

1. Introduction

OWASP Juice Shop is a deliberately insecure web application created to teach web security through hands-on, realistic exercises. It intentionally contains a wide variety of vulnerabilities — from classic SQL injection and XSS to authentication weaknesses and insecure direct object references — mapped closely to the OWASP Top 10. Because it's gamified with flags and challenges, Juice Shop is ideal for both beginners learning the basics of web security and experienced practitioners practicing advanced exploitation techniques.

This report documents a structured penetration test performed against an instance of OWASP Juice Shop hosted on the TryHackMe platform. The goal was not only to capture challenge flags but to treat the exercise as a mini-audit: identify weaknesses, exploit them responsibly in the controlled lab environment, verify impact, and recommend practical remediations. The exercises covered reconnaissance, client-side analysis, manual exploitation using Burp Suite and browser developer tools, and limited automated testing (Burp Intruder) where appropriate.

Beyond technical verification, this assignment focuses on the real-world relevance of each vulnerability: how it could be abused in a production environment, what sensitive data or functionality is at risk, and which development or configuration changes would most effectively reduce that risk. The findings and mitigation suggestions that follow are therefore written with both developers and security reviewers in mind.

2. Methodology

The testing was carried out using:

- Kali Linux (attacker machine)
- Burp Suite (intercepting and modifying HTTP requests)
- TryHackMe VPN connection (for access to the lab environment)
- Browser Dev Tools (for analyzing JavaScript and client-side logic)

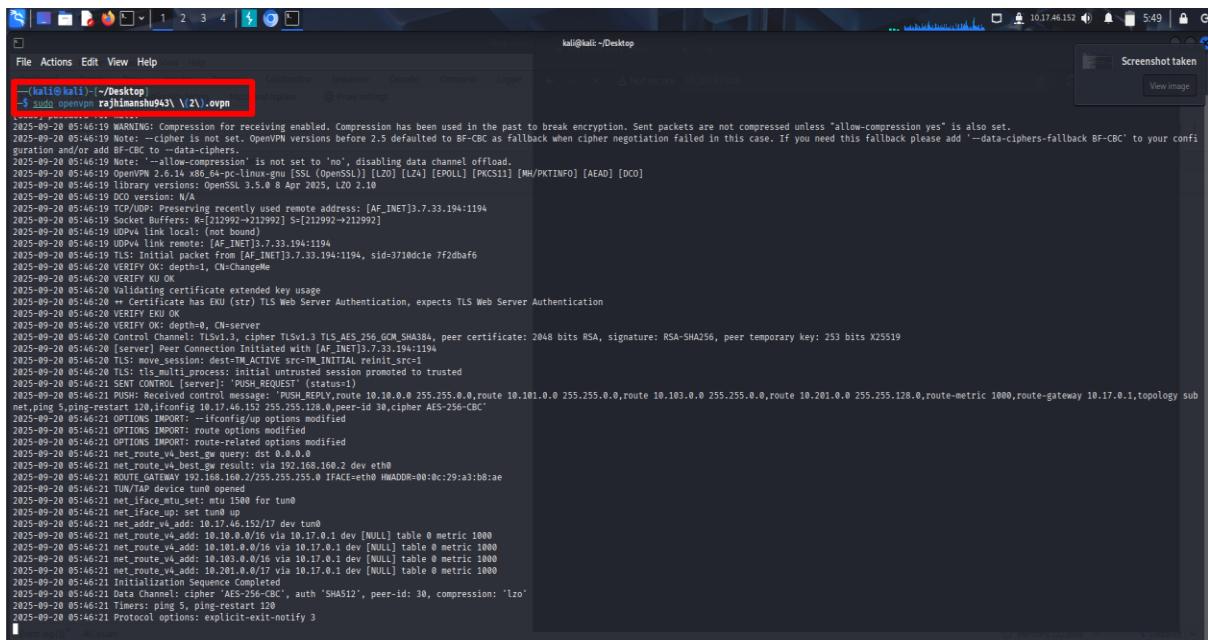
General approach followed:

1. Reconnaissance – browsing the application and intercepting requests.
2. Exploitation – performing SQLi, XSS, brute-force, and directory traversal.
3. Verification – confirming vulnerabilities by obtaining flags.
4. Documentation – recording exploitation steps and remediation.

3. Findings & Exploitation

Task 1: Open for Business!

Question 1: Deploy the VM attached to this task to get started! You can access this machine by using your [browser-based machine](#), or if you're connected through [OpenVPN](#).



```
File Actions Edit View Help Help Collaborator Sequencer Decoder Computer Logger X Preferences Help Network 10.201.57.109 10.17.46.152 5:49 Screenshot taken View image
(kali㉿kali)-[~/Desktop]
$ sudo openvpn --config /path/to/your-file.ovpn
[~(kali㉿kali)-[~/Desktop]
$ sudo openvpn --config /path/to/your-file.ovpn
2025-09-20 05:46:19 WARNING: Compression for receiving enabled. Compression has been used in the past to break encryption. Sent packets are not compressed unless "allow-compression yes" is also set.
2025-09-20 05:46:19 Note: --cipher is not set. OpenVPN versions before 2.5 default to BF-CBC as fallback when cipher negotiation failed in this case. If you need this fallback please add '--data-ciphers-fallback BF-CBC' to your configuration and/or add BF-CBC to --data-ciphers.
2025-09-20 05:46:19 Note: --allow-compression is not set to 'no', disabling data channel offload.
2025-09-20 05:46:19 OpenVPN 2.6.10 x86_64-pc-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL] [PKCS11] [MH/PKTINFO] [AEAD] [DCO]
2025-09-20 05:46:19 Library: OpenSSL 3.1.5.0 8 Apr 2025, LZO 2.16
2025-09-20 05:46:19 DCO version: N/A
2025-09-20 05:46:19 TCP/UDP: Preserving recently used remote address: [AF_INET]3.7.33.194:1194
2025-09-20 05:46:19 Socket Buffers: R=[212992->212992] S=[212992->212992]
2025-09-20 05:46:19 UDPv4 link local: [none bound]
2025-09-20 05:46:19 UDPv4 remote: [AF_INET]3.7.33.194:1194
2025-09-20 05:46:19 TLS: Initial packet from [AF_INET]3.7.33.194:1194, sid=7f2dbaf6
2025-09-20 05:46:20 VERIFY OK: depth=1, C=ChangeMe
2025-09-20 05:46:20 VERIFY OK: depth=0, C=ChangeMe
2025-09-20 05:46:20 Validating certificate extended key usage
2025-09-20 05:46:20 ** Certificate has EKU (str) TLS Web Server Authentication, expects TLS Web Server Authentication
2025-09-20 05:46:20 VERIFY OK: depth=1, C=ChangeMe
2025-09-20 05:46:20 VERIFY OK: depth=0, C=ChangeMe
2025-09-20 05:46:20 Control Channel: TLSv1.3 cipher AES_256_GCM_SHA384, peer certificate: 2048 bits RSA, signature: RSA-SHA256, peer temporary key: 253 bits X25519
2025-09-20 05:46:20 [server] Peer Connection Initiated with [AF_INET]3.7.33.194:1194
2025-09-20 05:46:20 TLS: move session: dest=TM_ACTIVE src=TM_INITIAL_src=1
2025-09-20 05:46:20 TLS: tis multi_process: initial untrusted session promoted to trusted
2025-09-20 05:46:20 [server] POST /index (stusius)
2025-09-20 05:46:21 PUSH Received configuration: <PSH>HTTP/2.0 10.18.0.0 255.255.0.0,route 10.101.0.0 255.255.0.0,route 10.103.0.0 255.255.0.0,route 10.201.0.0 255.255.128.0,route-metric 1000,route-gateway 10.17.0.1,topology sub
2025-09-20 05:46:21 ping 5, ping-restart 120,ifconfig 10.17.46.157 255.255.128.0,peer-id 30,cipher AES-256-GCM
2025-09-20 05:46:21 OPTIONS IMPORT: --lifecap/up options modified
2025-09-20 05:46:21 OPTIONS IMPORT: route options modified
2025-09-20 05:46:21 OPTIONS IMPORT: route-related options modified
2025-09-20 05:46:21 net_route_v4_best_nexthop query: dst 0.0.0.0
2025-09-20 05:46:21 net_route_v4_set: dst 0.0.0.0 via 192.168.100.2 dev eth0
2025-09-20 05:46:21 ROUTE_GATEWAY 192.168.100.2/255.255.255.0 IFACE=eth0 HWADDR=00:0c:29:a3:b8:ae
2025-09-20 05:46:21 TUN/TAP device tun0 opened
2025-09-20 05:46:21 net_iface_mtu_set: mtu 1500 for tun0
2025-09-20 05:46:21 net_iface_up: set tun0
2025-09-20 05:46:21 net_iface_up: dev tun0
2025-09-20 05:46:21 net_iface_up: 10.17.0.1/17 dev tun0
2025-09-20 05:46:21 net_route_v4_add: 10.101.0.0/16 via 10.17.0.1 dev [NULL] table 0 metric 1000
2025-09-20 05:46:21 net_route_v4_add: 10.103.0.0/16 via 10.17.0.1 dev [NULL] table 0 metric 1000
2025-09-20 05:46:21 net_route_v4_add: 10.201.0.0/17 via 10.17.0.1 dev [NULL] table 0 metric 1000
2025-09-20 05:46:21 Initialization Sequence Completed
2025-09-20 05:46:21 Data Channel: cipher AES-256-GCM, auth SHA512, peer-id: 30, compression: 'lzo'
2025-09-20 05:46:21 Timers: ping 5, ping-restart 120
2025-09-20 05:46:21 Protocol options: explicit-exit-notify 3
```

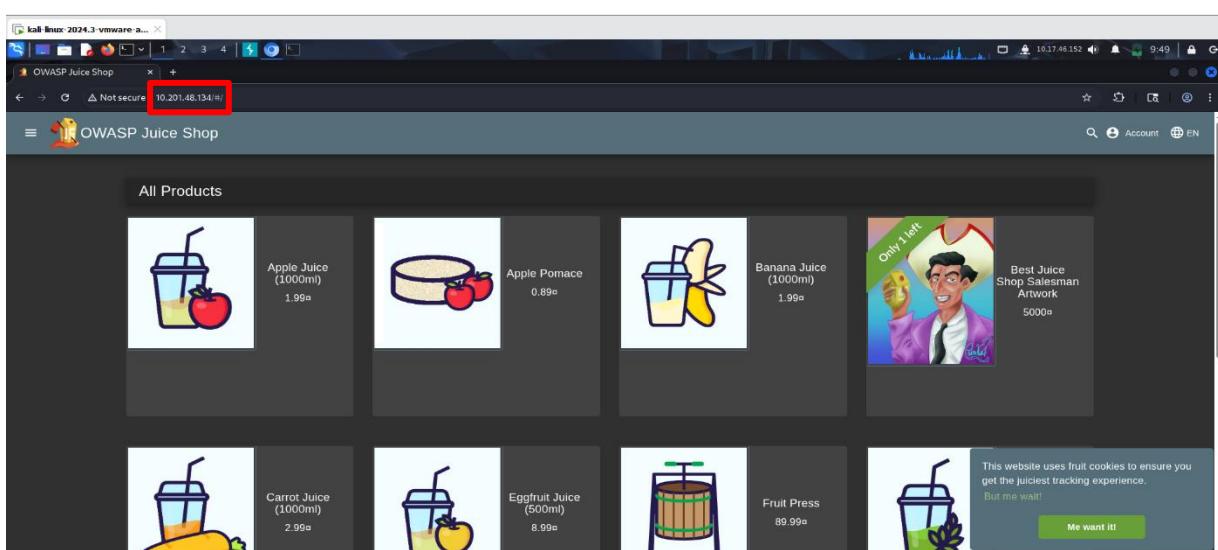
Step 1: Download the VPN .ovpn file from the Access Machine button on the TryHackMe room page.

Step 2: In Kali run:

```
sudo openvpn --config /path/to/your-file.ovpn
```

When it shows Initialization Sequence Completed, we're connected.

Question 2: Once the machine has loaded, access it by copying and pasting its IP into your browser; if you're using the browser-based machine, paste the machines IP into a browser on that machine.

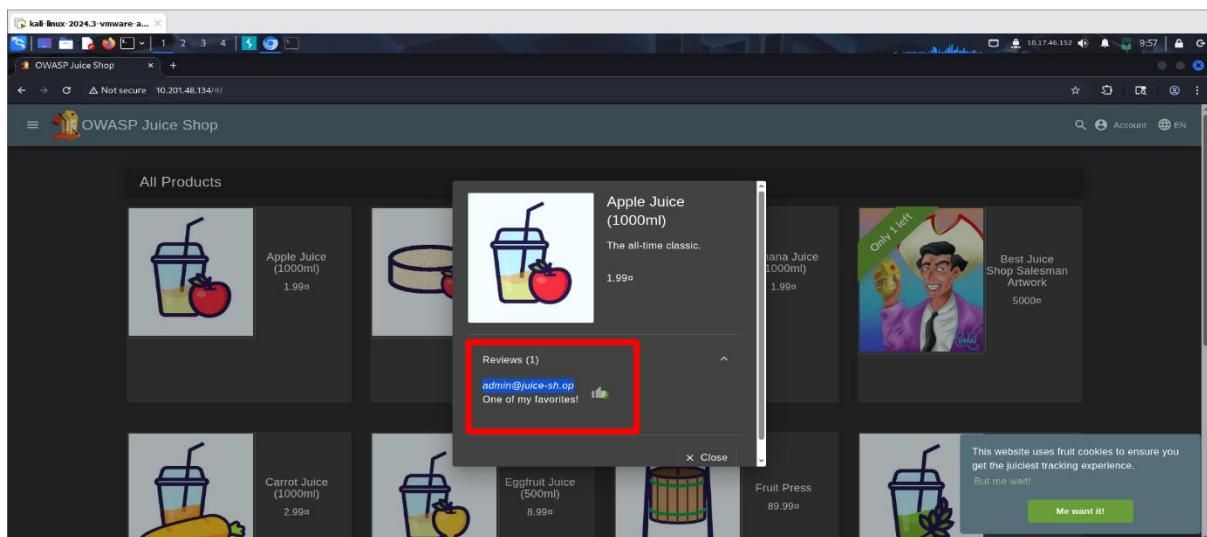


Step 1: Get the **Target IP** from the TryHackMe room.

Step 2: Open Burp → Proxy → **Open Browser** and enter **http://10.201.48.134**. The site will load through Burp so we can inspect and modify requests/responses.

Task 2: Let's go on an adventure!

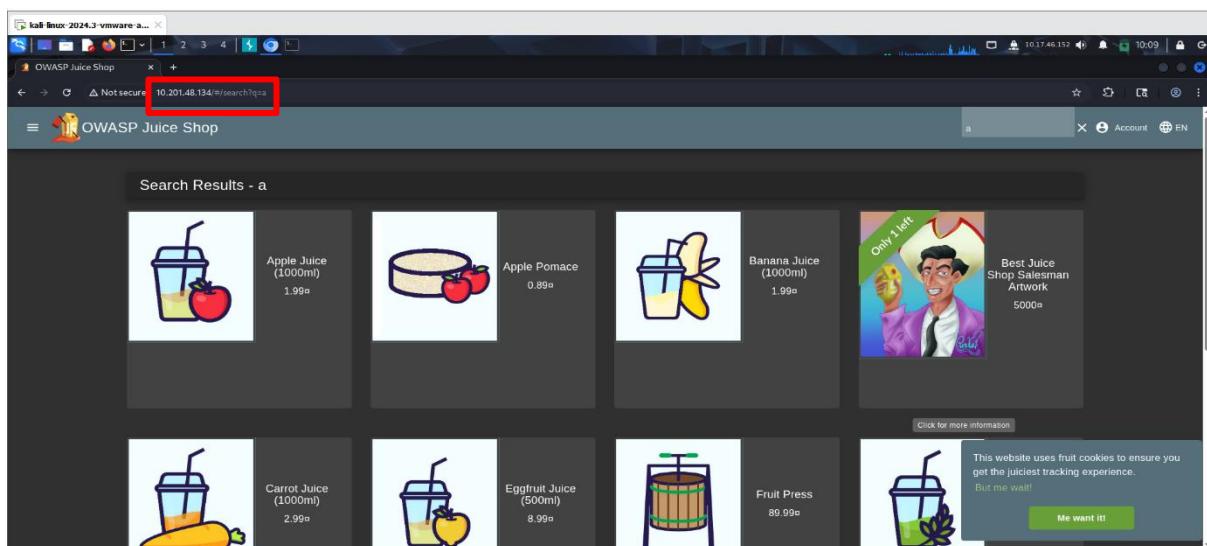
Question 1: What's the Administrator's email address?



Step 1: click on the apple juice product.

Step 2: Scroll to the Reviews section — the admin email appears there (info leakage).

Question 2: What parameter is used for searching?



Step 1: Click the magnifying glass to open the search bar.

Step 2: Type any text (e.g., a) and press Enter.

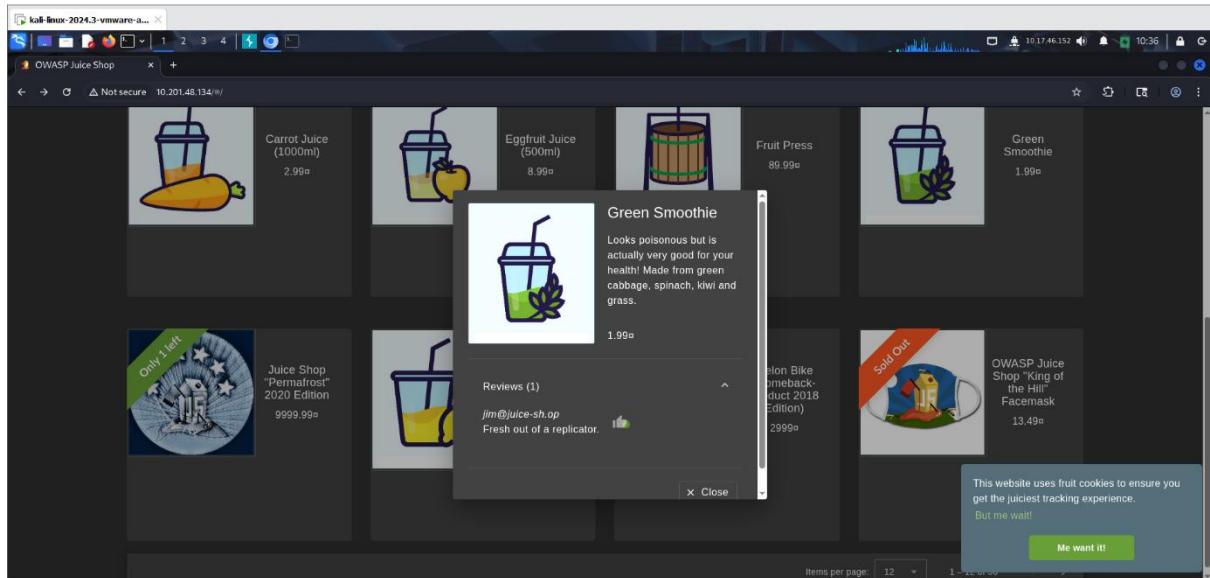
Step 3: Observe the URL change to:

<http://10.201.48.134/#/search?q=a>

Step 4: The part after ? is the search parameter.

Answer: q

Question 3: What show does Jim reference in his review?



Step 1: Click on the Green Smoothie product and check the reviews. One review mentions “replicator”.

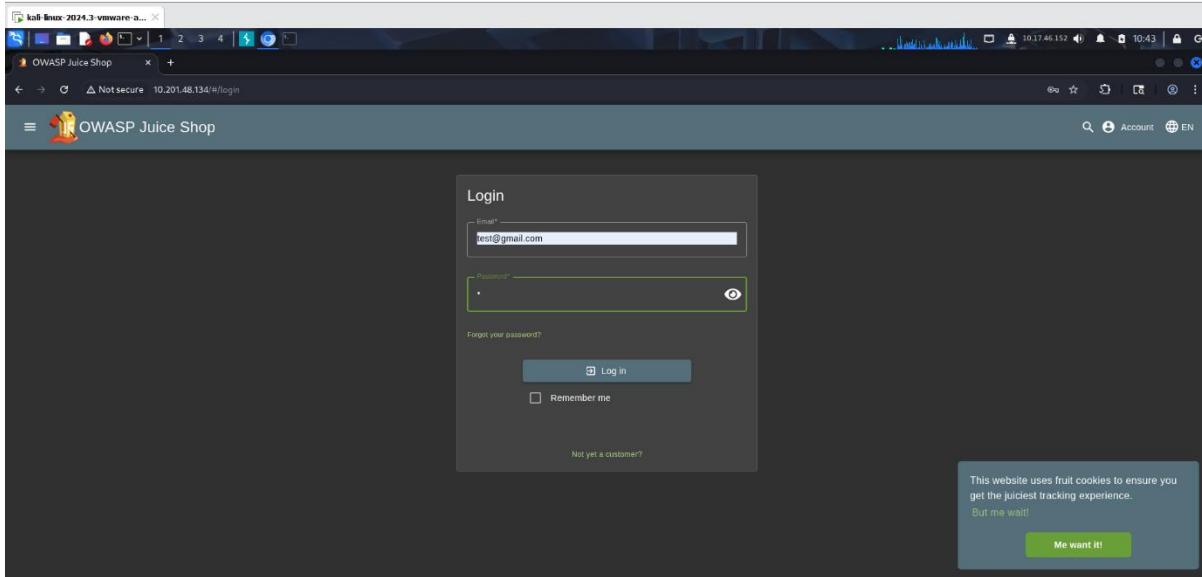
The screenshot shows the Wikipedia article for "Replicator (Star Trek)". The page content describes the replicator as a machine that can create food and drink. It mentions its first appearance in "Star Trek: The Next Generation" and its creation by Gene Roddenberry. The sidebar on the right allows users to customize the appearance of the text and images on the page.

Step 2: Search “replicator” on Google — it shows that it is related to the famous movie series Star Trek.

Answer: Star Trek

Task 3: Inject the juice

Question 1: Log into the administrator account!

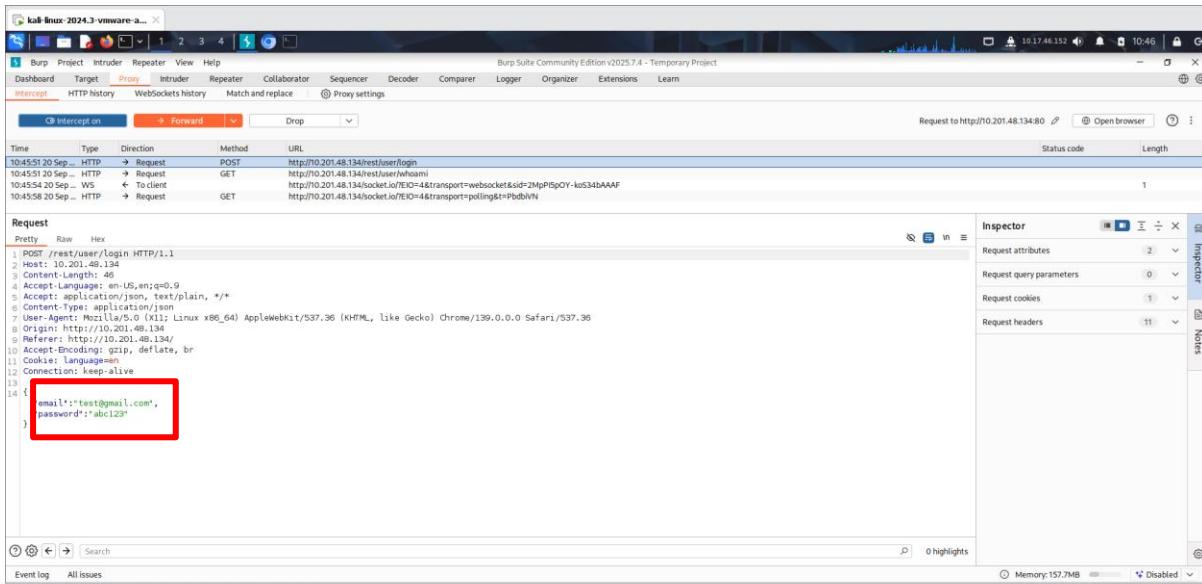


Step 1: Click on the Account section — the login page appears.

Step 2: Enter any random text in Email and Password fields (e.g., test@gmail.com and a).

Step 3: Open Burp Suite and go to Proxy → Intercept.

Step 4: Click the Login button on the webpage — Burp will intercept the request.



Step 5: In the intercepted request, replace the Email value with the SQL injection payload: '`' or 1=1--`

Leave the password as it is

Step 6: Forward the request in Burp. If successful, the login bypasses authentication and we can access the account to capture the flag.

You successfully solved a challenge: Login Admin (Log in with the administrator's user account.)
690fa3247a390651e0b26f947ba0b79b4f404a9

Your Basket (admin@juice-sh.op)

| Item | Quantity | Price |
|------------------------|----------|-------|
| Apple Juice (1000ml) | 2 | 1.99€ |
| Orange Juice (1000ml) | 3 | 2.99€ |
| Eggfruit Juice (500ml) | 1 | 8.99€ |

This website uses fruit cookies to ensure you get the juiciest tracking experience.
But me wait!

Why: The payload '`or 1=1--`' manipulates the SQL query to always return true, bypassing authentication. Burp intercepts the request so we can modify it before it reaches the server.

Question 2: Log into the Bender account!

Login

Email:

Password:

Remember me

Not yet a customer?

This website uses fruit cookies to ensure you get the juiciest tracking experience.
But me wait!

Step 1: Open the Login page and enter:

Email: `bender@juice-sh.op`

Password: `a`

Step 2: Open Burp Suite and go to Proxy → Intercept.

Step 3: Click the Login button on the webpage — Burp will intercept the request.

Step 4: In the intercepted request, modify the Email field to:

`bender@juice-sh.op'--`

Leave the password unchanged.

Burp Suite Community Edition v2025.7.4 - Temporary Project

Request to <http://10.201.48.134:80>

| Time | Type | Direction | Method | URL | Status code | Length |
|---------------------|------|-----------|--------|---|-------------|--------|
| 11:05:34 20 Sep ... | HTTP | → Request | POST | http://10.201.48.134/rest/user/login | | |
| 11:05:34 20 Sep ... | HTTP | → Request | GET | http://10.201.48.134/rest/user/whoami | | |

Request

```

Pretty Raw Hex
1 POST /rest/user/Login HTTP/1.1
2 Host: 10.201.48.134
3 Content-Length: 42
4 Accept-Language: en-US,en;q=0.9
5 Accept: application/json, text/plain, */*
6 Content-Type: application/json
7 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/139.0.0.0 Safari/537.36
8 Origin: http://10.201.48.134/
9 Referer: http://10.201.48.134/
10 Accept-Encoding: gzip, deflate, br
11 Content-Language: en
12 Connection: keep-alive
13
14 "email": "bender@juice-sh.op", --,
   "password": "a"

```

Inspector

Request attributes
Request query parameters
Request cookies
Request headers

Step 5: Forward the request in Burp. If successful, the page will log in and display the flag.

OWASP Juice Shop

You successfully solved a challenge: Login Bender (Log in with Bender's user account.)

5ff5052e879e6fe64124e64c82c84ebc809c6d4 Copy to clipboard

Your Basket (bender@juice-sh.op)

Raspberry Juice (1000ml) 1 4.99€

Total Price: 4.99€

Checkout

This website uses fruit cookies to ensure you get the juiciest tracking experience.
But me wait!

Me want it!

Why: Adding '--' comments out the rest of the SQL query, bypassing password verification and allowing login.

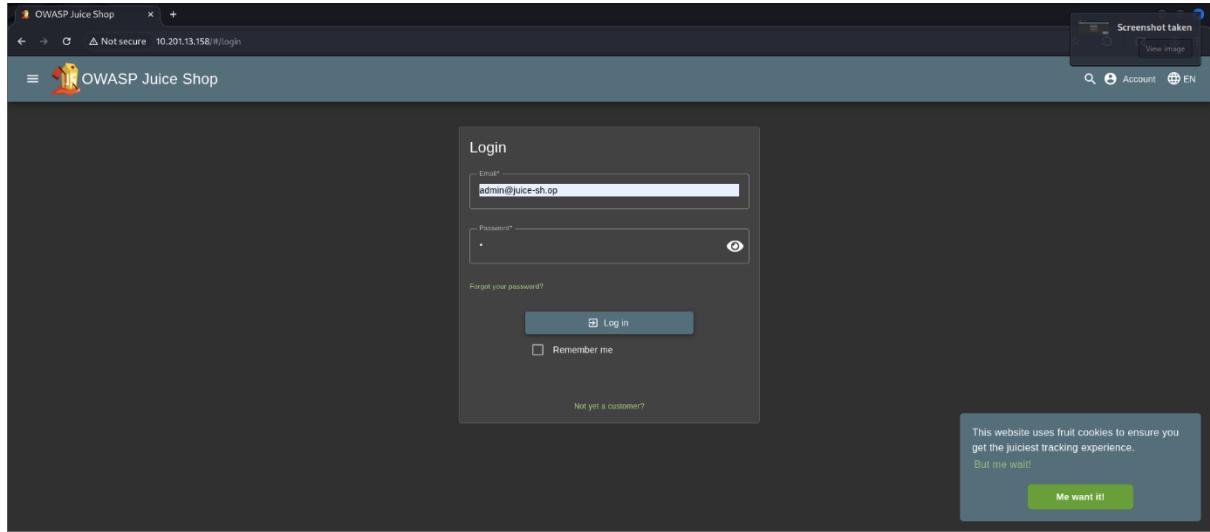
Task 4: Who broke my lock?!

Question 1: Brute force the Administrator account's password!

Step1: Open the Login page and enter:

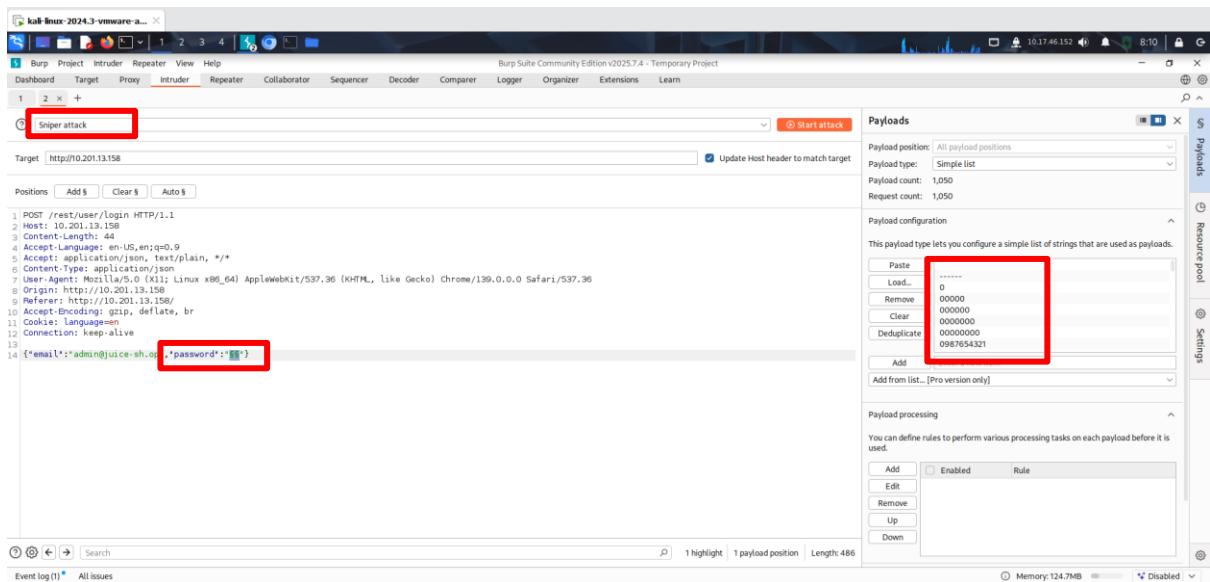
Email: admin@juice-sh.op

Password: a



Step 2: Open Burp Suite and Intercept the login request.

Step 3: Send the intercepted request to Intruder and set the attack type to Sniper.



The screenshot shows the Burp Suite interface with the Intruder tool open. A 'Sniper attack' is selected in the dropdown menu. The payloads list contains a single payload: '0' followed by 1050 zeros. The payload configuration shows a list of 1050 payloads starting with '0' and ending with '0987654321'. The payload processing section is empty.

Step 4: Place the § marker on the Password field and load the payload list best1050.txt. Start the attack.

Step 5: After some time, the correct password is found: admin123

2. Intruder attack of http://10.201.13.158

| Request | Payload | Status code | Response received | Error | Timeout | Length | Comment |
|---------|---------------|-------------|-------------------|-------|---------|--------|---------|
| 1 | POST / | 401 | 409 | | | 413 | |
| 111 | access | 401 | 419 | | | 413 | |
| 112 | access14 | 401 | | | | 413 | |
| 113 | account | 401 | 432 | | | 413 | |
| 114 | action | 401 | 405 | | | 413 | |
| 115 | admin | 401 | 418 | | | 413 | |
| 116 | admin1 | 401 | 316 | | | 413 | |
| 117 | admin12 | 401 | 302 | | | 413 | |
| 118 | admin123 | 200 | 346 | | | 1185 | |
| 119 | adminadmin | 401 | 521 | | | 413 | |
| 120 | administrator | 401 | 318 | | | 413 | |
| 121 | adriana | 401 | 409 | | | 413 | |
| 122 | agosto | 401 | 409 | | | 413 | |
| 123 | agustin | 401 | 409 | | | 413 | |
| 124 | ... | ... | ... | | | ... | |
| 14 | *email | | | | | | |

Request Response

```

Pretty Raw Hex
Content-Type: application/x-www-form-urlencoded
Accept-Language: en-US;q=0.9
Accept: application/json, text/plain, */*
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/139.0.0.0 Safari/537.36
Origin: http://10.201.13.158
Referer: http://10.201.13.158/
Accept-Encoding: gzip, deflate, br
Cookie: language=en
Connection: keep-alive
Content-Length: 23
Connection: keep-alive
14 [
  *email*':':admin@juice-sh.op",
  *password*':':admin123"
]

```

Event log (127 of 150) To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

Step 6: Use this password to log in as admin and capture the flag.

You successfully solved a challenge: Password Strength (Log in with the administrator's user credentials without previously changing them or applying SQL Injection.)

ff4ae0ffe31b0ffdea9bdd0207a16a3c01ac6c56 Copied!

You successfully solved a challenge: Login Admin (Log in with the administrator's user account.)

ff90fa3247a99d51e0b28f947ba0b79b4f404a9 Copied!

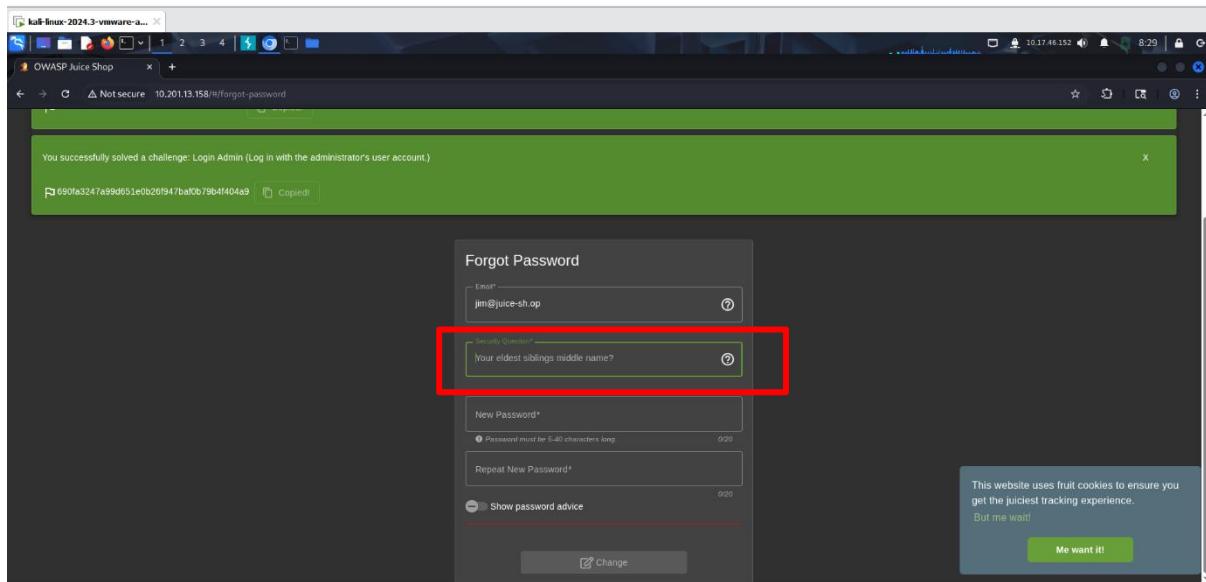
All Products

| | | | | | | | |
|--|-------------------------------|--|-----------------------|--|--------------------------------|--|---|
| | Apple Juice (1000ml) 1.99€ | | Apple Pomace 0.89€ | | Banana Juice (1000ml) 1.99€ | | Only 1 left Orange Juice This website uses fruit cookies to ensure you get the juiciest tracking experience. But me wait! Me want it! |
|--|-------------------------------|--|-----------------------|--|--------------------------------|--|---|

Why: Sniper attack in Burp tests one parameter at a time with a list of possible values. Here, it brute-forces the password until it finds the correct one.

Question 2: Reset Jim's password!

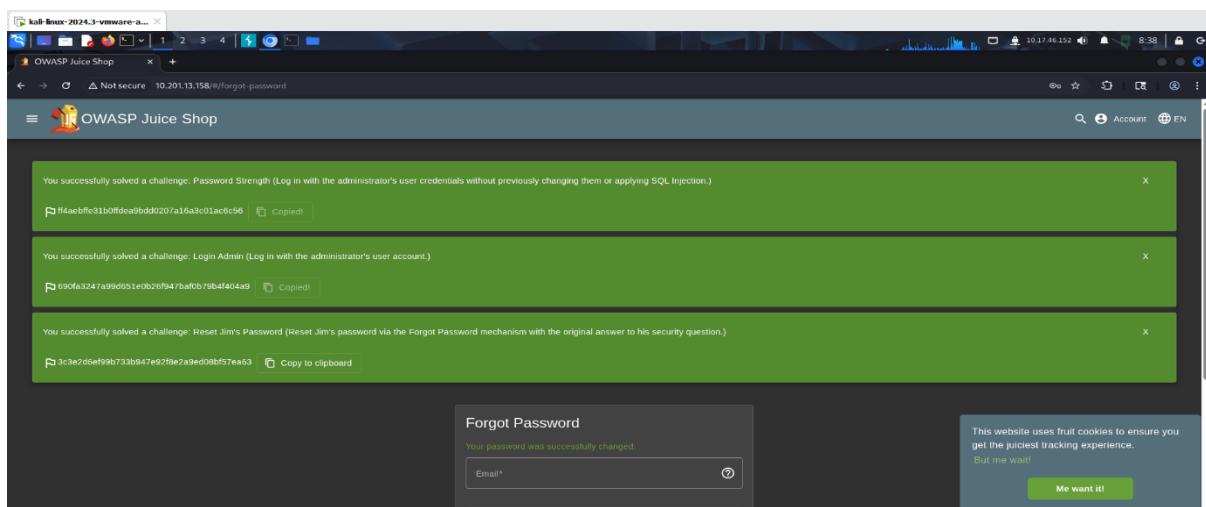
Step 1: Go to the Forget Password page and enter the email: jim@juice-sh.op. The security question appears: Your eldest sibling's middle name?



Step 2: Research online (e.g., Google) about Jim from Star Trek to find the eldest sibling's middle name — it is Samuel.

| | | |
|-------------------------------------|---|--|
| en.wikipedia.org/wiki/James_T._Kirk | <p>training aboard the <i>USS Republic</i>. He was then promoted to lieutenant [4] or grade and returned to Starfleet Academy as a student instructor.^[4] According to a friend, students could either "think or sink" in his class, and Kirk himself was "a stack of books with legs".^[6] Upon graduating in the top five percent, Kirk was promoted to lieutenant and served aboard the <i>USS Farragut</i>.^[4] While assigned to the <i>Farragut</i>, Kirk commanded his first planetary survey and survived a deadly attack by a bizarre cloud-like creature that killed a large portion of the <i>Farragut</i>'s crew,^[4] including his commanding officer, Captain Garrovick. Kirk blamed himself for years for hesitating to fire his assigned weapons upon seeing the threat until a later encounter with the creature showed that firing immediately with conventional weapons would have been useless.</p>  <p>Kirk became <i>Starfleet</i>'s youngest starship captain after receiving command of the <i>USS Enterprise</i> for a five-year mission,^[4] three years of which are depicted in the original <i>Star Trek</i> series (1966–1969).^[7] Kirk's most significant relationships in the television</p> | <p>Position</p> <p>Commander Captain Admiral Chief of Starfleet Operations <i>USS Enterprise</i>: Commanding officer <i>USS Enterprise-A</i>: Commanding officer United Federation of Planets Starfleet</p> <p>Affiliation</p> <p>George Kirk (father) Winona Kirk (mother) George Samuel Kirk (brother) Tiberius Kirk (grandfather) James (maternal grandfather) Aurelan Kirk (sister-in-law) Peter Kirk (nephew) 2 other nephews</p> <p>Family</p> <p>David Marcus</p> <p>Children</p> <p>Iowa, United States, Earth</p> <p>Origin</p> |
|-------------------------------------|---|--|

Step 3: Enter Samuel in the security question and set a new password, e.g., Admin@123.



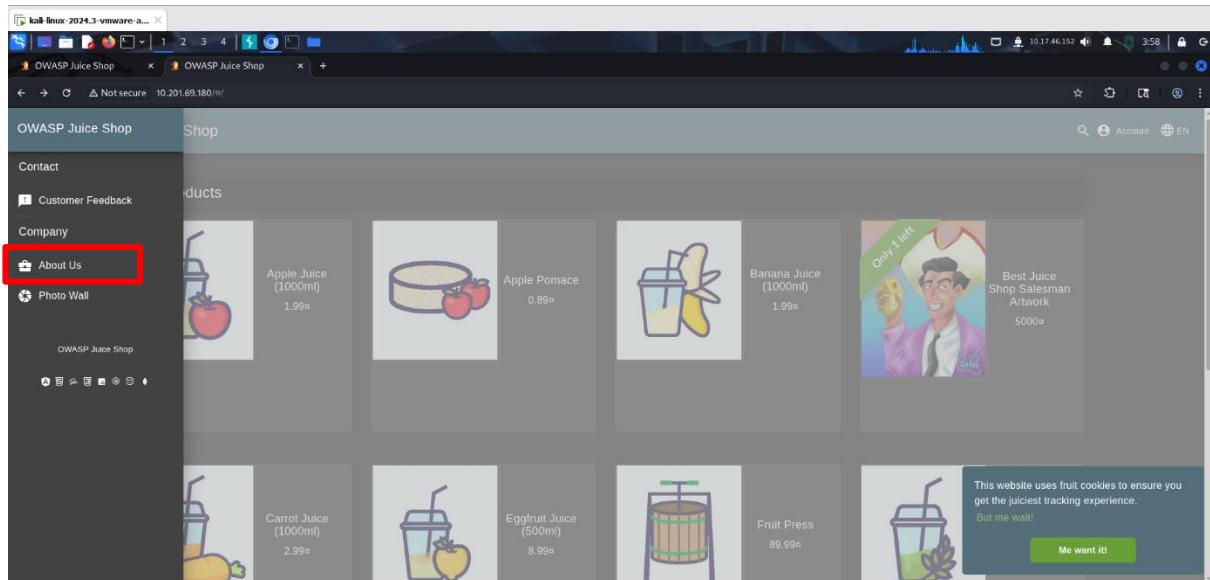
Step 4: Submit the form. The password is successfully changed, and the flag is revealed.

Why: Answering the security question correctly allows password reset and access to the account.

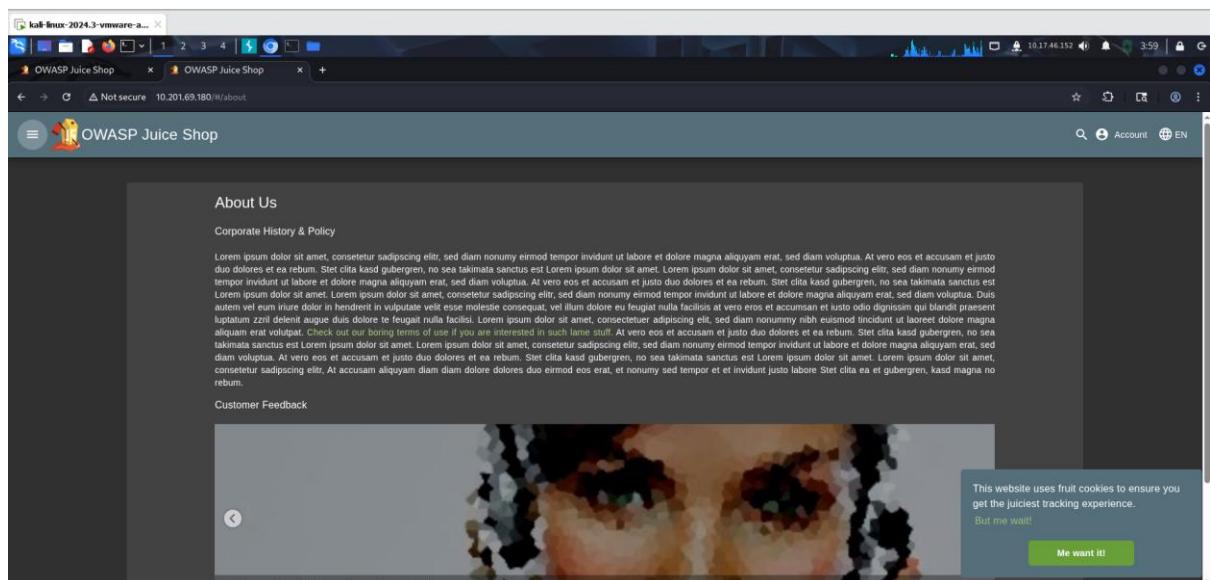
Task 5: AH! Don't look!

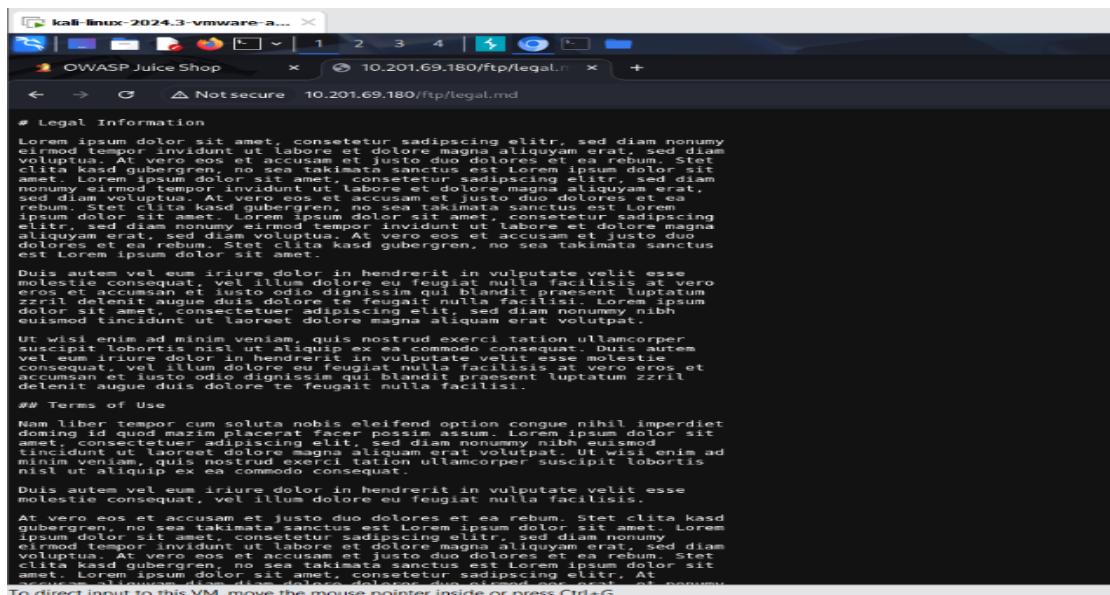
Question 1: Access the Confidential Document!

Step 1: Navigate to the about us page.



Step 2: There we find a link of <http://10.201.69.180/ftp/legal.md>. Then Navigating to that /ftp/ directory reveals that it is exposed to the public.





Legal Information

At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et justo odio dignissim quis blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et justo odio dignissim quis blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi.

Terms of Use

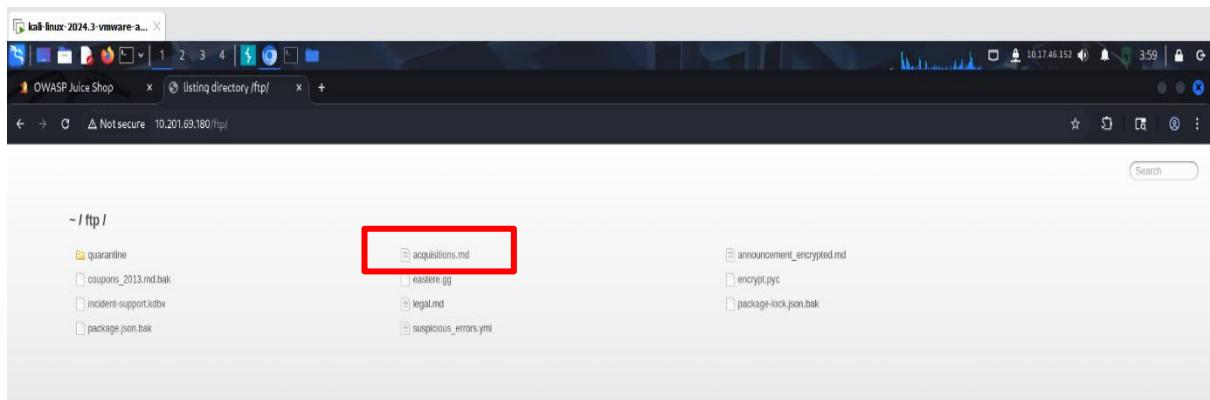
Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id egestas. Mauris dapibus, fermentum neque, posuere etiam vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et justo odio dignissim quis blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi.

Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis.

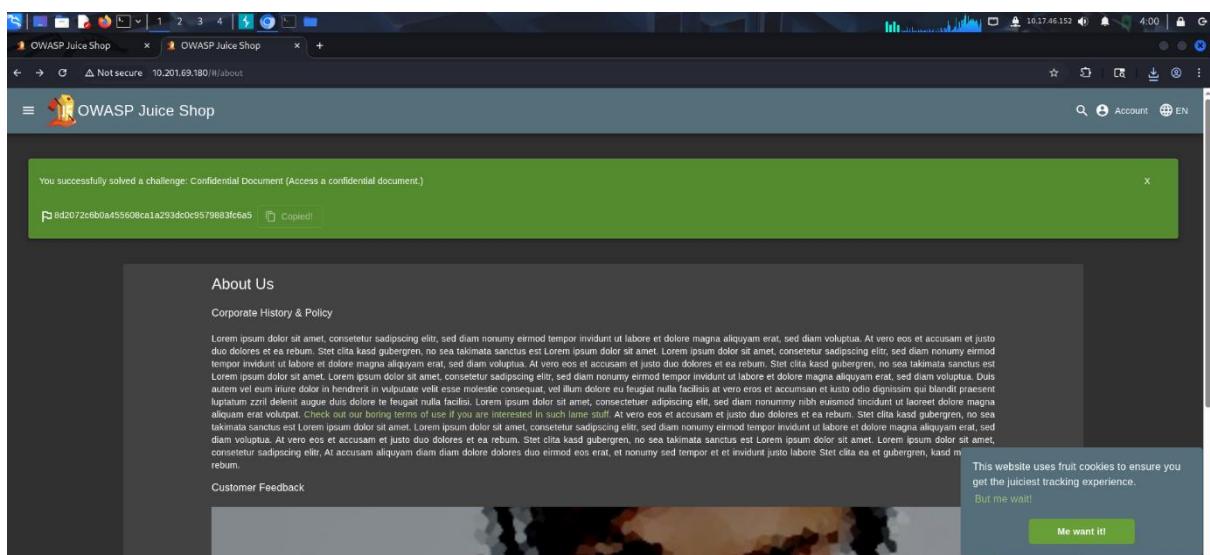
At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

Step 3: the ftp page will show this files where we download the **acquisitions.md** and save it.

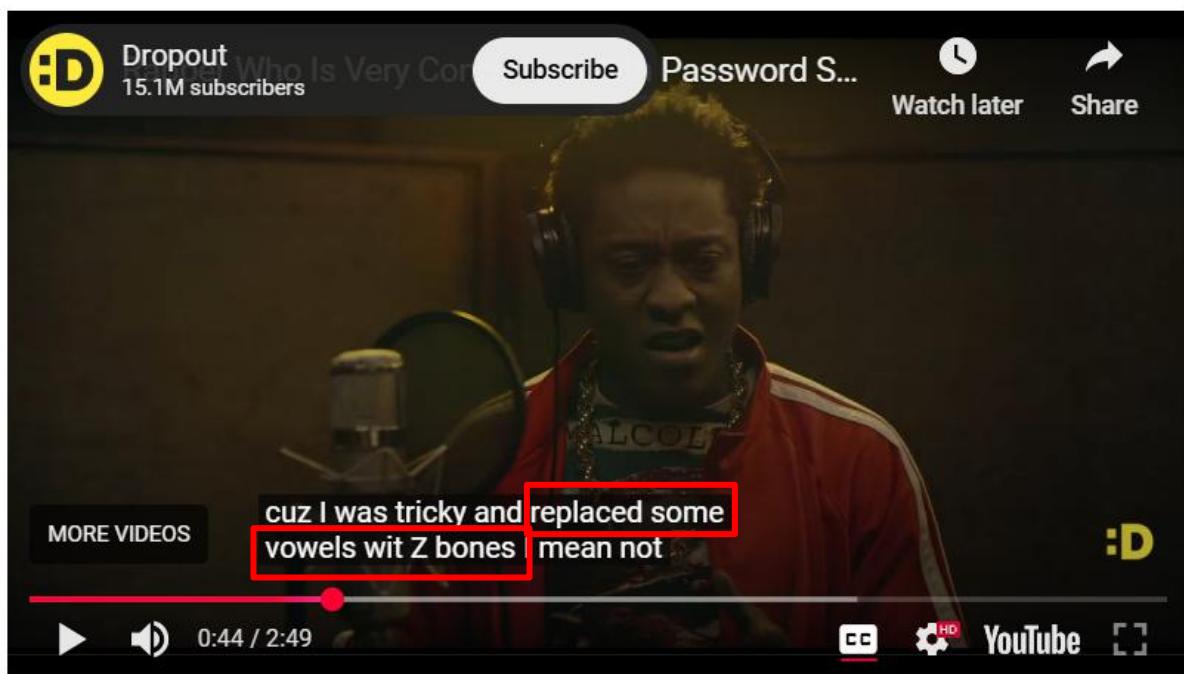
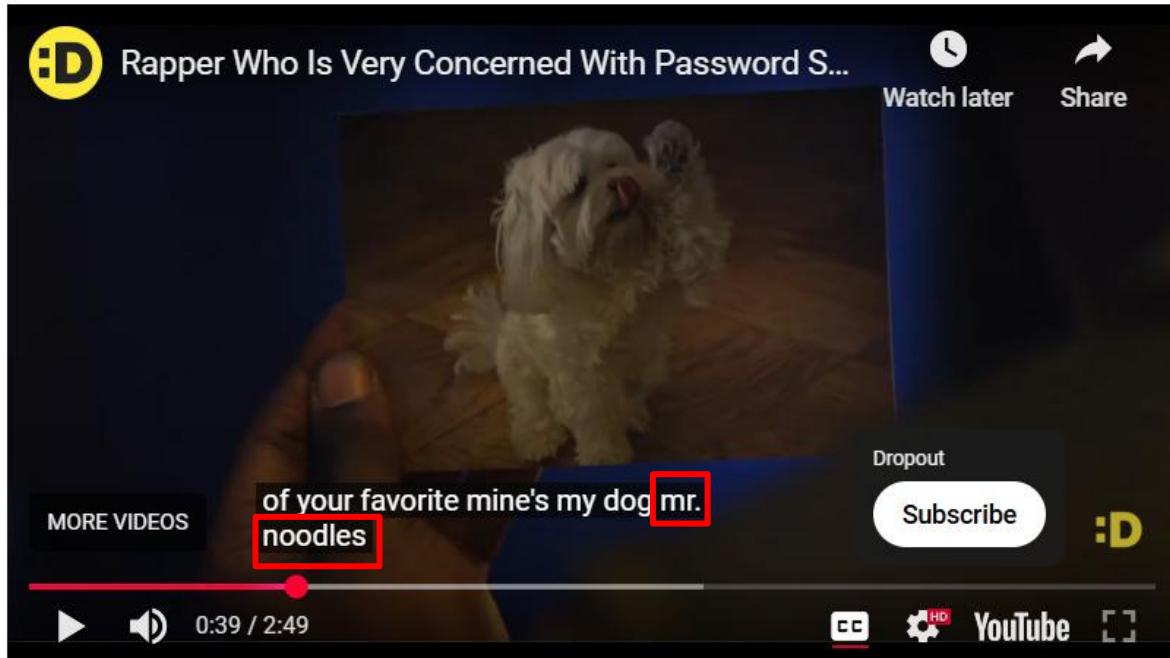


Step 4: After saving the file we get the flag.



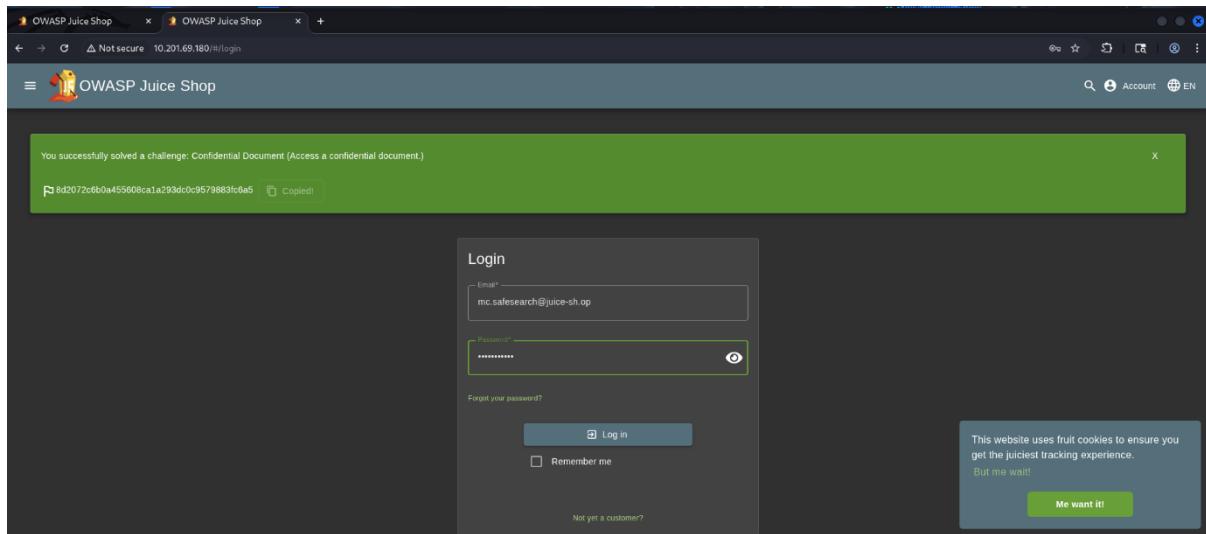
Question 2: Log into MC SafeSearch's account!

Step 1: In this task we have to login to the mc safesearch's account. For password he given a hint on the song. Which is mentioned in below image

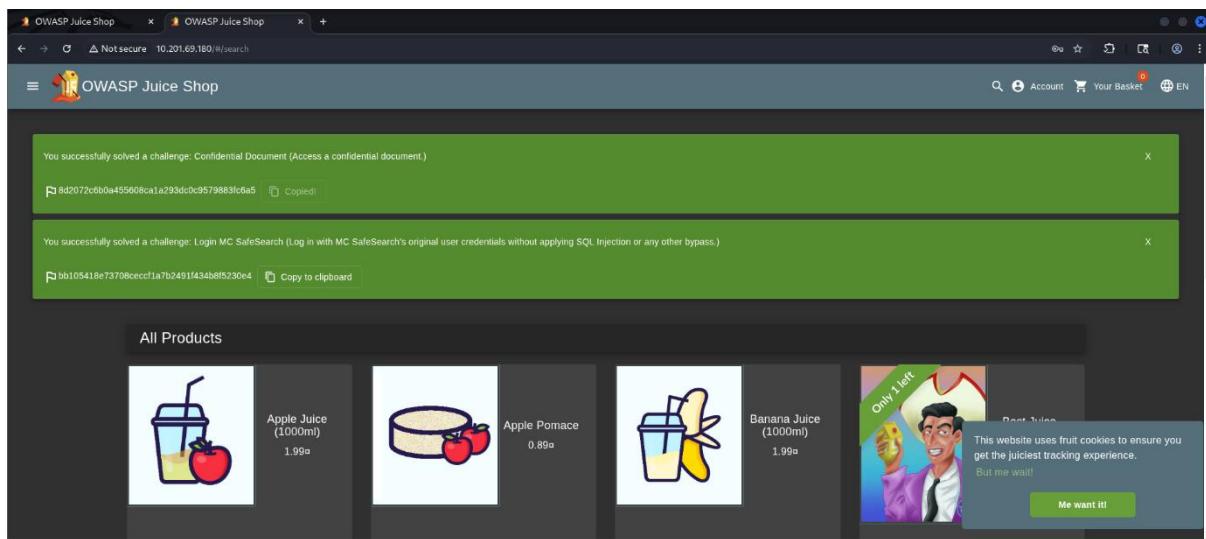


In this he mentioned that his dog name is the password and he replaced vowels with zeros. Which make password as "Mr. N00dles".

Step 2: now we use mc.safesearch@juice-sh.op as email id and password as "Mr. N00dles". And try to login the account.

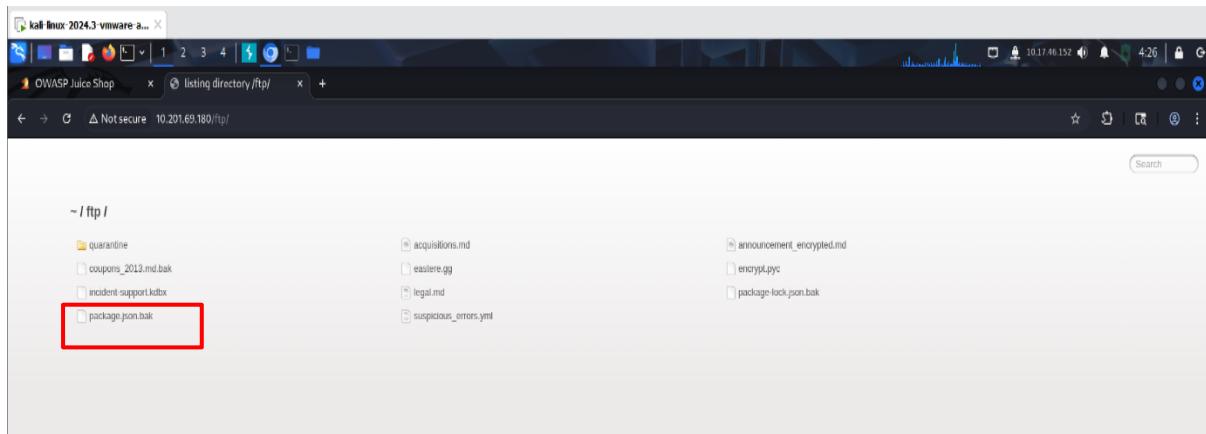


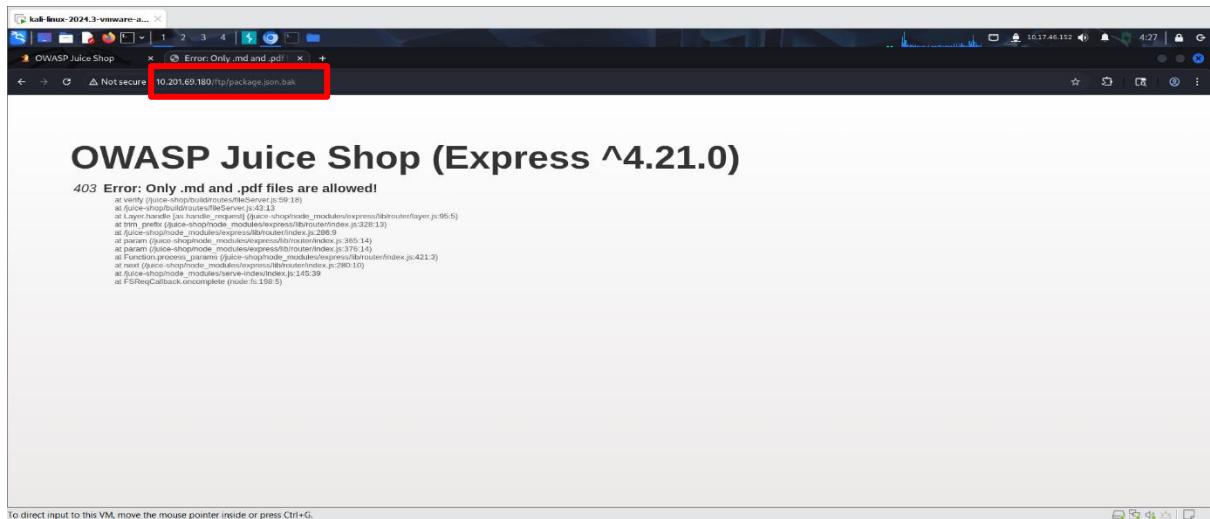
Step 3: Using these credentials we successfully login to the account and get the flag.



Question 3: Download the Backup file!

Step 1: We will now go back to the <http://10.201.69.180/ftp/> folder and try to download **package.json.bak**. But it seems we are met with a 403 which says that only .md and .pdf files can be downloaded.

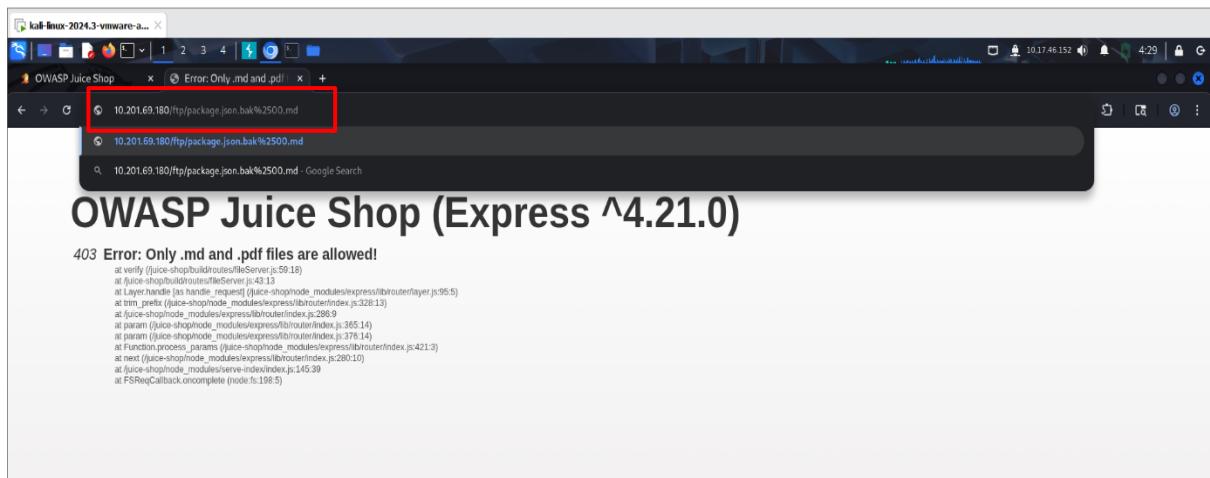




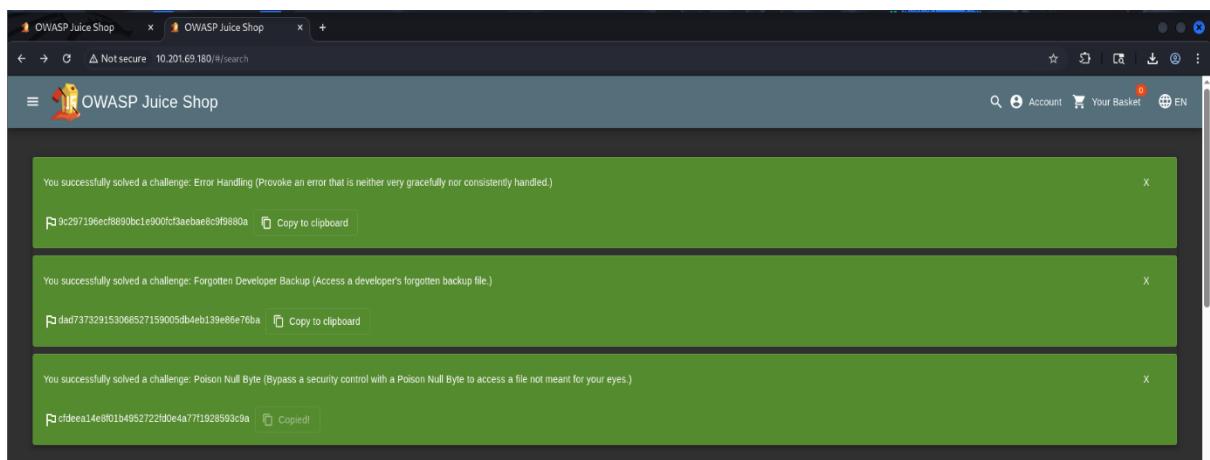
Step 2: To get around this, we will use a character bypass called "**Poison Null Byte**". A Poison Null Byte looks like this: **%00**.

Note: as we can download it using the url, we will need to encode this into a url encoded format.

The Poison Null Byte will now look like this: **%2500**. Adding this and then a **.md** to the end will bypass the 403 error!



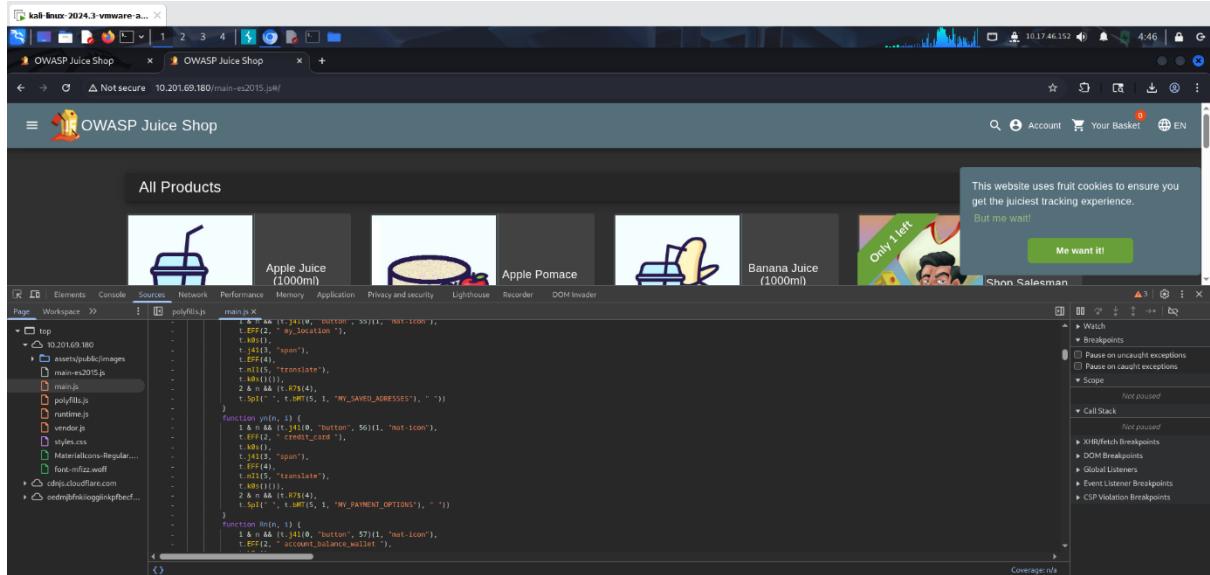
Step 3: after that we can get the flag.



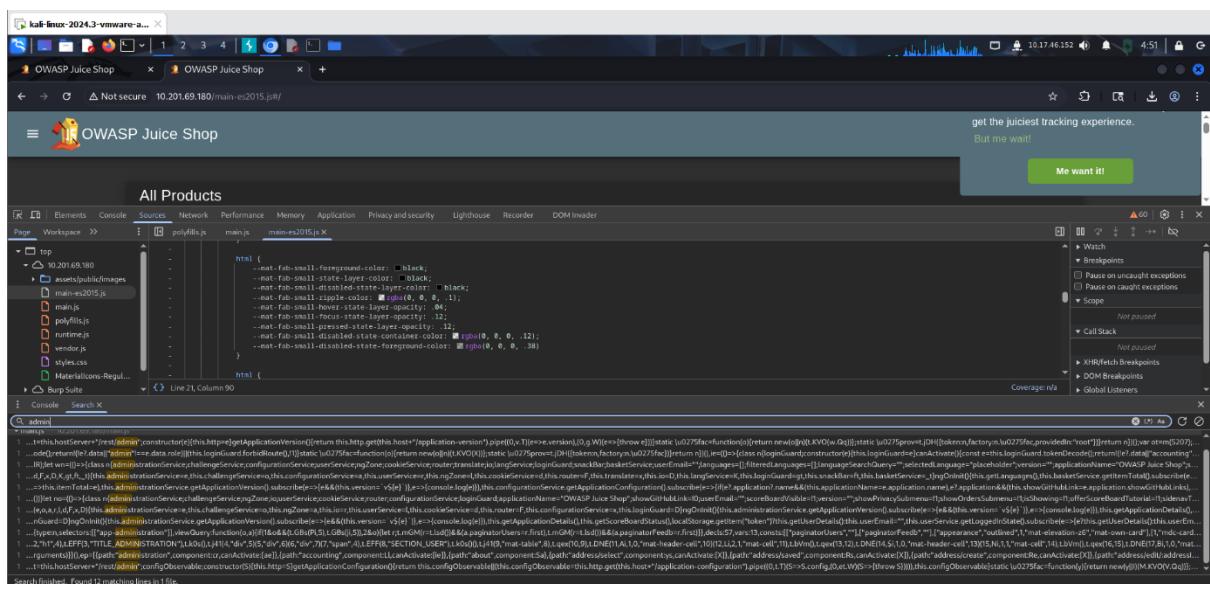
Task 6: Who's flying this thing?

Question 1: Access the administration page!

Step 1: Open Chrome → Developer Tools → Sources.



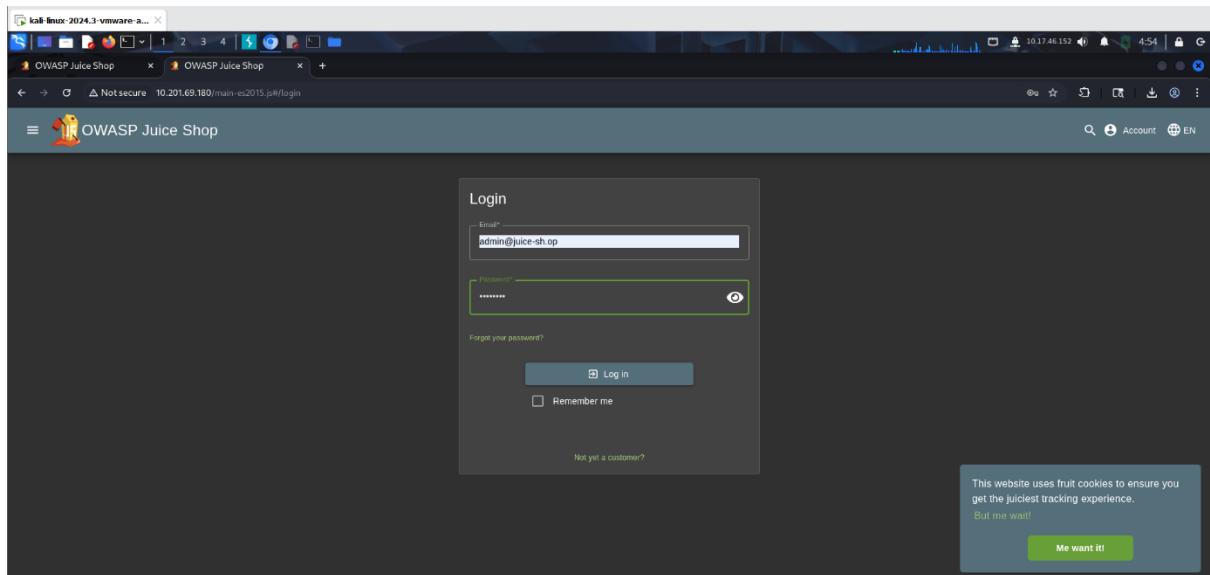
Step 2: Refresh the target page so the debugger/sources panel populates with the site's files. In the file list, find and open main-es2015.js.



Step 3: Click the { } button at the bottom to format the file into readable code. Use the search box (Ctrl/Cmd+F) and search for the term admin.

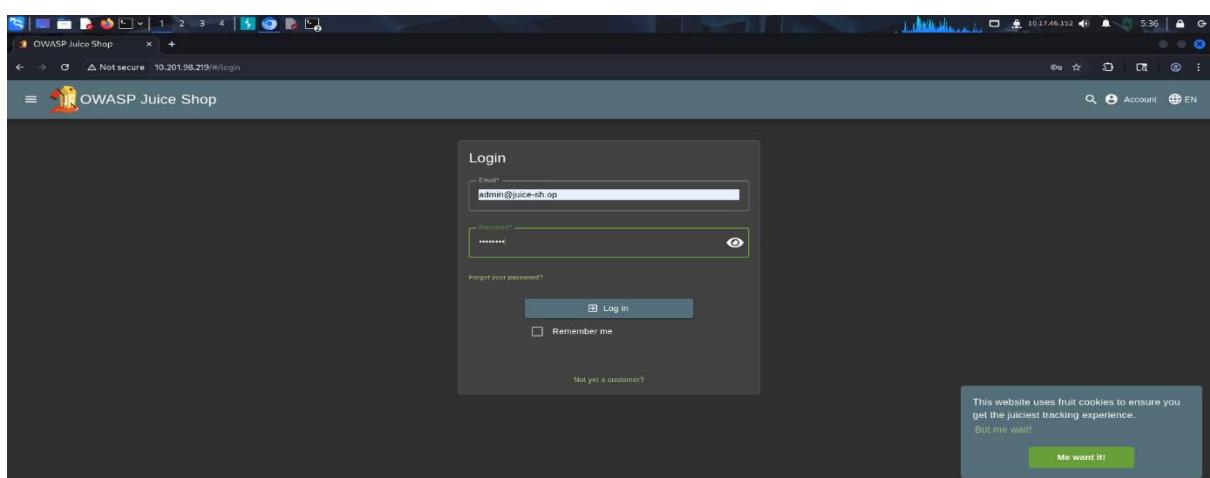
Step 4: Inspect the matches and locate the entry that reads path: administration (we can also see related about/path entries nearby). Note the implied route /#/administration from that path value.

Step 5: After login. Try visiting <http://10.201.98.219/#/administration> in browser.



Question 2: View another user's shopping basket!

Step 1: Log in as **admin** and click **Your Basket** in the app.



Step 2: Burp Suite is running and Proxy → Intercept is On.

Step 3: Click through/refresh the basket page and **forward** intercepted requests until we see a request like:

GET /rest/basket/1 HTTP/1.1

(this is the request that reads the basket for user id 1).

Request details for the highlighted request:

```
GET /rest/basket/1 HTTP/1.1
Host: localhost:8080
...
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGFdODM0JzcdwNjZKzL1w1ZGF0YSlseyJpZC1GM9wi0DNxcmhbhMs011c1UwPpbCI0fRkbwUoQp1aewLNx0m011w1cGFzc3vcmQjO1iwMkYd1ydiYmQH11MDuNnhwNjLZ3E4YjUMCt1s1nWb0J0Jz01ph1stnWb1hVA2Vra2V1j011w1bGp2dExvZ2LuSx41xMcA4Ny40Nq4xNT1L1CwcmmaxL1Kw1h2J01Jhcs3NlJhPhmchLb01j21tWd1cy91cOvxFp2L2Rf1hB821p1b5dwe1L1C1b3PwL2V1j010yMDE4p0O.CyktbFnsze7P8nZ2ZBV1j7AyEB049h640-MK5dftED09y%2B9tz962bb78p4d4u_1N1tba1tG01d0rSwLACAHjBj1Utrr-kdy7Wbpsol0EaKFuQkD-20AZXQj+xct31ed6yra2q3pLzU-prHkz0-df7tPSAbH4
Accept-Language: en-US,en;q=0.9
Accept: application/json, text/plain, */*
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/139.0.0.0 Safari/537.36
Referer: http://10.201.98.219/
Accept-Encoding: gzip, deflate, br
Cookie: language_token=
...
10 If-None-Match: W/"51e-ycVrb1j0cR0ifgqntwknruwA"
11 Connection: keep-alive
12
13
14
15
16
17
18
19
```

Step 3: Right-click the intercepted request → **Change Request** (or view it in the Intercept tab) and edit the path: replace /rest/basket/1 with /rest/basket/2. Forward the modified request to the server.

Step 4: Inspect the server response (in Burp's Response tab or in browser) — it should return the basket contents for **UserID 2**.

Step 5: we get the flag after that.

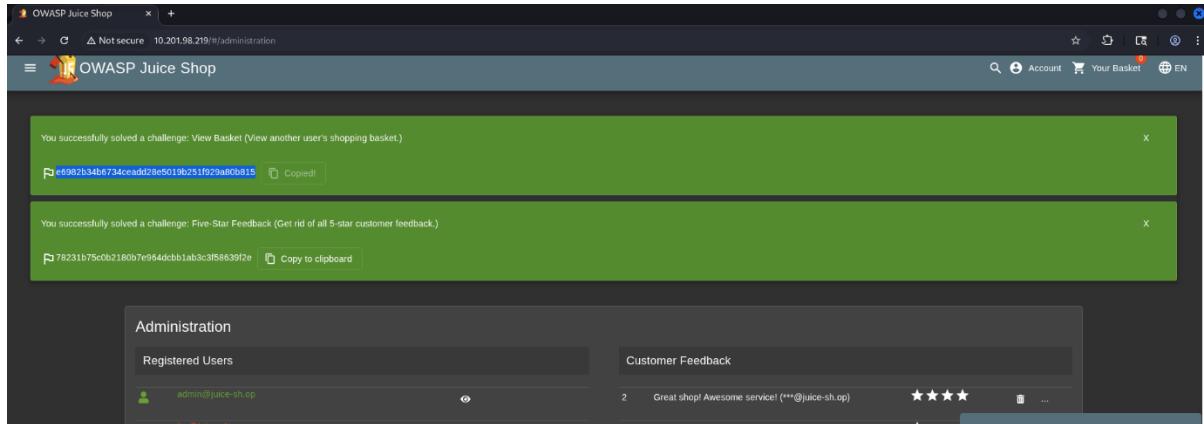
Why: the app exposes a numeric resource ID in the request path; by changing that ID we're requesting a different user's basket.

Question 3: Remove all 5-star reviews!

Step 1: Open browser and go to <http://10.201.98.219/#/administration>. Ensure we are logged in as admin.

Step 2: On the **Administration** page, find the **Reviews** section or reviews table/list. Locate the review that has **5 stars**.

Step 3: Click the **bin / trash** icon next to that 5-star review. After this we find the flag once page get reloaded.



Why: The bin icon issues a delete request for that review record; removing it can trigger the app to reveal state/flags in this lab.

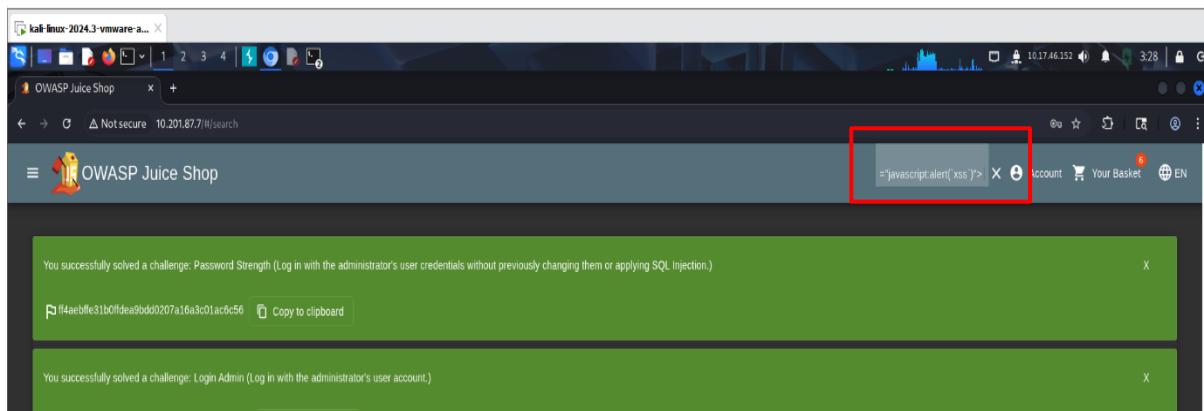
Task 7: Where did that come from?

Question 1: Perform a DOM XSS!

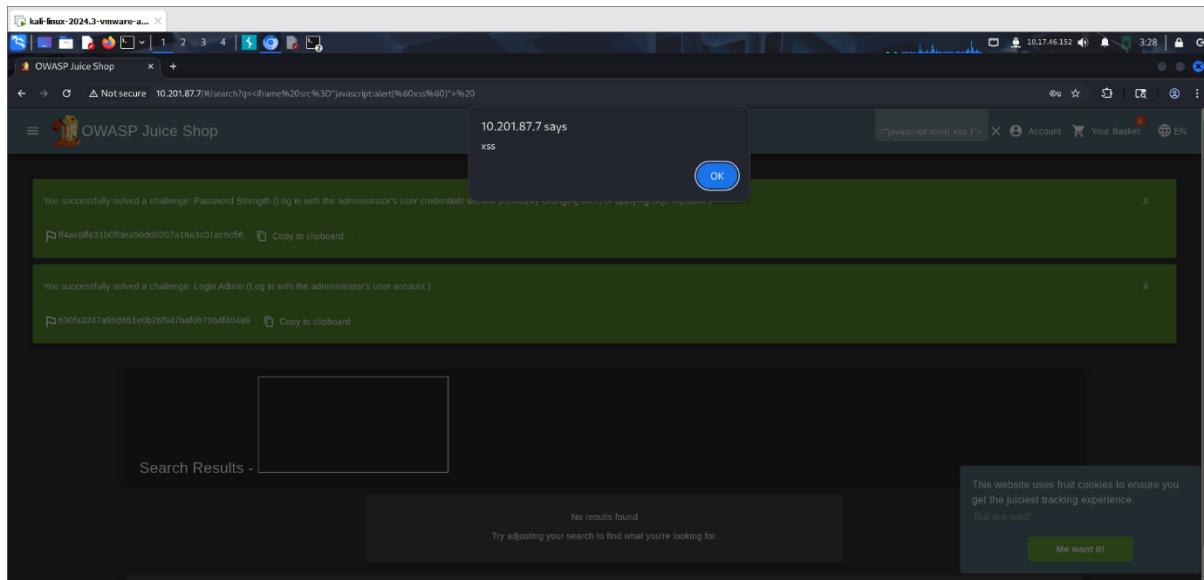
Step 1: Open the web application and locate the **search bar**.

Step 2: Input the following payload into the search bar:

```
<iframe src="javascript:alert('xss')">
```



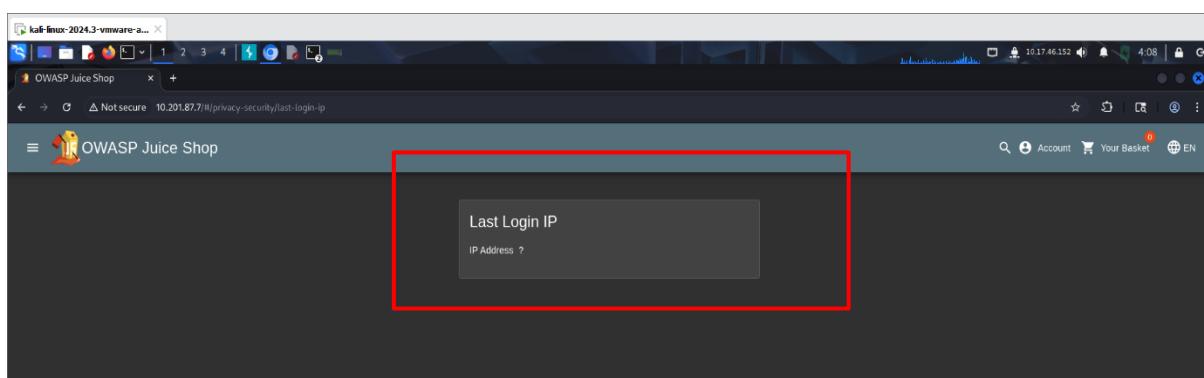
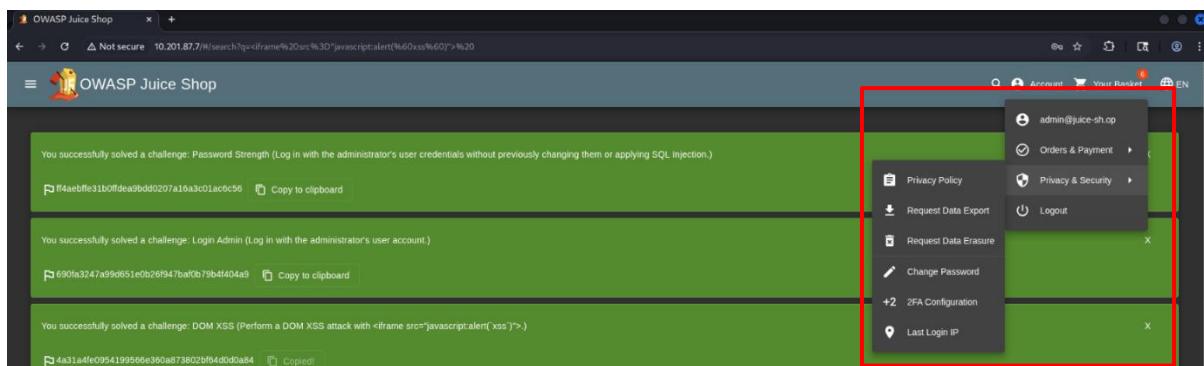
Step 3: Press **Enter** or submit the search. An alert box (xss) should pop up.



Question 2: Perform a persistent XSS!

Step 1: **Login** to the admin account.

Step 2: Navigate to the **Last Login IP** page. We should see the last IP address (e.g., 0.0.0.0 or 10.x.x.x).



Step 3: **Logout** to trigger a new login record — this is needed so the server logs a new IP. Make sure **Burp Suite Intercept** is **On** to capture the logout request.

Step 4: In Burp, go to the **Headers** tab of the intercepted request and add a new header:

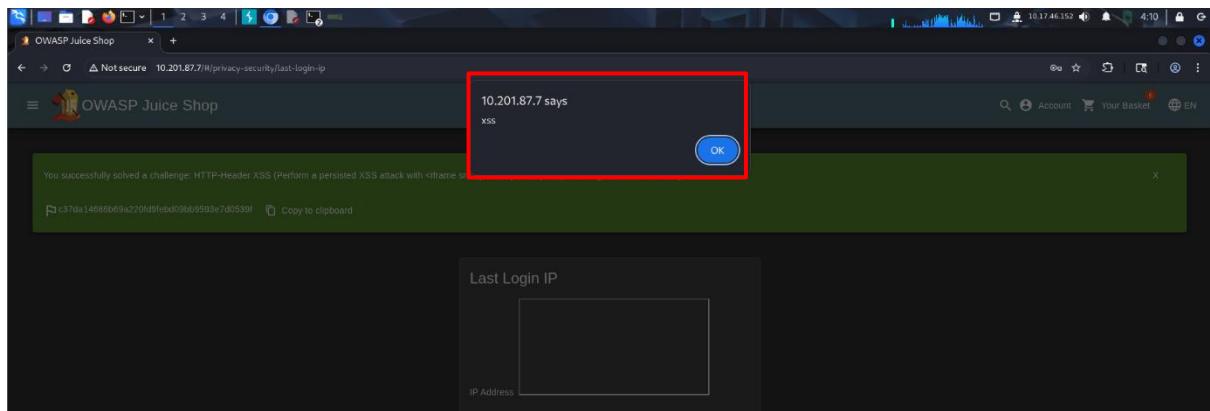
True-Client-IP: <iframe src="javascript:alert(`xss`)">

The screenshot shows the Burp Suite interface with the 'Headers' tab selected. A red box highlights the 'Request' pane where a new header 'True-Client-IP' has been added. The value is set to '<iframe src="javascript:alert(`xss`)">'. The 'Inspector' pane on the right shows the modified request with the new header included.

This screenshot shows the Burp Suite interface after the modification. The 'Request' pane now includes the 'True-Client-IP' header with the specified value. The 'Inspector' pane shows the full modified request. The status code is listed as 1, indicating the request is ready to be sent.

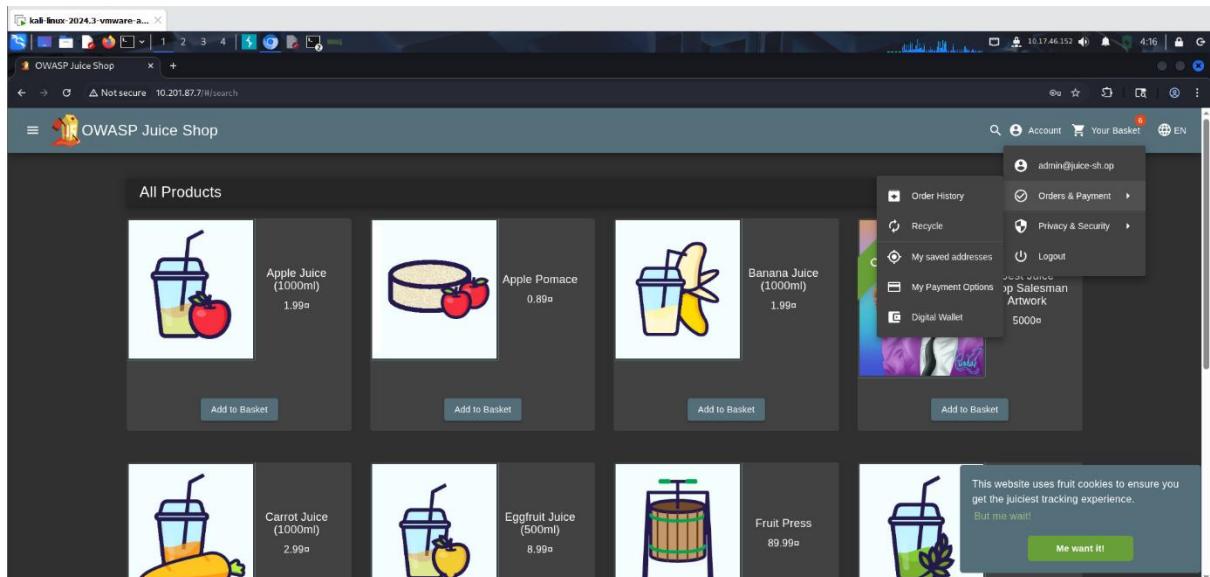
Step 5: Forward the modified request to the server. And Log back into the **admin** account and navigate again to the **Last Login IP** page. The XSS alert (xss) should trigger.

The screenshot shows the OWASP Juice Shop login page. The 'Email' field contains 'admin@juice-sh.op'. The 'Password' field is filled with '.....'. Below the form, a message box displays: 'This website uses fruit cookies to ensure you get the juiciest tracking experience. But me wait!'. At the bottom right of the page, there is a green button labeled 'Me want it!'.

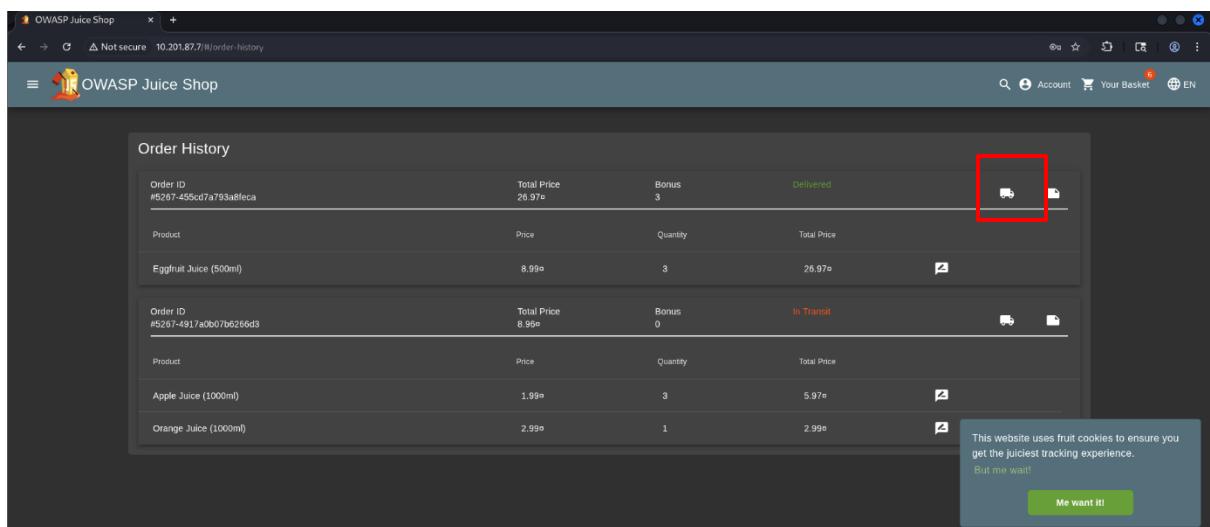


Question 3: Perform a reflected XSS!

Step 1: Login to the admin account. Navigate to the Order History page.

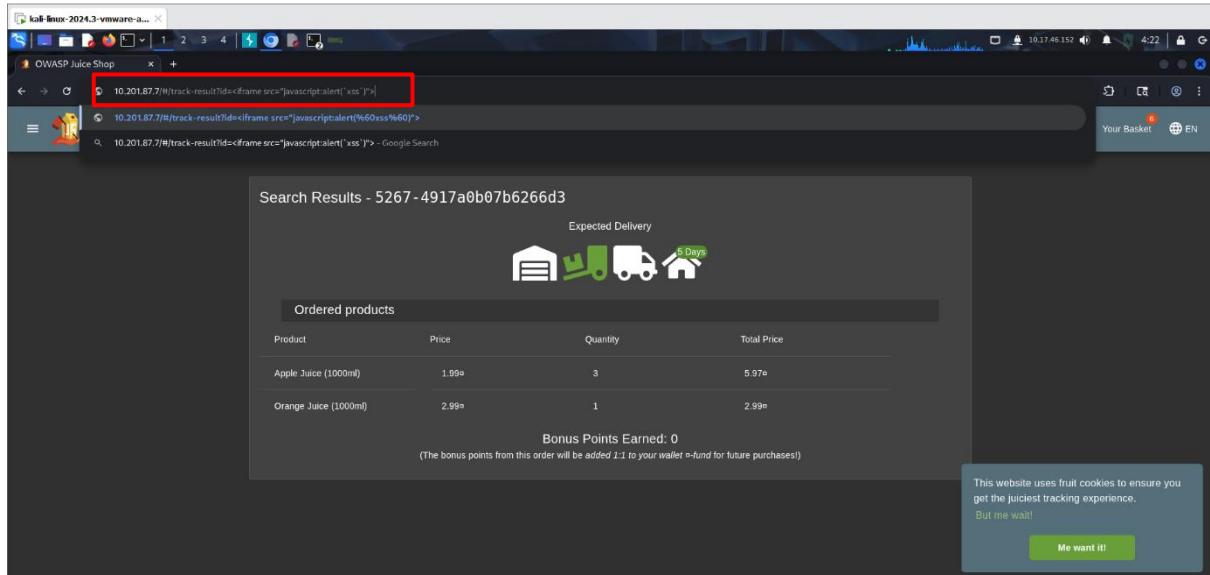


Step 2: Locate an order and click the Truck icon to open the tracking result page.



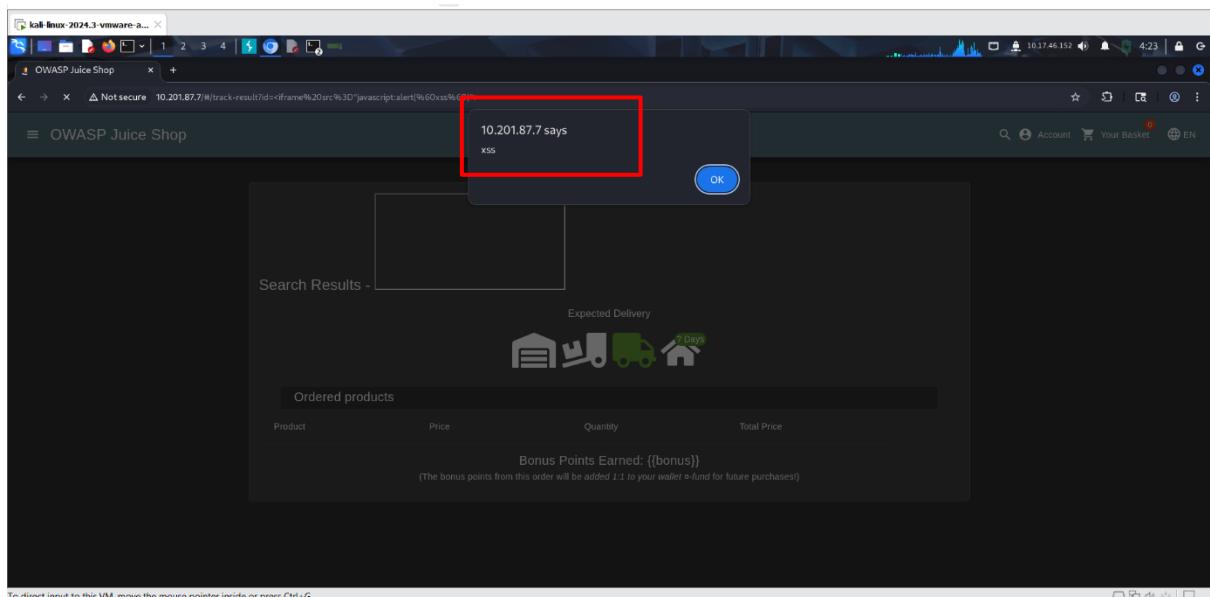
Step 3: Replace the tracking ID in the URL or input field with the XSS payload:

```
<iframe src="javascript:alert(`xss`)">
```

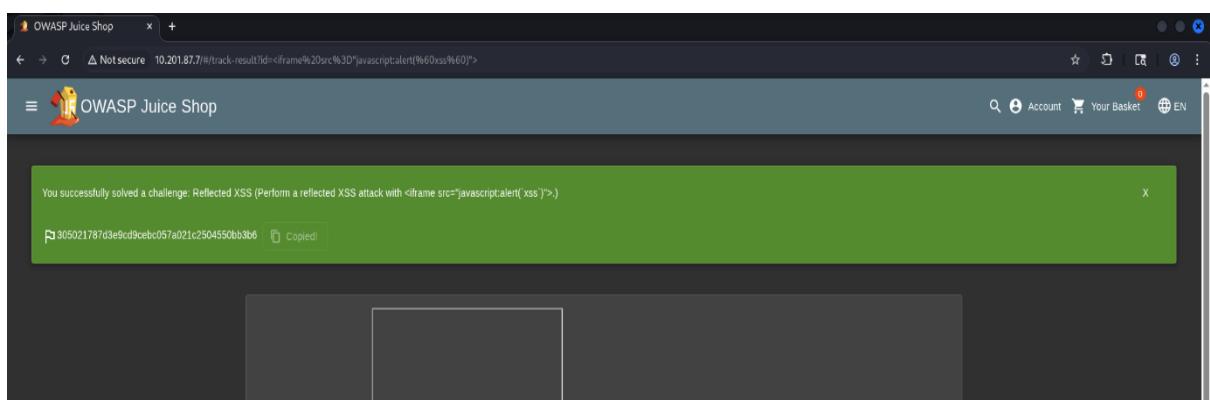


The screenshot shows a Linux desktop environment with a web browser window open to the OWASP Juice Shop. The URL in the address bar is highlighted with a red box and contains the XSS payload: `10.201.87.7/#/track-result?id=<iframe src='javascript:alert(`xss`)'>`. The page displays a search result for product ID 5267-4917a0b07b6266d3, showing two items: Apple Juice (1000ml) and Orange Juice (1000ml). Below the table, a message says "Bonus Points Earned: 0". On the right side, there's a cookie consent banner with the text: "This website uses fruit cookies to ensure you get the juiciest tracking experience. But me wait!" and a "Me want it!" button.

Step 4: Submit the URL/input. And refresh the page — an **alert box** with xss should appear, confirming the XSS vulnerability. Then we get the flag.



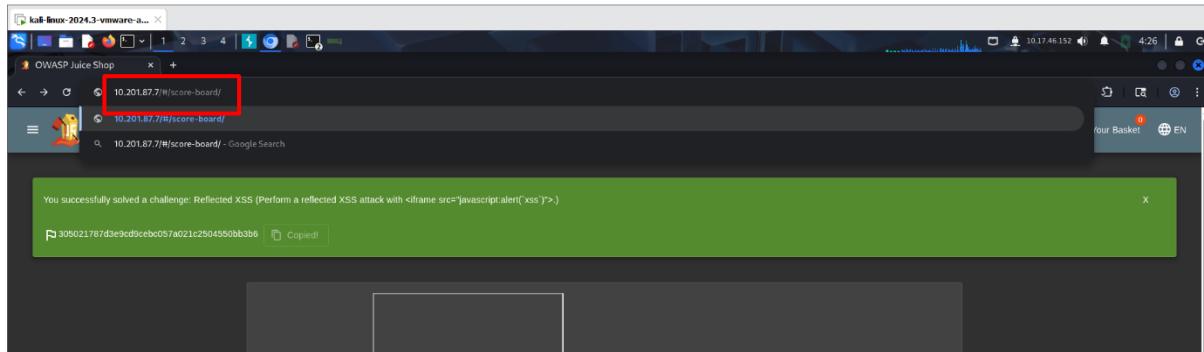
The screenshot shows the same browser window after the XSS payload was submitted. A modal alert box is displayed with the text "10.201.87.7 says" followed by "xss". An "OK" button is visible in the bottom right corner of the alert box. The background page content remains the same as in the previous screenshot.



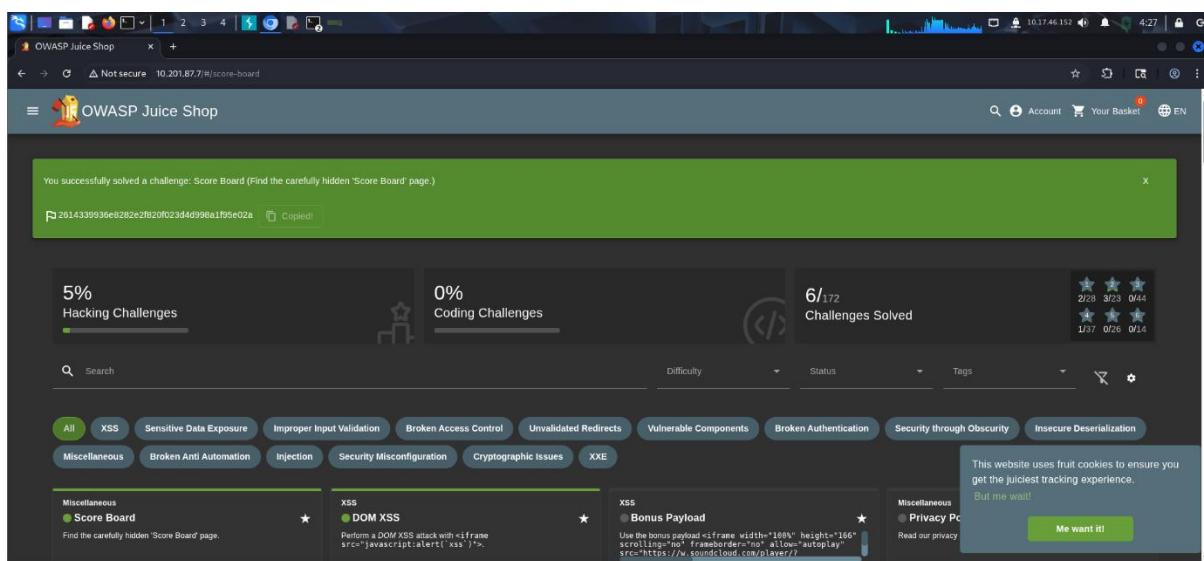
The screenshot shows the browser after the XSS attack was successful. A green notification bar at the top of the page states: "You successfully solved a challenge: Reflected XSS (Perform a reflected XSS attack with `<iframe src='javascript:alert('xss')'`>.)". Below this, a message says "Copied!" next to the string `30502178763e9cd9cebc057a021c2504550b63d6`.

Task 8: Exploration!

Step 1: add this to the url `/#/score-board/` and press enter.



Step 2: It will show the Score Board. And we get the flag.



4. Conclusion

This assessment of OWASP Juice Shop demonstrated several high-risk vulnerabilities, including:

- SQL Injection (authentication bypass)
- Broken Authentication (brute-force, weak recovery)
- Insecure Direct Object Reference (IDOR)
- Cross-Site Scripting (XSS)
- Sensitive Data Exposure via Directory Traversal

The exercise highlights the importance of secure coding practices, especially in line with the OWASP Top 10. By applying input validation, strong authentication, least-privilege access, and secure coding frameworks, most of these flaws can be mitigated effectively.