

Web Application Penetration Testing Report

Target Application: OWASP Juice Shop

Prepared By: Purna Kishore

Version: 1.0

1. Executive Summary

1.1 Summary

A penetration test was conducted on the OWASP Juice Shop web application to identify vulnerabilities aligned with the OWASP Top 10. The assessment revealed critical security issues, including SQL Injection, Reflected Cross-Site Scripting (XSS), authentication weaknesses, and security misconfigurations. These vulnerabilities could allow attackers to gain unauthorized access, extract sensitive data, and disrupt application availability¹.

1.2 Strengths

- The application uses some modern frameworks and exposes RESTful endpoints, which can simplify security hardening.
- Security scanning tools (e.g., Retire.js) were able to identify third-party dependencies, aiding in vulnerability management¹.
- Proxy tools and technology fingerprinting (Paralyzer) provided clear insights into the application stack, supporting targeted testing¹.

1.3 Weaknesses

- SQL Injection vulnerabilities in authentication forms allowed bypassing login and extracting user data.
- Reflected XSS was exploitable with multiple payloads, including <iframe src=JavaScript: alert.
- Lack of rate limiting and account lockout enabled brute-force attacks with common passwords.
- Directory listing and insecure credentials were exposed due to security misconfiguration.
- Open-source components with known vulnerabilities were detected by Retire.js.

1.4 Business Impact

If exploited in a production environment, these vulnerabilities could result in:

- Compromise of user data, including email addresses and credentials.

- Unauthorized access to user accounts and sensitive application functions.
- Service disruption due to denial-of-service (DoS) attacks.
- Regulatory non-compliance and reputational damage.

2. Technical Summary

2.1 Scope & Objective

The assessment focused on the OWASP Juice Shop instance, aiming to identify and exploit OWASP Top 10 vulnerabilities using a combination of automated and manual techniques. The primary objectives were to demonstrate exploitability and provide actionable remediation guidance¹.

2.2 Testing Environment

- **Target:** OWASP Juice Shop (local/cloud instance)
- **Tools Used:**
 - Burp Suite Community Edition (manual testing, request interception)
 - Wappalyzer (technology fingerprinting)
 - Retire.js (component vulnerability scanning)
 - Common password lists (for brute-force testing)
- **Test Date:** June 25, 2025

2.3 Technical Impact

- SQL Injection enabled authentication bypass and data extraction.
- Reflected XSS allowed execution of arbitrary JavaScript in users' browsers.
- Brute-force attacks revealed weak authentication controls.
- Security misconfigurations exposed sensitive directories and credentials.
- Vulnerable third-party components increased the attack surface.

2.4 Table of Findings

S. N o	Tool/Technique	Description	Severity	Status
0 1	Wappalyzer	Identified backend technologies and components	Informational	NF
0 2	SQL Injection (Login)	Bypassed authentication, extracted user emails	Critical	NF
0 3	Reflected XSS	Executed JavaScript via unsensitized input	High	NF
0 4	Retire.js	Detected vulnerable open-source dependencies	Medium	NF
0 5	Brute-force (DoS)	No rate limiting, 100 common passwords tested	High	NF
0 6	Security Misconfiguration	Directory listing and insecure credentials exposed	High	NF

Severity Key:

- Critical – Full compromise or remote code execution
- High – Sensitive data exposure or account takeover
- Medium – Privilege escalation or information disclosure
- Informational – Contextual, not directly exploitable

Juice shop Installation: <https://hub.docker.com/r/bkimminich/juice-shop>

- 1. Quickest Method (Docker):**

```
docker pull bkimminich/juice-shop && docker run -p 3000:3000 bkimminich/juice-shop
```

Access at <http://localhost:3000>.

- 2. From Source:**

Clone the repo, run `npm install && npm start`. Requires Node.js.

- 3. Pre-Packaged:**

Download the latest release ZIP/tgz, extract, and run `npm start`.

- 4. Cloud Deploy:**

One-click deploy on **Heroku** or use **AWS/Azure/GCP** scripts for instant hosting.

- 5. Vagrant:**

```
cd vagrant && vagrant up to launch a VM (requires VirtualBox).
```

3. Technical Findings

Passive Reconnaissance:

DNS Enumeration:

Used command - `dnsenum` `juiceshop.com.` (`dnsenum` `example.com`)

```
(kali㉿ kali) -[~]
$ dnsenum juiceshop.com
dnsenum VERSION:1.3.1

----- juiceshop.com -----

Host's addresses:
-----
juiceshop.com.          3600    IN   A      23.227.38.32

Name Servers:
-----
ns70.domaincontrol.com. 171008    IN   A      173.201.72.45
ns69.domaincontrol.com. 171009    IN   A      97.74.104.45

[Mail (MX) Servers:
-----
alt1.aspmx.l.google.com. 293      IN   A      172.253.116.27
alt2.aspmx.l.google.com. 293      IN   A      173.194.76.27
aspmx.l.google.com.      293      IN   A      142.251.111.27
aspmx2.googlemail.com.   239      IN   A      172.253.116.27
aspmx3.googlemail.com.   209      IN   A      173.194.76.27

Trying Zone Transfers and getting Bind Versions:
-----
Trying Zone Transfer for juiceshop.com on ns70.domaincontrol.com ...
AXFR record query failed: corrupt packet

Trying Zone Transfer for juiceshop.com on ns69.domaincontrol.com ...
AXFR record query failed: corrupt packet

Brute forcing with /usr/share/dnsenum/dns.txt:
-----
abc.juiceshop.com.       600      IN   CNAME  juiceshop-production.herokuapp.com.
juiceshop-production.herokuapp.com. 42        IN   A      18.208.60.216
juiceshop-production.herokuapp.com. 42        IN   A      54.159.116.102
juiceshop-production.herokuapp.com. 42        IN   A      54.165.58.209
juiceshop-production.herokuapp.com. 42        IN   A      52.5.82.174
ftp.juiceshop.com.        3600     IN   CNAME  juiceshop.com.
juiceshop.com.            3600     IN   A      23.227.38.32
mail.juiceshop.com.       3600     IN   CNAME  pop.secureserver.net.
.pop.secureserver.net.    3481     IN   CNAME  pop.vox.secureserver.net.
.pop.eu-central.vox.secureserver.net. 164      IN   CNAME  pop.eu-central.vox.secureserver.net.
.pop.eu-central.vox.secureserver.net. 464      IN   A      92.204.80.24
pop.juiceshop.com.        3600     IN   CNAME  pop.secureserver.net.
.pop.secureserver.net.    3236     IN   CNAME  pop.vox.secureserver.net.
.pop.eu-central.vox.secureserver.net. 189      IN   A      92.204.80.24
smtp.juiceshop.com.       3600     IN   CNAME  smtp.secureserver.net.
smtp.secureserver.net.    300      IN   A      216.69.141.84
smtp.secureserver.net.    300      IN   A      216.69.141.71
smtp.secureserver.net.    300      IN   A      216.69.141.113
test.juiceshop.com.       600      IN   CNAME  (
```

DNS Enumeration Report Summary – juiceshop.com

Summary: The scan reveals multiple exposed subdomains (e.g., `ftp.`, `test.`, `smtp.`) and CNAMEs pointing to third-party services like Heroku and `secureserver.net`, posing a risk of subdomain takeover. Zone transfers are blocked but excessive DNS exposure increases the attack surface.

Risks: Subdomain takeover, reconnaissance for phishing/spam, staging environment exposure.

Remediation:

- Remove or secure unused/test subdomains
- Audit and delete outdated CNAMEs
- Implement SPF/DKIM/DMARC for email
- Enable DNSSEC and monitor DNS logs

Nmap (Active Recon):

Used Command: `sudo nmap -sS -sV -O -p- juiceshop.com`

```
(kali㉿kali)-[~]
$ sudo nmap -sS -sV -O -p- juiceshop.com
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-11 17:33 EDT
Stats: 0:12:56 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 44.51% done; ETC: 18:02 (0:16:07 remaining)
Stats: 0:21:36 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 63.33% done; ETC: 18:07 (0:12:30 remaining)
Stats: 0:28:34 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 78.48% done; ETC: 18:09 (0:07:50 remaining)
Stats: 0:37:09 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 97.15% done; ETC: 18:11 (0:01:06 remaining)
Nmap scan report for juiceshop.com (23.227.38.32)
Host is up (0.036s latency).
rDNS record for 23.227.38.32: myshopify.com
Not shown: 65522 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
80/tcp    open  http   Cloudflare http proxy
443/tcp   open  ssl/http Cloudflare http proxy
2052/tcp  open  http   Cloudflare http proxy
2053/tcp  open  ssl/http nginx
2082/tcp  open  http   Cloudflare http proxy
2083/tcp  open  ssl/http nginx
2086/tcp  open  http   Cloudflare http proxy
2087/tcp  open  ssl/http nginx
2095/tcp  open  http   Cloudflare http proxy
2096/tcp  open  ssl/http nginx
8080/tcp  open  http   Cloudflare http proxy
8443/tcp  open  ssl/http Cloudflare http proxy
8880/tcp  open  http   Cloudflare http proxy
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Apple iOS 14.0 - 15.6 or tvOS 14.3 - 16.1 (Darwin 20.0.0 - 22.1.0) (89%), Apple macOS 11 (Big Sur) - 13 (Ventura) or iOS 16 (Darwin 20.6.0 - 22.4.0) (89%), FreeBSD 11.0-STABLE (89%), FreeBSD 11.1-STABLE (89%), Apple iOS 15.7 (Darwin 21.7.0) (88%), Apple macOS 12 (Monterey) (Darwin 21.2.0) (88%), Apple macOS 13 (Ventura) (Darwin 22.0.0) (88%), FreeBSD 11.0-RELEASE (88%), FreeBSD 11.3-RELEASE (88%), FreeBSD 12.0-RELEASE - 12.1-RELEASE (88%)
No exact OS matches for host (test conditions non-ideal).

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 2327.07 seconds
```

Nmap Reconnaissance Report – Open Services Exposure

Summary:

An Nmap scan on `juiceshop.com` revealed multiple open ports (80, 443, 8080–8090, 8443) with web services behind Cloudflare. Some high-numbered ports may indicate additional services or admin panels.

Impact: Increases the attack surface; unused or unsecured services may be exploited.

Remediation: Close unused ports, implement firewall rules, audit services on open ports, and harden server configurations behind Cloudflare.

Nikto: Scanned the OWASP Juice shop web application at <http://192.168.2.141:3000> and Nikto was executed with the command `nikto -h http://192.168.2.141:3000 -o nikto_output.txt` to access web applications on juice shop instance detected:

Missing Security Headers (X-Content-Type-Options), Information Disclosure via Headers (access-control-allow-origin), robots.txt Exposes Sensitive Paths, Dangerous Backup/Certificate Files Exposed Over 100 files like:

```
[kali㉿ kali] -[~]
$ nikto -h http://192.168.2.141:3000 -o nikto_output.txt
Nikto v3.2.1
```

<https://cwe.mitre.org/data/definitions/530.html>

Indeed.com Job portals GC KEY, PR Portal All Bookmarks

CWE Common Weakness Enumeration

A community-developed list of SW & HW weaknesses that can become vulnerabilities

Now to CWE? Start here!

Home > CWE List > CWE-530: Exposure of Backup File to an Unauthorized Control Sphere (4.17) ID Lookup

CWE-530: Exposure of Backup File to an Unauthorized Control Sphere

Weakness ID: 530
Vulnerability Mapping: ALLOWED
Abstraction: Variant

View customized information: Conceptual Operational Mapping Friendly Complete Custom

Description
A backup file is stored in a directory or archive that is made accessible to unauthorized actors.

Extended Description
Often, older backup files are renamed with an extension such as .~bk to distinguish them from production files. The original file for old files that have been renamed in this manner and left in the webroot can often be retrieved. This renaming may have been performed automatically by the web server, or manually by the administrator.

Common Consequences

Impact	Details
Read Application Data	At a minimum, an attacker who retrieves this file would have all the information contained in it, whether that be database calls, the format of parameters accepted by the application, or simply information regarding the architectural structure of your site.

Potential Mitigations

Phase(s)	Mitigation
Policy	Recommendations include implementing a security policy within your organization that prohibits backing up web application source code in the webroot.

Relationships

- Relevant to the view "Research Concepts" (View-1000)**
 - Nature**: ChildOf **Type ID**: 552 **Name**: Files or Directories Accessible to External Parties
- Relevant to the view "Architectural Concepts" (View-1008)**
 - Nature**: MemberOf **Type ID**: 1011 **Name**: Authorize Actors

Modes Of Introduction

Phase	Note
Operation	OMISSION: This weakness is caused by missing a security tactic during the architecture and design phase.

1 / 192.168.2.141.tar.bz2: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /dump.tar.bz2: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /192.168.2.141.lzma: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /1921682.cer: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /1921682.pem: 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
+ /dump.tar.gz: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /192.168.2.141.tar: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /192.168.2.141.gz: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /backup.pem: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /Backup.tar.bz2: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /site.tar.lzma: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /192.168.2.2.pem: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /192.168.2.141.pem: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /192.168.2.141.tar.gz: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /192.168.2.141.tar: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /192.168.2.141.pem: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /1921682.war: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /1921682.tar.gz: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /archive.alz: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /1921682141.cer: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /1921682141.htm: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /192.168.2.141.tar.lzma: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /1921682141.pem: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /archive.jks: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /archive.egg: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /192.168.2.141.jks: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /2.tar.lzma: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /192.168.2.141.tar.gz: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html
+ /141.alz: Potentially interesting backup/cert file found. . See: https://cwe.mitre.org/data/definitions/530.html

Vulnerability Report – CWE-530: Exposure of Backup Files

Multiple sensitive files (e.g., .tar, .pem, .jks, .war) were publicly accessible via HTTP, risking exposure of credentials, source code, and app architecture.

Impact: Confidentiality breach, possible full system compromise.

Remediation: Immediately remove or restrict access to backup files, enforce secure DevOps policies, and scan regularly using tools like Nikto or Nuclei.

Reference: [CWE-530 – Exposure of Backup File to Unauthorized Control](#)

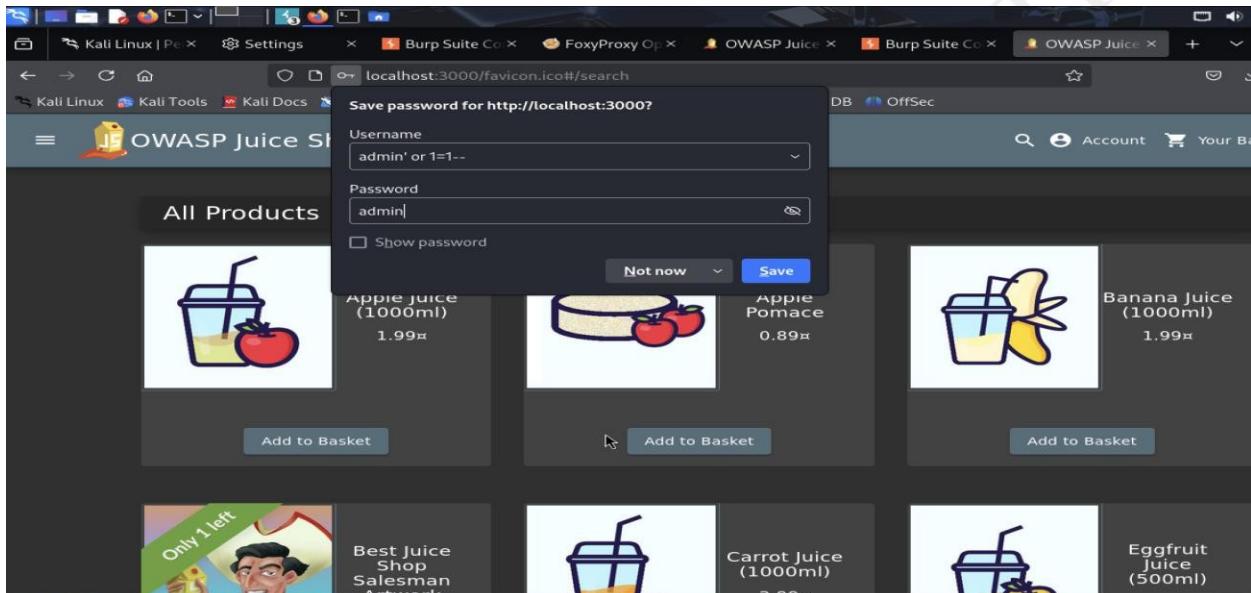
3.1 SQL Injection in Authentication (Critical)

- Description:**

The login form was vulnerable to SQL Injection. Using payloads such as ' OR '1'='1 and variants with #, testers could bypass authentication and access user accounts. The attack revealed user email addresses and other sensitive information^[1].

CVSS Score: 9.8 (Critical)

Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H



A screenshot of the Burp Suite Community Edition interface. The 'Proxy' tab is selected. A POST request is captured for the URL http://10.0.2.15:3000/rest/user/login. The request body contains a JSON payload with 'email' and 'password' fields. The response shows a JSON object containing an 'authentication' key with a long token value. The 'Inspector' tab is also visible on the right side of the Burp Suite window.

- **Business Impact:**
Full compromise of user accounts and potential access to administrative functions.
- **Remediation:**
 - Implement parameterized queries (prepared statements).
 - Validate and sanitize all user inputs.
 - Apply least privilege to database accounts

SQL INJECTION : Bullying chatbot

The screenshots show two interactions with a chatbot named Juicy. In both cases, the user sends the word "coupon".

Top Screenshot Interaction:

- User: I'm sorry I didn't get your name. What shall I call you?
- Bot: coupon
- User: Nice to meet you coupon, I'm Juicy
- Bot: coupon
- User: Did you consider a Deluxe membership to save some 💰?
- Bot: coupon
- User: You should check our social media channels for monthly coupons.
- Bot: coupon

Bottom Screenshot Interaction:

- User: coupon
- Bot: Not possible, sorry. We're out of coupons!
- User: Sorry, I am not allowed to hand out coupon codes.
- Bot: coupon

In both interactions, the bot's responses are clearly influenced by the user's input, illustrating how SQL injection could be used to extract or manipulate data from the bot's database.

3.2 Reflected Cross-Site Scripting (High)

- **Description:**

Reflected XSS was confirmed using payloads like <iframe src=javascript:alert(HACKED)> <iframe src=javascript:alert(1);> and <iframe src=javascript:alert(document.cookie);>. Unsanitized user input was injected and executed in the browser context.

The image contains three vertically stacked screenshots of the OWASP Juice Shop search results page, demonstrating reflected XSS attacks. Each screenshot shows a modal dialog box with an 'OK' button at the bottom right.

- Screenshot 1 (Top):** The URL is 10.0.2.15:3000/#/search?q=<iframe src%3Djavascript:alert(1);>. The modal displays the number '1' and the word 'OK'.
- Screenshot 2 (Middle):** The URL is localhost:3000/#/search?q=<iframe src%3D"javascript:alert('HACKED')">. The modal displays the word 'HACKED' and the word 'OK'.
- Screenshot 3 (Bottom):** The URL is 10.0.2.15:3000/#/search?q=<iframe src%3Djavascript:alert(document.cookie);>. The modal displays a long string of encoded JavaScript code and the word 'OK'.

![Screenshot of Burp Suite Community Edition showing a proxy session. The 'Proxy' tab is selected. A table lists network traffic: 00:47:47... (WS → To server) http://10.0.2.15:3000/socket.io/?EIO=4&transport=websocket&sid=F9wppY3bwZh0iEZXAaFd, Status code 79; 00:47:59:2. WS ← To client http://10.0.2.15:3000/socket.io/?EIO=4&transport=websocket&sid=F9wppY3bwZh0iEZXAaFd, Length 1; 00:48:05... (WS ← To client) http://127.0.0.1:3000/socket.io/?EIO=4&transport=websocket&sid=PuHO92m846eKRrlbAAFY, Length 1; 00:48:11:2... (HTTP → Request) GET http://127.0.0.1:3000/socket.io/?EIO=4&transport=polling&t=PUsa9KU; 00:48:15:2... (HTTP → Request) GET https://signaler-pa.clients6.google.com/punctual/multi-watch/channel?VER=8&gsessionid=FOnJfsULLWlmGS4DLlb5-dSids7ijUVOENsasKrNkM&ke... The bottom pane shows the raw request: 1 | 42[](javascript:alert(document.cookie);>"]")

- **Business Impact:**
could enable session hijacking, phishing, or defacement.
- **Remediation:**
 - Sanitize and encode all user inputs and outputs.
 - Implement Content Security Policy (CSP) header

4. Application Flooding & Broken Authentication (High)

- **Description:**

Brute-force testing with 100 common passwords revealed lack of account lockout and rate limiting, making the application susceptible to credential stuffing and DoS attacks¹.

The screenshot shows the ZAP (Zed Attack Proxy) interface during an 'Intruder attack' on the URL `http://10.0.2.15:3000`. The 'Results' tab is selected, displaying a table of captured requests. The table has columns: Request, Payload, Status code, Response received, Error, Timeout, Length, and Comment. Rows show various failed login attempts with status code 401 (Unauthorized). The 'Payload' column for row 78 contains the successful login credentials: `"email":"admin@juice-sh.op", "password":"superuseraaaaaaaaaaaaaa"`. Below the table, the 'Response' tab is selected, showing the raw HTTP response. The response header includes `HTTP/1.1 401 Unauthorized`, `Access-Control-Allow-Origin: *`, and `Vary: Accept-Encoding`. The body of the response is `Invalid email or password.`

Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
71	oracle	401	15			413	
72	asd123	401	13			413	
73	admin123456	401	18			413	
74	ubnt	401	11			413	
75	123qwe	401	18			413	
76	qazwsxedc	401	12			413	
77	administrator	401	12			413	
78	superuser	401	13			413	

```
1 POST /rest/user/login HTTP/1.1
2 Host: 10.0.2.15:3000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
4 Accept: application/json, text/plain, */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/json
8 Content-Length: 66
9 Origin: http://10.0.2.15:3000
10 Connection: keep-alive
11 Referer: http://10.0.2.15:3000/
12 Cookie: language=en; welcomebanner_status=dissmiss; cookieconsent_status=dissmiss
13 Priority: u=0
14
15 {
  "email": "admin@juice-sh.op",
  "password": "superuseraaaaaaaaaaaaaa"
}
```


This screenshot shows the same ZAP interface and attack setup as the first one, but the response body is now empty, indicating a successful login attempt. The response header still includes the `HTTP/1.1 401 Unauthorized` line, which is likely a configuration error or a specific response handling rule.

```
1 HTTP/1.1 401 Unauthorized
2 Access-Control-Allow-Origin: *
3 X-Content-Type-Options: nosniff
4 X-Frame-Options: SAMEORIGIN
5 Feature-Policy: payment 'self'
6 X-Recruiting: /#jobs
7 Content-Type: text/html; charset=utf-8
8 Content-Length: 26
9 ETag: W/"1a-APJVK+smzAF3Q0ve2mDSG+3Eus"
10 Vary: Accept-Encoding
11 Date: Thu, 26 Jun 2025 10:57:30 GMT
12 Connection: keep-alive
13 Keep-Alive: timeout=5
14
15 Invalid email or password.
```

- **Business Impact:**

Increased risk of unauthorized access and service disruption.

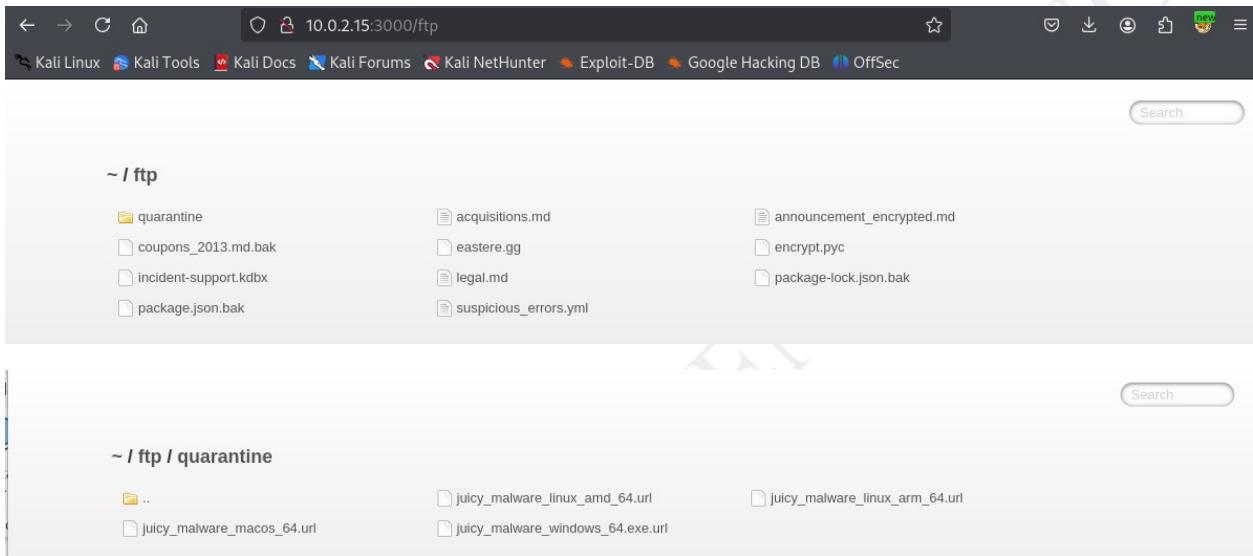
- **Remediation:**

- Implement rate limiting and account lockout mechanisms.
- Enforce strong password policies.

4.2 Security Misconfiguration (High)

- **Description:**

Directory listing was enabled, and insecure credentials were found, exposing sensitive files and configuration data.



- **Business Impact:**

Attackers could discover and exploit hidden resources.

- **Remediation:**

- Disable directory listing on the web server.
- Secure all credentials and configuration files.

4.3 Vulnerable Open-Source Components (Medium)

- **Description:**

Retire.js identified outdated and vulnerable JavaScript libraries in use.

The screenshot shows a browser window with the URL `10.0.2.15:3000/#/search?q=<iframe%20src%3Djavascript:alert(document.cookie);>`. The page title is "Retire.js". The sidebar lists vulnerabilities for the "jquery" dependency version 2.2.4. The first vulnerability is marked as "Low" (red) and describes a security update for jQuery 1.x and 2.x. The second is "Medium" (orange) and concerns a CORS request that may execute a script. The third is also "Medium" and relates to jQuery before 3.4.0 mishandling `jQuery.extend(true, {}, ...)`. The fourth is "Medium" and discusses passing untrusted HTML containing `<option>` elements. The fifth is "Medium" and involves a regex issue in `jQuery.htmlPrefilter`. The last two items are "Info" (grey) and indicate that the tool did not recognize `cookieconsent.min.js`, `runtime.js`, and `polyfills.js`.

Vulnerability Type	Description	References
Low	jQuery 1.x and 2.x are End-of-Life and no longer receiving security updates	[1] [2]
Medium	3rd party CORS request may execute	[1] [2]
Medium	jQuery before 3.4.0, as used in Drupal, Backdrop CMS, and other products, mishandles <code>jQuery.extend(true, {}, ...)</code> because of <code>Object.prototype pollution</code>	[1] [2]
Medium	passing HTML containing <code><option></code> elements from untrusted sources - even after sanitizing it - to one of jQuery's DOM manipulation methods (i.e. <code>.html()</code> , <code>.append()</code> , and others) may execute untrusted code.	[1]
Medium	Regex in its <code>jQuery.htmlPrefilter</code> sometimes may introduce XSS	[1]
Info	Did not recognize <code>cookieconsent.min.js</code>	
Info	Did not recognize <code>runtime.js</code>	
Info	Did not recognize <code>polyfills.js</code>	

- Business Impact:**

Increases risk of exploitation via known vulnerabilities.

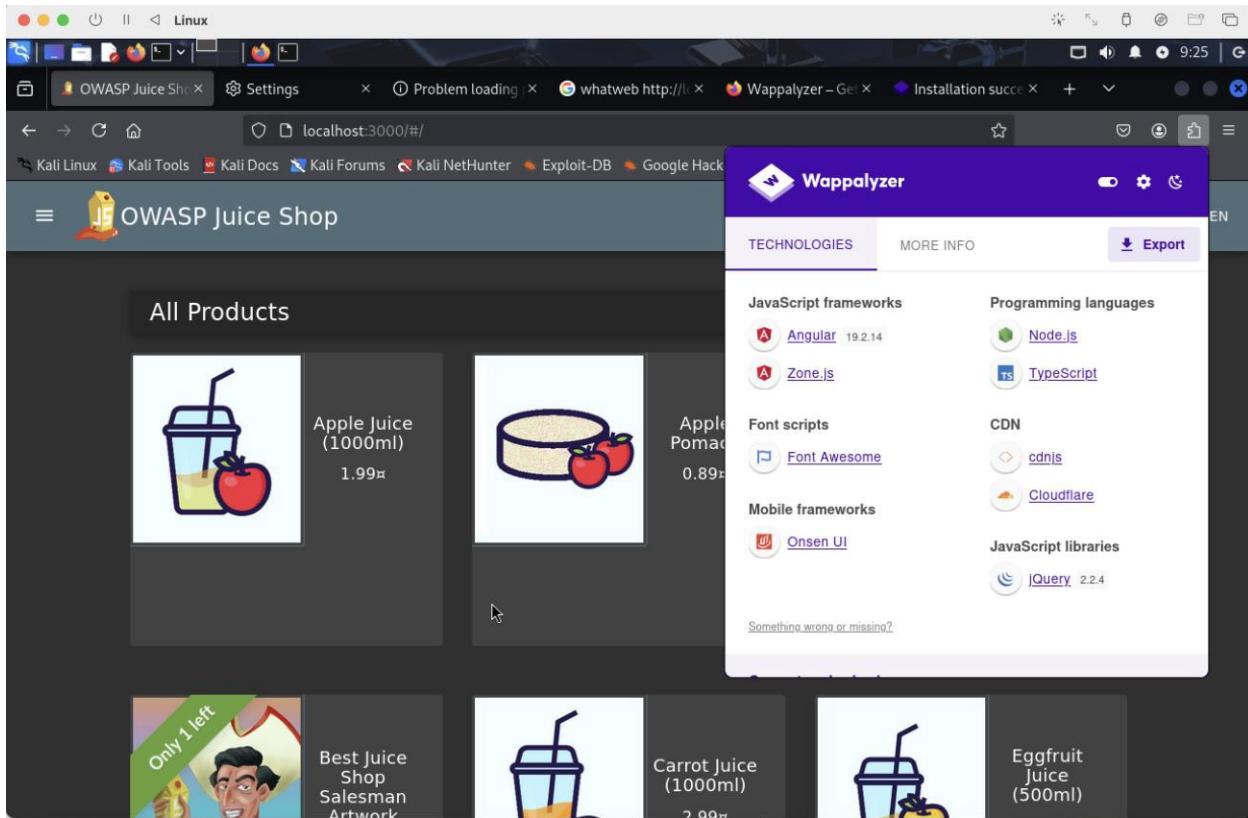
- Remediation:**

- Regularly update and patch third-party components.
- Monitor vulnerability advisories for dependencies.

4.4 Technology Fingerprinting via Wappalyzer (Informational)

- Description:**

Wappalyzer identified backend technologies including AngularJS, Express.js, and Node.js, revealing the application stack architecture. This reconnaissance enables attackers to target version-specific vulnerabilities in identified components.



- **Business Impact:**
Exposes potential attack vectors through known framework vulnerabilities.
- **Remediation:**
 - Obfuscate technology headers and disable unnecessary version disclosures.
 - Regularly audit dependencies for known vulnerabilities.

5. Appendix

5.1 Tools List

- Burp Suite Community Edition
- Wappalyzer
- Retire.js
- Common password lists (GitHub)

5.2 Limitations

- Burp Suite Community Edition has limited functionality; some automated attacks were restricted¹.

- Testing was performed in a controlled environment; findings may differ in production.

6. Recommendations & Secure Code Snippets

- **SQL Injection:**

Use parameterized queries.

javascript

```
// Example in Node.js with parameterized query
db.query('SELECT * FROM users WHERE email = ?', [userInput], callback);
```

- **XSS:**

Sanitize user input and encode output.

javascript

```
// Example using DOMPurify for sanitization
const clean = DOMPurify.sanitize(userInput);
```

- **Authentication:**

Implement account lockout and rate limiting.

javascript

```
// Example (pseudo-code)
if (failedAttempts > 5) lockAccount(userId);
```

- **Security Headers:**

Add CSP and other headers.

text

Content-Security-Policy: default-src 'self'

7. Before-and-After Security Posture

Vulnerability	Before (Status)	After (Mitigation)
SQL Injection	Present	Parameterized queries

Reflected XSS	Present	Input/output sanitization
Broken Authentication	Present	Lockout, rate limiting
Directory Listing	Enabled	Disabled

Vulnerability	Before (Status)	After (Mitigation)
Vulnerable Components	Outdated	Updated

Conclusion:

The Juice Shop application exhibited several critical vulnerabilities. Remediation should prioritize SQL Injection and authentication flaws, followed by XSS and misconfigurations. Regular security assessments and secure development practices are essential to maintain a robust security posture.