Cyber Security Internship

(Bug Bounty Report – OWASP Juice Shop)

Harsh M Parikh

ph400764@gmail.com

1. Scope & Objective

Scope: OWASP Juice Shop (training instance)
Objective: Perform reconnaissance and safe vulnerability discovery; document findings, reproduce PoC in lab-only environment, provide mitigations and recommendations.

2. Reconnaissance (commands & notes — lab only)

DNS / Host

- dig +short juice-shop.lab.example
- nslookup juice-shop.lab.example

Port / Service Enumeration

- nmap -sV -p- --top-ports 1000 juice-shop.lab.example
- nmap -sC -sV -p80,443 juice-shop.lab.example

Web discovery

- gobuster dir -u http://juice-shop.lab.example -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,html,js
- Browser inspection (DevTools) for hidden endpoints, JS files

Subdomain / Hostname enumeration (lab-only)

- sublist3r -d lab.example (configured to lab DNS)
- amass enum -d lab.example -o amass_out.txt

API discovery

- Interrogate
  endpoints: /rest/products, /api/feedback, /login, /search, /profile

3. Findings (Simulated / Educational)

Finding 1 — Reflected XSS (Low / Medium)

ph400764@gmail.com

Endpoint: /search?q=<payload> (lab)
Overview: Input reflected into HTML output without contextual escaping.
PoC (lab-only):

1. Enter payload into search box: <script>/*lab-test*/alert('xss')</script>
2. Observe script execution in browser (DevTools console screenshot captured).

Impact: In a real app, this could allow session theft, UI manipulation, or phishing.
Recommended Fixes:

- Contextual HTML escaping / templating that auto-escapes.
- Implement CSP header.
- Input validation + output encoding libraries.

Finding 2 — Insecure Direct Object Reference (IDOR) (Medium)

Endpoint: GET /api/users/{id}
Overview: Sequential numeric IDs allow low-privilege authenticated users to query other users' data.
PoC (lab-only):

1. Login as low-privilege user.
2. GET /api/users/101 → returns profile for user 101 (not the requester).
3. Capture request/response (HAR), redacted and included as artifact.

Impact: Unauthorized data disclosure of PII or account info.
Recommended Fixes:

- Enforce server-side authorization checks on every resource (verify ownership).
- Use non-predictable identifiers (UUIDs) and access control checks.

Finding 3 — Missing Rate Limiting on Auth (Low)

ph400764@gmail.com

Endpoint: /rest/auth/login
Overview: No per-IP/account rate limiting observed on authentication endpoint.
PoC (lab-only):

- Automated login attempts (low volume in lab): observed unlimited tries.

Impact: Brute-force attempts or account enumeration.
Recommended Fixes:

- Add rate limiting per IP/account with exponential backoff.
- Implement account lockouts or progressive delays for failed attempts.
- Monitor and alert on brute-force patterns.

Finding 4 — Command Injection (educational example)

Endpoint: /api/diagnose?host= (lab)
Overview: In lab, a diagnostic endpoint accepts host input and performs a server-side command. If not properly sanitized, command injection is possible in concept.
PoC (lab-only):

- Submit benign payload in lab environment demonstrating output concatenation only (no real destructive commands).
- Evidence captured as request/response screenshot (artifact).

Recommended Fixes:

- Never invoke shell with unsanitized user input. Use language-specific APIs that avoid shell interpretation.
- Validate and whitelist allowed input; escape and sandbox operations.

4. Reproduction Steps (high-level)

Steps below were executed only on the lab instance.

1. Open browser to lab Juice Shop instance.

2. For XSS: input non-malicious test payload in search; observe DOM and console. Capture screenshot and HAR.
3. For IDOR: authenticate → fetch GET /api/users/{id} with different IDs; capture responses.
4. For rate-limiting: simulate repeated failed logins; capture logs and server responses.
5. Save all request/response HARs; redact sensitive values before attaching.

5. Risk Ratings and Suggested Remediation Priority

- IDOR (Medium): Priority — High. Patch authorization logic immediately.
- Reflected XSS (Medium): Priority — Medium. Fix escaping and implement CSP.
- Missing Rate Limiting (Low): Priority — Medium. Add throttling and monitoring.

6. Recommendations (general)

- Apply strict server-side access control checks (deny-by-default).
- Context-aware escaping for all user-supplied output; adopt CSP.
- Harden authentication (rate-limit, MFA for critical users).
- Regular DAST scans for regression, but always validate findings manually.
- Maintain a triage and disclosure process for real-world findings.

7. Conclusion

This simulated, educational engagement on OWASP Juice Shop allowed safe demonstration of common web vulnerabilities (XSS, IDOR, authentication weaknesses). The recommendations above will reduce exposure and should be validated with follow-up tests once applied.

ph400764@gmail.com