**Security Assessment Report**

**Target: OWASP Juice Shop (**[**http://localhost:3000**](http://localhost:3000)**)**

**Date: January 20, 2025**

**Overview**

This report documents the findings from a comprehensive security assessment conducted on the OWASP Juice Shop web application hosted on localhost:3000. The assessment was carried out using **Nikto v2.5.0**, an open-source web server scanner designed to detect common vulnerabilities, server misconfigurations, and security flaws.

The OWASP Juice Shop, a deliberately vulnerable web application designed for security training, demonstrated several areas of concern. These issues, if not addressed, could lead to unauthorized access, information disclosure, or potential exploitation.

**Objective of the Assessment**

The primary objectives of this security assessment were:

1. To identify potential vulnerabilities in the web server configuration and hosted application.
2. To analyze exposed directories, files, and other resources that could compromise the server.
3. To provide actionable recommendations for mitigating identified risks and strengthening the server’s security posture.

**Scope**

The assessment focused solely on the OWASP Juice Shop instance running on http://localhost:3000. Scans targeted:

* Server headers and configurations.
* Accessible files, directories, and sensitive resources.
* Missing or misconfigured security controls.

**Findings**

**1. General Information**

* **IP Address:** 127.0.0.1
* **Port:** 3000
* **Server Banner:** No banner retrieved, which limits the identification of server software.

While the absence of a server banner reduces information leakage, it complicates version-specific vulnerability identification.

**2. Security Headers**

The web server is missing critical security headers, including:

* **X-Content-Type-Options:**
  + Not set, allowing browsers to interpret content MIME types differently from those declared.
  + **Risk:** Can lead to MIME-type confusion attacks, where attackers trick the browser into executing malicious files.
* **Recommendation:**  
  Add the header X-Content-Type-Options: nosniff to mitigate this risk.

Other potentially missing headers that should be considered include:

* **Content-Security-Policy (CSP):** Prevents unauthorized resource loading.
* **Strict-Transport-Security (HSTS):** Enforces secure connections over HTTPS.

**3. Sensitive Files and Directories**

The server exposes several sensitive files and directories, increasing the risk of unauthorized access and information leakage:

* **Robots.txt:**
  + Found at /robots.txt, with the following entry: /ftp/.
  + The directory /ftp/ is accessible, returning a 200 OK response.

**Risk:** Attackers could use this information to discover potentially sensitive files or directories.

**Recommendation:** Avoid listing sensitive directories in robots.txt. If necessary, secure these directories with authentication or IP restrictions.

* **Backup and Archive Files:**  
  The following files, identified during the scan, may contain sensitive configurations or credentials:
  + /backup.cer
  + /backup.tar
  + /site.tar.lzma
  + /dump.tar.bz2
  + /localhost.pem

**Risk:** These files can expose critical data such as encryption keys, database dumps, or system configurations, enabling attackers to compromise the application or underlying infrastructure.

**Recommendation:**

* + Immediately review and remove unnecessary backup or archive files from the server.
  + Implement secure storage practices for sensitive files, ensuring they are not accessible via the web server.

**4. Additional Potentially Interesting Files**

Other files were identified that may require further review for sensitive content:

* /archive.tar
* /database.egg
* /localhost.war
* /dump.jks

**Recommendation:**  
Audit these files to determine their purpose. If they are unnecessary, delete them. If they must remain on the server, restrict access and encrypt their contents.

**Critical Observations and Risks**

1. **Insufficient Security Headers:** The absence of key headers like X-Content-Type-Options and potentially CSP and HSTS makes the application vulnerable to common web attacks, including content injection and phishing.
2. **Exposure of Sensitive Files:** The availability of backup and certificate files poses a significant risk of information leakage, potentially leading to server compromise.
3. **Directory Listings in Robots.txt:** Including /ftp/ in robots.txt unnecessarily exposes the directory to potential exploitation.

**Detailed Recommendations**

To mitigate the identified vulnerabilities and improve the overall security of the OWASP Juice Shop, the following measures should be implemented:

**1. Harden Web Server Configuration:**

* Remove or secure all sensitive files, including backups, certificates, and archives.
* Review file permissions and ensure sensitive files are not accessible to unauthorized users.

**2. Implement Missing Security Headers:**

Configure the server to include the following headers:

* X-Content-Type-Options: nosniff
* Content-Security-Policy: default-src 'self'; script-src 'self';
* Strict-Transport-Security: max-age=31536000; includeSubDomains; preload

**3. Restrict Access to Robots.txt Entries:**

* Avoid listing sensitive directories like /ftp/ in robots.txt.
* Consider blocking access to the /robots.txt file entirely or restricting directory access using authentication.

**4. Perform Regular Vulnerability Scanning:**

* Conduct regular scans using tools like Nikto, OWASP ZAP, or Burp Suite to identify emerging vulnerabilities.
* Regularly update server software and libraries to address known vulnerabilities.

**Conclusion**

The OWASP Juice Shop application exhibited several vulnerabilities due to exposed files, weak server configurations, and the absence of key security headers. These issues, while often overlooked, can be exploited by attackers to compromise the system and extract sensitive data.

By addressing the recommendations outlined in this report, the security posture of the application can be significantly improved, reducing the risk of exploitation and ensuring a safer environment for testing and development.

**References**

1. CWE-530: <https://cwe.mitre.org/data/definitions/530.html>
2. OWASP Secure Headers Project: https://owasp.org/www-project-secure-headers/
3. Mozilla Developer Network (MDN): <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers>
4. Nikto Documentation: https://cirt.net/Nikto2