strategies (WAF, CSP, monitoring).