Project Title

Web Application Penetration Testing Using DVWA

Submitted By

Name: Akksai Prathaan
Course: Cybersecurity / Ethical Hacking
Platform: Kali Linux
Tool Used: DVWA (Damn Vulnerable Web Application)

---

1. Introduction

Web applications are increasingly targeted by cyber attackers due to vulnerabilities such as improper input validation, weak authentication mechanisms, and poor security configurations. This project focuses on identifying and exploiting common web application vulnerabilities using **DVWA (Damn Vulnerable Web Application)** in a controlled environment.

The objective of this project is to understand how attacks work, analyze their impact, and implement preventive security measures.

---

2. Objectives

- To understand common web application vulnerabilities
- To perform hands-on penetration testing
- To exploit vulnerabilities in a controlled environment
- To understand attack mitigation techniques
- To document findings professionally

---

3. Tools & Technologies Used

Tool	Purpose
Kali Linux	Penetration testing OS
DVWA	Vulnerable web application
Burp Suite	Intercepting & analyzing HTTP traffic
MariaDB	Database backend
Apache	Web server
Browser	Testing and payload execution

---

4. System Architecture

```
Attacker (Kali Linux)
        |
        | HTTP Requests
        v
DVWA Web Application (Apache + PHP)
        |
        v
     MariaDB
```

---

5. Vulnerability Assessment & Exploitation

5.1 SQL Injection

Description:
 SQL Injection allows attackers to manipulate database queries by injecting malicious SQL commands.

Payload Used:

```
' UNION SELECT user(), database() #
```

Result:

- Extracted database name and user credentials.
- Confirmed lack of input validation.

Impact:

- Unauthorized access to sensitive data.

Mitigation:

- Use prepared statements
- Parameterized queries
- Input validation

5.2 Cross-Site Scripting (XSS)

Reflected XSS

Payload:

```
<script>alert('XSS')</script>
```

Stored XSS

Payload:

```
<script>alert('XSS')</script>
```

Result:

- JavaScript executed on page load.
- Persistent vulnerability confirmed.

Impact:

- Session hijacking
- User redirection

- Data theft

Mitigation:

- Output encoding
- Input validation
- Content Security Policy (CSP)

5.3 Command Injection

Payload Used:

```
127.0.0.1; whoami
```

Result:

- Server executed system-level commands.

Impact:

- Full system compromise possible.

Mitigation:

- Avoid system calls
- Use allowlists
- Input sanitization

5.4 Brute Force Attack

Method:

- Used Burp Suite Intruder
- Targeted login form

- Used rockyou.txt wordlist

Result:

- Successfully discovered valid credentials

Impact:

- Unauthorized access to protected resources

Mitigation:

- Account lockout
- CAPTCHA
- Rate limiting
- Multi-factor authentication (MFA)

---

6. Incident Response Process

Phase 1: Identification

Unauthorized access detected through login attempts.

Phase 2: Containment

- Blocked attacker IP using firewall
- Disabled compromised account

Phase 3: Eradication

- Restarted services
- Cleared malicious activity

Phase 4: Recovery

- Reset credentials
- Monitored logs

Phase 5: Lessons Learned

- Implement layered security
- Continuous monitoring is essential

---

7. Security Recommendations

- Enforce strong password policies
- Enable MFA
- Sanitize user inputs
- Implement WAF
- Use HTTPS
- Regular security audits

---

8. Conclusion

This project successfully demonstrated how common web vulnerabilities can be exploited if proper security measures are not implemented. Through hands-on testing, the importance of secure coding practices, continuous monitoring, and incident response planning was clearly understood.

---

9. Learning Outcomes

✔ Hands-on penetration testing experience
✔ Understanding OWASP Top 10 vulnerabilities
✔ Improved cybersecurity skills
✔ Real-world attack simulation

10. References

- OWASP WebGoat
- OWASP Top 10
- DVWA Official Documentation