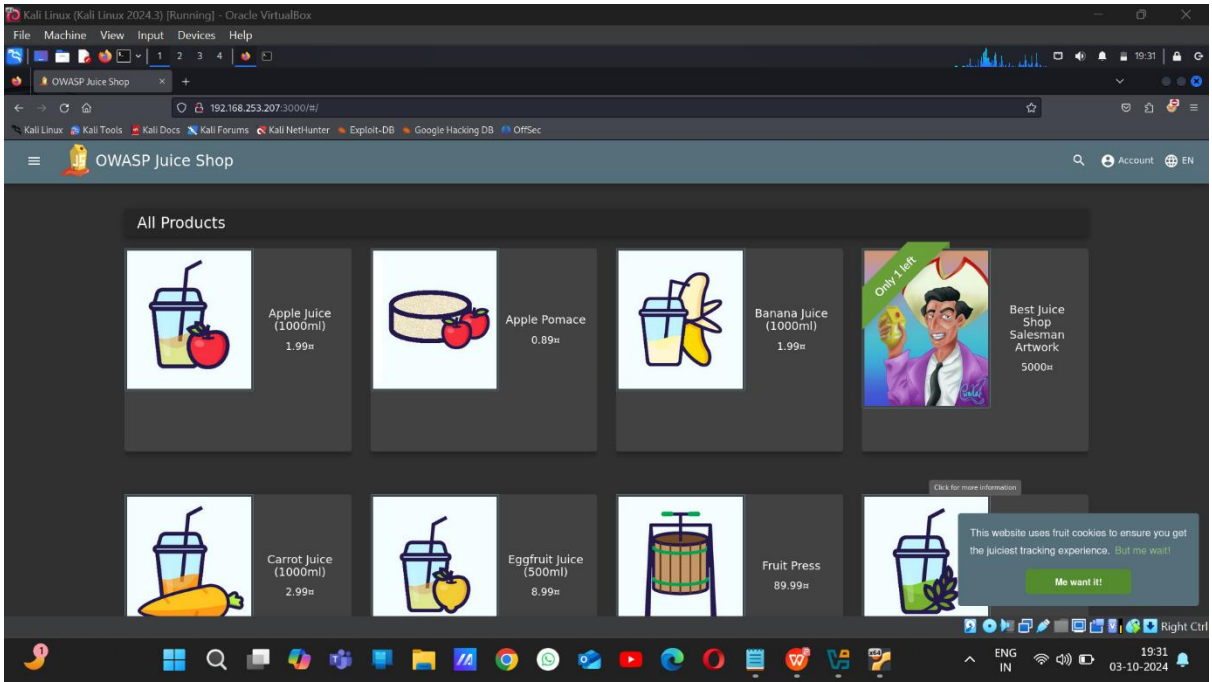
Capstone Project:

Ethically Hacking an E-Commerce Website

1) E-Commerce Website n can be accessed in a browser by entering the Machine IP followed by the port number 3000 after the machine boots up or running a Nmap scan on the IP will show details about the target and reveal the open ports in which the site could be accessed on

- ❖ Nmap involves gathering critical information about a target system or network to identify potential vulnerabilities.
- ❖ To Know the verisons used the command : nmap -sV 192.168.253.180
- ❖ -sV: Enables service and version detection

2. E-Commerce Website n can be accessed in a browser by entering the Machine IP followed



by the port number 3000 .

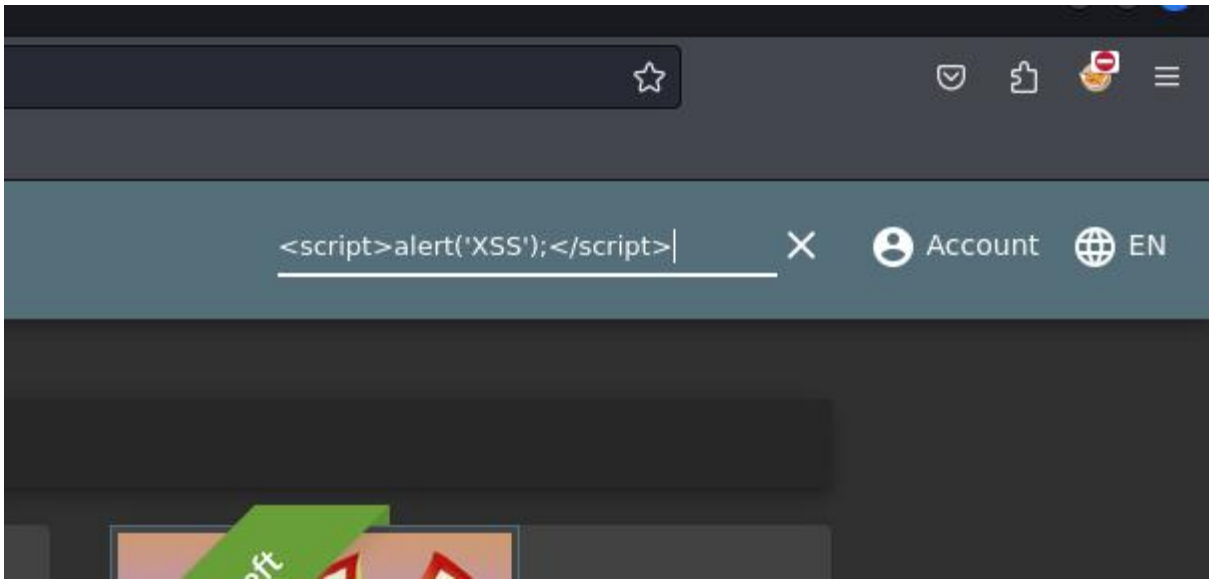
Vulnerabilities

DOM XSS

DOM XSS stands for Document Object Model-based Cross-site Scripting. A DOM-based XSS attack is possible if the web application writes data to the Document Object Model without proper sanitization. The attacker can manipulate this data to include XSS content on the web page, for example, malicious JavaScript code.

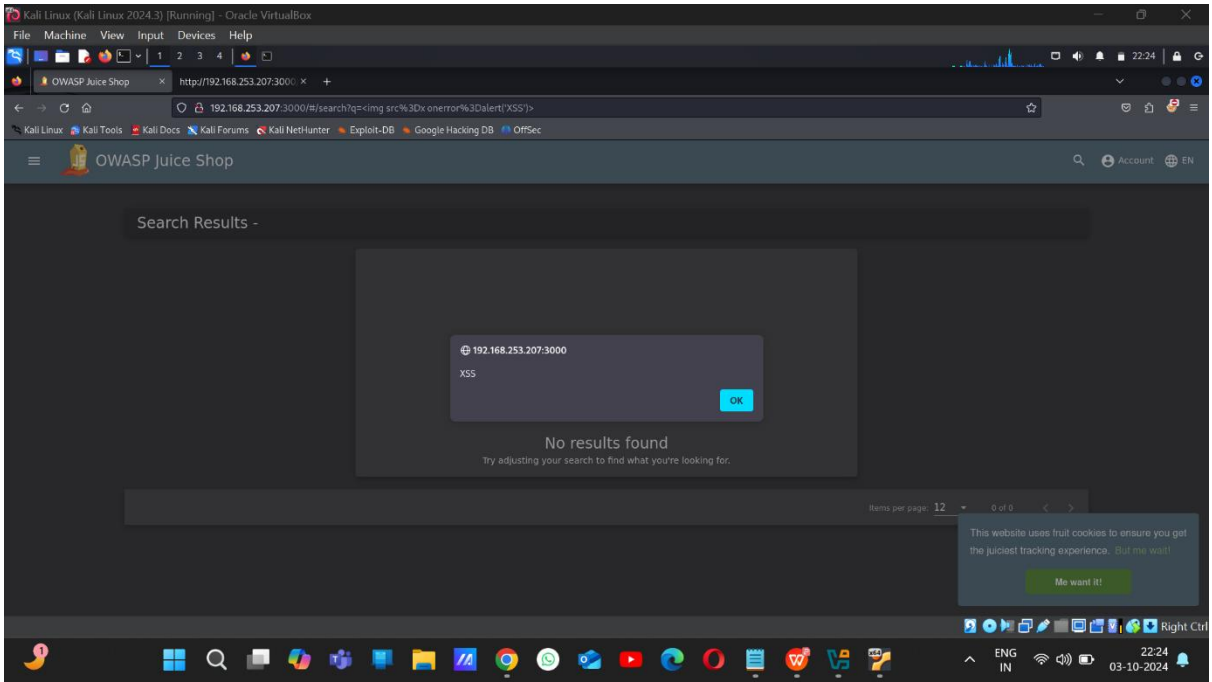
By typing XSS payload in ther search bar and pressing enter , the vulneribility could be found.

The payload, <img src=x onerror=alert(1)>, attempts to execute JavaScript by using the onerror attribute of an image element. When the image fails to load (src=x), it triggers the



onerror event, which executes the alert(1) JavaScript code.

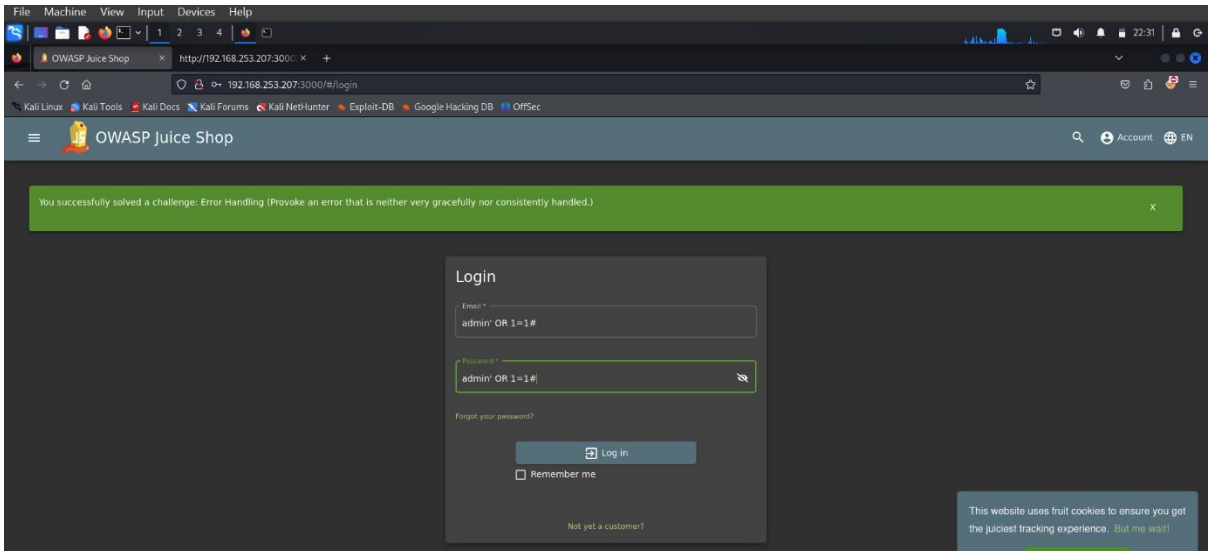
The alert dialog box pops up it shows that javascript code executed and the site is vulnerable to XSS attack.

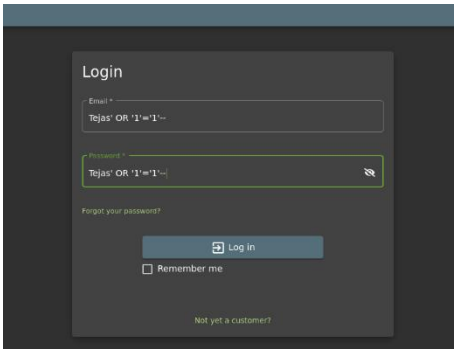


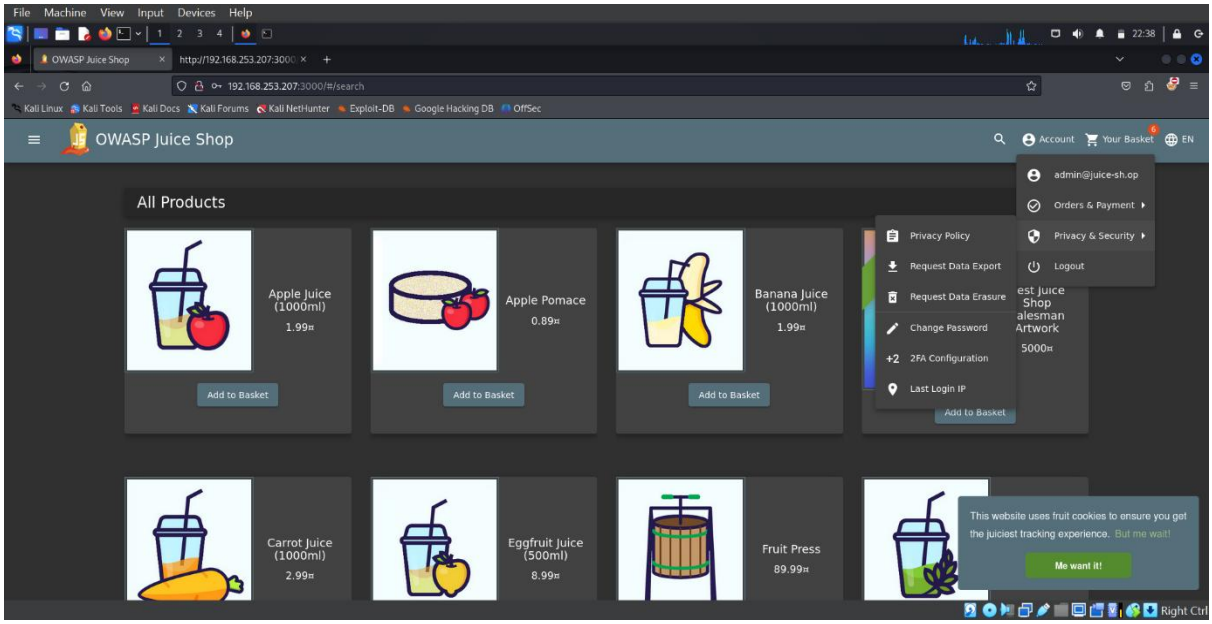
Login as Admin using SQL Injection

SQL injection, or SQLi, is a code injection technique that allows an attacker to manipulate data in a SQL database.Allow attackers to spoof identity, tamper with existing data, cause repudiation issues,etc.

Enter SQL injection payload in both the email and password and press enter to Login as





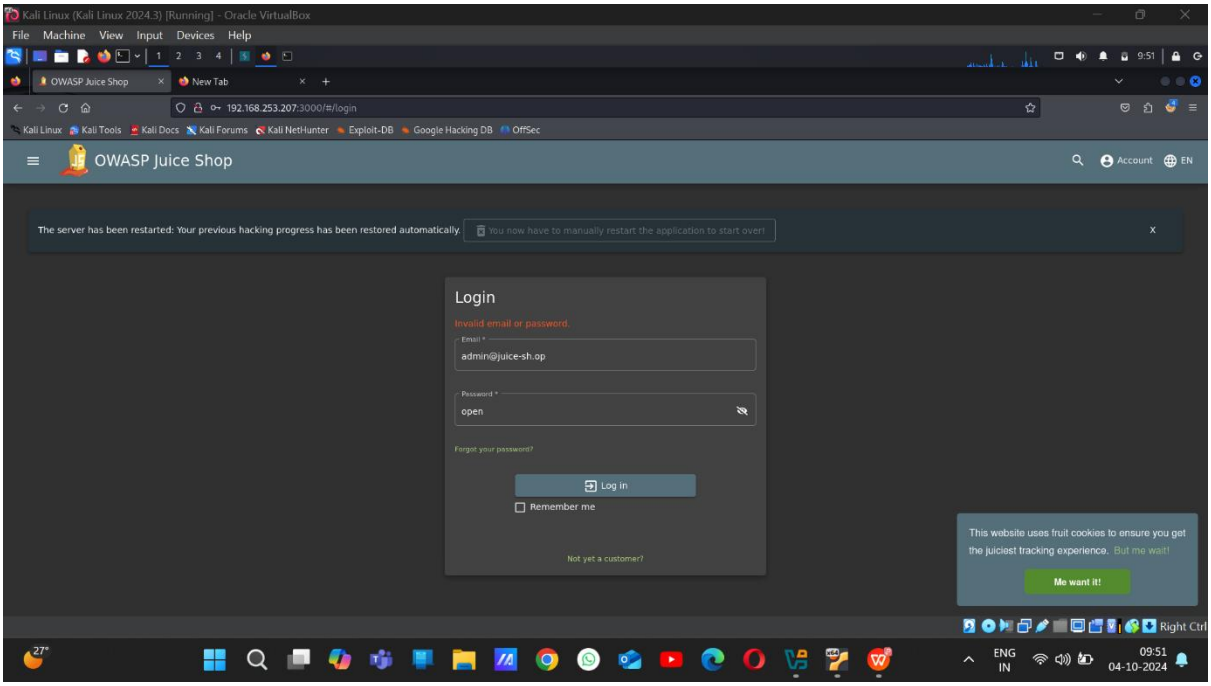


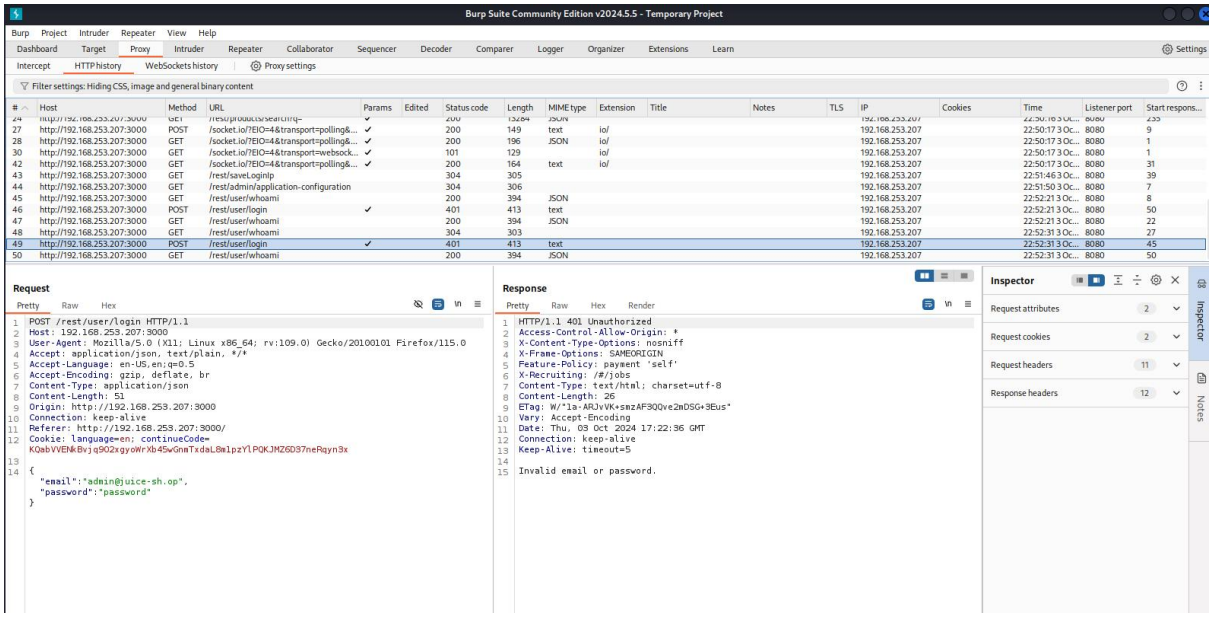
Admin.

Login as Admin without SQL injection (Using Brute-forcing Attack)

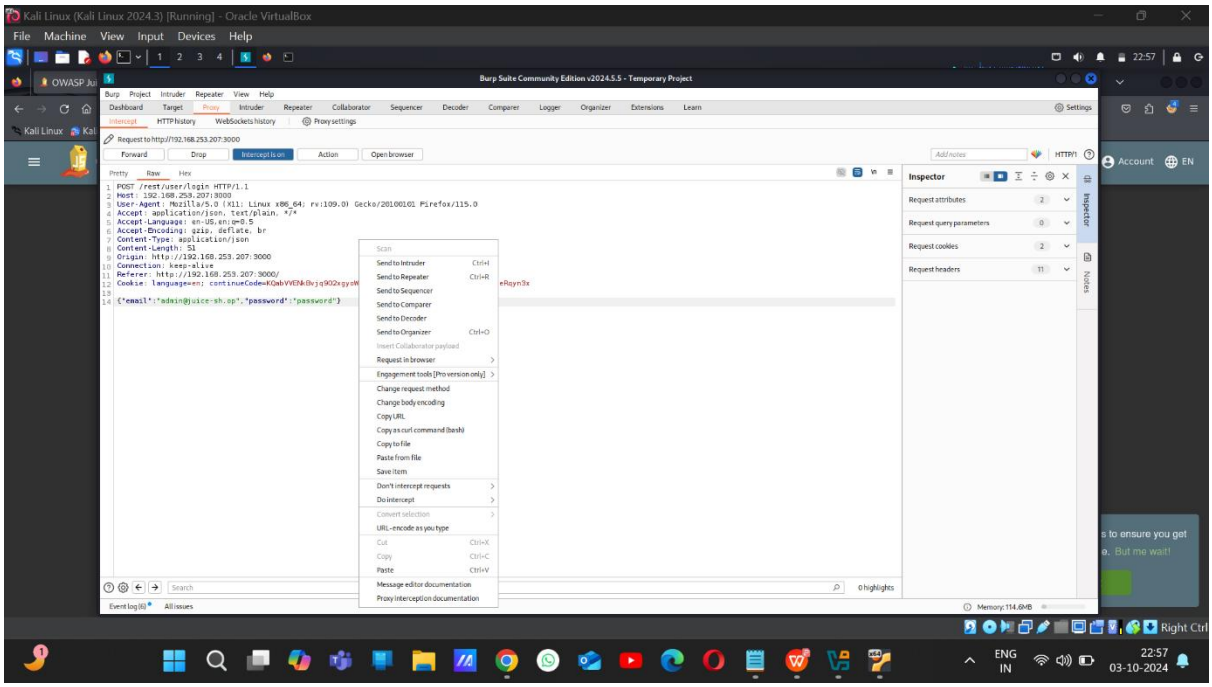
Brute Force Attack: If the login system lacks rate-limiting or CAPTCHA verification, attackers can repeatedly send login attempts, trying different combinations of usernames and passwords. A strong dictionary of common passwords, paired with a known admin username or email, can be used to automate attempts to gain access. Leads to unauthorised access.

1. Enter the email address of admin, random password and attempt to login capture the login request in burpsuite.

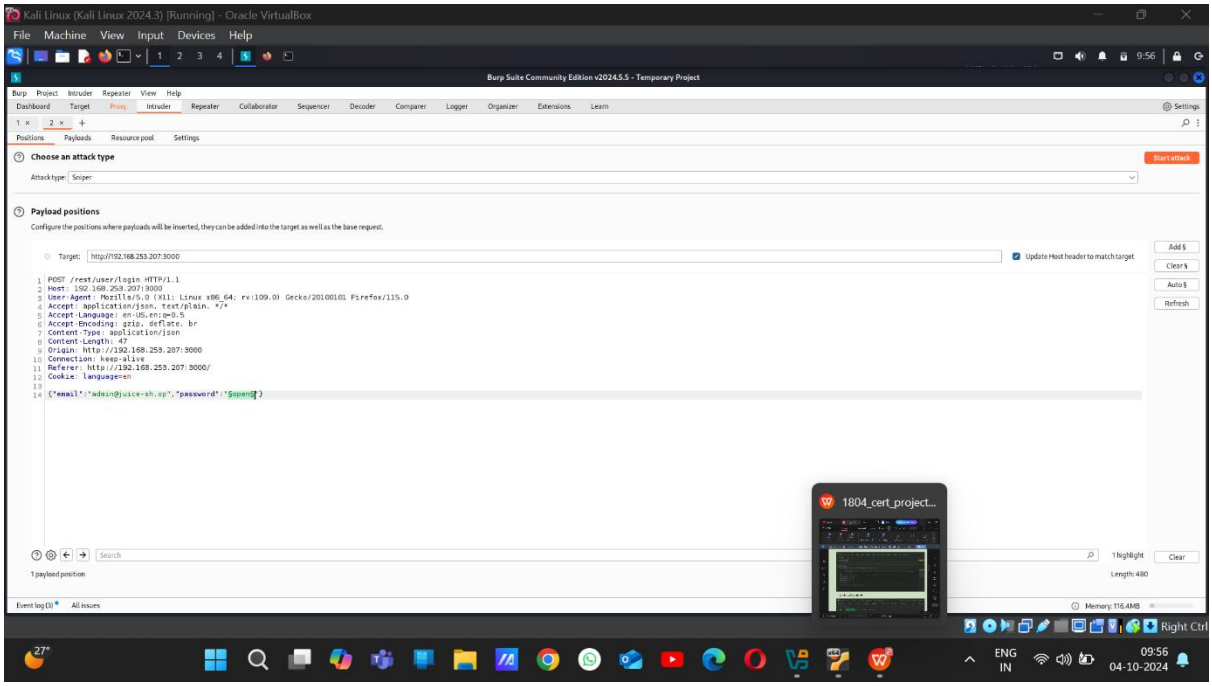




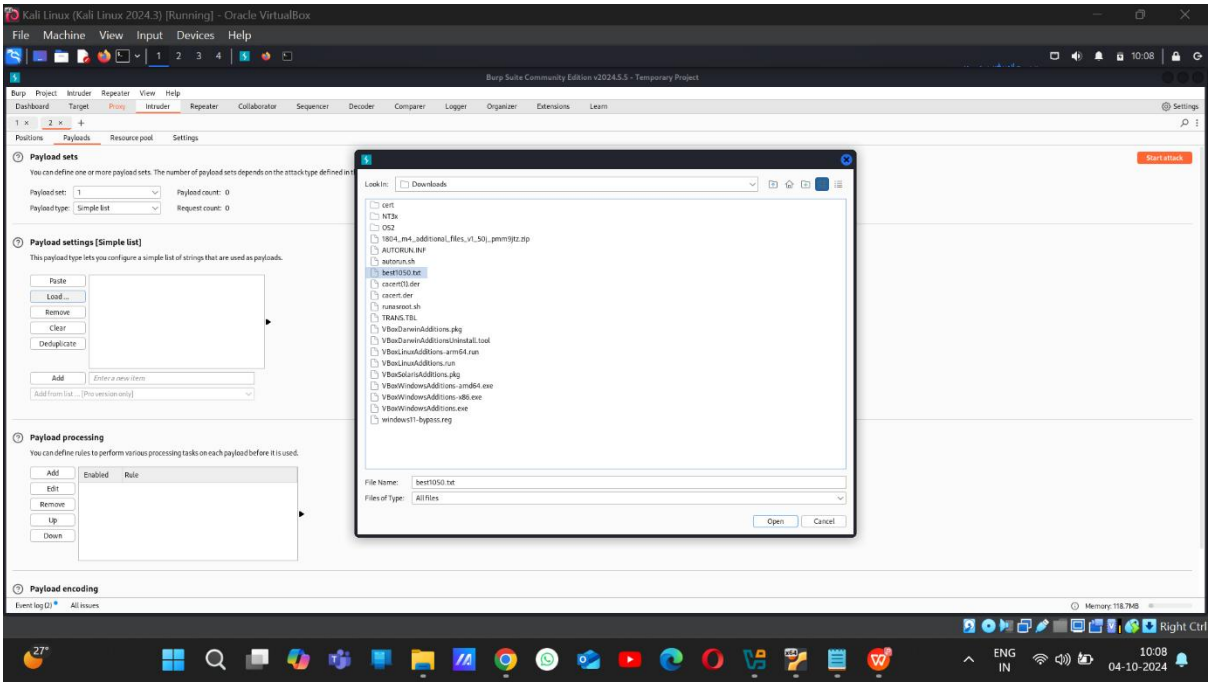
2. Send Request to burp intruder.

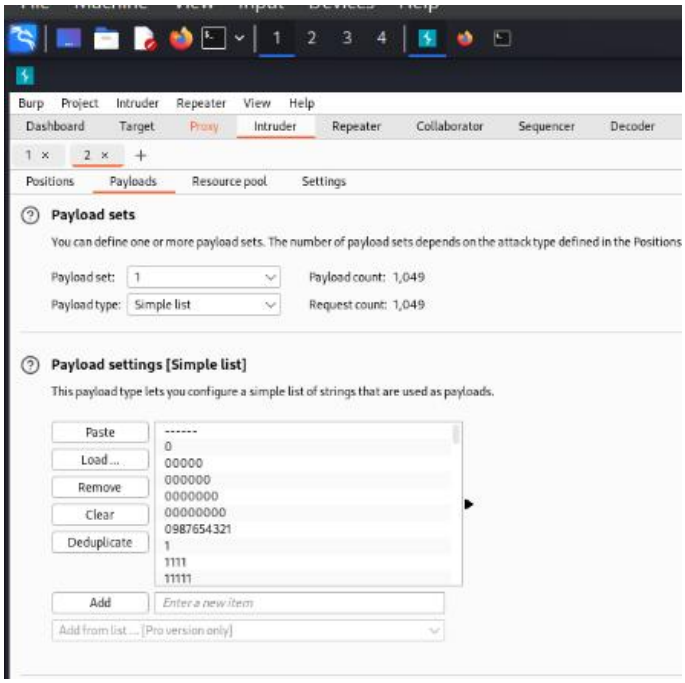


3. Add the payload injecting position to only the password value in the captured request as shown below.



4. Download a common password word list from google. Click the payload tab and load it in the “payload option” section from your system and click on “start attack” button.

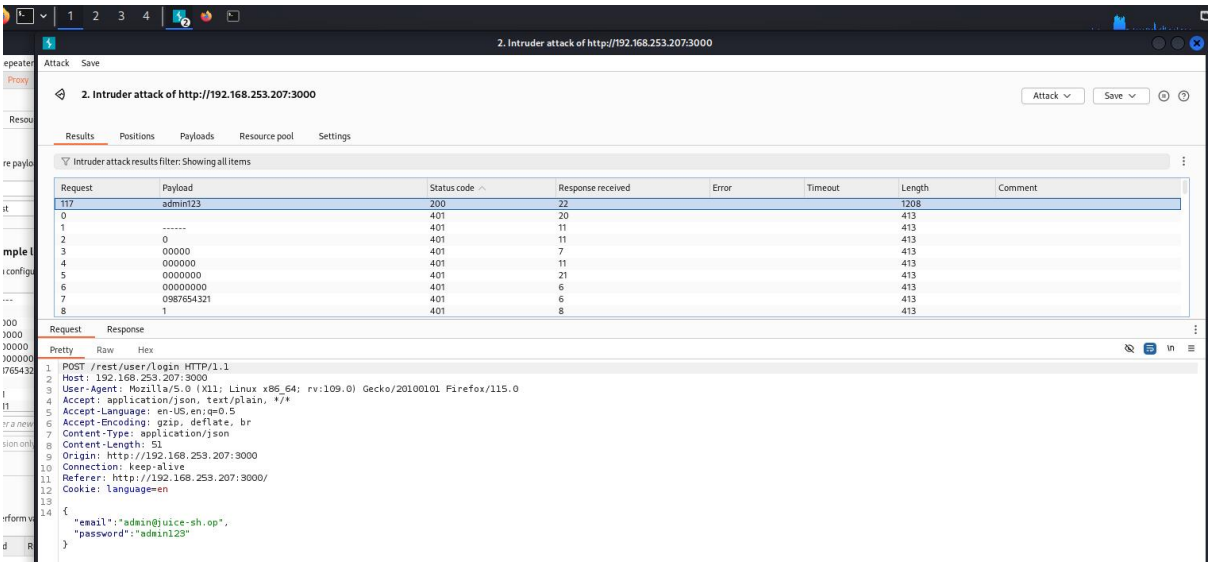


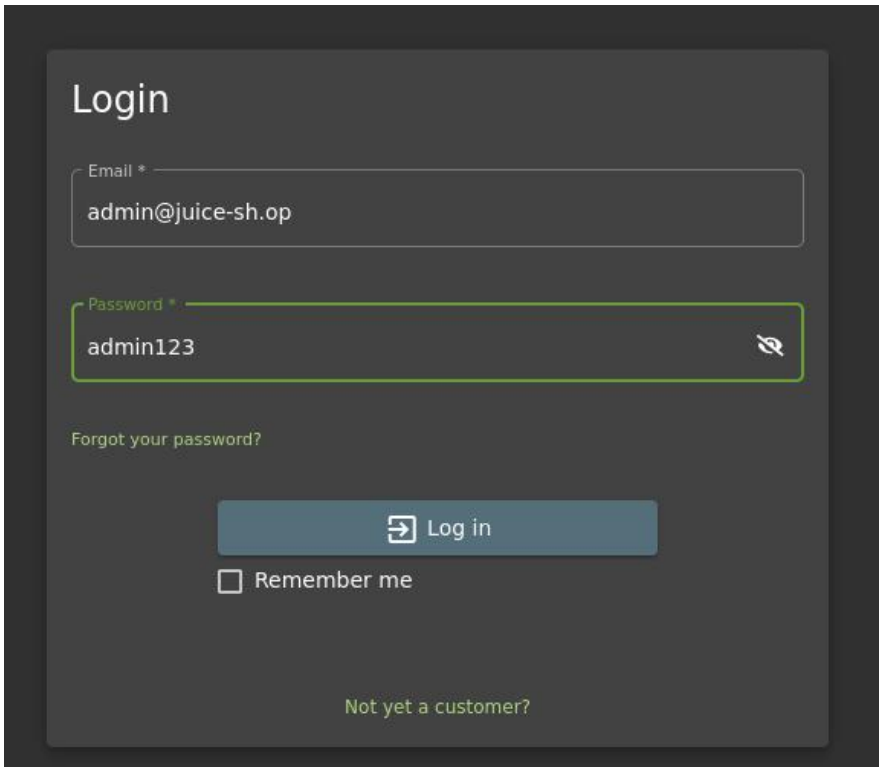


5. The progress of the attack starts. To know the correct password you need to find the 200 success status code from the list of requests that are being sent .

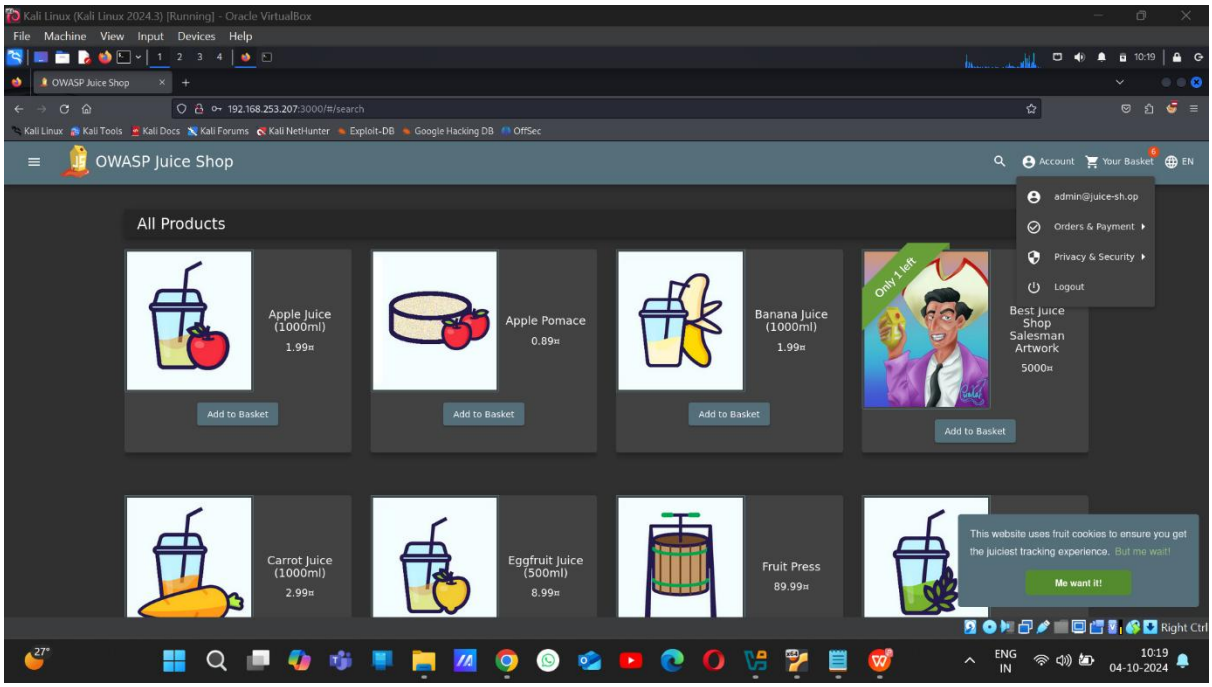
6. Click the status column header to arrange the status column in ascending or descending order to identify the status code 200 quickly.

7. As shown below a request successfully returned 200 for the payload “admin123” which means it successfully logged in as the administrator and “admin123” is the password of the admin you can use it to log in regularly.





8. Logged in Sucessfully.



Security Misconfiguration vulnerability

Where sensitive files and directories are left exposed on a server due to improper configurations. This could happen in cases like forgotten FTP directories, misconfigured permissions, or even files that were supposed to be deleted but remain publicly accessible.

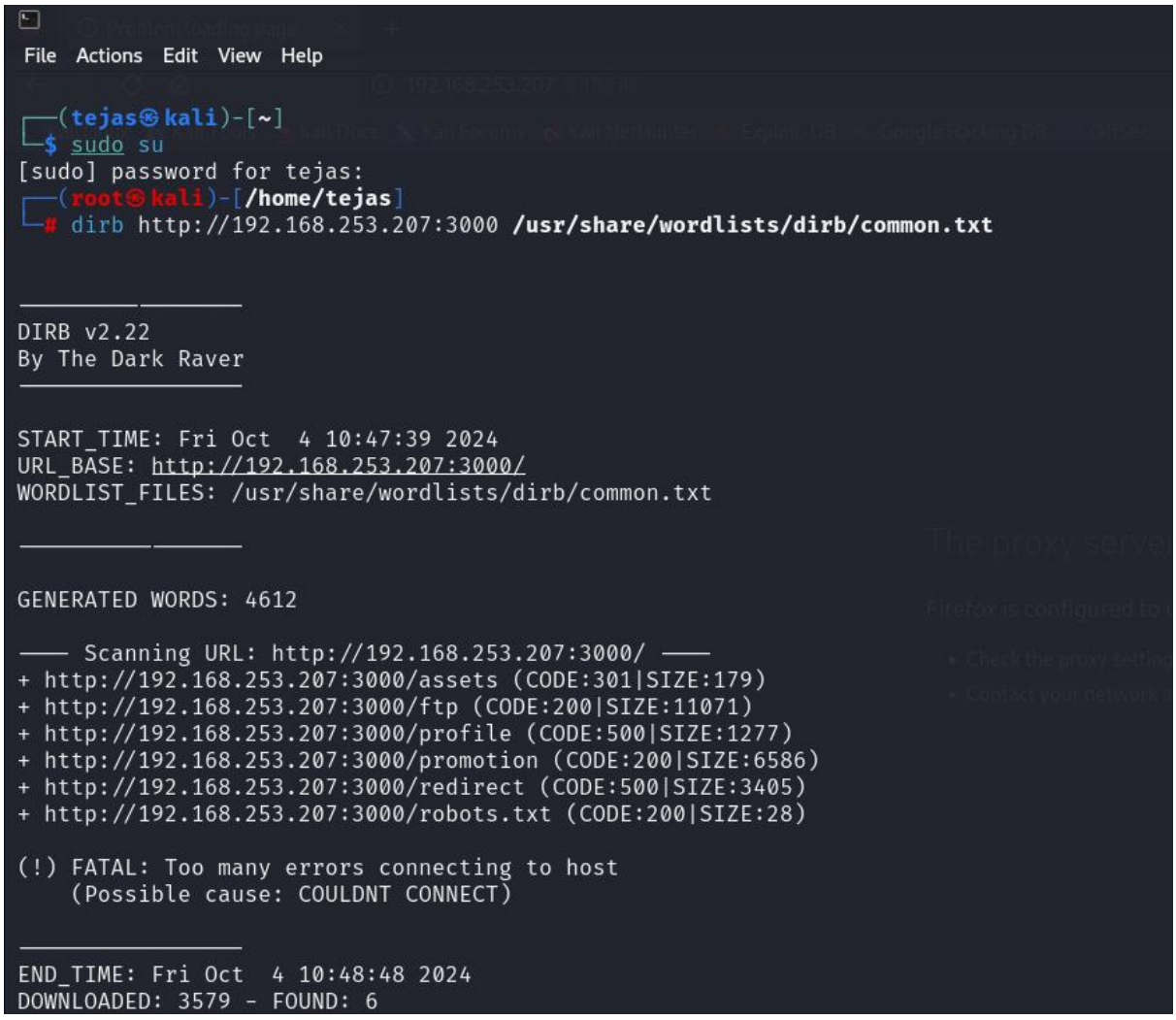
Here, used dirb tool. DIRB is a Web Content Scanner. It looks for existing (and/or hidden) Web Objects. It basically works by launching a dictionary based attack against a web server and analyzing the responses.

1.“/usr/share/wordlists/dirb/common.txt” is the wordlist that dirb uses to find common directory names. It contains a list of commonly used directories and filenames.

Used command : dirb http://192.168.253.207:3000 /usr/share/wordlists/dirb/common.txt

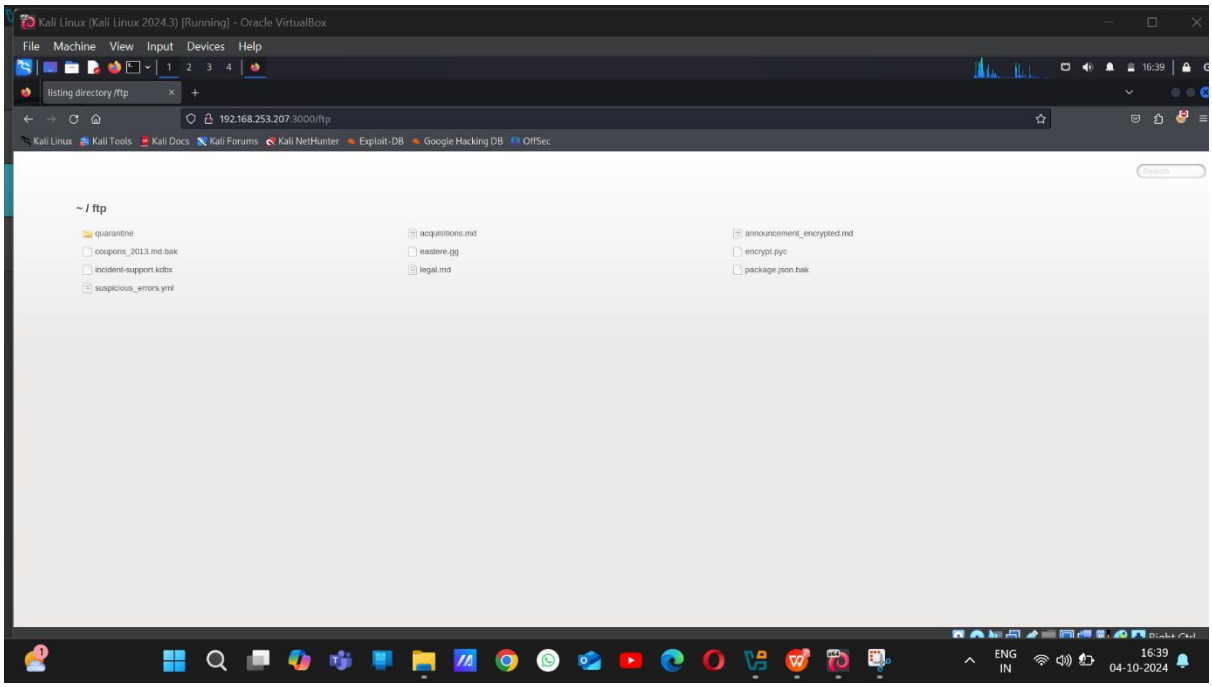
To find the Hidden directory.

2. When the scan completes analyse the results to see if anything would be a point of interest which leads to sensitive data exposure.



3. /ftp directory of the target is most likely to have sensitive files as it could be the directory listing of an ftp server. Browsing the ftp Directory.

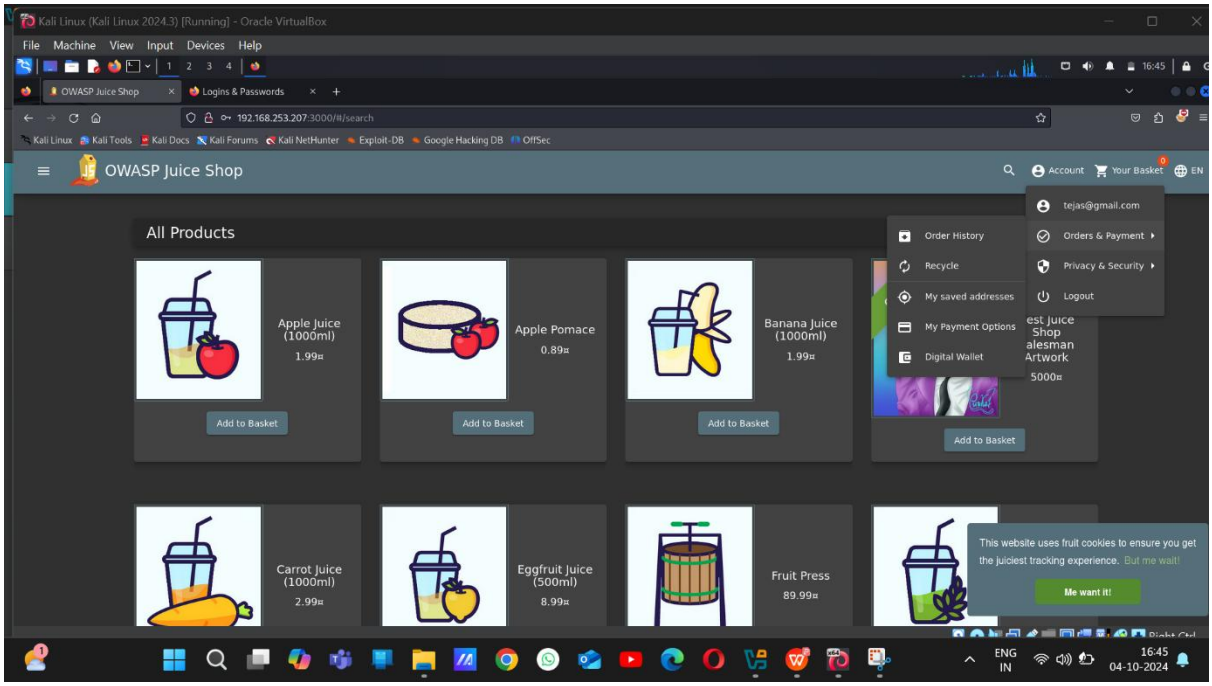
4. As shown below , sensitive files are available for public download without any protection



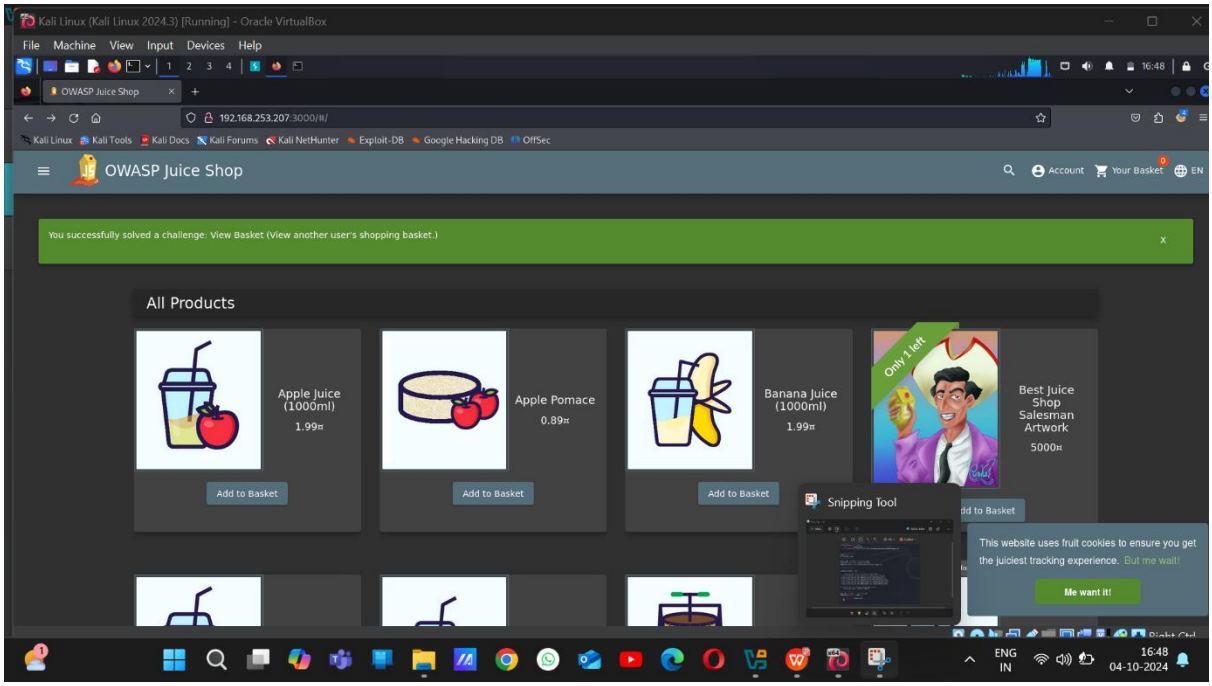
Viewing other user baskets (security vulnerability)

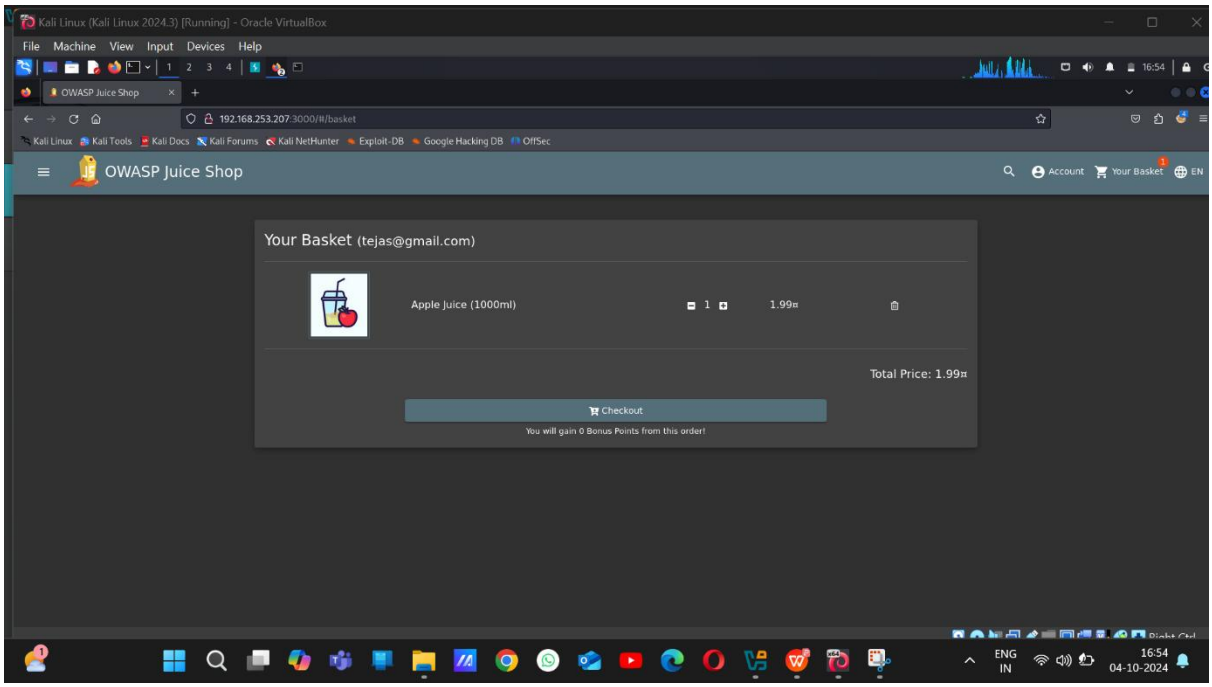
Where unauthorized users can view other users' data, such as their shopping baskets or other private information, in a web application. This violates the principles of **confidentiality** and **integrity** by allowing unauthorized access to sensitive information.

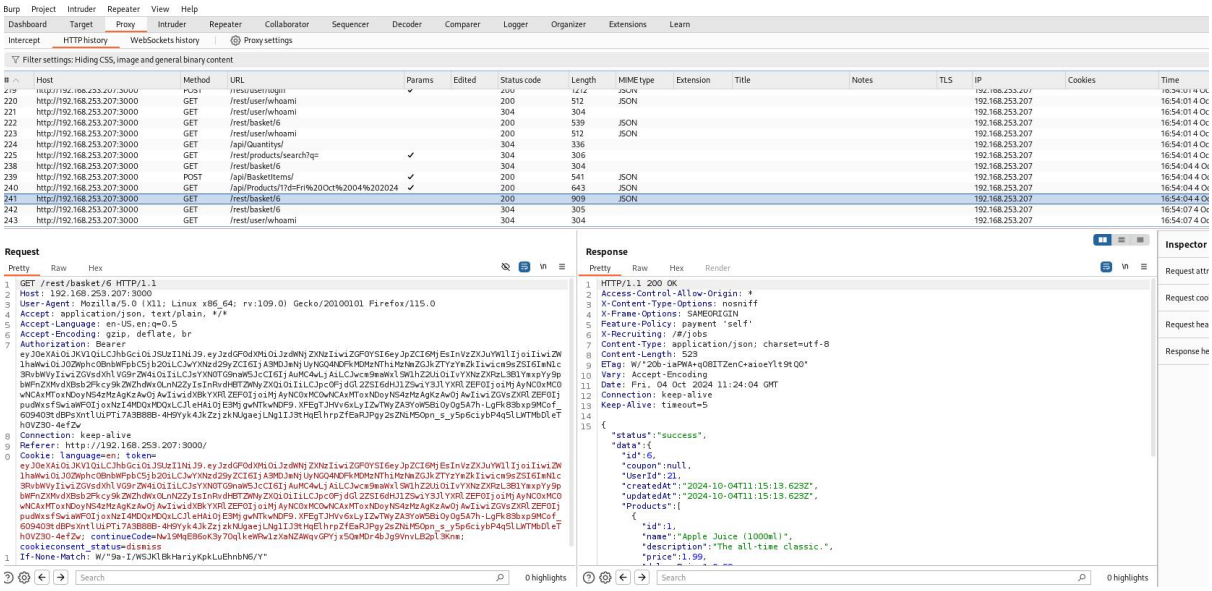
1.First register and login as a regular user



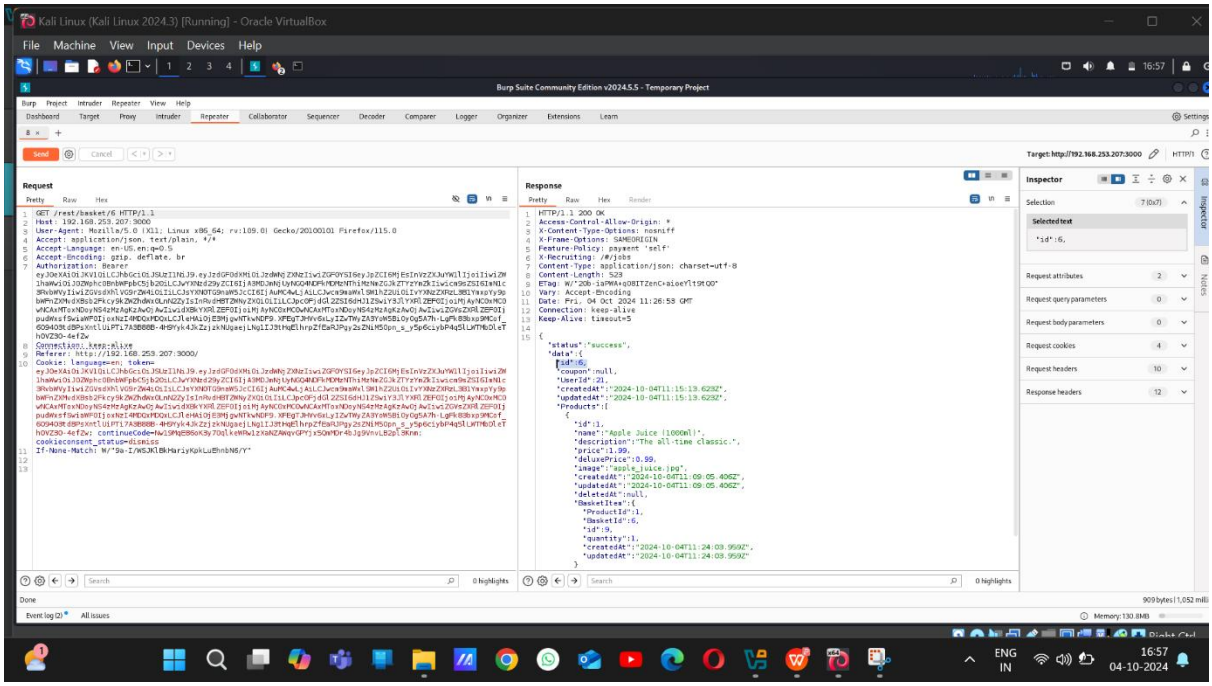
2. Capture the request(view basket) using burpsuite on clicking view basket.




Kali Linux (Kali Linux 2024.3) [Running] - Oracle VirtualBox
File Machine View Input Devices Help
OWASP Juice Shop
192.168.253.207:3000/#/basket
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
OWASP Juice Shop
Account
Your Basket
EN
Your Basket (tejas@gmail.com)
Apple Juice (1000ml)
1
1.99¤
Total Price: 1.99¤
Checkout
You will gain 0 Bonus Points from this order!
ENG
IN
16:54
04-10-2024

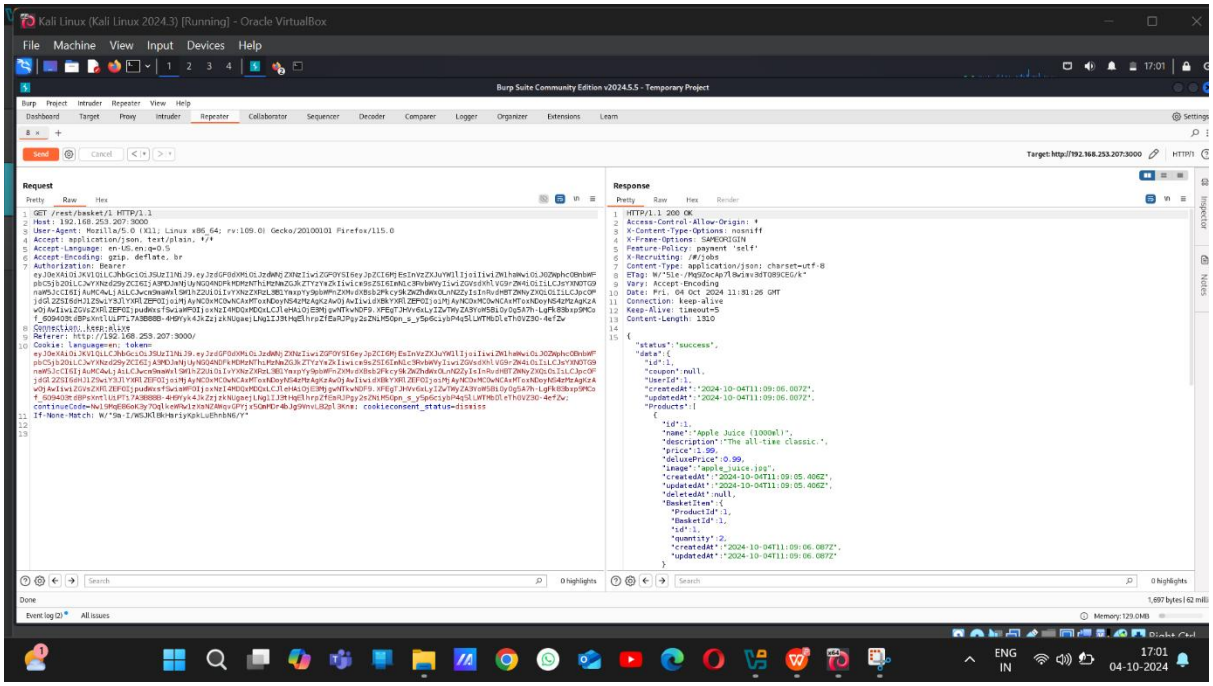

Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Intercept HTTP history WebSockets history Proxy settings
Filter settings: Hiding CSS, image and general binary content
Request
Response
Inspector
0 highlights
0 highlights

3. Send the request to the repeater tab to further analyze it.



4. Over here the id is reflected in the response from the api data sent as a request, and the id is linked to the individual basket assigned to the user, to see if it is vulnerable.we test it with another value to see if it can be accessed.

5. As seen below, doing this, and altering a different basket id we could view the basket content of other users .

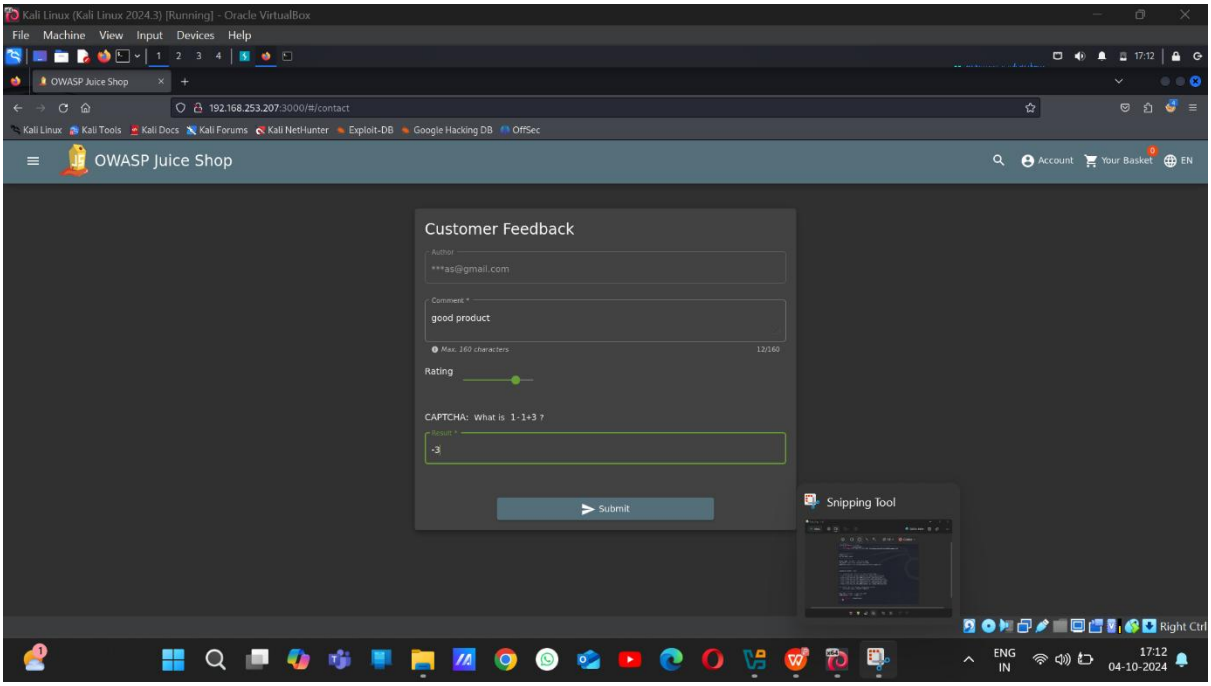


Forging Feedback

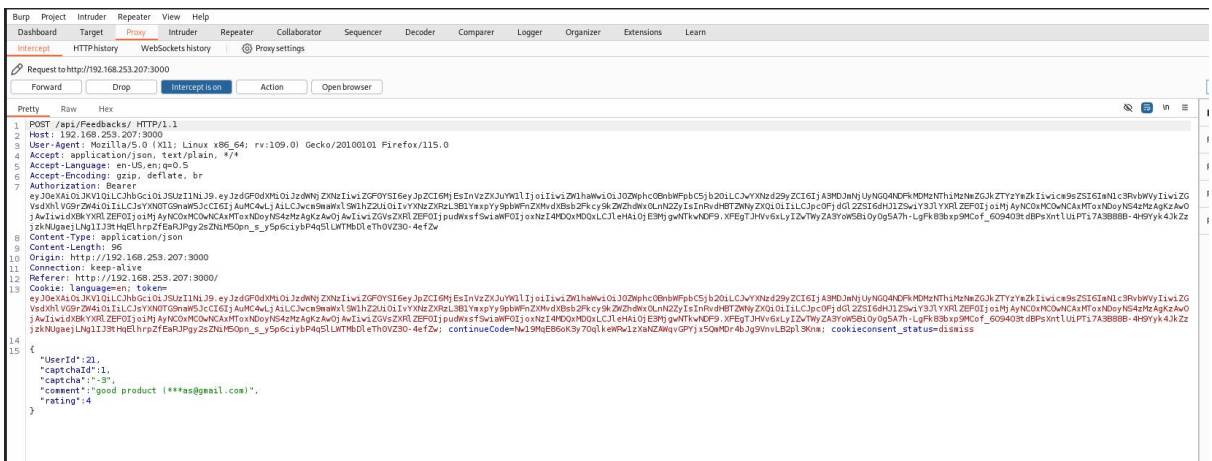
It is related to **impersonation attacks** and breaks an application's **authentication** and **authorization** systems. In this scenario, users can submit feedback while impersonating other users, which undermines trust and integrity in the system.

This problem arises when an application fails to properly authenticate and authorize users before accepting feedback or other actions.

1. Go to the Customer feedback page located in the application **<target>/#/contact** directory as shown below.



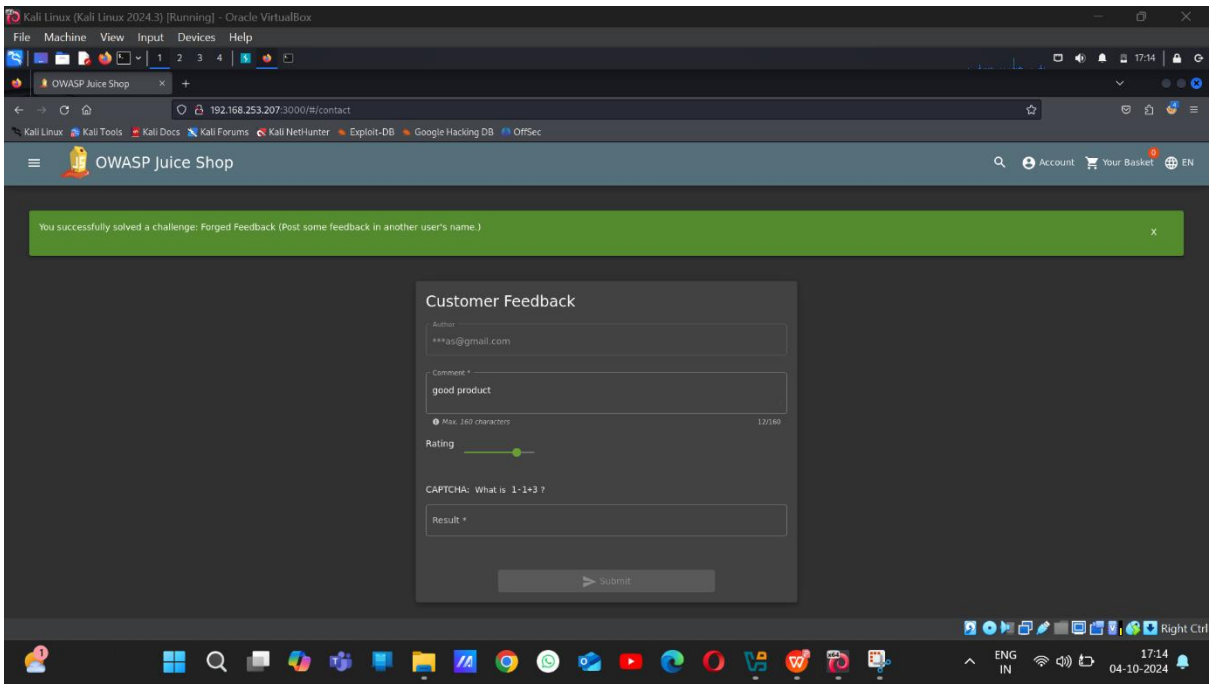
2. Fill the feedback form and press on submit button. Capture the submit request in the burpsuite.



3. Edit the user id parameter to impersonate another user as shown below and forward the request.

```
VsdXhlVG9rZW4iOiIiLCJsYXN0TG9naW5JcCI6IjAuMC4wLjAiLCJwc
jAwIiwidXBkYXRlZEF0IjoiMjAyNC0xMC0wNCAxMToxNDoyNS4zMzAg
jzkNUgaejLNglIJ3tHqElhrpZfEaRJPgy2sZNiM5Opn_s_y5p6ciybP
.4
.5 {
     "captchaId":1,
     "captcha":"-3",
     "comment":"good product (***as@gmail.com)",
     "rating":4,
     "UserId":1
   }
```

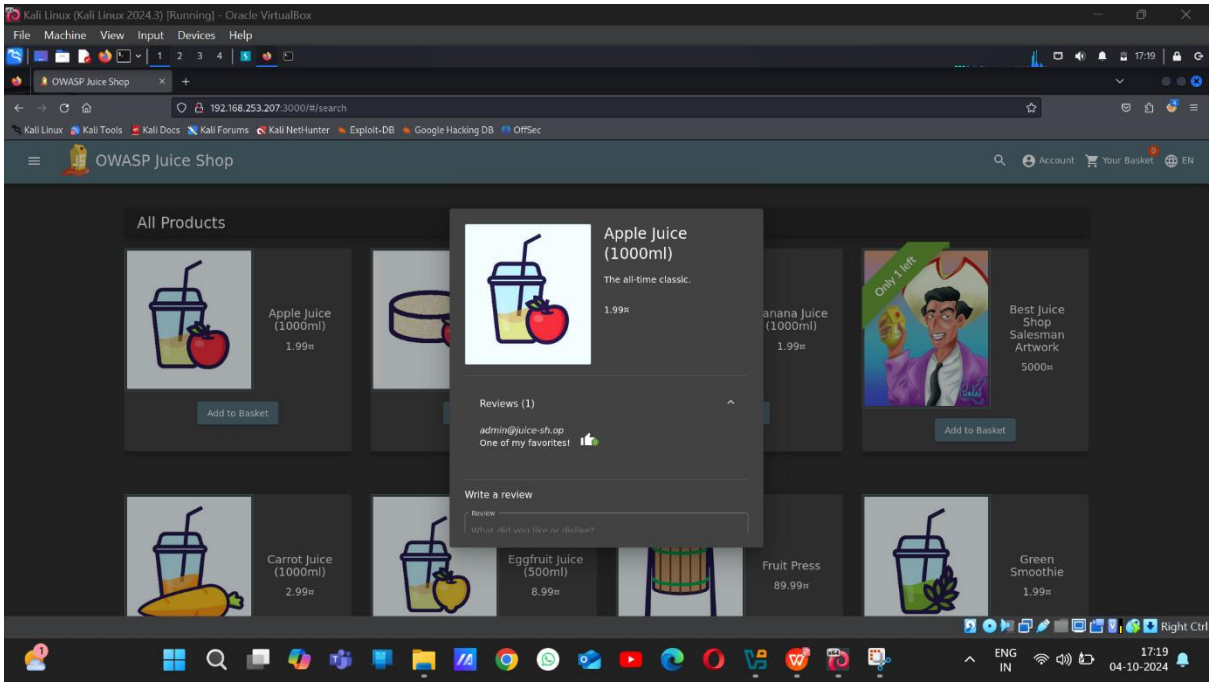
4. The request will be successful and the feedback will be made as admin as admin userID is 1.



Forging Review

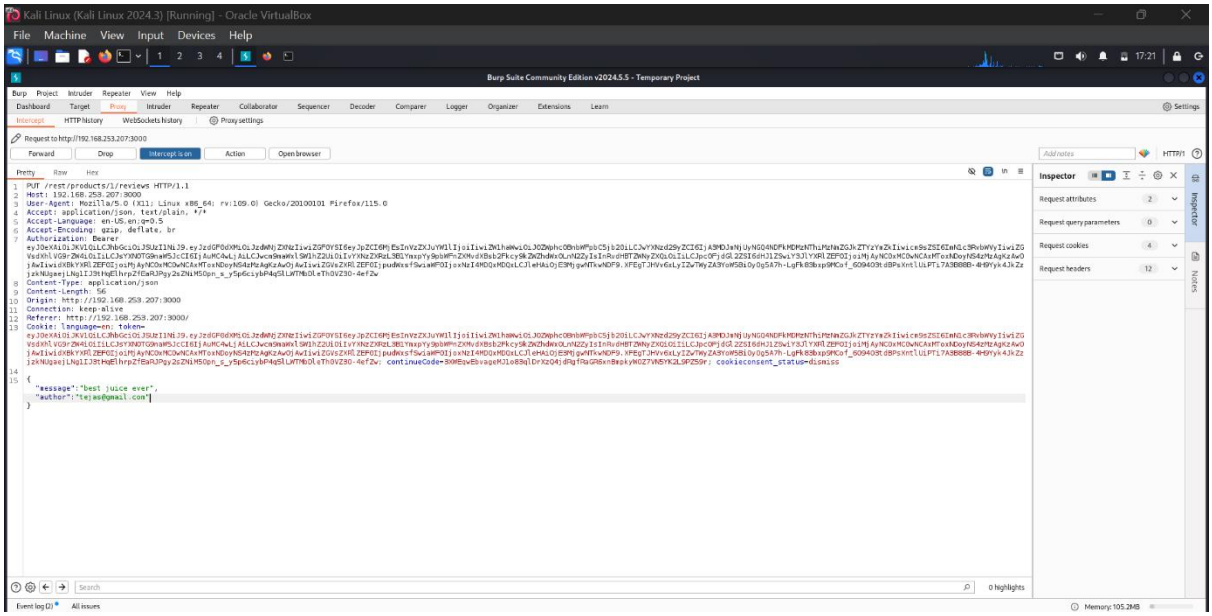
Forging a review is similar to forging feedback, where an attacker manipulates the system to post a review impersonating another user. This type of attack undermines the **integrity** of the review system and breaches both **authentication** and **authorization** mechanisms in the application.

1.Login your account and click on any product.

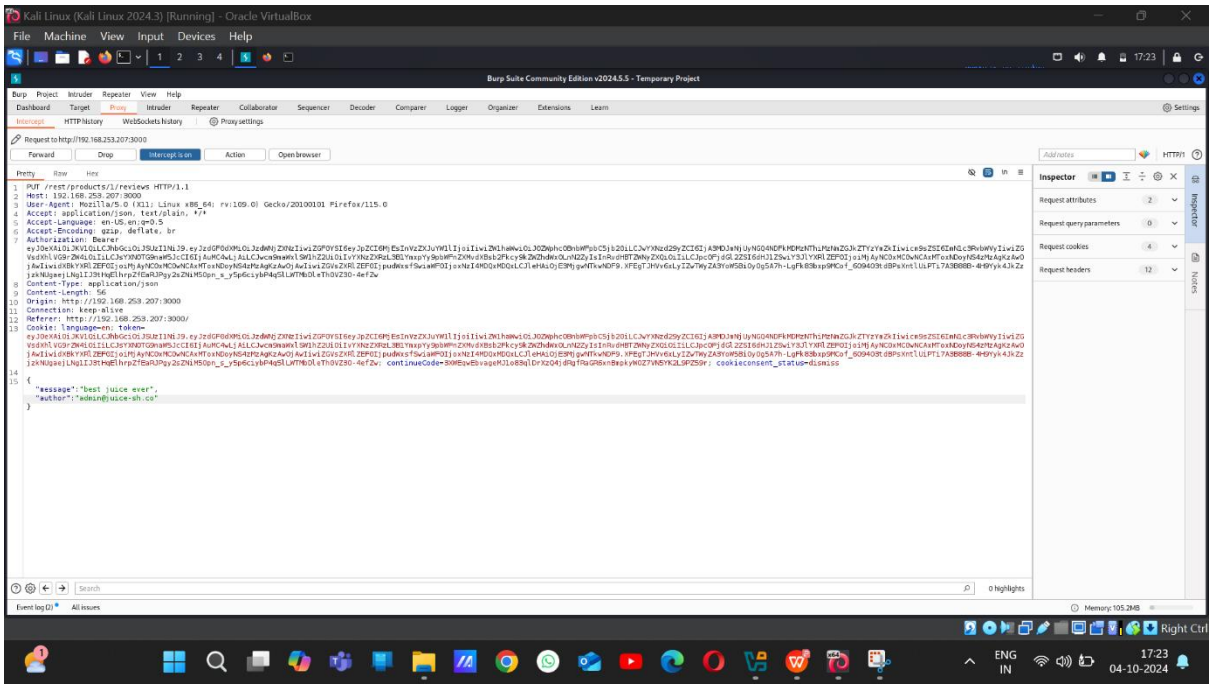


2.Write any reviw like “Best juice ever” and press submit and intercept the review request in burp suite.

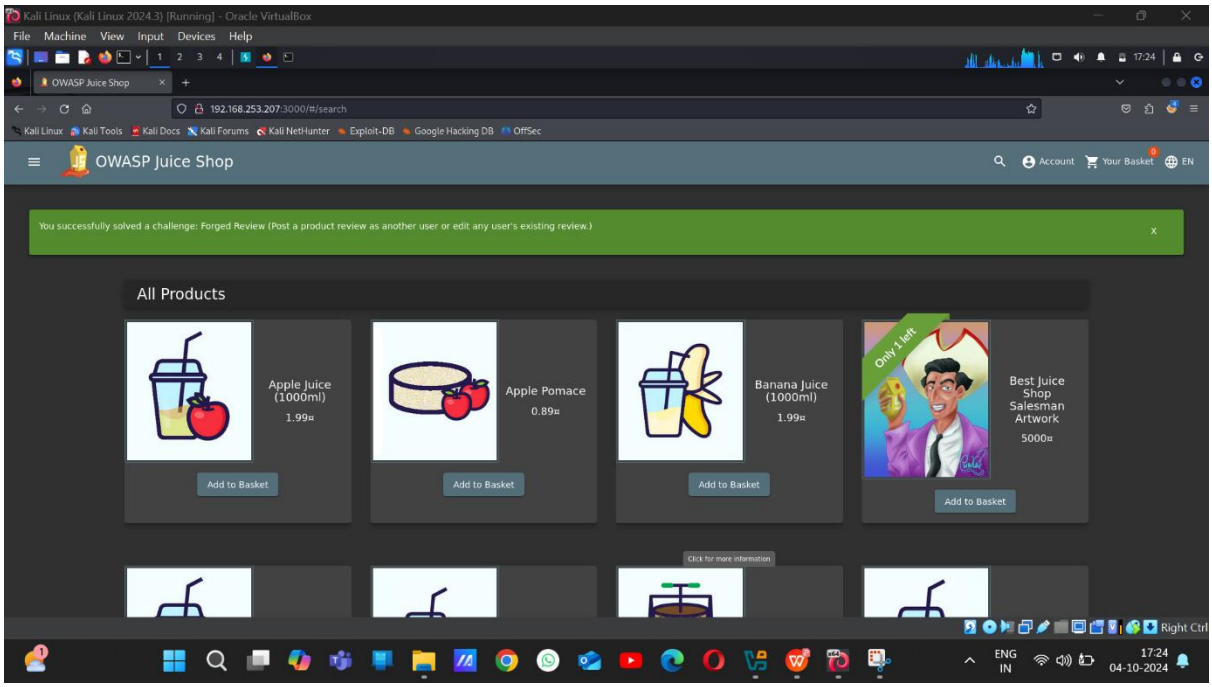
3.Capture the review request



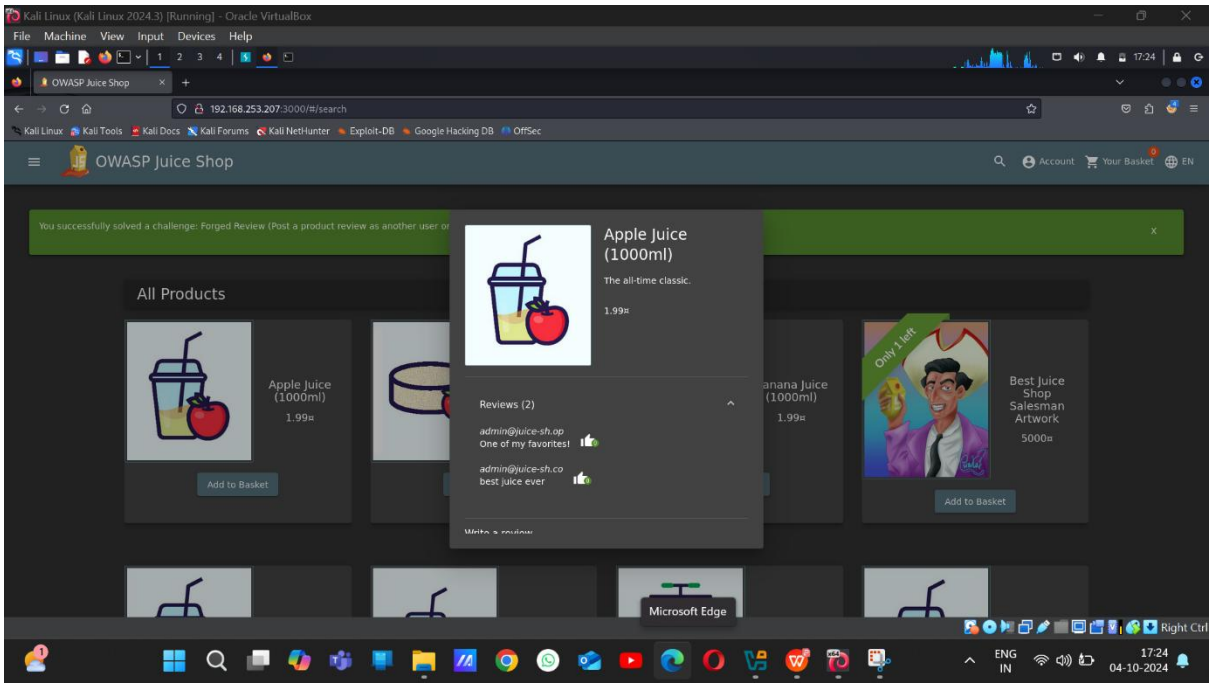
3. Change the author value to any other user and forward the request.



4. Review Forging is successfully performed.



5. Check the updated review, it would be post with a review impersonating another user .

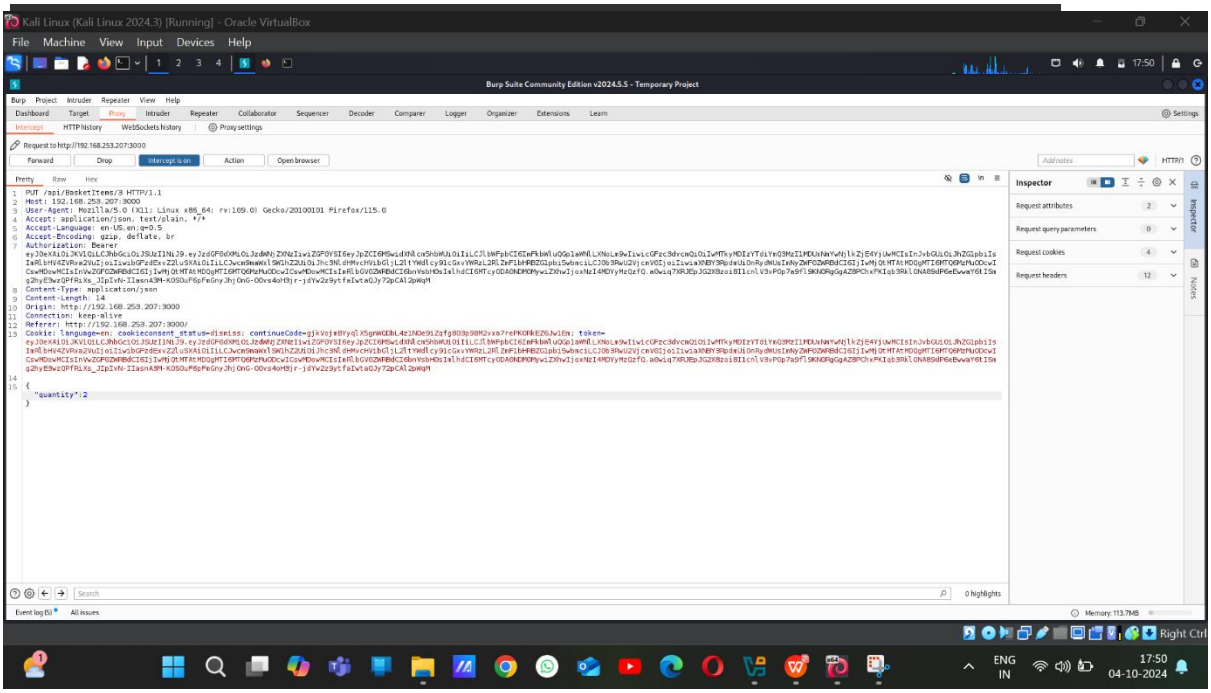


Manipulate other users basket

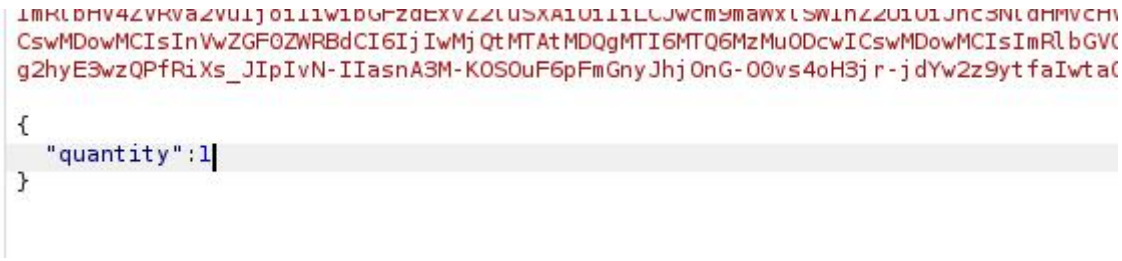
Manipulating another user's basket (or shopping cart) is a serious security vulnerability that can undermine the integrity and trustworthiness of an application, particularly in e-commerce platforms. This type of vulnerability often involves exploiting **API weaknesses** or **session management flaws**, and it typically requires an attacker to bypass certain security measures.

1. Login , click on add basket and capture the add a basket request in the burpsuite.

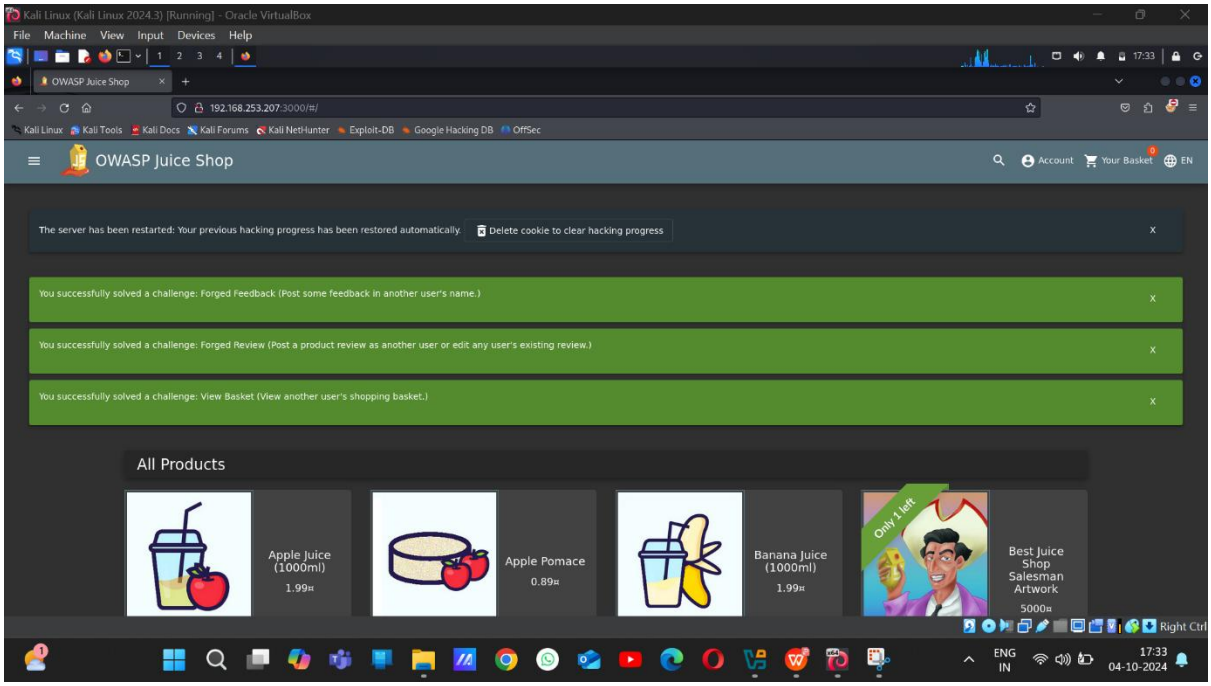
2. Quantity of basket of the victims has value “2” in the payload like shown .



3. By changing the quantity value to 1 and forward the request .



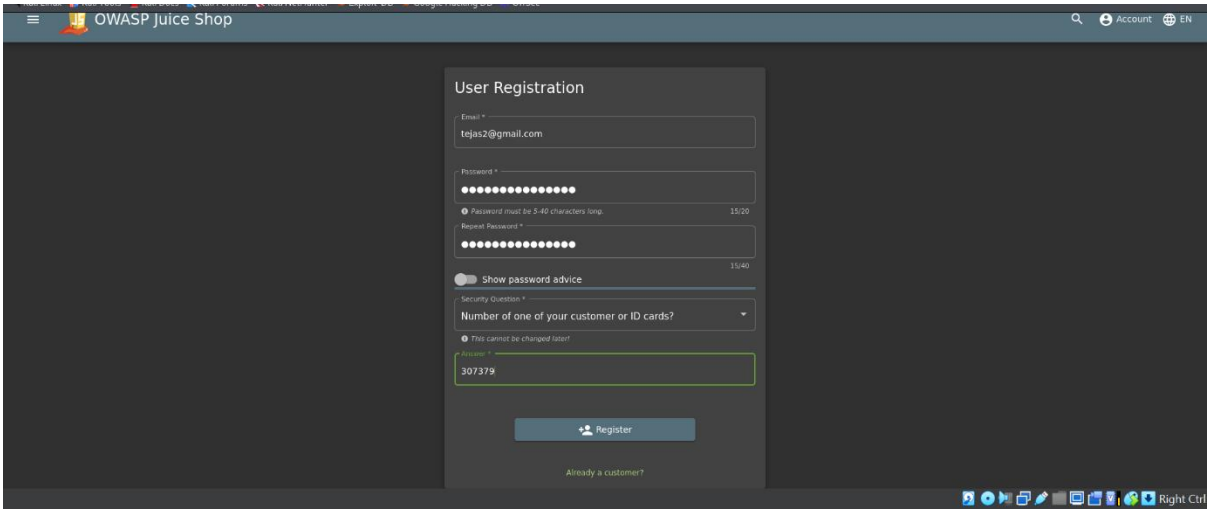
3. Log out and login to the victims account to verify if the victim’s basket is manipulated.



Admin Registration

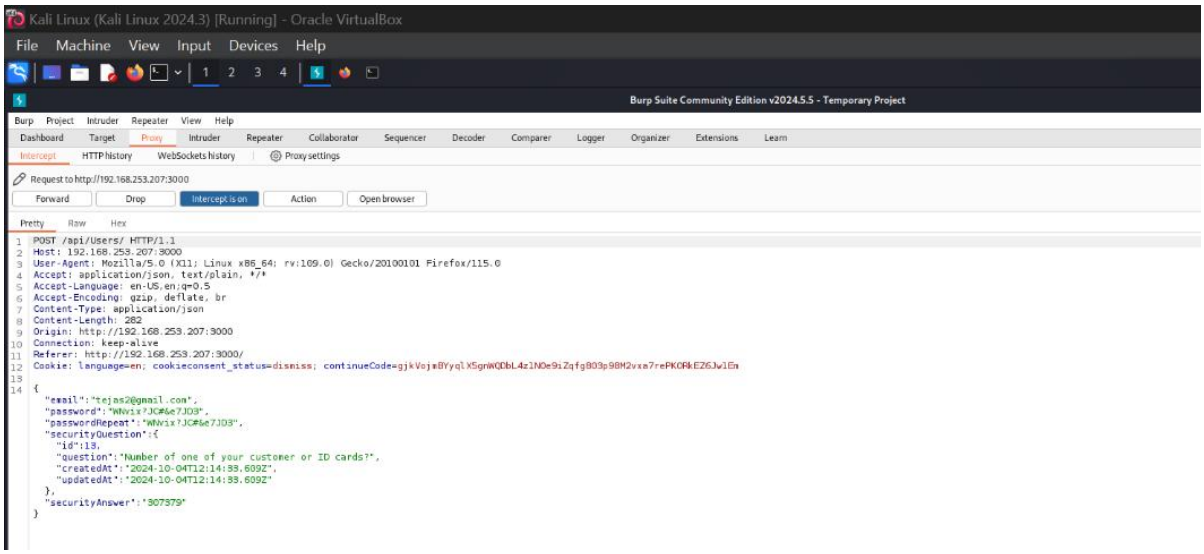
It is improper handling of **authorization** and **input validation**, specifically in the context of user registration. If the application allows regular users to register as administrators or privileged users, it undermines the entire **role-based access control** (RBAC) system, which is critical for maintaining security in applications.

In this scenario, it appears that the registration function does not properly **sanitize input** or enforce **authorization checks**, allowing normal users to potentially gain administrative privileges.

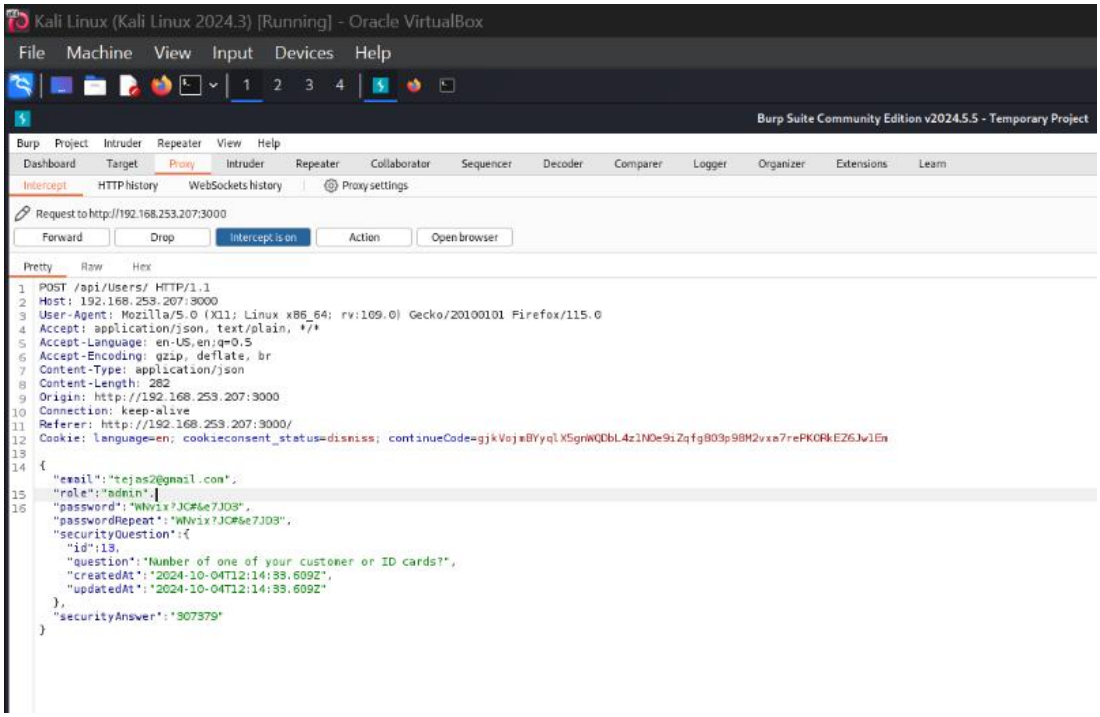


1. Go to login page and click on, “Not yet a customer ?” to start a registration process.

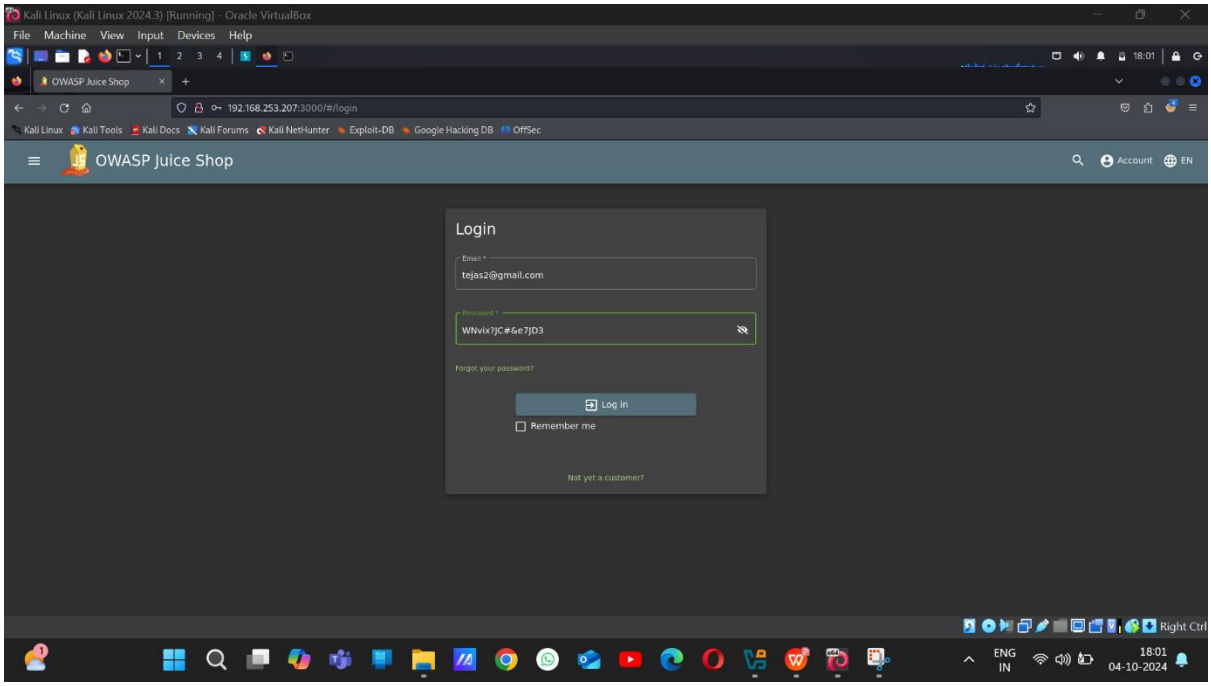
2. Enter the registration details and click on Register and capture the register request in Burpsuite.

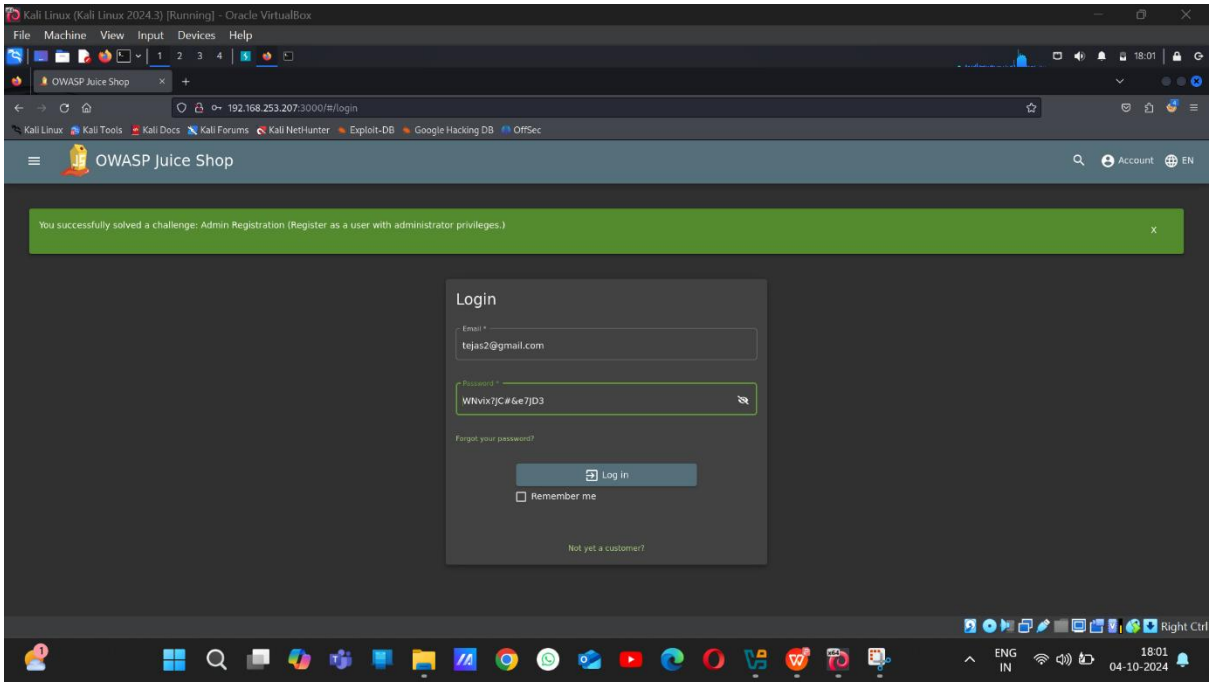


3. Add a payload as “role”:”admin” as shown below and forward the request .

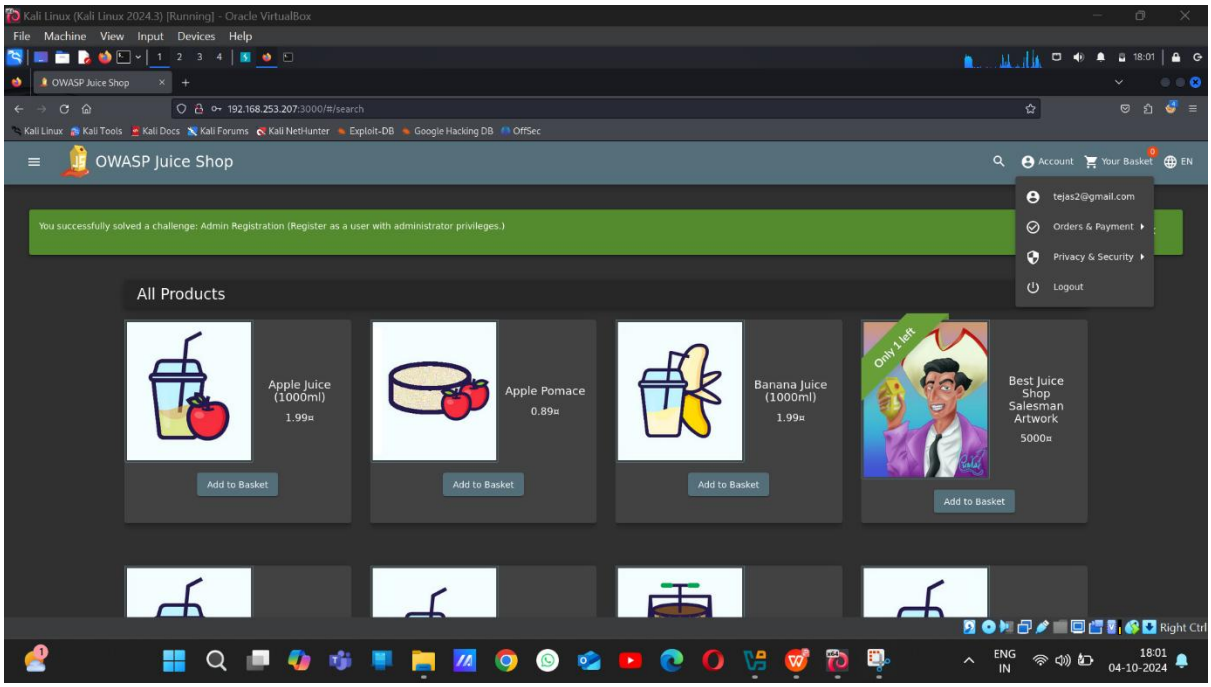


4. Account will be created with admin privileges, you will be able to login with the credentials to perform actions as equal as the Administrator as shown below.





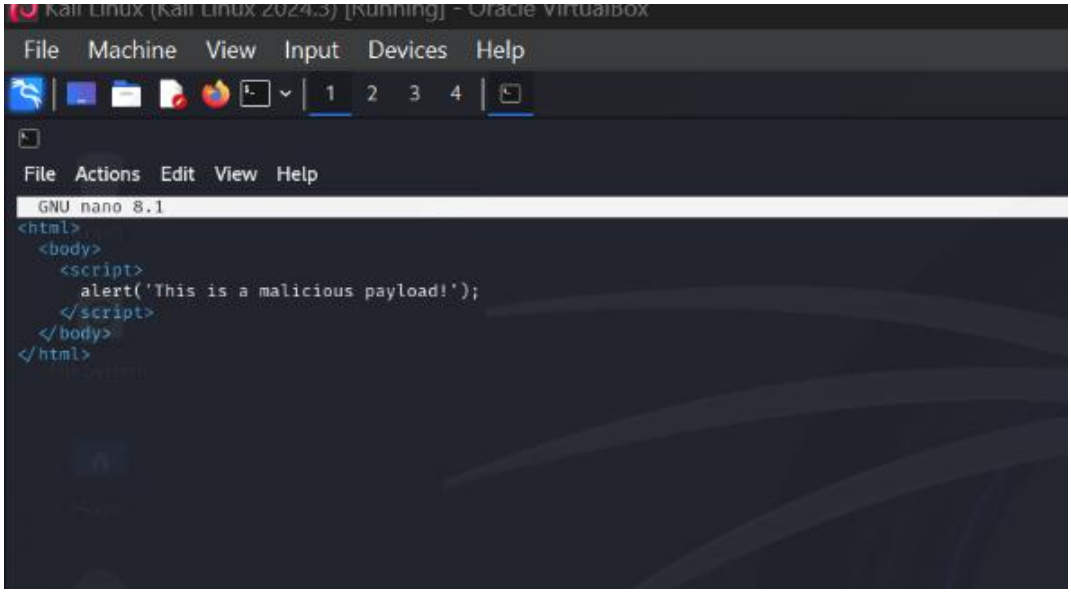
5. Successfully logged in with admin privileges.



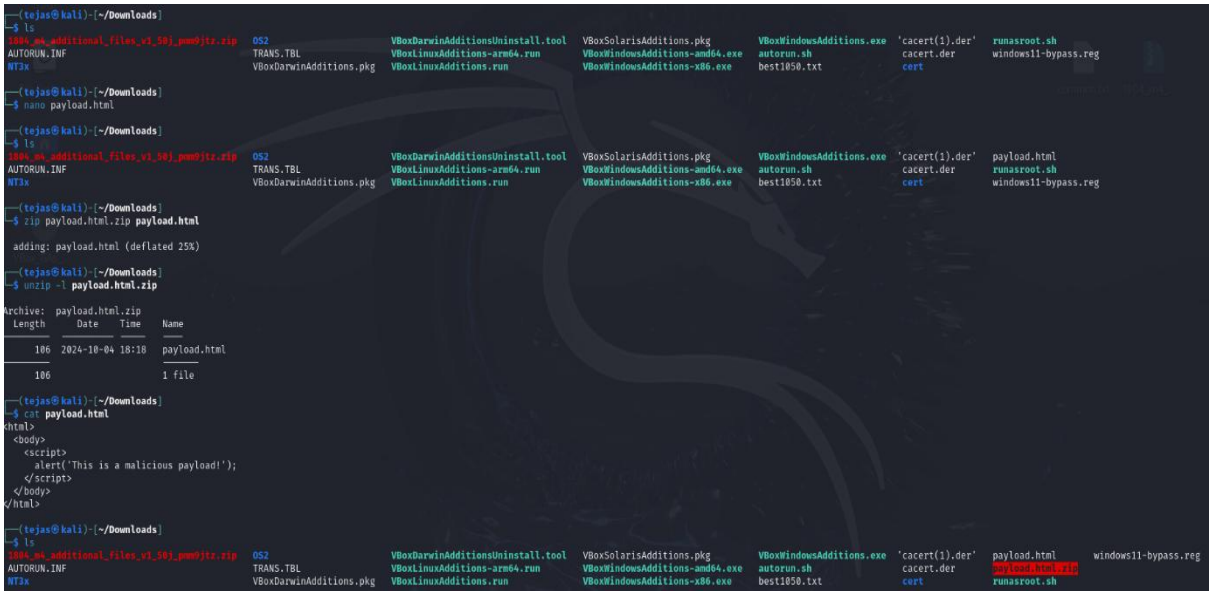
File Type Upload

Uploading unsupported or harmful file types, such as **.exe, .sh, .php,** or **.html**, to a server can lead to critical vulnerabilities, including **remote code execution (RCE)**, **cross-site scripting (XSS)**, or other forms of **malware injection**. If file uploads are not properly restricted and validated, attackers can exploit this to execute malicious code, steal sensitive data, or take control of the server.

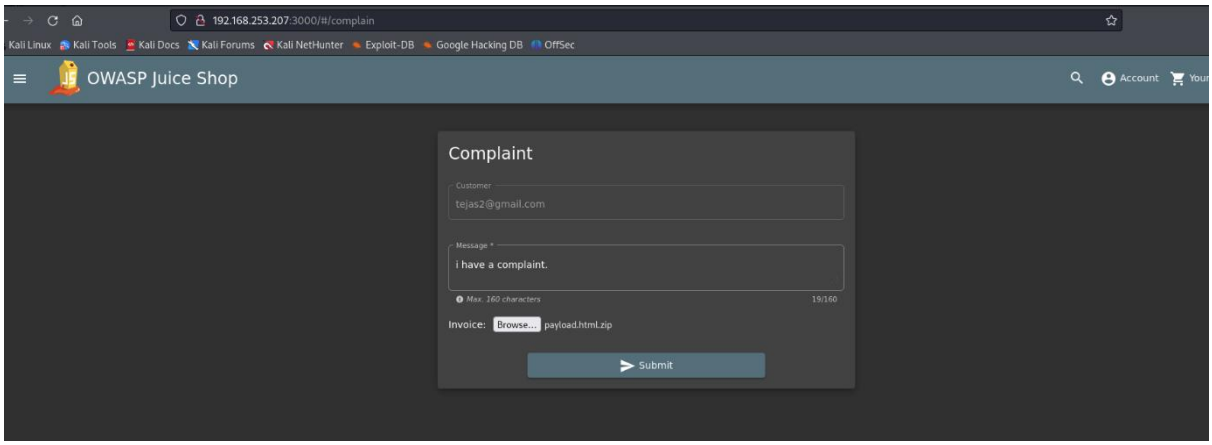
1.Creating a payload with HTML extension.

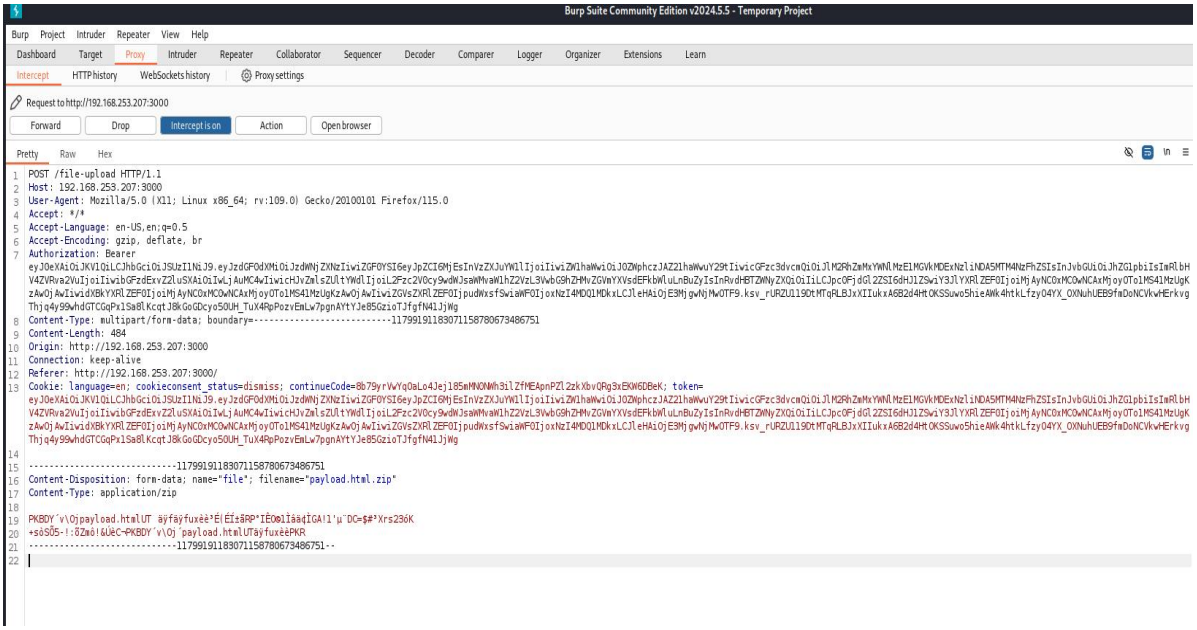


2. Attach the file extension as .zip this is to bypass the local file picker . Now filename is “payload.html.zip”.

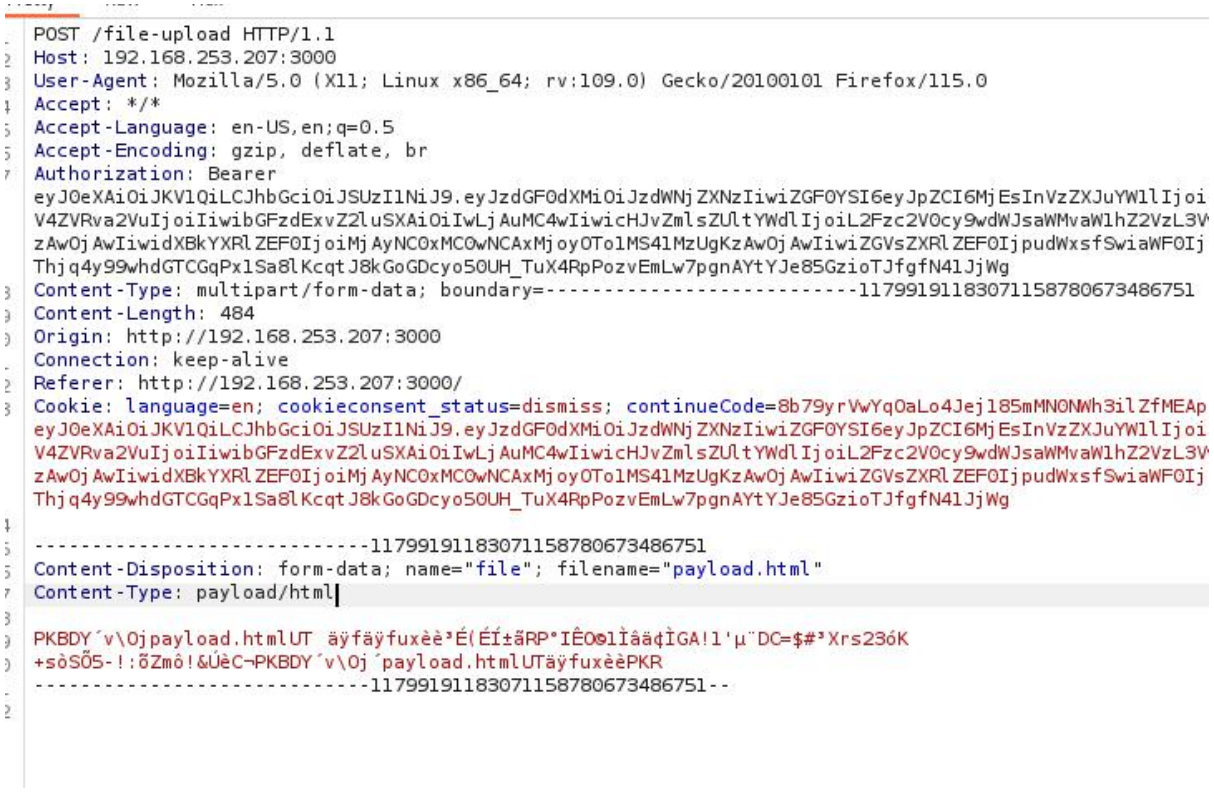


3. Go to complain page in the Application. And write any message and attach payload which we have created “payload.html.zip”. click on submit and intercept the file upload request in Burp Suite for analyzing.





4. Now Edit the filename and Content-type to “payload/html” and forward the request.



```
POST /file-upload HTTP/1.1
Host: 192.168.253.207:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZCI6MjEsInVzZXJuYW1lIjoi
V4ZVRva2VuIjoiIiwibGFzdExvZ2luSXAiOiIwLjAuMC4wIiwicHJvZmlsZUltYWdlIjoiL2Fzc2V0cy9wdWJsaWMvaW1hZ2VzL3V
zAwOjAwIiwidXBkYXRlZEF0IjoiMjAyNC0xMC0wNCAxMjoyOTo1MS41MzUgKzAwOjAwIiwiZGVsZXRlZEF0IjpudWxsfSwiaWF0Ij
Thjq4y99whdGTCGqPx1Sa8lKcqtJ8kGoGDcyo50UH_TuX4RpPozvEmLw7pgnAYtYJe85GzioTJfgfN41JjWg
Content-Type: multipart/form-data; boundary=---------------------------117991911830711587806734867 51
Content-Length: 484
Origin: http://192.168.253.207:3000
Connection: keep-alive
Referer: http://192.168.253.207:3000/
Cookie: language=en; cookieconsent_status=dismiss; continueCode=8b79yrVwYqOaLo4Jej185mMN0NWh3ilZfMEAp
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZCI6MjEsInVzZXJuYW1lIjoi
V4ZVRva2VuIjoiIiwibGFzdExvZ2luSXAiOiIwLjAuMC4wIiwicHJvZmlsZUltYWdlIjoiL2Fzc2V0cy9wdWJsaWMvaW1hZ2VzL3V
zAwOjAwIiwidXBkYXRlZEF0IjoiMjAyNC0xMC0wNCAxMjoyOTo1MS41MzUgKzAwOjAwIiwiZGVsZXRlZEF0IjpudWxsfSwiaWF0Ij
Thjq4y99whdGTCGqPx1Sa8lKcqtJ8kGoGDcyo50UH_TuX4RpPozvEmLw7pgnAYtYJe85GzioTJfgfN41JjWg

-----------------------------117991911830711587806734867 51
Content-Disposition: form-data; name="file"; filename="payload.html"
Content-Type: payload/html

PKBDY´v\Ojpayload.htmlUT äÿfäÿfuxèè³É(ÉÍ±ãRP°IÊO¤lÌâä¢ÌGA!1'µ¨DC=$#³Xrs23óK
+sòSÕ5-!:õZmô!&ÚèC¬PKBDY´v\Oj´payload.htmlUTäÿfuxèèPKR
-----------------------------117991911830711587806734867 51--
```

5. The request will be successfully made and it will bypass the restrictions and now, I am able to upload the file with a non supported file type.