CW1

PRACTICAL PENTESTING

DECEMBER 15, 2021

Toqa Mahmoud

CU1900305

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

EXECUTIVE SUMMARY

The major purpose of this web application (Black box) penetration testing project was to discover any possible areas of concern with the programme in its current condition, as well as to establish the extent to which the system may be compromised by an attacker with a certain skill set and motive. The report highlights security flaws, erroneous business logic, and a lack of best security practises. The tests were carried out under the mask of an attacker, but no harm was done to the application's functionality or operation.

SCOPE OF TESTING

Security assessment includes testing for security loopholes in the scope defined below. Apart from the following, no other information was provided.

Application: https://juice-shop.herokuapp.com/

SUMMARY OF RESULTS

For the Web Application Security Assessment, the graph below displays the amount of security holes for each severity level. A total of 28 vulnerabilities were discovered in this report should be patched as soon as possible.

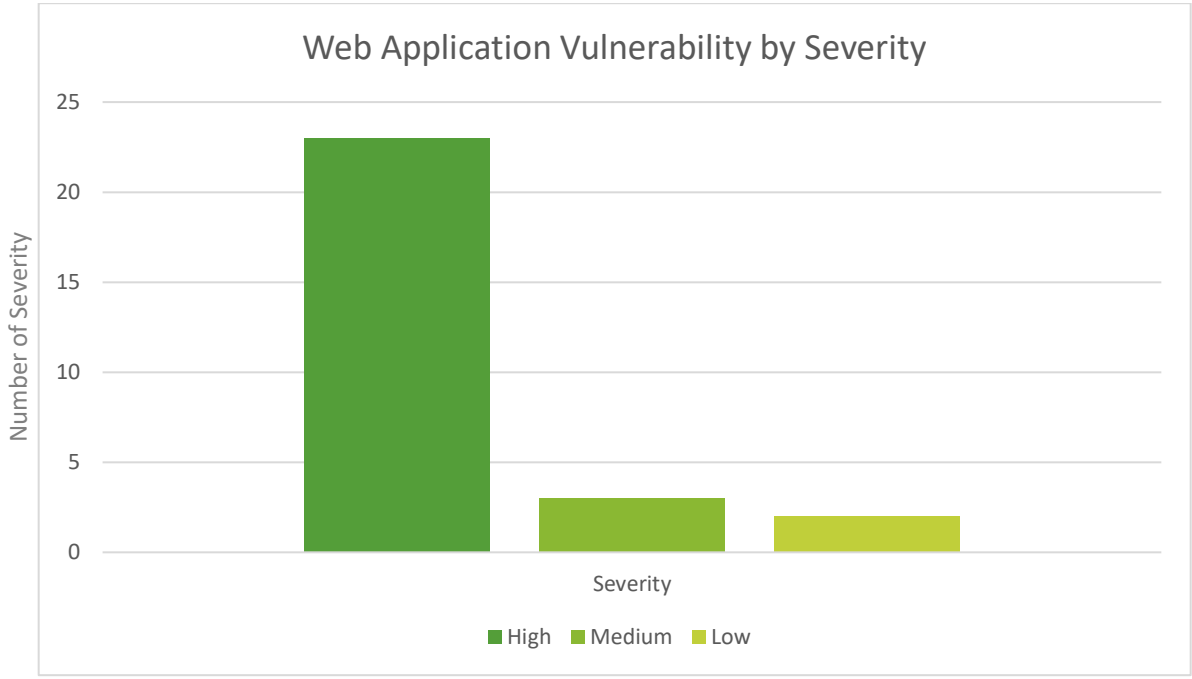


Figure 1 Web Application Vulnerability by Severity

The application was discovered to be vulnerable to various attacks using authentication techniques and installed authorisation restrictions, which might lead to unauthorised entry to the programme and sensitive information compromise. The programme is particularly prone to a variety of exploitable flaws as a consequence of insufficient input validation methods. The bulk of these problems are caused by Cross-Site Scripting flaws. Successful Cross Site Scripting attacks may result in the revelation of user data and the exploitation of the site to deceive people into visiting other dangerous sites, resulting in reputation harm, as well as financial costs. To address all of the highlighted concerns, the application should be re-engineered. It is clear that most of the developers in charge for the website have not embraced or followed a secure and uniform application development framework or standard.

INJECTION

Injection issues allow attackers to send malicious code to another machine. These attacks include calls to the operating system via system inquiries, the use of external apps via shell commands, and SQL calls to backend databases. Scripts written in Perl, Python, and other languages can be injected and executed in poorly built applications. Any software that uses an interpreter of any type is always at danger of developing an injection vulnerability. These exploits are simple to carry out, and more tools that look for these flaws are becoming available.

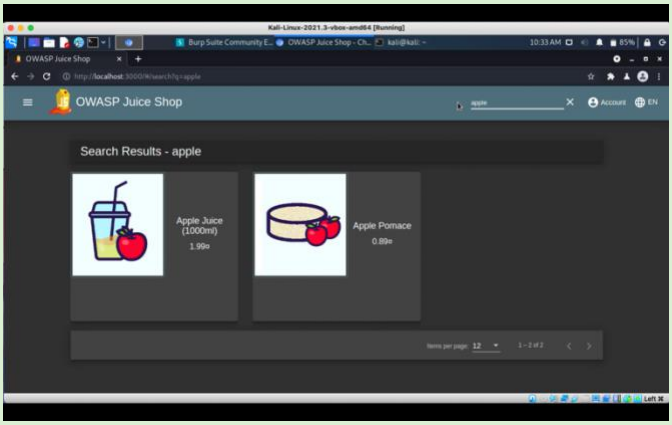
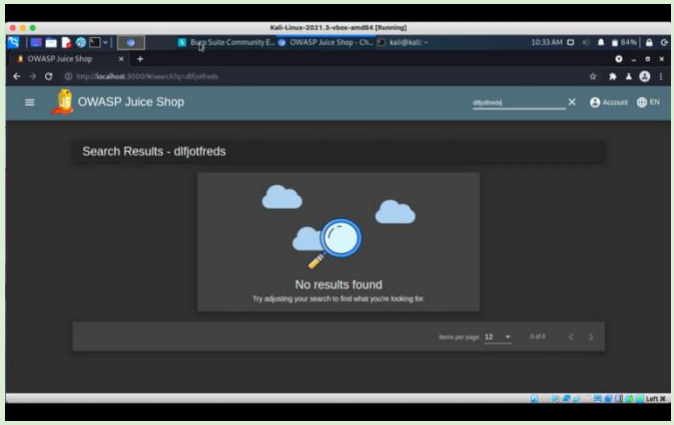
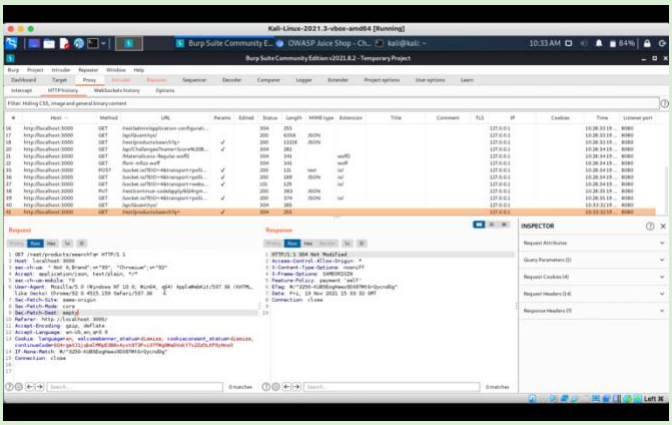
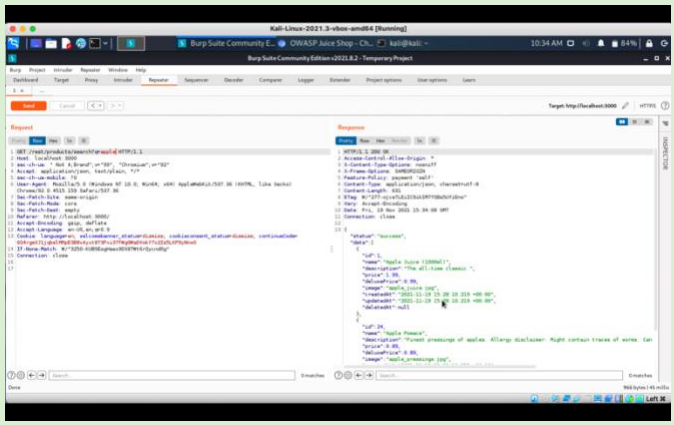
DATABASE SCHEMA

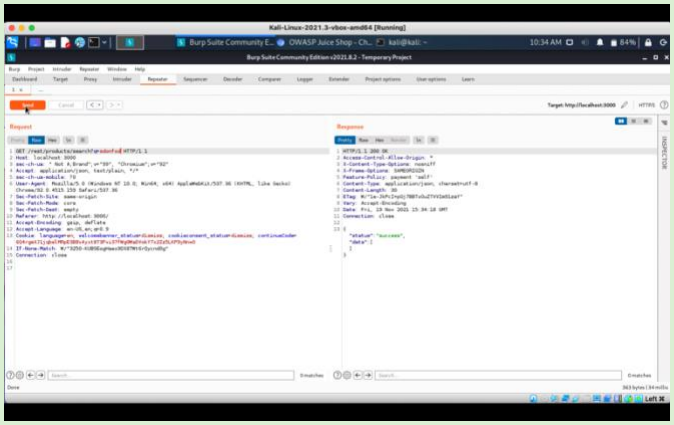
Description: Exfiltrate the database schema through SQL Injection.

Severity: High

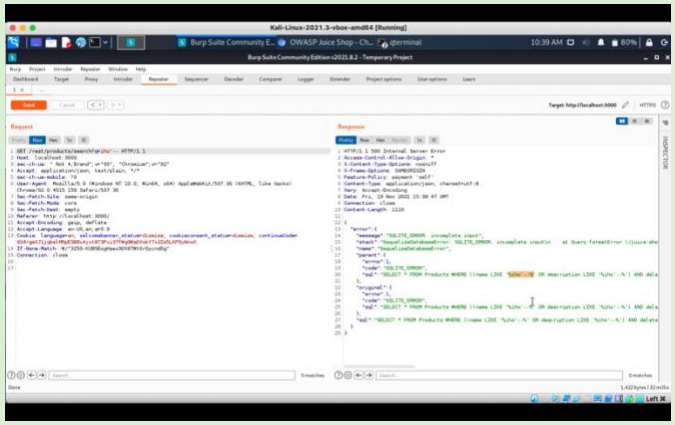
Steps to Reproduce:

Table 1 DB SCHEMA

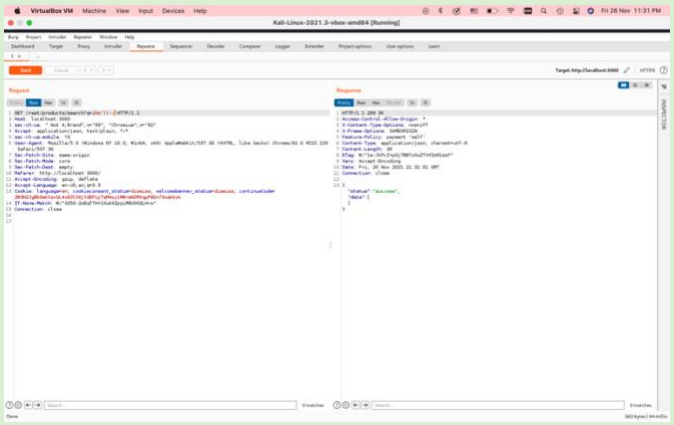
1: use the search function	2: search for something that is not found
	
3: check HTTP history for the request	**4: send the request to repeater**
	
5: searching for something that is not found returned a succesful status	**6: adding '— returned an error which showed tge DB type.**

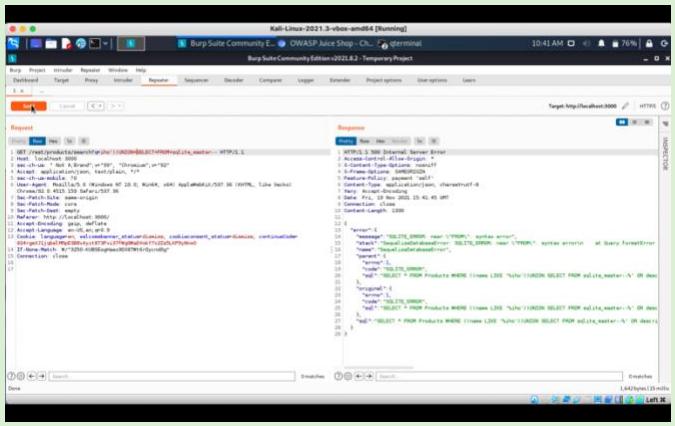


7: adding)) to close the brackets returned a successful response

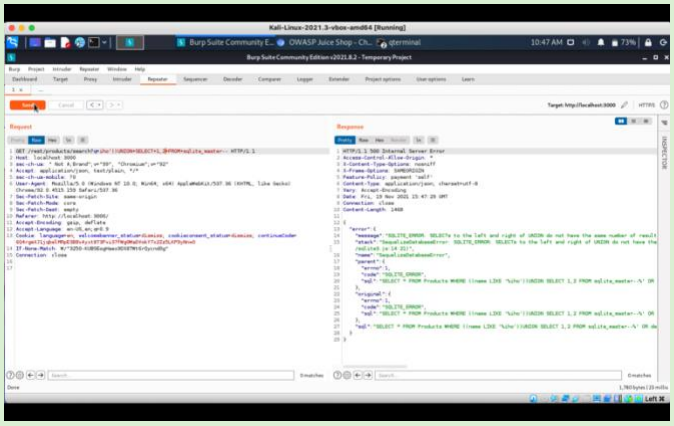


7: adding the union injection returned a columns-related error

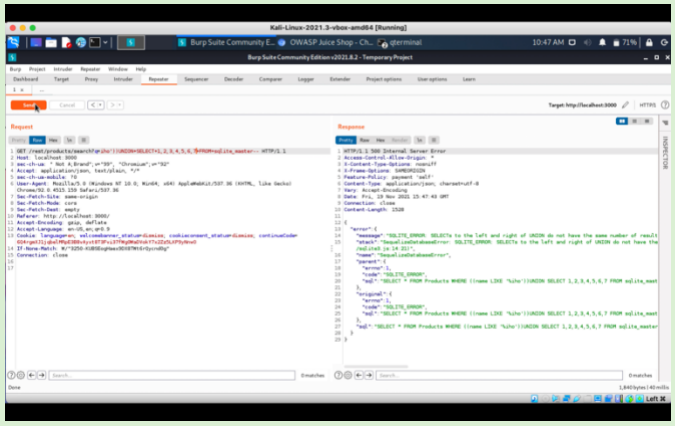




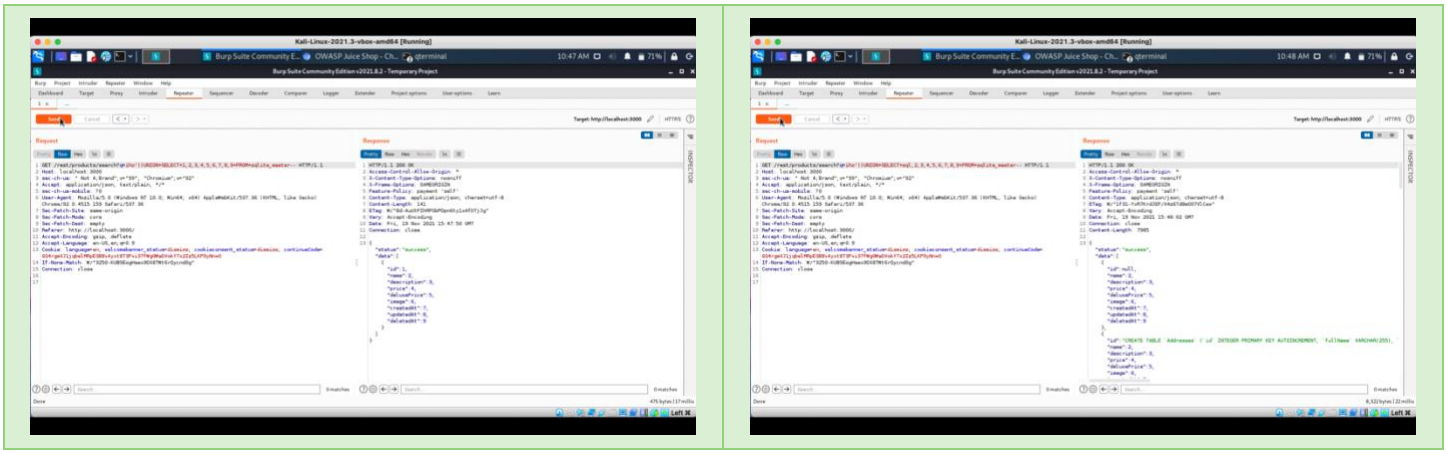
9: try numbers until you reach the right number of columns



10: turns out it has 9 columns



10: add sql instead of 1 to retrieve all the data



Expected Result:

a'))UNION+SELECT+sql,2,3,4,5,6,7,8,9+FROM+sqlite_master--

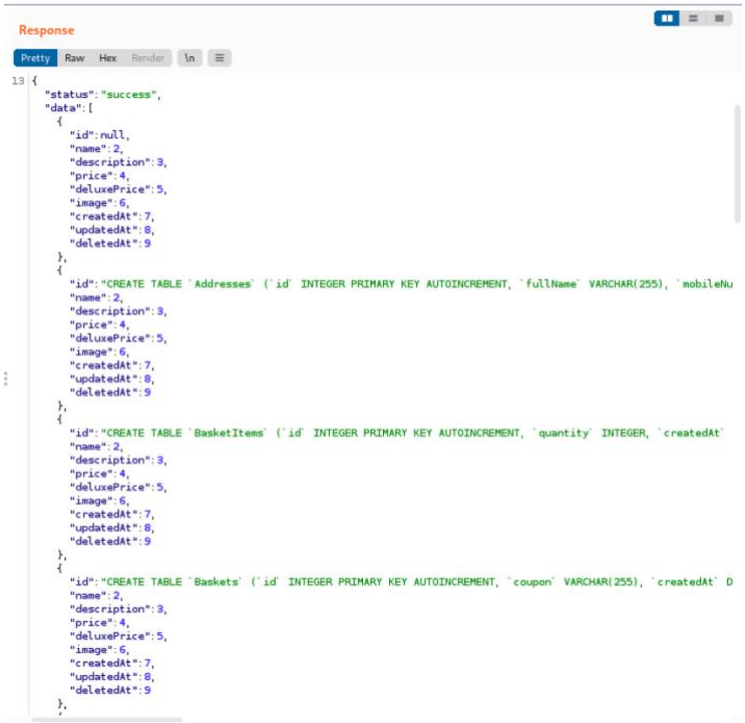


Figure 2 DB SCHEMA RESULT

USER CREDENTIALS

Description: Retrieve user credentials via SQL Injection.

Severity: High

Steps to Reproduce: After successfully retrieving all the tables, I was able to retrieve users' data.

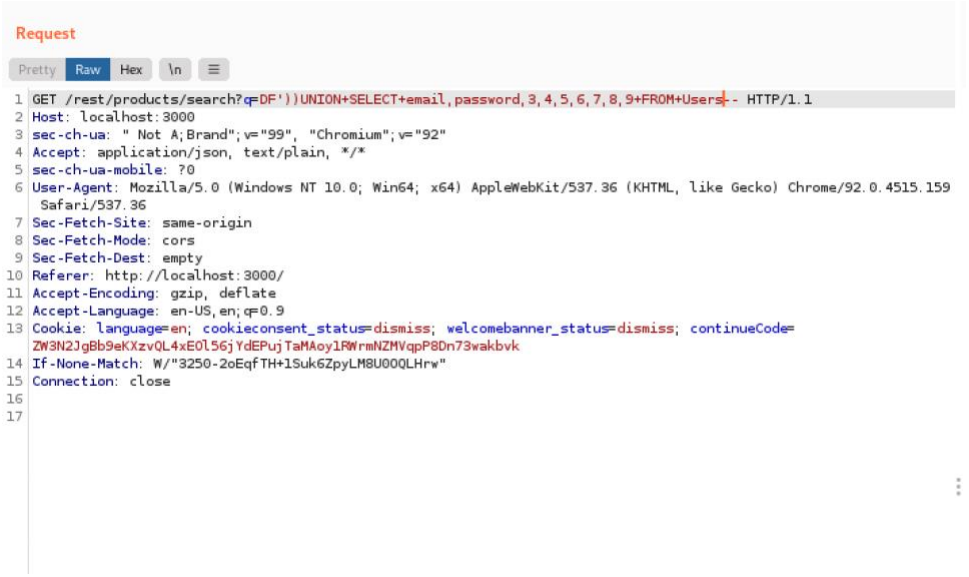


Figure 3 USER CREDENTIALS

Expected Result:

a'))UNION+SELECT+email,password,3,4,5,6,7,8,9+FROM+Users--

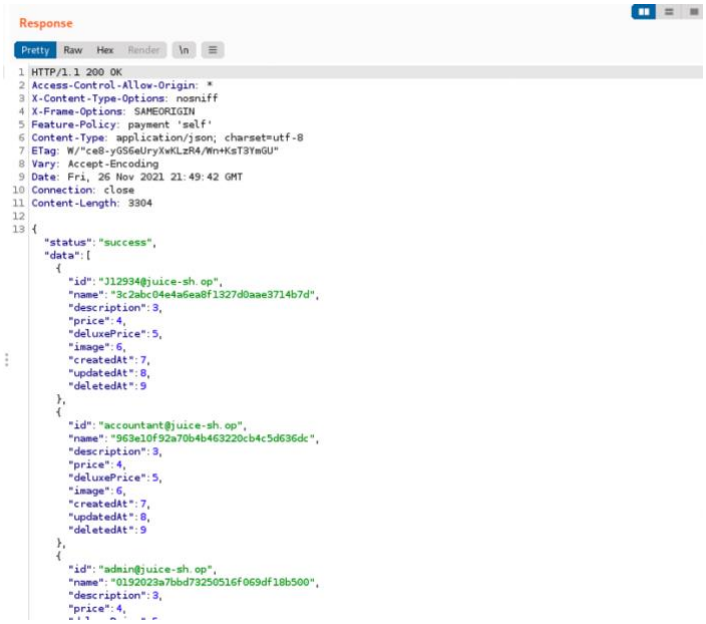


Figure 4 USER CREDENTIALS RESULT

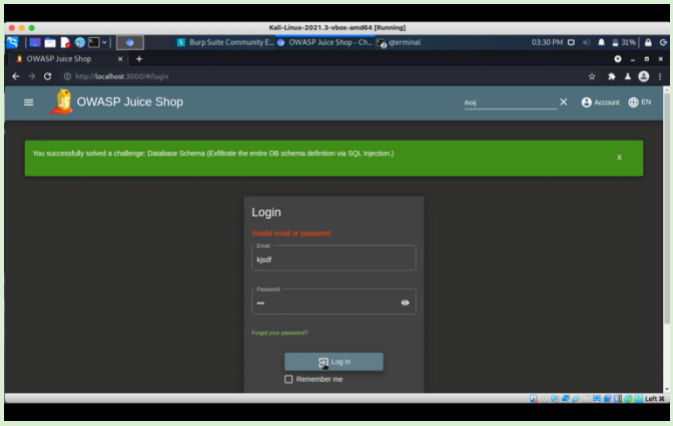
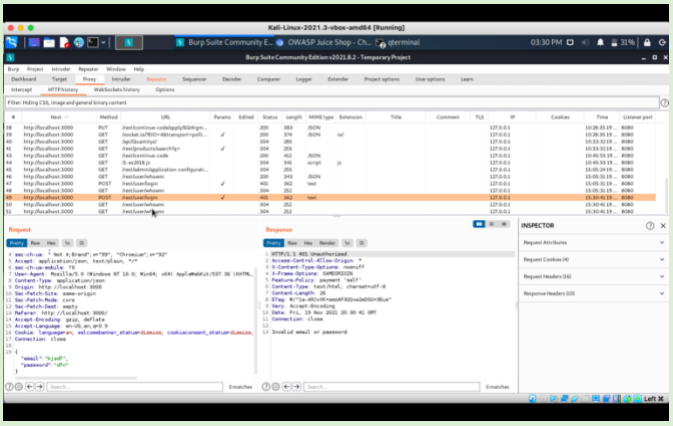
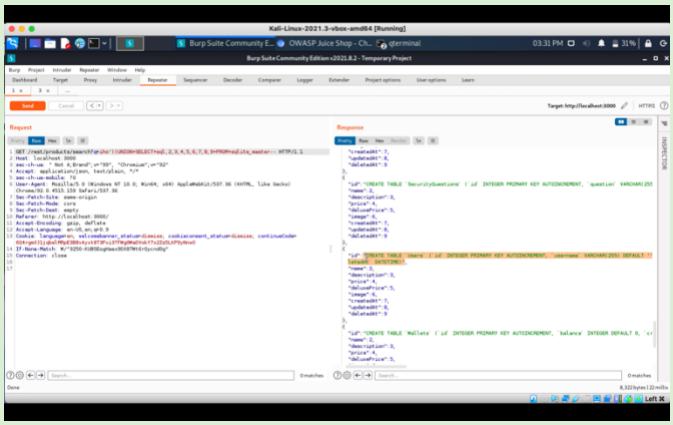
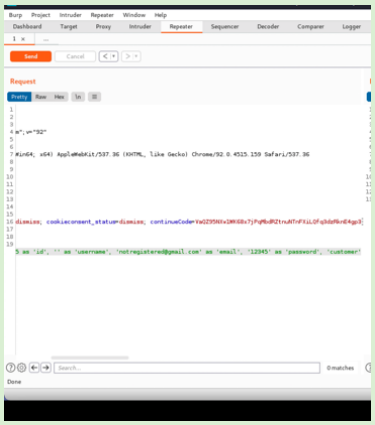
NON-EXISTING ACCOUNT

Description: Log in with a non-existing account, without registering that user.

Severity: high

Steps to Reproduce:

Table 2 NON-EXISTING ACCOUNT

1: login with any credentials	2: send the request to repeater
	
3: using the DB SQL injection vulnerability, copy the SQL statement to create a new user.	**4: paste it in the email field and edit it to register a new user**
	

Expected results:

' UNION SELECT * FROM (SELECT 15 as 'id', '' as 'username', notregistered@juice-sh.op' as 'email', '12345' as 'password', 'accounting' as 'role', '123' as 'deluxeToken', '1.2.3.4' as 'lastLoginIp' , 'default.svg' as 'profileImage', '' as 'totpSecret', 1 as 'isActive', '1999-08-16 14:14:41.644 +00:00' as 'createdAt', '1999-08-16 14:33:41.930 +00:00' as 'updatedAt', null as 'deletedAt')--

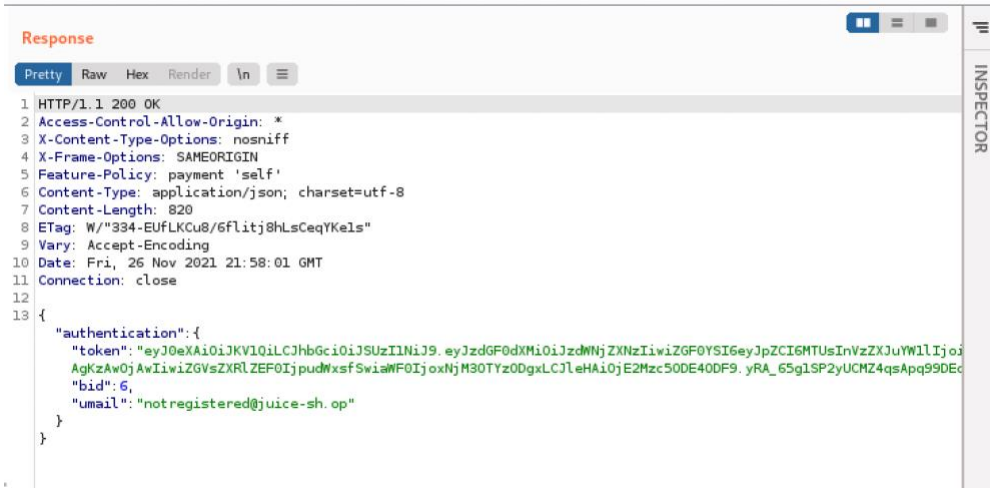


Figure 5 NON-EXISTING ACCOUNT RESULT

LOGIN ADMIN

Description: Log in the administrator's account.

Severity: High

Steps to Reproduce: type admin' or 1=1-- in the email field

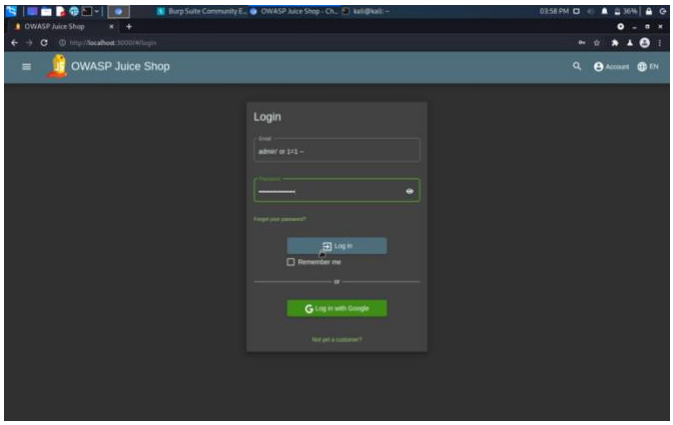


Figure 6 LOGIN ADMIN

Expected Result:

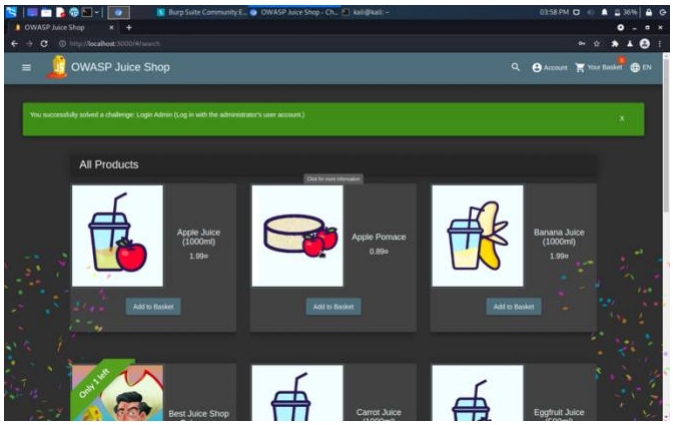


Figure 7 LOGIN ADMIN RESULT

LOGIN BENDER

Description: Log in Bender's account.

Severity: High

Steps to Reproduce: type bender@juice-sh.op' or 1=1-- in the email field

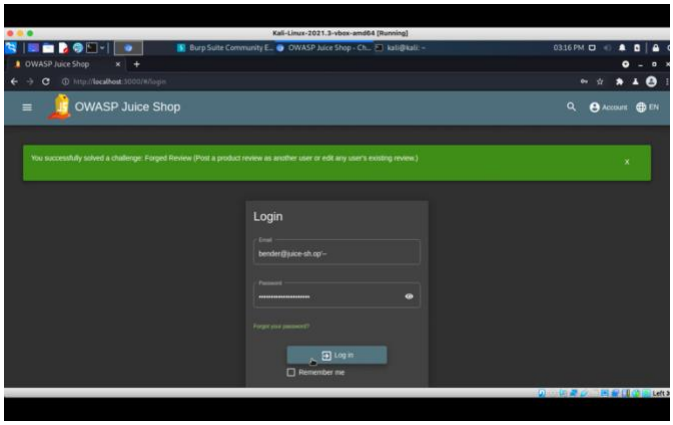


Figure 8 LOGIN BENDER

Expected result:

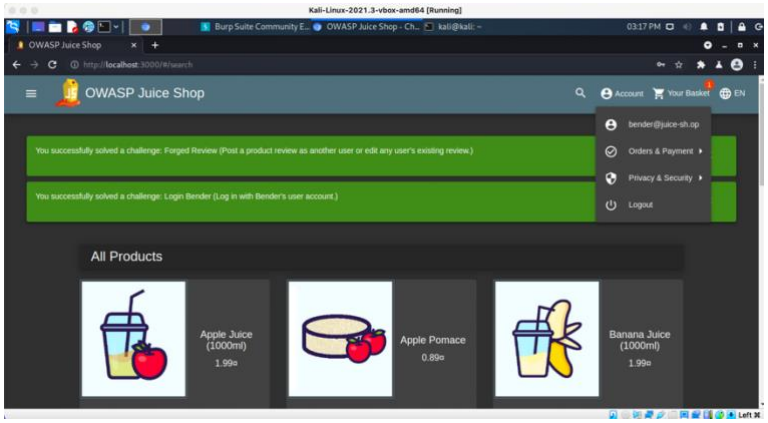


Figure 9 LOGIN BENDER RESULT

MITIGATION

- **Sanitization**
    - To avoid the injection of dangerous code into the system, examine, clean, and filter data inputs from users, APIs, and web services for any undesired characters and strings.
- **Use Parameterized Queries**
    - Fill in the arguments with placeholders and provide the values during runtime.
- **stored procedure**
    - Provide SQL queries with parameters that are automatically parameterized. The distinction between prepared statements and stored procedures is that stored procedures have SQL code defined in the database before being invoked from the application.
- **Whitelist Input Validation**
    - Before allowing the input to be processed further, make sure it complies with the desired type, size, range, or other format criteria.
- **Enforcing Least Privilege**
    - Limit the rights provided to each database account in your system to minimize the potential impact of a successful SQL injection exploit.

BROKEN AUTHENTICATION

Broken authentication is a security term that attackers use to spoof real people on the internet. It has to do with problems in session management and credential management. Because attackers employ hijacked session IDs or stolen login details to impersonate a user, both are labeled failed authentication.

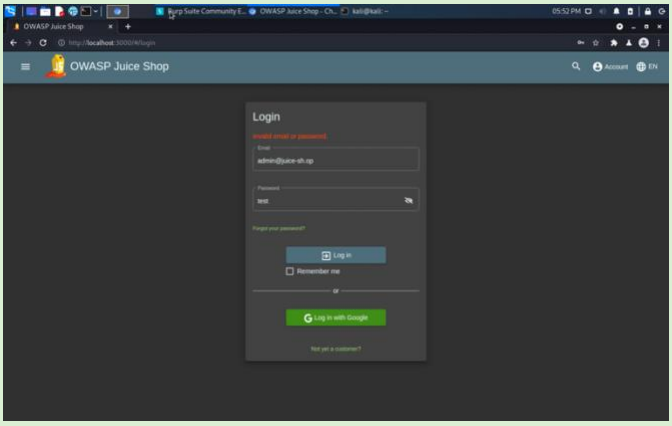
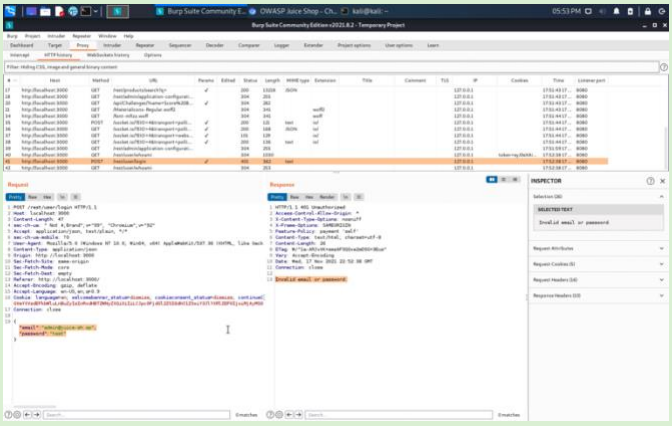
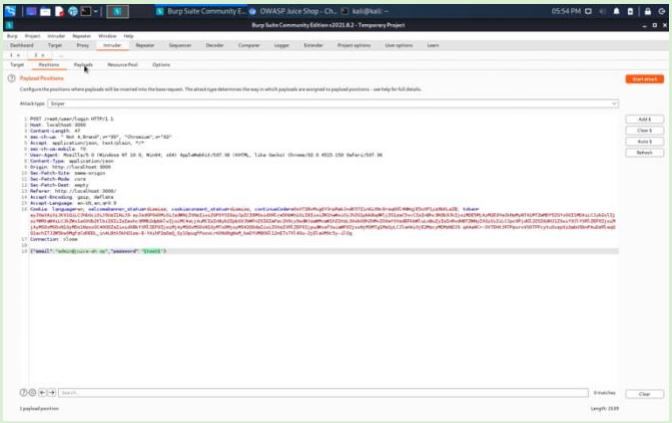
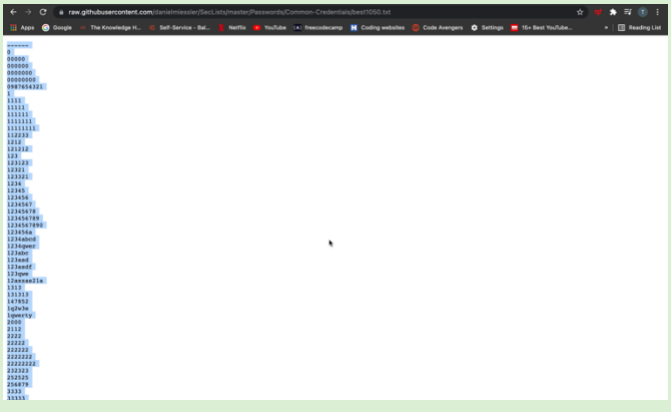
PASSWORD STRENGTH

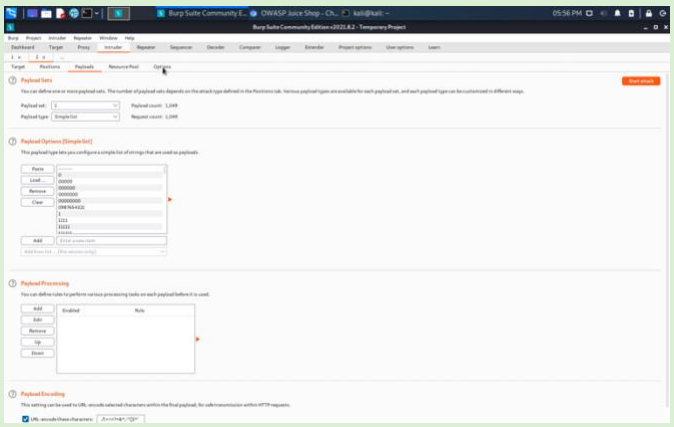
Description: brute force the administrator's password

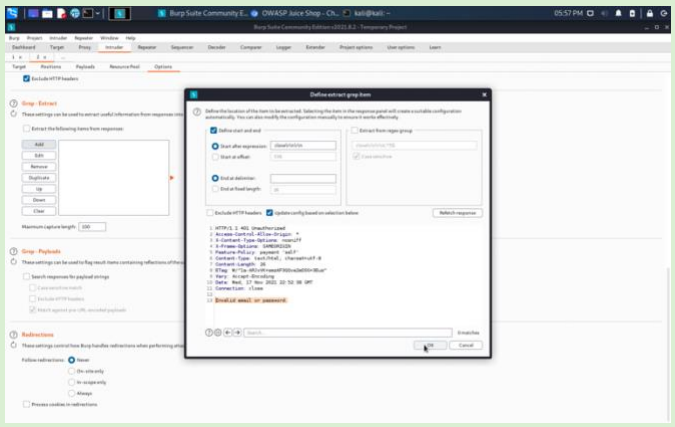
Severity: High

Steps to Reproduce:

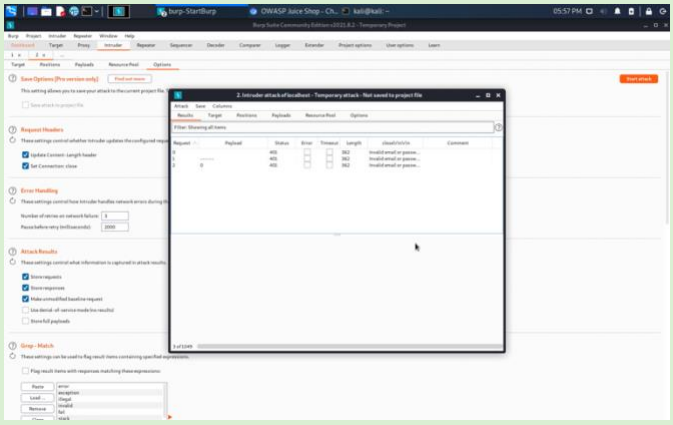
Table 3 PASSWORD STRENGTH

1: login with any credentials	2: check HTTP history for the request
	
3: send it to intruder and mark the password string as the payload	**4: copy a password payload**
	
5: paste the payload in burpsuite	**6: grep the error message**



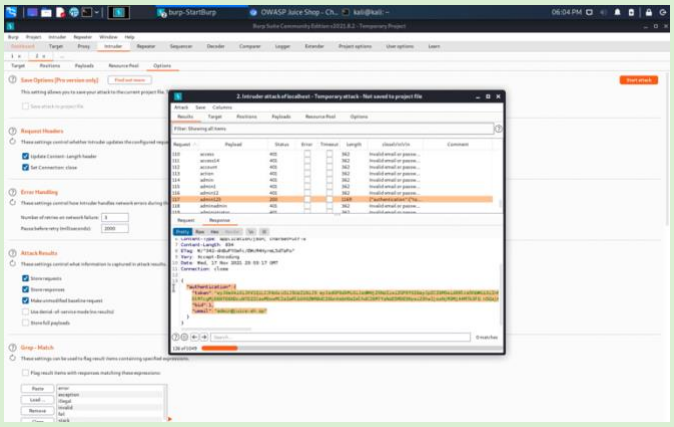
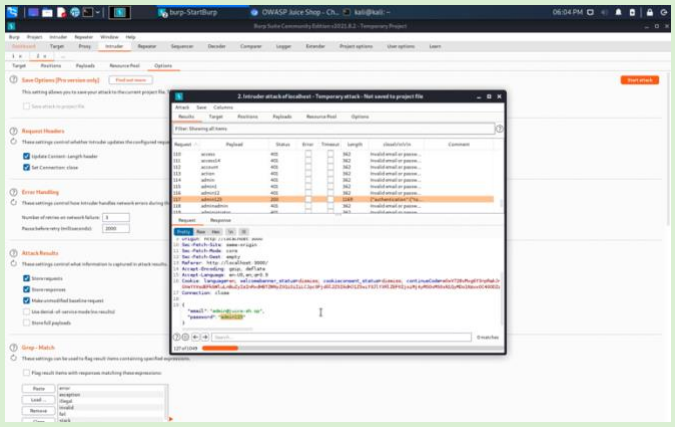


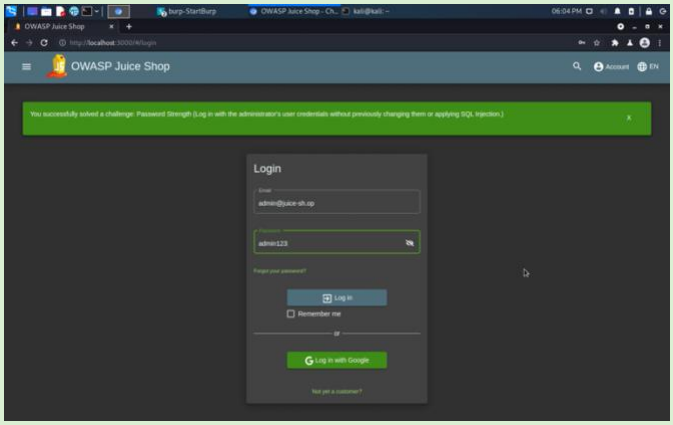
7: start attack

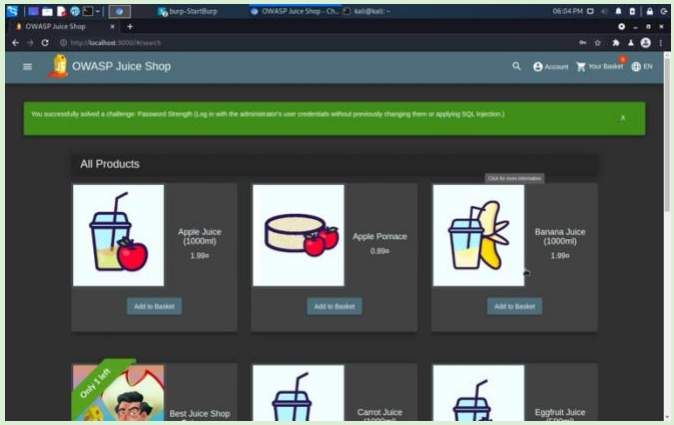


Expected result:

Table 4 PASSWORD STRENGTH RESULT

The successful status	Check the response for the password
	
Login with the credentials	





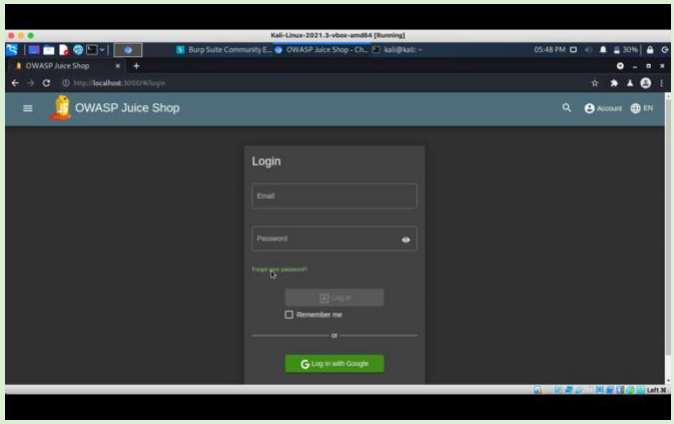
RESET JIM'S PASSWORD

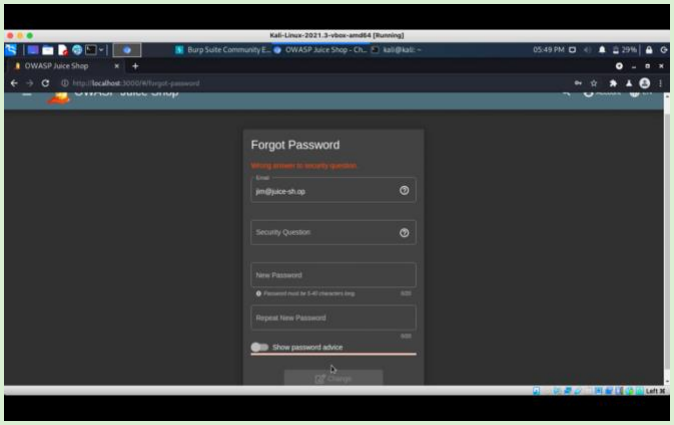
Description: Reset someone's password via Forgot Password.

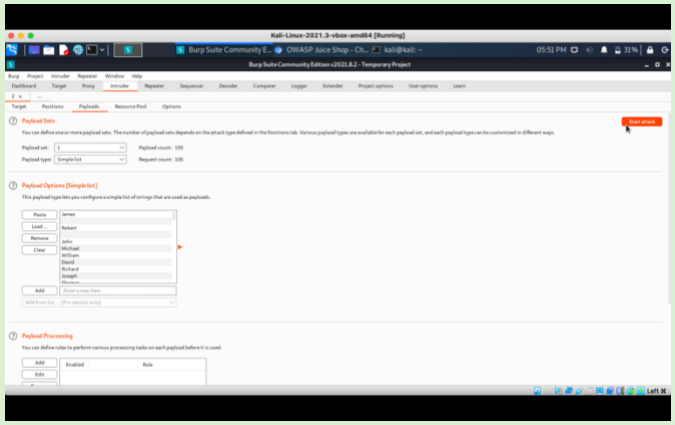
Severity: High

Steps to Reproduce:

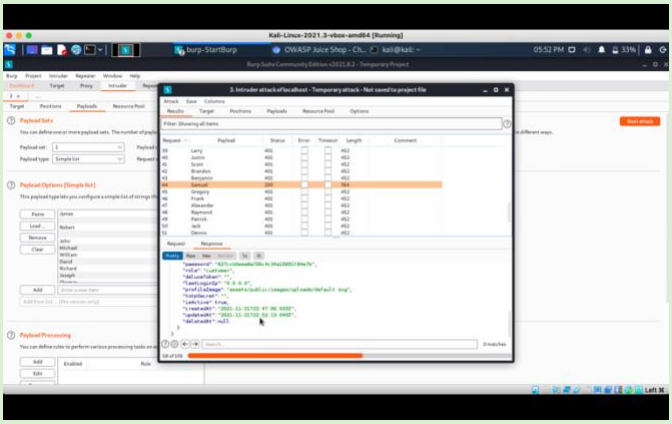
Table 5 RESET JIM'S PASSWORD

1: look for an email to change its password	2: press on forgot password
	
3: fill in the form and submit	**4: send the request to intruder and paste a payload of common male names**



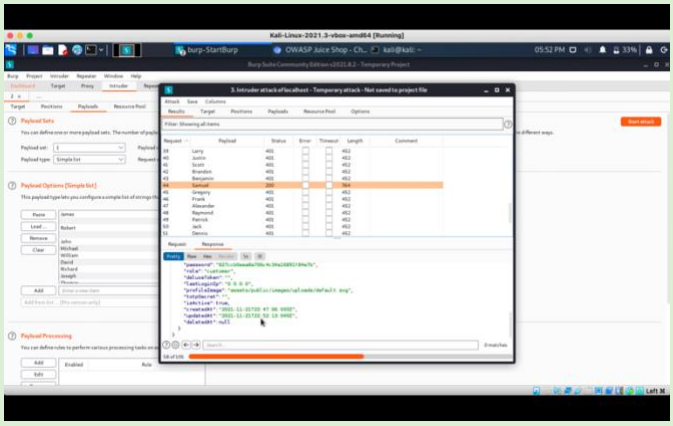
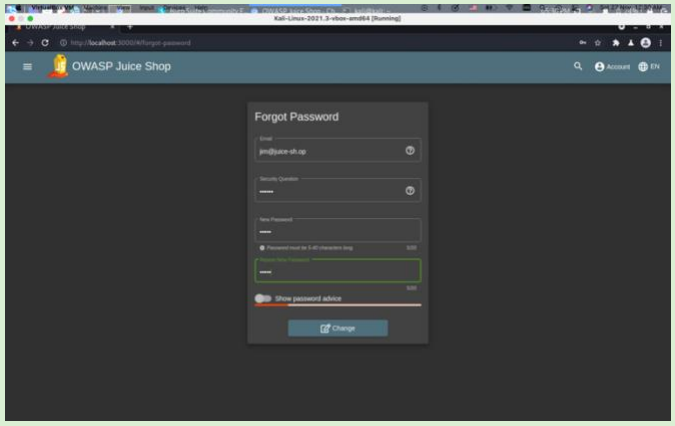


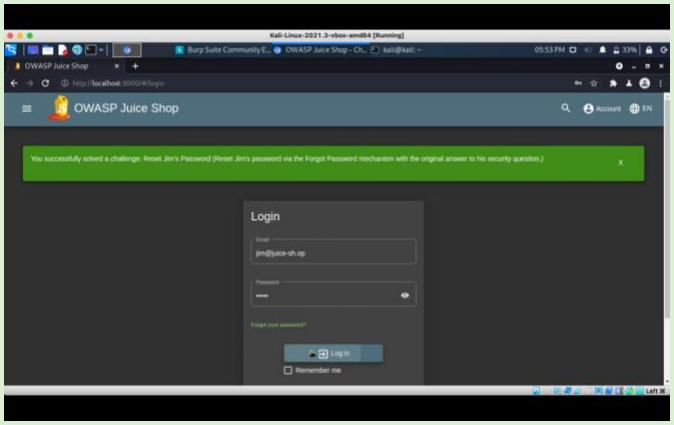
5: start attack

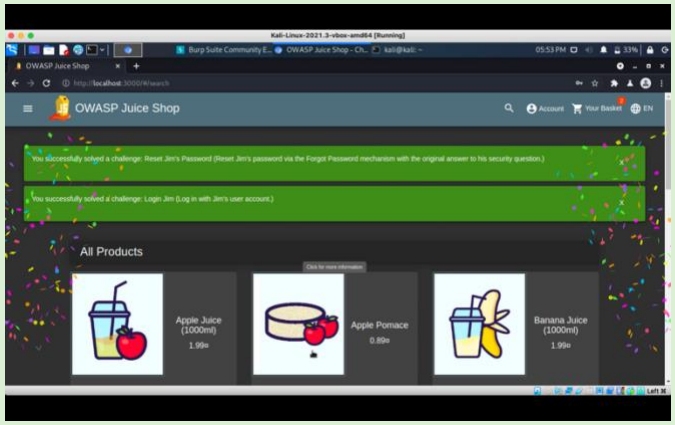


Expected result:

Table 6 RESET JIM'S PASSWORD RESULT

Samuel is the answer to the security question	Change the password
	
Login with new credentials	





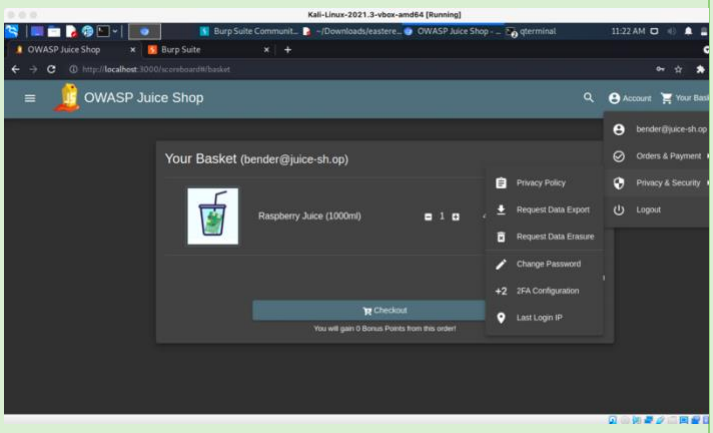
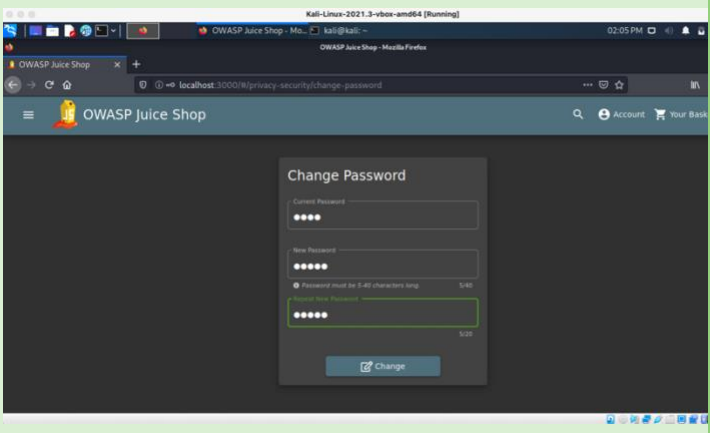
CHANGE BENDER'S PASSWORD

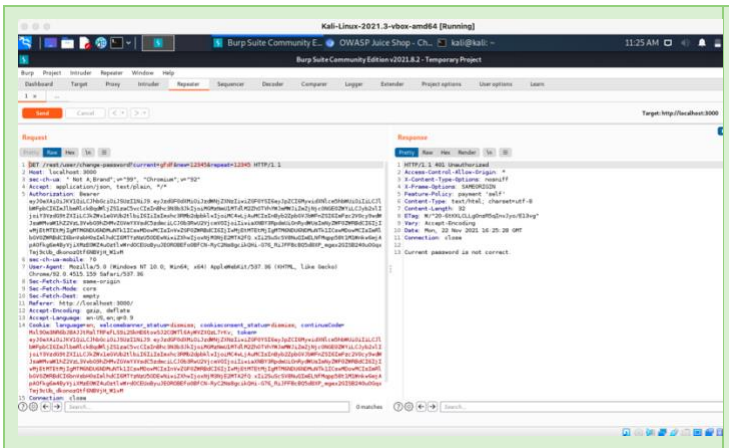
Description: Reset Bender's password without using reset Password.

Severity: High

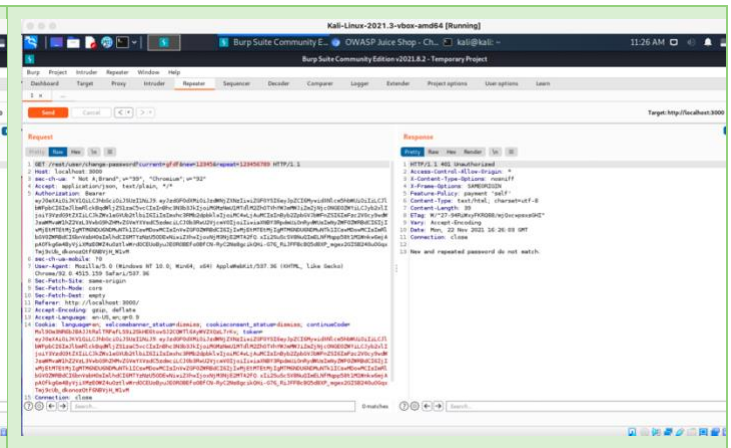
Steps to Reproduce:

Table 7 CHANGE BENDER'S PASSWORD

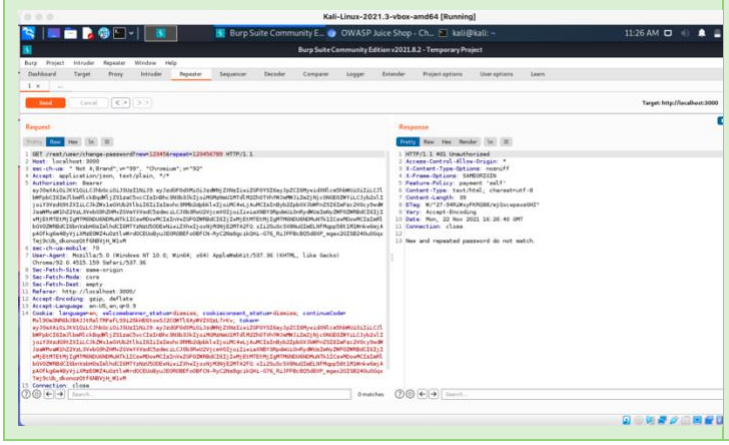
1: login bender's account	2: try changing the password
	
3: send the request to repeater	**4: changing the inputs showed an error**

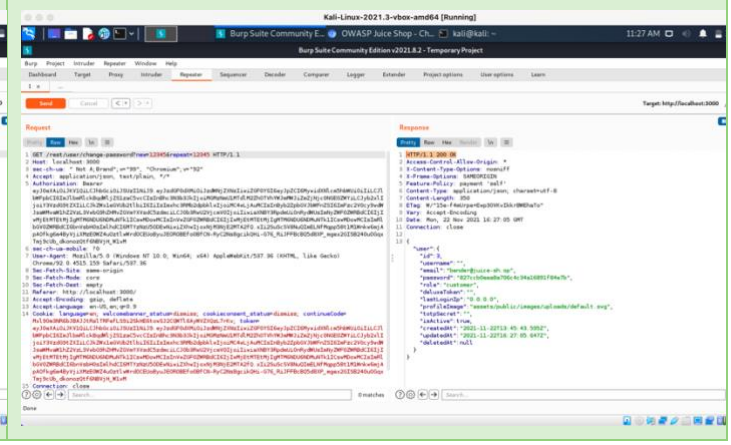


5: removing current input showed the same error



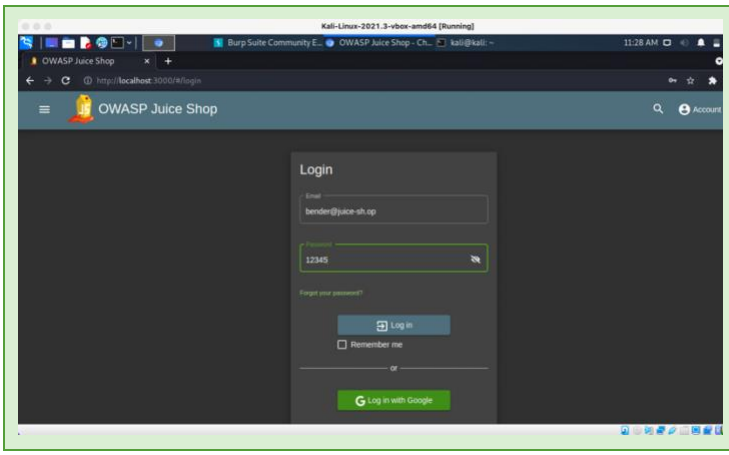
6: match the passwords showed a successful message

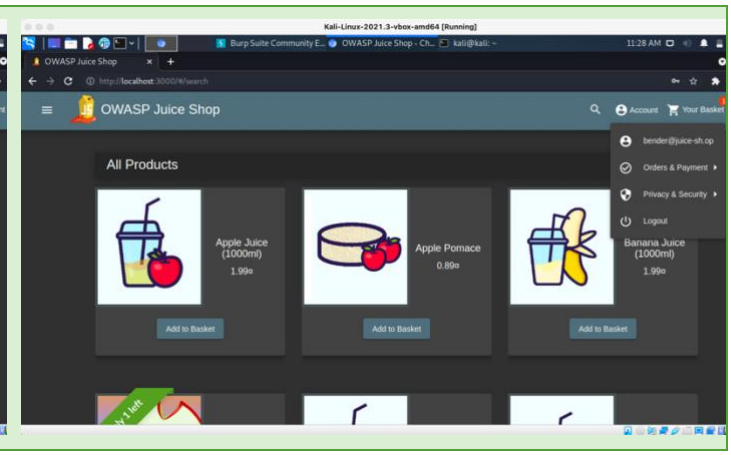




Expected Result:

Table 8 CHANGE BENDER'S PASSWORD RESULT





MITIGATION

- Brute-Force Protection
  - Restrict the number of times a given IP address may log in.
- Anomaly detection

- Discover unusual things, occurrences, or observations that stand out from the rest of the data and raise doubts.

- Multifactor Authentication
  - To obtain access to an account, the user must supply two or more verification criteria.

CROSS SITE SCRIPTING

XSS attacks occur when an attacker utilises a web application to transmit malicious script to a separate end user. The weaknesses that allow these exploits to succeed are prevalent, and may be identified anywhere a web application receives user input without validating or encoding it in its output. Because it believes the script came from a reliable source, the malicious script can retrieve any cookies, authentication information, or other confidential material stored by the browser and utilised with that site.

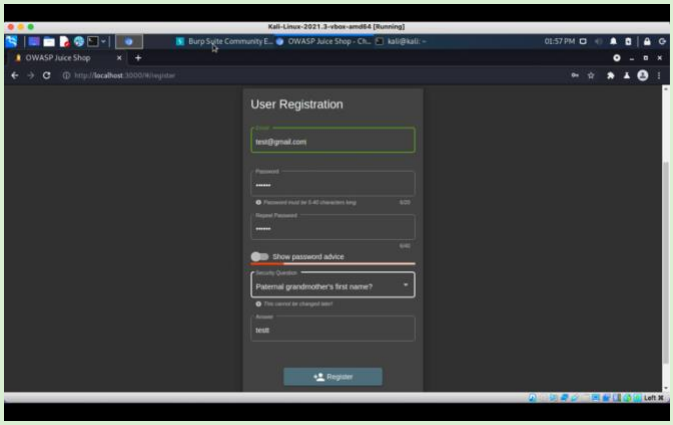
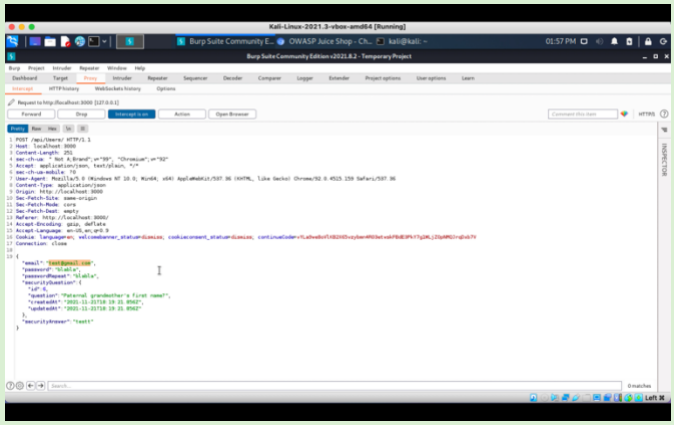
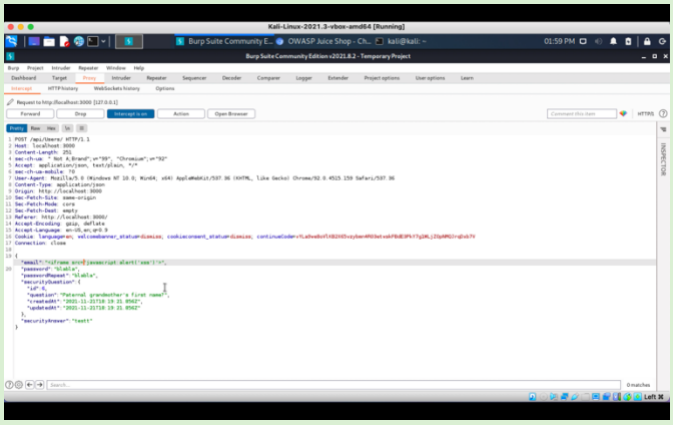
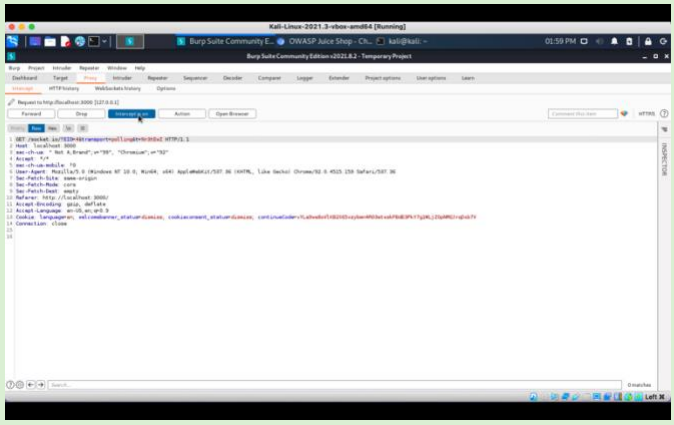
CLIENT-SIDE XSS PROTECTION

Description: Perform a XSS attack with <iframe src="javascript:alert(`xss`)"> bypassing a *client-side* security mechanism.

Severity: High

Steps to Reproduce:

Table 9 CLIENT-SIDE XSS

1: register as a new user	2: change the email address in the interceptor
	
3: email address changed to an XSS script	**4: forward it**
	

Expected result:

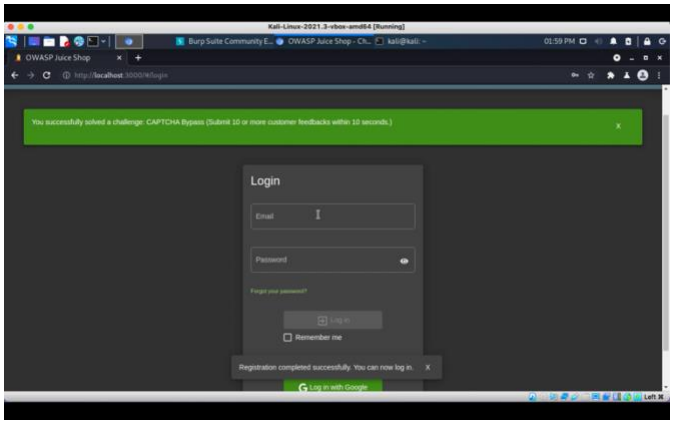
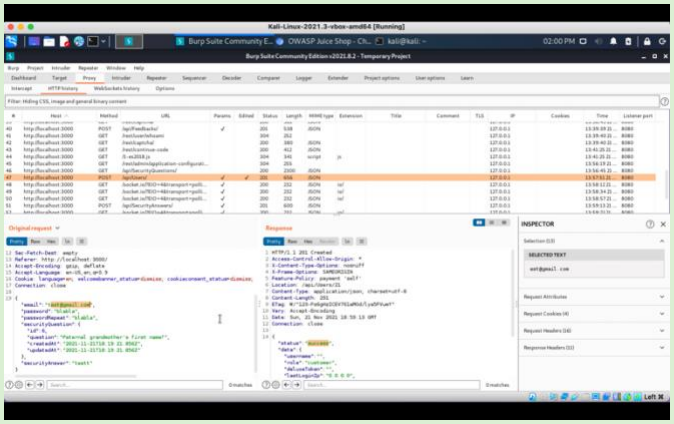
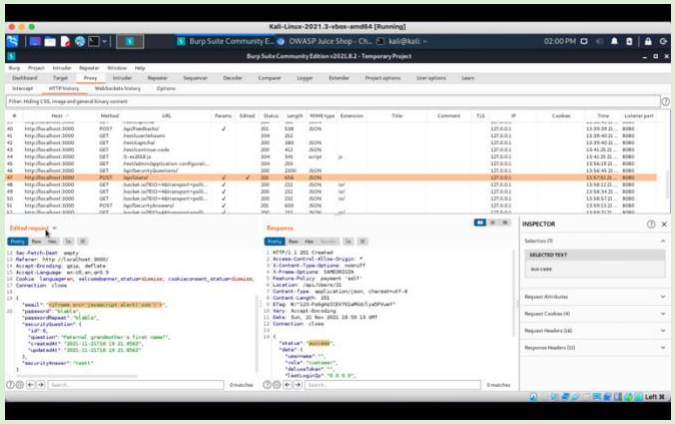


Figure 10 CLIENT-SIDE XSS RESULT

Table 10 REQUESTS

Original request	Edited request
	

DOM XSS

FIRST DOM XSS

Description: Perform a *DOM* XSS attack with <iframe src="javascript:alert(`xss`)">.

Severity: High

Steps to Reproduce: enter <iframe src="javascript:alert('Hi')"> in the search input field

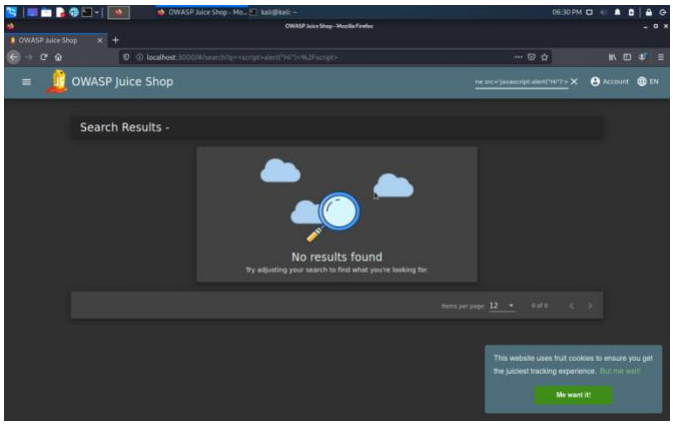
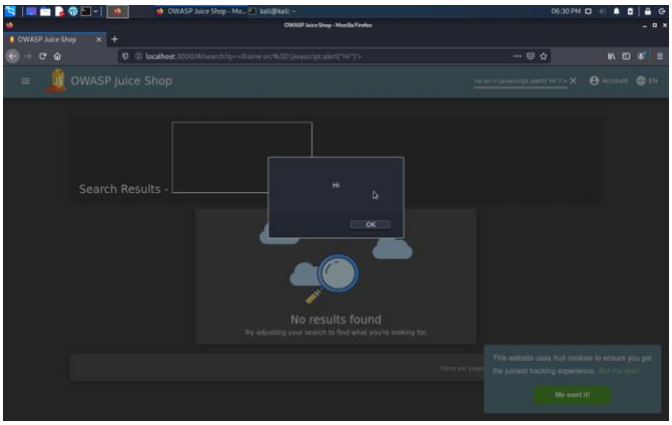
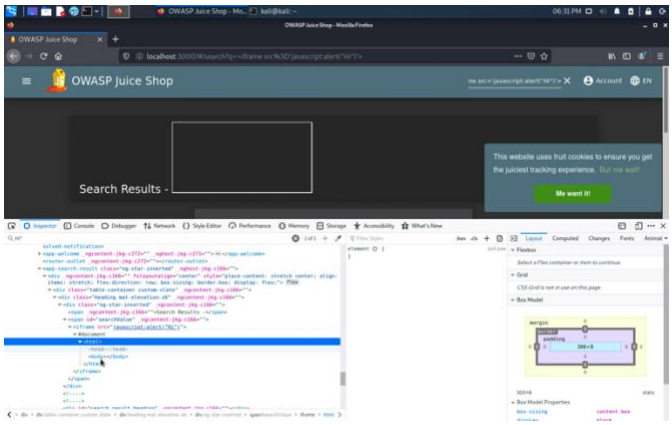


Figure 11 DOM XSS

Expected result:

Figure 12 DOM XSS RESULTS





SECOND DOM XSS

Description: Use the payload <iframe width="100%" height="166" scrolling="no" frameborder="no" allow="autoplay" src="https://w.soundcloud.com/player/?url=https%3A//api.soundcloud.com/tracks/771984076&color=%23ff5500&auto_play=true&hide_related=false&show_comments=true&show_user=true&show_reposts=false&show_teaser=true"></iframe> .

Severity: High

Steps to Reproduce: enter <iframe width="100%" height="166" scrolling="no" frameborder="no" allow="autoplay" src="https://w.soundcloud.com/player/?url=https%3A//api.soundcloud.com/tracks/771984076&color=%23ff5500&auto_play=true&hide_related=false&show_comments=true&show_user=true&show_reposts=false&show_teaser=true"></iframe> in the search input field

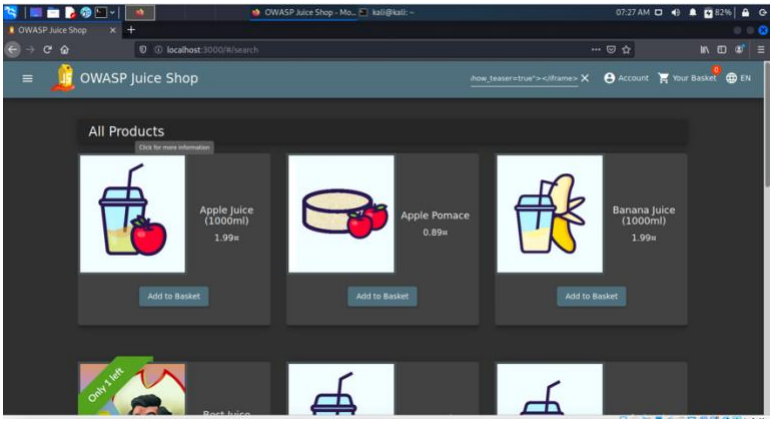


Figure 13 Soundcloud XSS

Expected result:

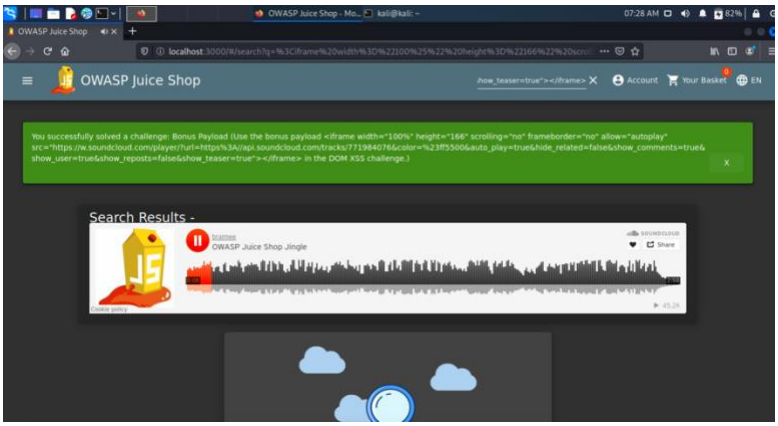


Figure 14 Soundcloud XSS

MITIGATION

- Filter input as soon as it arrives.
  - As much as feasible, filter based on legitimate input.
- On the output, encode the data.
  - To avoid the output from being misinterpreted as active material, encode it.
- Make use of the proper response headers.
  - To guarantee that browsers perceive your replies correctly, utilise the Content-Type and X-Content-Type-Options headers.

IMPROPER INPUT VALIDATION

An intruder can create input in a way that the remainder of the programme does not expect when software fails to adequately validate it. For that reason, system components might accept unwanted input, leading in a modification in control flow or the execution of arbitrary code.

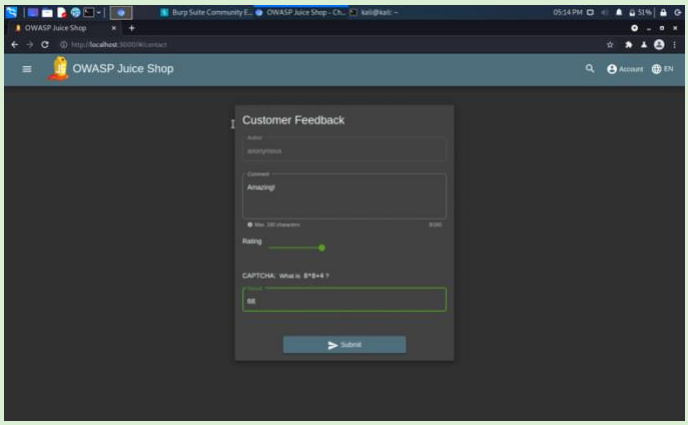
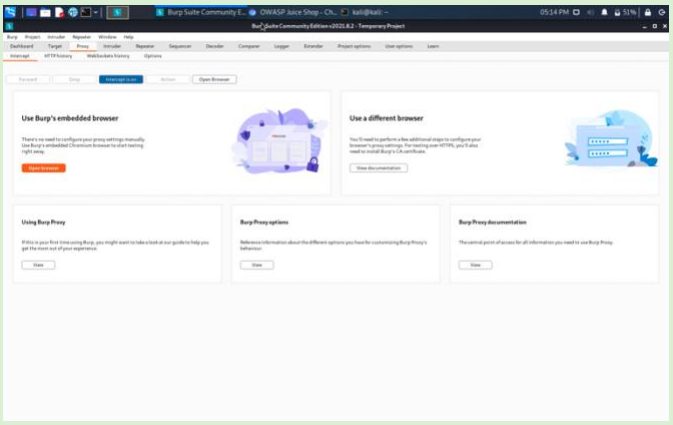
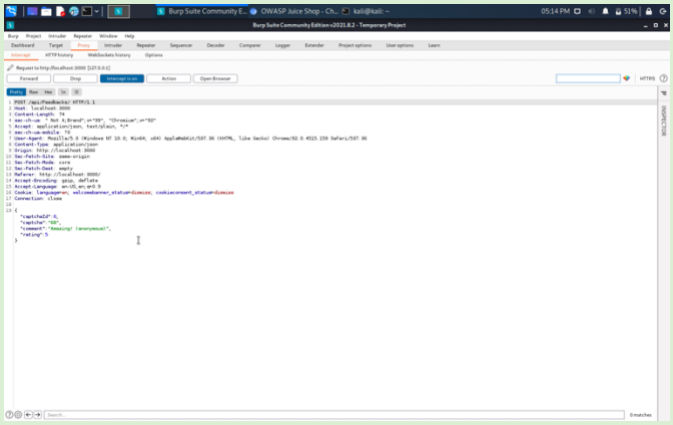
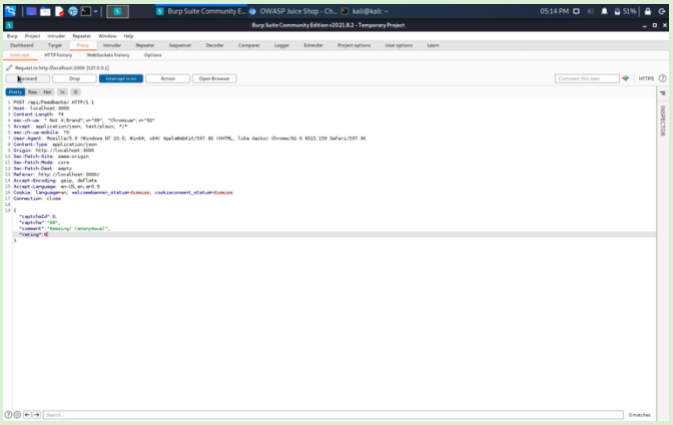
FEEDBACK TAMPERING

Description: Give a zero-star feedback to the store.

Severity: medium

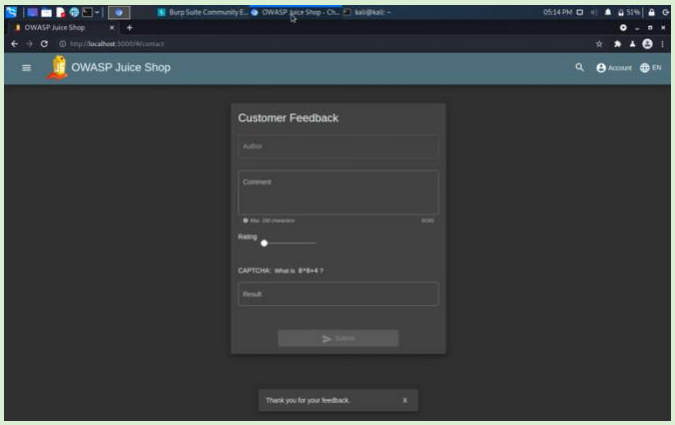
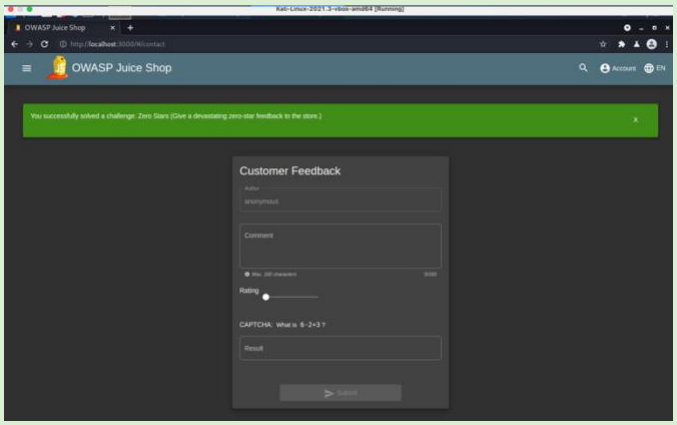
Steps to Reproduce:

Table 11 FEEDBACK TAMPERING

Step 1: fill in the feedback form	Step 2: turn on intercepter
	
Step 3: submit the form	**Step 4: change rating 5 to 0**
	

Expected result:

Table 12 CONFIDENTIAL DOCUMENT RESULT

Thank you message printed	
	

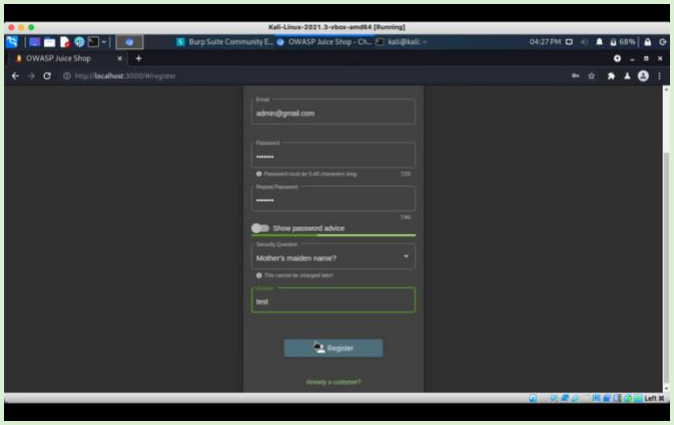
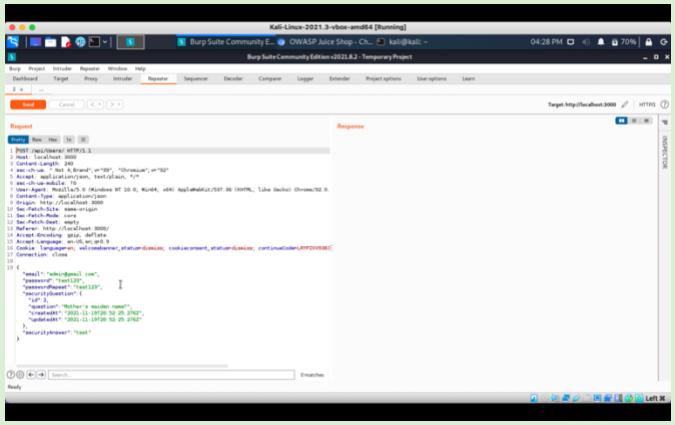
ADMIN REGISTRATION

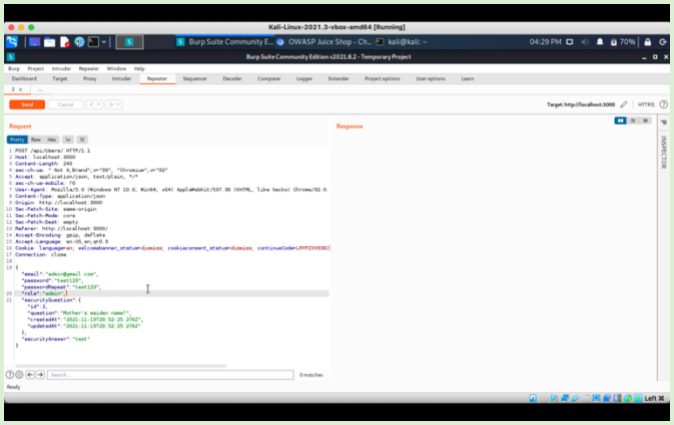
Description: Register as a user with administrator privileges.

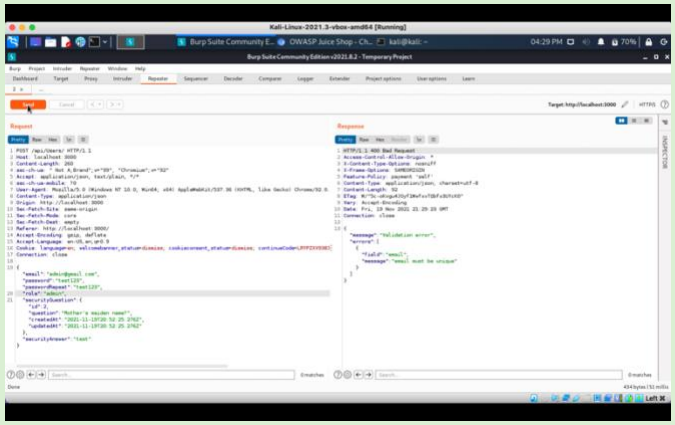
Severity: High

Steps to Reproduce:

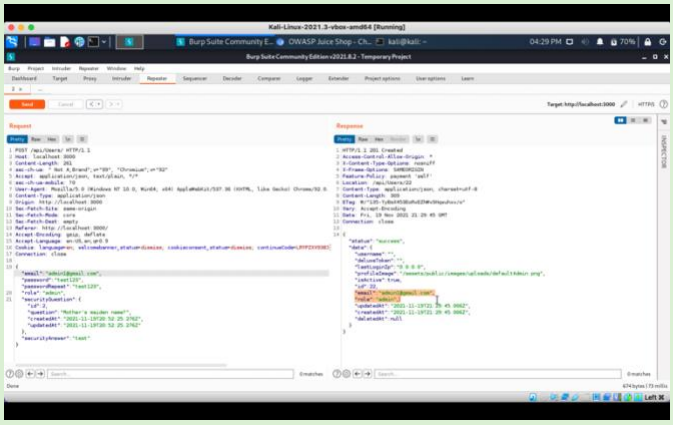
Table 13 ADMIN REGISTRATION

1: register as a normal user	2: send the request to repeater
	
4: add role as an admin	**Received an error**





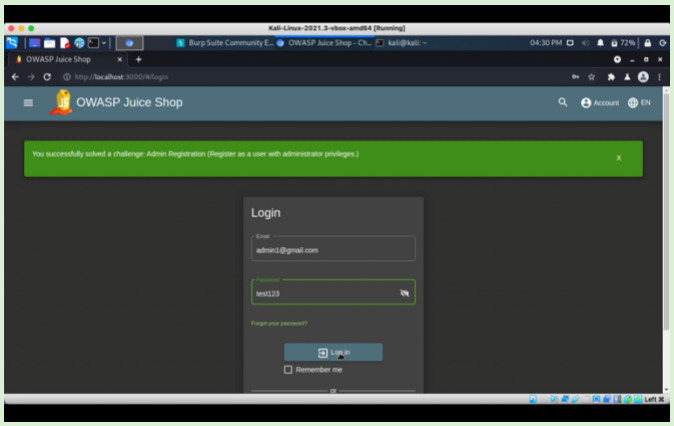
5: changed the email

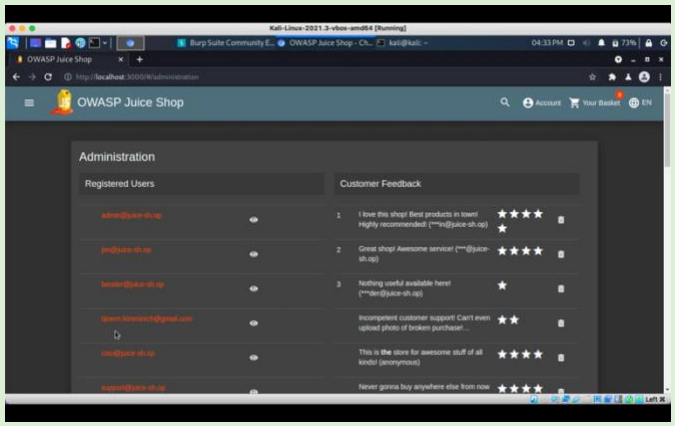


Expected results:

Table 14 ADMIN REGISTRATION RESULT

login with the new user as an admin	Successful!





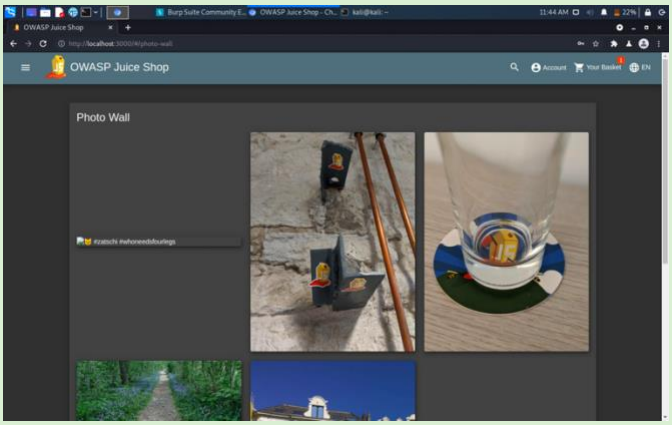
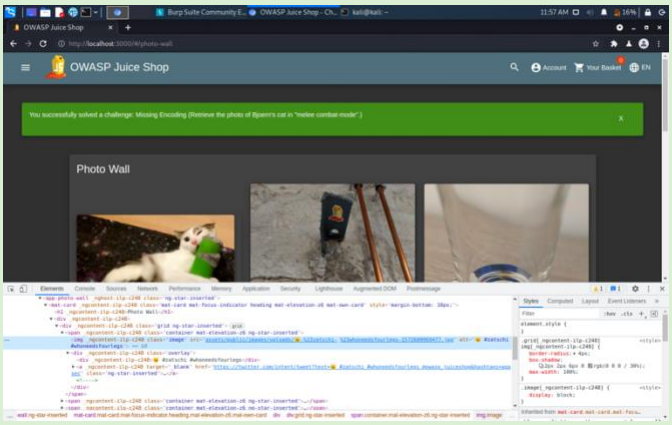
MISSING ENCODING

Description: Retrieve the photo of Bjoern's cat in "melee combat-mode".

Severity: Low

Steps to Reproduce:

Table 15 MISSING ENCODING

1: go to the photo wall	2: encode the src of the image (change the # to %26)
	

Expected result:

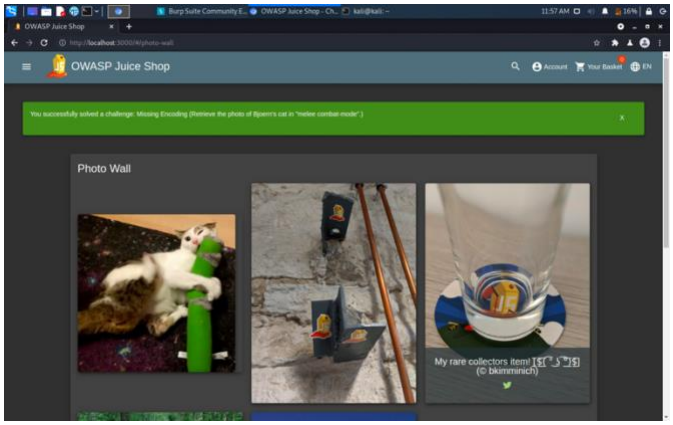


Figure 15 MISSING ENCODING RESULT

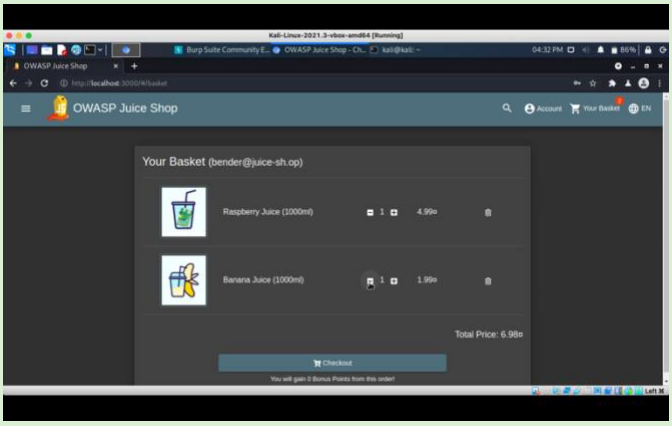
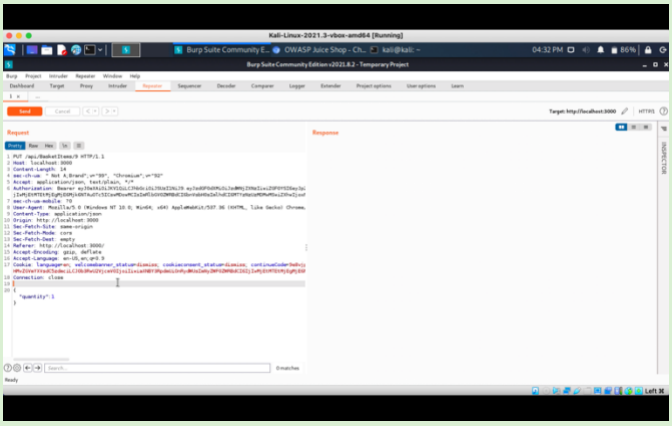
NEGATIVE PRICE

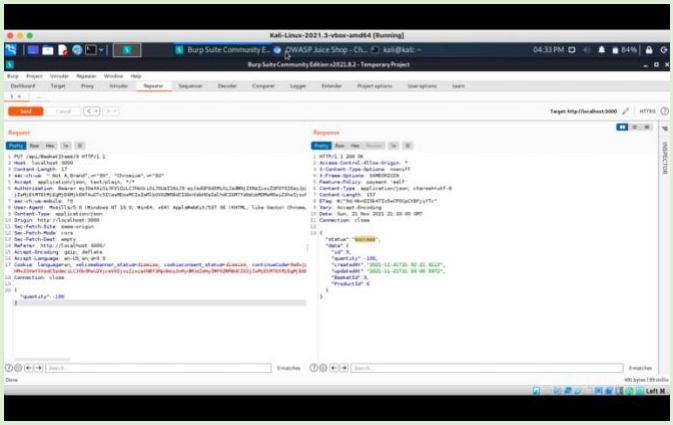
Description: Change an order price to a negative number.

Severity: High

Steps to Reproduce:

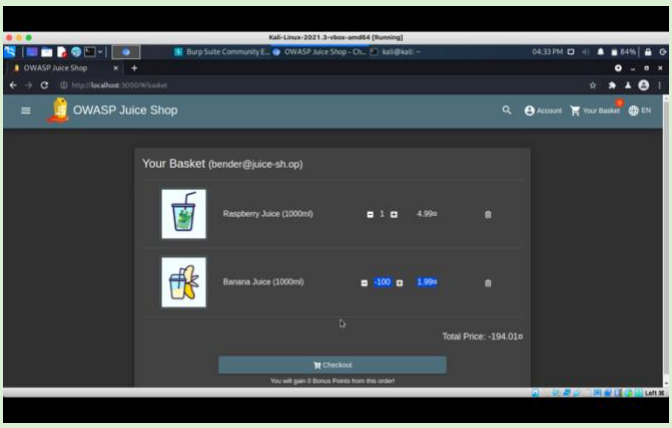
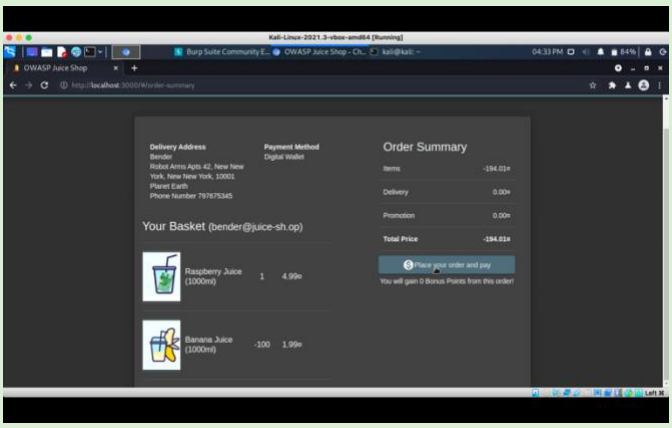
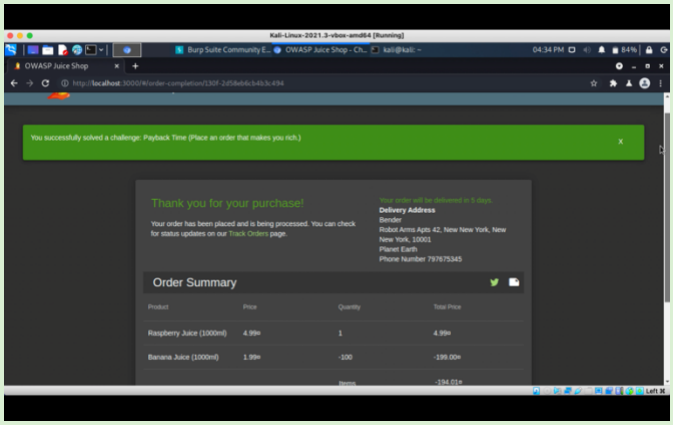
Table 16 NEGATIVE PRICE

1: trying the minus sign to reach a negative number will fail	2: send the request to repeater
	

3: change the quantity to a negative number


Expected result:

Table 17 NEGATIVE PRICE RESULT

successfully changed	Proceed to checkout
	
Successful	
	

REPETITIVE REGISTRATION

Description: Rregister a new user with non-matching passwords.

Severity: High

Steps to Reproduce: try registering with 2 different passwords

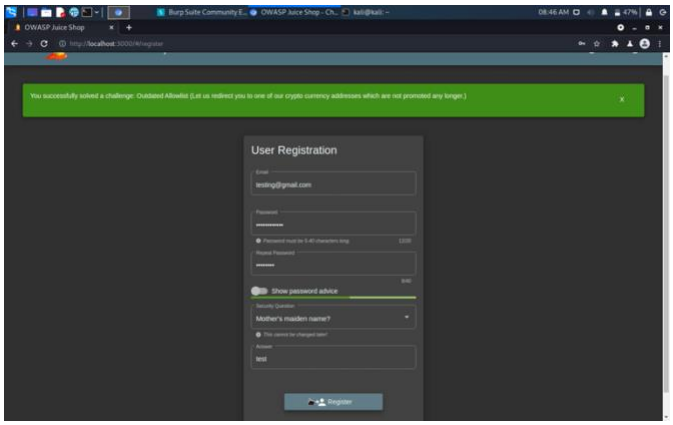
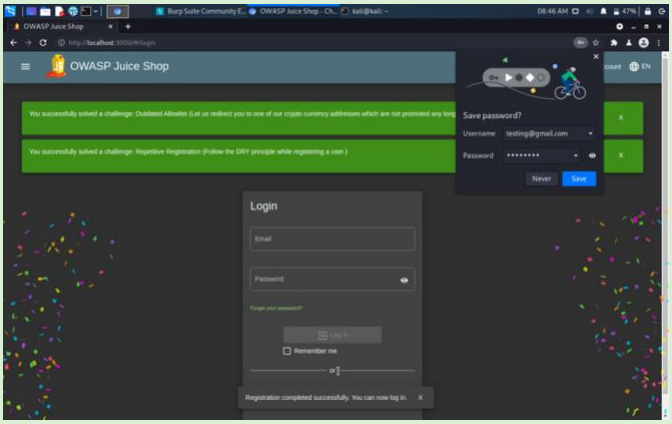
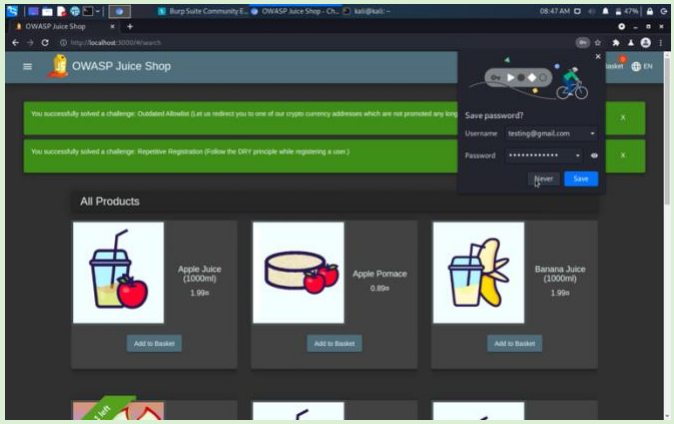


Figure 16 REPETITIVE REGISTRATION

Expected result:

Table 18 REPETITIVE REGISTRATION RESULT

Registered successfully
 

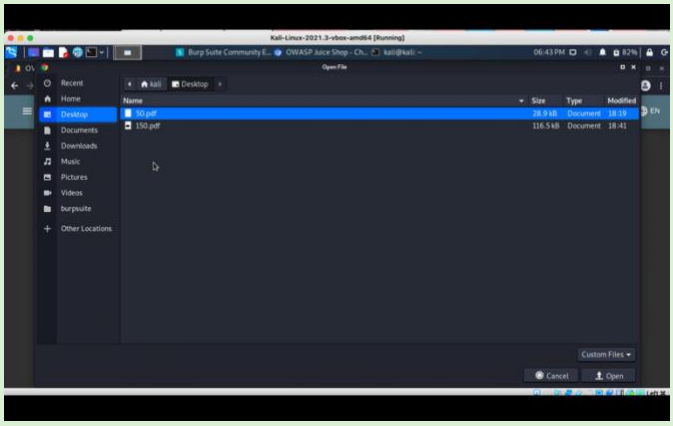
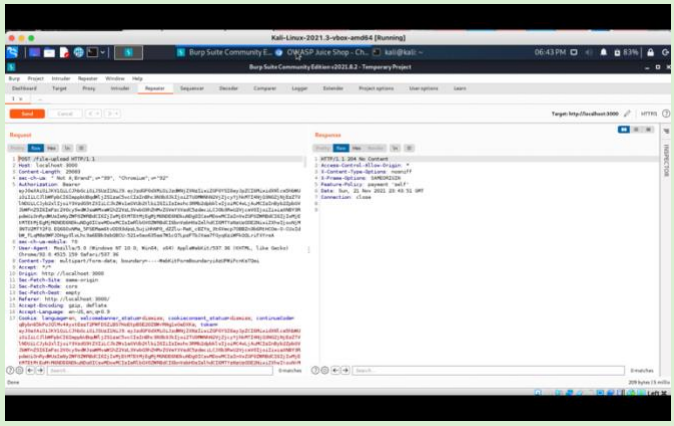
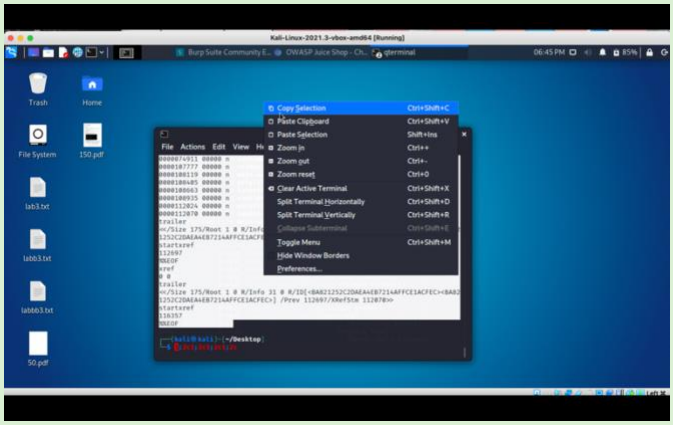
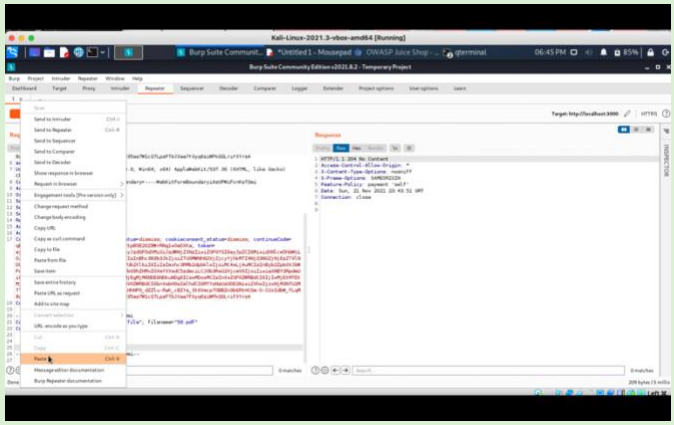
UPLOAD SIZE

Description: Upload a file larger than 100 kB.

Severity: high

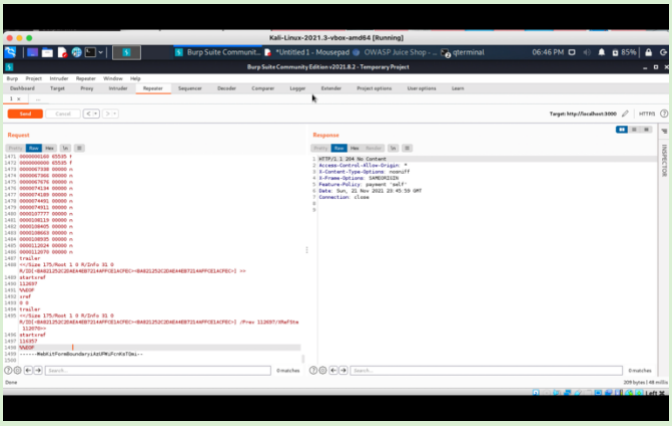
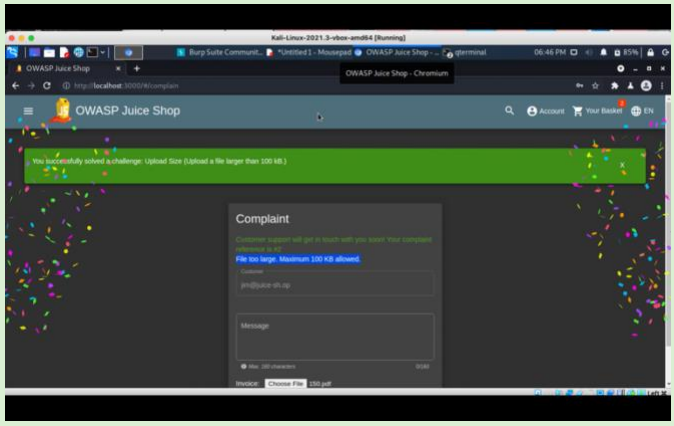
Steps to Reproduce:

Table 19 UPLOAD SIZE

1: upload a file less than 100KB	2: send the request to repeater
	
3: copy the content of the large file	**4: paste it instead of the content of the small file**
	

Expected result:

Table 20 UPLOAD SIZE RESULT

Succesful	
	

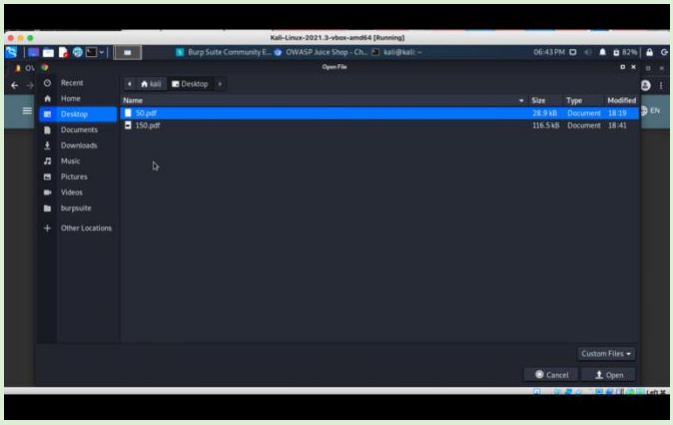
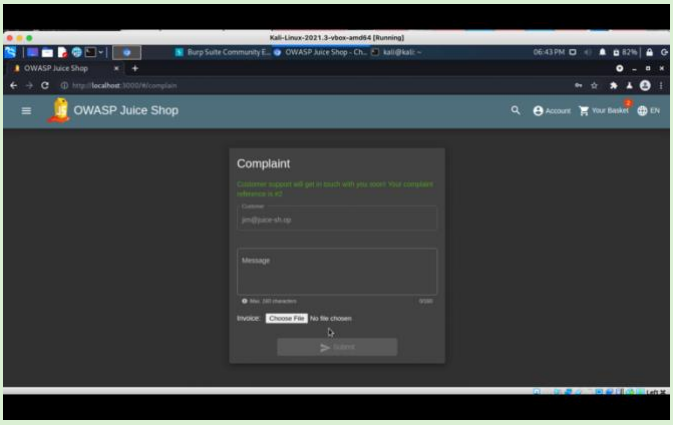
UPLOAD TYPE

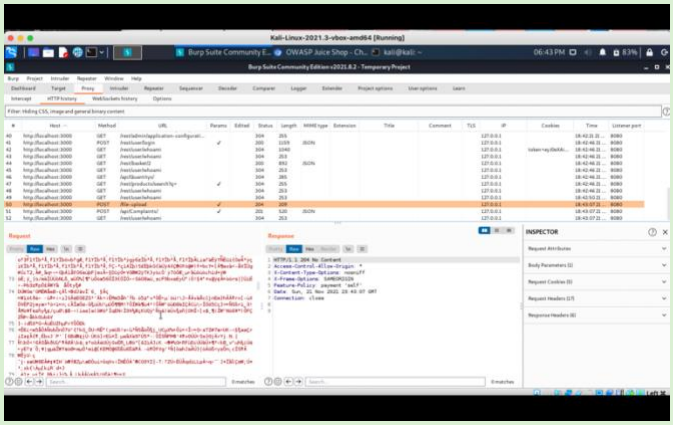
Description: Upload a file that has no .pdf or .zip extension.

Severity: high

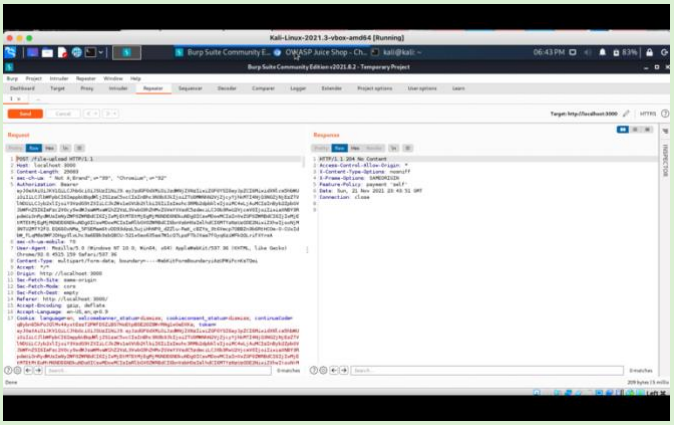
Steps to Reproduce:

Table 21 UPLOAD TYPE

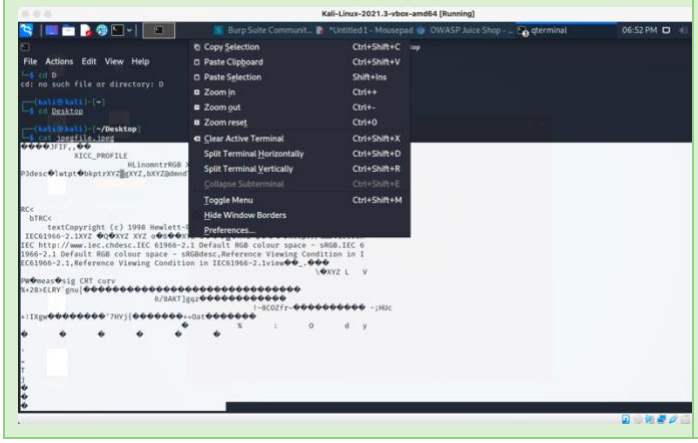
1: upload an accepted file type	
	
2: send the request to repeater	

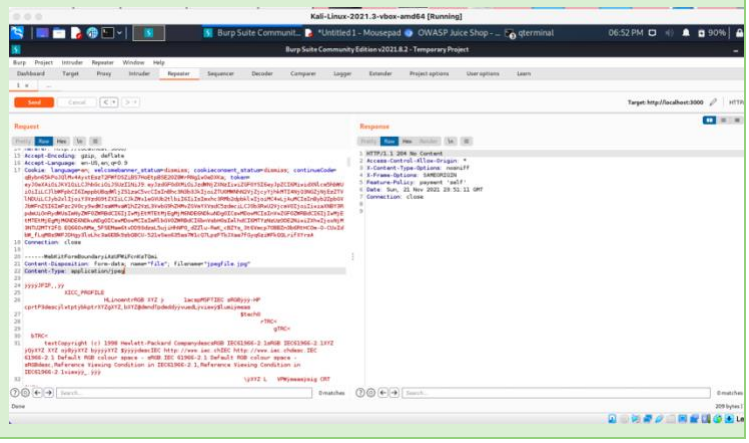


3: copy the content of the unsupported file



4: paste it instead of the content of the supported file, and change the file extension to .jpg





Expected result:

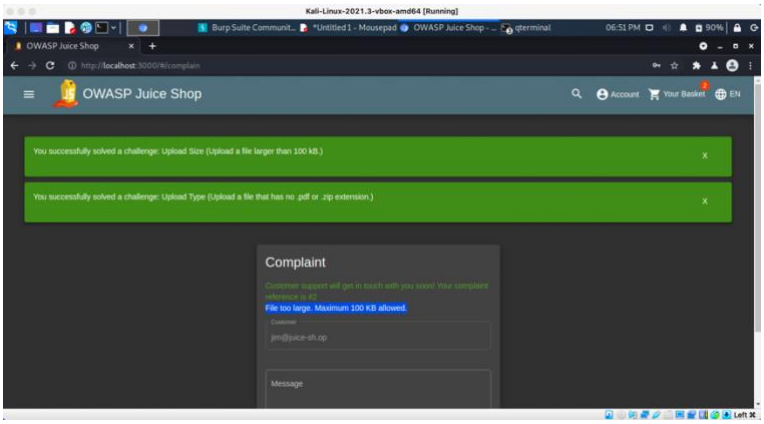


Figure 17 UPLOAD TYPE RESULT

MITIGATION

IMPLEMENTING INPUT VALIDATION

- Numerical variables and dates must have a minimum and maximum value range, whereas strings must have a minimum and maximum length.

ALLOW LIST

- Specify exactly what is permitted and what is not permitted.

VALIDATING FREE-FORM UNICODE TEXT

Because of the vast number of characters that must be permitted in free-form text, particularly Unicode characters, it is thought to be challenging to validate.The following should be the major means of input validation:

- Normalization.
- Character category allow-listing
- Individual character allow-listing.

VALIDATING RICH USER CONTENT

- All clients' data managed have to be encoded when delivered to the HTML page (e.g. XSS).

FILE UPLOAD VALIDATION

- Make a list of the extensions that are permitted.
- Change the filename to something produced by the application.
- Set a filename length restriction.
- Verify the file type.
- Set a file size limit.
- Allow only authorised users to upload files.
- Run the file via an antivirus.
- Ensure that any libraries you're using are properly set and up to date.

EMAIL VALIDATION

SYNTACTIC VALIDATION

The ideal approach to authenticate email addresses is to do some preliminary checks, then provide the address to the mail server to capture the exception if it denies it. This implies that any application may rest assured that its mail server will be able to deliver emails to any address it validates. Initial validation has to be as straightforward as:

- There are two portions to the email address, separated by @.

- There are no harmful characters in the email address (e.g. single quotes).
- Only characters, digits, hyphens, and periods appear in the domain part.
- The email is of a sufficient size.

SEMANTIC VALIDATION

The goal of semantic validation is to determine whether the email is accurate. The typical method is to send the user a verification email. To establish ownership, users should be emailed URLs that contain a unique token with an expiration time.

BROKEN ACCESS CONTROL

Broken Access Control leads to privilege escalation, which occurs when a user is granted advantages to which they are not authorised. These permissions include erasing files, seeing confidential information, or installing malicious software. There are two types of privilege escalation:

- Vertical privilege escalation, occurs when a lower-privilege person has access to a higher-privilege person's information.
- Horizontal privilege escalation, in which a regular user has access to services or material that are only available to other regular users.

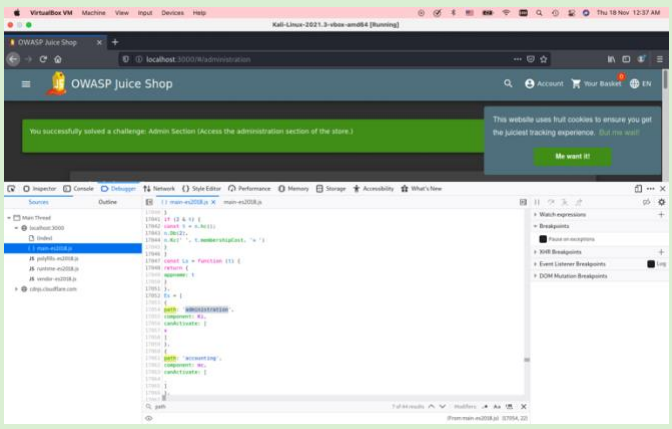
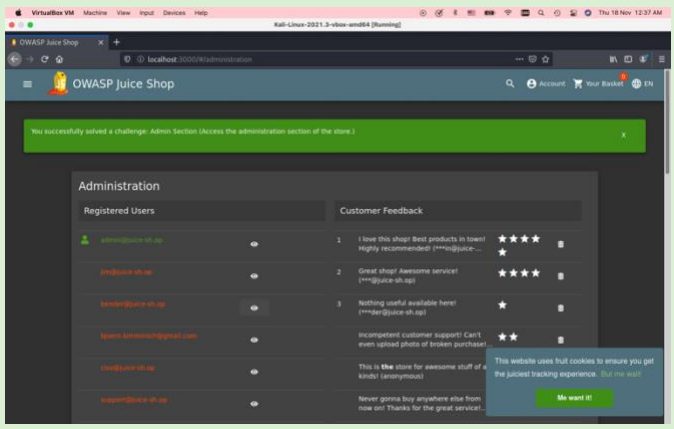
ADMIN SECTION

Description: Access the administration section of the store.

Severity: High

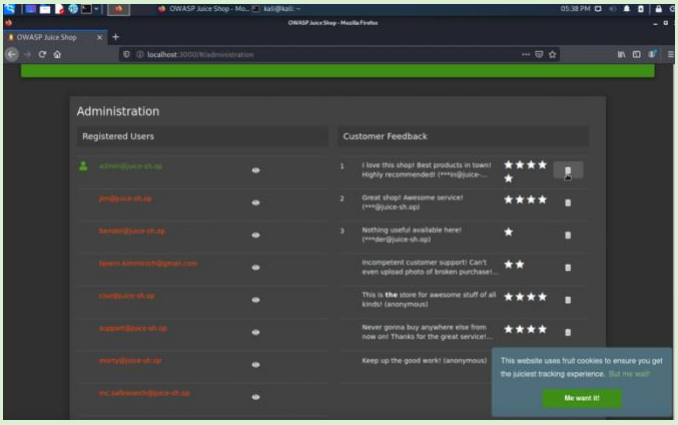
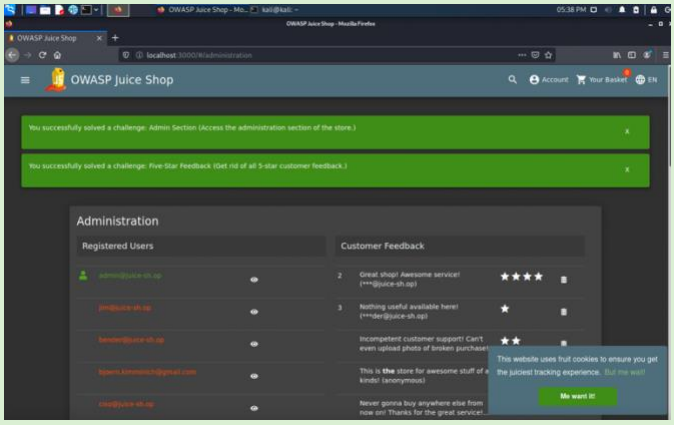
Steps to Reproduce:

Table 22 ADMIN SECTION

1: inspect for paths	2: use it in the URL
	

Expected result:

Table 23 ADMIN SECTION RESULT

Successful!	I also managed to delete feedbacks successfully
	

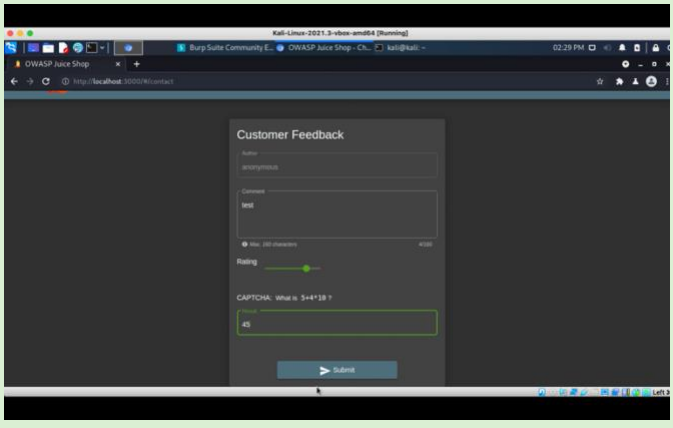
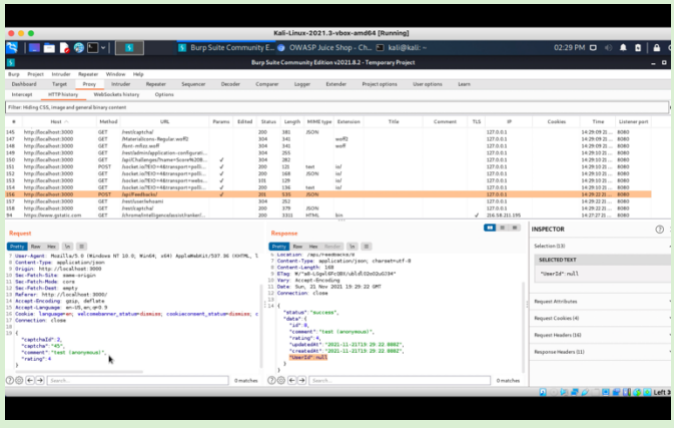
FORGED FEEDBACK

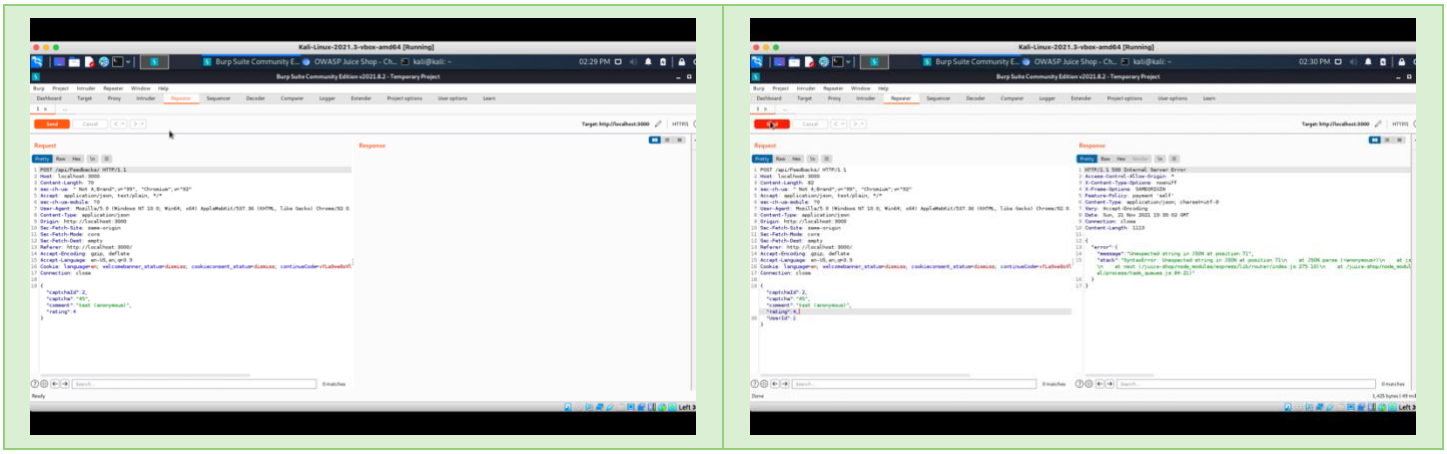
Description: Post some feedback in another user's name.

Severity: medium

Steps to Reproduce:

Table 24 FORGED FEEDBACK

1: submit a feedback	Notice that the userid in response is null
	
2: send the request to repeater	**3: add userid**



Expected result:

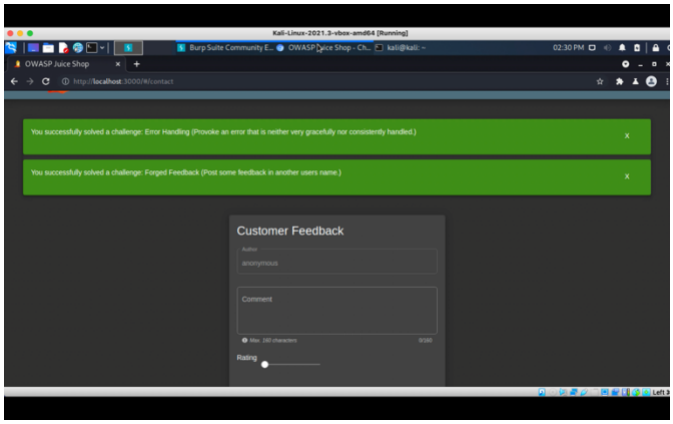


Figure 18 FORGED FEEDBACK RESULT

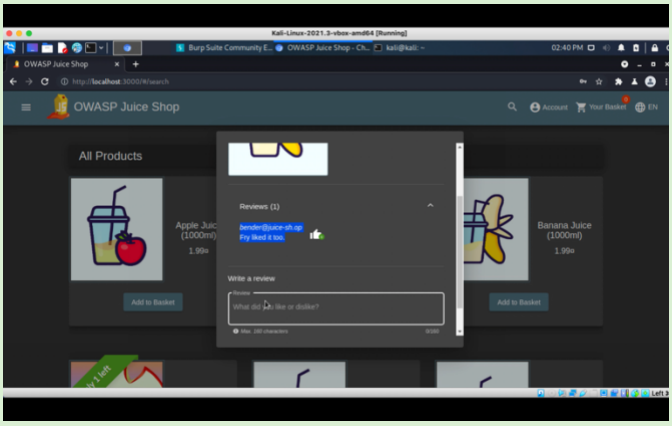
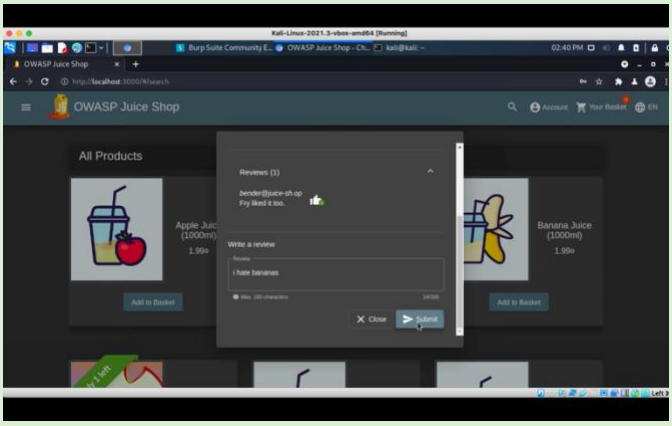
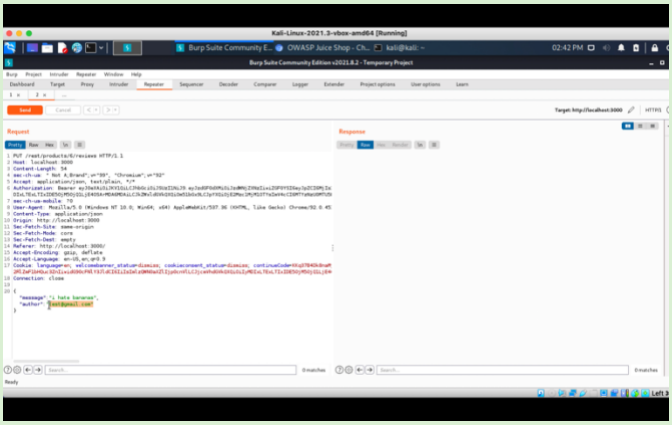
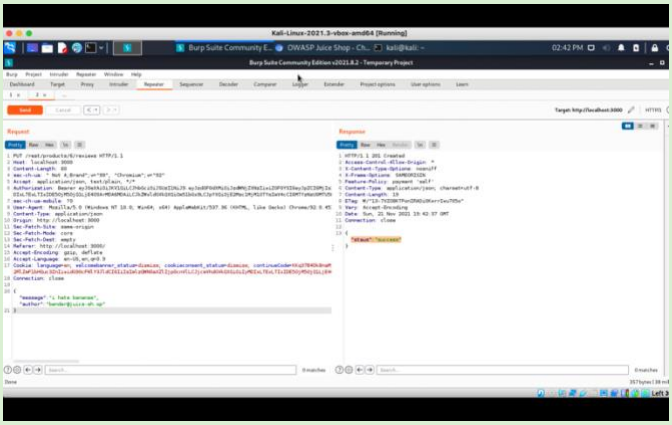
FORGED REVIEW

Description: Post a product review as another user.

Severity: medium

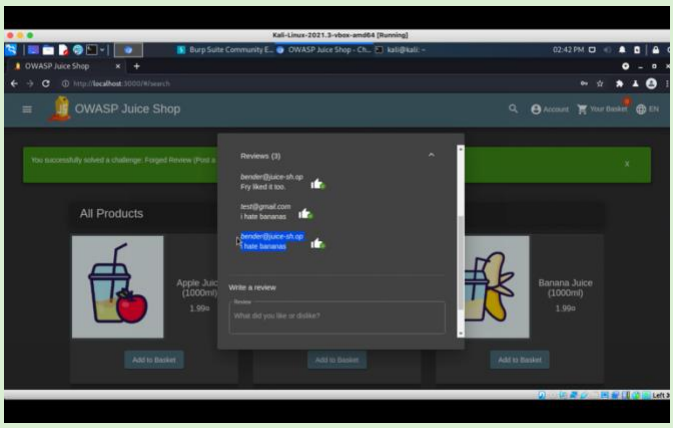
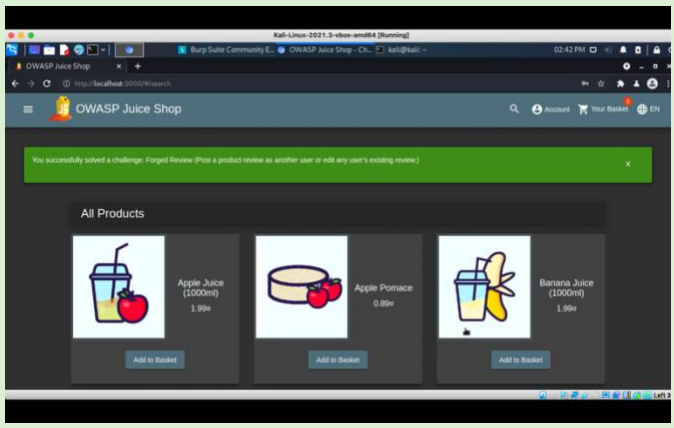
Steps to Reproduce:

Table 25 FORGED REVIEW

1: check for emails to forge	2: write a review
	
3: send the request to repeater	**4: paste the new email and send it**
	

Expected result:

Table 26 FORGED REVIEW RESULT

Successful	
	

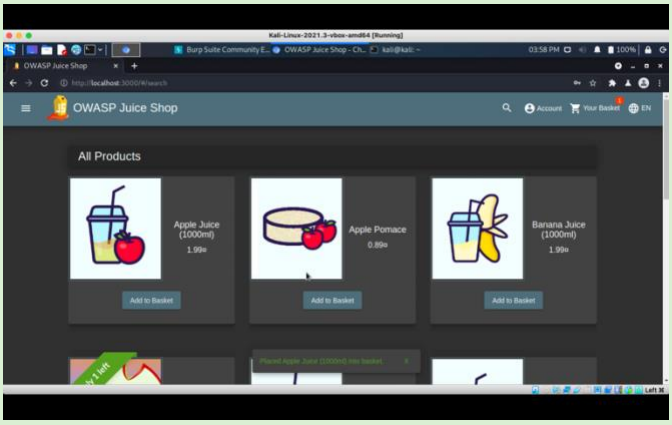
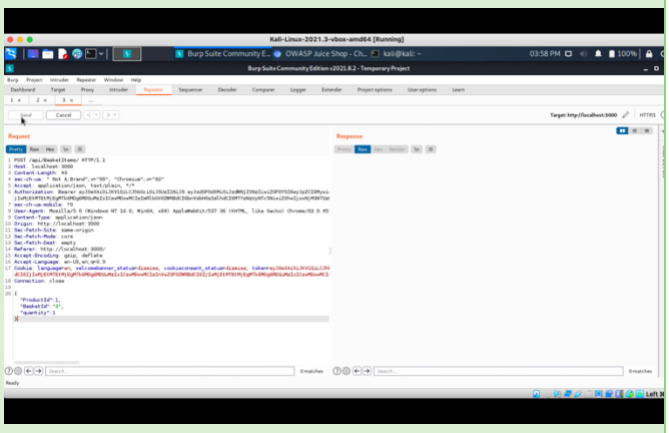
MANIPULATE BASKET

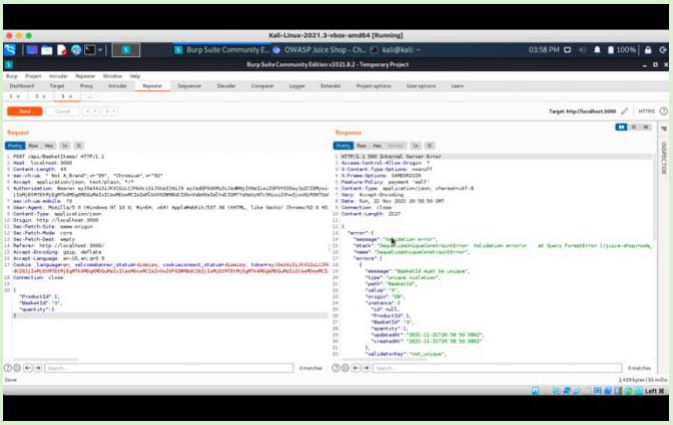
Description: Put products into another user's basket.

Severity: High

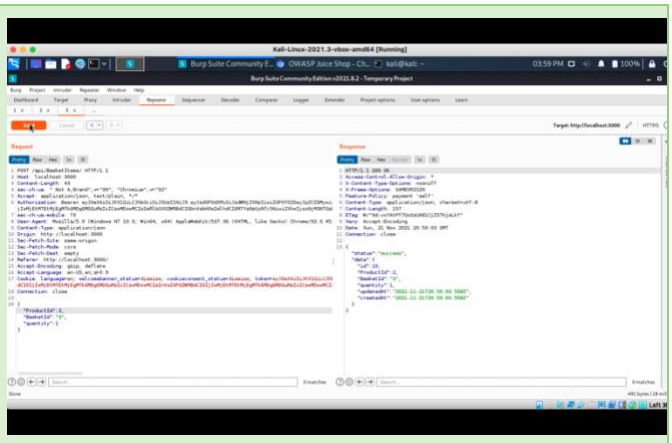
Steps to Reproduce:

Table 27 MANIPULATE BASKET

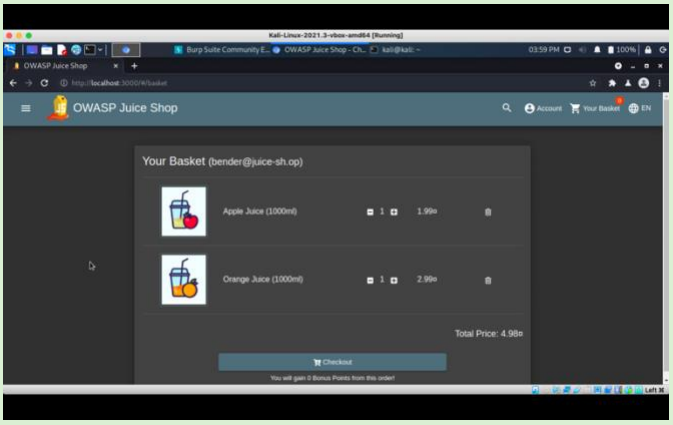
1: add item to basket	2: send request to repeater
	
3: try sending the request again showed a failed response	**4: try changing the productid showed a successful response**



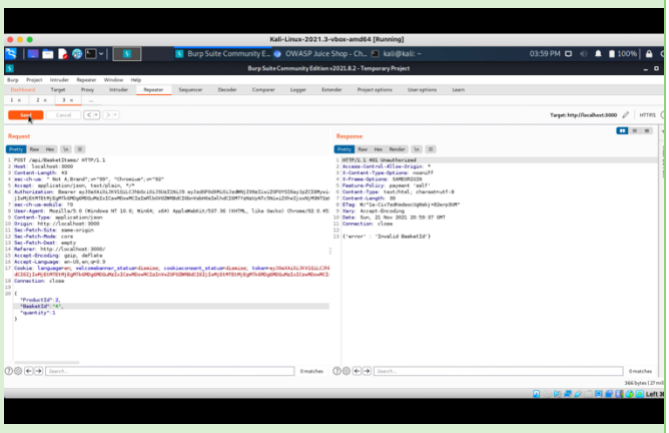
Succesfully added an orange juice



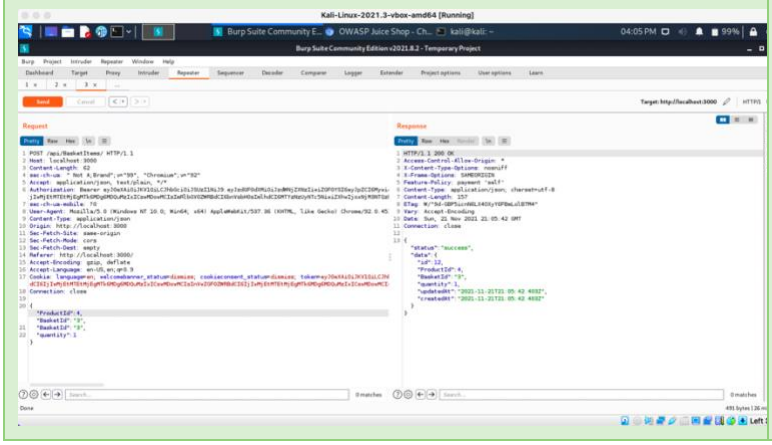
5: try changing the bucketid showed an error

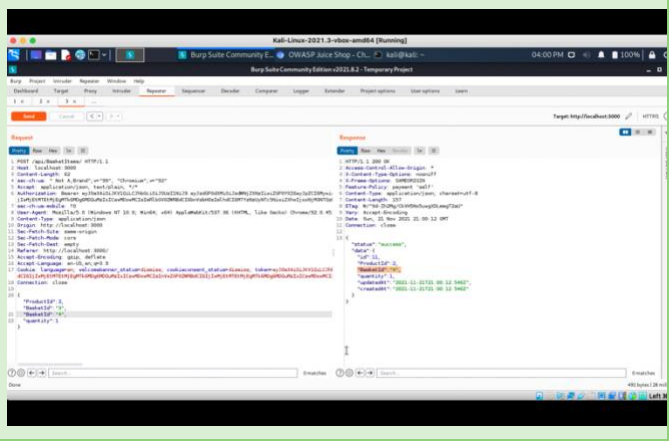


6: duplicate the pruduct id showed a successful response



7: change one bucketid to a different id showed a successful response





Expected result:

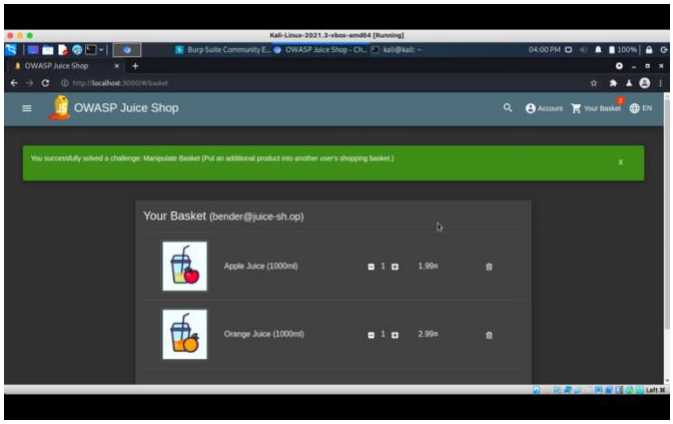


Figure 19 MANIPULATE BASKET RESULT

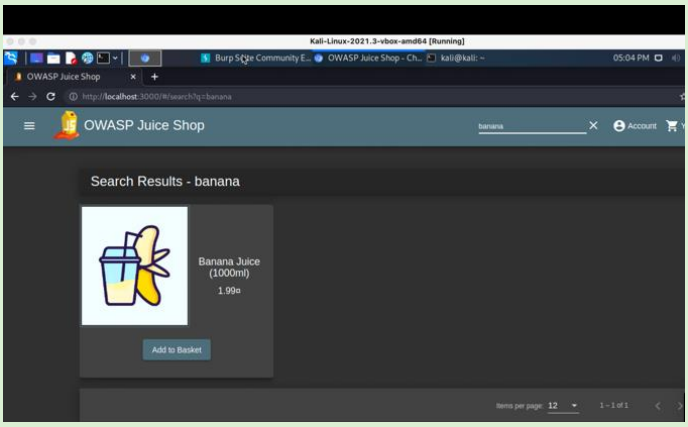
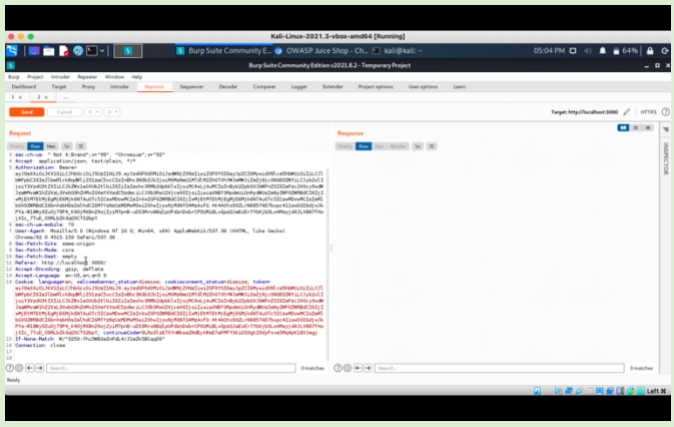
PRODUCT TAMPERING

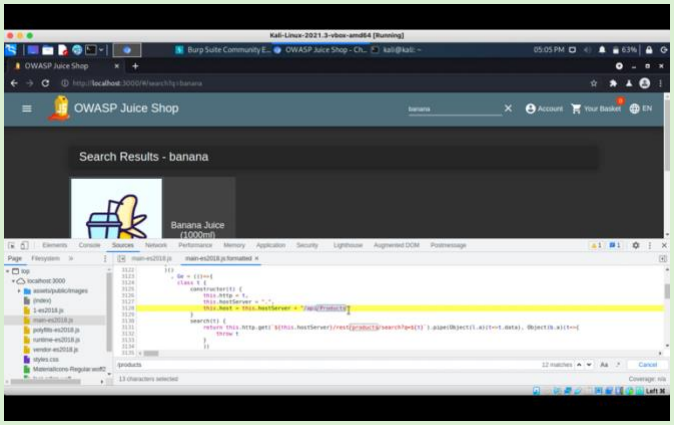
Description: Change the product price.

Severity: High

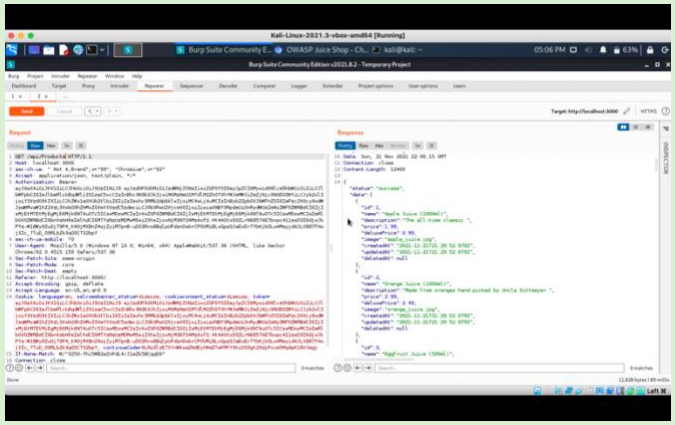
Steps to Reproduce:

Table 28 PRODUCT TAMPERING

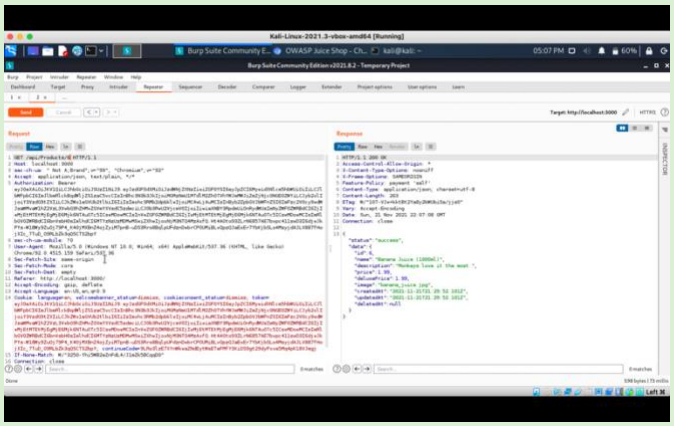
1: look for any product	2: send the request to repeater
	
4: look for products path	**5: try api/productss returned a successful response**



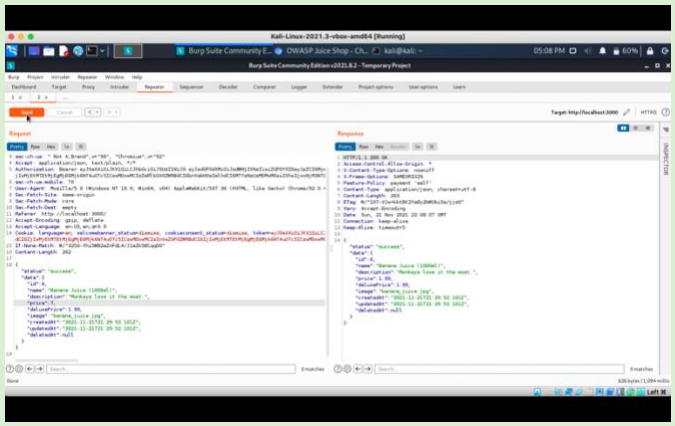
6: retrieving one product by adding /6 returned a successful response



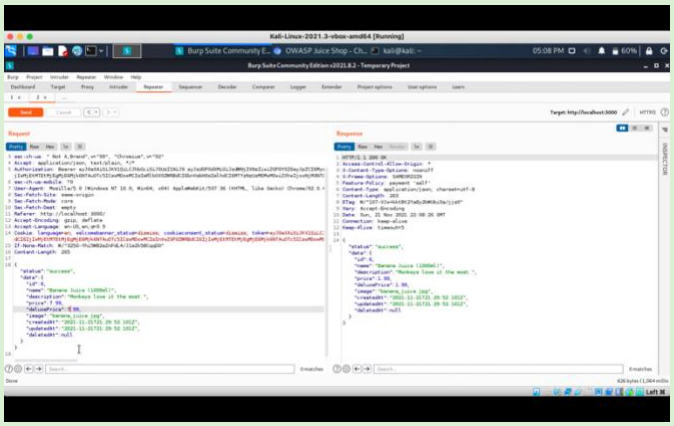
7: copy paste the product data in the request

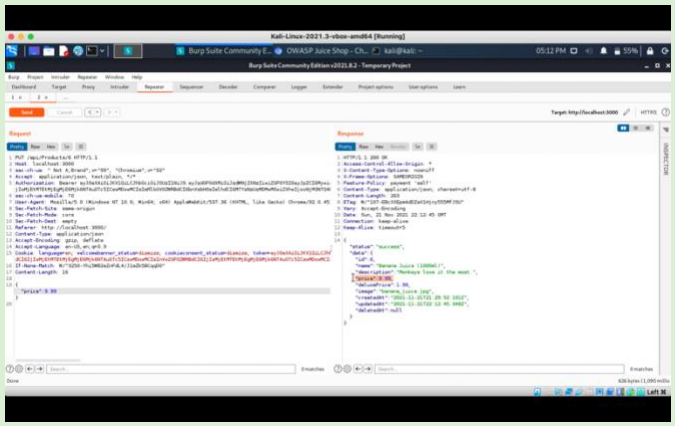


8: changing the price failed



9: kept only the price data and sent it





Expected result:

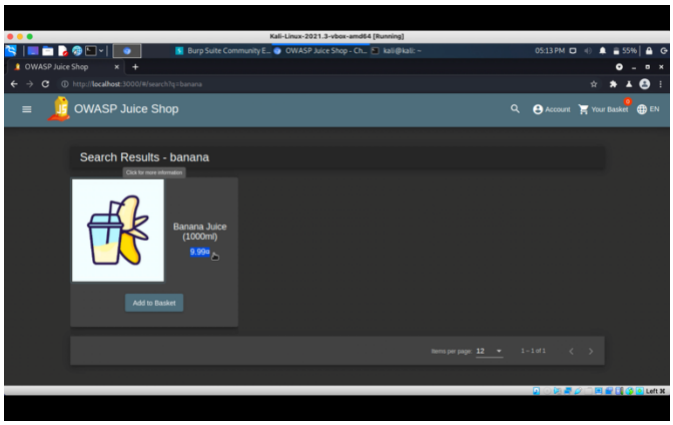


Figure 20 PRODUCT TAMPERING RESULT

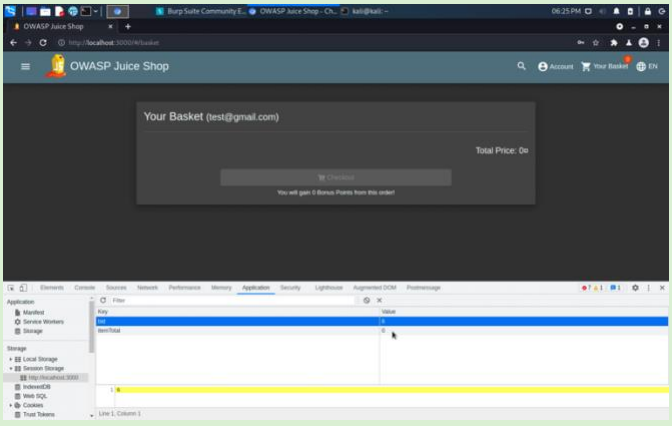
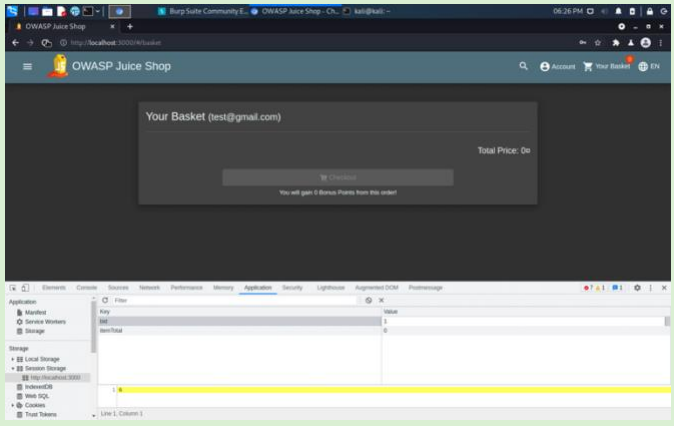
VIEW BASKET

Description: View another user's shopping basket.

Severity: High

Steps to Reproduce:

Table 29 VIEW BASKET

1: inspect session storage	2: change the bid to from 6 to 1
	

Expected result:

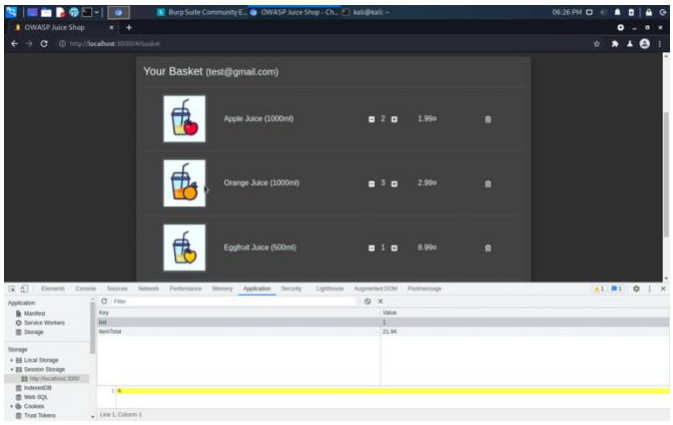


Figure 21 VIEW BASKET

MITIGATION

- Refer to your objects via an array of key-value pairs.
- Modify the names of web pages' default names.
- Make sure that all pages are authenticated.
- Customize error codes and exceptions.

SENSITIVE DATA EXPOSURE

Exposure, damage, loss, alteration, unauthorised disclosure, or access to critical data, is known as sensitive data exposure. Poor database protection, misconfigurations of datastores, incorrect use of data systems, and other factors can lead to data exposure. There are three main forms of sensitive data exposure:

- Confidentiality Breach.
- Integrity breach.
- Permanent and temporary loss of information.

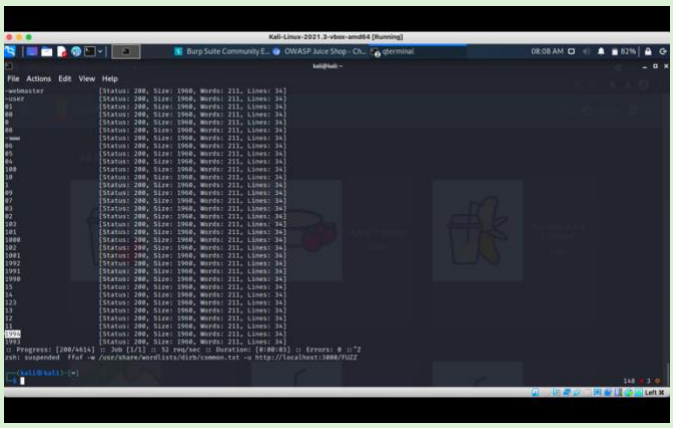
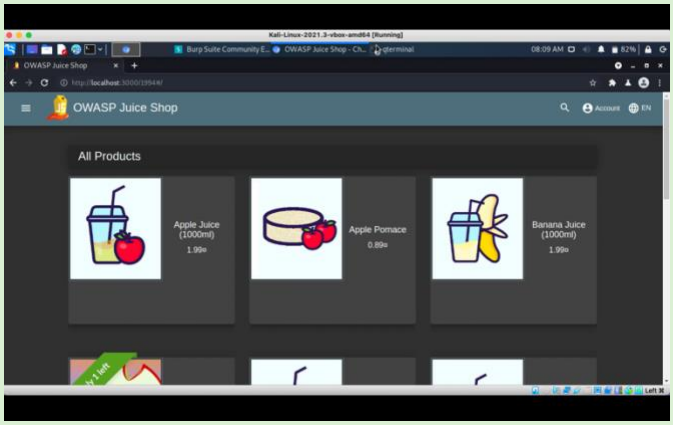
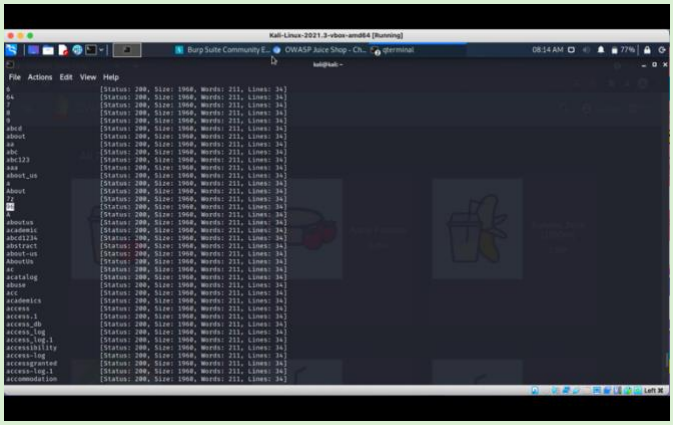
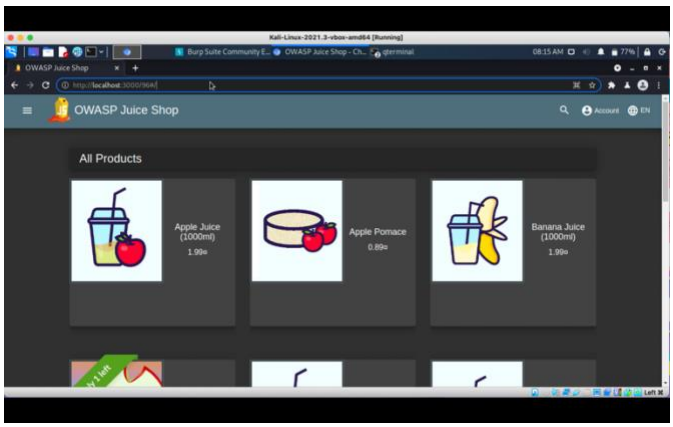
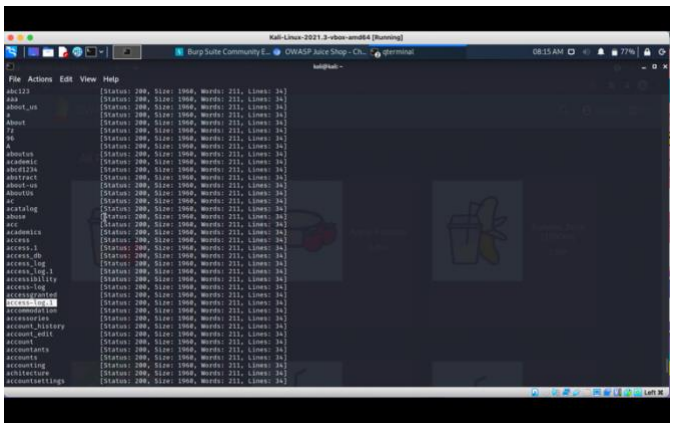
CONFIDENTIAL DOCUMENT

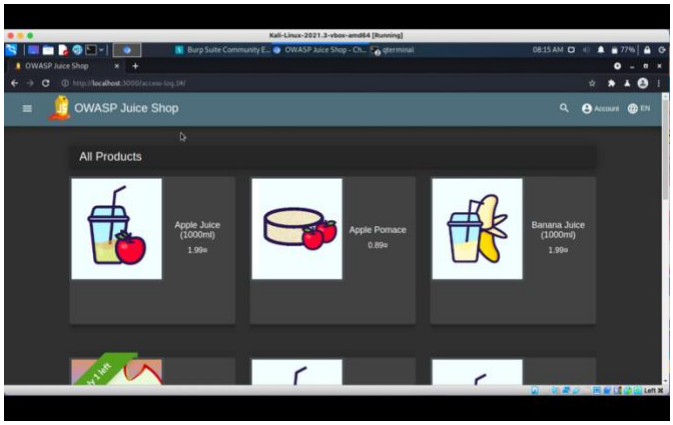
Description: Access confidential documents.

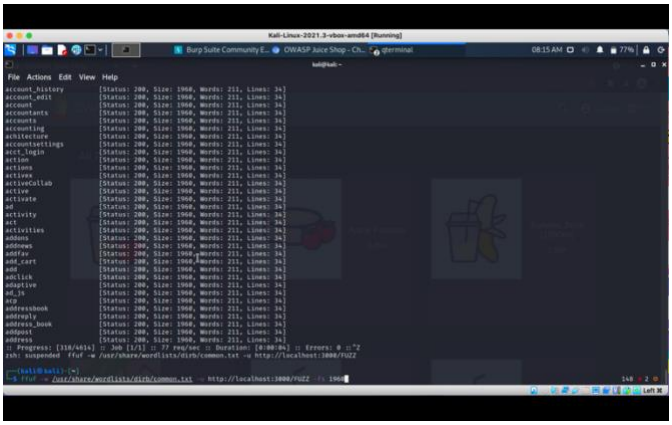
Severity: High

Steps to Reproduce: using the common.txt file in the ffuf tool

Table 30 CONFIDENTIAL DOCUMENT

1: show all redirects using ffuf	
	
2: all redirects are opening the same page	
	
	
2	**3: filter out the content length of 1960**





Expected results:

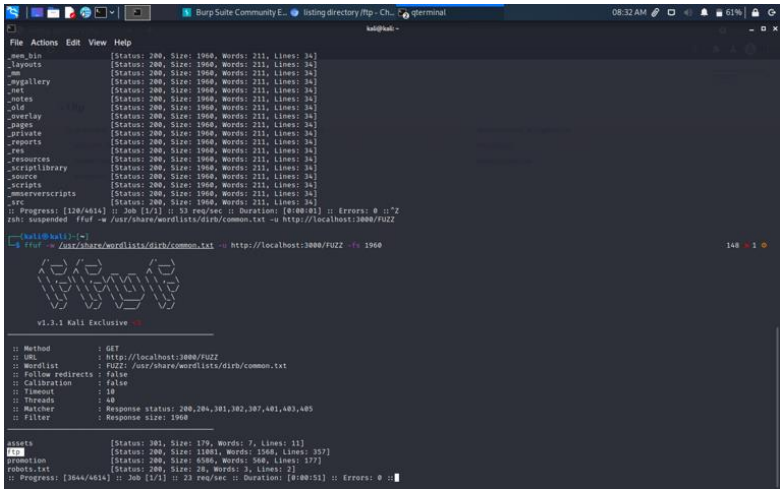
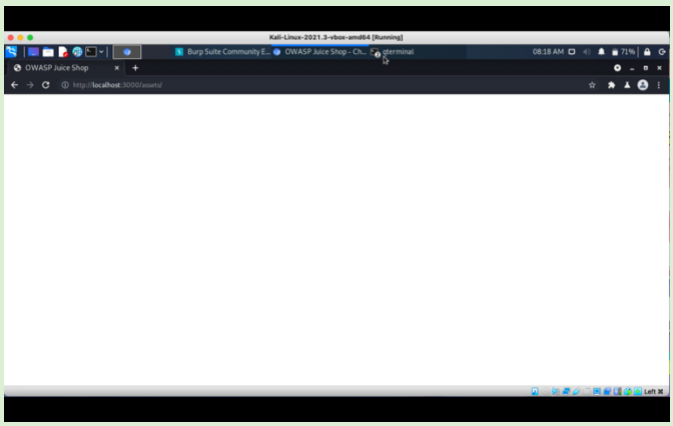
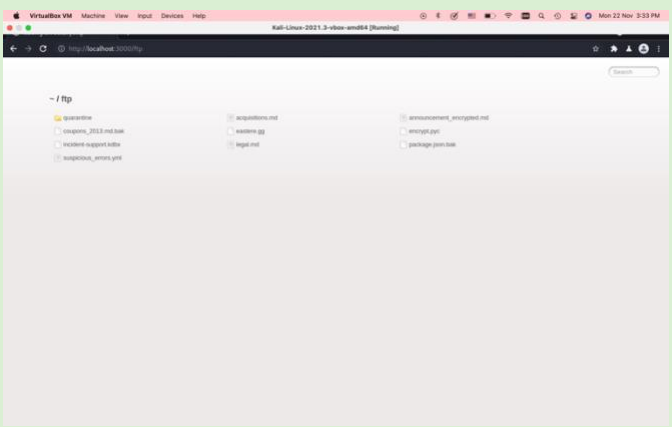
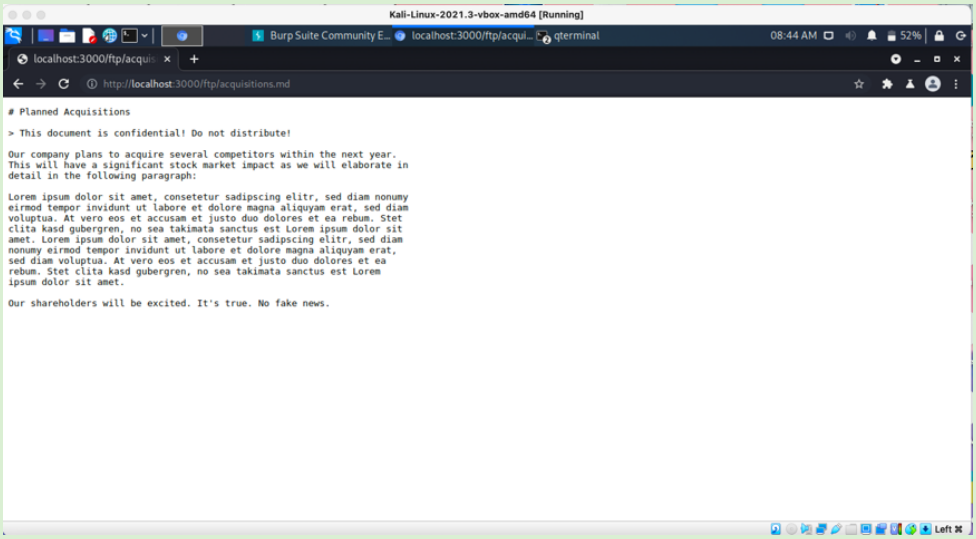


Figure 22 CONFIDENTIAL DOCUMENT RESULT

Table 31 CONFIDENTIAL DOCUMENT RESULT

/assets redirect showed nothing	/ftp redirect showed confidential documents
	

Confidential document accessed



MITIGATION

- Catalog Data
    - Keep track of all data stored in the systems and do an audit. This will provide you a clear image of who owns the data, where it's stored, how safe it is, and how it's governed.
- Assess Data Risks
    - Have a full understanding of the information risk and then dedicate funds and resources for risk mitigation actions. The bigger the chance of damage, the more sensitive the data is.
- Immediate Response
    - Have a strong breach response strategy in place to respond quickly to the disclosure of sensitive data.

UNVALIDATED REDIRECTS

Unvalidated redirect may be utilized to create a URL that passes the access control check in the application and then redirect the intruder to privileged functionalities that they would not usually have access to.

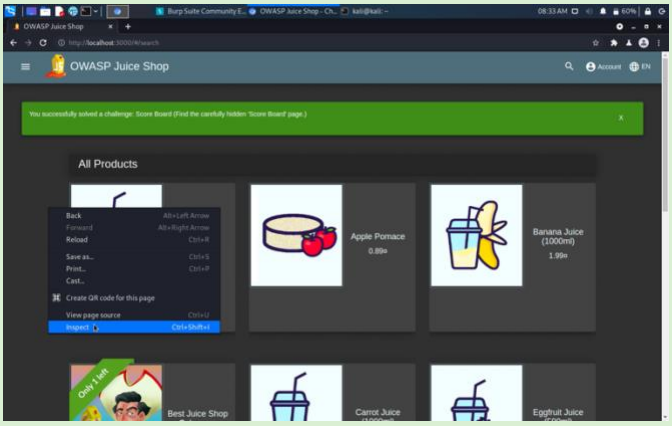
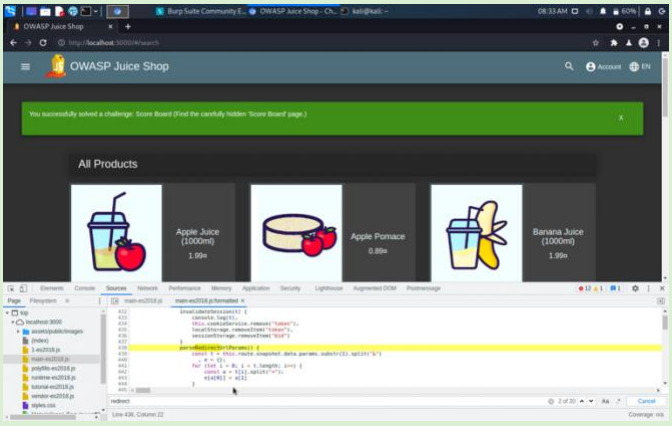
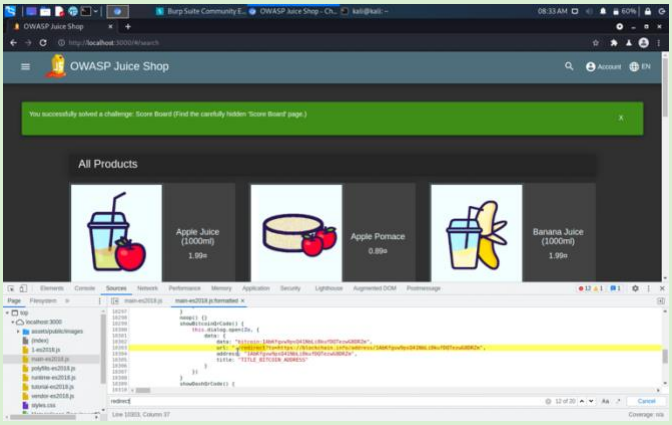
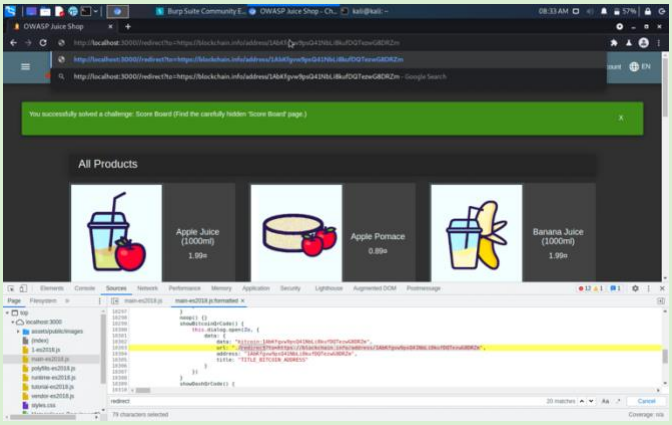
OUTDATED WHITELIST

Description: redirect to one of a crypto currency addresses.

Severity: high

Steps to Reproduce:

Table 32 OUTDATED WHITELIST

1: inspect the webpage	2: check for redirect links in the sources tab
	
3: found a redirect link for a blockchain account	**4: copy paste it in the URL**
	

Expected result:

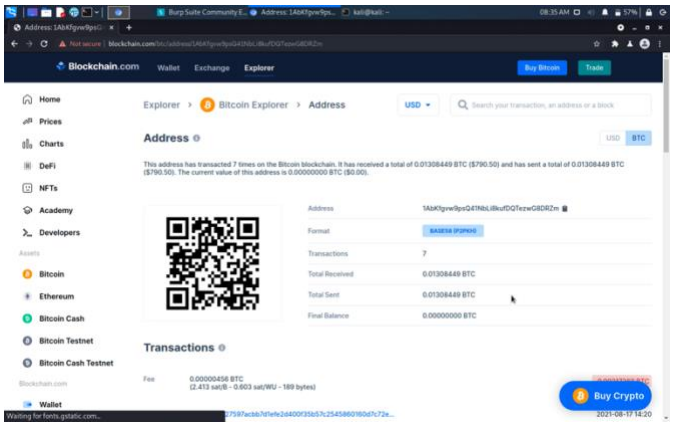


Figure 23 OUTDATED WHITELIST RESULT

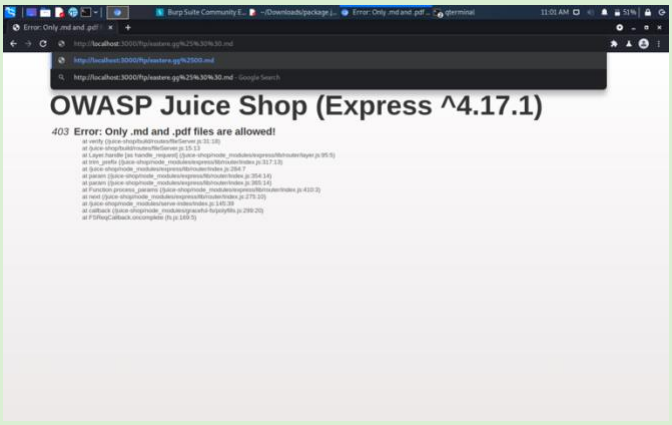
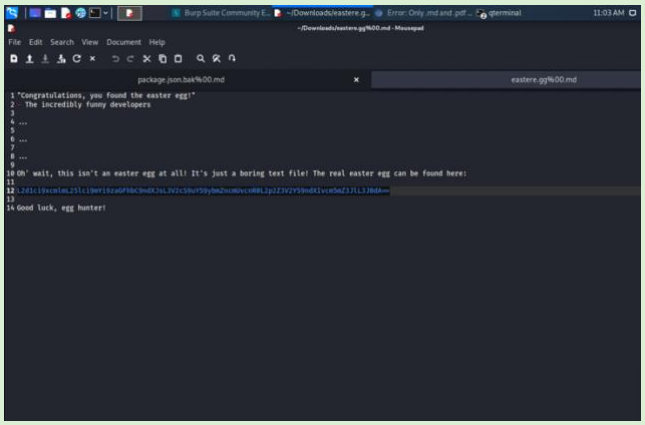
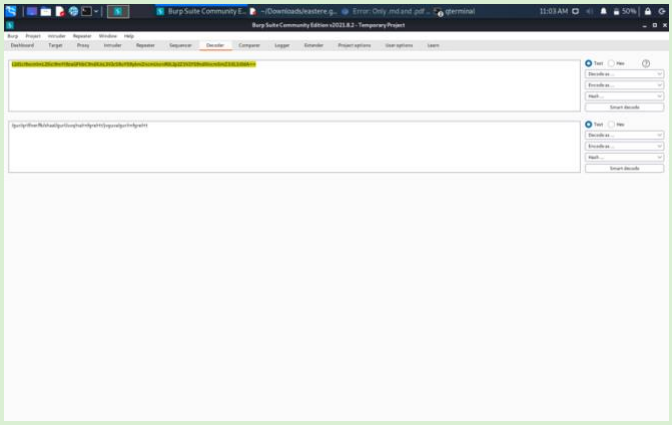
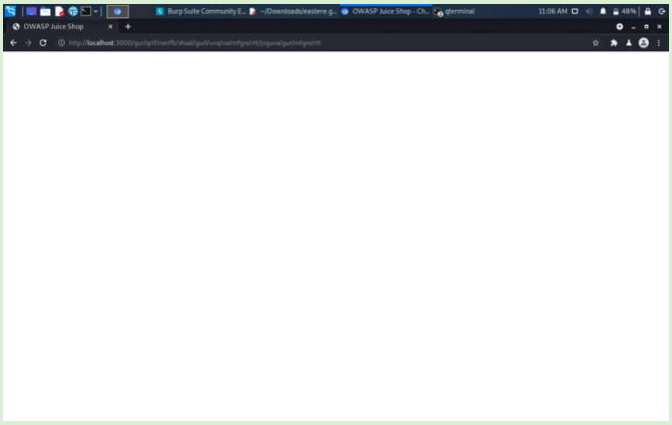
NESTED EASTER EGG

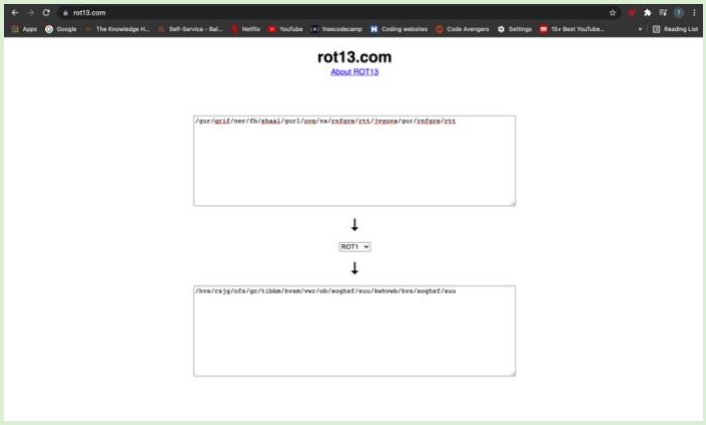
Description: Apply some advanced cryptanalysis to find *the real* easter egg.

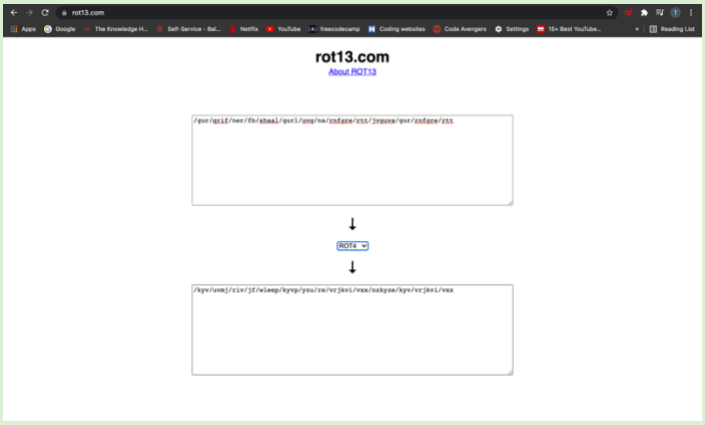
Severity: low

Steps to Reproduce:

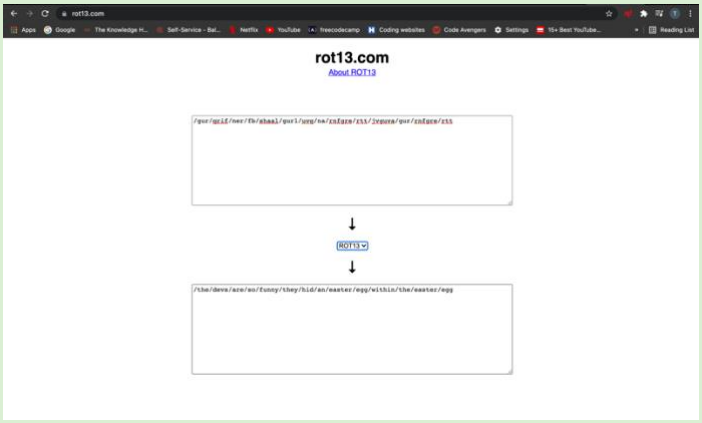
Table 33 NESTED EASTER EGG

1: from the ftp redirect, access the easter.gg file	Successfully downloaded the file by adding .md
	
2: Try decoding the URL	**failed**
	
3: Try rot 13	





Successfully returned a human readable redirect



Expected result:

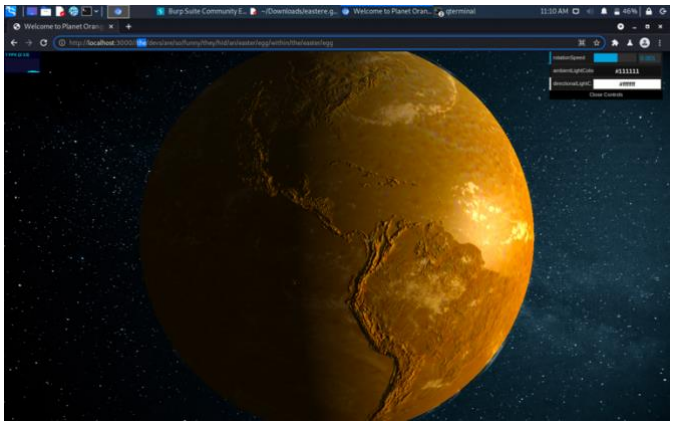


Figure 24 NESTED EASTER EGG RESULT

MITIGATION

- Do not accept the URL as a user input for the destination if it is utilised.
- If user input is unavoidable, ensure that the value provided is correct, acceptable for the programme, and approved for the user.
- Create a list of trustworthy URLs to sanitise input.

- Make all redirects go via a page telling users that they're leaving your site.

BROKEN ANTI-AUTOMATIONS

Websites should not rely completely on anti-automations, because attackers can easily bypass them.

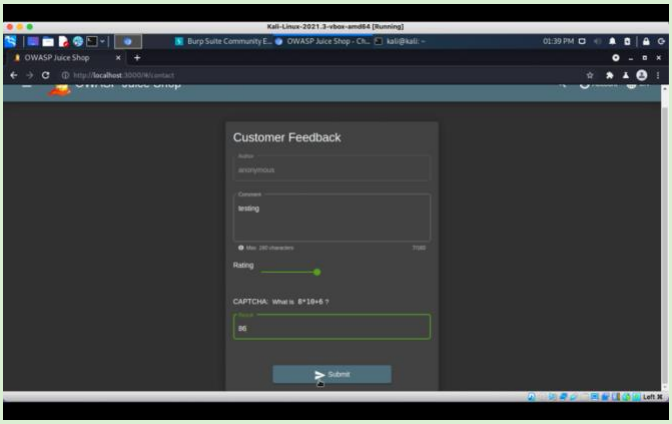
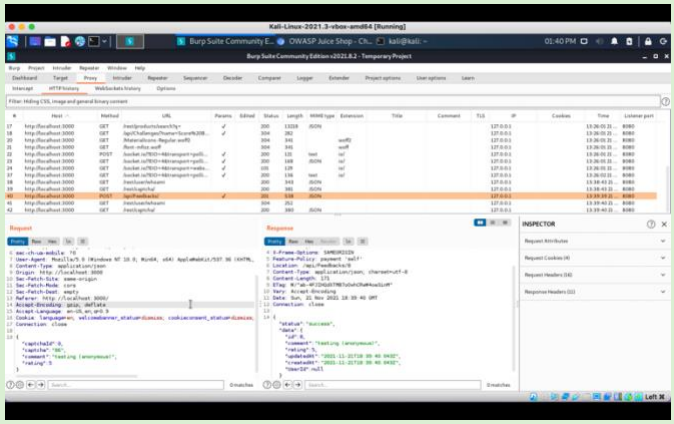
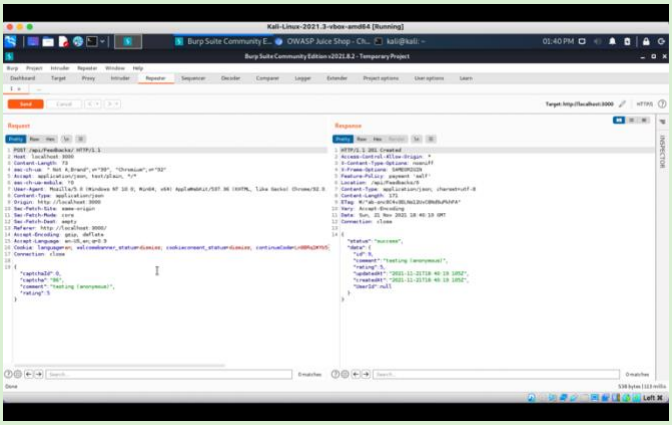
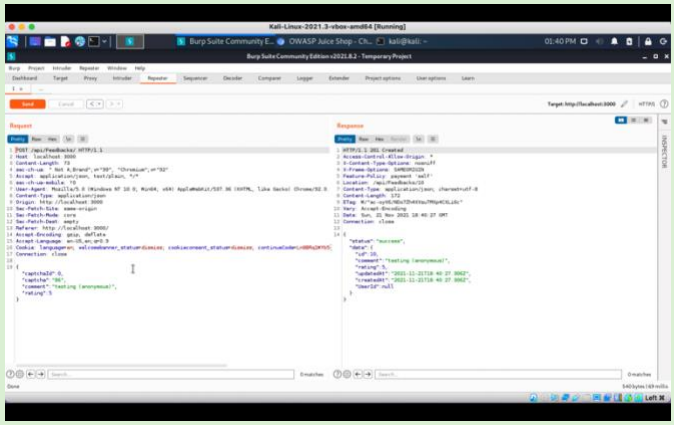
CAPTCHA BYPASS

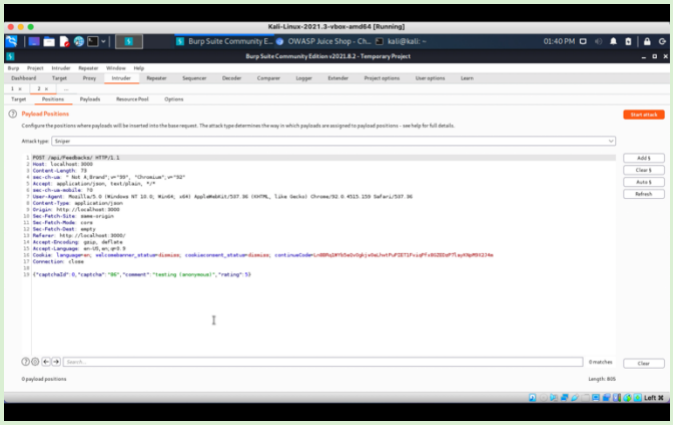
Description: Submit 10 or more customer feedbacks within 10 seconds.

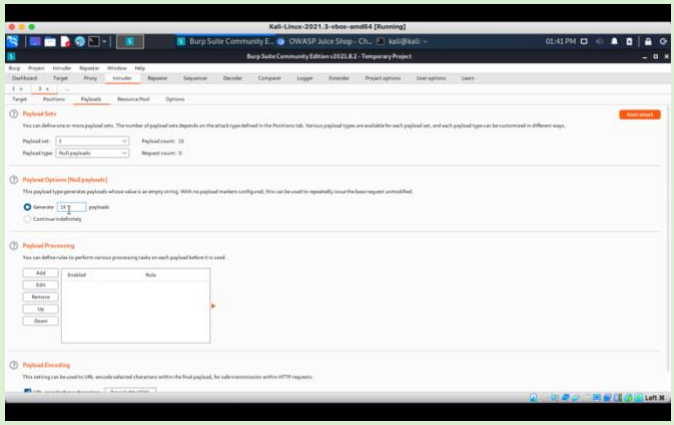
Severity: High

Steps to Reproduce:

Table 34 CAPTCHA BYPASS

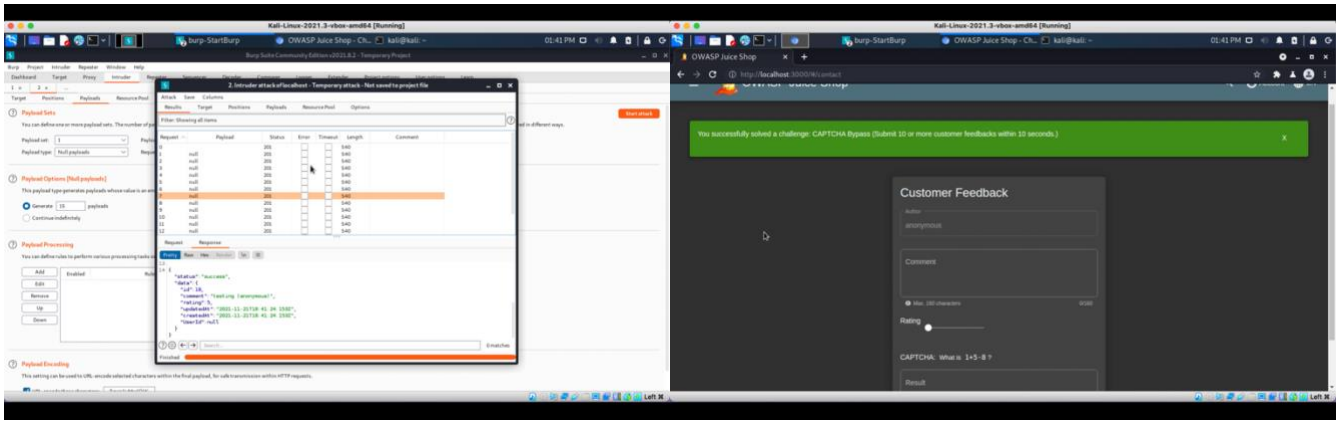
1: submit a feedback	2: send it to repeater
	
3: every time the request is sent, a new id is created.	
	
4: send it to intruder	**5: make a null payload and generate it 15 times**





Expected result:

Figure 25 CAPTCHA BYPASS RESULTS



MITIGATION

- IP Blacklisting
  - Block any IP address that sends suspiciously large amounts of traffic to a website.
- Response Time Monitoring
  - This safeguard guards against bots that solve CAPTCHAs in a fraction of the time that humans do.
- Switching between CAPTCHAs
  - This reduces the chances of bots adapted to decipher explicit CAPTCHAs succeeding.

CLEARING TRACKS

The penetration testing method concludes with the clearing of tracks. The goal is to get rid of any digital clues that the pen tester could have overlooked during the earlier phases of the test The goal of this step is to hide all of the little indications that could reveal the nature of his crimes. Clearing tracks entails countermeasures from anti-Incident Response and anti-Forensics. The following methods can be used to clear tracks:

1. Turn off auditing.
2. Getting rid of logs.
3. Changing logs and registry files.

4. Delete all files and folders that were generated.

CONCLUSION

The Juiceshop web application's penetration testing has been accomplished. This testing was carried out using the technologies and threats that were present at the time of the report's publication. This report examines and discusses all of the discovered security issues.

REFERENCES

Cheatsheetseries.owasp.org. 2021. *Authentication - OWASP Cheat Sheet Series*. [online] Available at: <https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html> [Accessed 5 December 2021].

Cheatsheetseries.owasp.org. 2021. *Authorization - OWASP Cheat Sheet Series*. [online] Available at: <https://cheatsheetseries.owasp.org/cheatsheets/Authorization_Cheat_Sheet.html> [Accessed 5 December 2021].

Cheatsheetseries.owasp.org. 2021. *DOM based XSS Prevention - OWASP Cheat Sheet Series*. [online] Available at: <https://cheatsheetseries.owasp.org/cheatsheets/DOM_based_XSS_Prevention_Cheat_Sheet.html> [Accessed 5 December 2021].

Cheatsheetseries.owasp.org. 2021. *Input Validation - OWASP Cheat Sheet Series*. [online] Available at: <https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html> [Accessed 5 December 2021].

Cheatsheetseries.owasp.org. 2021. *SQL Injection Prevention - OWASP Cheat Sheet Series*. [online] Available at: <https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html> [Accessed 5 December 2021].

Cheatsheetseries.owasp.org. 2021. *Unvalidated Redirects and Forwards - OWASP Cheat Sheet Series*. [online] Available at: <https://cheatsheetseries.owasp.org/cheatsheets/Unvalidated_Redirects_and_Forwards_Cheat_Sheet.html> [Accessed 5 December 2021].

Cheatsheetseries.owasp.org. 2021. *Vulnerability Disclosure - OWASP Cheat Sheet Series*. [online] Available at: <https://cheatsheetseries.owasp.org/cheatsheets/Vulnerability_Disclosure_Cheat_Sheet.html> [Accessed 5 December 2021].

DataArt, 2020. *Web Application Penetration Testing Report*. [online] DataArt. Available at: <https://uploads-ssl.webflow.com/58926952c918e87a3e6e40eb/5ebd6740d395bad501b4a4d5_WebApplication_PenetrationTesting_Report_Example_Fully_Redacted.pdf> [Accessed 5 December 2021].

Kimminich, B., 2021. *Introduction · Pwning OWASP Juice Shop*. [online] Pwning.owasp-juice.shop. Available at: <https://pwning.owasp-juice.shop/> [Accessed 5 December 2021].

Owasp.org. 2021. *OWASP Top Ten Web Application Security Risks | OWASP*. [online] Available at: <https://owasp.org/www-project-top-ten/> [Accessed 5 December 2021].

Sqlite.org. 2021. *The Schema Table*. [online] Available at: <https://sqlite.org/schematab.html> [Accessed 5 December 2021].

APPENDIX A: RISK RATING SCALE

Each vulnerability has been assigned a qualitative severity factor (High, Medium, or Low). The severity classification of activities is depicted in the tables below:

Severity	Definition
Low	If taken advantage of, it will have little influence on the firm. Information released has no substantial negative consequences, no rejection or legal ramifications, and has little to no influence on regulatory or standards compliance.
Medium	Financial effect is moderate, with the possibility of legal penalties and reputational damage.
High	Significant financial loss, brand and corporate identity harm as a result of probable media engagement, exposure, and data compromise

APPENDIX B: TOOLS LIST

Tool	Description
Burp	Burp Suite is an all-in-one web application exploitation tool.
ffuf	ffuf is a tool that allows typical directory discovery.
Inspect	Inspect lets you edit the HTML and CSS of the page displayed in the browser.
Rot13	ROT-13 tool is a particular case of the Caesar cipher, where the shift is equal to 13.