CYBER SECURITY REPORT

Description:
This project shows how to find and use weaknesses in OWASP Juice Shop. The goal is to learn about common ways hackers attack websites and understand why strong security is important.
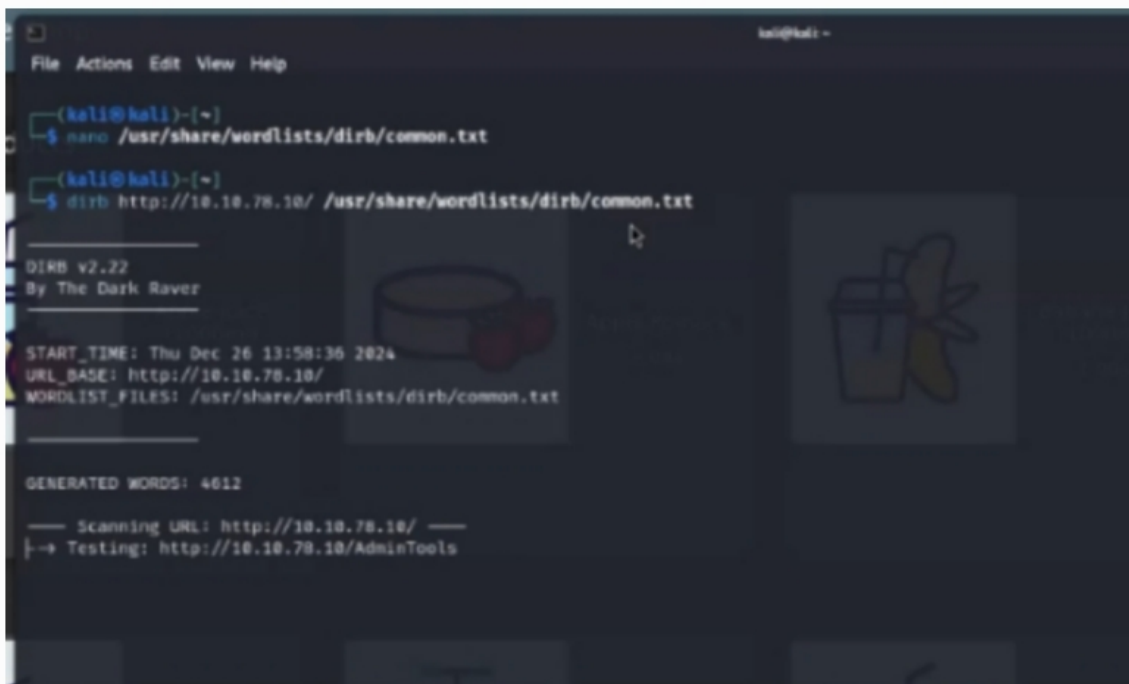
Tools used:

1.dirb

2.burpsuite

attacks performed:
1.url dictionary attack
2.SQL injection
3.bruteforce attack
4.XSS attack

URL dictionary attack

I tried using dirb to find the admin page of the OWASP Juice Shop, but it didn't show up. This meant the admin page wasn't handled on the server side, where tools like dirb would normally detect it. Instead, the admin page was part of the client side, which is handled in the browser. I found the admin page by checking the JavaScript code in the webpage, where its path was revealed.
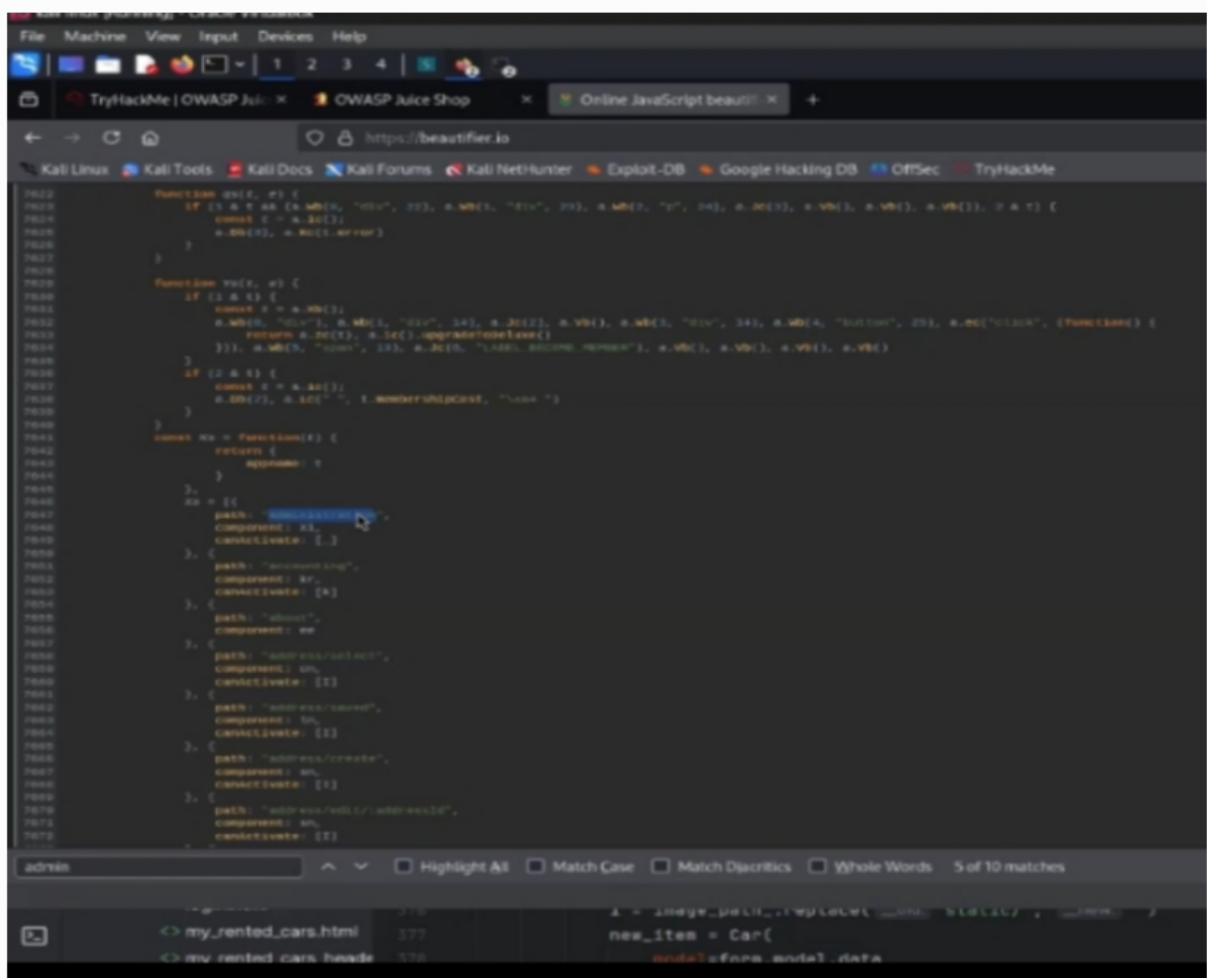
admin page:
/administration
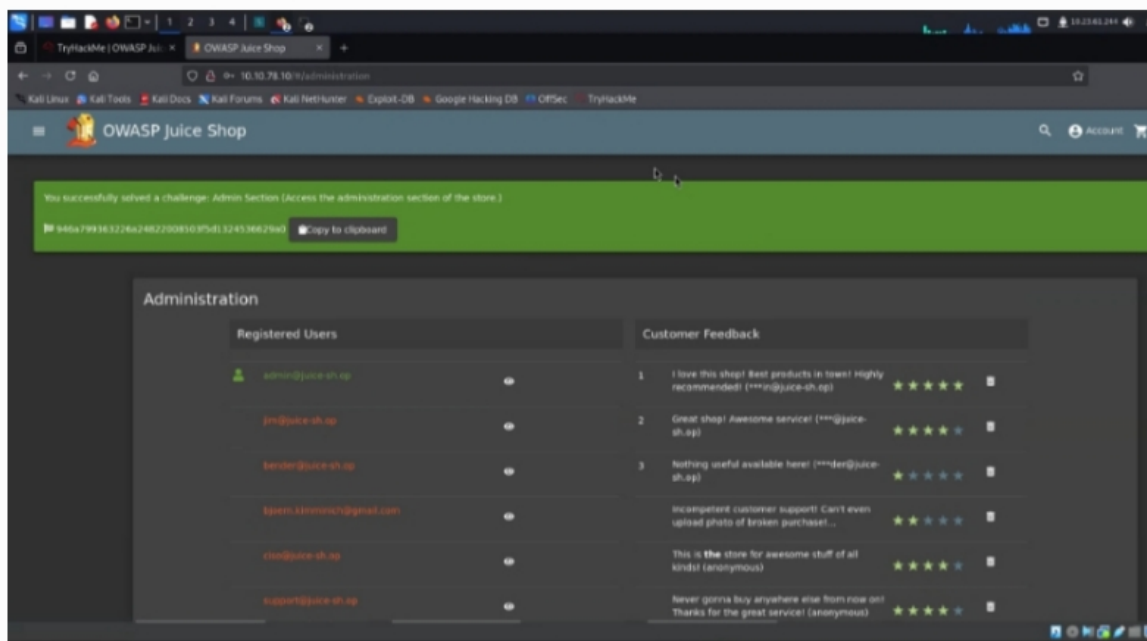
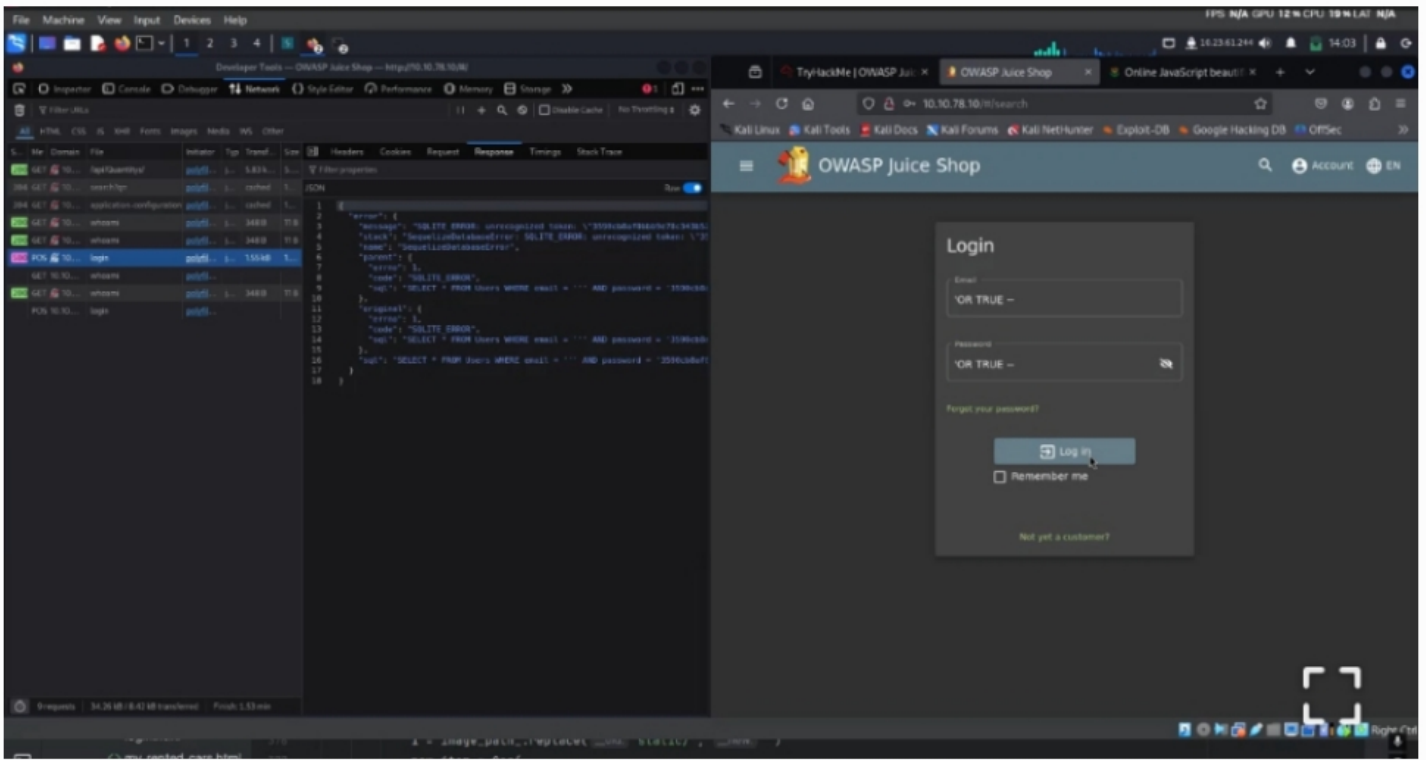# SQL injection

I used SQL injection to log into an account by entering 'OR TRUE -- in both the email and password fields. This allowed me to access the admin@juice-sh.op account. Next, I plan to use brute force to find the password for this account.

Brute force attack

Since I already had access to the admin@juice-sh.op account, I suspected that the password might be related to the word "admin" or "administration." To test this, I used the small.txt wordlist, which contains many variations of these terms. Then, I used the Intruder tool in Burp Suite to brute-force the password for the admin@juice-sh.op account.
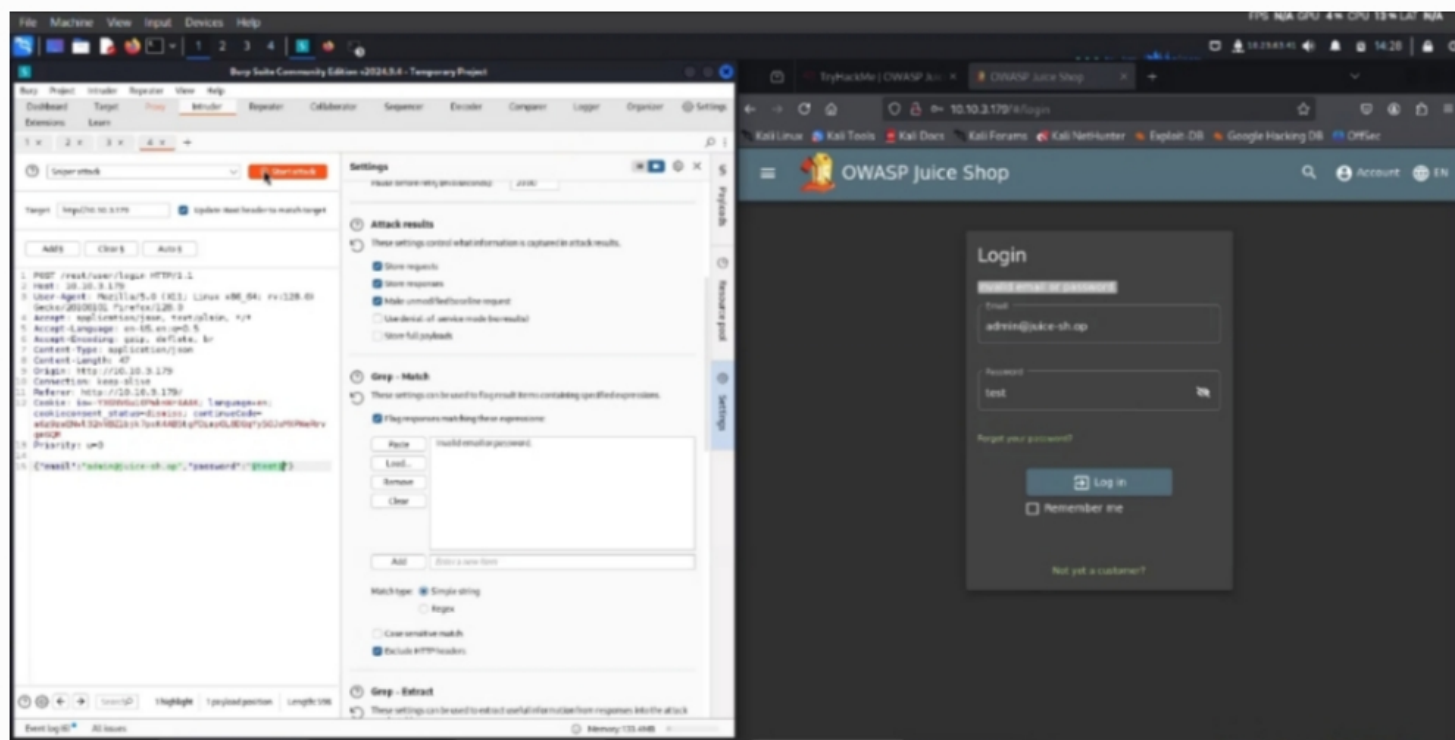
and the password of admin@juice-sh.op turned out to be admin123

XSS attack

The last attack I performed was testing for XSS. I started by adding a <script></script> tag in the search bar to check if the site was vulnerable to XSS, but it turned out not to be. Next, I tested for HTML injection by entering an <h1></h1> tag, and the site was vulnerable. Using this, I injected JavaScript code inside the <h1> tag with an onclick event to successfully perform an XSS attack.
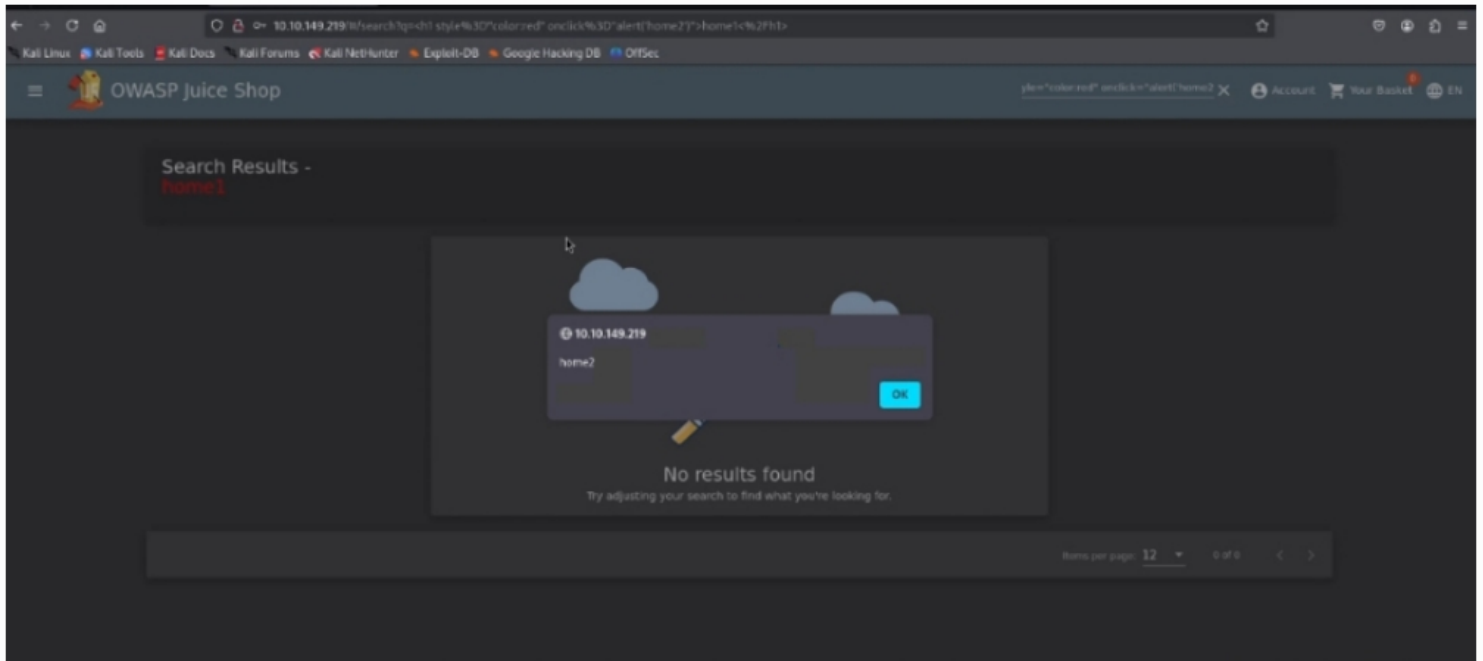
Explaining 4 attacks used generally

1.Dirb Attack (URL dictionary attack):

This is a technique used to find hidden directories or files on a web server. Attackers use a tool like dirb to scan a website by trying out many common directory and file names (like /admin, /login, or /config). If a hidden directory or file exists but is not properly protected, the attacker can access it.

2.SQL Injection:

SQL Injection occurs when an attacker inserts malicious SQL code into an input field (like a login form or search box) on a website. If the website doesn't properly handle user inputs, the attacker can manipulate the SQL query to retrieve or modify data in the website's database. For example, they might get access to sensitive information like usernames and passwords.

3.Brute Force Attack:

A brute force attack is when an attacker tries every possible combination of characters to guess something, like a password or encryption key. It's like repeatedly trying different combinations of numbers and letters until the correct one is found. If the password is weak or short, the attacker can eventually guess it.

4.Cross-Site Scripting (XSS):

XSS happens when an attacker injects malicious JavaScript code into a website's content. When other users visit the page, the malicious code is executed in their browsers, allowing the attacker to steal cookies, session tokens, or manipulate the content they see. For example, they might steal login information or trick the user into clicking on something harmful.

How to prevent attacks

1.Preventing Dirb Attack (Directory Buster):

Use proper server configuration: Disable directory listing on the server so that attackers can't see the structure of the website.

Obfuscate or rename sensitive paths: Avoid using common names like /admin or /login for critical directories.

2.Preventing SQL Injection:

Validate and sanitize user input: Ensure all user inputs are checked for validity, and dangerous characters (like -- or ') are removed or escaped. Limit database privileges: Use the principle of least privilege, ensuring database accounts only have the permissions they need.

3.Preventing Brute Force Attack:

Use strong passwords: Encourage users to create complex passwords (mix of letters, numbers, and symbols).
Implement account lockout: After a certain number of failed login attempts, lock the account temporarily or require CAPTCHA verification.
Enable multi-factor authentication (MFA): Require a second form of verification (like a code sent to a phone) to add an extra layer of security.

4.Preventing Cross-Site Scripting (XSS):

Sanitize user input: Remove or encode potentially harmful characters from user inputs (like <, >, and &), so that they are not executed as HTML or JavaScript.
Use Content Security Policy (CSP): Implement CSP headers to restrict which scripts can be executed on your site, preventing malicious scripts from running.
Validate and escape output: Ensure that all dynamic content (especially user-generated content) is properly encoded when displayed in the browser, so it can't execute as a script.

2305073	Mina Ayman
2305048	Youssef Ahmed
2305049	Youssef Hassan