**Technical Report: Exploitation of OWASP Juice Shop**

**Web Penetration Testing Report**

**Executive Summary**

* **Purpose:** This report highlights the vulnerabilities discovered during penetration testing of the OWASP Juice Shop application. It provides an overview of key findings, their impact, and critical vulnerabilities.
* **High-Level Impact:** The vulnerabilities identified include SQL Injection, brute-force attacks, and Cross-Site Scripting (XSS), all of which pose significant risks to the application’s security.
* **Summary of Recommendations:** Implement proper input validation, enforce account lockouts, and utilize rate-limiting and Content Security Policies (CSP).

**Scope and Methodology**

* **Scope:** The testing focused on the OWASP Juice Shop application, targeting its websites, APIs, and functionalities.
* **Approach:** A combination of black-box and grey-box testing methodologies was employed.
* **Tools Used:**
  + SQL Injection
  + Burp Suite
  + Browser Developer Tools

**Vulnerability Findings**

**Critical Vulnerabilities**

**1. Enumeration to Find Admin Path**

* **Description:** SQL Injection and manual inspection techniques were used to analyze the URL structure and discover hidden admin paths.
* **Risk and Impact:** The attacker gains access to admin functionality, enabling further exploitation.
* **Evidence:** Successfully discovered admin paths through application responses.
* **Remediation Steps:**
  + Implement complex URL structures.
  + Prevent information leakage in responses.
  + Regularly audit and secure exposed endpoints.

**2. Brute Force on Admin Credentials**

* **Description:** A brute-force attack was conducted using Burp Suite on the admin account admin@juice-sh.op.
* **Risk and Impact:** The admin password admin123 was successfully obtained, granting full administrative access to the application.
* **Evidence:** Logs and screenshots of successful brute-force attempts.
* **Remediation Steps:**
  + Enforce rate-limiting and account lockout mechanisms.
  + Implement multi-factor authentication.
  + Encourage strong password policies.

**3. Cross-Site Scripting (XSS) in Product Search**

* **Description:** A malicious script <iframe src="javascript:alert(xss)"> was injected into the product search bar.
* **Risk and Impact:** The script executed in the victim’s browser, potentially stealing session cookies or redirecting users.
* **Evidence:** Screenshots and logs showing successful script execution.
* **Remediation Steps:**
  + Validate and sanitize all user inputs.
  + Use Content Security Policies (CSP) to block unauthorized scripts.
  + Regularly test input fields for XSS vulnerabilities.

**4. CSP Bypass on Profile Page**

* **Description:** A script <script>alert(xss)</script> was injected into the profile page, successfully bypassing Content Security Policies (CSP).
* **Risk and Impact:** This allows attackers to execute arbitrary scripts, leading to session hijacking or data theft.
* **Evidence:** Screenshots of successful script execution in the profile page.
* **Remediation Steps:**
  + Strengthen CSP rules to prevent inline script execution.
  + Sanitize user inputs on all profile-related fields.
  + Regularly test and update CSP configurations.

**5. Composite Attack: Feedback Manipulation and Basket Inspection**

* **Description:** Two vulnerabilities were exploited:
  1. **Feedback Manipulation:** The web application allowed the removal of 5-star feedback through DOM manipulation.
  2. **Basket Inspection:** Inspecting the basket elements revealed details about user orders and potential exposure of sensitive information.
* **Risk and Impact:** Feedback manipulation can damage the application’s reputation, while basket inspection may lead to user privacy breaches.
* **Evidence:**
  1. Screenshots showing the removal of feedback.
  2. Logs of basket element inspections.
* **Remediation Steps:**
  1. Secure client-side validations to prevent feedback tampering.
  2. Encrypt and obfuscate basket data to ensure user privacy.
  3. Perform regular security reviews of user-facing features.

**Exploitation and Attack Simulation**

* **Tools and Techniques Used:**
  + SQL Injection for admin path enumeration.
  + Burp Suite for brute-force attacks.
  + Browser Developer Tools for XSS payload injection and feedback manipulation.
* **Outcome and Impact:**
  + Admin paths were discovered, granting access to sensitive functionalities.
  + The admin account was compromised, enabling full control of the application.
  + XSS vulnerabilities allowed execution of malicious scripts in users’ browsers.
  + CSP bypass and composite attacks demonstrated additional risks to user data and application reputation.

**Conclusion**

* **Summary of Security Posture:** The OWASP Juice Shop application is vulnerable to SQL Injection, brute-force attacks, XSS, CSP bypass, and client-side manipulation, posing severe risks to user data and application integrity.
* **Overall Risk Level:** High
* **Next Steps for Remediation:**
  1. Implement input validation and sanitization.
  2. Enforce account lockout mechanisms and rate-limiting.
  3. Regularly audit and patch the application to address vulnerabilities.
  4. Strengthen Content Security Policies (CSP).
  5. Secure client-side functionalities to prevent manipulation.

By addressing these vulnerabilities, the application’s security can be significantly improved.

Made by :

1. Mahmoud Ashour Abdelhady 2305101

2. Meyer Adel Kher 2305083

3. Ziad Hossam Abdelaziz 2305526