3 Proposed Solution Testing and Findings

Website : bWAPP

Software Used : Burp Suite

1. Testing Methodology

The testing process involved:

- **Intercepting and analyzing HTTP requests** to identify security flaws.
- *Scanning for OWASP Top 10 vulnerabilities, including SQL Injection (SQLi), Cross-Site Scripting (XSS), and Broken Authentication.*
- **Exploiting vulnerabilities** to verify their impact.
- Implementing security fixes and re-testing to confirm successful mitigation.

2. Initial Findings (Pre-Mitigation Scan)

Burp Suite scan and manual testing revealed several security flaws in bWAPP, categorized as follows:

Critical Vulnerabilities

🟩 SQL Injection (SQLi)

- **Issue:** User input fields in login and search forms were vulnerable to SQLi.
- **Impact:** Allowed database access, data extraction, and potential account takeover.
- **Burp Suite Test:** Used **Burp Repeater** to inject SQL payloads (' OR 1=1 --).

🟩 Remote Code Execution (RCE)

- **Issue:** Insecure file upload allowed remote shell execution.
- **Impact:** Could lead to complete server compromise.
- **Burp Suite Test:** Captured file upload requests and modified content-type to execute shell commands.

High-Risk Vulnerabilities

🟩 Cross-Site Scripting (XSS)

- **Issue:** Input fields failed to sanitize JavaScript code.
- **Impact:** Attackers could inject malicious scripts to steal cookies or perform phishing attacks.
- **Burp Suite Test:** Injected payload (<script>alert('XSS')</script>) via **Burp Intruder**.

🟩 Broken Authentication & Weak Passwords

- **Issue:** Lack of brute-force protection on the login page.
- **Impact:** Allowed credential stuffing attacks.
- **Burp Suite Test:** Used **Burp Intruder** to perform an **automated brute-force attack** on login credentials.

3. Security Solutions Implemented

Based on the **Burp Suite findings**, the following remediation steps were applied:

SQL Injection Prevention

✓ Implemented **prepared statements and parameterized queries**.

✓ Input validation to reject malicious SQL payloads. **Medium & Low-Risk Vulnerabilities**

🟩 **Sensitive Information Exposure** – Found exposed session tokens in URL parameters.

🟩 **Clickjacking** – Application allowed framing, making it vulnerable to UI redressing attacks.

🟩 **Missing Security Headers** – Lack of **X-Frame-Options**, **Content Security Policy (CSP)**, and **HSTS**.

XSS Mitigation

✓ Enabled **input sanitization** and **output encoding** to prevent script execution.

✓ Applied **Content Security Policy (CSP)** headers.

🔷 Authentication & Access Control

✓ Enforced **strong password policies** and **account lockout mechanisms**.

✓ Added **CAPTCHA verification** to prevent brute-force attacks.

🔷 Secure File Upload Handling

✓ Restricted allowed file types and enabled **server-side validation**.

✓ Implemented sanitization of filenames to prevent RCE.

Security Hardening

✓ Added HTTP security headers to prevent Clickjacking and data exposure.

✓ Enabled HTTPS enforcement to protect data in transit.

4. Post-Mitigation Scan Results

After implementing security fixes, a second **Burp Suite scan and manual retesting** were conducted. The results showed:

🟩 Critical vulnerabilities reduced to zero.

🟩 **XSS** and **SQLi** fully mitigated after input validation and encoding.

🟩 **Brute-force protection** enabled, preventing login abuse.

🟩 Security headers implemented, enhancing protection against **clickjacking** and **XSS.**

🟩 Some low-risk issues remain but do not pose immediate threats.