

# Cybersecurity Internship– Week 6 Report

**Intern Name:** Muhammad Ali Kashif

**Project:** Advanced Security Audits & Final Deployment OWASP Juice Shop

**Week:** 6

## Objective

The objective of this week was to conduct advanced security audits on the OWASP Juice Shop application, evaluate compliance with industry security standards, and prepare the application for secure deployment.

Multiple automated and manual security testing tools were used, including **OWASP ZAP, Nikto, Trivy, and Burp Suite**, to identify vulnerabilities, misconfigurations, and potential attack vectors.

The assessment revealed several security weaknesses such as missing security headers, vulnerable dependencies, exposed cryptographic keys, and cross-domain risks. These findings highlight the importance of proactive security testing before production deployment.

Overall, this security audit demonstrates the implementation of industry-standard penetration testing practices and secure deployment methodologies.

## 1. Application Deployment

The OWASP Juice Shop application was deployed locally using Docker to simulate a real-world web application environment suitable for penetration testing.

The containerized setup ensured consistency, isolation, and efficient vulnerability analysis.

**Screenshot:** OWASP Juice Shop running on localhost inside Docker

```
(kali㉿kali)-[~]
$ sudo docker ps
```

CONTAINER ID	IMAGE	COMMAND NAMES	CREATED	STATUS	PORTS
8774a121ab4b	bkimminich/juice-shop	"/nodejs/bin/node /j..." festive_driscoll	3 hours ago	Up 3 hours	0.0.0.0 :3000→3000/tcp, :::3000→3000/tcp

```
(kali㉿kali)-[~]
$
```

## 2. Security Audits & Compliance

### 2.1 OWASP ZAP Scan

An automated baseline scan was performed using OWASP ZAP to identify common web vulnerabilities.

**Key Findings:**

- Missing Content Security Policy (CSP)
- Cross-domain misconfiguration
- Dangerous JavaScript functions detected
- Suspicious comments in source files
- Insufficient site isolation protections

These vulnerabilities could potentially allow attackers to inject malicious scripts, exploit browser behavior, or access sensitive data.

**Screenshot:** OWASP ZAP scan highlighting security warnings.

```
(kali@kali)-[~]
└─$ docker run --network host \
-v $(pwd):/zap/wrk:/rw \
-t ghcr.io/zaproxy/zaproxy:stable zap-baseline.py \
-t http://localhost:3000 \
-r zap_report.html

Using the Automation Framework
Total of 95 URLs
PASS: Vulnerable JS Library (Powered by Retire.js) [10003]
PASS: In Page Banner Information Leak [10009]
PASS: Cookie No HttpOnly Flag [10010]
PASS: Cookie Without Secure Flag [10011]
PASS: Re-examine Cache-control Directives [10015]
PASS: Content-Type Header Missing [10019]
PASS: Anti-clickjacking Header [10020]
PASS: X-Content-Type-Options Header Missing [10021]
PASS: Information Disclosure - Debug Error Messages [10023]
PASS: Information Disclosure - Sensitive Information in URL [10024]
PASS: Information Disclosure - Sensitive Information in HTTP Referrer Header [10025]
PASS: HTTP Parameter Override [10026]
PASS: Off-site Redirect [10028]
PASS: Cookie Poisoning [10029]
PASS: User Controllable Charset [10030]
PASS: User Controllable HTML Element Attribute (Potential XSS) [10031]
PASS: Viewstate [10032]
PASS: Directory Browsing [10033]
PASS: Heartbleed OpenSSL Vulnerability (Indicative) [10034]
PASS: Strict-Transport-Security Header [10035]
PASS: HTTP Server Response Header [10036]
PASS: Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s) [10037]

mouse pointer inside or press Ctrl+G

http://localhost:3000/runtime.js (200 OK)
http://localhost:3000/sitemap.xml (200 OK)
WARN-NEW: Timestamp Disclosure - Unix [10096] x 5
http://localhost:3000 (200 OK)
http://localhost:3000 (200 OK)
http://localhost:3000 (200 OK)
http://localhost:3000/sitemap.xml (200 OK)
http://localhost:3000/sitemap.xml (200 OK)
WARN-NEW: Cross-Domain Misconfiguration [10098] x 5
http://localhost:3000 (200 OK)
http://localhost:3000/polyfills.js (200 OK)
http://localhost:3000/robots.txt (200 OK)
http://localhost:3000/runtime.js (200 OK)
http://localhost:3000/sitemap.xml (200 OK)
WARN-NEW: Modern Web Application [10109] x 5
http://localhost:3000 (200 OK)
http://localhost:3000/ (200 OK)
http://localhost:3000/juice-shop/build/routes/fileServer.js:43:13 (200 OK)
http://localhost:3000/juice-shop/node_modules/express/lib/router/layer.js:95:5 (200 OK)
http://localhost:3000/sitemap.xml (200 OK)
WARN-NEW: Dangerous JS Functions [10110] x 2
http://localhost:3000/main.js (200 OK)
http://localhost:3000/vendor.js (200 OK)
WARN-NEW: Insufficient Site Isolation Against Spectre Vulnerability [90004] x 10
http://localhost:3000 (200 OK)
http://localhost:3000/ (200 OK)
http://localhost:3000/ftp (200 OK)
http://localhost:3000/juice-shop/build/routes/fileServer.js:43:13 (200 OK)
http://localhost:3000/sitemap.xml (200 OK)
FAIL-NEW: 0 FAIL-INPROG: 0 WARN-NEW: 10 WARN-INPROG: 0 INFO: 0 IGNORE: 0 PASS: 57

(kali@kali)-[~]
```

2.2 Nikto Web Server Scan

Nikto was used to detect server misconfigurations and exposed files.

Key Findings:

- Presence of uncommon HTTP headers
- Potential backup and certificate files accessible
- Missing X-Content-Type-Options header
- Robots.txt exposing sensitive directories

Such issues may assist attackers in reconnaissance and increase the attack surface.

**Screenshot:** Nikto scan results showing web server weaknesses

```
(kali@kali)~$ nikto -h http://localhost:3000

- Nikto v2.5.0

+ Target IP: 127.0.0.1
+ Target Hostname: localhost
+ Target Port: 3000
+ Start Time: 2026-02-09 07:38:16 (GMT-5)

+ Server: No banner retrieved
+ /: Retrieved access-control-allow-origin header: *.
+ /: Uncommon header 'x-recruiting' found, with contents: /#/jobs.
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ /robots.txt: Entry '/ftp/' is returned a non-forbidden or redirect HTTP code (200). See: https://portswigger.net/kb/issues/00600600_robots-txt-file
+ /robots.txt: contains 1 entry which should be manually viewed. See: https://developer.mozilla.org/en-US/docs/Glossary/Robots.txt
+ assets/public/favicon.js.ico: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ /127.0.0.1.tar.bz2: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /backup.war: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /127.0.0.1.tar: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /database.jks: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /site.war: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /localhost.tgz: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /dump.war: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /dump.pem: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /archive.tar: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /localhost.war: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /dump.egg: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /site.alz: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /backup.tgz: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /dump.cer: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /site.tgz: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /dump.tar: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /ftp/: This might be interesting.
+ /public/: This might be interesting.
+ ERROR: Error limit (20) reached for host, giving up. Last error: error reading HTTP response
+ Scan terminated: 20 error(s) and 77 item(s) reported on remote host
+ End Time: 2026-02-09 07:42:57 (GMT-5) (281 seconds)

+ 1 host(s) tested
```

## 2.3 Container Vulnerability Scanning (Trivy)

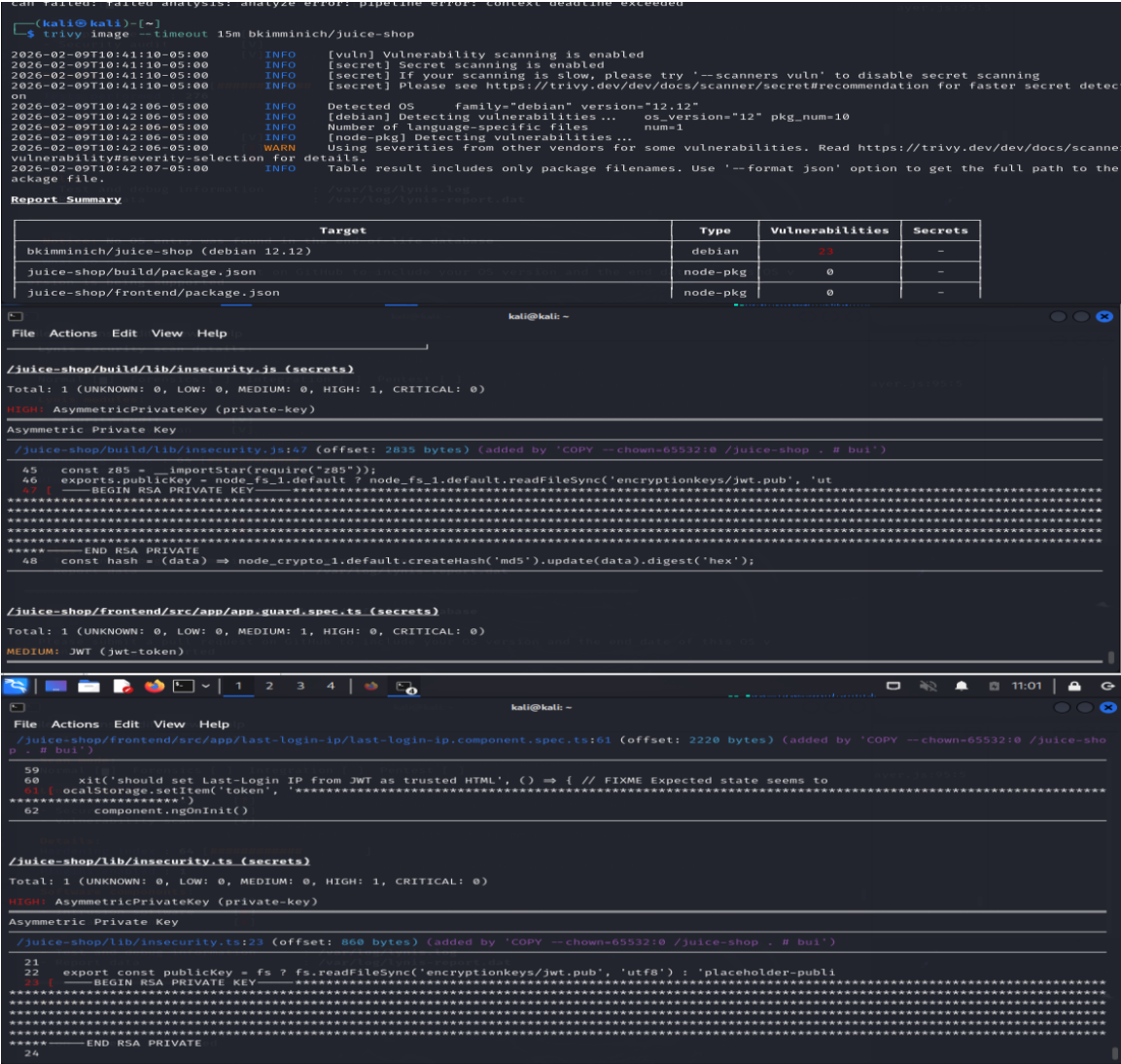
Trivy was used to scan the Docker image for vulnerable dependencies and secrets.

### Critical Observations:

- **CRITICAL:** vm2 sandbox escape vulnerability
- **HIGH:** Denial-of-service risk in WebSocket package
- Detection of an exposed **RSA private key**
- Hardcoded JWT tokens found in project files

Exposure of private keys represents a severe cryptographic risk that could compromise authentication mechanisms.

Screenshot: Trivy container scan identifying critical vulnerabilities.



3. OWASP Top 10 Compliance Check

The application was evaluated against OWASP Top 10 security risks.

OWASP Risk

Observation

Broken Access Control

Restricted admin routes detected

Cryptographic Failures

Private key exposure identified

Injection

Potential risk due to unsafe JS functions

Security Misconfiguration

Missing headers and CSP

## OWASP Risk

## Observation

Vulnerable Components

Multiple outdated dependencies

Identification & Authentication Failures JWT tokens present in files

Software Integrity Failures

Vulnerable packages detected

The assessment confirms that intentionally vulnerable applications like Juice Shop expose risks commonly found in real-world systems.

## 4. Secure Deployment Practices

### 4.1 Dependency & Image Scanning

Container images were scanned prior to deployment to identify exploitable libraries.

This proactive approach reduces the likelihood of deploying vulnerable software into production environments.

### 4.2 Docker Security Practices Followed

- Used official container images
- Performed vulnerability scanning
- Maintained isolated runtime environment
- Avoided exposing unnecessary ports

**Screenshot:** Running Juice Shop container.

## 5. Final Penetration Testing

Manual penetration testing was conducted using **Burp Suite** to intercept and analyze HTTP traffic between the client and server.

### Activities Performed:

```
(kali㉿kali)-[~]
$ sudo docker ps
```

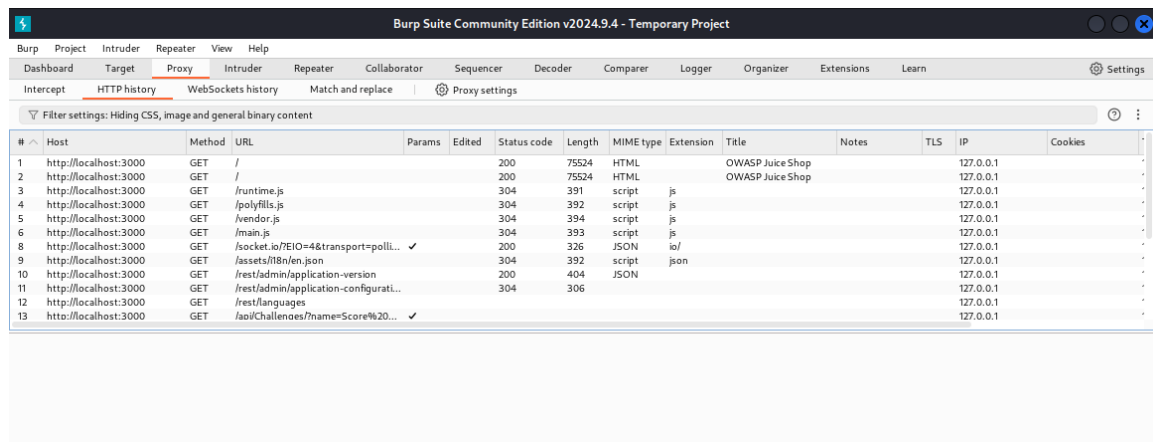
CONTAINER ID	IMAGE	COMMAND NAMES	CREATED	STATUS	PORTS
8774a121ab4b	bkimminich/juice-shop	"/nodejs/bin/node /j..." festive_driscoll	3 hours ago	Up 3 hours	0.0.0.0

```
(kali㉿kali)-[~]
$
```

- Intercepted HTTP requests
- Analyzed authentication behavior
- Observed session handling
- Evaluated request/response structure

This testing provided deeper insight into application behavior beyond automated scans.

**Screenshot:** HTTP request intercepted during manual penetration testing.



The screenshot shows the Burp Suite interface with the 'Intercept' tab selected. The table below displays a list of intercepted HTTP requests.

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes	TLS	IP	Cookies
1	http://localhost:3000	GET	/			200	75524	HTML		OWASP Juice Shop			127.0.0.1	
2	http://localhost:3000	GET	/runtime.js			200	75524	HTML		OWASP Juice Shop			127.0.0.1	
3	http://localhost:3000	GET	/polyfills.js			304	391	script	js				127.0.0.1	
4	http://localhost:3000	GET	/polyfills.js			304	392	script	js				127.0.0.1	
5	http://localhost:3000	GET	/vendor.js			304	394	script	js				127.0.0.1	
6	http://localhost:3000	GET	/main.js			304	393	script	js				127.0.0.1	
8	http://localhost:3000	GET	/socket.io/?EIO=4&transport=poll...		✓	200	326	JSON	io/				127.0.0.1	
9	http://localhost:3000	GET	/assets/i18n/en.json			304	392	script	json				127.0.0.1	
10	http://localhost:3000	GET	/rest/admin/application-version			200	404	JSON					127.0.0.1	
11	http://localhost:3000	GET	/rest/admin/application-configurati...			304	306						127.0.0.1	
12	http://localhost:3000	GET	/rest/languages										127.0.0.1	
13	http://localhost:3000	GET	/api/challenges/?name=Score%20...		✓								127.0.0.1	

## 6. Security Improvements Recommended

The following actions are recommended before deploying any production-grade application:

- Implement a strong **Content Security Policy**
- Remove hardcoded secrets and private keys
- Regularly update dependencies
- Enable secure HTTP headers
- Perform continuous vulnerability scanning
- Apply least-privilege access control
- Encrypt sensitive data
- Conduct periodic penetration testing

## 7. Challenges Faced

During the security assessment, several technical challenges were encountered:

- Docker networking configuration issues

- Tool installation errors
- Connectivity problems during automated scans

These were resolved through systematic troubleshooting and configuration adjustments.

## 8. Conclusion

This week focused on performing comprehensive security audits and preparing the application for secure deployment.

By leveraging automated scanners and manual penetration testing tools, multiple vulnerabilities were successfully identified and analyzed.

The exercise reinforced the importance of integrating security testing throughout the software development lifecycle to minimize cyber risks and strengthen application resilience.

---

---

**End of Report**