Penetration Test Report

About the Pentester:

Roei Kotzero is a Cybersecurity Specialist and Penetration Tester with hands-on experience in ethical hacking, vulnerability assessment, and security testing. He has completed a one-year Cybersecurity Certificate Program and expanded his expertise through independent study, completing labs on TryHackMe, HackTheBox, and PortSwigger. Skilled in computer networking, scripting, malware analysis, and cloud security, Roei has practical experience with modern security tools such as Wireshark, Nmap, Burp Suite, Kali Linux, and Metasploit. His areas of focus include infrastructure and web application penetration testing, Active Directory security, and cloud environments (AWS, Azure, GCP). Roei combines a strong technical foundation with continuous hands-on practice, making him well-versed in both offensive and defensive cybersecurity strategies.

1. Executive Summary

Objective: The objective of this penetration test was to evaluate the security posture of the **OWASP Juice Shop web application**, a deliberately insecure platform designed for training and awareness of common web application vulnerabilities.

Scope:
Primary Target: `OWASP Juice Shop` web application (deployed locally or hosted in a training environment).

Functional Areas Tested:

- User authentication and session management
- Product search, reviews, and feedback functionality

Approach: Black Box Testing.

Testing Approach

- The penetration test was conducted using a **Black Box methodology**, meaning testers had **no prior knowledge** of the application's source code, internal

architecture, or administrative credentials.

- The perspective simulated a real-world **external attacker** attempting to compromise the application from the internet, with only the application URL as a starting point.

- Testing involved reconnaissance, vulnerability identification, exploitation attempts, and documentation of findings.

Business Impact: Attackers could steal sensitive customer data, disrupt business operations, and damage Juice Shop's reputation.

2. Engagement Details

Client Name: Juice Shop
Test Dates: 14th-21st August 2025
Testers: ECOM School PT Team, Directed by Roei Kotzero
Test Type: Black Box

3. Methodology

1. Reconnaissance & Information Gathering

- Performed passive and active information gathering on the application.

- Collected version banners, error messages.

2. Threat Modeling & Attack Surface Mapping

- Mapped exposed features (login, registration, product reviews).

3. Vulnerability Discovery

- Conducted manual and automated testing using tools such as **Burp Suite**.

- Focused on vulnerabilities from the **OWASP Top 10**, including:

    - Injection flaws (SQLi)

- Cross-Site Scripting (XSS)
- Broken Authentication & Session Management
- Security Misconfigurations
- Sensitive Data Exposure

4. Exploitation

- Attempted to exploit discovered vulnerabilities to validate risk and demonstrate impact.
- Example:
  - Using SQL injection to bypass authentication.

5. Post-Exploitation & Privilege Escalation

- Determined the extent of access gained after successful exploitation.
- Assessed whether vulnerabilities could lead to **account takeover**, **data theft**, or **system compromise**.

6. Reporting

- Documented all findings with **evidence, impact analysis, and remediation recommendations**.
- Provided both a **technical report** (for developers and security engineers) and an **executive summary** (for management).

4. Scope of Work

In-Scope Assets:
- juiceshop (Deployed on Local Host)

5. Findings Overview

Severity	Count	Examples

Critical	2	SQL Injection, RCE in File Upload
High	3	Weak Password Policy, Missing Patches, Exposed Admin Panel
Medium	4	Insecure TLS Config, Verbose Error Messages
Low	5	Banner Grabbing, Directory Listing
Informational	3	Outdated server version disclosure

6. Detailed Findings

Vulnerability Number 5: Insecure Direct Object Reference in Product Reviews

Severity: Critical

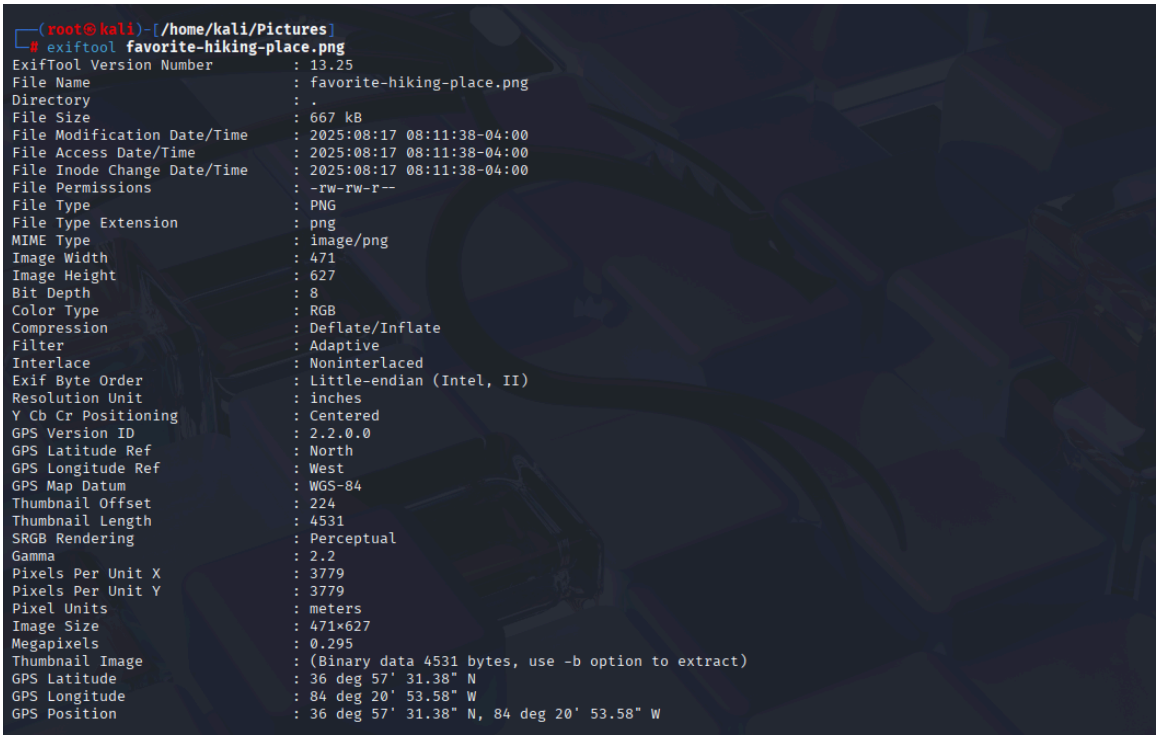
Description: The application's Photo Wall feature exposes sensitive metadata (EXIF data) of uploaded images, including GPS coordinates. This information is accessible directly from the client side without sanitization or stripping. During testing, the metadata of photos uploaded by a known user was retrieved, revealing sensitive location details.

Evidence: Johnny uploaded a photo of his favourite place hiking, I downloaded the photo, Used and Exiftool to reach the metadata and found the coordinates of where he hikes. John's security question was "Where is your favourite place to hike?"

Impact: Leakage of personal information (location, device type, timestamps).
Attackers can use metadata to answer security questions, leading to **account takeover**.
Severe privacy violations.

Mitigation Ways: Strip all metadata (EXIF, GPS, device info) from images before storage or public display.
Provide only sanitized versions of photos to users.
Encourage secure password reset flows using email/SMS tokens or MFA instead of

knowledge based questions.



Vulnerability Number 2: SQL Injection in Login Form

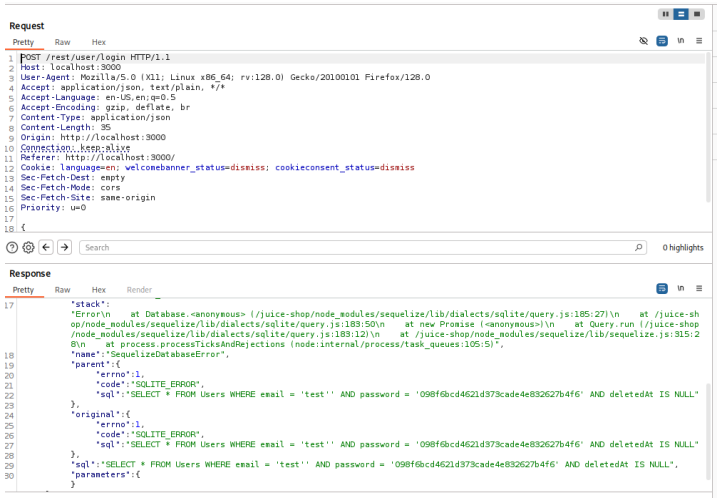
Severity: Critical

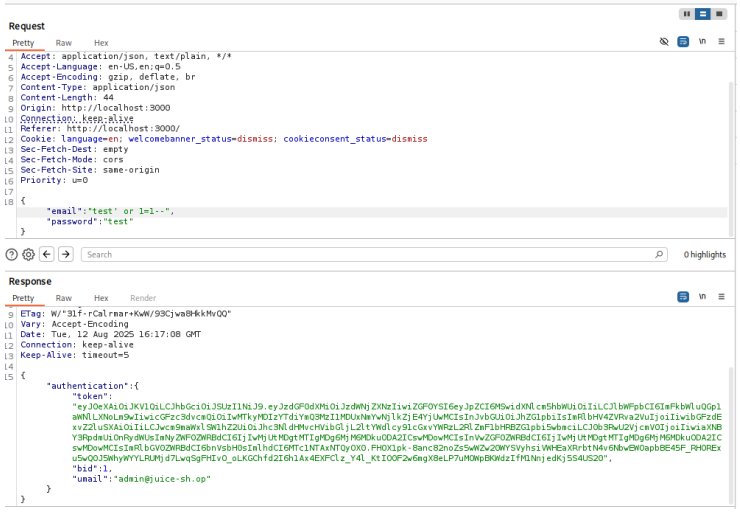
Description: The login form at shop.acme.com/login is vulnerable to SQL injection via the username parameter.

Evidence: Payload ' OR '1'='1 bypassed authentication, returning admin access.

Impact: Full access, including customer PII and payment records.

Mitigation Ways: Implement parameterized queries and input validation. Deploy WAF (Web App Firewall) rules.





Vulnerability Number 3: Login Page Vulnerable to Brute Force

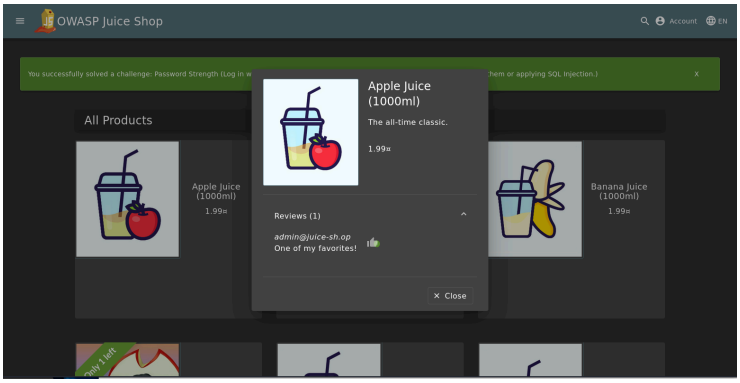
Severity: High

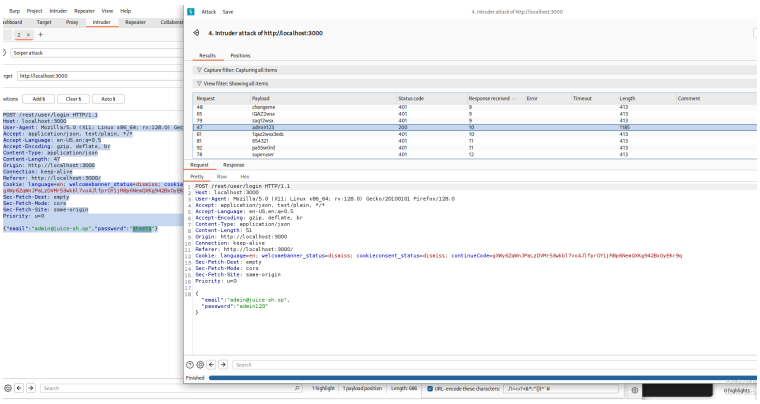
Description: The login functionality of the application does not implement any form of brute force protection. An attacker can submit unlimited password attempts without being limited, or blocked.

Evidence: Multiple failed login attempts were made against the login page. No account lockout or timeout was triggered. This allowed successful brute forcing of the **admin account password.**

Impact: An attacker could compromise valid user accounts, including privileged accounts (admin etc..). Successful exploitation results in full application compromise, unauthorized access to sensitive data, and potential manipulation of business functionality.

Mitigation Ways: Implement account lockout or exponential backoff after several failed login attempts. Deploy CAPTCHA challenges after repeated login failures. Enforce strong password policies and encourage Multi-Factor Authentication.





Vulnerability Number 4: Forgot Password Page Reveals Valid User Accounts

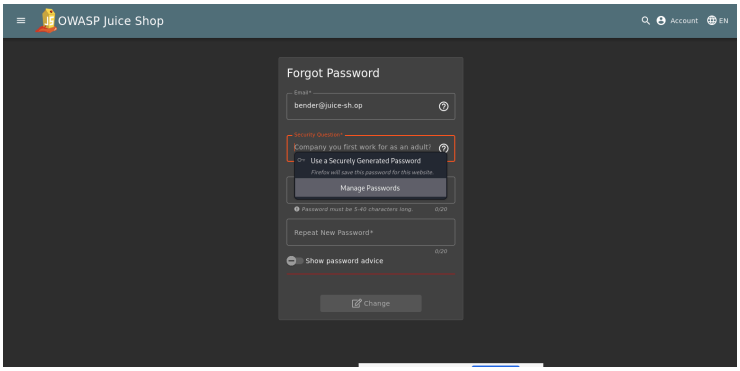
Severity: Medium

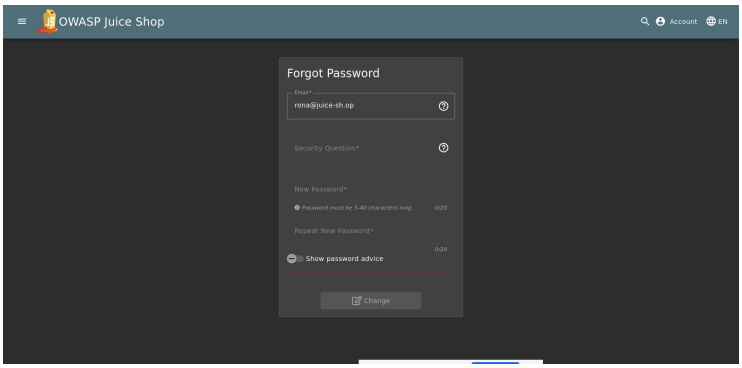
Description: The Forgot Password page discloses whether an email address exists in the application's user database. When a valid email is entered, the page comes up with a security question and when an invalid email is entered, no question appears. This difference in behavior allowed me to enumerate valid accounts.

Evidence: When I input email: bender@juice-sh.op a security question popped up "Company you first work for as an adult?" - When I input email: rona@juice-sh.op no security question popped up. This method confirms if the account exists without the need to authenticate.

Impact: I could perform enumeration to gather a list of valid users, Which will then be used to perform brute force attacks on, Raises the possibility of targeted attacks on crucial users such admin etc…

Mitigation Ways: Ensure that the application's responses to valid and invalid emails are indistinguishable. Always return a generic message such as: If the email exists in our database, a password reset link has been sent.
Implement rate limiting and monitoring for repeated password reset requests.





Vulnerability Number 5: Insecure Direct Object Reference in Product Reviews

Severity: High

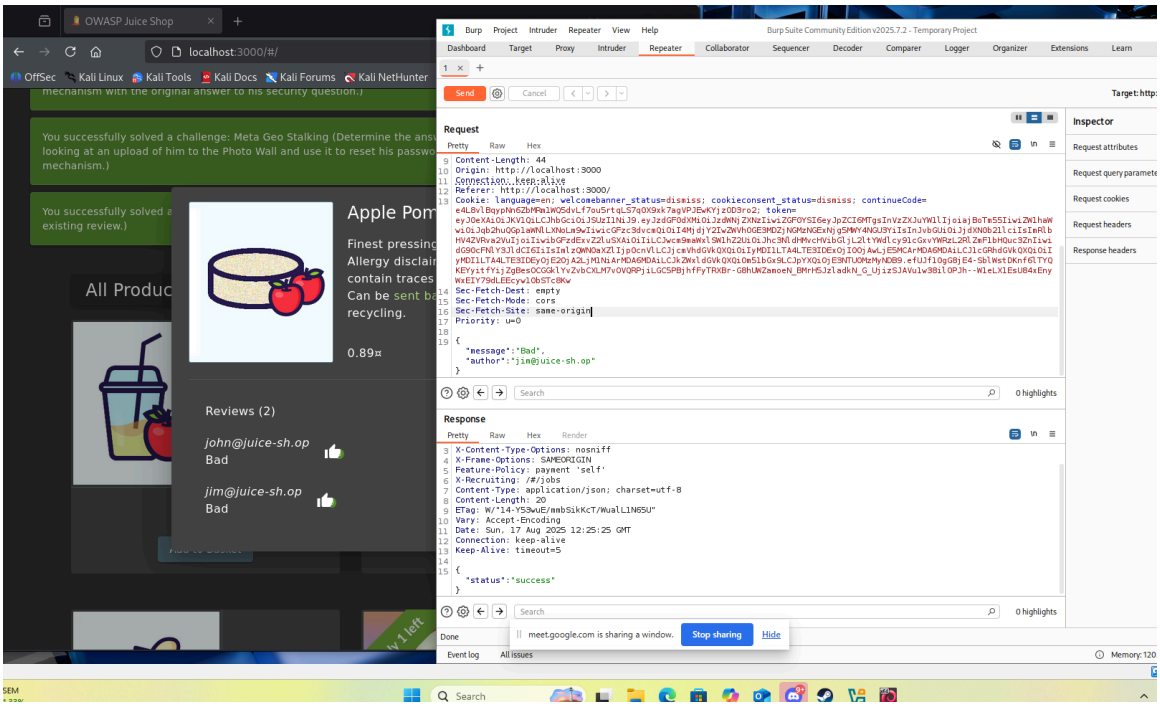
Description: The product review functionality is vulnerable to Insecure Direct Object Reference. When submitting a review, the application relies on a client supplied field to identify the reviewer. By modifying this parameter in the request, a user can impersonate another known account and submit reviews under their identity.

Evidence: I commented in another user's name by changing the author in the request from john@juice-sh.op to jim@juice-sh.op

Impact: Attackers can impersonate other users, damaging their reputation or misleading customers. If **admin or privileged accounts** are impersonated, attackers can undermine trust in official communication. Potential brand damage if false reviews appear under trusted users.

Mitigation Ways: Enforce server-side authorization checks to ensure reviews are always bound to the authenticated user's session, not client supplied values.

Ignore or overwrite any author fields supplied in client requests.



8. Conclusion

Juice Shop's platform is vulnerable to critical exploitation, which could lead to a full compromise of customer data and systems. Addressing the vulnerabilities should be the highest priority, followed by tightening access controls and improving overall security hygiene.

Ongoing testing and integration of secure development practices is strongly recommended.

9. Appendices

Tools Used:
- Burp Suite, Exiftool

References:
- OWASP Top 10 (2021)
- OWASP Cheat Sheet
- NIST SP 800-115 – Technical Guide to Information Security Testing
- OWASP Juice Shop Project