Final Report: Damn Vulnerable Web Application (DVWA) Security Analysis

Project Title: Securing SecureCorp: Vulnerability Analysis and Countermeasures using DVWA
Student Name: Saurav Dhapola
Course: Bachelor of Computer Applications (BCA)
Project Focus: Ethical Hacking and Cybersecurity
Date: 15-10-2024

1. Introduction

This report outlines the results of a security evaluation performed on the Damn Vulnerable Web Application (DVWA) in the context of simulating real-world attacks against SecureCorp. DVWA, an intentionally vulnerable web application, was used to explore various cyber threats. The purpose of the exercise was to identify potential vulnerabilities and apply appropriate countermeasures to protect SecureCorp from common attacks.

2. Objectives

The primary objectives of the project were:

- To identify and exploit vulnerabilities in the DVWA.
- To assess the resilience of SecureCorp against specific cyber threats, including client-side, web application, and Wi-Fi security.
- To provide recommendations for mitigating identified vulnerabilities and improving security posture.

3. Methodology

The methodology followed a structured approach for vulnerability assessment, penetration testing, and analysis:

1. **Reconnaissance:** Information gathering on DVWA.
2. **Vulnerability Assessment:** Identifying weak points within DVWA using tools like OWASP ZAP and manual analysis.
3. **Exploitation:** Using DVWA to simulate various attack vectors such as SQL injection, cross-site scripting (XSS), brute-force attacks, etc.
4. **Mitigation:** Proposing defense mechanisms and fixes for the vulnerabilities identified.
5. **Report Generation:** Documenting findings and recommendations.

4. Analysis of Attack Vectors

4.1 Reconnaissance

Reconnaissance was conducted to gather intelligence on DVWA’s structure and potential attack surfaces. Tools such as `Nmap` and `Whois` were used to map out the environment.

4.2 SQL Injection

- **Vulnerability:** The DVWA login page was vulnerable to SQL injection. Exploiting this allowed unauthorized access to secure areas.
- **Exploit:** A basic SQL query bypass using `' OR '1'='1` provided administrator-level access.
- **Mitigation:** Parameterized queries and prepared statements should be implemented to prevent such attacks.

4.3 Cross-Site Scripting (XSS)

- **Vulnerability:** The DVWA "message board" functionality was vulnerable to reflected XSS, allowing the injection of malicious scripts.
- **Exploit:** Injecting `<script>alert('XSS Vulnerability')</script>` successfully executed a script in the user’s browser.
- **Mitigation:** Input validation and output encoding are essential to mitigate XSS attacks.

4.4 Brute-Force Attacks

- **Vulnerability:** The login page was susceptible to brute-force password guessing due to lack of account lockout mechanisms.
- **Exploit:** Using tools like `Hydra`, a brute-force attack was carried out, resulting in compromised credentials.
- **Mitigation:** Implementing account lockout policies and CAPTCHA could mitigate this risk.

4.5 File Inclusion

- **Vulnerability:** The application allowed local file inclusion (LFI) via improper input sanitization in file upload fields.
- **Exploit:** LFI allowed access to sensitive files such as `/etc/passwd`.
- **Mitigation:** Input validation and sanitization should be enforced to prevent file inclusion vulnerabilities.

5. Tools Used

- **Burp Suite:** For intercepting and modifying web traffic.
- **OWASP ZAP:** For automated vulnerability scanning.
- **Hydra:** To perform brute-force attacks.
- **Nmap:** To gather reconnaissance data and open ports.
- **DVWA:** As the core web application for testing vulnerabilities.

6. Results

The vulnerabilities discovered in DVWA can lead to severe security issues if exploited:

- **SQL Injection:** Full database compromise.
- **Cross-Site Scripting (XSS):** Client-side attacks and session hijacking.
- **Brute-Force Attacks:** Compromised user accounts.
- **File Inclusion:** Access to sensitive files and system compromise.

7. Recommendations

7.1 Secure Coding Practices

- **Input Validation:** Ensure all user inputs are validated against strict criteria.
- **Parameterized Queries:** Use prepared statements to defend against SQL injection.
- **Output Encoding:** Encode output to prevent XSS attacks.

7.2 Security Hardening

- **Account Lockout Mechanisms:** Implement account lockout policies after a certain number of failed login attempts.
- **CAPTCHA:** Integrate CAPTCHA on login pages to mitigate brute-force attacks.

7.3 Ongoing Vulnerability Management

- Regularly scan applications using tools like OWASP ZAP.
- Implement a bug bounty program to incentivize security researchers to identify vulnerabilities.

8. Conclusion

The analysis of DVWA has highlighted several critical vulnerabilities that can lead to a compromise of SecureCorp's web application. By following the proposed mitigation techniques, these risks can be minimized, and SecureCorp can enhance its overall security posture. This project provided a hands-on opportunity to understand various web vulnerabilities, exploitation techniques, and preventive measures.