Ronald Li

OWASP JuiceShop

Penetration Testing Findings Report

Date: April 25, 2024
Version 1.0

Please note this is an incomplete report published April 25, 2024. The report will continue to be updated as testing continues.

Table of Contents

Introduction

This project emulates a pentesting engagement for a fictional client owning the OWASP JuiceShop application. This written report is the output of the penetration testing that would be provided to the client at the end of the engagement. While many write-ups and videos regarding OWASP JuiceShop exist online, **no external sources of information were used for this project.** This is to practice our pentesting workflow and also to replicate a true pentest where we would not have external assistance.

The main focus of this project was to demonstrate web application penetration testing capabilities. As such, some items that would normally be part of a pentest are not included here. Notably:

1. The use of vulnerability scanners due to its ability to automate findings
2. Some mitigation recommendations

OWASP JuiceShop was chosen as the target for the following reasons:

1. Free and accessible
2. Resembles a modern web application. Vulnerabilities are not immediately identifiable without technical know-how
3. Filled with a large number of vulnerabilities

For more information on OWASP JuiceShop visit: https://owasp.org/www-project-juice-shop/

Confidentiality Statement

This report is the exclusive property of Ronald Li. Duplication, redistribution, or use, in whole or in part, in any form, requires consent of the author.

The contents of this document do not constitute legal advice. The assessment detailed herein is performed for practice purposes only.

Contact Information

Name	Title	Contact
Ronald Li	Penetration Tester	Email: ronald.li.121206@gmail.com

Executive Summary

Two critical issues were discovered and should be remediated immediately. The login feature could be bypassed with SQL injection that allows an attacker to gain administrative access to the application. The second critical issue is the administrator account uses a weak password that could be found either through login bruteforcing, password cracking, or searching up the hash in an online database. The administrative account password should be changed immediately. The ease of the exploits combined with the high impact makes this very high risk. We recommend the application be taken offline immediately until both of these vulnerabilities are remediated.

Throughout our testing of the JuiceShop application we have several consistent issues. A lack of server-side input validation can be found for many input fields on the application. This includes verification of the previous password in the “Forgot Password” feature, entering a CAPTCHA, and many of the fields during User Registration. While the browser may restrict invalid inputs, this does not stop an attacker from bypassing these controls with the use of a proxy tool, as demonstrated in this report.

Another consistent issue was the lack of access control for directories that are intended for authenticated users. This was found on the majority of pages for authenticated users. Only several pages had appropriate access control.

The use of a CSRF token should be implemented whenever the user performs a sensitive action. This should be used on features such as Password Change and user Data Erasure.

On a positive note, we found the cryptography used for the application to be fairly secure, with the exception of hashing passwords with MD5 which is an outdated hashing algorithm. The use of signed JWT meant that attackers can’t tamper with the token to bypass authentication or perform privilege escalation.

Tools

Testing was mainly performed using BurpSuite Community Edition. A full list of tools is listed below:

Tool	**Description**
BurpSuite Community Edition	Web Proxy
OWASP ZAP	Spidering, directory and file discovery
Dirbuster	Directory Discovery
Hash-Identifier	Hash Identification
Hashcat	Password cracking
DotDotPwn	Directory traversal testing

Scope of Work

Penetration testing to be performed only on the web application hosted 192.168.18.7 on port 3000 of the local server. No social engineering tests were performed. No tests on the web server or network infrastructure were performed.
The version of the web application is OWASP Juice Shop v16.0.0
The test is performed from a Kali Linux VM on the same local network as the web server.

Methodology

The objective of this engagement is to find as many vulnerability findings as possible. This can differ from other pentests where the objective is to gain administrative access to the application.

Our scope of work was conducted as a black box penetration test. There was no information provided by the application owner regarding the application. No external sources of information regarding the application was used to support the penetration test.

All testing methodology performed was described in this report. Tests not written in this report can be assumed to not have been performed.

Vulnerability findings are defined under the following severity categories:

Critical: Serious security issue that could allow for remote code execution, root level privileges. Exploit can compromise confidentiality, integrity, availability to a significant degree. Exploit could also be of low complexity to execute.

Examples: SQL injection, command injection, remote shell

Major: Attackers could access sensitive information. Impact is limited compared to critical findings. Conditions to perform the exploit is typically also more limited relative to critical issues.

Examples: session hijacking, XSS or execution of client-side scripts, poor TLS configuration allowing for sniffing

Minor: Lack of some security measure, configuration error, or low level of information leakage

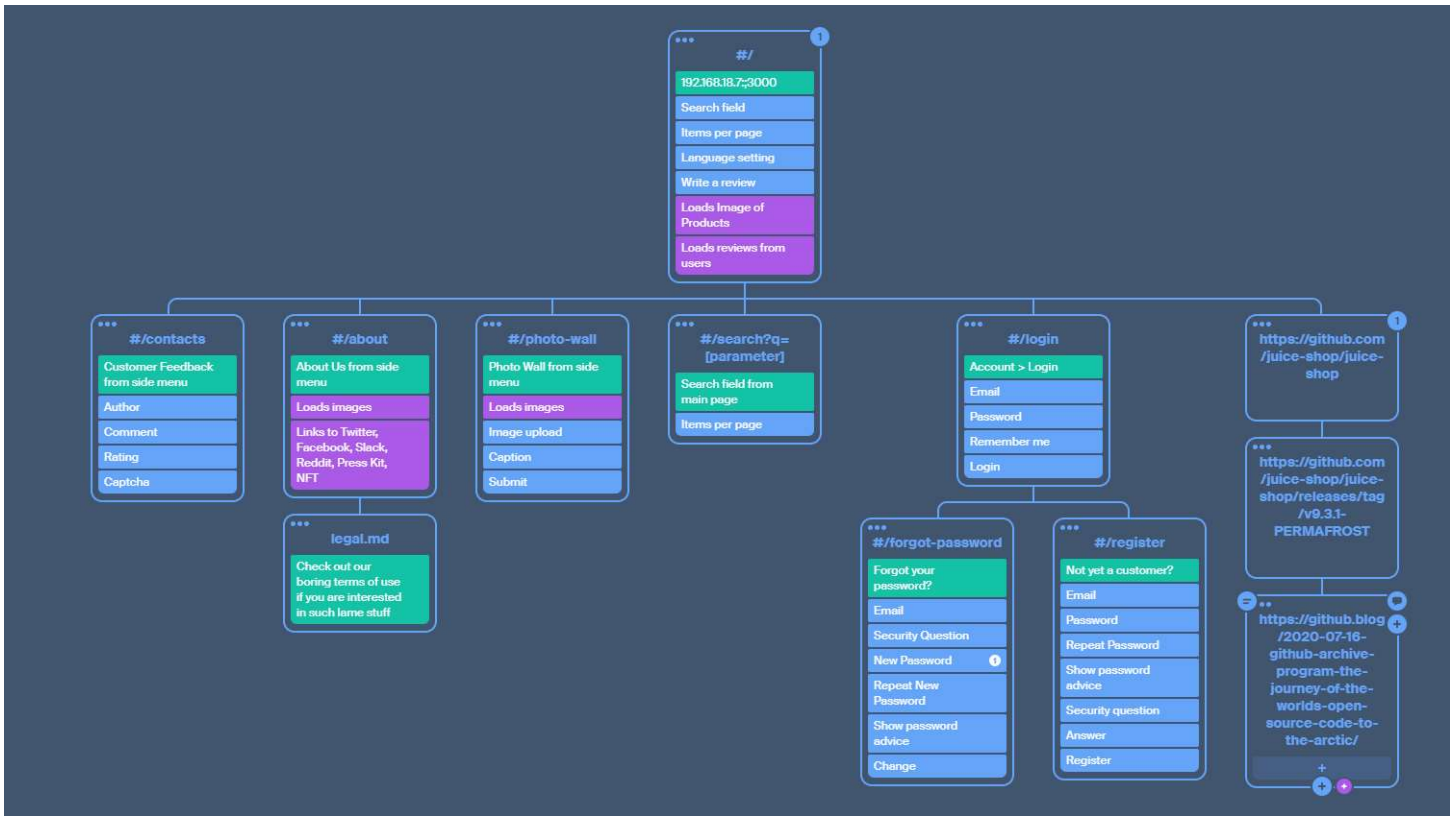
Examples: Application revealing the version number of an application or OS

Info: Not a security issue but does not follow best practices, or could become a security issue in the future

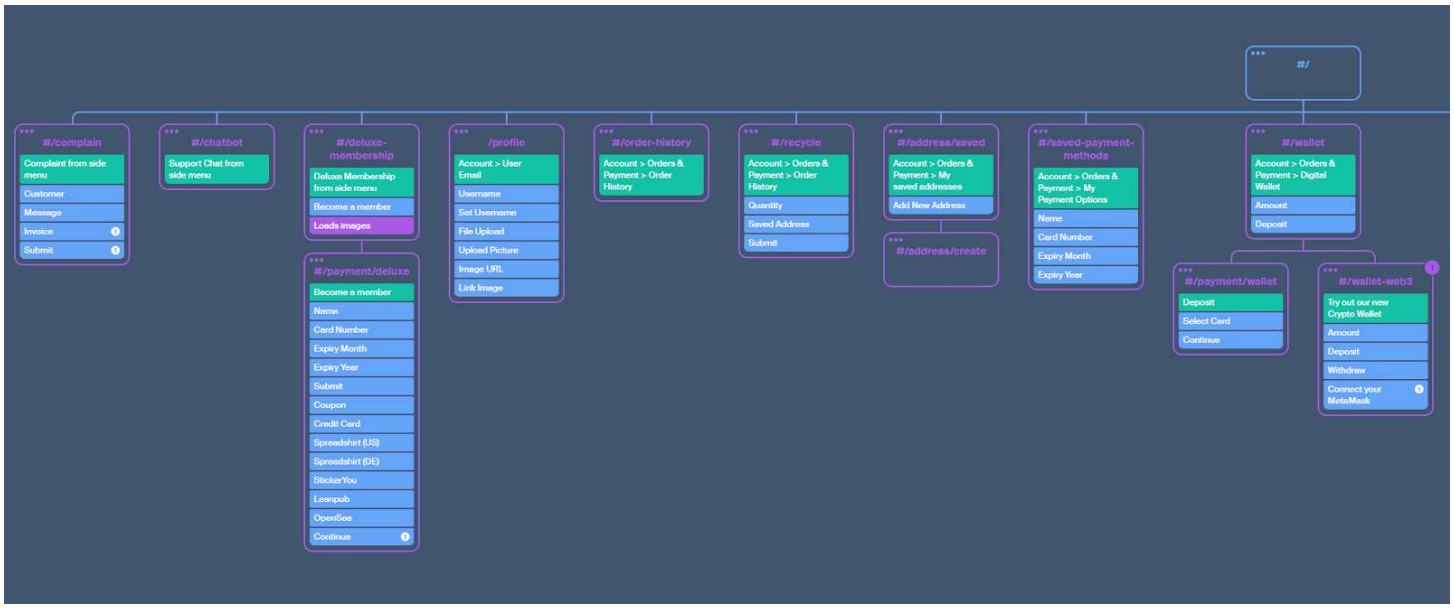
Examples: Using an older dependency version but it does not contain known vulnerabilities, use of multiple authentication mechanisms where single sign-on is more appropriate

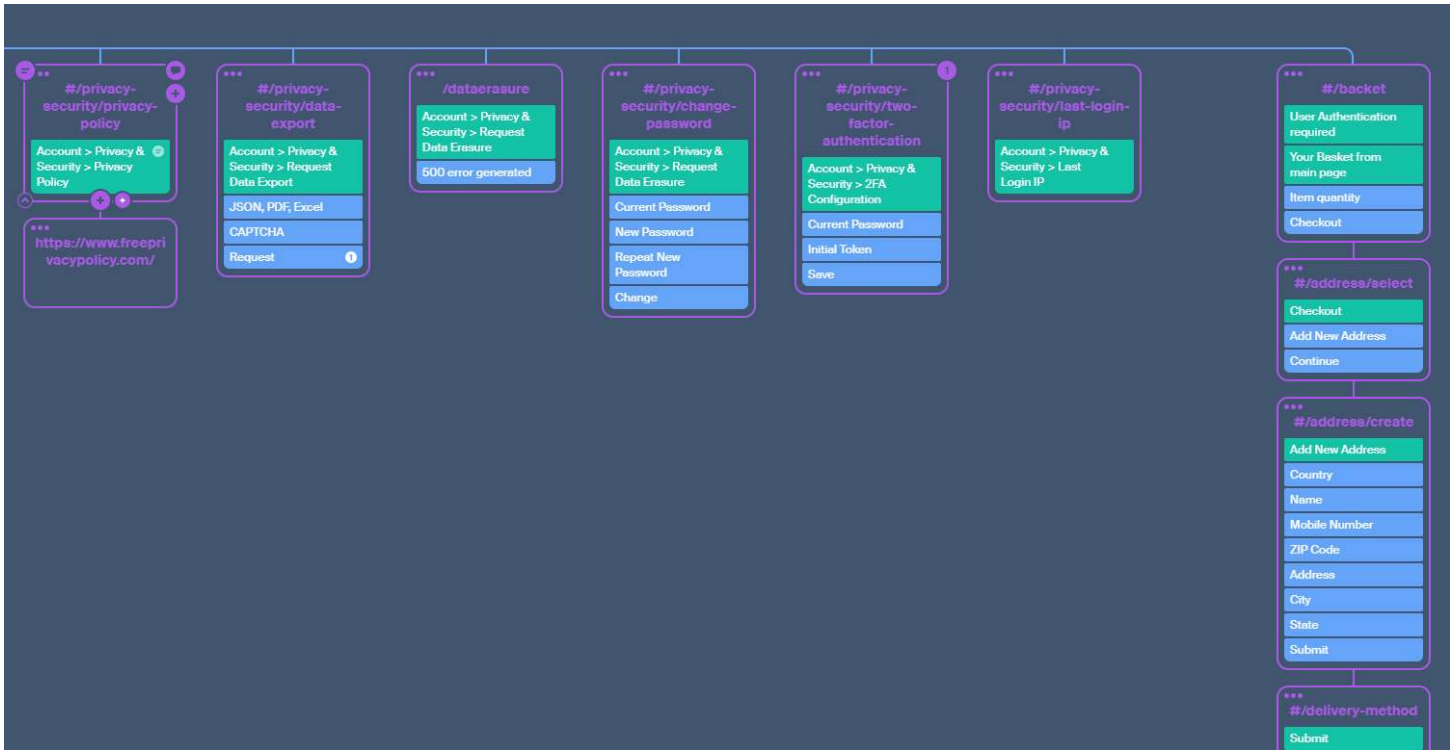
Site Mapping

Manual inspection located the following pages as an unauthenticated user.



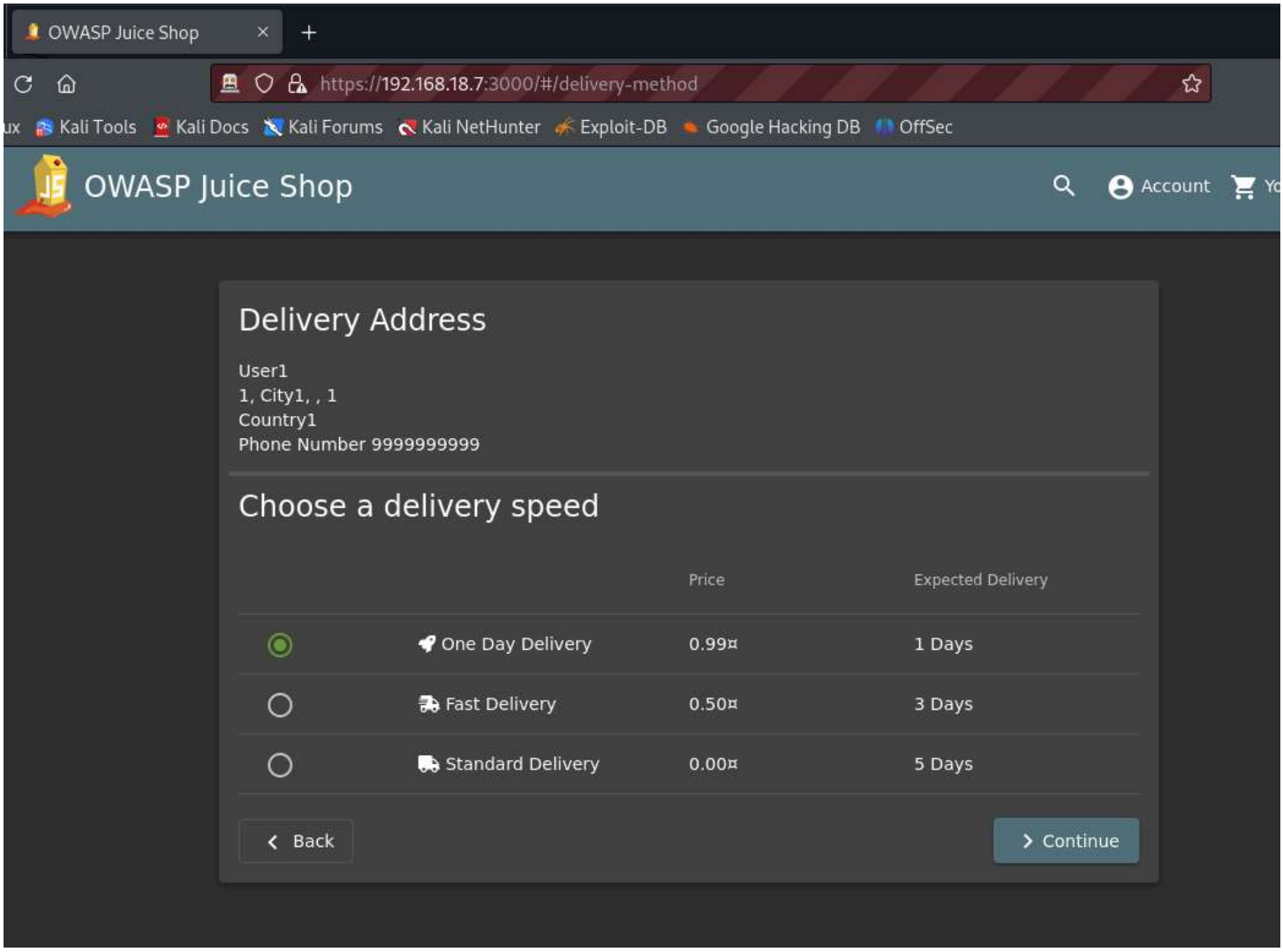
As an authenticated user, additional pages were accessible, noted by the purple box. The features to make a complaint, speak with a chatbot, purchase deluxe membership, add shopping items to the basket are now available.

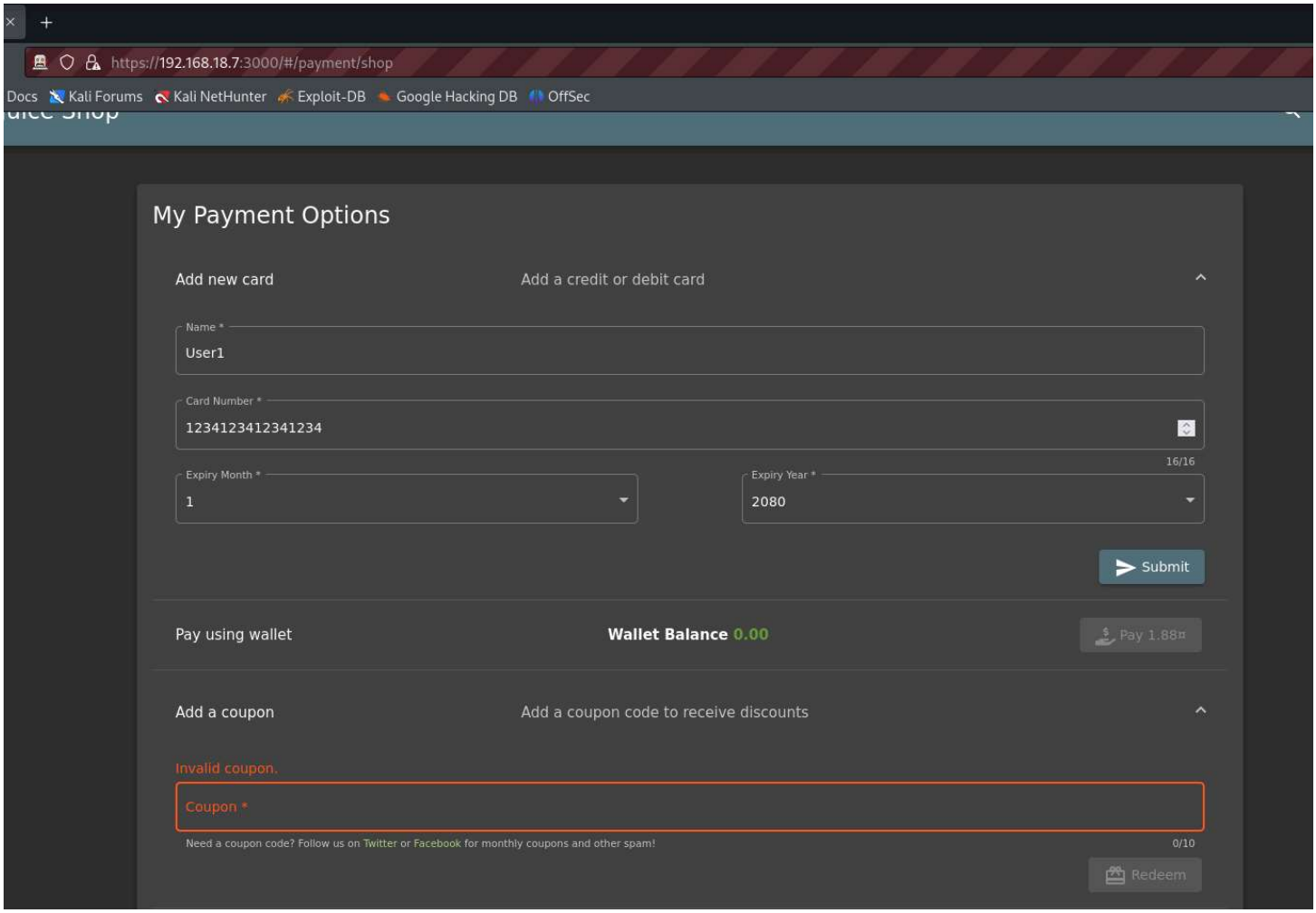


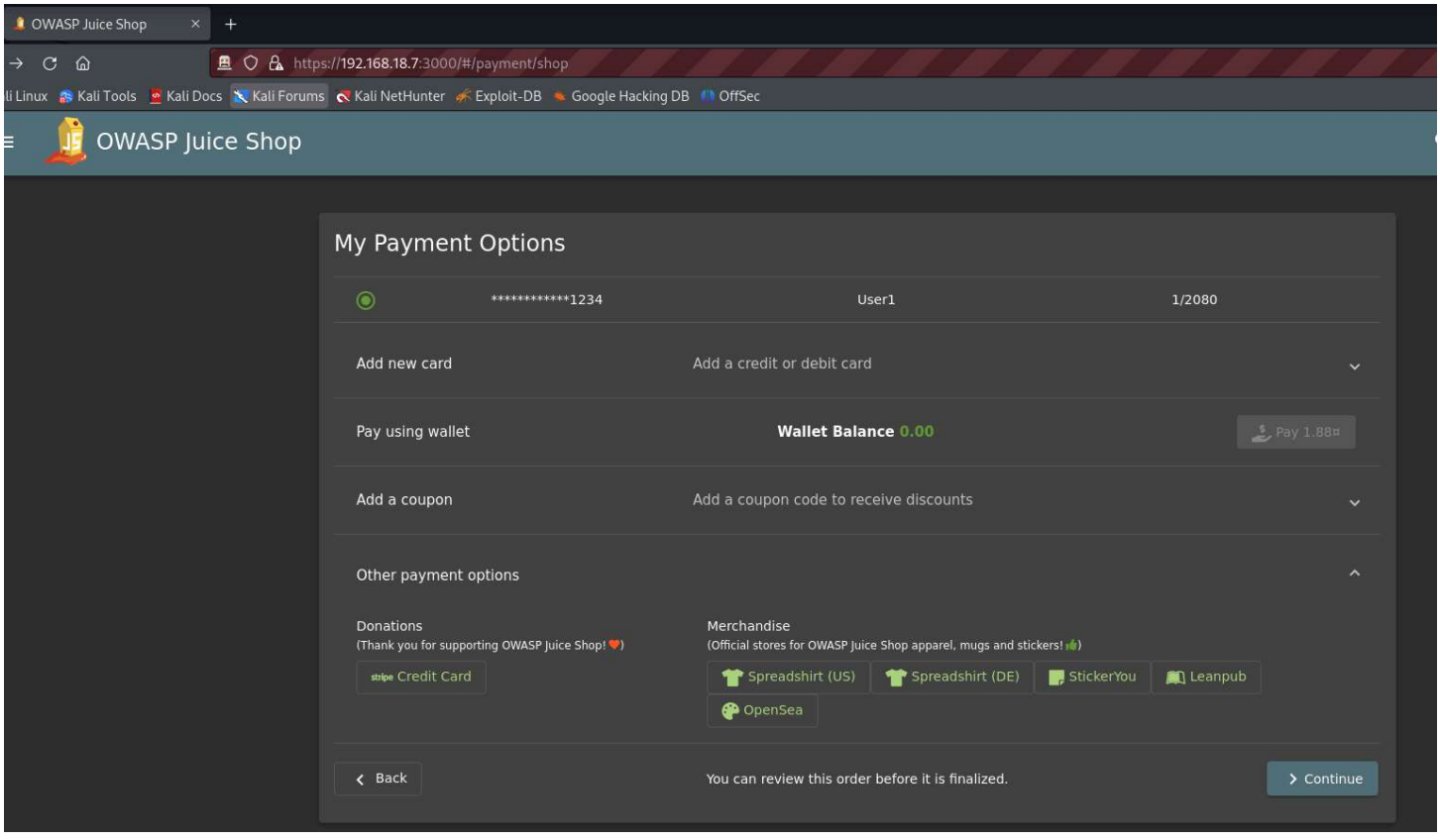


Using OWASP ZAP http://192.168.18.7:3000/ was spidered. The traditional spider and AJAX spider were both used, although only the traditional spider found additional pages.

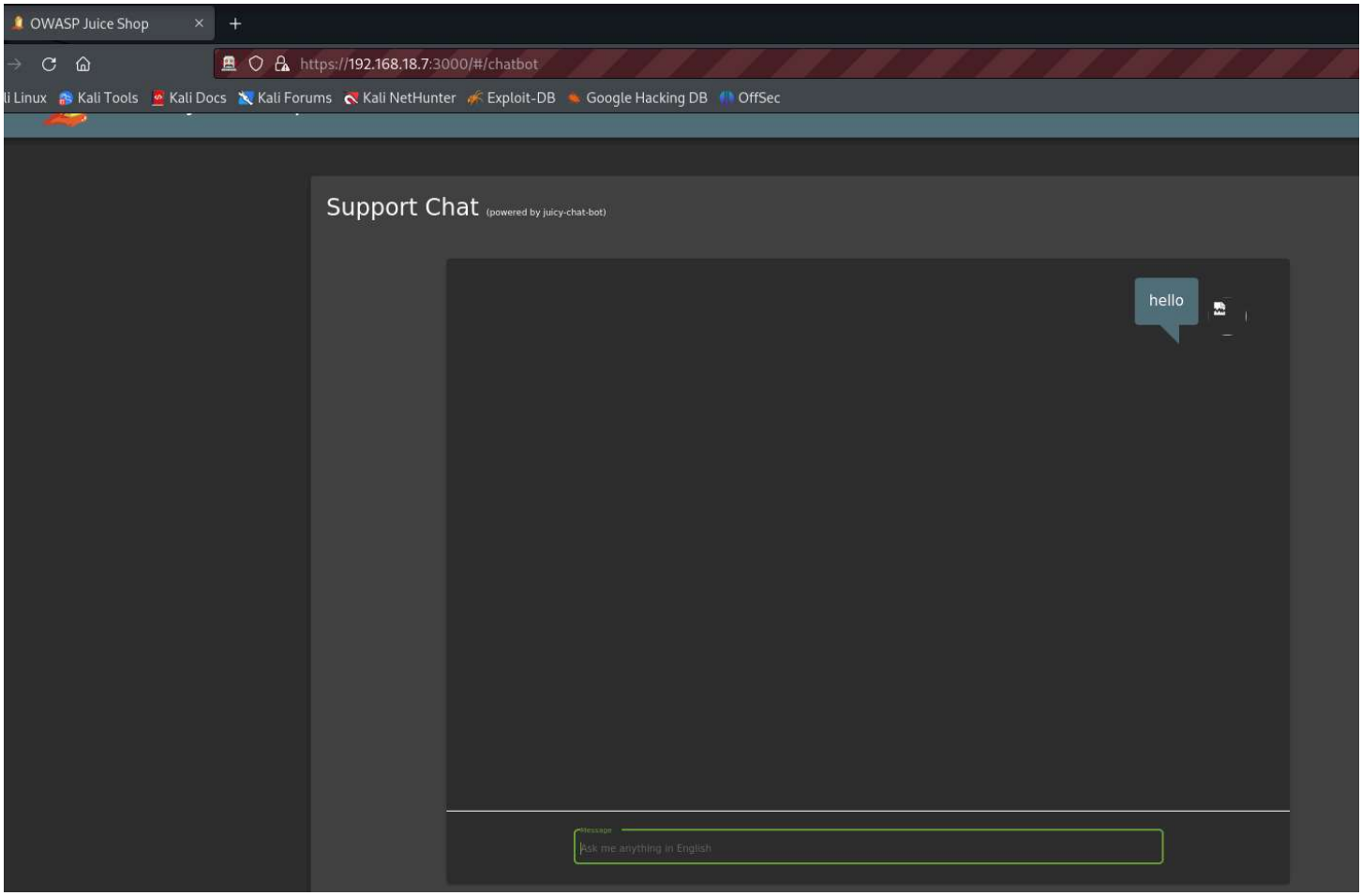
Purchase Process




https://192.168.18.7:3000/#/payment/shop
Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
My Payment Options
Add new card
Add a credit or debit card
Name *
User1
Card Number *
1234123412341234
16/16
Expiry Month *
1
Expiry Year *
2080
Submit
Pay using wallet
Wallet Balance 0.00
Pay 1.88¤
Add a coupon
Add a coupon code to receive discounts
Invalid coupon.
Coupon *
Need a coupon code? Follow us on Twitter or Facebook for monthly coupons and other spam!
0/10
Redeem

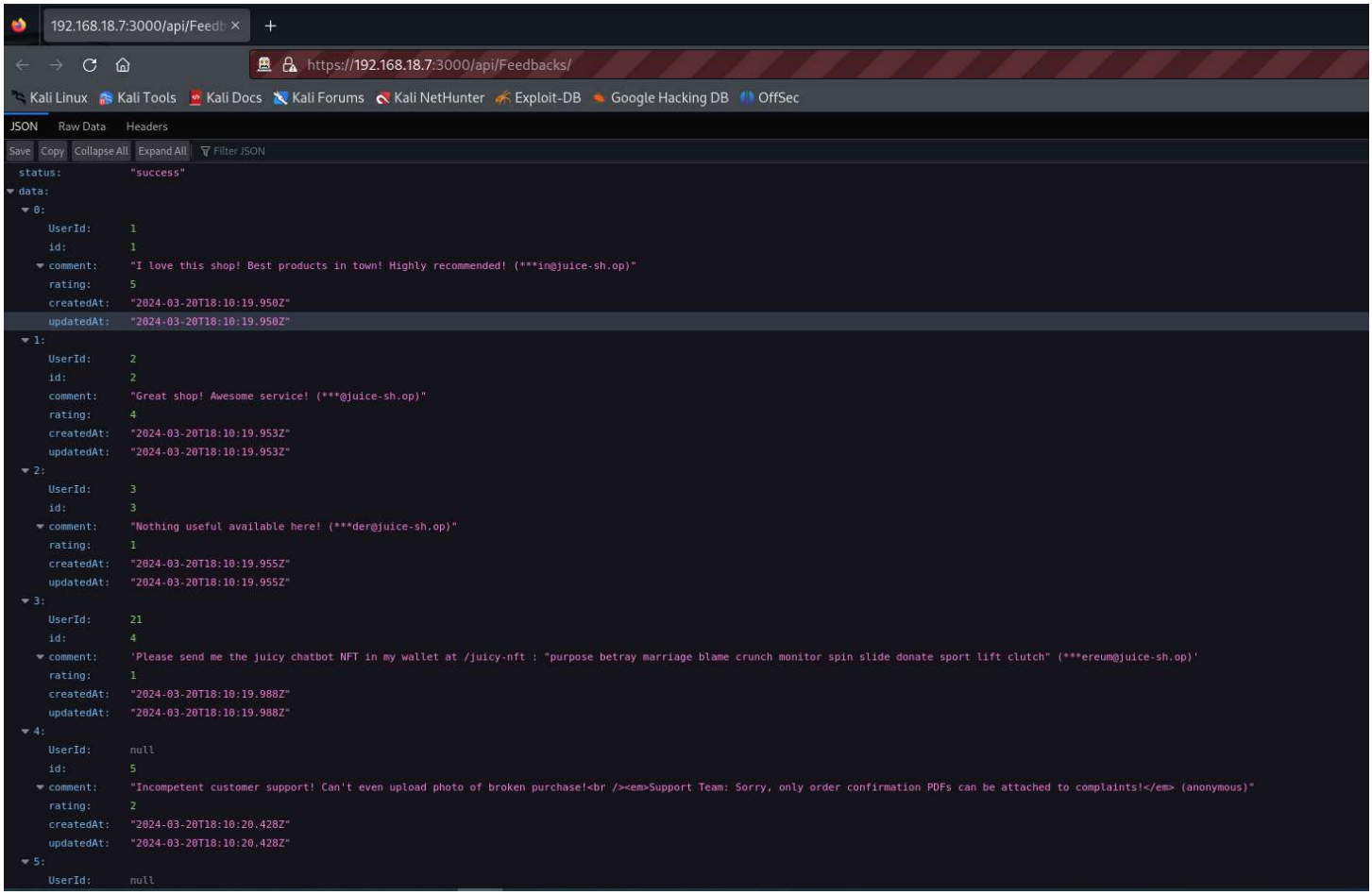

OWASP Juice Shop
https://192.168.18.7:3000/#/payment/shop
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
OWASP Juice Shop
My Payment Options
**********1234
User1
1/2080
Add new card
Add a credit or debit card
Pay using wallet
Wallet Balance 0.00
Pay 1.88¤
Add a coupon
Add a coupon code to receive discounts
Other payment options
Donations
(Thank you for supporting OWASP Juice Shop!)
Credit Card
Merchandise
(Official stores for OWASP Juice Shop apparel, mugs and stickers!)
Spreadshirt (US)
Spreadshirt (DE)
StickerYou
Leanpub
OpenSea
Back
You can review this order before it is finalized.
Continue

Chatbot

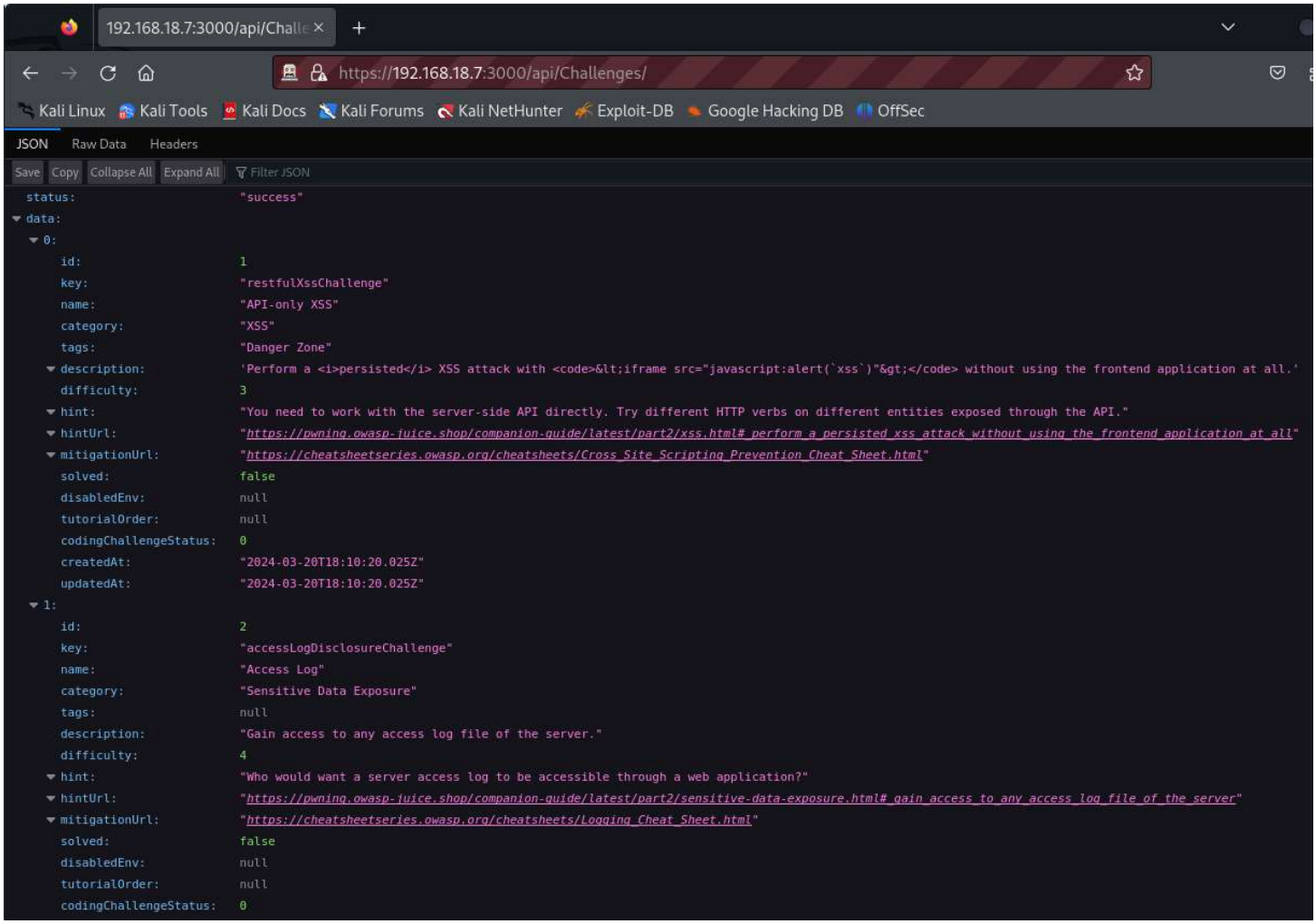


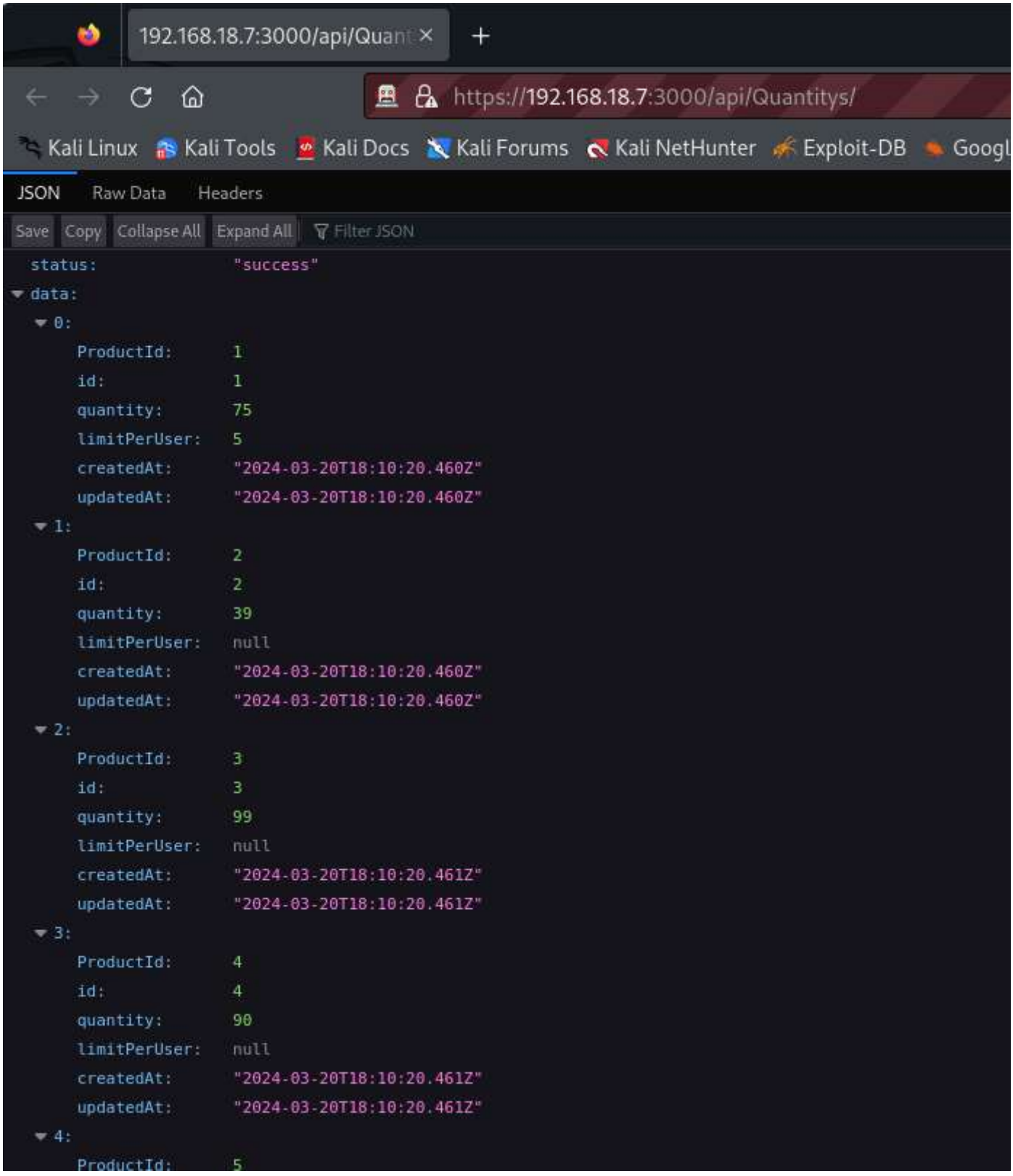
API

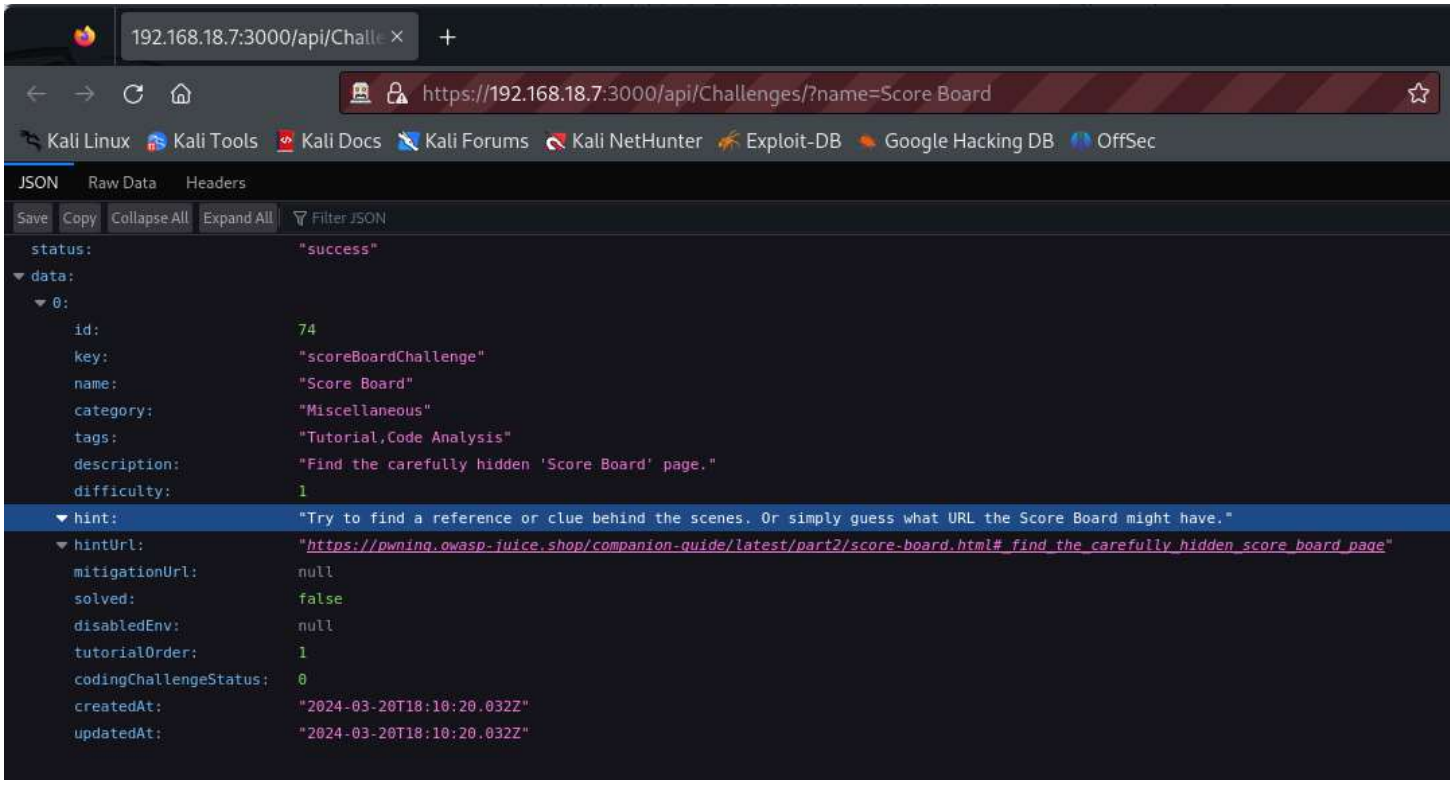
The path API contained several pages that had no access control and displayed inappropriate level information. The /api/Feedbacks page contained UserID.

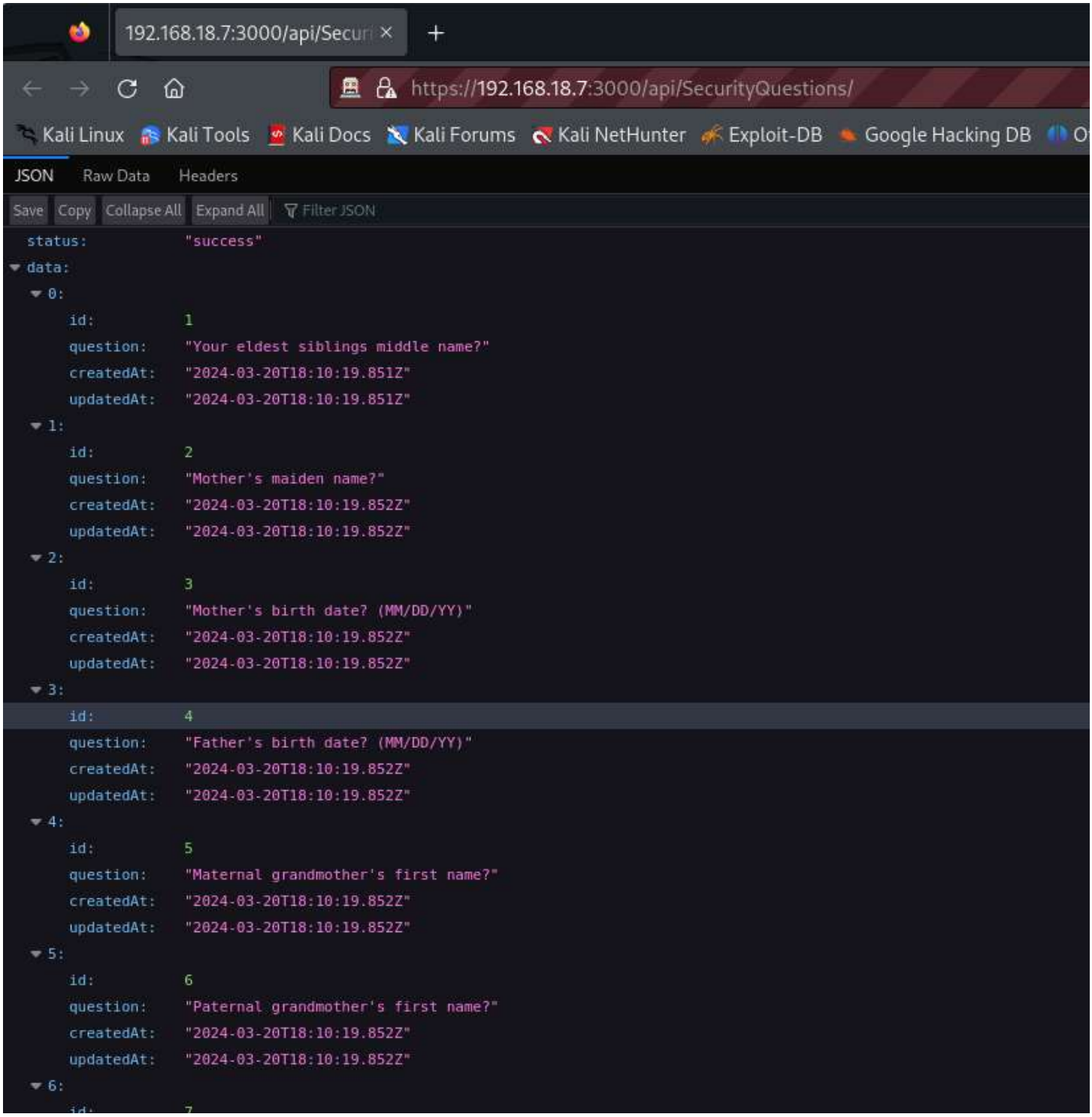


Finding	Severity	Description
	Info/TBD?	The presence of User ID could allow for user enumeration.

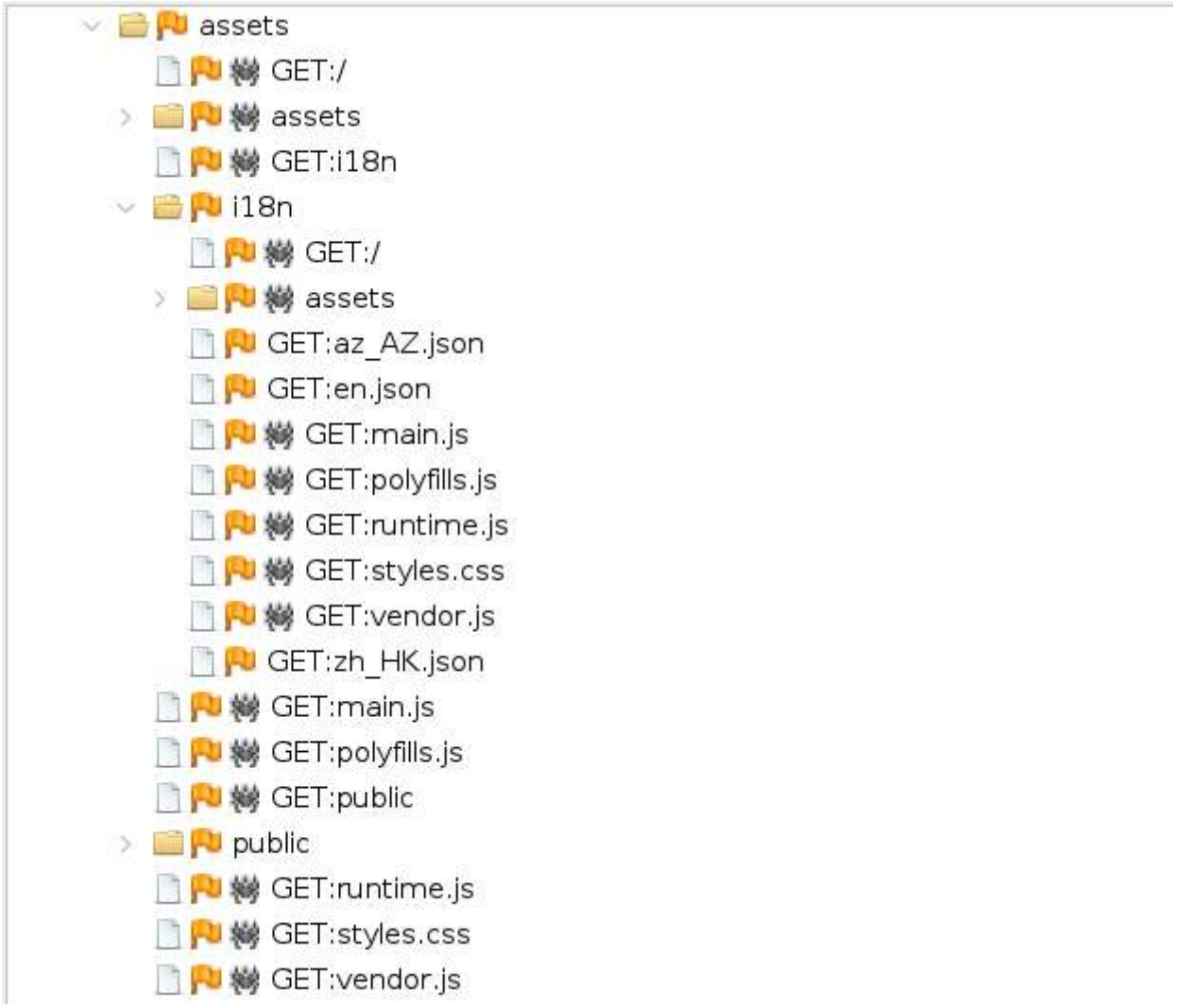

192.168.18.7:3000/api/Challe
https://192.168.18.7:3000/api/Challenges/
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All
Filter JSON
status: "success"
data:
0:
id: 1
key: "restfulXssChallenge"
name: "API-only XSS"
category: "XSS"
tags: "Danger Zone"
description: 'Perform a persisted XSS attack with <code><iframe src="javascript:alert(`xss`)"></code> without using the frontend application at all.'
difficulty: 3
hint: "You need to work with the server-side API directly. Try different HTTP verbs on different entities exposed through the API."
hintUrl: "https://pwning.owasp-juice.shop/companion-guide/latest/part2/xss.html#_perform_a_persisted_xss_attack_without_using_the_frontend_application_at_all"
mitigationUrl: "https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html"
solved: false
disabledEnv: null
tutorialOrder: null
codingChallengeStatus: 0
createdAt: "2024-03-20T18:10:20.025Z"
updatedAt: "2024-03-20T18:10:20.025Z"
1:
id: 2
key: "accessLogDisclosureChallenge"
name: "Access Log"
category: "Sensitive Data Exposure"
tags: null
description: "Gain access to any access log file of the server."
difficulty: 4
hint: "Who would want a server access log to be accessible through a web application?"
hintUrl: "https://pwning.owasp-juice.shop/companion-guide/latest/part2/sensitive-data-exposure.html#_gain_access_to_any_access_log_file_of_the_server"
mitigationUrl: "https://cheatsheetseries.owasp.org/cheatsheets/Logging_Cheat_Sheet.html"
solved: false
disabledEnv: null
tutorialOrder: null
codingChallengeStatus: 0


192.168.18.7:3000/api/Quant
https://192.168.18.7:3000/api/Quantitys/
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All
Filter JSON
status: "success"
data:
0:
ProductId: 1
id: 1
quantity: 75
limitPerUser: 5
createdAt: "2024-03-20T18:10:20.460Z"
updatedAt: "2024-03-20T18:10:20.460Z"
1:
ProductId: 2
id: 2
quantity: 39
limitPerUser: null
createdAt: "2024-03-20T18:10:20.460Z"
updatedAt: "2024-03-20T18:10:20.460Z"
2:
ProductId: 3
id: 3
quantity: 99
limitPerUser: null
createdAt: "2024-03-20T18:10:20.461Z"
updatedAt: "2024-03-20T18:10:20.461Z"
3:
ProductId: 4
id: 4
quantity: 90
limitPerUser: null
createdAt: "2024-03-20T18:10:20.461Z"
updatedAt: "2024-03-20T18:10:20.461Z"
4:
ProductId: 5


192.168.18.7:3000/api/Challe
https://192.168.18.7:3000/api/Challenges/?name=Score Board
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All
Filter JSON
status: "success"
data:
0:
id: 74
key: "scoreBoardChallenge"
name: "Score Board"
category: "Miscellaneous"
tags: "Tutorial,Code Analysis"
description: "Find the carefully hidden 'Score Board' page."
difficulty: 1
hint: "Try to find a reference or clue behind the scenes. Or simply guess what URL the Score Board might have."
hintUrl: "https://pwning.owasp-juice.shop/companion-guide/latest/part2/score-board.html#_find_the_carefully_hidden_score_board_page"
mitigationUrl: null
solved: false
disabledEnv: null
tutorialOrder: 1
codingChallengeStatus: 0
createdAt: "2024-03-20T18:10:20.032Z"
updatedAt: "2024-03-20T18:10:20.032Z"



An /assets/ directory was found without any interesting findings.

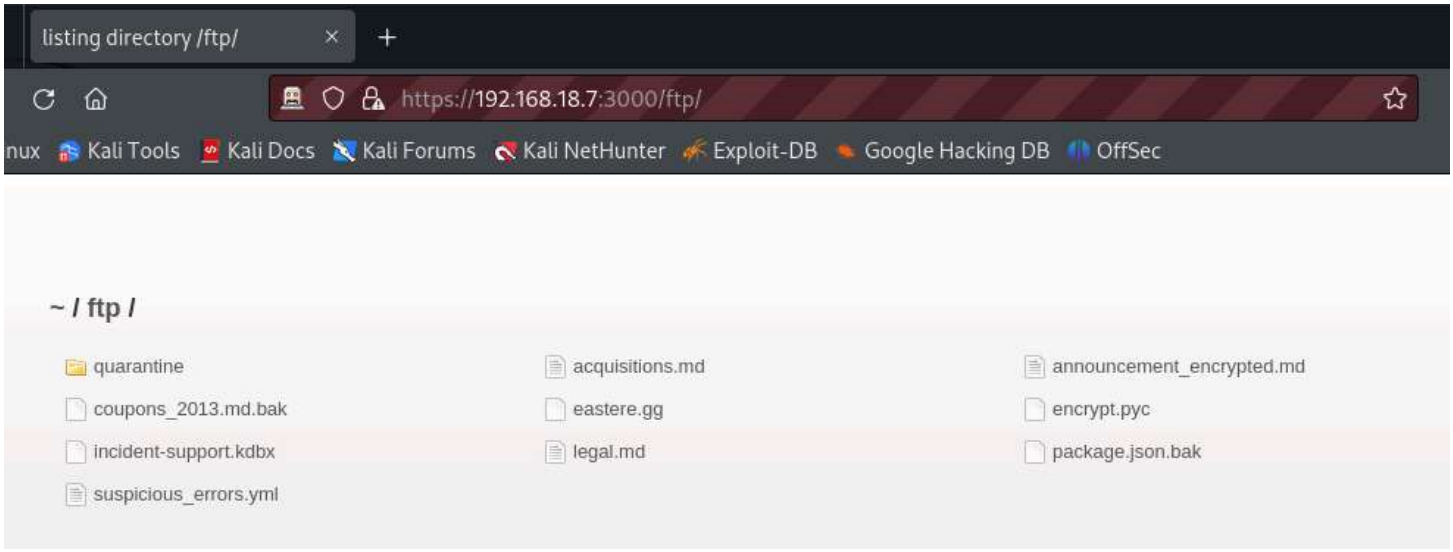


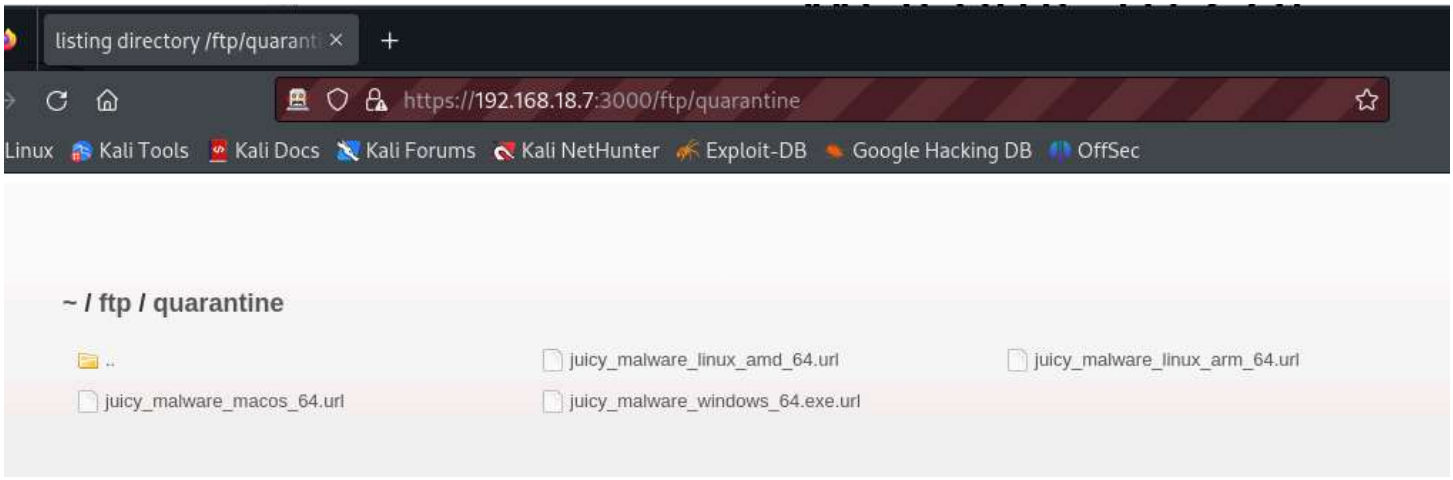
FTP

The ftp directory was discovered

- GET:ftp
- ftp
  - GET:..\..
  - GET:/
  - GET:acquisitions.md
  - GET:announcement_encrypted.md
  - GET:coupons_2013.md.bak
  - GET:eastere.gg
  - GET:encrypt.pyc
  - GET:incident-support.kdbx
  - GET:legal.md
  - GET:package.json.bak
  - GET:quarantine
  - quarantine
    - GET:juicy_malware_linux_amd_64.url
    - GET:juicy_malware_linux_arm_64.url
    - GET:juicy_malware_macos_64.url
    - GET:juicy_malware_windows_64.exe.url
  - GET:suspicious_errors.yml
- GET:main.js
- GET:polyfills.js
- GET:rest

The ftp directory had various documents visible without any access control.

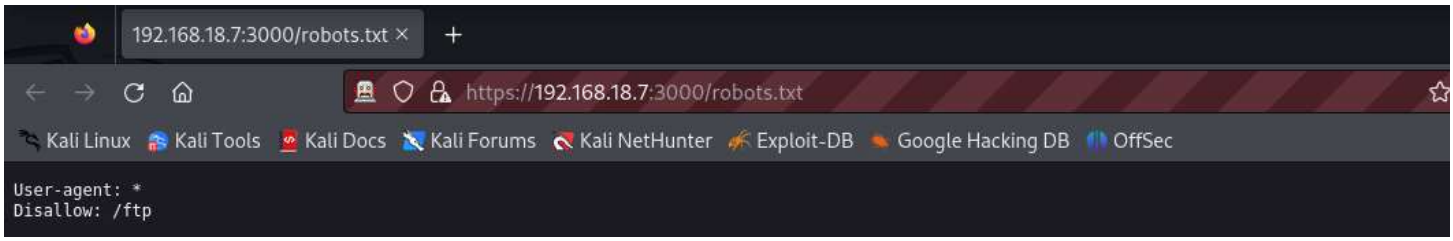




Finding	Severity	Description
	Major	Sensitive information should not be located in an area with no access control.

Finding	Severity	Description
	Minor	FTP is an insecure protocol and needs to be used with TLS or a secure channel. FTP should be replaced with a secure alternative such as SFTP.

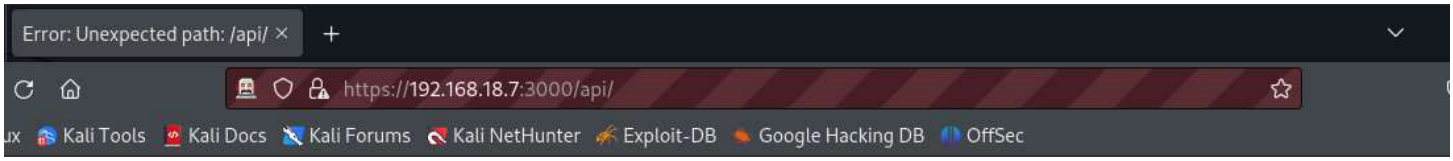
The ftp directory is also referenced in /robots.txt which can attract attention to the directory.



Finding	Severity	Description
	Minor	/robots.txt should not reference locations with sensitive information

Error pages and other pages

A 500 error page from unexpected path shows a lot of information, including that it is built using Angular. A generic error page should be used instead.



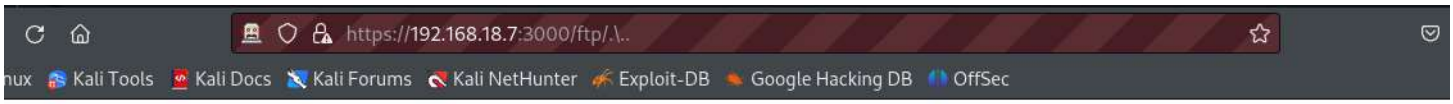
OWASP Juice Shop (Express ^4.17.1)

500 **Error: Unexpected path: /api/**

at C:\Users\ronli\Documents\juice-shop_16.0.0\build\routes\angular.js:38:18
at Layer.handle [as handle_request] (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\layer.js:95:5)
at trim_prefix (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:328:13)
at C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:286:9
at Function.process_params (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:346:12)
at next (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:280:10)
at C:\Users\ronli\Documents\juice-shop_16.0.0\build\routes\verify.js:168:5
at Layer.handle [as handle_request] (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\layer.js:95:5)
at trim_prefix (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:328:13)
at C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:286:9
at Function.process_params (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:346:12)
at next (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:280:10)
at C:\Users\ronli\Documents\juice-shop_16.0.0\build\routes\verify.js:105:5
at Layer.handle [as handle_request] (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\layer.js:95:5)
at trim_prefix (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:328:13)
at C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:286:9
at Function.process_params (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:346:12)
at next (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:280:10)
at logger (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\morgan\index.js:144:5)
at Layer.handle [as handle_request] (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\layer.js:95:5)
at trim_prefix (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:328:13)
at C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:286:9

Finding	Severity	Description
	Minor	Error messages should not contain information on the application or the server. A generic error message should be used.

A 403 error page displayed similar issues



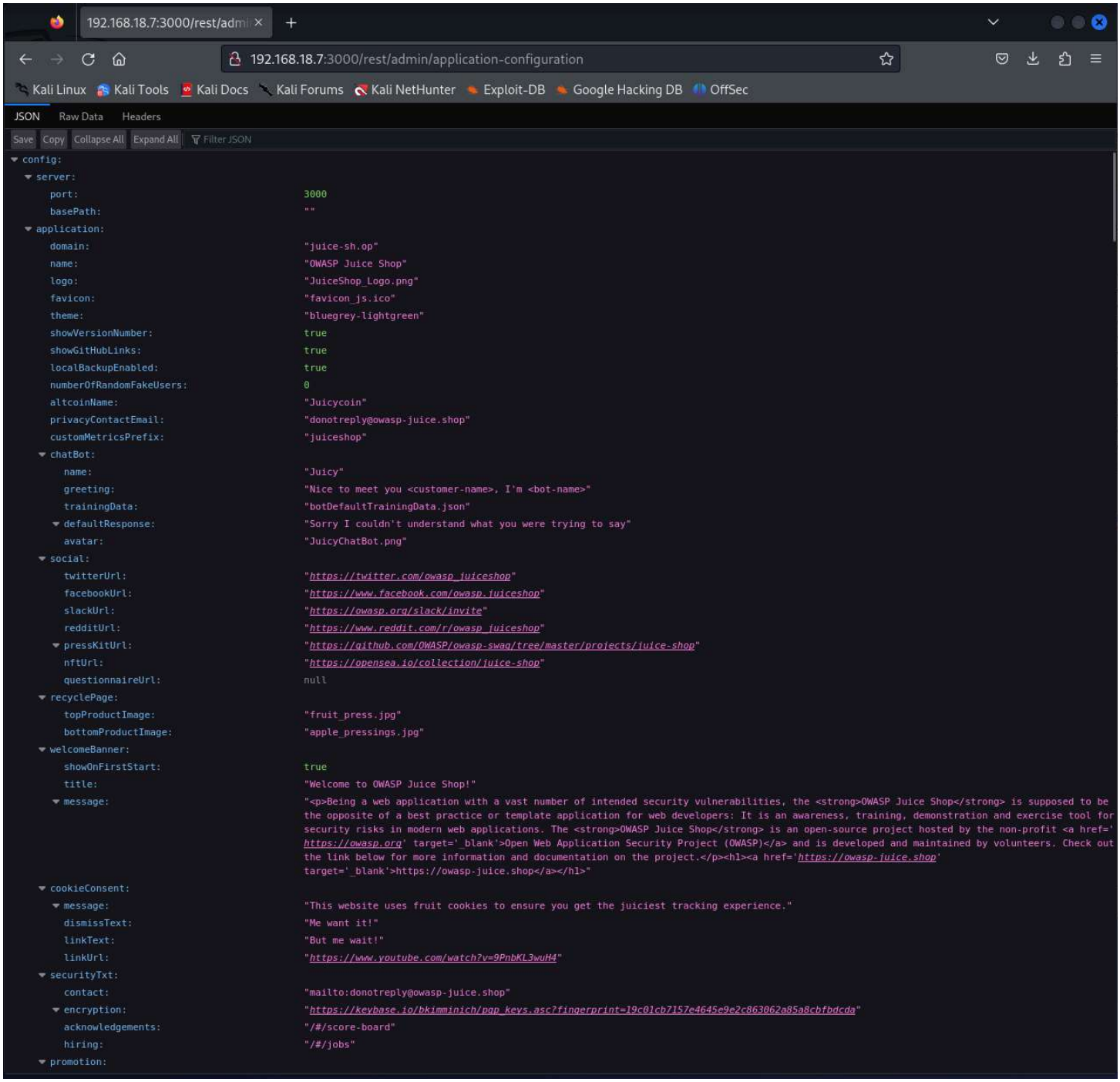
OWASP Juice Shop (Express ^4.17.1)

403 **ForbiddenError: Forbidden**

at C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\serve-index\index.js:125:19
at Layer.handle [as handle_request] (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\layer.js:95:5)
at trim_prefix (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:328:13)
at C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:286:9
at Function.process_params (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:346:12)
at next (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:280:10)
at serveIndexMiddleware (C:\Users\ronli\Documents\juice-shop_16.0.0\build\server.js:250:9)
at Layer.handle [as handle_request] (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\layer.js:95:5)
at trim_prefix (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:328:13)
at C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:286:9
at Function.process_params (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:346:12)
at next (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:280:10)
at C:\Users\ronli\Documents\juice-shop_16.0.0\build\lib\antiCheat.js:71:5
at Layer.handle [as handle_request] (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\layer.js:95:5)
at trim_prefix (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:328:13)
at C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:286:9
at Function.process_params (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:346:12)
at next (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:280:10)
at C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:646:15
at next (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:265:14)
at Function.handle (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:175:3)
at router (C:\Users\ronli\Documents\juice-shop_16.0.0\node_modules\express\lib\router\index.js:47:12)

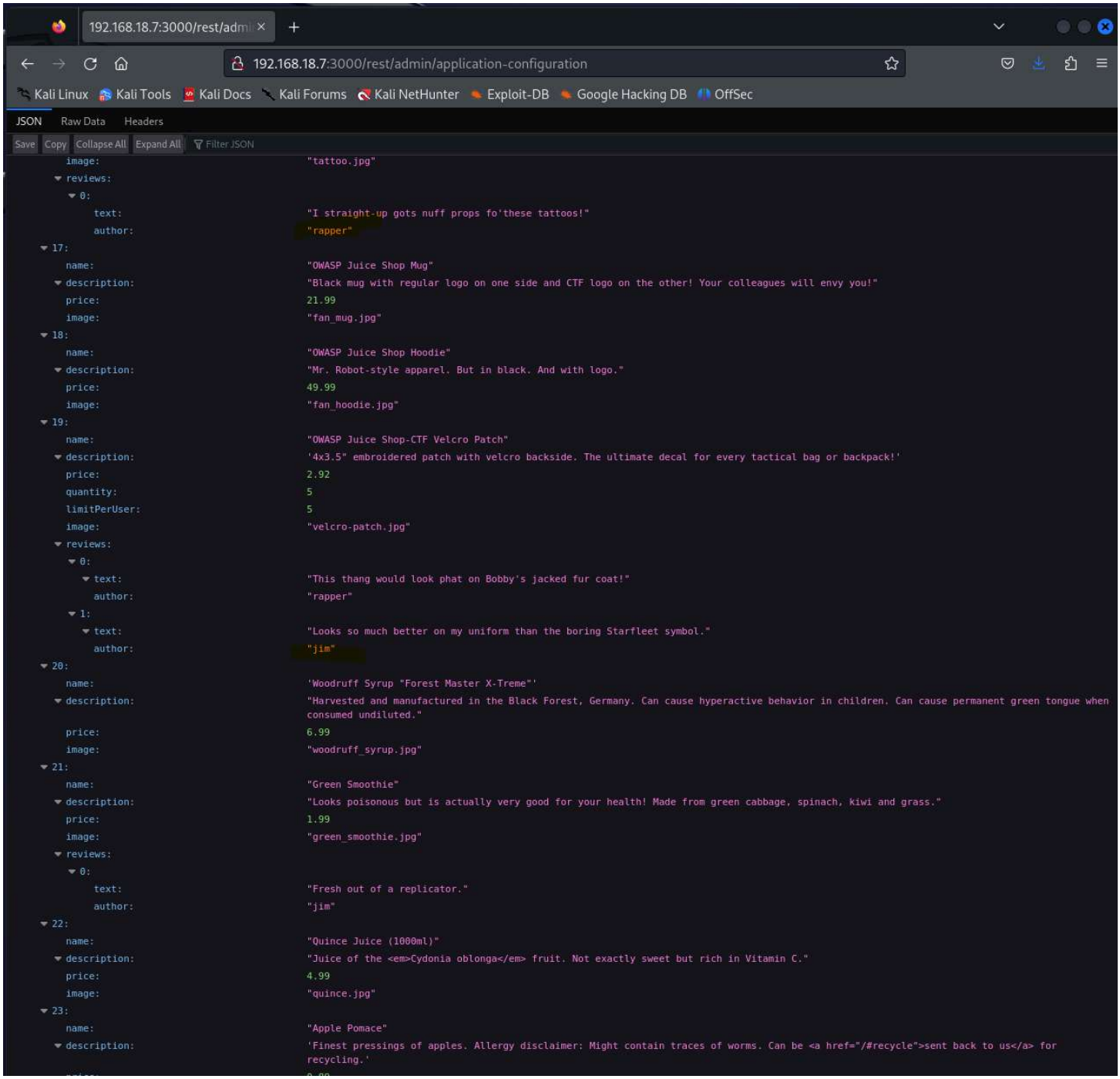
No issues were found with /get, /rest and /user directory.
Additional miscellaneous pages were found and shown in the screenshot below.

When loading the application, an /admin/application-configuration could be found displaying json data.



User information can be found in the file, including:

- admin
- rapper
- jim
- uvogin
- morty
- bjoernOwasp
- bender
- stan
- accountant
- bjoernGoogle (user)



Discovering Hidden Directories

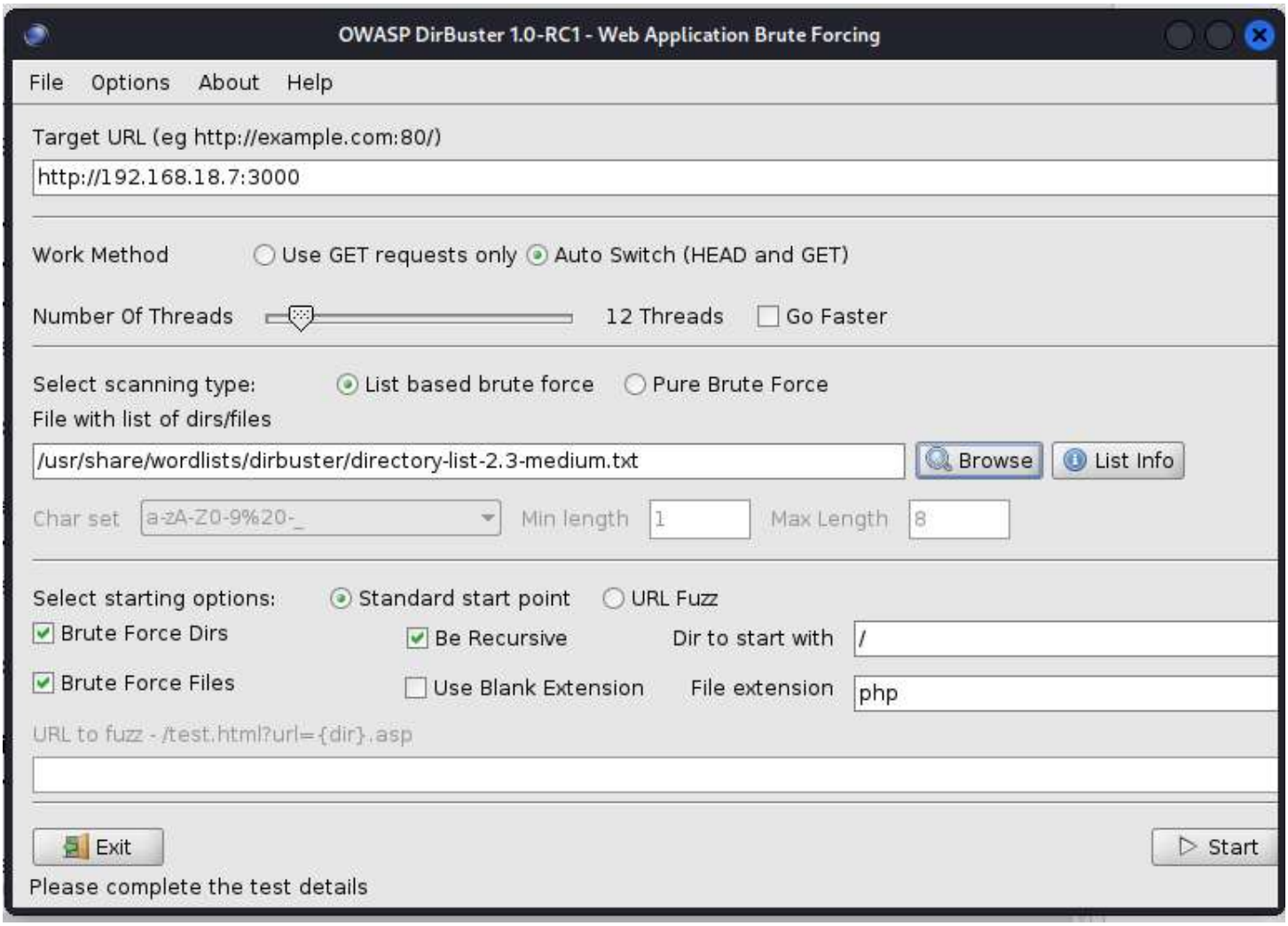
The Forced Browse Directory (and Children) feature was used with ZAP. With the tool, the following directories were discovered:

- /video
- /promotion

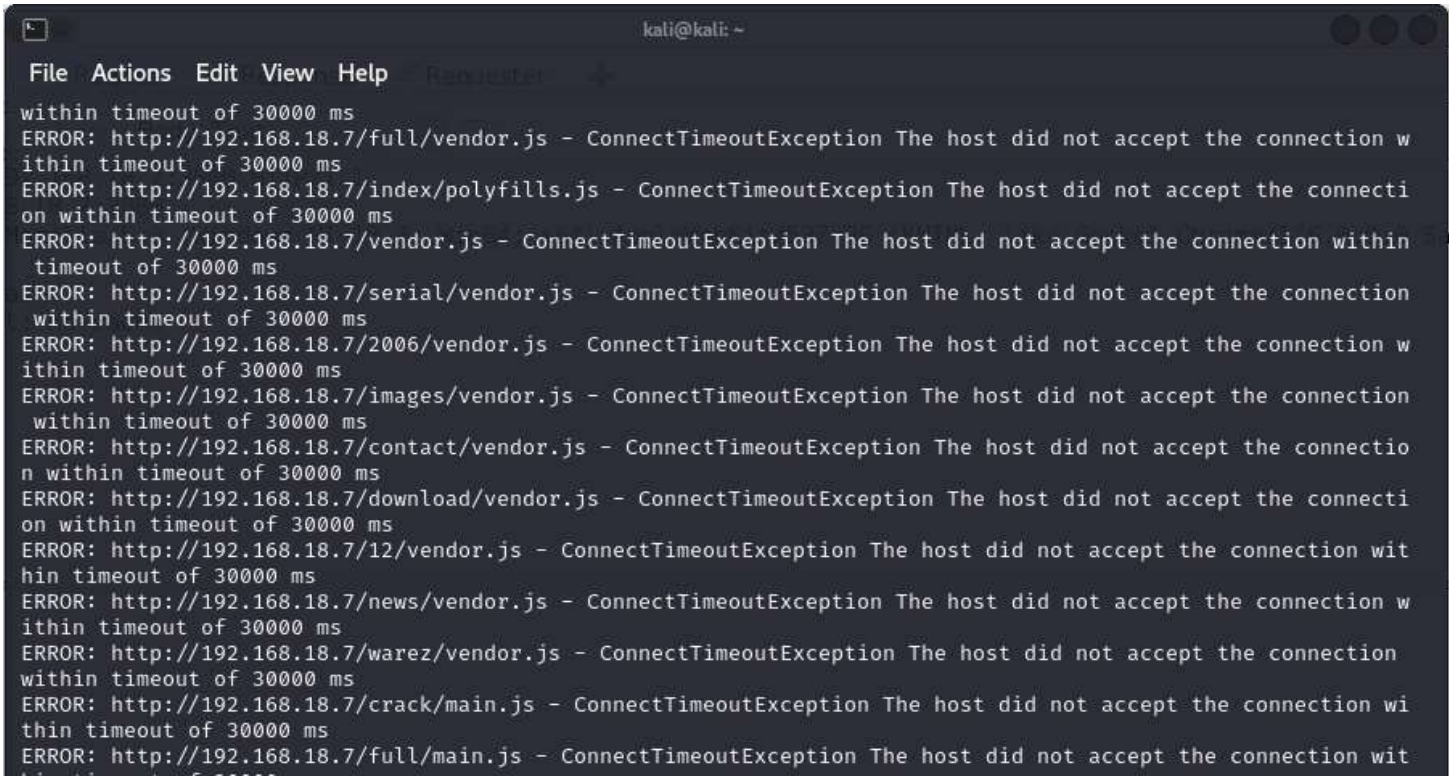
Testing could not be completed as rapidly browsing directories occasionally caused the web application to crash, either by an error caused by visiting a directory, or an error due to insufficient resources. The test was aborted at about 3% completion.

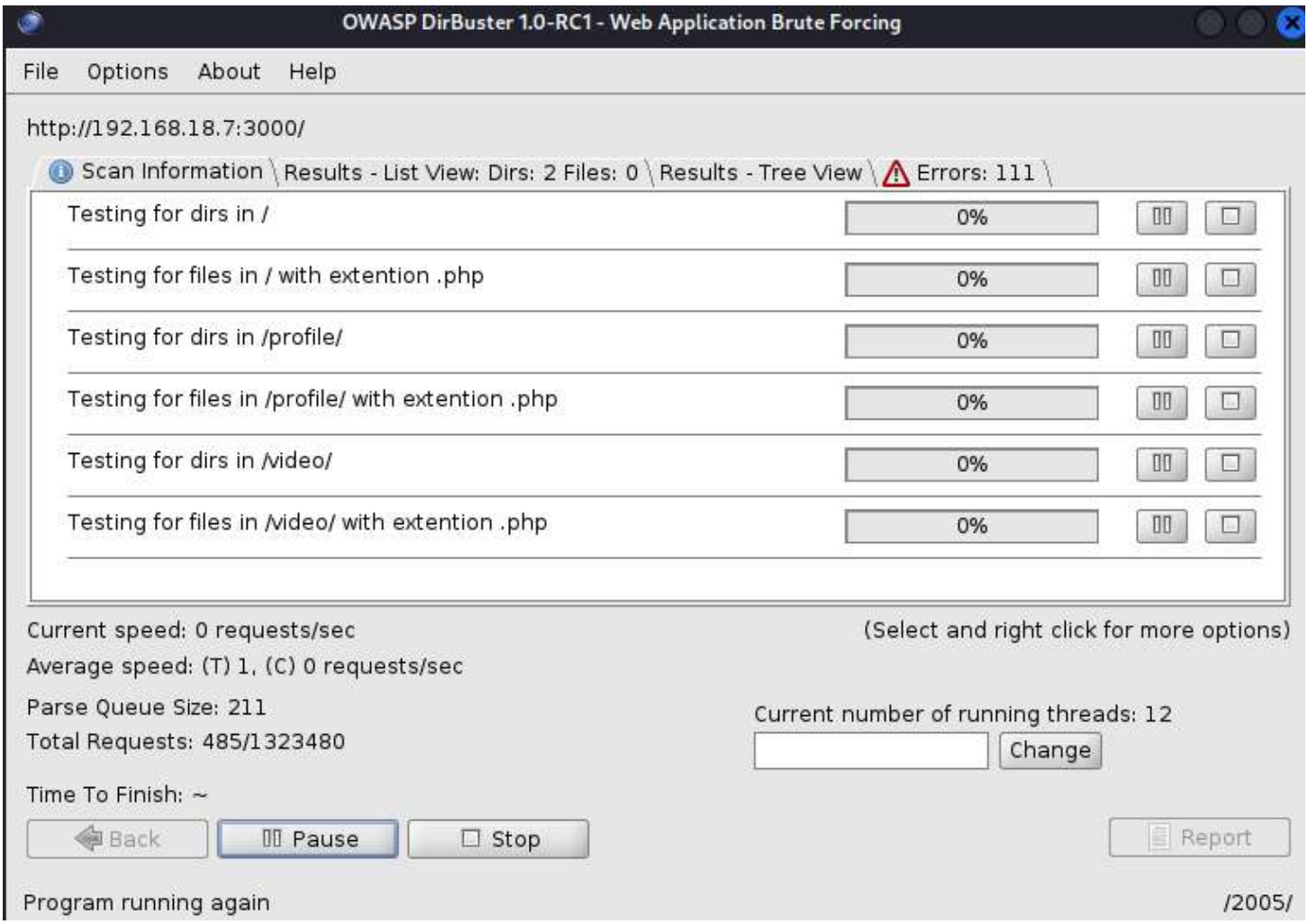
```
FATAL ERROR: Ineffective mark-compacts near heap limit Allocation failed - JavaScript heap out of memory
----- Native stack trace -----

 1: 00007FF76E0B71AB node::SetCppgcReference+16075
 2: 00007FF76E02DCC6 v8::base::CPU::num_virtual_address_bits+79190
 3: 00007FF76E02FED5 v8::base::CPU::num_virtual_address_bits+87909
 4: 00007FF76EA9F061 v8::Isolate::ReportExternalAllocationLimitReached+65
 5: 00007FF76EA887F8 v8::Function::Experimental_IsNopFunction+1336
 6: 00007FF76E8EA120 v8::Platform::SystemClockTimeMillis+659328
 7: 00007FF76E8F63A3 v8::Platform::SystemClockTimeMillis+709123
 8: 00007FF76E8F3D04 v8::Platform::SystemClockTimeMillis+699236
 9: 00007FF76E8E6E40 v8::Platform::SystemClockTimeMillis+646304
10: 00007FF76E8FC4BA v8::Platform::SystemClockTimeMillis+733978
11: 00007FF76E8FCD37 v8::Platform::SystemClockTimeMillis+736151
12: 00007FF76E90596E v8::Platform::SystemClockTimeMillis+772046
13: 00007FF76E9162B6 v8::Platform::SystemClockTimeMillis+839958
14: 00007FF76E91AE88 v8::Platform::SystemClockTimeMillis+859368
15: 00007FF76E580AE5 v8::internal::Version::GetString+88101
16: 00007FF76E75579A v8::base::Thread::StartSynchronously+1171290
17: 00007FF76E756681 v8::base::Thread::StartSynchronously+1175105
18: 00007FF76EA39908 v8::SharedValueConveyor::SharedValueConveyor+302472
19: 00007FF76EA397B0 v8::SharedValueConveyor::SharedValueConveyor+302128
20: 00007FF70EB1A9FA
```

Dirbuster was used with the following settings:



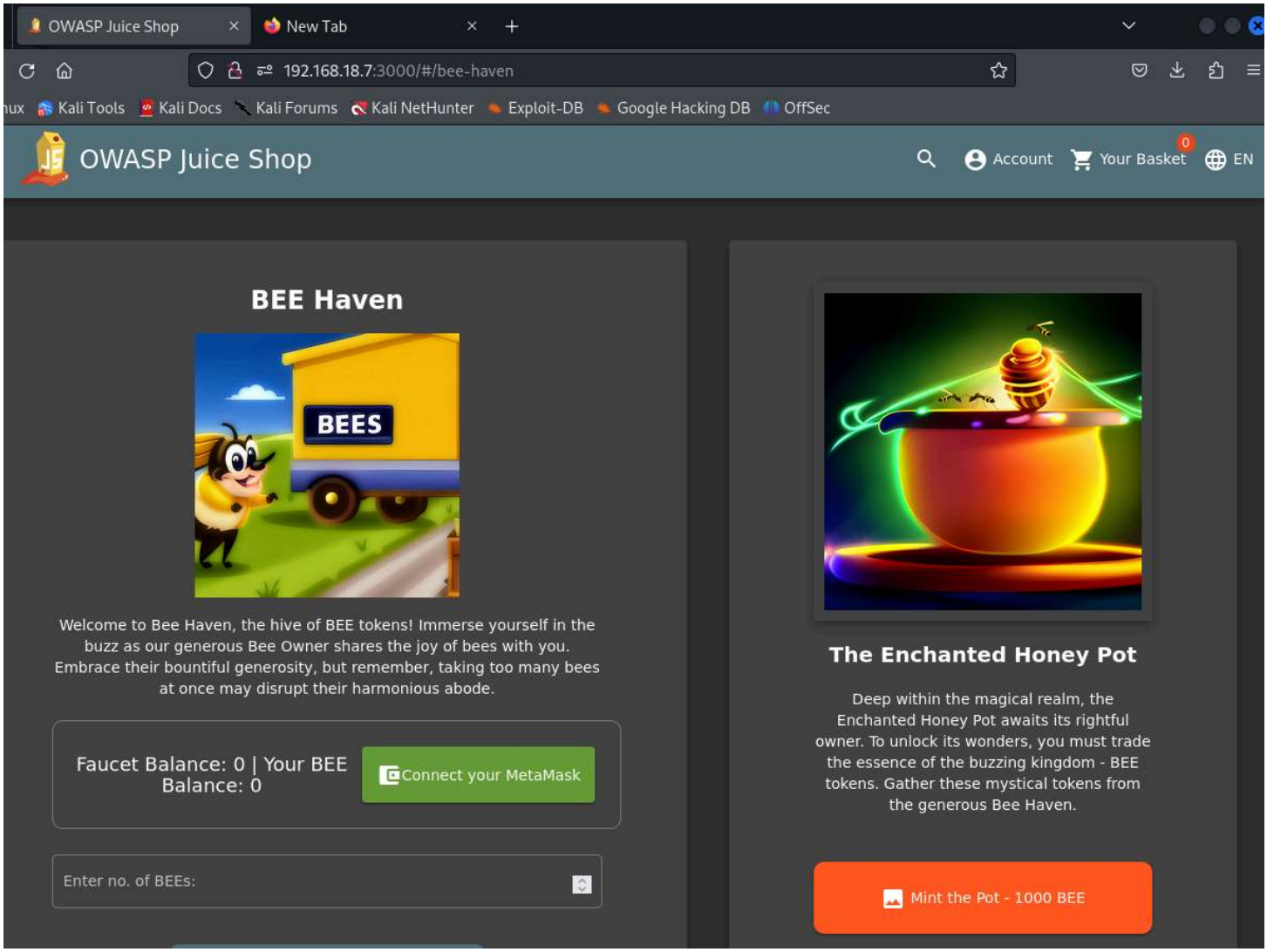
Similar to the Forced browse directory tests, frequent errors caused the test to be abandoned at about 500 requests.


kali@kali: ~
File Actions Edit View Help
within timeout of 30000 ms
ERROR: http://192.168.18.7/full/vendor.js - ConnectTimeoutException The host did not accept the connection within timeout of 30000 ms
ERROR: http://192.168.18.7/index/polyfills.js - ConnectTimeoutException The host did not accept the connection within timeout of 30000 ms
ERROR: http://192.168.18.7/vendor.js - ConnectTimeoutException The host did not accept the connection within timeout of 30000 ms
ERROR: http://192.168.18.7/serial/vendor.js - ConnectTimeoutException The host did not accept the connection within timeout of 30000 ms
ERROR: http://192.168.18.7/2006/vendor.js - ConnectTimeoutException The host did not accept the connection within timeout of 30000 ms
ERROR: http://192.168.18.7/images/vendor.js - ConnectTimeoutException The host did not accept the connection within timeout of 30000 ms
ERROR: http://192.168.18.7/contact/vendor.js - ConnectTimeoutException The host did not accept the connection within timeout of 30000 ms
ERROR: http://192.168.18.7/download/vendor.js - ConnectTimeoutException The host did not accept the connection within timeout of 30000 ms
ERROR: http://192.168.18.7/12/vendor.js - ConnectTimeoutException The host did not accept the connection within timeout of 30000 ms
ERROR: http://192.168.18.7/news/vendor.js - ConnectTimeoutException The host did not accept the connection within timeout of 30000 ms
ERROR: http://192.168.18.7/warez/vendor.js - ConnectTimeoutException The host did not accept the connection within timeout of 30000 ms
ERROR: http://192.168.18.7/crack/main.js - ConnectTimeoutException The host did not accept the connection within timeout of 30000 ms
ERROR: http://192.168.18.7/full/main.js - ConnectTimeoutException The host did not accept the connection wit


OWASP DirBuster 1.0-RC1 - Web Application Brute Forcing
File Options About Help
http://192.168.18.7:3000/
Scan Information
Results - List View: Dirs: 2 Files: 0
Results - Tree View
Errors: 111
Testing for dirs in /
0%
Testing for files in / with extention .php
0%
Testing for dirs in /profile/
0%
Testing for files in /profile/ with extention .php
0%
Testing for dirs in /video/
0%
Testing for files in /video/ with extention .php
0%
Current speed: 0 requests/sec
(Select and right click for more options)
Average speed: (T) 1, (C) 0 requests/sec
Parse Queue Size: 211
Total Requests: 485/1323480
Current number of running threads: 12
Change
Time To Finish: ~
Back
Pause
Stop
Report
Program running again
/2005/

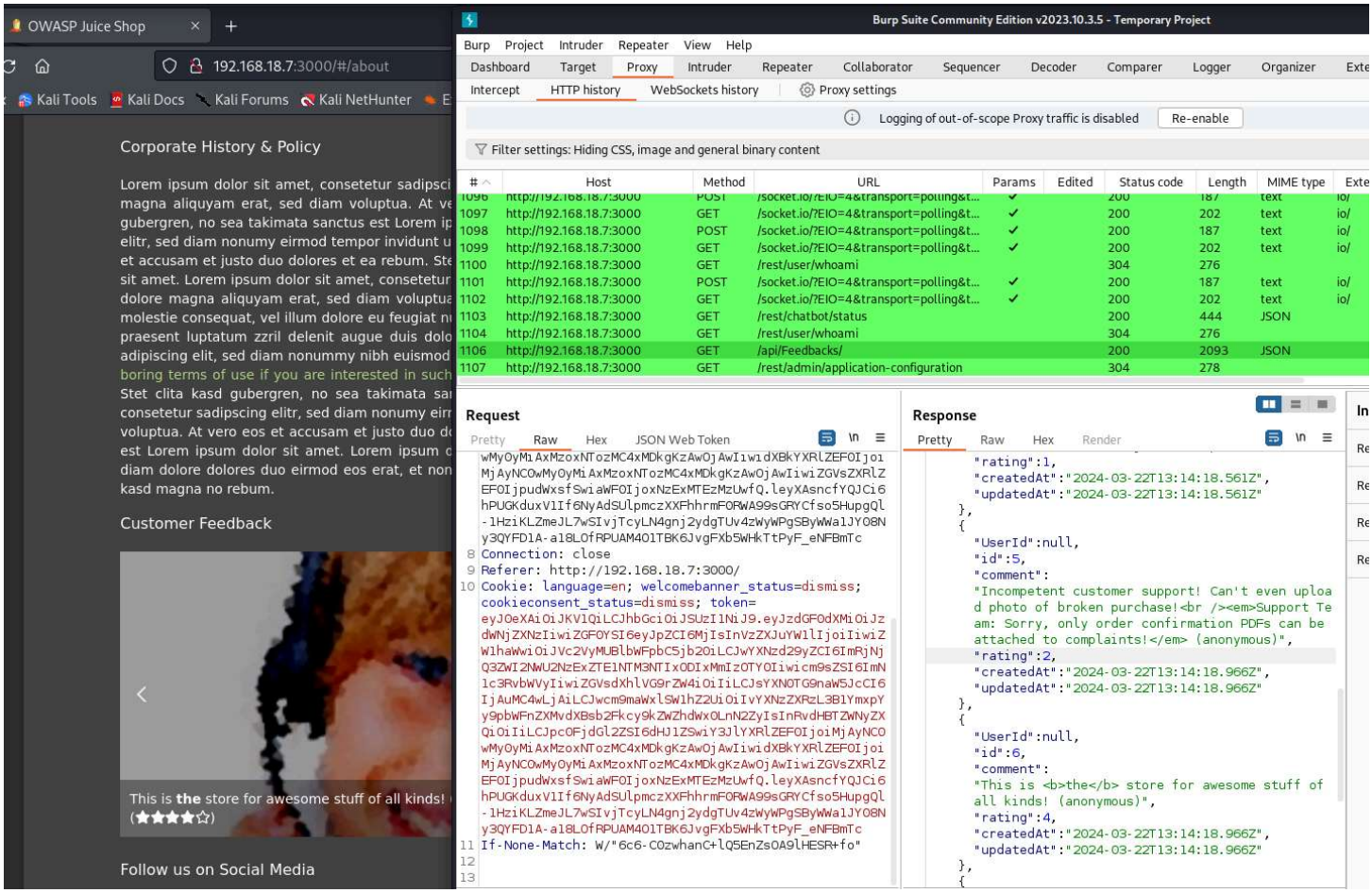
Access Control

The directories discovered were browsed to spot missing access control or additional hidden directories in the HTTP response. The #/bee-haven directory was discovered in this manner by viewing the /rest/admin/application-configuration.



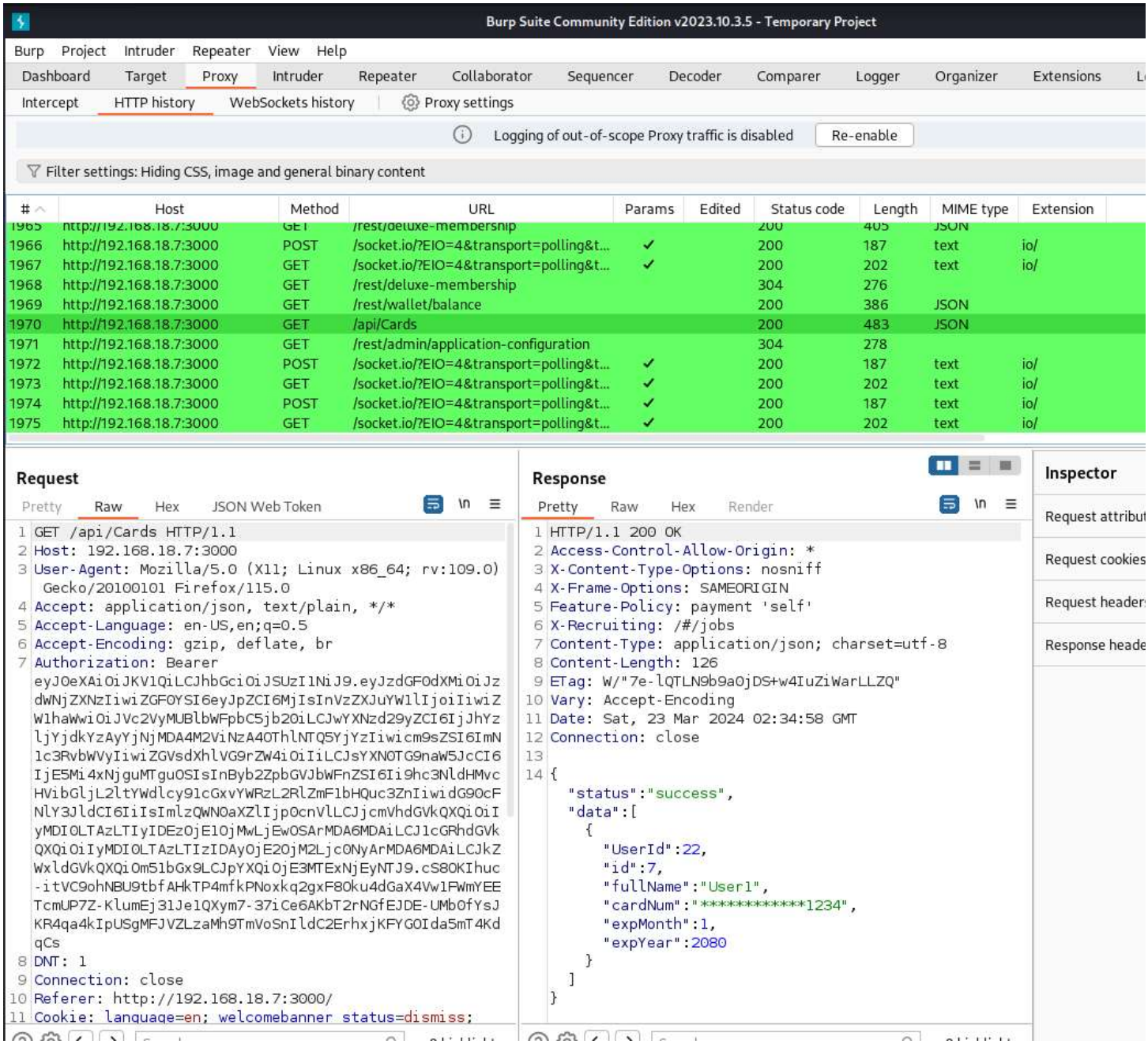
Finding	Severity	Description
	Minor	The hidden /#/bee-haven directory could be discovered through viewing HTTP responses. Implement appropriate access control for private pages.

HTTP response was inspected to see if globally unique identifiers for users were present. Browsing to the #/about page retrieves the /api/Feedbacks. The page shows user id with each comment. The comment seems to include HTML tags, with a potential for XSS.



Finding	Severity	Description
	Minor	The disclosure of user id reveals too much information to attackers.

Visiting payment options returns information about the user's card information, but no sensitive information is disclosed.



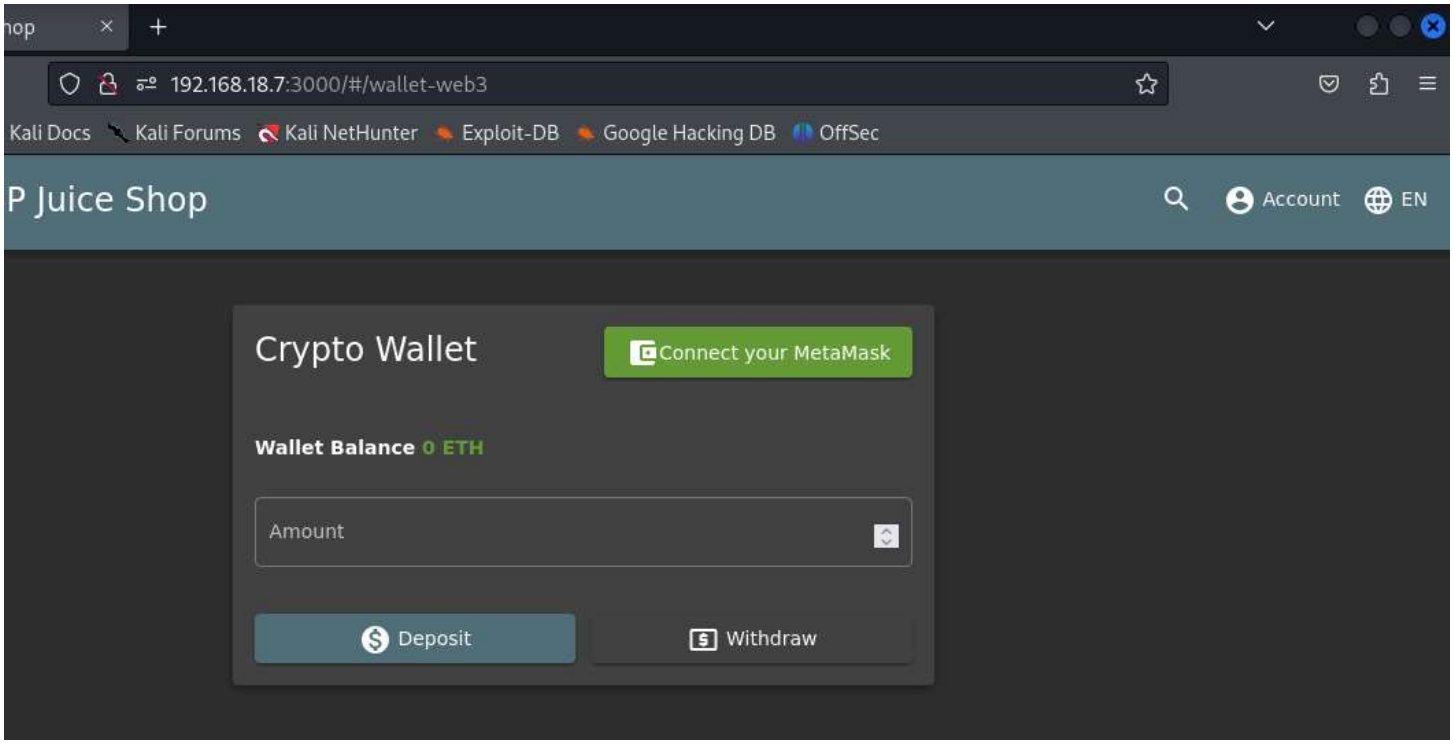
An attempt was made to visit all member only pages while unauthenticated to test for access control. Several pages could be accessed without authentication.

Finding	Severity	Description
	Minor	#/complain should not be accessible for unauthenticated users.

Most of these pages are not concerning as no actions can be performed without a linked account.

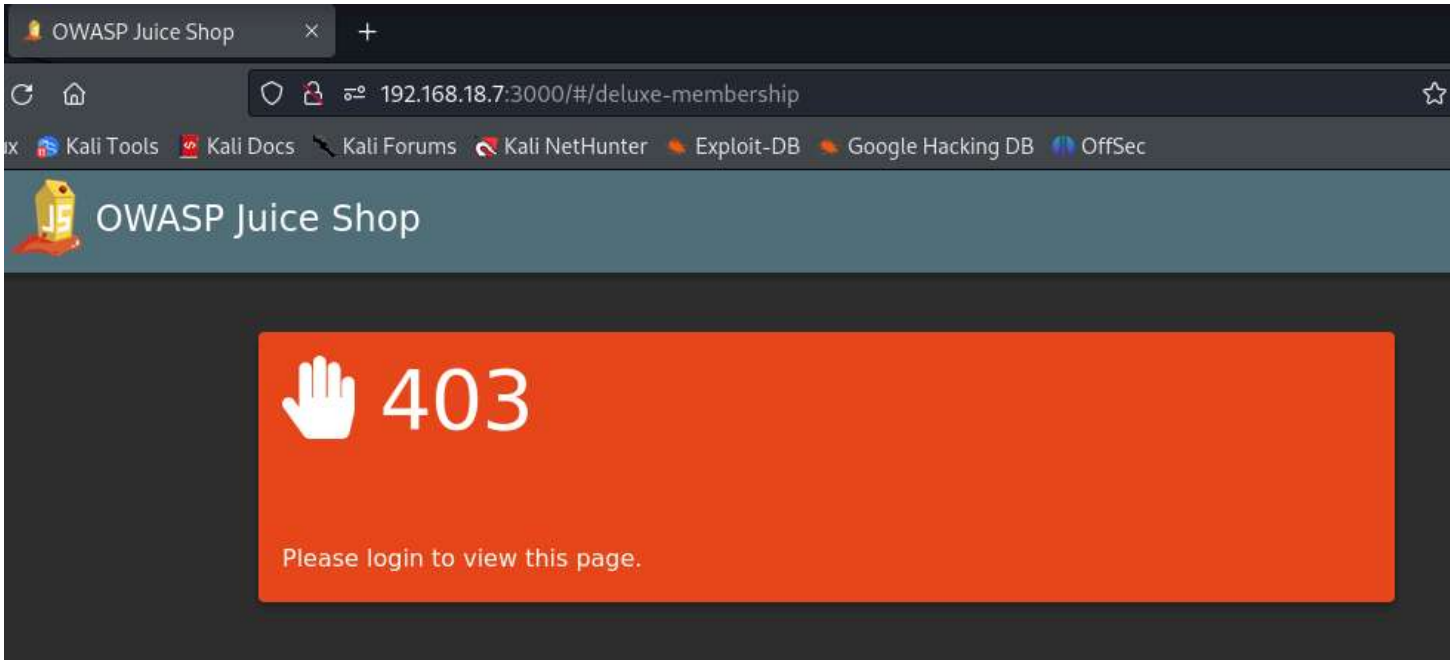
Finding	Severity	Description
	Info	These directories should not be accessible for unauthenticated users. #/chatbot #/payment/deluxe #/order-history

		#/recycle #/saved-payment-methods #/payment/shop #/payment/wallet #/wallet #/wallet-web3 #/privacy-security/privacy-policy #/privacy-security/change-password #/privacy-security/data-export #/privacy-security/last-login-ip #/privacy-security/two-factor-authentication #/basket #/delivery-method #/order-summary



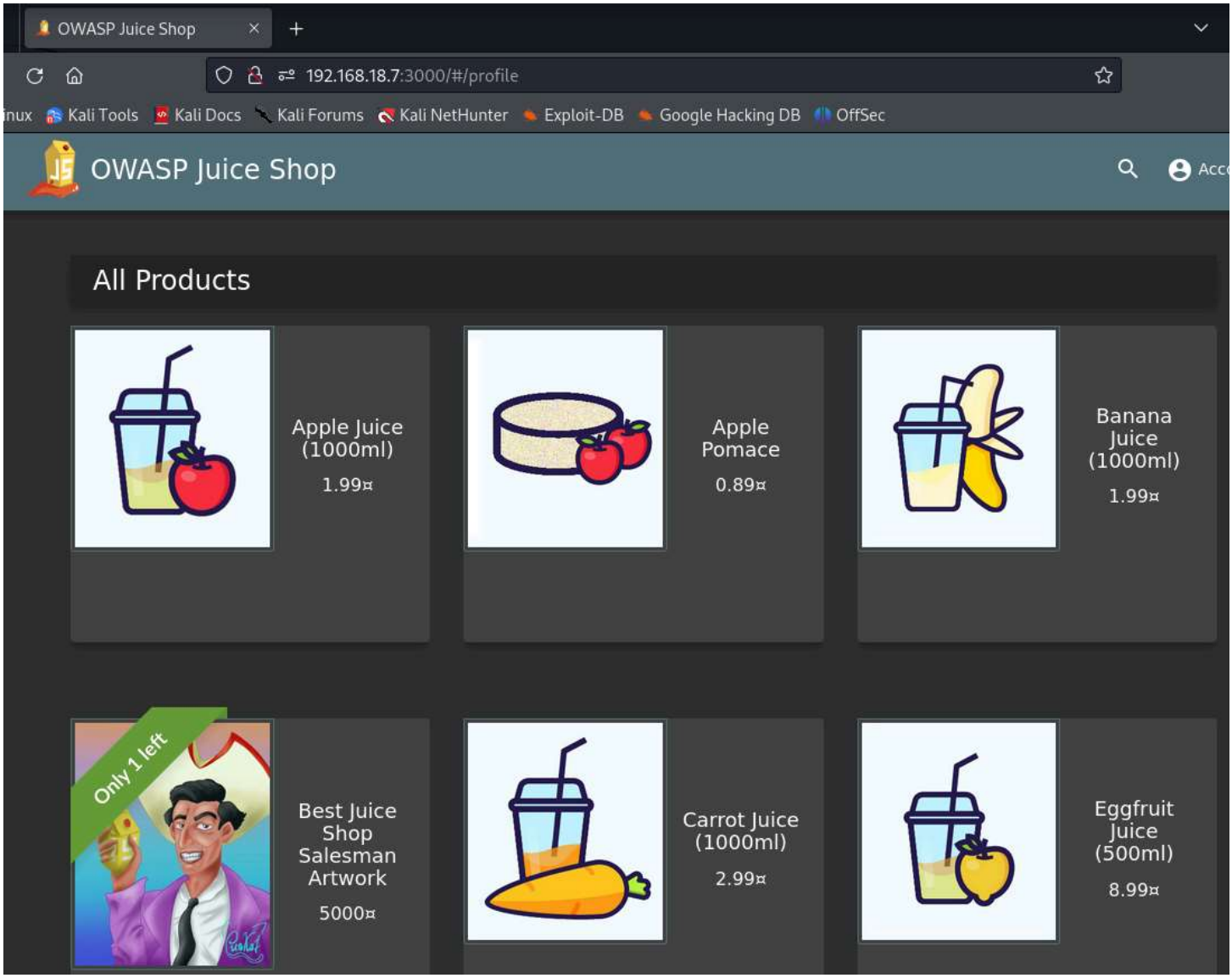
The following pages returned an appropriate access denied message:

#/deluxe-membership
#/address/saved
#/address/create
#/address/select



The following pages returned the user to the main page:

#/profile
#/address/delivery-method



Cookies were inspected to find attributes that could be tampered with to bypass access control. No issues were found while testing all directories and functionality of the web application. Further investigation into the cookie can be found in the Session Management section.

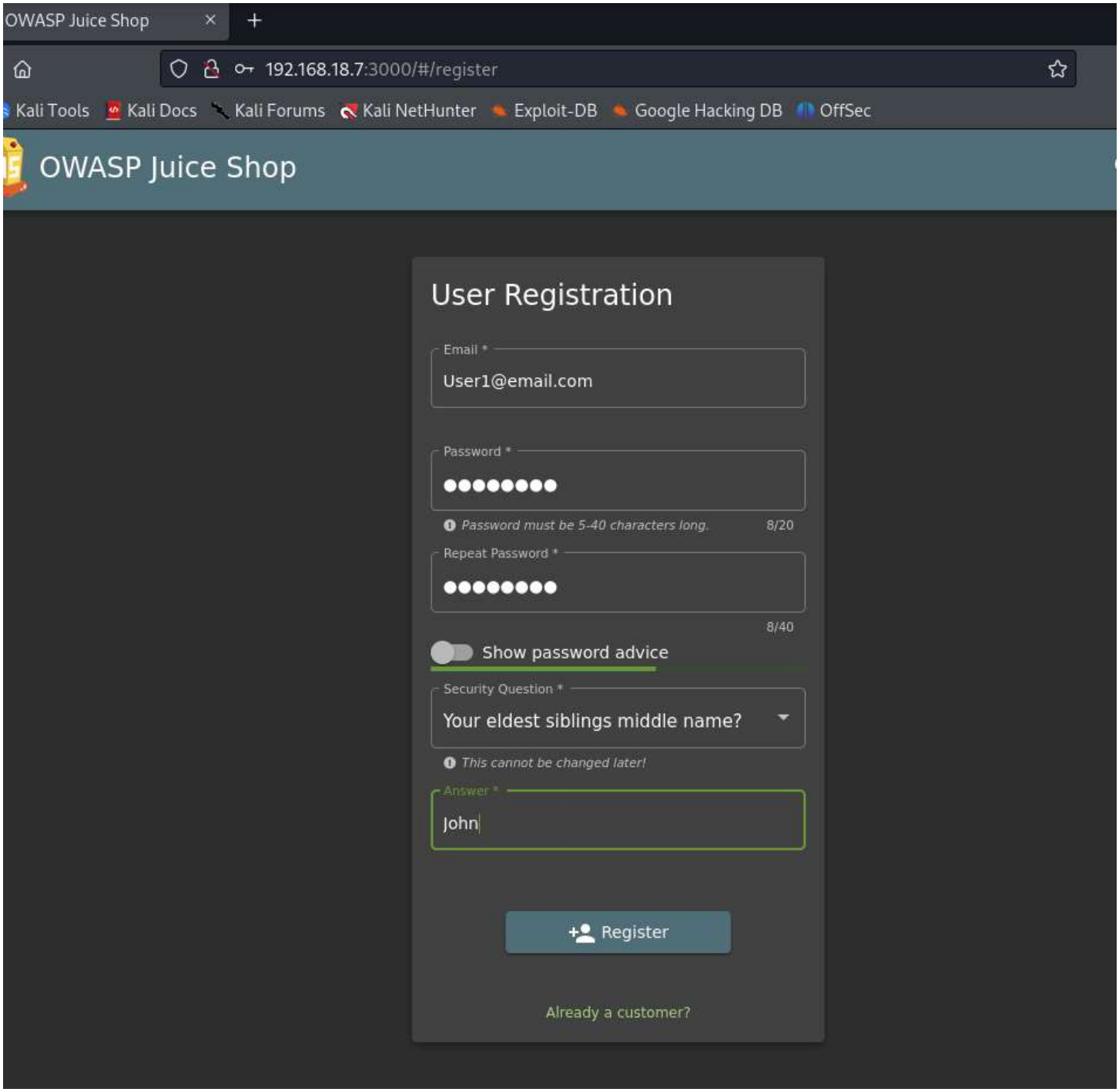
No issue:	Cookies did not contain attributes that could be used to bypass access control.

Authentication

General Authentication Features

User Registration

A user account was created with the credentials User1@email.com//Password



Finding	Severity	Description
	Info	Registration process does not verify the user's email by sending a confirmation to the email address. This could allow for automated account creation.

Finding	Severity	Description
	Minor	Password complexity requirements do not follow best practices. This allows for creation of weak passwords susceptible to brute force attacks. Minimum password

		complexity should be an 8 character password with an uppercase character, number, and a symbol.

Registering a user generates this stored answer. Repeating the registration with the same answer generates the same answer, suggesting this is a cryptographic process.

2596	http://192.168.18.7:3000	POST	/socket.io/?EIO=4&transport=polling&t...	✓	200	187	text	io/
2597	http://192.168.18.7:3000	GET	/socket.io/?EIO=4&transport=polling&t...	✓	200	202	text	io/
2598	http://192.168.18.7:3000	POST	/socket.io/?EIO=4&transport=polling&t...	✓	200	187	text	io/
2599	http://192.168.18.7:3000	GET	/socket.io/?EIO=4&transport=polling&t...	✓	200	202	text	io/
2600	http://192.168.18.7:3000	POST	/api/Users/	✓	201	694	JSON	
2601	http://192.168.18.7:3000	POST	/api/SecurityAnswers/	✓	201	623	JSON	
2602	http://192.168.18.7:3000	GET	/rest/admin/application-configuration		304	278		

Request

Pretty Raw Hex

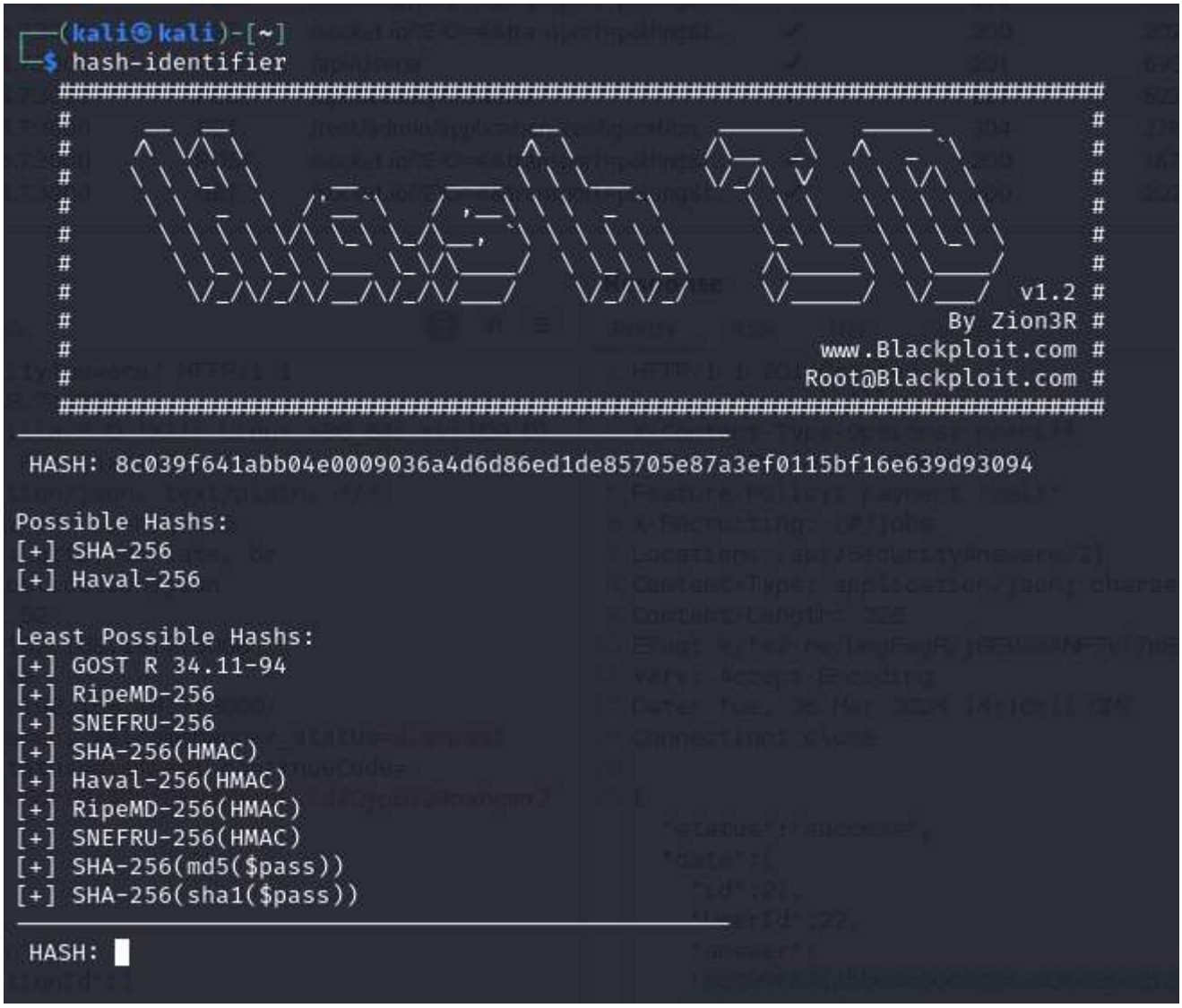
```
1 POST /api/SecurityAnswers/ HTTP/1.1
2 Host: 192.168.18.7:3000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0)
  Gecko/20100101 Firefox/115.0
4 Accept: application/json, text/plain, */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/json
8 Content-Length: 52
9 Origin: http://192.168.18.7:3000
10 Connection: close
11 Referer: http://192.168.18.7:3000/
12 Cookie: language=en; welcomebanner_status=dismiss;
   cookieconsent_status=dismiss; continueCode=
   aj4QDO4KyOqPJ7j2novp9EQ38gYVAJlGM1wWxalND5reZRLzmXk6B
   bmzZRb3
13
14 {
     "UserId":22,
     "answer":"John",
     "SecurityQuestionId":1
   }
```

Response

Pretty Raw Hex Render

```
1 HTTP/1.1 201 Created
2 Access-Control-Allow-Origin: *
3 X-Content-Type-Options: nosniff
4 X-Frame-Options: SAMEORIGIN
5 Feature-Policy: payment 'self'
6 X-Recruiting: /#/jobs
7 Location: /api/SecurityAnswers/21
8 Content-Type: application/json; charset=utf-8
9 Content-Length: 226
10 ETag: W/"e2-dXprqhxUsS1OZQQknyNYd6/vBrk"
11 Vary: Accept-Encoding
12 Date: Sat, 23 Mar 2024 13:54:30 GMT
13 Connection: close
14
15 {
     "status":"success",
     "data":{
       "id":21,
       "UserId":22,
       "answer":
       "3c039f641abb04e0009036a4d6d86ed1de85705e87a3ef01
       15bf16e639d93094",
       "SecurityQuestionId":1,
       "updatedAt":"2024-03-23T13:54:30.572Z",
       "createdAt":"2024-03-23T13:54:30.572Z"
     }
   }
```

Entering the value into hash-identifier generates the following result



Placing the hash into the hash analyzer at https://www.tunnelsup.com/hash-analyzer/ retrieves a similar result. This is likely a SHA-256 hash.

Hash Analyzer

Tool to identify hash types. Enter a hash to be identified.

8c039f641abb04e0009036a4d6d86ed1de85705e87a3ef0115bf16e639d93094

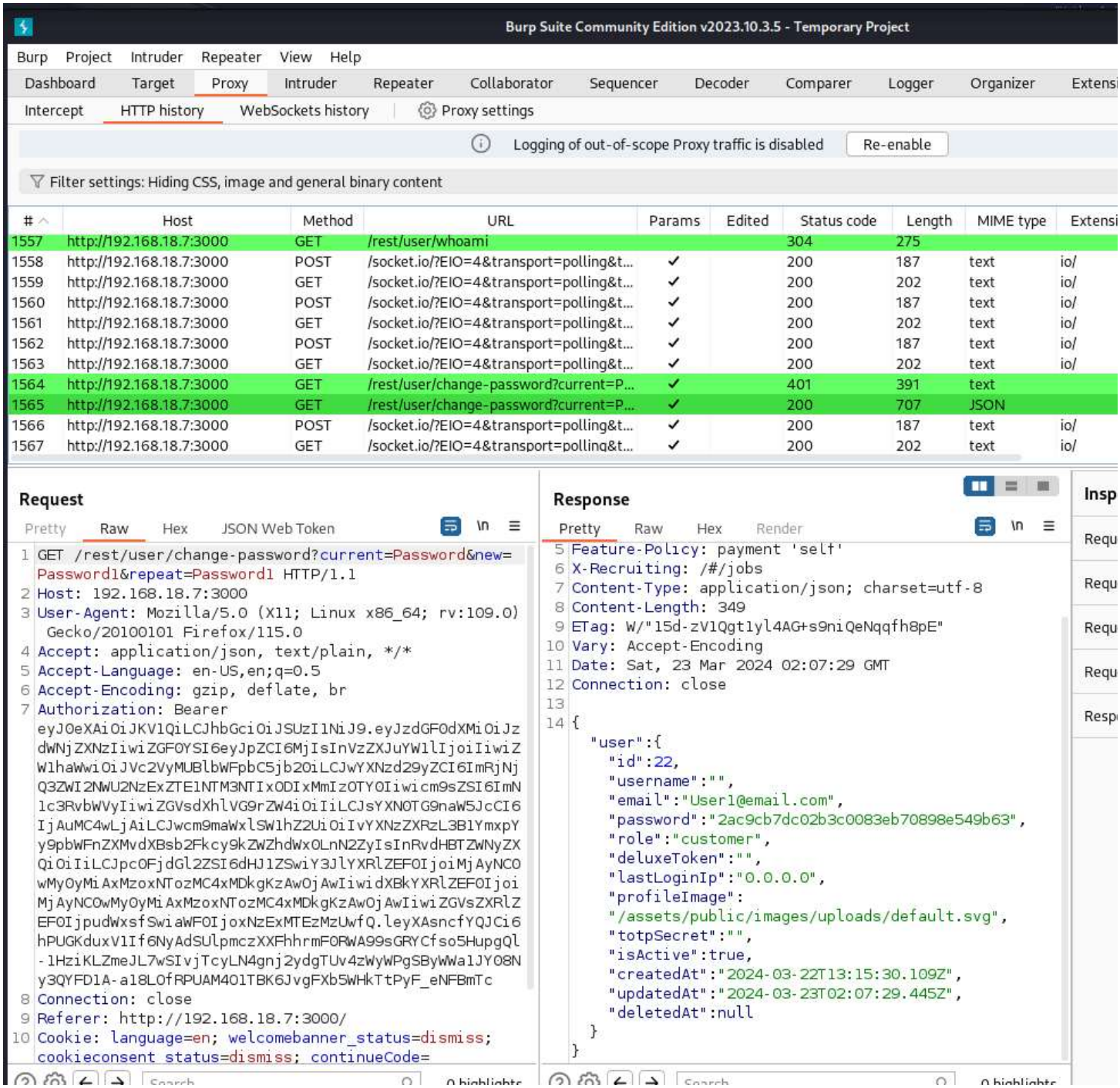
Analyze

Hash:	8c039f641abb04e0009036a4d6d86ed1de85705e87a3ef0115bf16e639d93094
Salt:	Not Found
Hash type:	SHA2-256
Bit length:	256
Character length:	64
Character type:	hexidecimal

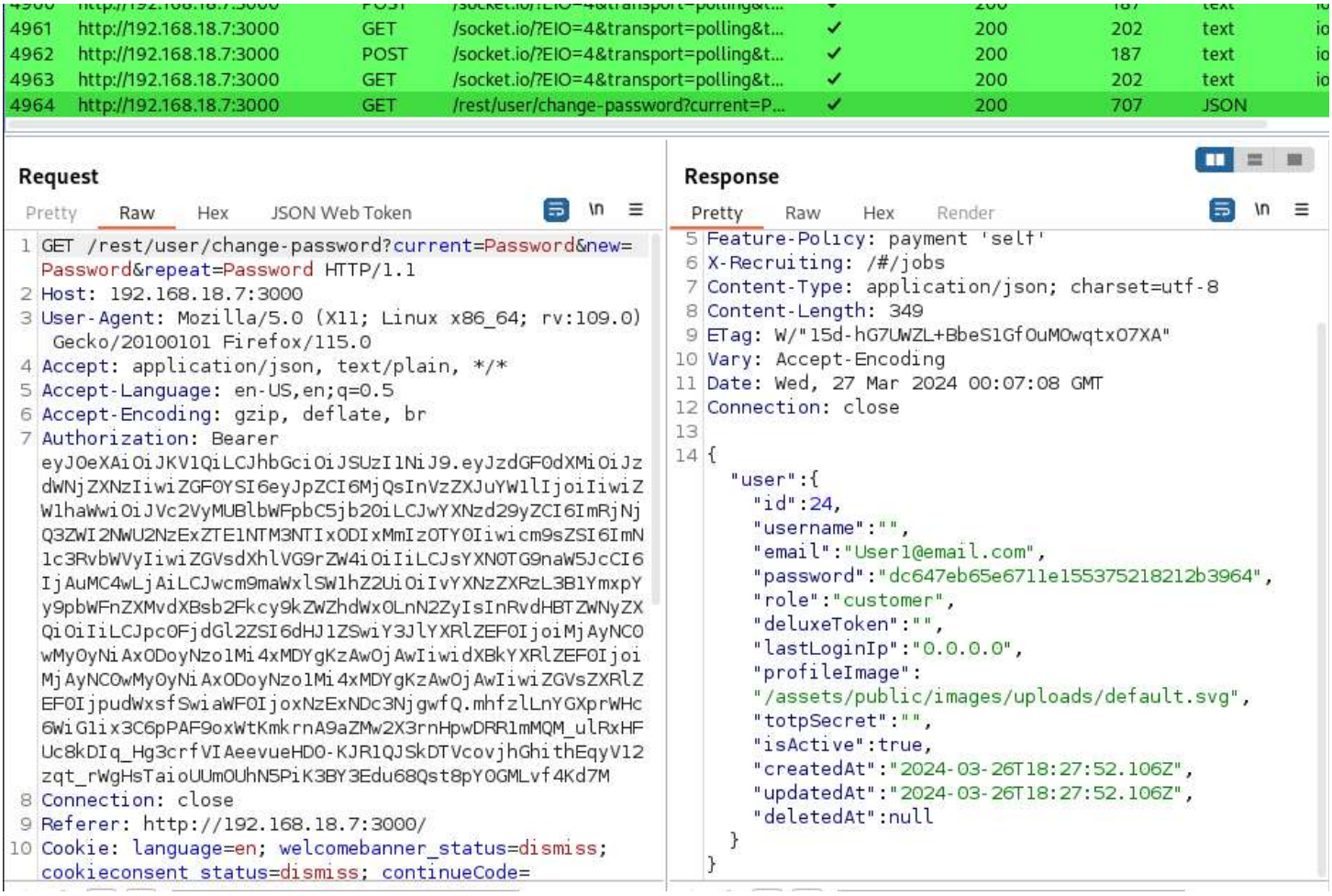
No issue:	SHA-256 is a secure hashing algorithm

Password Change

Changing the password passes the password as a URL parameter

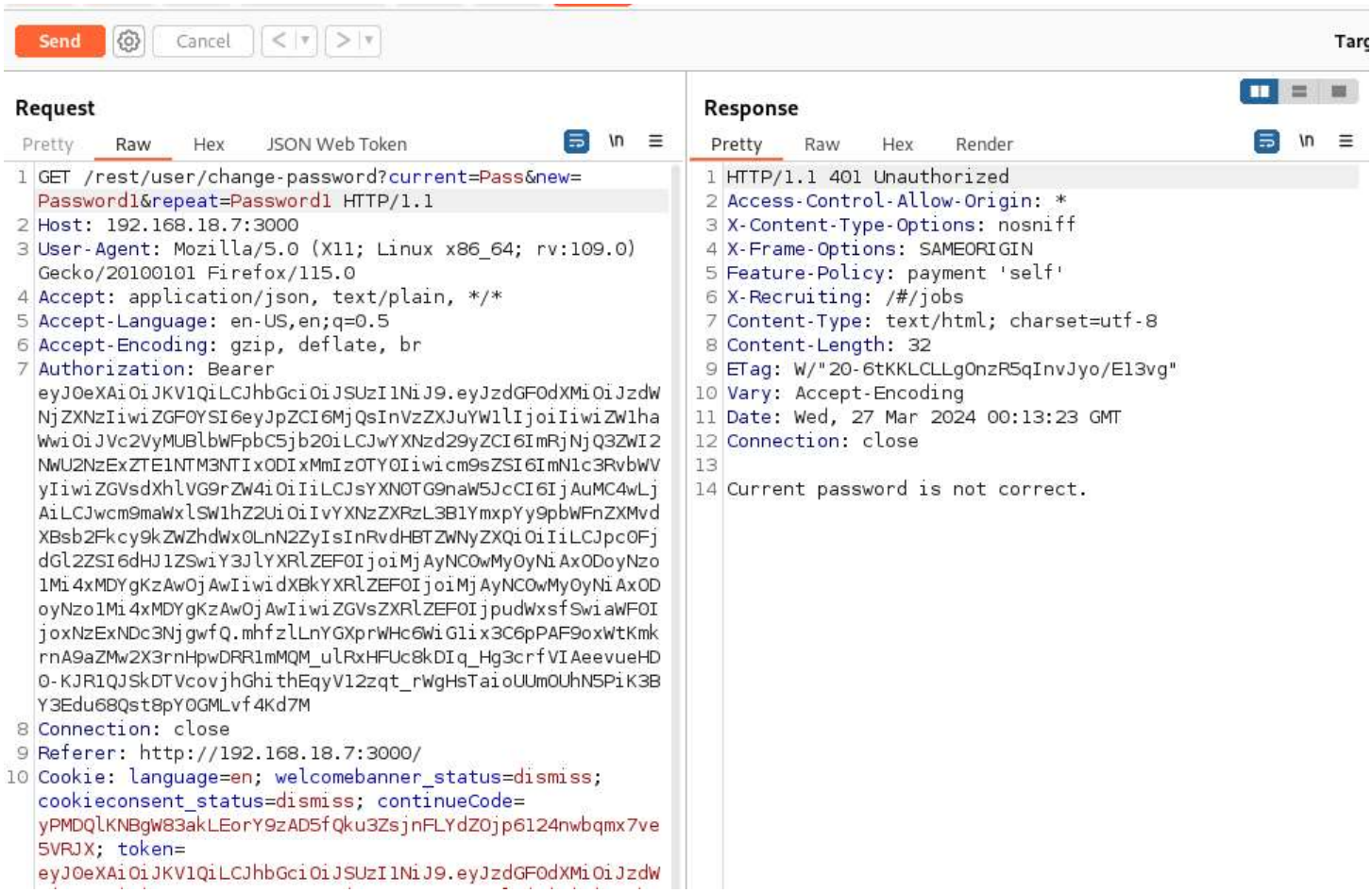


Finding	Severity	Description
	Major	Passing credentials in cleartext as parameters in a GET request allows anyone watching the web traffic to capture the credentials. This should be changed to a POST request.

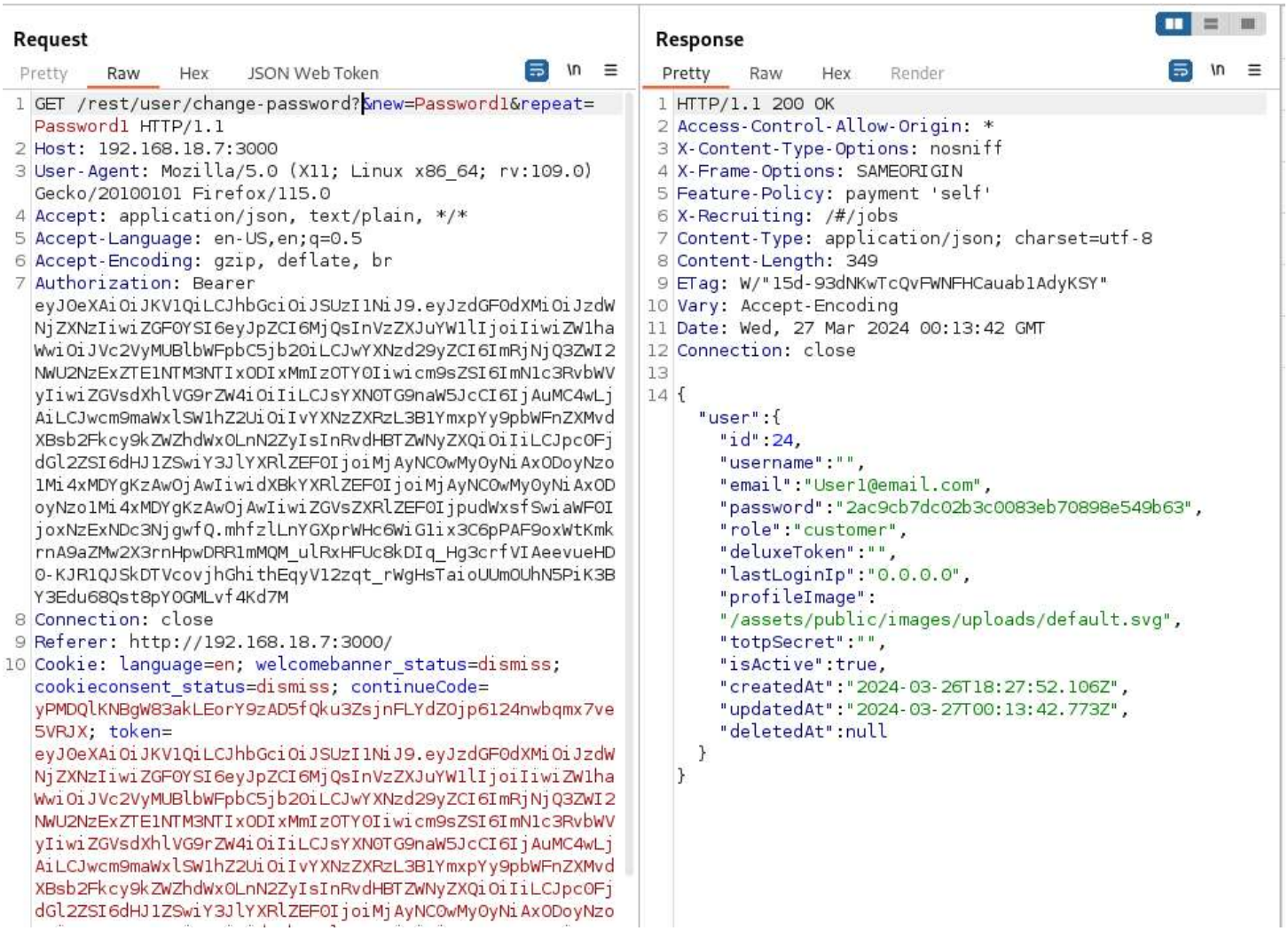


Finding	Severity	Description
	Info	The change password feature allows users to re-enter the current password.

An attempt was made to change the password without knowing the original password and bypass the password authentication. The initial attempt was unsuccessful.



However, a second attempt was successful by removing the “current” parameter.



We were subsequently able to login with the password changed using this method.

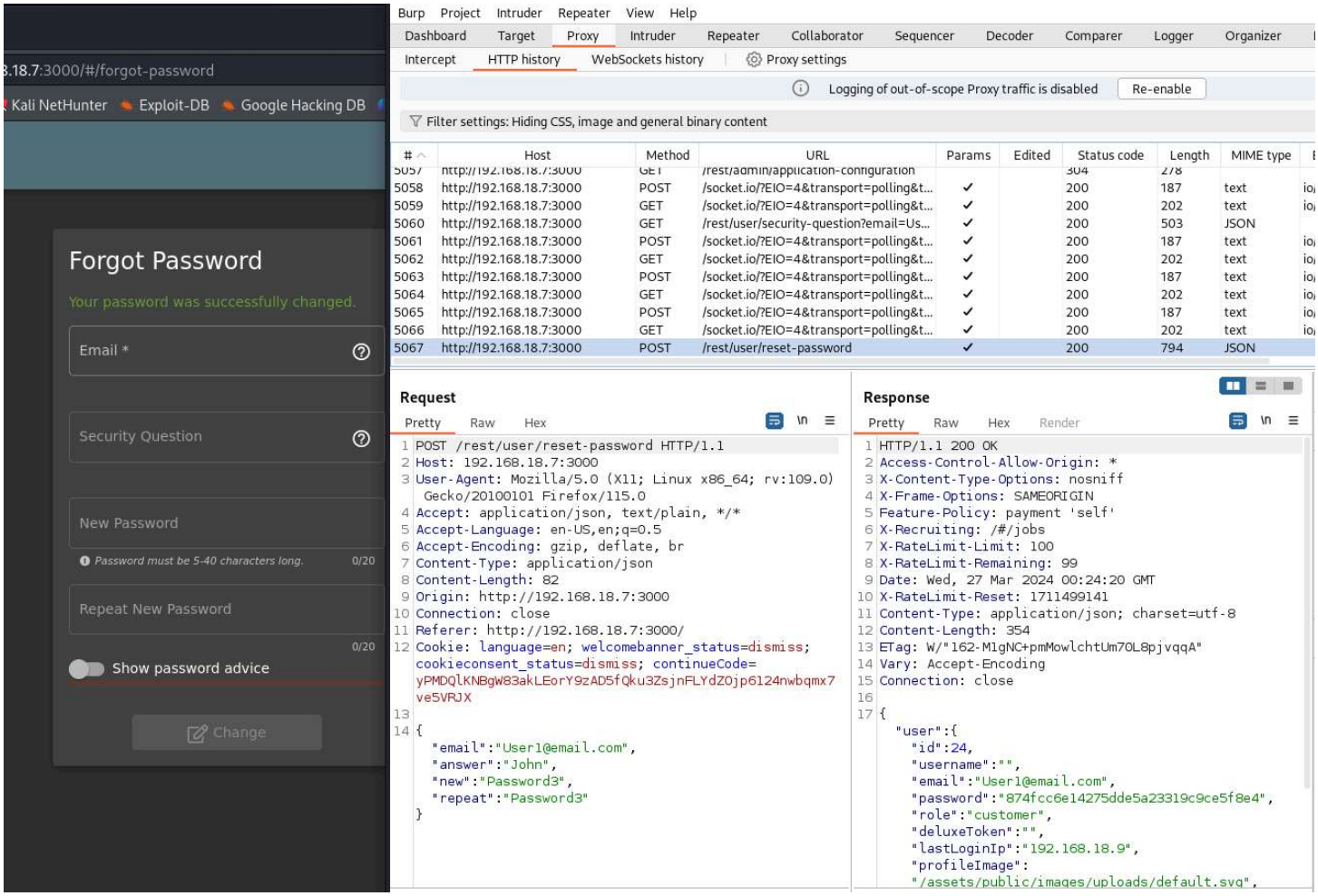
Finding	Severity	Description
	Major	The password verification feature can be bypassed, allowing attackers to change passwords without knowing the original password. This strengthens session hijacking attacks. Server-side validation needs to be corrected.

The Change Password feature is performed without the use of a CSRF token.

Finding	Severity	Description
	Minor	CSRF token is missing in the Change Password feature. CSRF tokens are effective in preventing CSRF attacks.

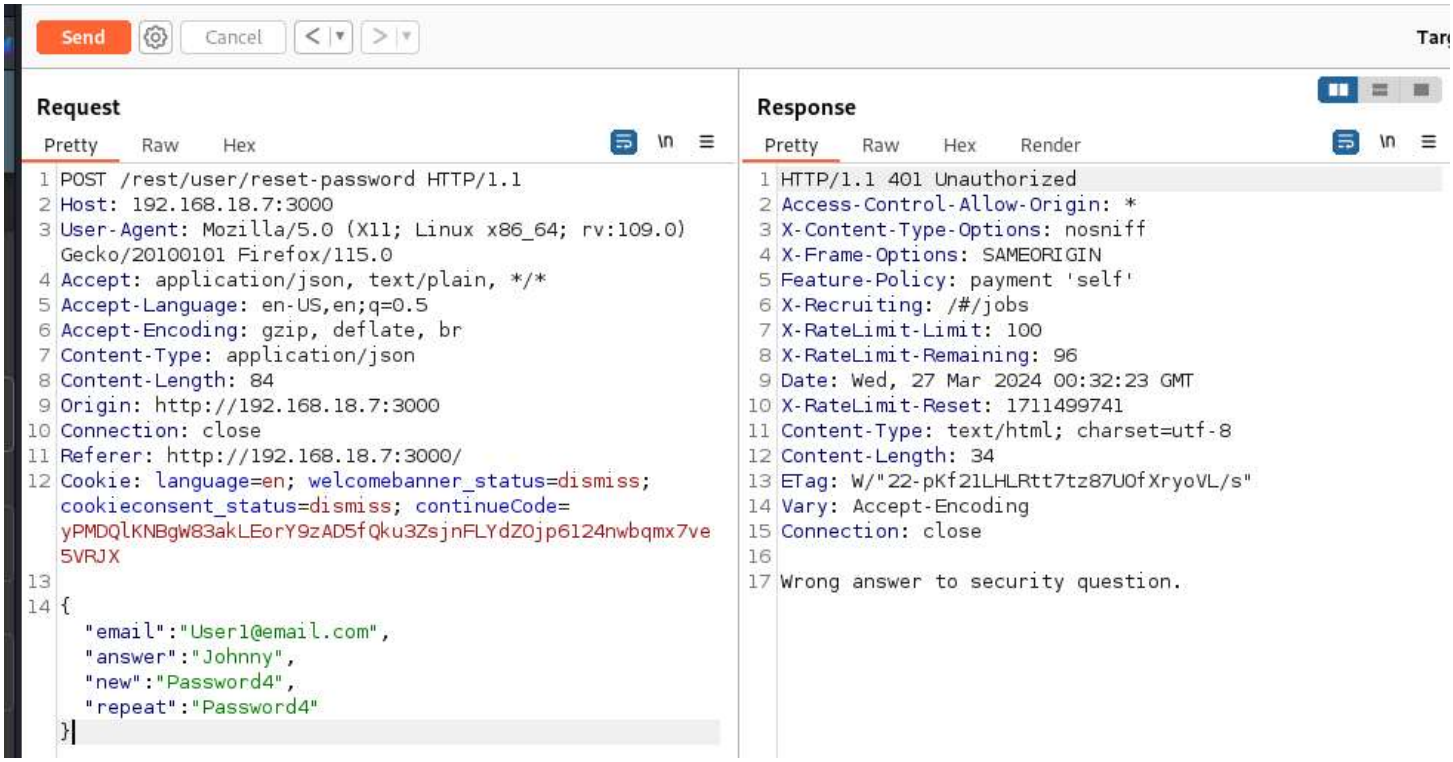
Forgot Password

The Forgot Password feature was tested.



Finding	Severity	Description
	Minor	The Forgot Password feature does not follow best practices. The use of a security question is not recommended as that is susceptible to attacker reconnaissance and OSINT. Best practices would be to send a temporary link to the user email address.

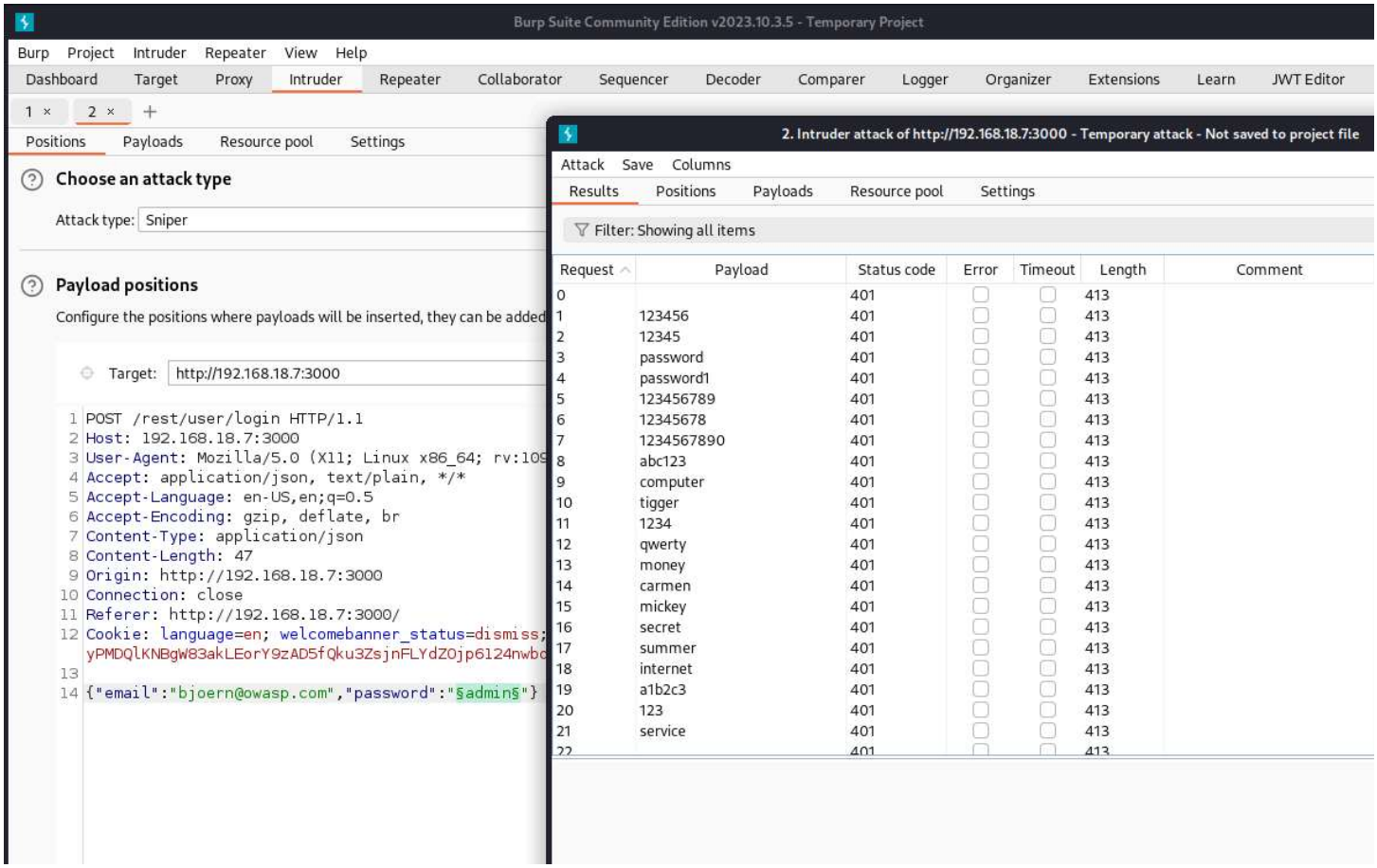
The input validation of the Forgot Password feature was tested by 1. Using mismatching passwords 2. Using an incorrect security question answer 3. Removing the security answer. The input validation showed no issues in any of these tests.



No issue:	The input validation of the Forgot Password feature is secure.

Brute Force Attacks on Login

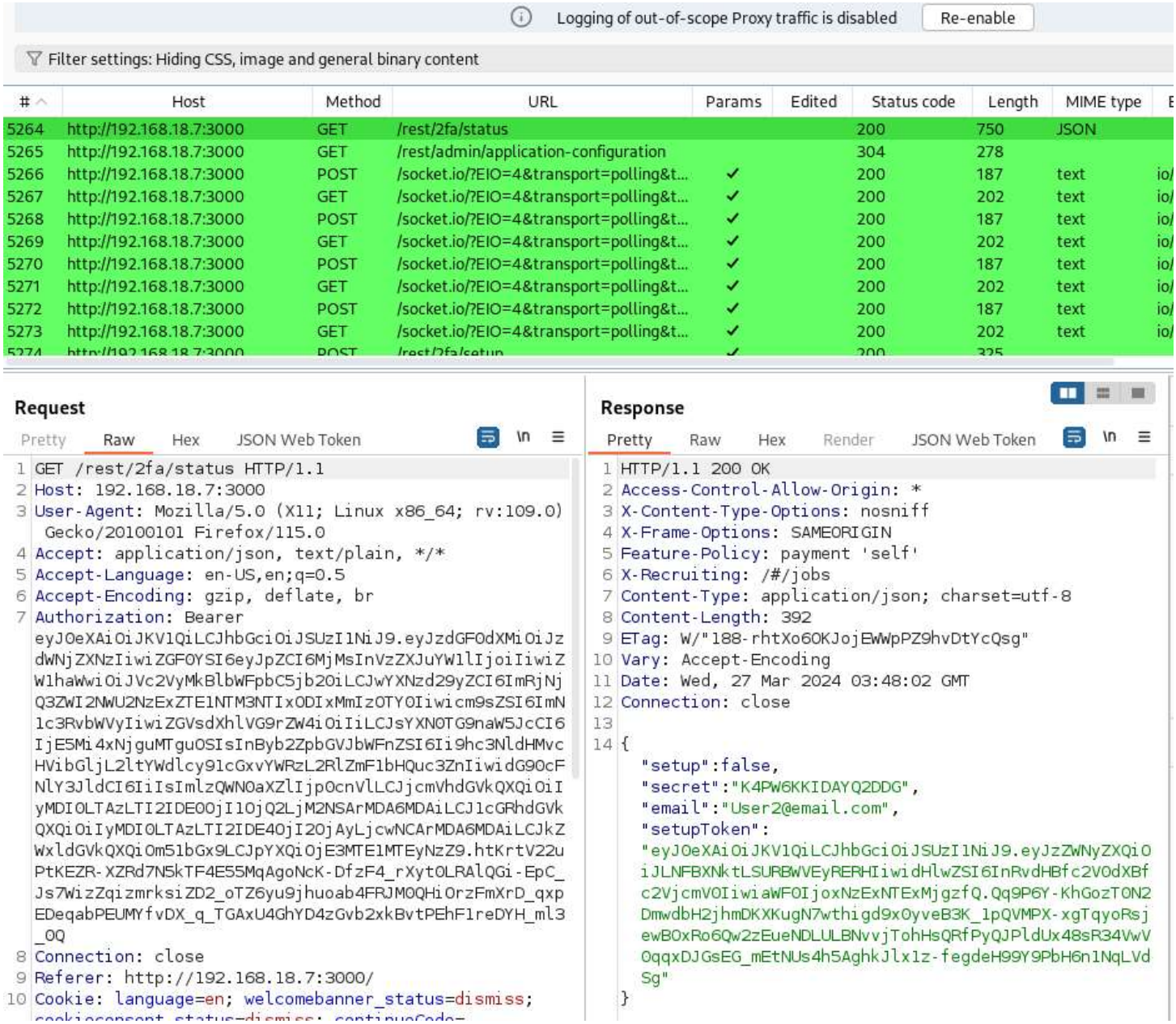
An automated brute force attack was performed against the login process.



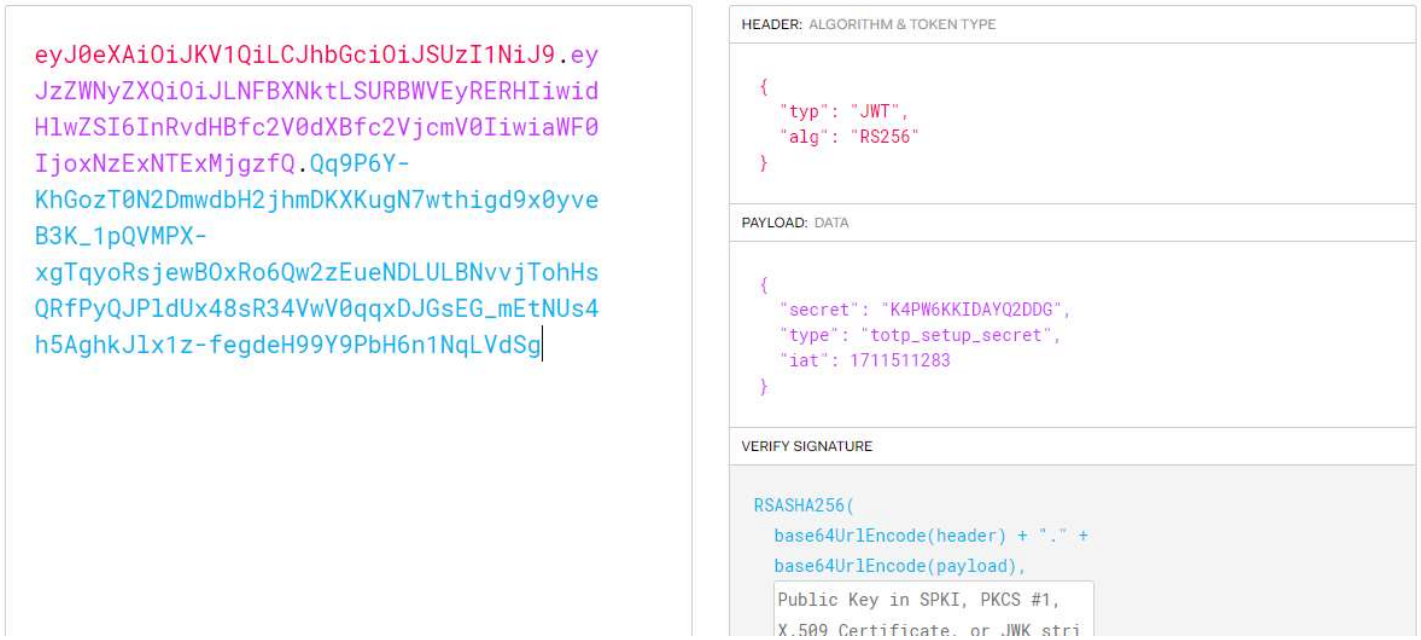
Finding	Severity	Description
	Minor	Brute force attacks allow attackers to potentially discover credentials. An account lockout after 3 failed logins should be implemented to prevent automated attacks.

Two Factor Authentication

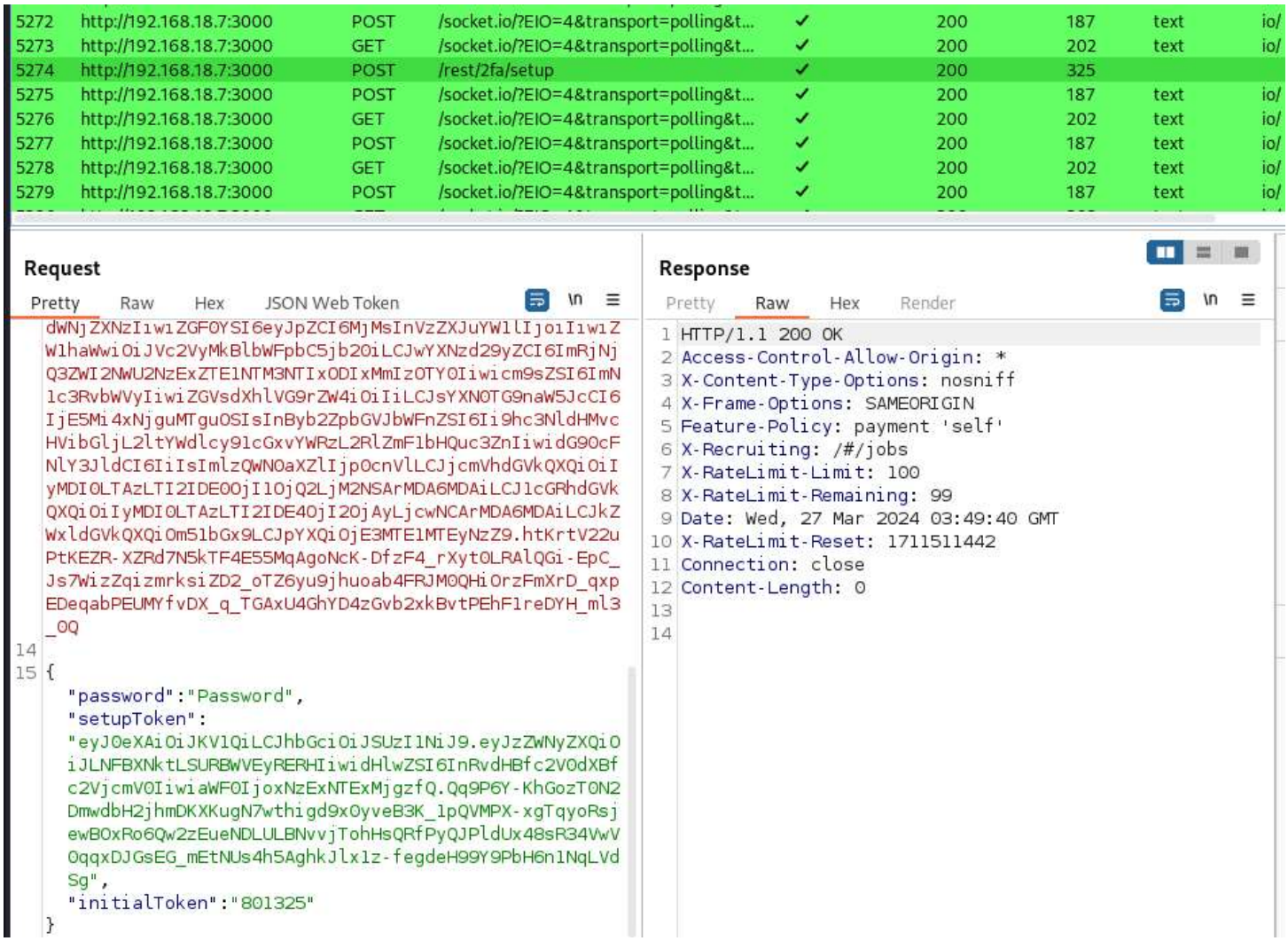
The 2FA Authentication was set up on the User@email.com account using the Microsoft Authenticator app.



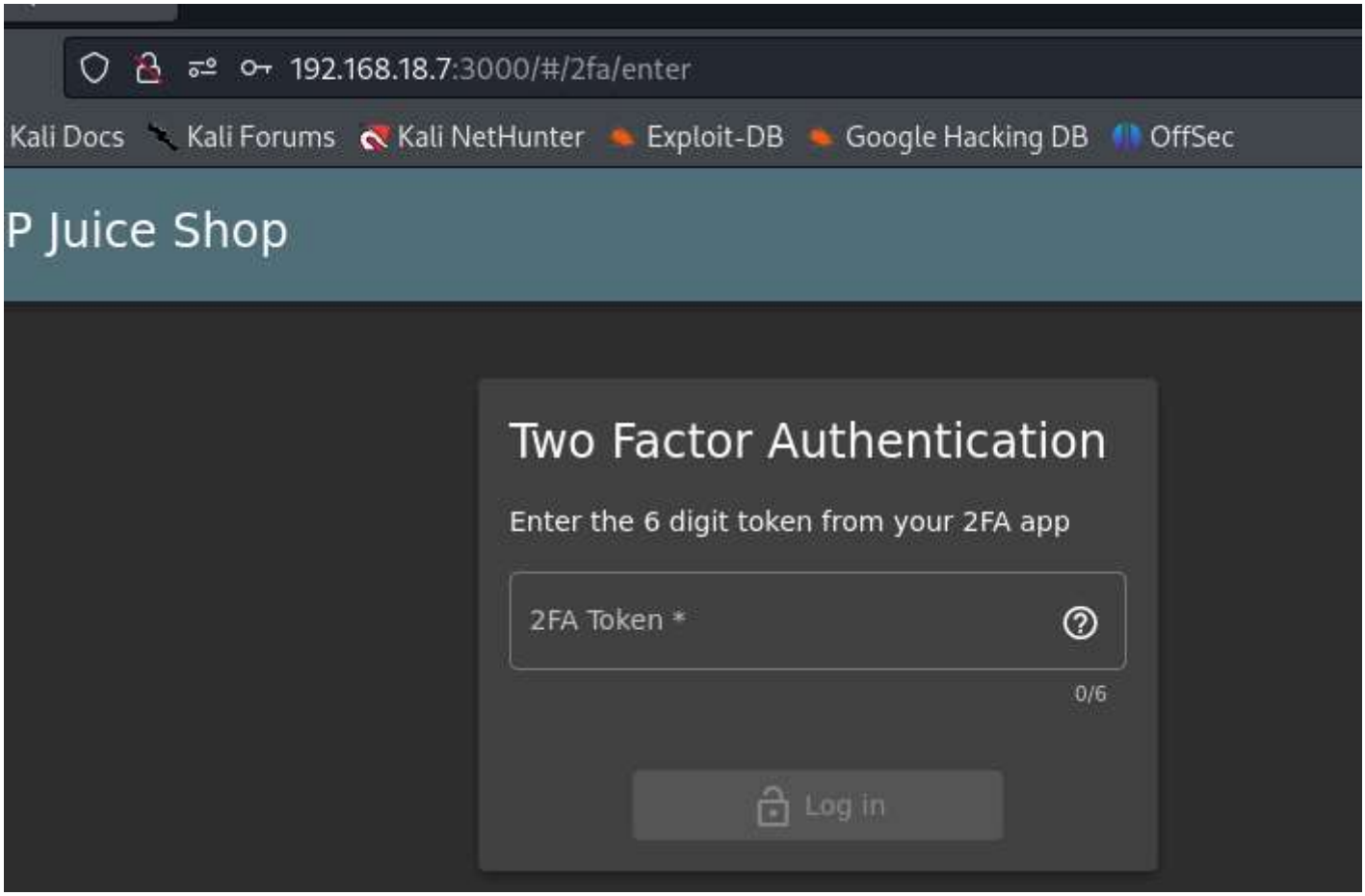
The JWT provided contained the following information.



Setting up the 2FA involved sending the same JWT back along with the initial token value.



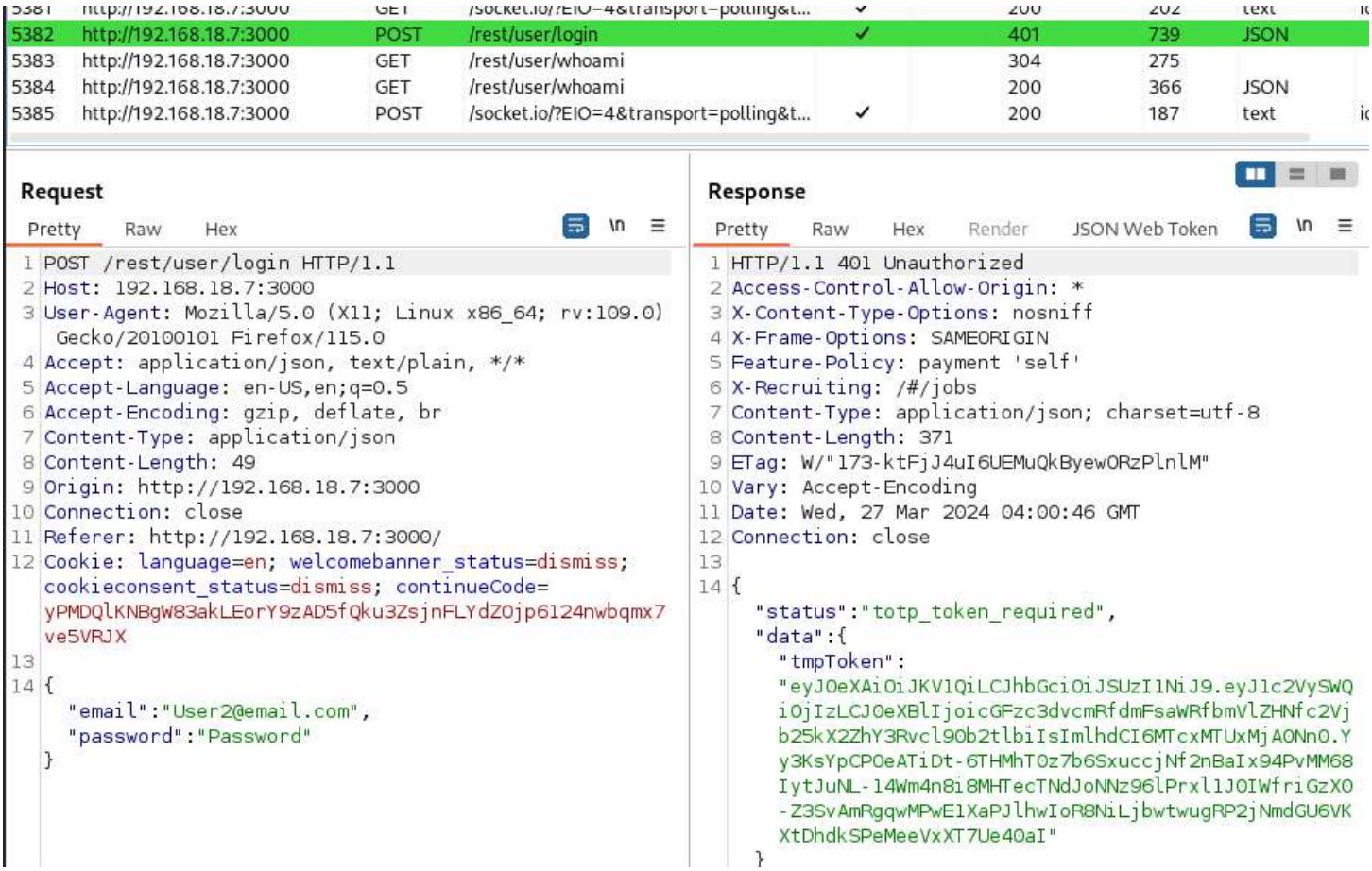
A 2FA Token prompt now shows up in login.



An unsuccessful attempt was made to bypass the 2FA by visiting an authenticated page #/deluxe-membership without entering the 2FA Token.

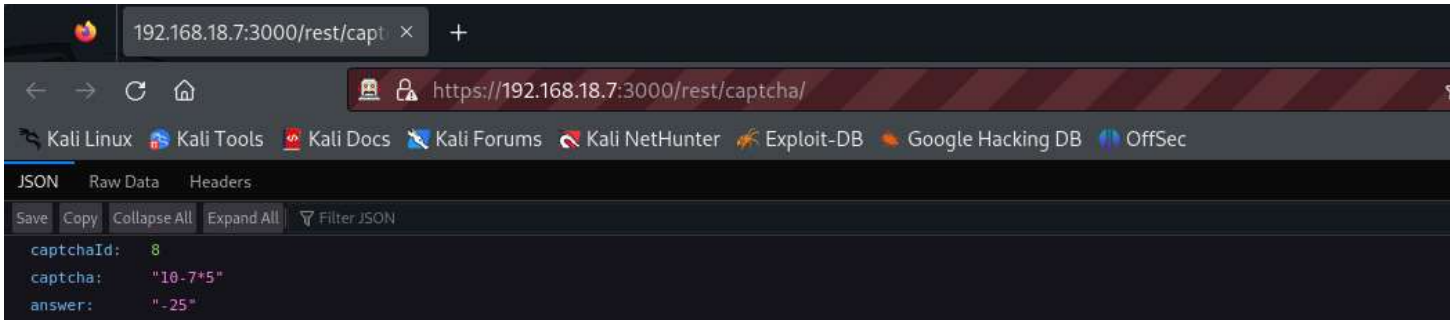
No issue:	The 2FA could not be bypassed by skipping it.

Reviewing the returned temporary token did not show any issues.



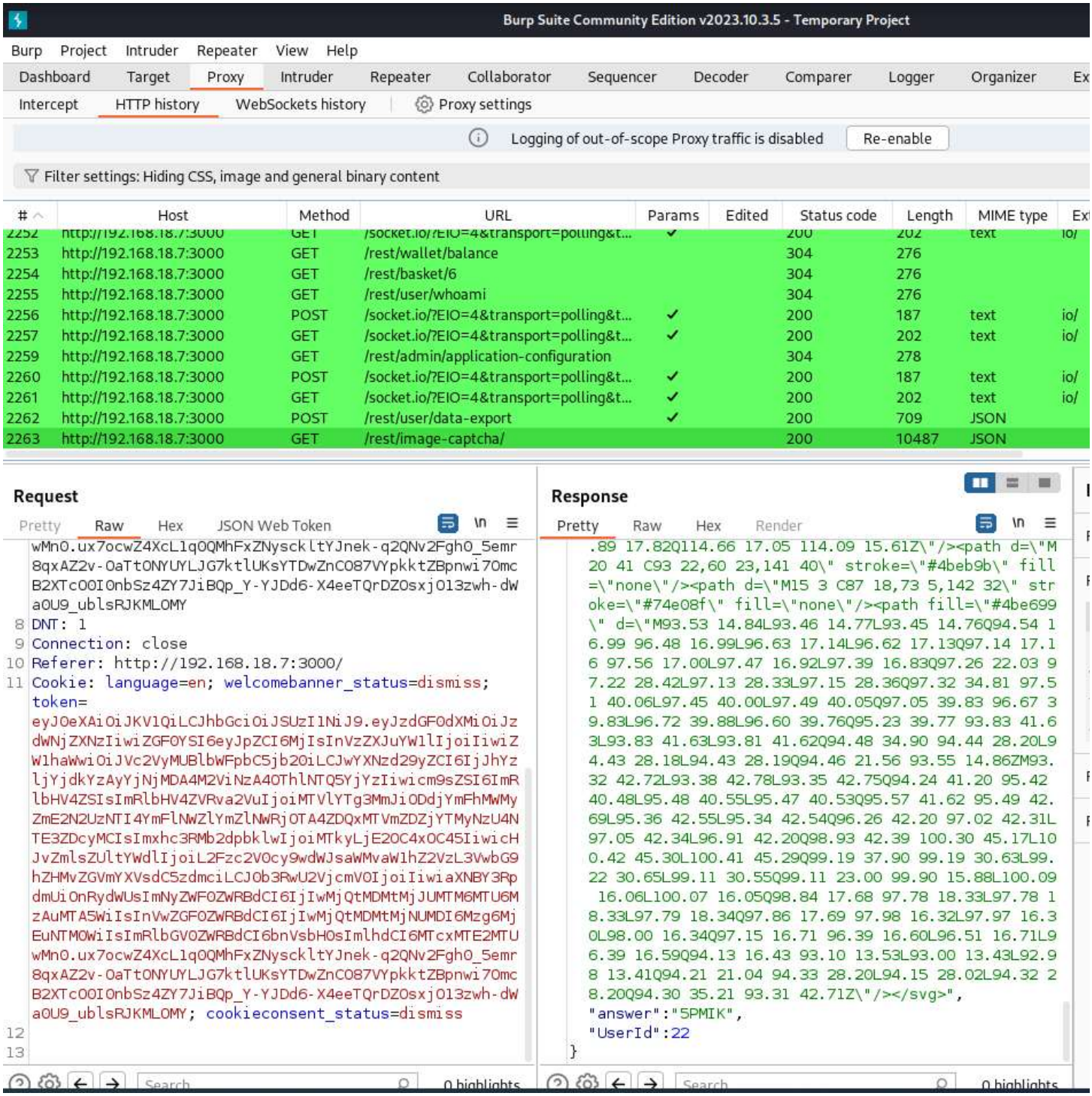
Authentication in Other Areas of the Web Application

The Captcha of the Customer Feedback page could be bypassed by viewing the answer of the Captcha.



Finding	Severity	Description
	Major	CAPTCHA is not implemented effectively in the Customer Feedback page.The disclosure of the CAPTCHA results could allow attackers to perform automated attacks.

In the data export page, the CAPTCHA can be found by reading the HTTP response

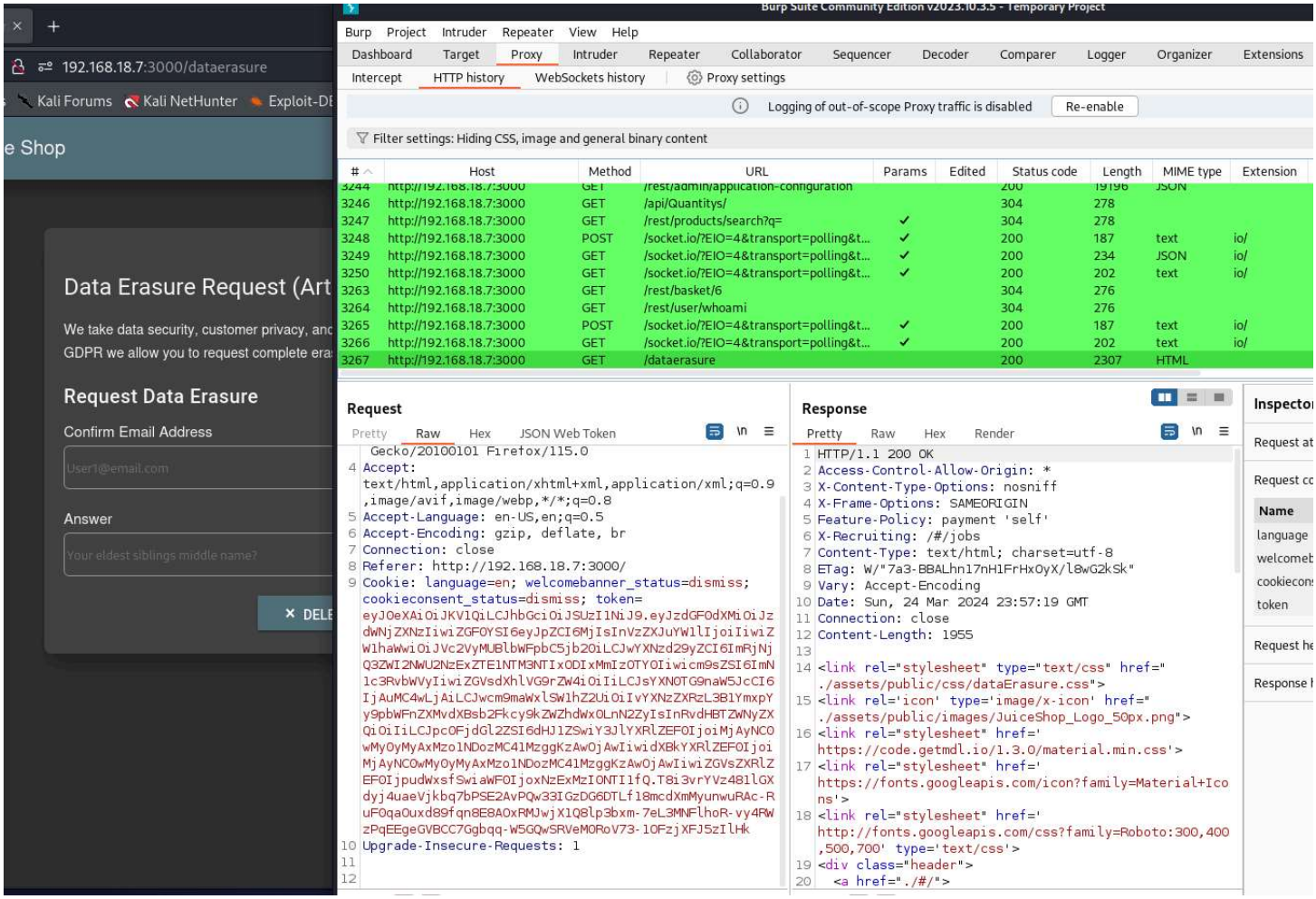


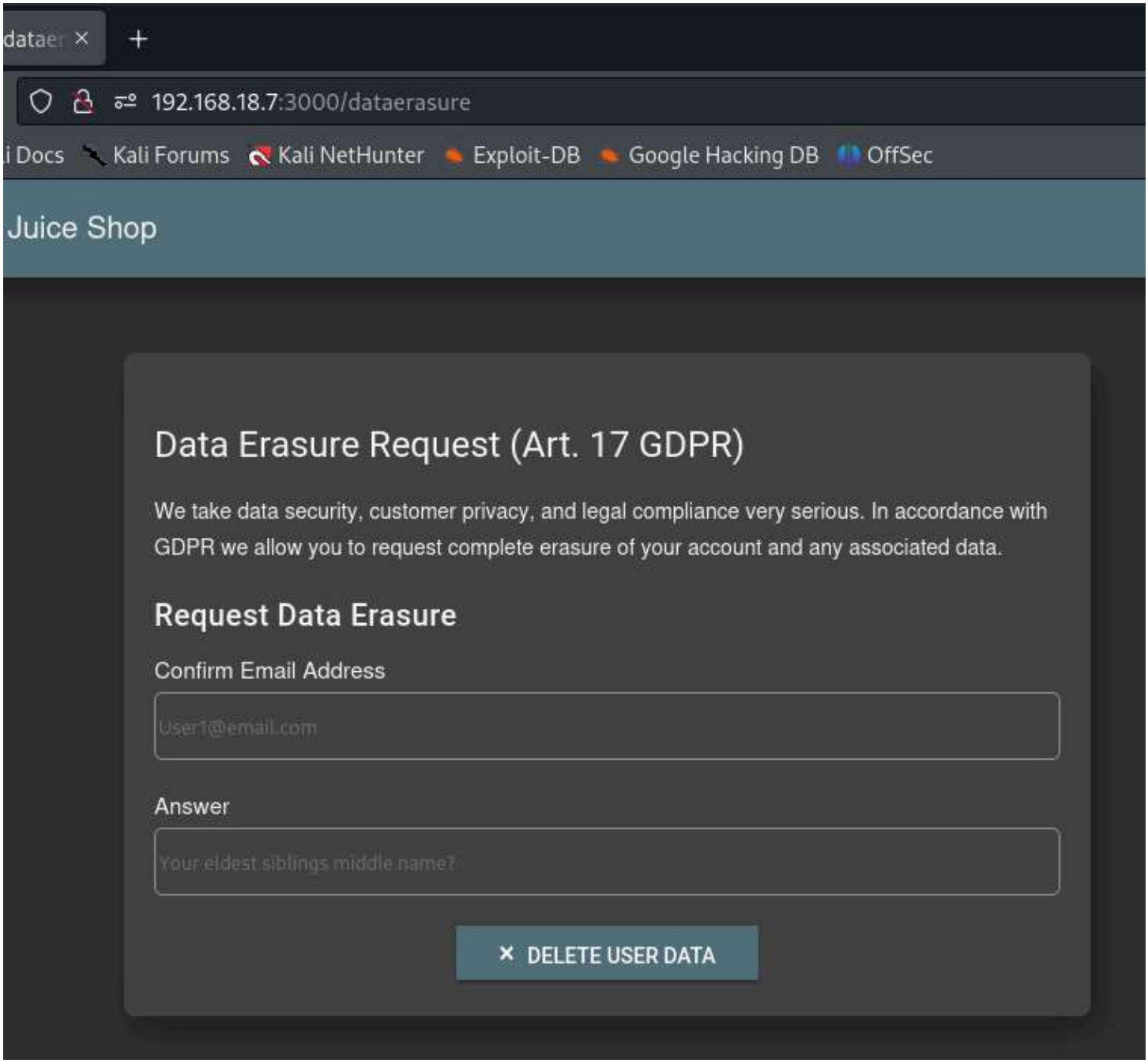
Finding	Severity	Description
	Major	CAPTCHA is not implemented effectively in the Data Disclosure page. The disclosure of the CAPTCHA results could allow attackers to bypass authentication.

The request for data export is performed without the use of a CSRF token.

Finding	Severity	Description
	Minor	The request for data export feature should use CSRF tokens. CSRF tokens are effective in preventing CSRF attacks.

Visiting the /dataerasure directory generates the Upgrade-Inseure-Requests header

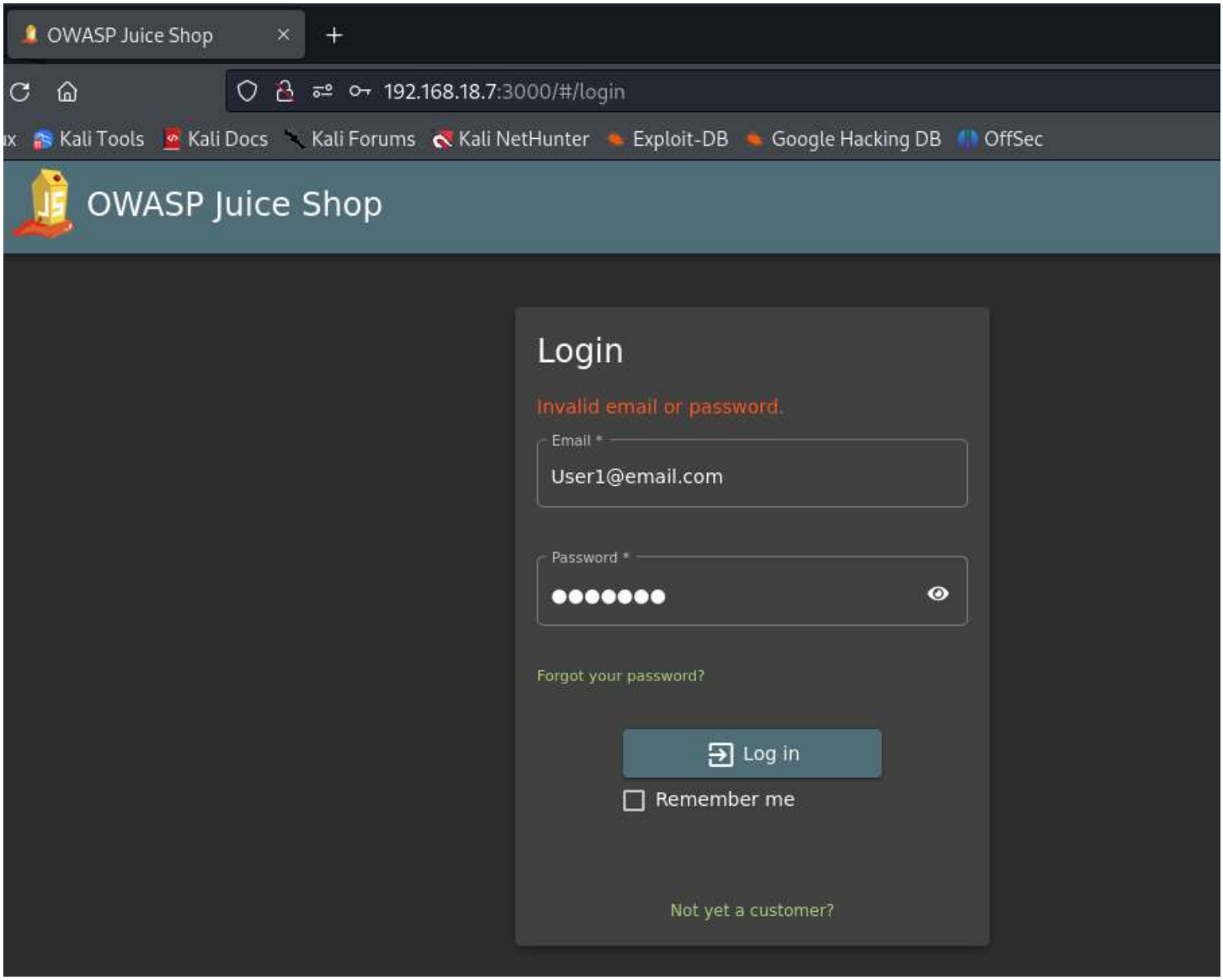




Finding	Severity	Description
	Info	The Data Erasure page asks the user to confirm their email address yet the email address is displayed on the page.

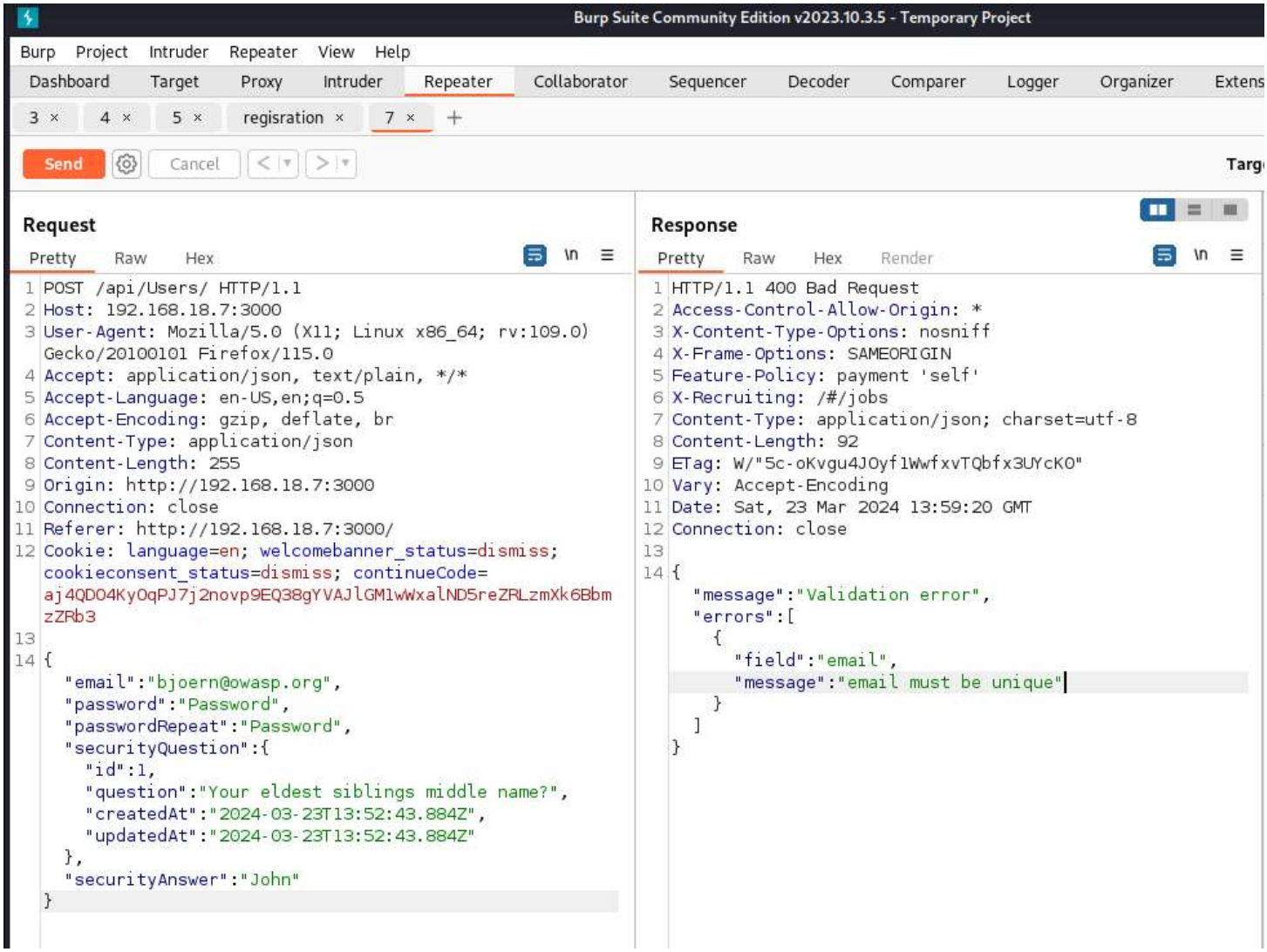
Error Messages During Authentication

The authentication process was inspected to see if too much information is revealed in a failed attempt.



No issue:	Failed authentication does not reveal too much information to an attacker.

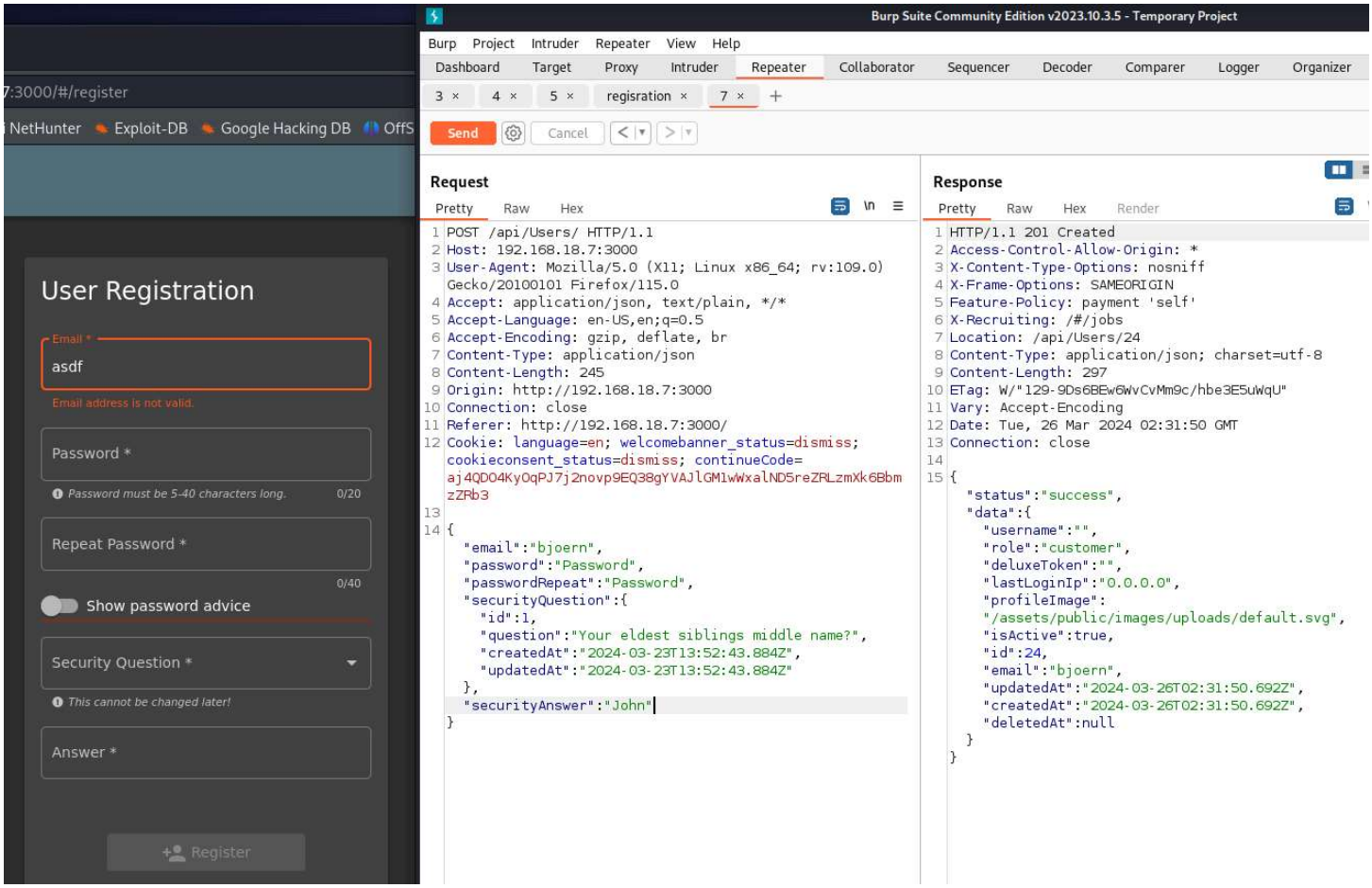
Attempting to register with an existing email generates an output message that the email must be unique.



Finding	Severity	Description
	Minor	The error message of “email must be unique” discloses too much information to the attacker and allows for user enumeration.

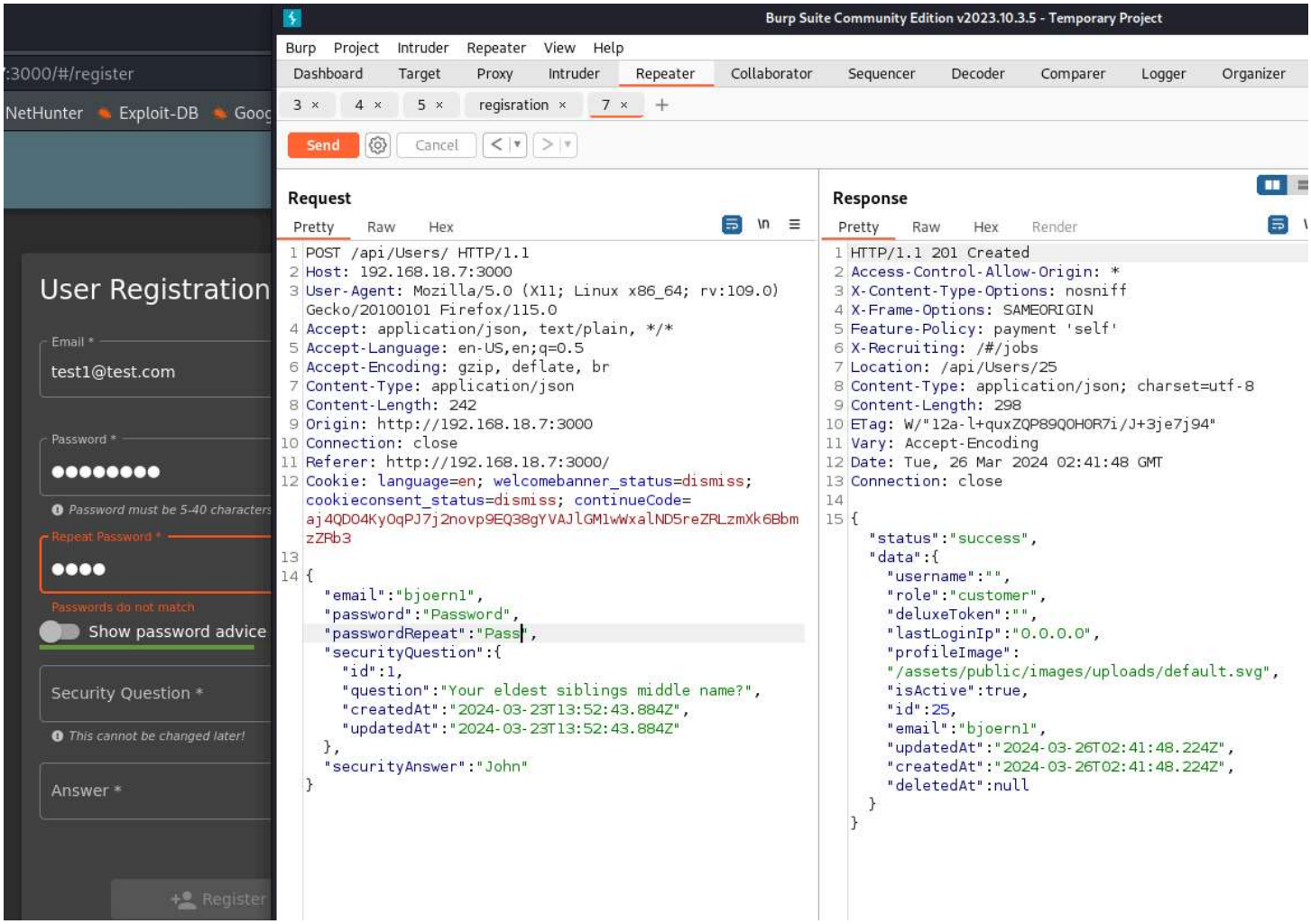
Input Validation During Authentication

When registering a new user, the email field requires the @ symbol and a character afterwards. This input validation could be bypassed by manipulating the client-side HTTP request.

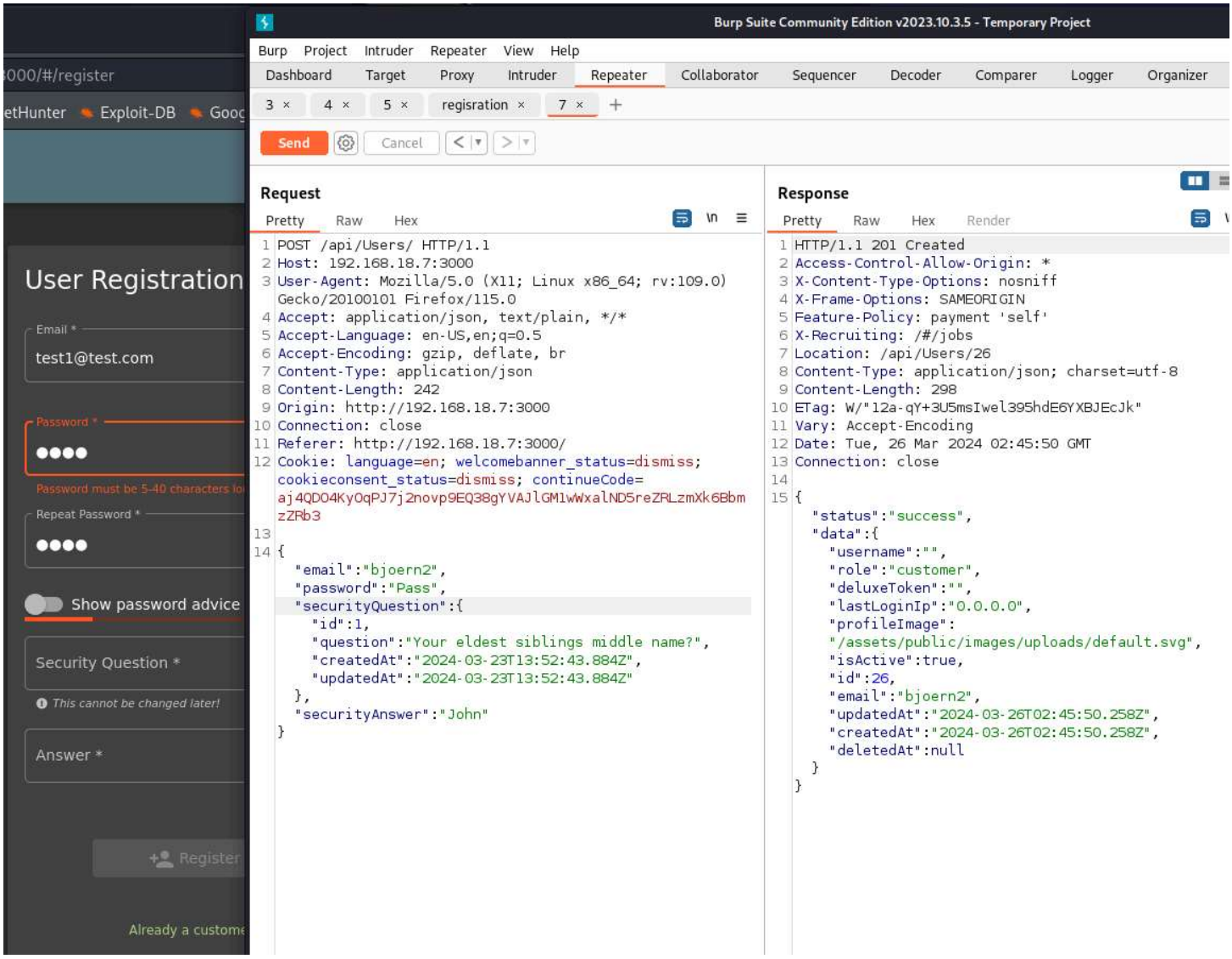


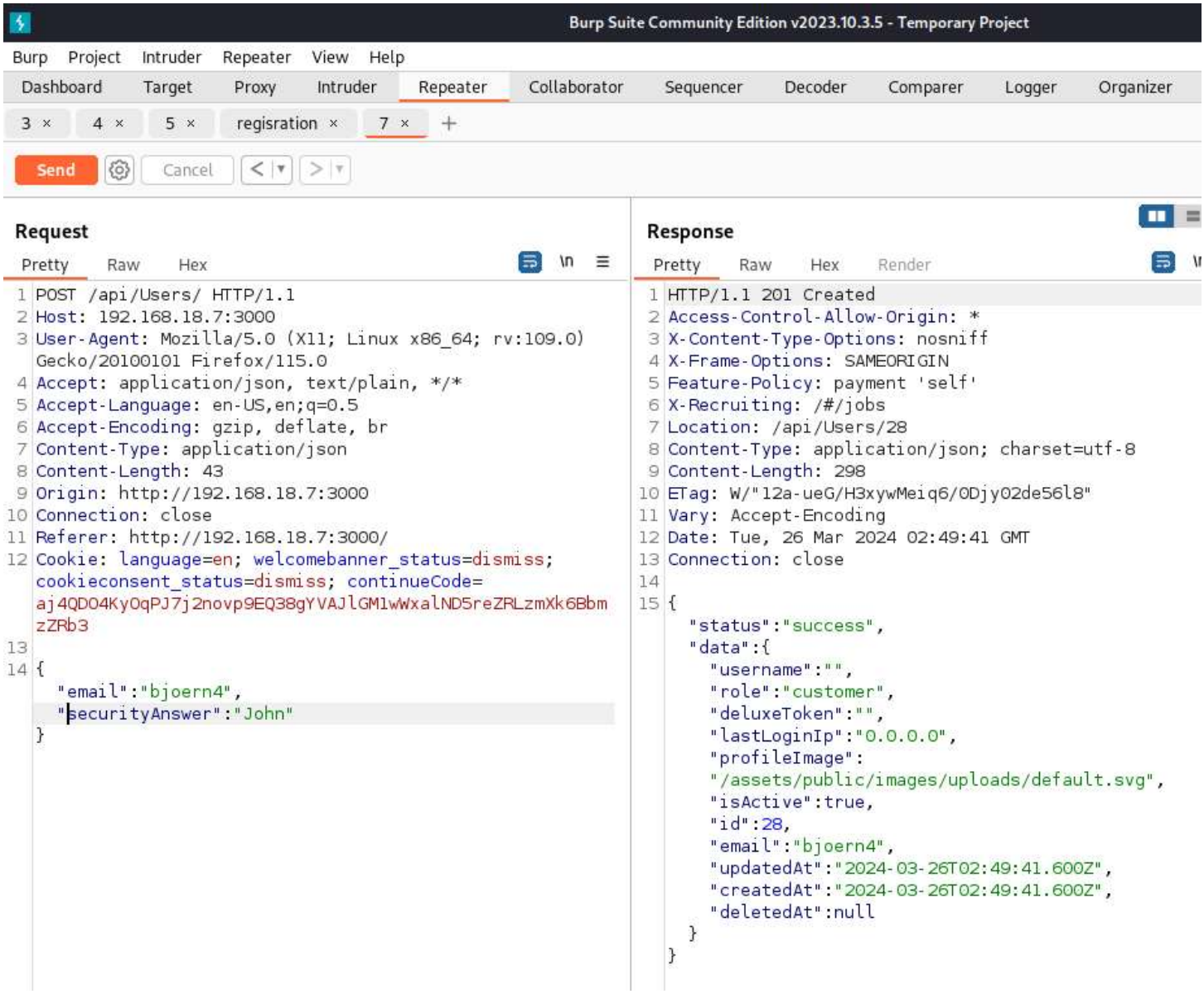
Test this for SQL injection

When registering a new user, the password fields are supposed to match, as indicated in the browser. This input validation could be bypassed by manipulating the client-side HTTP request.

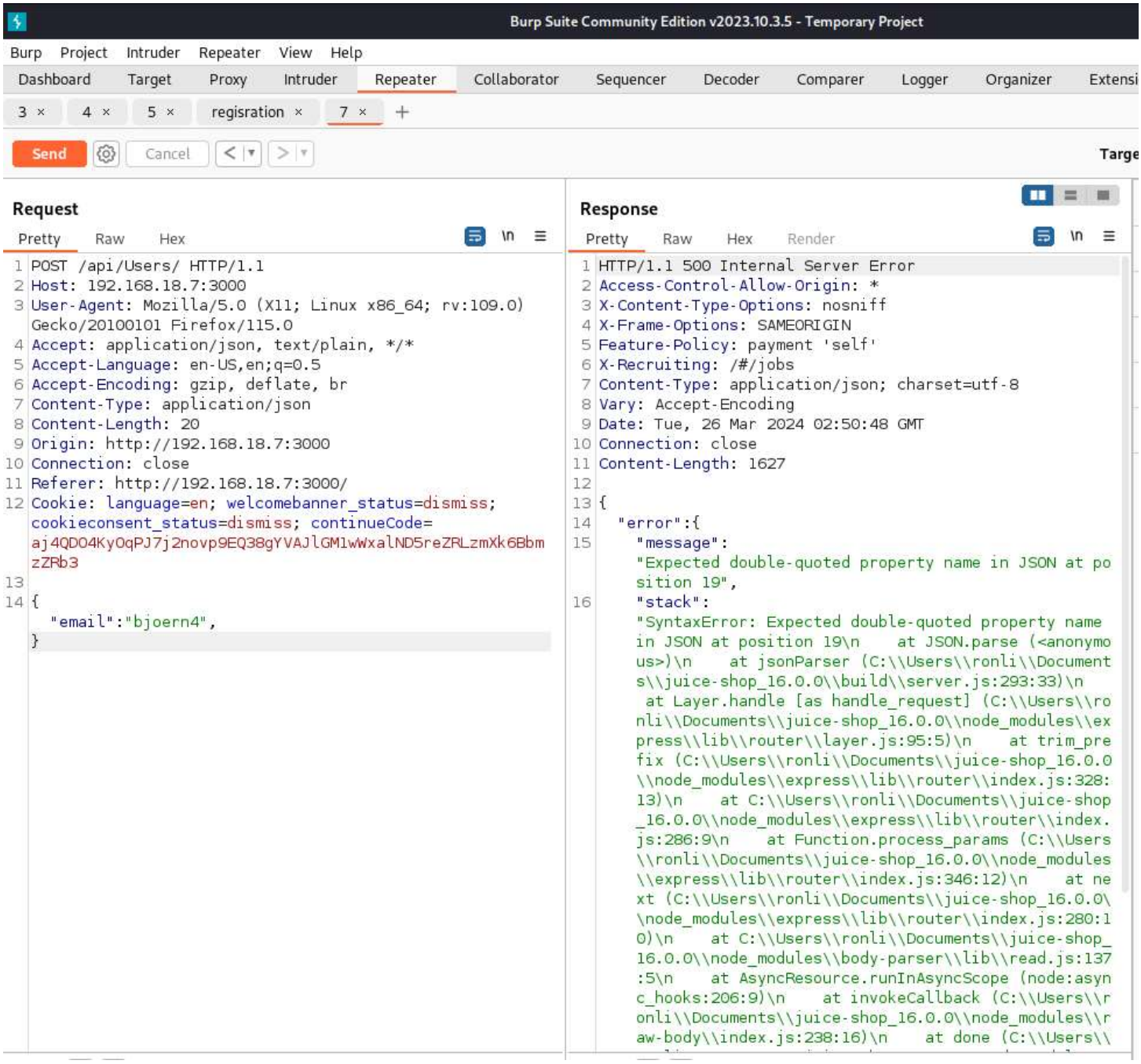


Similarly, the password length could be bypassed. The Password field, the Repeat Password field, and the Security Question field could be removed entirely and the registration will still process.


000/#/register
etHunter
Exploit-DB
Goog
User Registration
Email *
test1@test.com
Password *
Password must be 5-40 characters lo
Repeat Password *
Show password advice
Security Question *
This cannot be changed later!
Answer *
Register
Already a custom
Burp Suite Community Edition v2023.10.3.5 - Temporary Project
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer
3 × 4 × 5 × regisration × 7 × +
Send Cancel
Request
Pretty Raw Hex
1 POST /api/Users/ HTTP/1.1
2 Host: 192.168.18.7:3000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: application/json, text/plain, */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/json
8 Content-Length: 242
9 Origin: http://192.168.18.7:3000
10 Connection: close
11 Referer: http://192.168.18.7:3000/
12 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode= aj4QDO4KyOqPJ7j2novp9EQ38gYVAJlGM1wWxalND5reZRLzmXk6BbmzZRb3
13
14 {
"email":"bjoern2",
"password":"Pass",
"securityQuestion":{
"id":1,
"question":"Your eldest siblings middle name?",
"createdAt":"2024-03-23T13:52:43.884Z",
"updatedAt":"2024-03-23T13:52:43.884Z"
},
"securityAnswer":"John"
}
Response
Pretty Raw Hex Render
1 HTTP/1.1 201 Created
2 Access-Control-Allow-Origin: *
3 X-Content-Type-Options: nosniff
4 X-Frame-Options: SAMEORIGIN
5 Feature-Policy: payment 'self'
6 X-Recruiting: /#/jobs
7 Location: /api/Users/26
8 Content-Type: application/json; charset=utf-8
9 Content-Length: 298
10 ETag: W/"12a-qY+3U5msIwel395hdE6YXBJEcJk"
11 Vary: Accept-Encoding
12 Date: Tue, 26 Mar 2024 02:45:50 GMT
13 Connection: close
14
15 {
"status":"success",
"data":{
"username":"",
"role":"customer",
"deluxeToken":"",
"lastLoginIp":"0.0.0.0",
"profileImage":
"/assets/public/images/uploads/default.svg",
"isActive":true,
"id":26,
"email":"bjoern2",
"updatedAt":"2024-03-26T02:45:50.258Z",
"createdAt":"2024-03-26T02:45:50.258Z",
"deletedAt":null
}
}



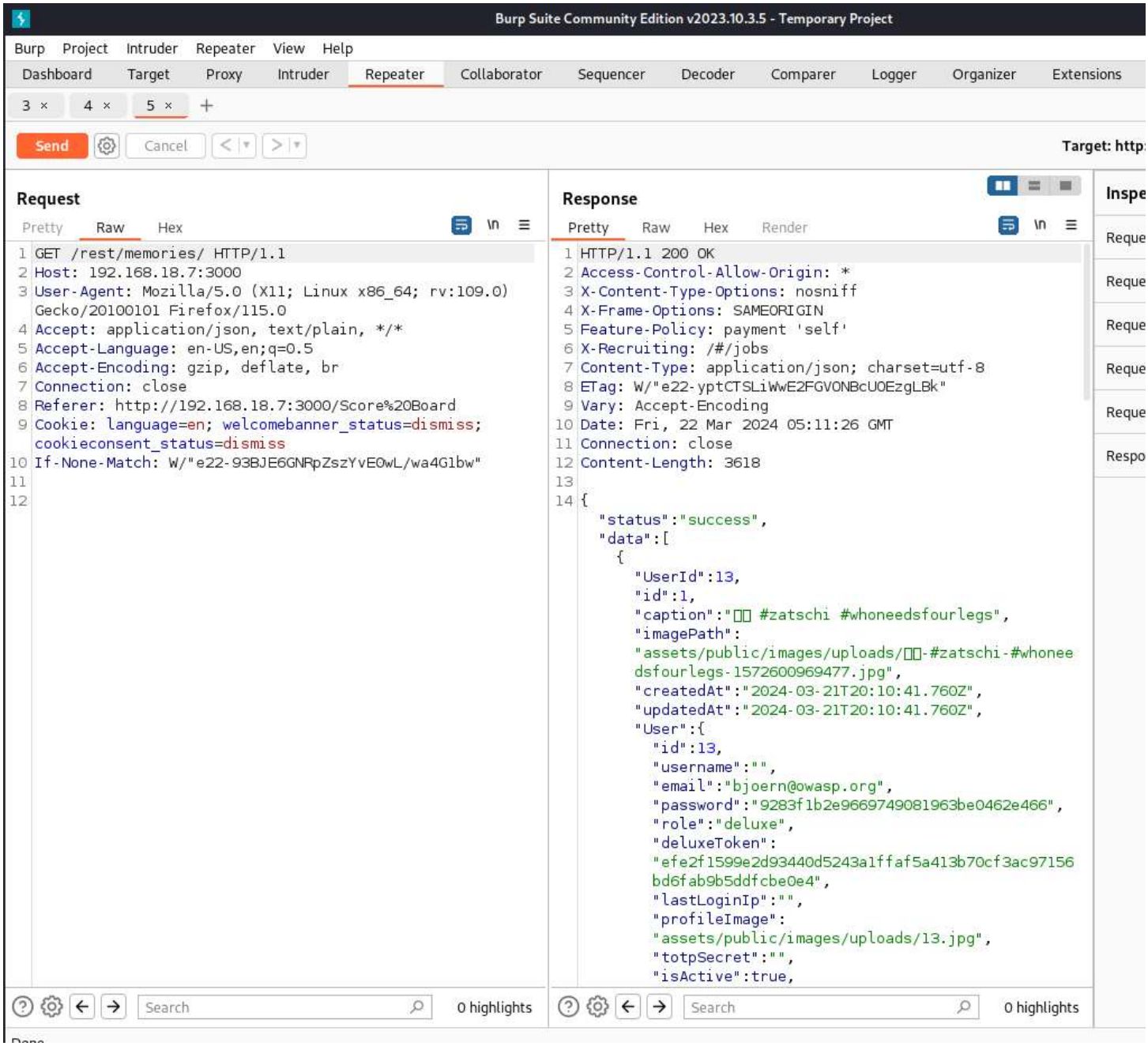
The Security Answer field must be included. Removing the field generates an error.



Finding	Severity	Description
	Minor	Input validation is not performed on the email, password, repeat password, and security question field. This could have unintended results on the database. Input validation should be performed server-side.

Discovered Credentials

While browsing to #/photo-wall, the response of /rest/memories/ could be seen. Reviewing this displayed the information of multiple users



"id":13,"username":"","email":"bjoern@owasp.org","password":"9283f1b2e9669749081963be0462e466","role":"deluxe","deluxeToken":"efe2f1599e2d93440d5243a1ffaf5a413b70cf3ac97156bd6fab9b5ddfcbe0e4","lastLoginIp":"","profileImage":"assets/public/images/uploads/13.jpg",

"id":4,"username":"bkimminich","email":"bjoern.kimminich@gmail.com","password":"6edd9d726cbdc873c539e41ae8757b8c","role":"admin","deluxeToken":"","lastLoginIp":"","profileImage":"assets/public/images/uploads/defaultAdmin.png",

"id":21,"username":"evmrox","email":"ethereum@juice-sh.op","password":"2c17c6393771ee3048ae34d6b380c5ec","role":"deluxe","deluxeToken":"b49b30b294d8c76f5a34fc243b9b9cccb057b3f675b07a5782276a547957f8ff","lastLoginIp":"","profileImage":"assets/public/images/uploads/default.svg",

"id":18,"username":"j0hNny","email":"john@juice-sh.op","password":"00479e957b6b42c459ee5746478e4d45","role":"customer","deluxeToken":"","lastLoginIp":"","profileImage":"assets/public/images/uploads/default.svg",

"id":19,"username":"E=ma²","email":"emma@juice-sh.op","password":"402f1c4a75e316afec5a6ea63147f739","role":"customer","deluxeToken":"","lastLoginIp":"","profileImage":"assets/public/images/uploads/default.svg",

Reviewing the password indicates this is likely an MD5 hash.

Finding	Severity	Description
	Major	Password hashes should not be included in the HTTP response for any reason. Exposing password hashes leaves them susceptible to password cracking attacks.

Finding	Severity	Description
	Minor	MD5 is not suitable for password hashing as it is cryptographically weak. A modern hashing algorithm such as SHA256 should be used.

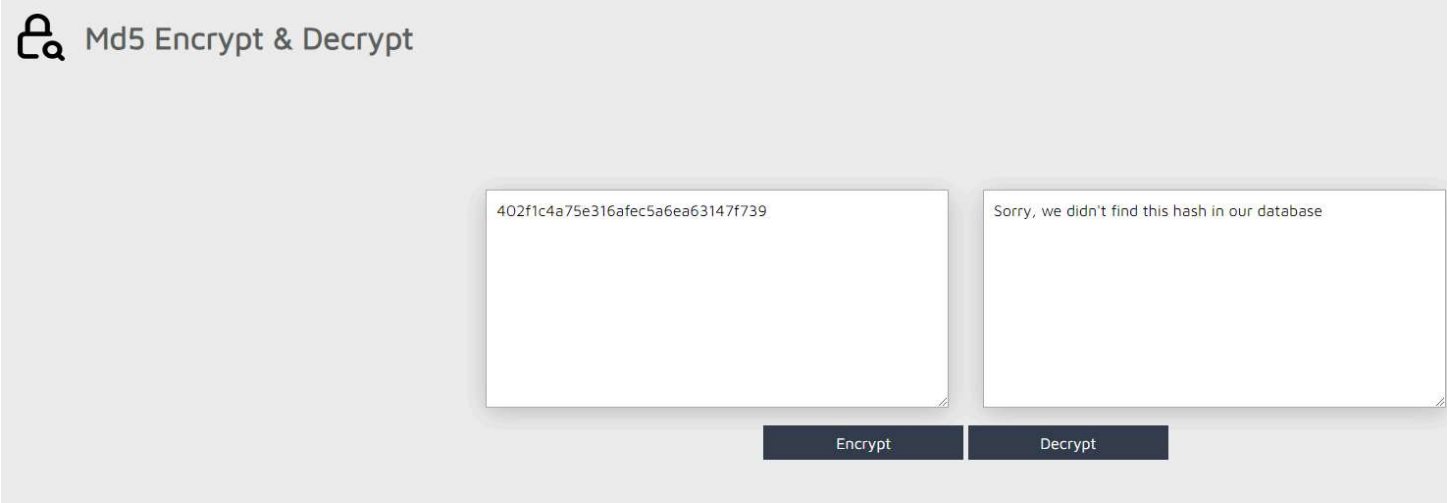
Note: MD5 is acceptable to use for software integrity purposes.

Finding	Severity	Description
	Minor	Unsalted password hashes are susceptible to rainbow table attacks. Password hashes should be salted

The five hashes were recorded and hashcat was used to crack them. The password hashes were attacked in a straight mode using rockyou.txt as the wordlist. The password “private” was found for ethereum@juice-sh.op.

```
kali@kali: ~
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~]
└─$ hashcat -a 0 -m 0 JuiceShopHash /usr/share/wordlists/rockyou.txt
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 4.0+debian  Linux, None+Asserts, RELOC, SPIR, LLVM 15.0.7, SLEEF, DISTRO, POCL_D
EBUG) - Platform #1 [The pocl project]
====================================================================================================
* Device #1: cpu-sandybridge-Intel(R) Core(TM) i5-10400 CPU @ 2.90GHz, 2909/5882 MB (1024 MB allocatable), 8
MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hashes: 5 digests; 5 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Salt
* Raw-Hash

ATTENTION! Pure (unoptimized) backend kernels selected.
Pure kernels can crack longer passwords, but drastically reduce performance.
If you want to switch to optimized kernels, append -O to your commandline.
See the above message to find out about the exact limits.

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 2 MB

Dictionary cache built:
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344392
* Bytes.....: 139921507
* Keyspace..: 14344385
* Runtime...: 2 secs

2c17c6393771ee3048ae34d6b380c5ec:private
Approaching final keyspace - workload adjusted.
```

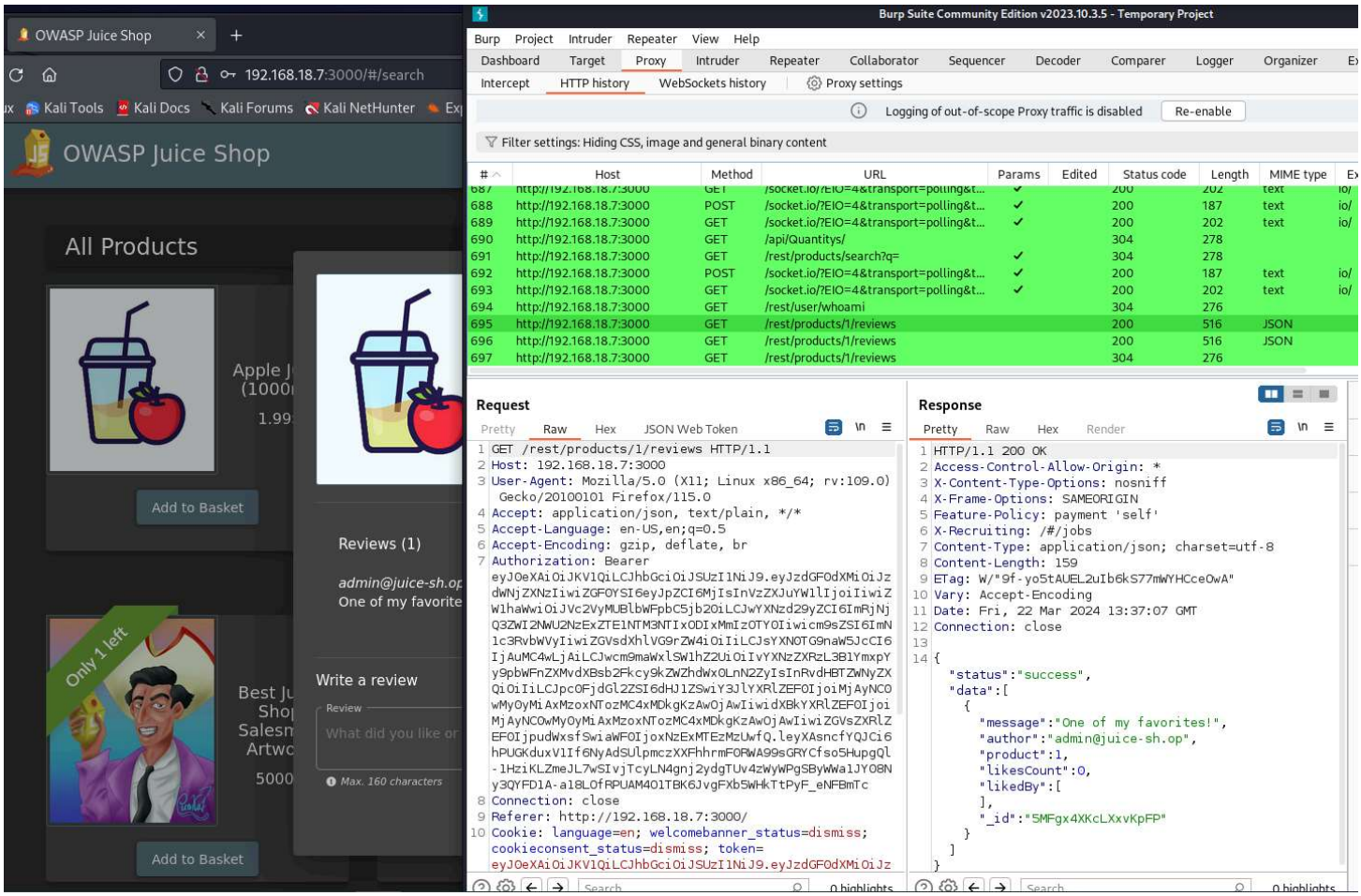
An online database of md5 hashes was checked for any previously discovered hashes matching the ones we have. Looking at https://md5decrypt.net/en/ this did not reveal any new passwords.



Browsing the reviews of the products at the main page shows the author and id in the HTTP response. The following users were found:

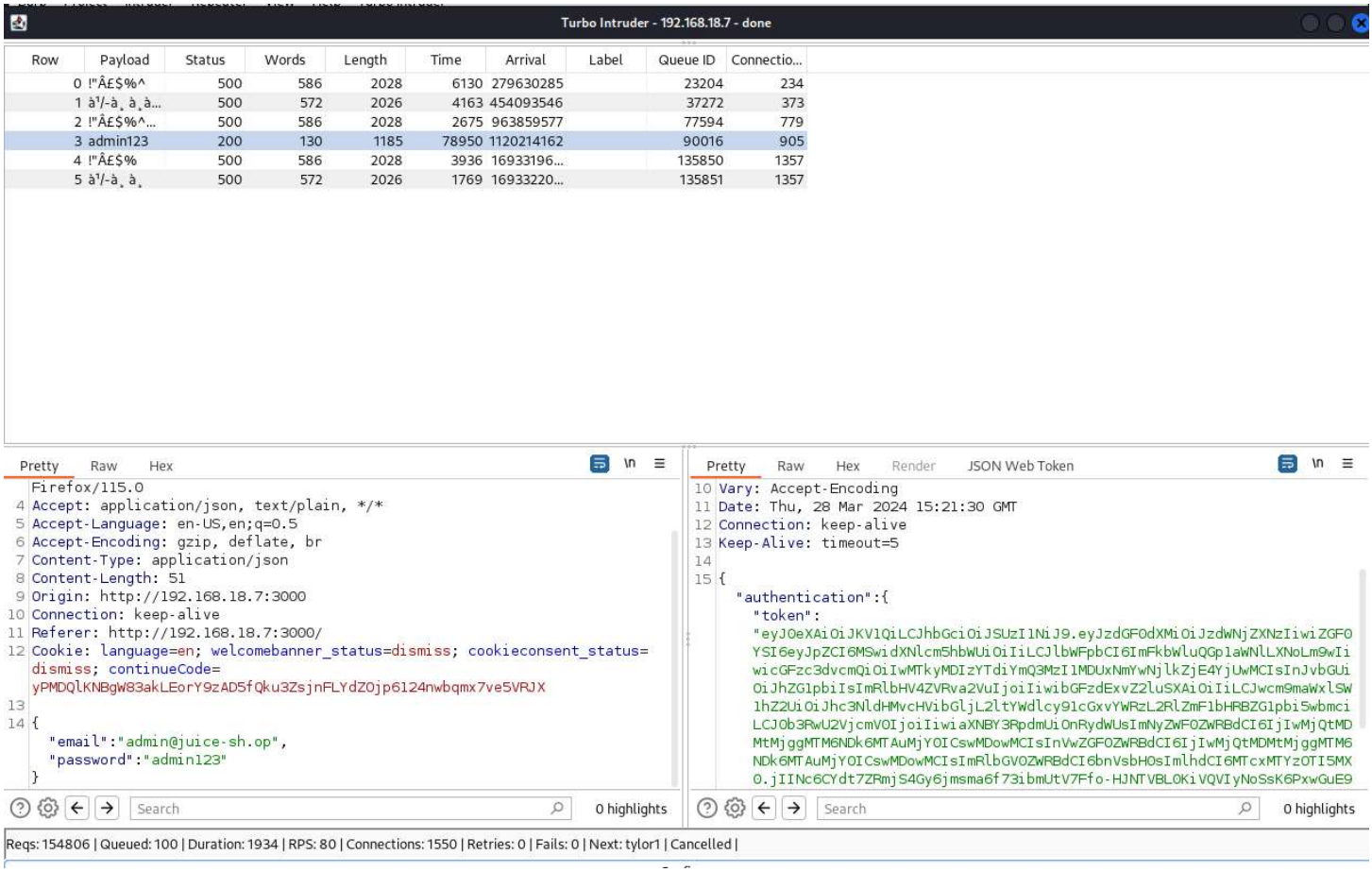
"author":"admin@juice-sh.op","product":1,"likesCount":0,"likedBy":[],"_id":"5MFgx4XKcLXxvKpFP"
"author":"bender@juice-sh.op","product":6,"likesCount":0,"likedBy":[],"_id":"KQsRcyx4i2aX9AeNc"
"author":"stan@juice-sh.op","product":42,"likesCount":0,"likedBy":[],"_id":"cmJRge6y9gqde3Gvq"
"author":"bender@juice-sh.op","product":42,"likesCount":0,"likedBy":[],"_id":"RNzAREe7d8MgrxYxG"
"author":"uvogin@juice-sh.op","product":30,"likesCount":0,"likedBy":[],"_id":"Gj8BdXpCPLooWnoPx"
"author":"admin@juice-sh.op","product":3,"likesCount":0,"likedBy":[],"_id":"snuWKLtkQgvFa3FRj"
"author":"jim@juice-sh.op","product":22,"likesCount":0,"likedBy":[],"_id":"KToEBigKHWqitoCkv"
"author":"mc.safesearch@juice-sh.op","product":41,"likesCount":0,"likedBy":[],"_id":"fH7EsETiRWyRuL7Hv"
"author":"uvogin@juice-sh.op","product":38,"likesCount":0,"likedBy":[],"_id":"FXxZRCfd2E674yuqq"
"author":"accountant@juice-sh.op","product":43,"likesCount":0,"likedBy":[],"_id":"4GpSEsmFYXz4vDgcb"
"author":"bjoern@owasp.org","product":35,"likesCount":0,"likedBy":[],"_id":"e5roe9zfEoEXgTjqD"
"author":"morty@juice-sh.op","product":32,"likesCount":0,"likedBy":[],"_id":"xrmx4sareo2nJjH8A"

These “id” do not appear to be hashes. Of note is that the id appeared to vary with the same user across different reviews.

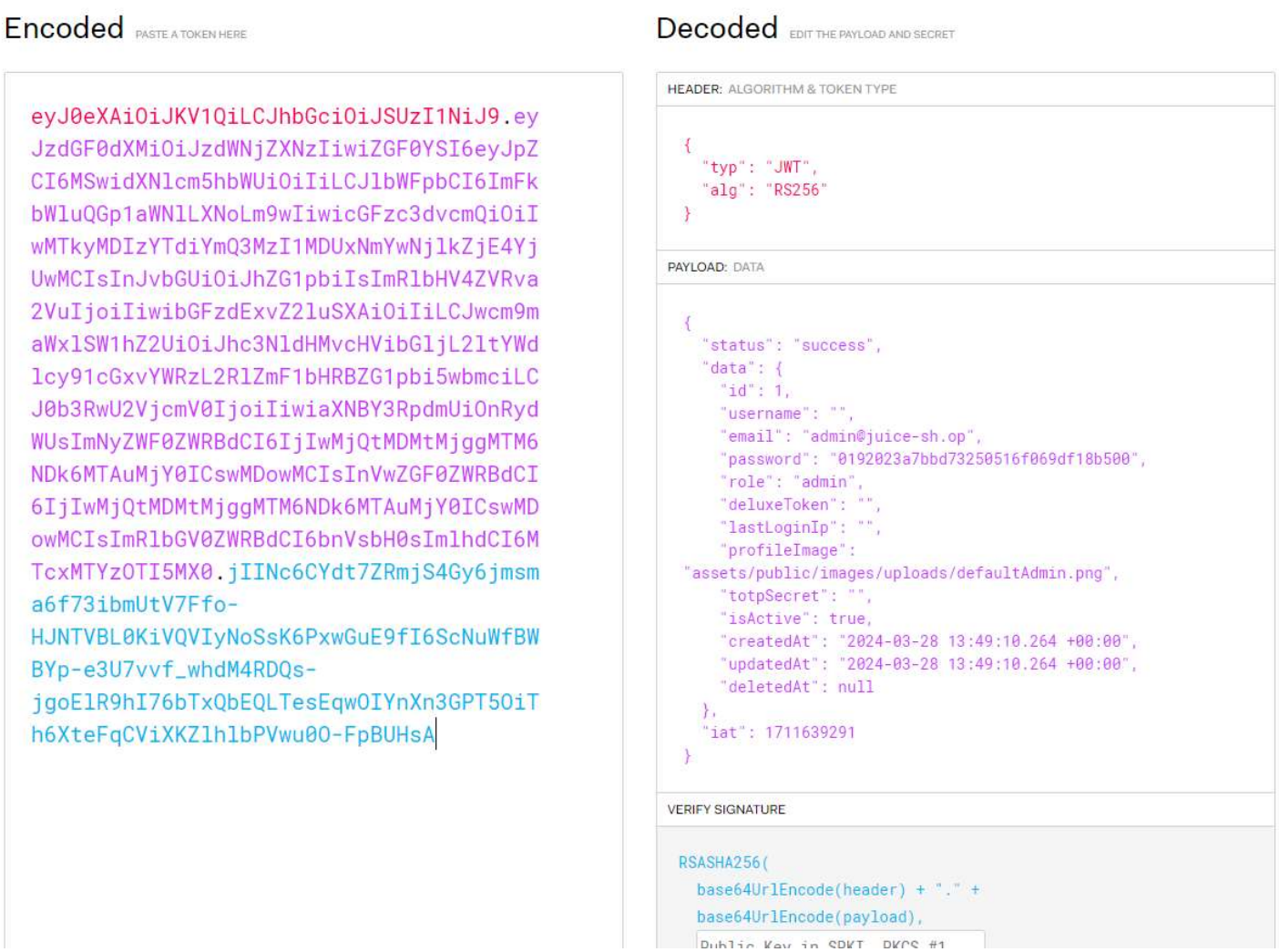


The discovered account emails were checked for previously leaked passwords on haveibeenpwned.com. No password breaches were discovered.

An attempt was made to brute force login attempts on the discovered users. 75,000 requests were made on bjoern@owasp.org before the attempt was aborted.
The password for admin@juice-sh.op was found to be admin123, after 90,016 requests.



With the admin jwt appearing as such.



Finding	Severity	Description
	Major	Administrator credentials should use strong passwords to reduce the effectiveness of password guessing and automated attacks. The administrator password should be changed immediately.

Finding	Severity	Description
	Major	Administrator authentication should use multi-factor authentication to significantly reduce the effectiveness of automated attacks.

Summary Table of Discovered Credentials

Id	Username	Email	Password hash	Password
13		bjoern@owasp.org	9283f1b2e9669749081963be0462e466	
4	bkimminich	bjoern.kimminich@gmail.com	6edd9d726cbdc873c539e41ae8757b8c	
21	evmrox	ethereum@juice-sh.op	2c17c6393771ee3048ae34d6b380c5ec	private

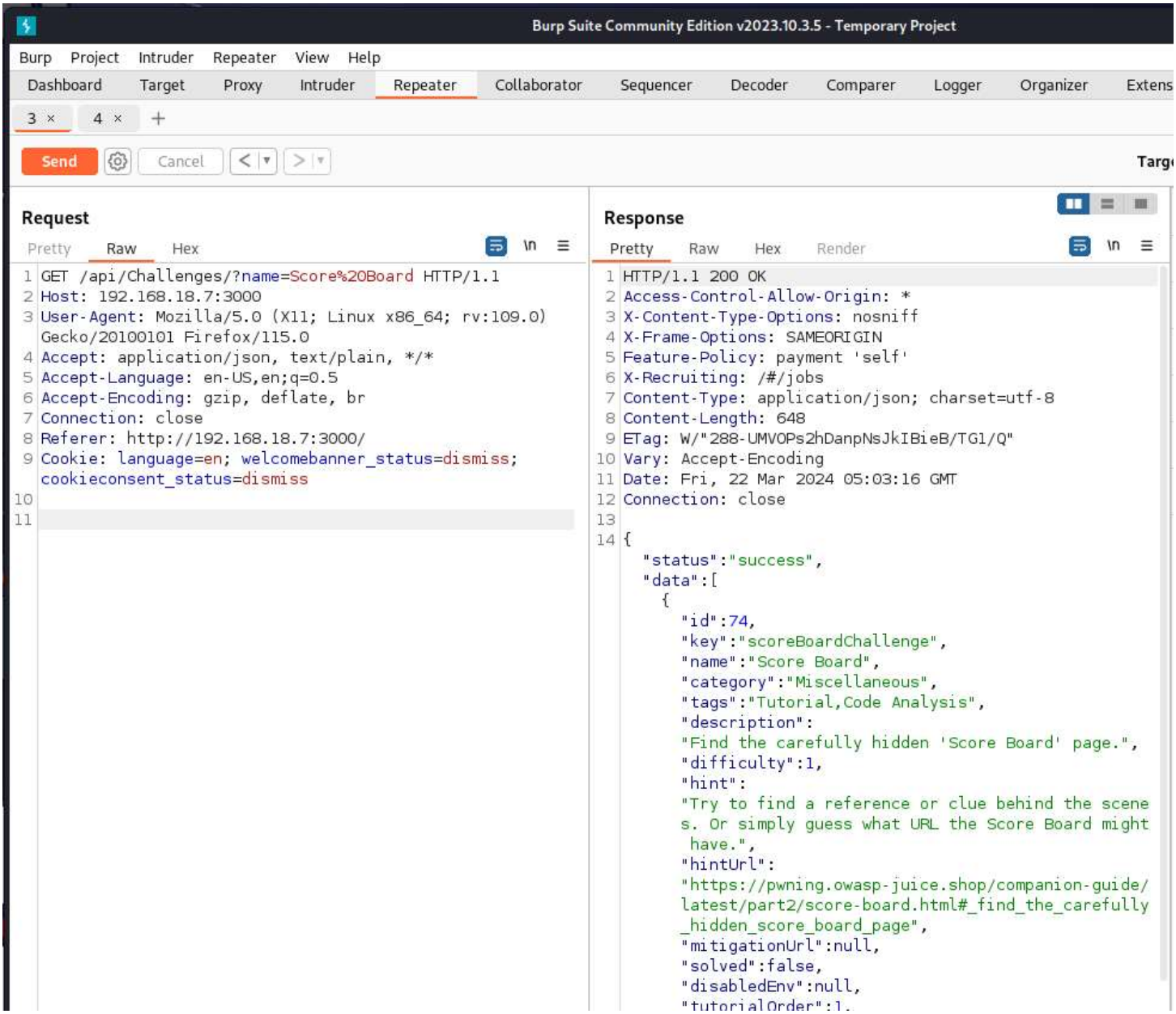
18	j0hNny	john@juice-sh.op	00479e957b6b42c459ee5746478e4d45	
19	E=ma²	emma@juice-sh.op	402f1c4a75e316afec5a6ea63147f739	
		admin@juice-sh.op		admin123
		bender@juice-sh.op		
		stan@juice-sh.op		
		uvogin@juice-sh.op		
		jim@juice-sh.op		
		mc.safesearch@juice-sh.op		
		accountant@juice-sh.op		
		morty@juice-sh.op		

Directory Traversal

Looking at the images on the main page, there were no images loaded by a GET request. Similarly, on the #/about, #/photo-wall, the images were loaded in the same way as a response from /rest/admin/application-configuration and there is no opportunity for directory traversal.

No issue:	No resources loaded through GET requests led to directory traversal.

Directory Traversal was manually attempted on the name parameter of /api/Challenges, with no vulnerabilities found



Directory traversal was attempted using DotDotPwn. The application was configured as such.

No successful results were found as the URL redirects to the main Juice Shop page.

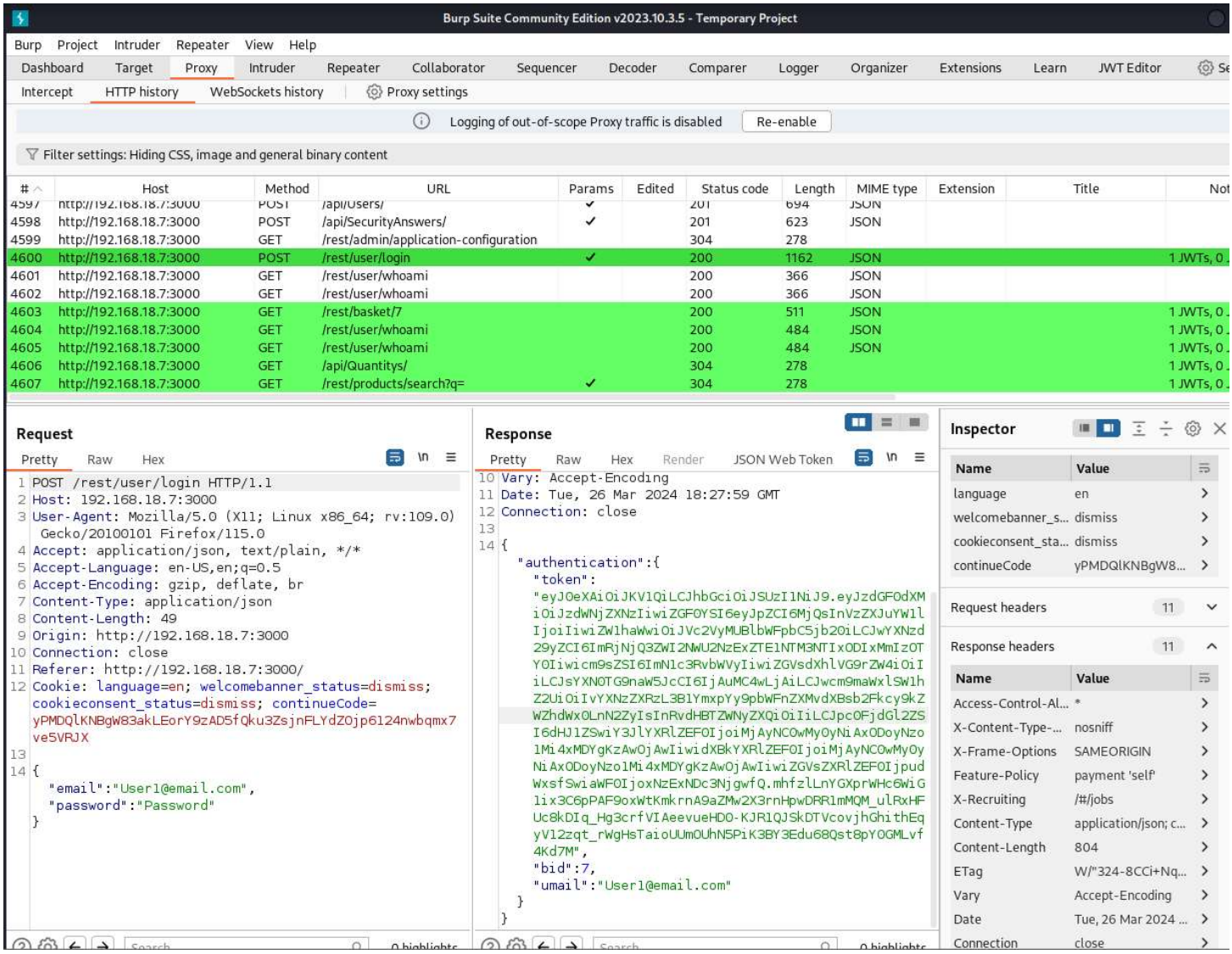
No issue:	No HTTP GET requests led to directory traversal.

Directory Traversal was attempted with the discovered ftp directory. Upon further investigation it appears to a directory named ftp rather than an ftp service that could be susceptible to directory traversal.

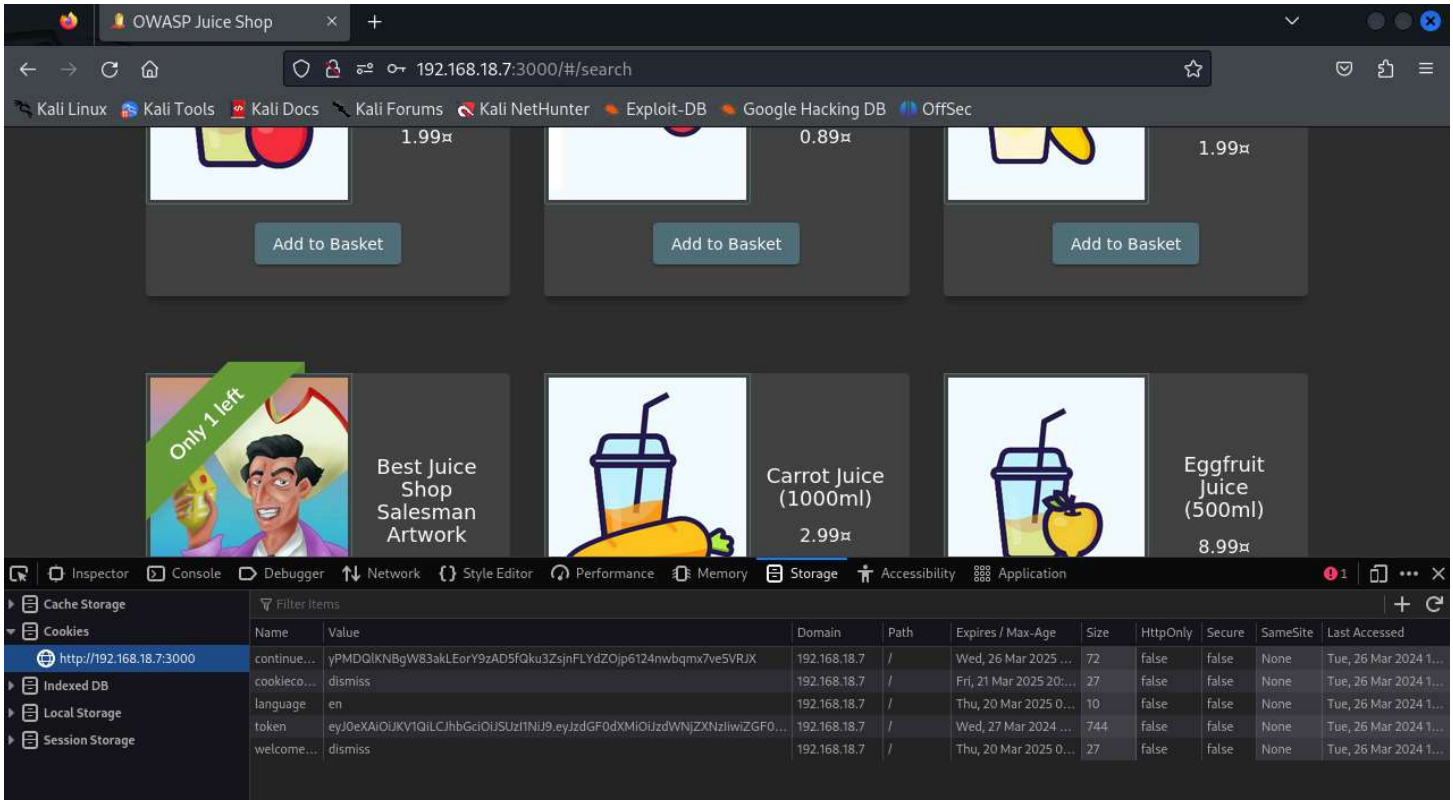
No issue:	No directory traversal could be performed though ftp.

Session Management

Successful authentication generates a JWT.



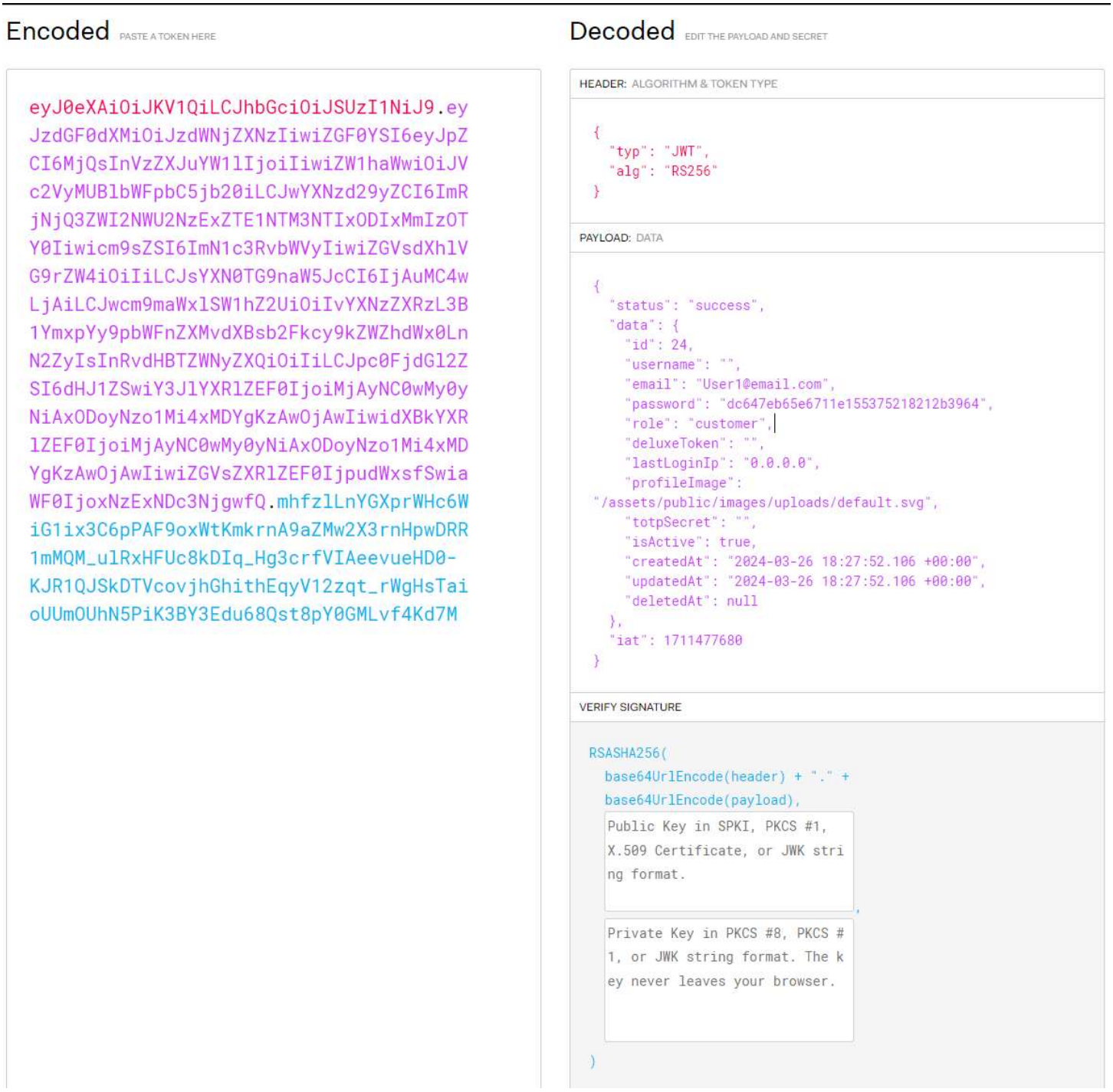
Finding	Severity	Description
	Info	JWT is not ideal for session management. This does not follow best practices. A randomised session identifier should be used.



Finding	Severity	Description
	Major	The session token is sent without the HttpOnly attribute. This attribute prevents client-side scripts from accessing the cookie.

Finding	Severity	Description
	Info	The session token is sent without the Secure attribute. This attribute prevents the token from being sent over an insecure connection. Best practice would be to set this attribute to True. However, this can’t be performed currently as TLS connection is not enabled.

Decoding the JWT reveals the following information:

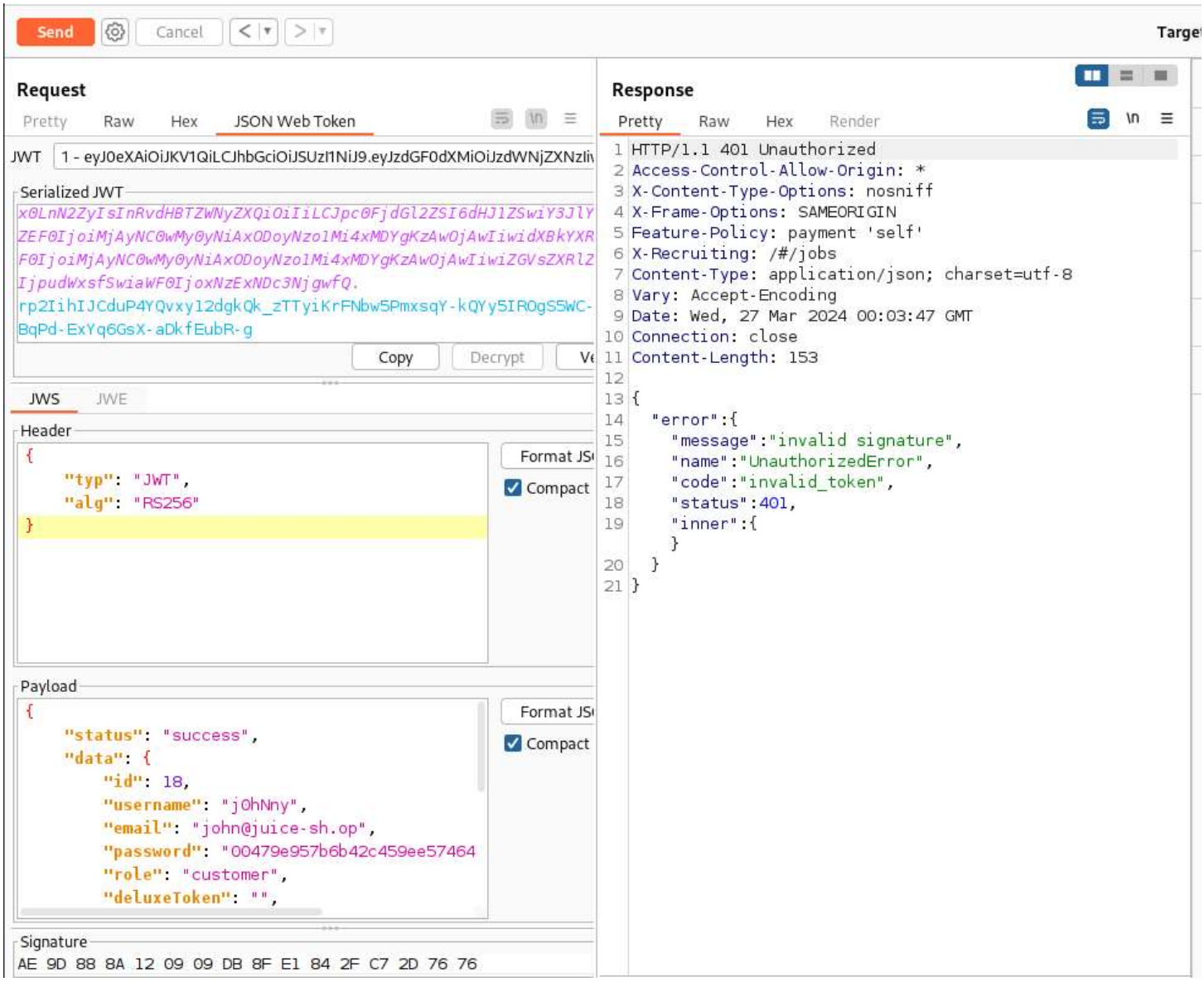


The JWT does not have an expiry period set. The JWT was also observed to function after over 12 hours of inactivity. This suggests that there is no server-side expiry either.

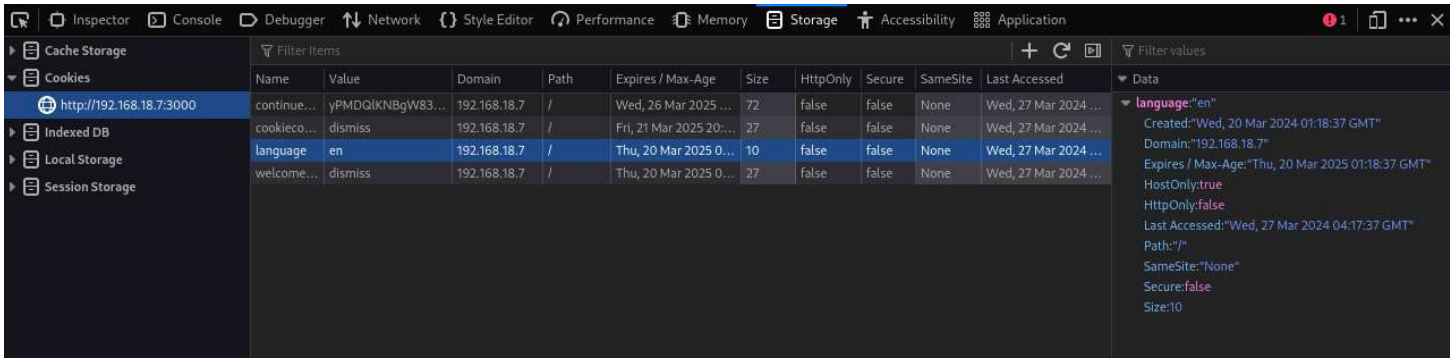
Finding	Severity	Description
	Minor	The JWT has no expiry period. This increases exposure to session hijacking. This is a typical issue concerning JWTs. Session IDs should be used instead.

No issue:	RS256 (RSA SHA-256) is a standard and acceptable signing cryptography to use.

An attempt was made to edit the JWT with the information of another user. The details of user "john@juice-sh.op" that was discovered during the site mapping portion of the penetration test was entered. The token was signed by a self-generated RS256 key. The attack was unsuccessful. Further testing

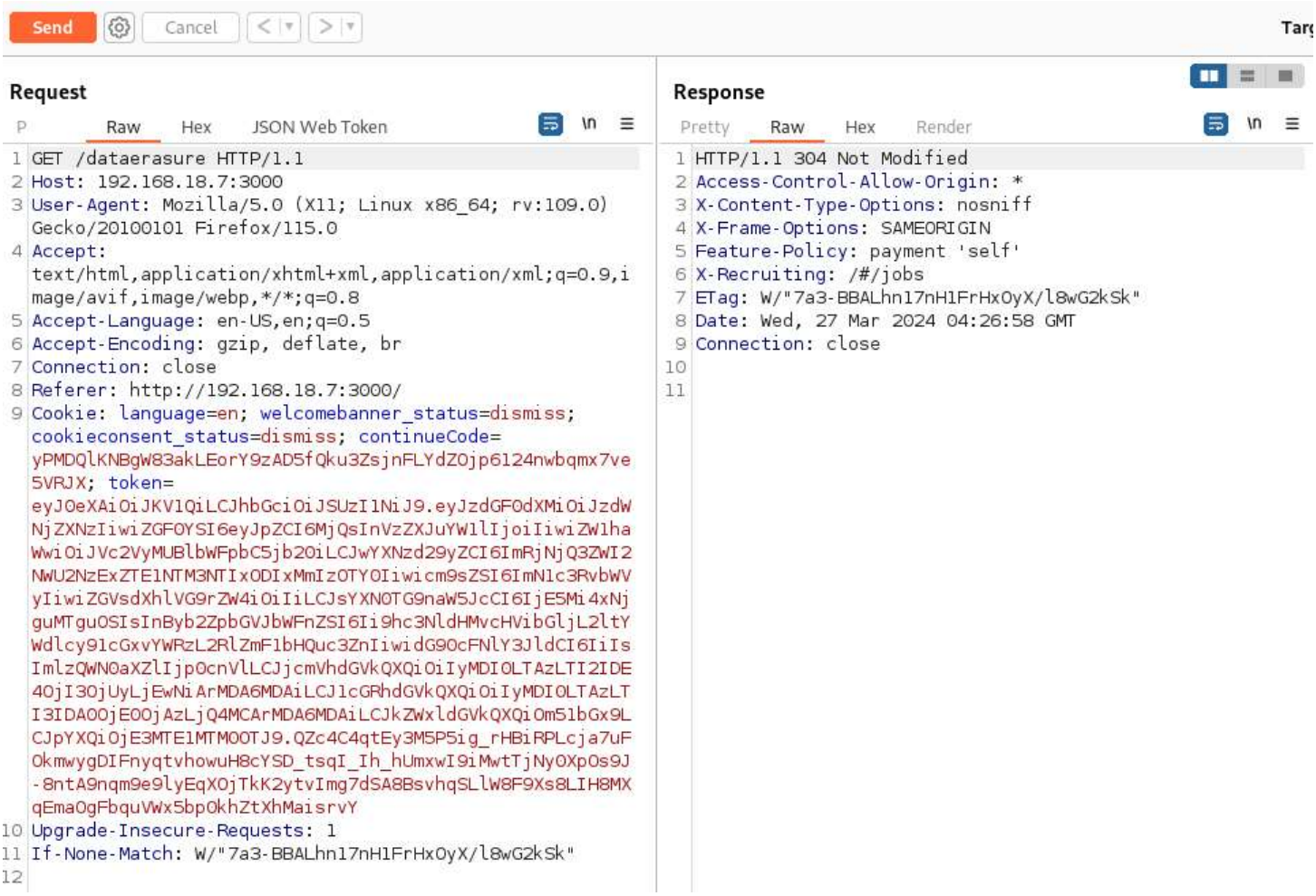


Logging out of the application removes the session token from the browser.



No issue:	The client-side removal of session tokens upon logging out follows best practices.

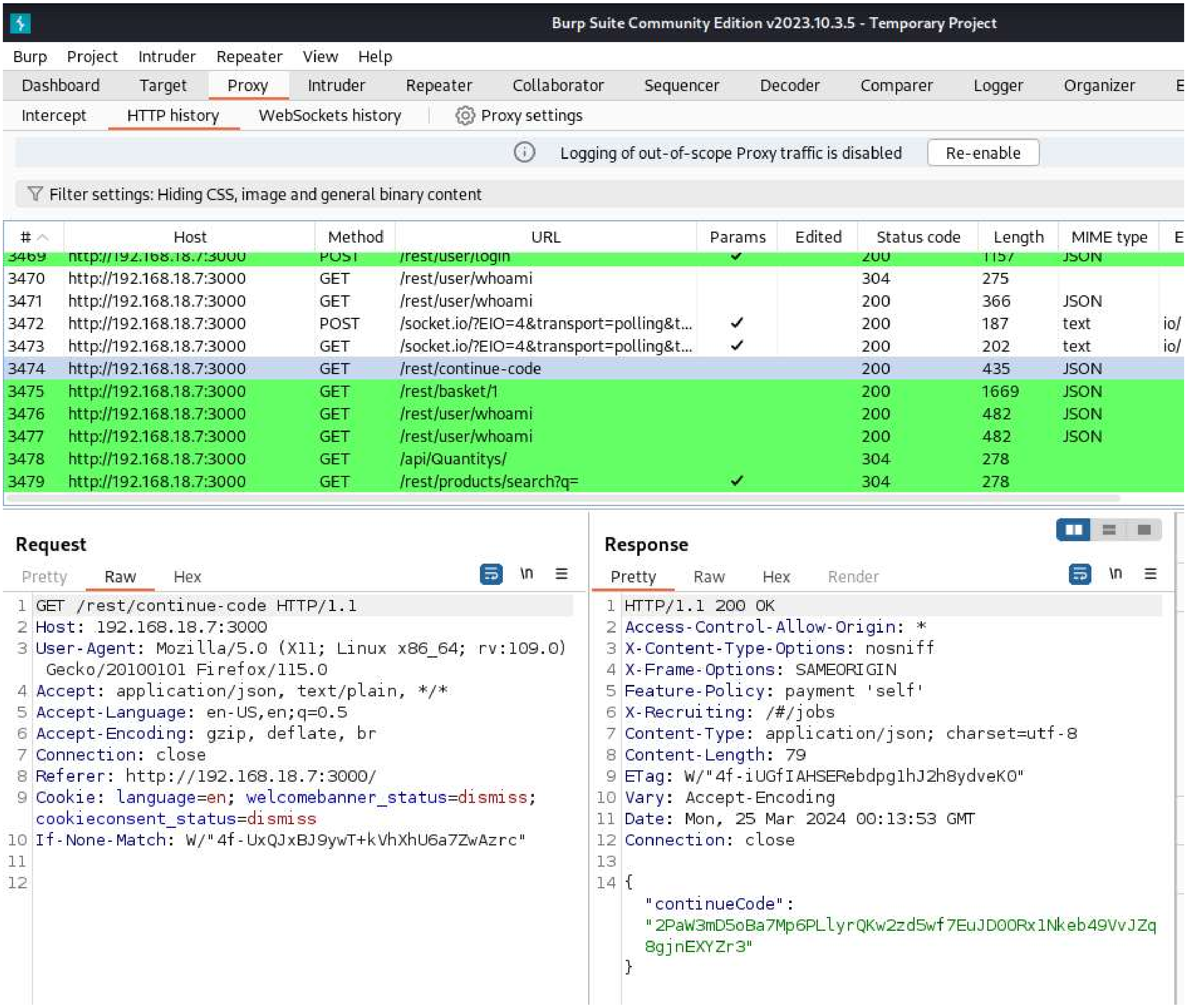
It was still possible to visit /dataerasure, a page that should only be accessible while authenticated. This indicates the token was not invalidated server-side.

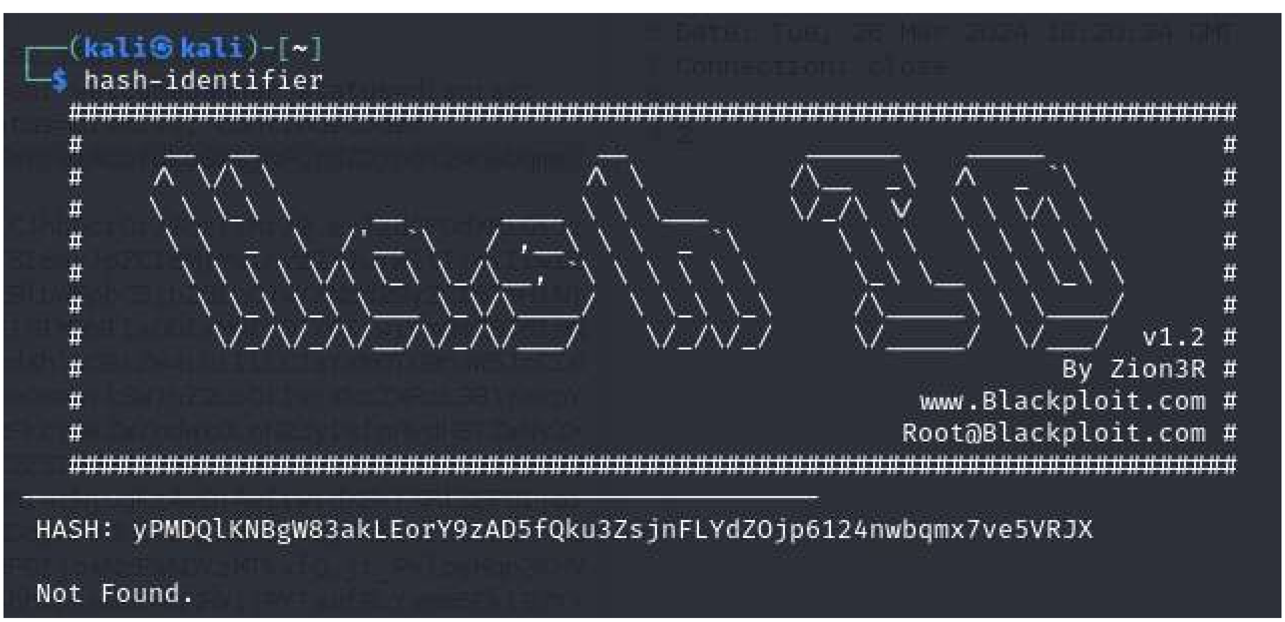


The same result could also be achieved by directly adding the token into the browser.

Finding	Severity	Description
	Minor	The JWT is not invalidated after logging out. This increases exposure to session hijacking. This is a typical issue concerning JWTs. JWT should be invalidated on both client-side and server-side upon logout. Alternatively, Session IDs should be used instead.

Completing an OWASP Juice Shop challenge for the first time generates a “continueCode”. The continueCode cookie keeps track of progress on the challenges. This is not a JWT and inputting the value in hash-identifier did not return any results.

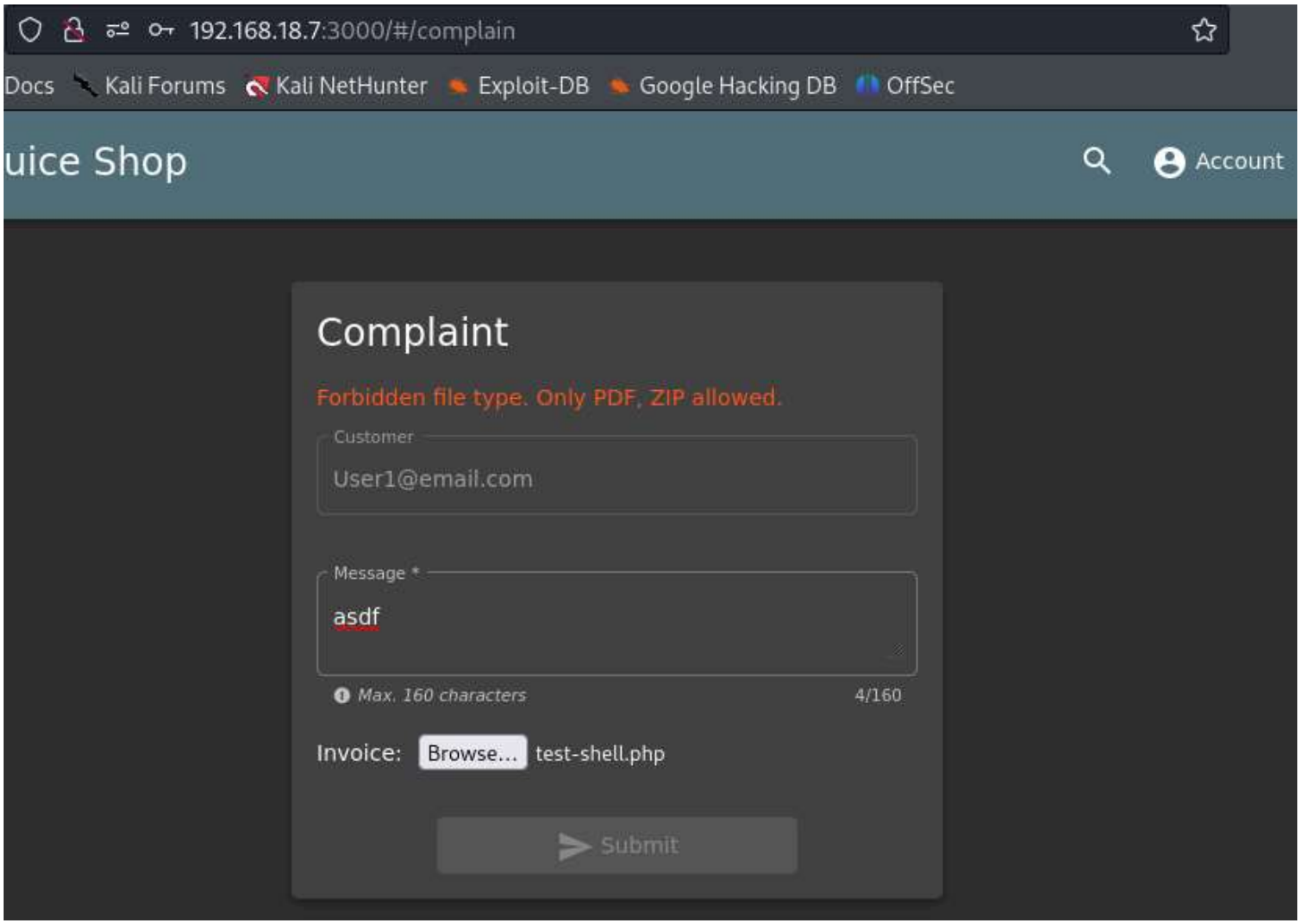

Burp Suite Community Edition v2023.10.3.5 - Temporary Project
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer
Intercept HTTP history WebSockets history Proxy settings
Logging of out-of-scope Proxy traffic is disabled
Re-enable
Filter settings: Hiding CSS, image and general binary content
Host Method URL Params Edited Status code Length MIME type
3470 http://192.168.18.7:3000 GET /rest/user/whoami 304 275
3471 http://192.168.18.7:3000 GET /rest/user/whoami 200 366 JSON
3472 http://192.168.18.7:3000 POST /socket.io/?EIO=4&transport=polling&t... ✓ 200 187 text
3473 http://192.168.18.7:3000 GET /socket.io/?EIO=4&transport=polling&t... ✓ 200 202 text
3474 http://192.168.18.7:3000 GET /rest/continue-code 200 435 JSON
3475 http://192.168.18.7:3000 GET /rest/basket/1 200 1669 JSON
3476 http://192.168.18.7:3000 GET /rest/user/whoami 200 482 JSON
3477 http://192.168.18.7:3000 GET /rest/user/whoami 200 482 JSON
3478 http://192.168.18.7:3000 GET /api/Quantitys/ 304 278
3479 http://192.168.18.7:3000 GET /rest/products/search?q= ✓ 304 278
Request
Pretty Raw Hex
GET /rest/continue-code HTTP/1.1
Host: 192.168.18.7:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: close
Referer: http://192.168.18.7:3000/
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss
If-None-Match: W/"4f-UxQJxBJ9ywT+kVhXhU6a7ZwAzrc"
Response
Pretty Raw Hex Render
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Content-Type: application/json; charset=utf-8
Content-Length: 79
ETag: W/"4f-iUGfIAHSERebdpg1hJ2h8ydveK0"
Vary: Accept-Encoding
Date: Mon, 25 Mar 2024 00:13:53 GMT
Connection: close
{
"continueCode": "2PaW3mD5oBa7Mp6PLlyrQKw2zd5wf7EuJD0ORx1Nkeb49VvJZq8gjnEXYZr3"
}


(kali㉿kali)-[~]
$ hash-identifier
v1.2
By Zion3R
www.Blackploit.com
Root@Blackploit.com
HASH: yPMDQlKNBgW83akLEorY9zAD5fQku3ZsjnFLYdZOjp6124nwbqmx7ve5VRJX
Not Found.

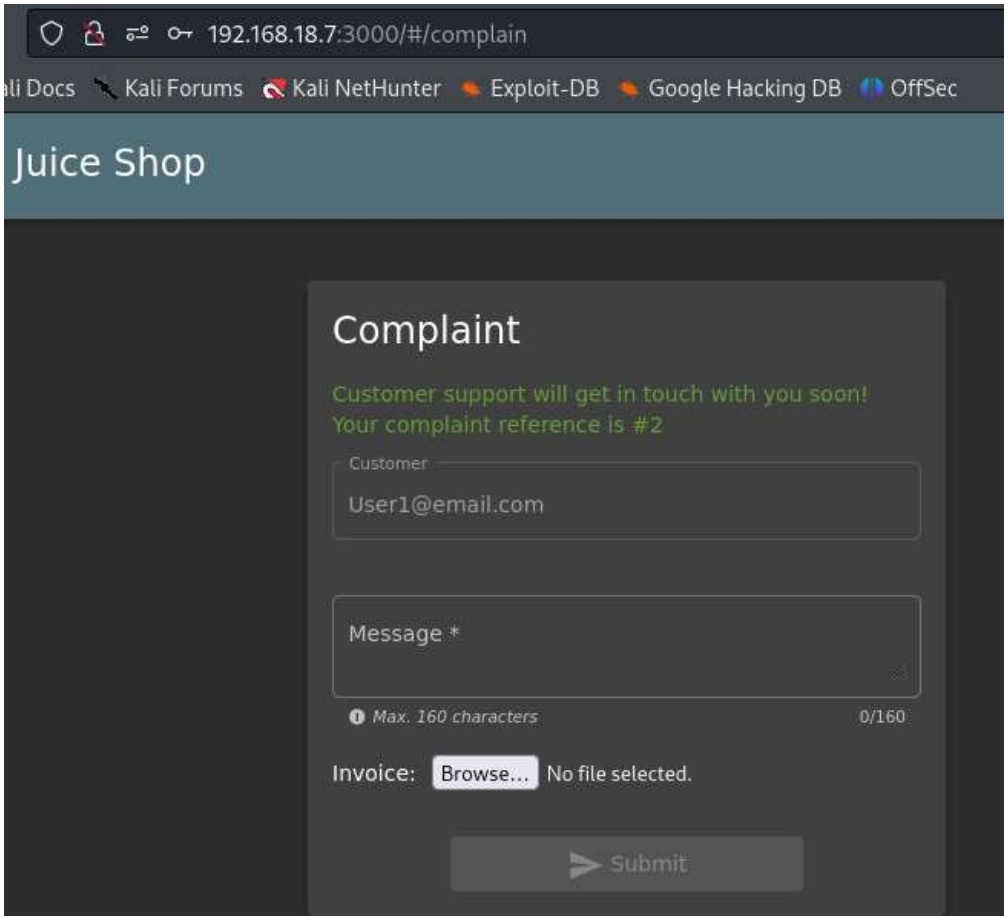
File Upload

Complaint

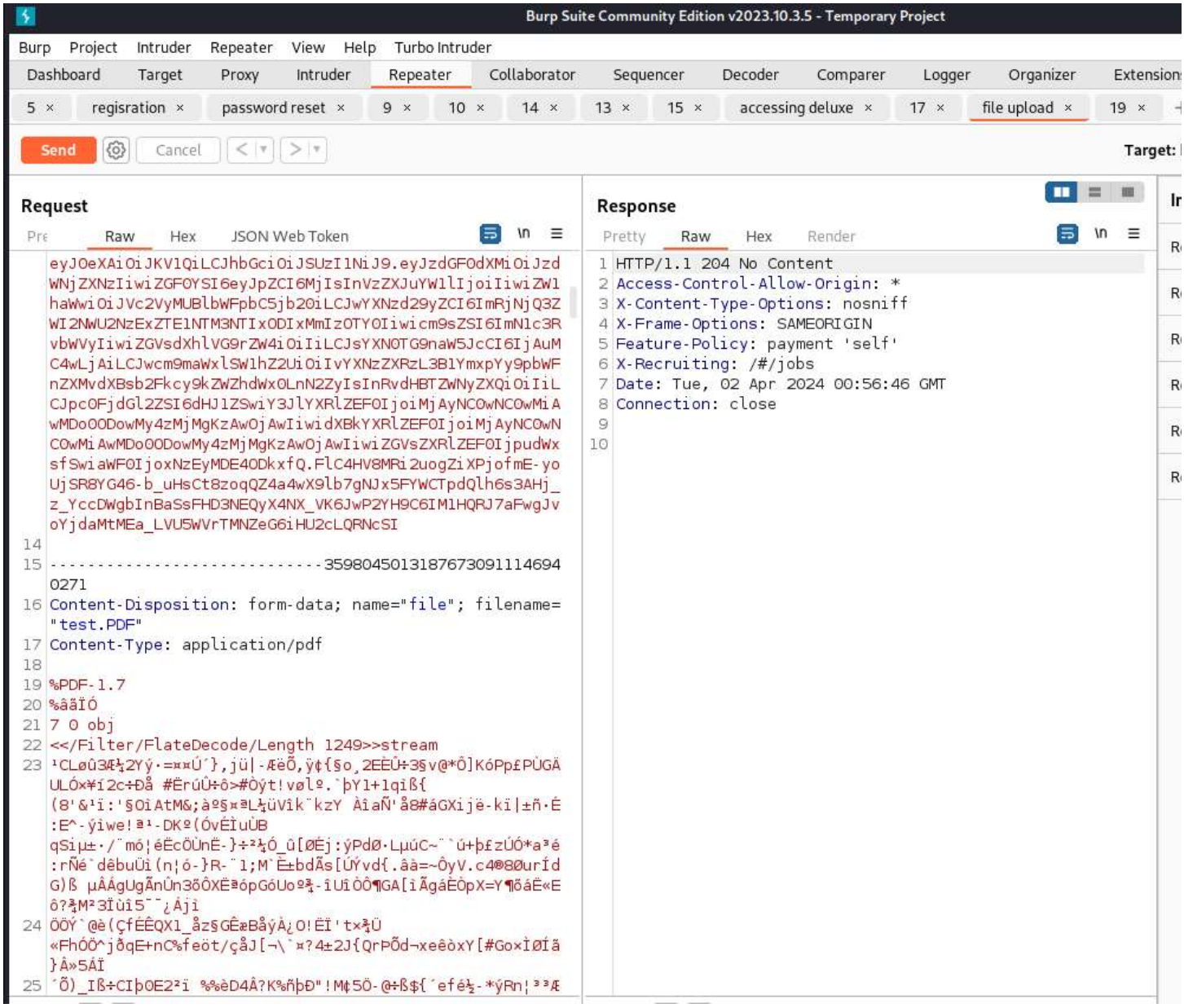
The #/complain directory was tested for file upload vulnerabilities. The file upload only allows .pdf or .zip file smaller than 100KB.



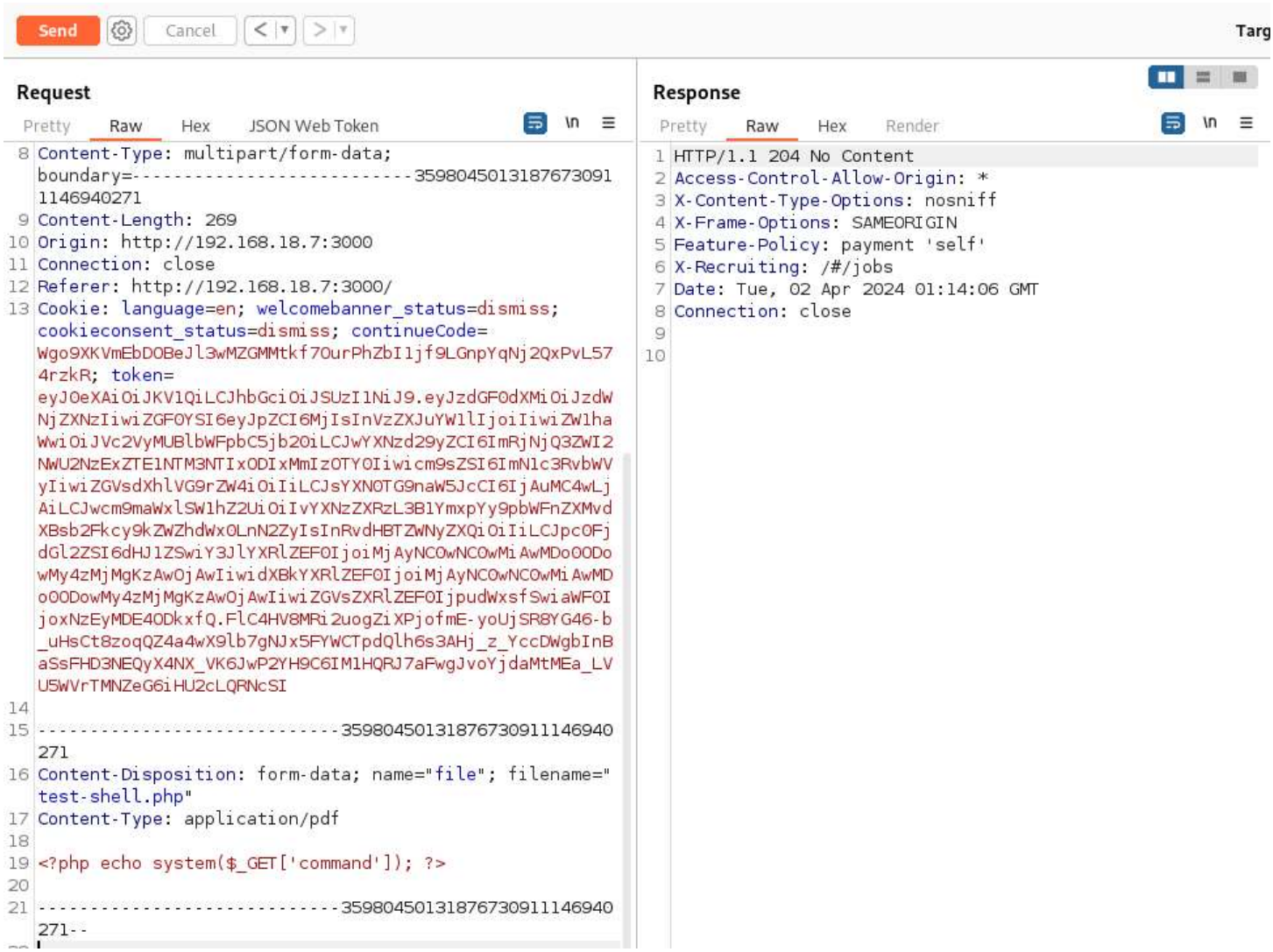
A test.pdf file was uploaded however the file could not be located.



We can see what the HTTP POST request for the file upload looks like.



The file type validation can be bypassed by modifying the post request. We are able to submit a php shell and see the same successful 204 response.

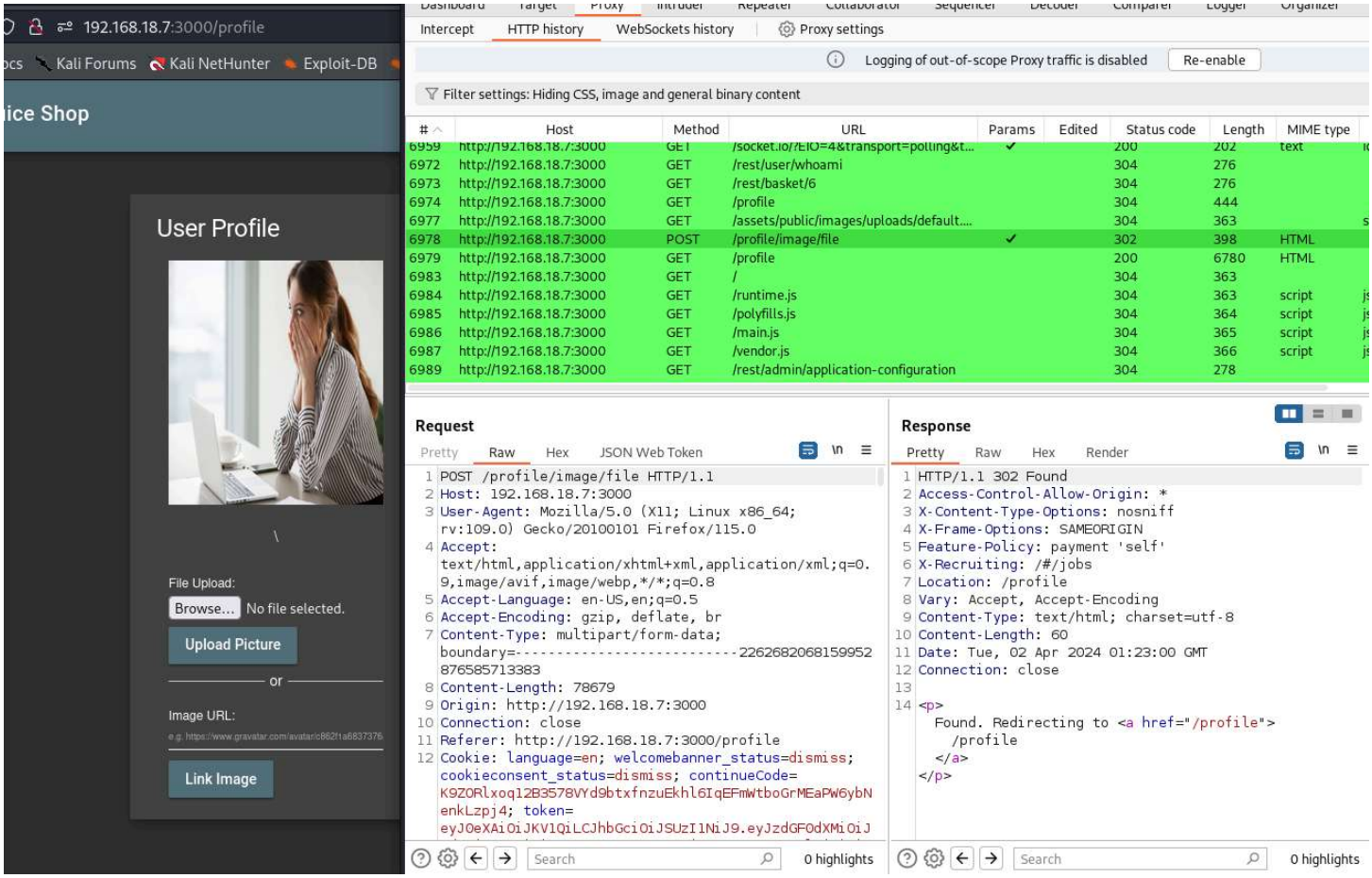


Finding	Severity	Description
	Major	The #/complaint file upload feature allows users to upload malicious files to the web server. This could be used by an attacker in combination with a directory traversal or SSRF to call the file for remote code execution . File type validation should be performed server-side.

We have not yet found a way to access the uploaded file.

User Profile

In the User Profile, a successful image upload in a POST request to /profile/image/file generates a 302 response. The upload should be limited to images only.



We are able to see the image gets renamed, in this case 22.jpg, and stored within assets/public/images/uploads/22.jpg and can be accessed through the URL

7113	http://192.168.18.7:3000	GET	/rest/basket/6			304	276	
7114	http://192.168.18.7:3000	GET	/rest/user/whoami			200	483	JSON
7115	http://192.168.18.7:3000	GET	/rest/user/whoami			200	483	JSON
7116	http://192.168.18.7:3000	GET	/api/Quantitys/			304	278	
7117	http://192.168.18.7:3000	GET	/rest/products/search?q=		✓	304	278	
7119	http://192.168.18.7:3000	GET	/api/Challenges/?key=nftMintChallenge		✓	304	277	
7120	http://192.168.18.7:3000	GET	/rest/web3/nftMintListen			304	276	
7121	http://192.168.18.7:3000	POST	/socket.io/?EIO=4&transport=polling&t...		✓	200	187	text
7122	http://192.168.18.7:3000	GET	/socket.io/?EIO=4&transport=polling&t...		✓	200	202	text
7123	http://192.168.18.7:3000	POST	/socket.io/?EIO=4&transport=polling&t...		✓	200	187	text

Request

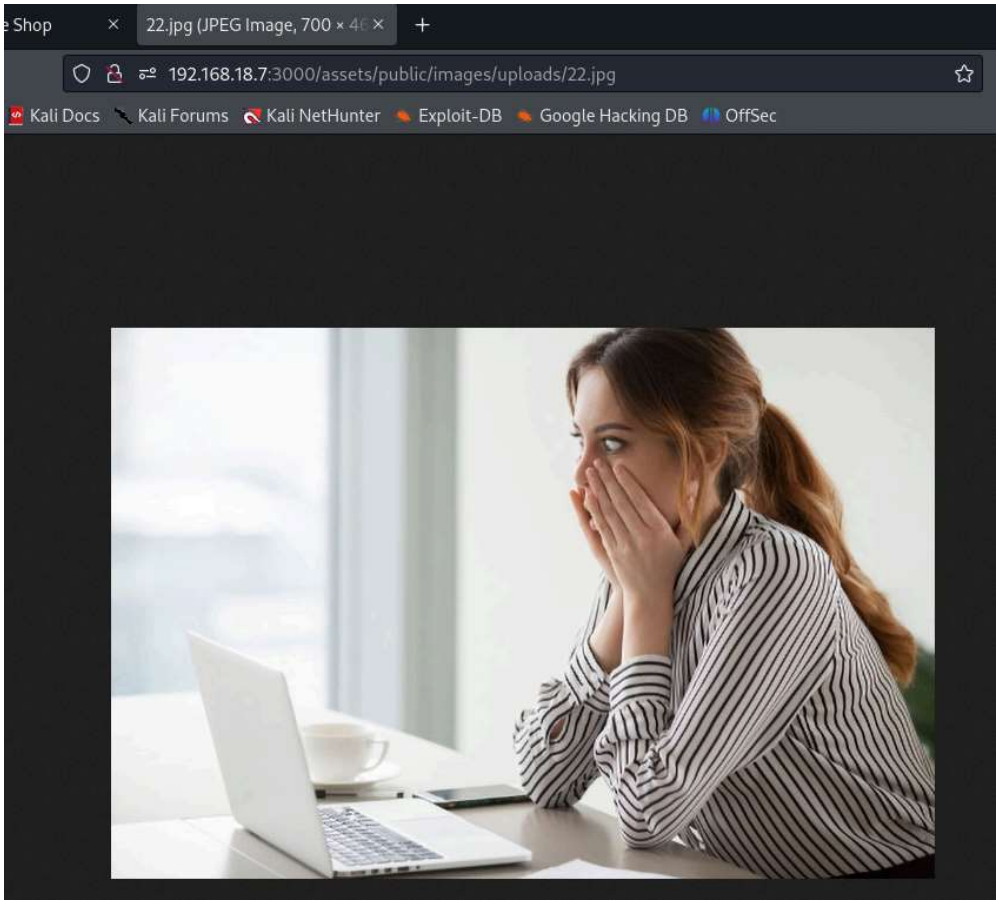
Pretty Raw Hex JSON Web Token

```
GET /rest/user/whoami HTTP/1.1
Host: 192.168.18.7:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0)
 Gecko/20100101 Firefox/115.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJz
dWNjZXNzIiwiZGF0YSI6eyJpZCI6MjIsInVzZXJuYW1lIjoiIiwiZ
W1haWwiOiJVc2VyMUBlbWFpbC5jb20iLCJwYXNzd29yZCI6ImRjNj
Q3ZWI2NWU2NzExZTE1NTM3NTIxODIxMmIzOTY0Iiwicm9sZSI6ImN
1c3RvbWVyIiwiZGVsdXhlVG9rZW4iOiIiLCJsYXN0TG9naW5JcCI6
IjE5Mi4xNjguMTguOSIsInByb2ZpbGVJbWFnZSI6ImFzc2V0cy9wd
WJsaWMvaW1hZ2VzL3VwbG9hZHMvMjIuanBnIiwidG9OcFNlY3JldC
I6IiIsImlzQWN0aXZlIjp0cnVlLCJjcmVhdGVkQXQiOiIyMDI0LTA
0LTAyIDAwOjQ4OjAzLjMyMyArMDA6MDAiLCJ1cGRhdGVkQXQiOiIy
MDI0LTA0LTAyIDAyOjU2OjMyLjE0MCArMDA6MDAiLCJkZWxldGVkQ
XQiOm51bGx9LCJpYXQiOjE3MTIwMjY2MzZ9.jisnVc5To5UE6fx7P
eP-8lts93r6a5ZXtx25NUa-t0BffPSrnGCJWT7q_q-m-Hkux-Vl61
nn2fAAMpc2zgHN_8NDHTdoN_EB6AtGG042xgKZ8VobZEF_NgtIau0
```

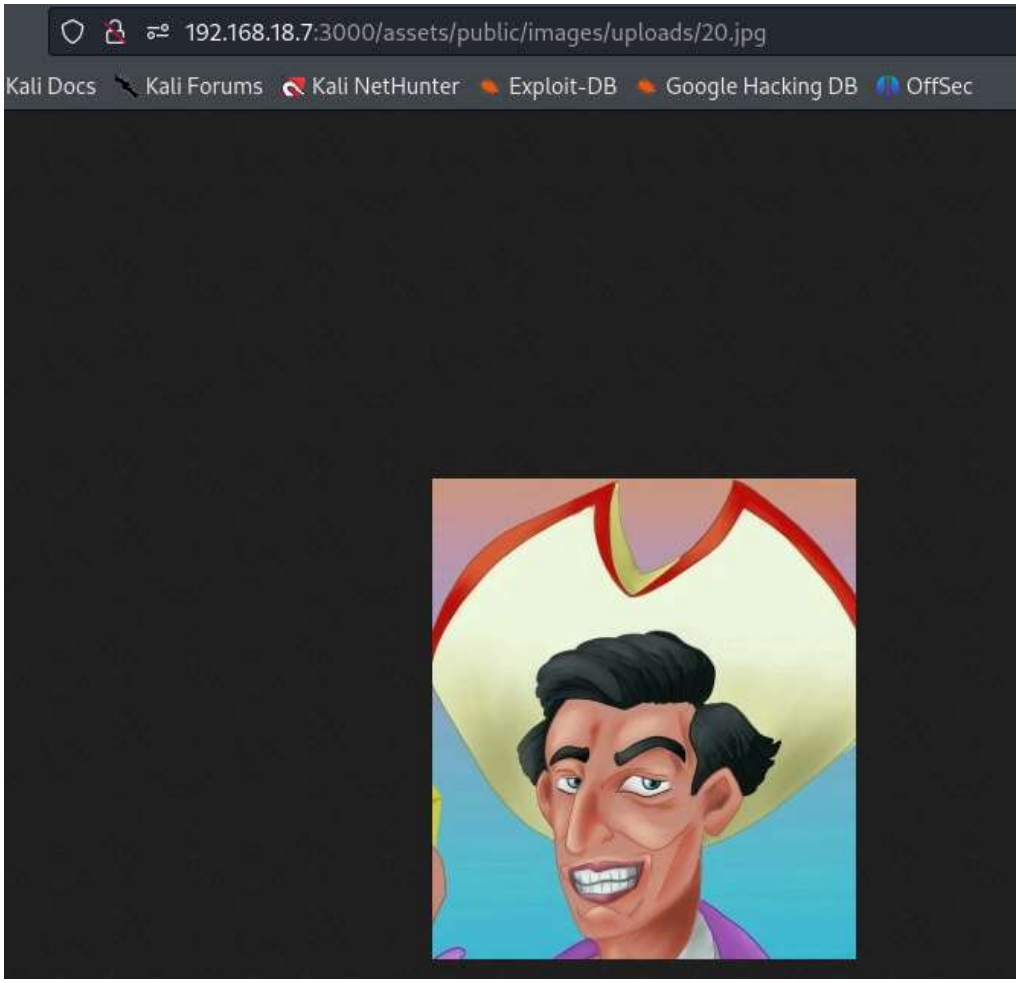
Response

Pretty Raw Hex Render

```
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Content-Type: application/json; charset=utf-8
Content-Length: 126
ETag: W/"7e-11UxYq/mq2A5tvSZzwD3bMS4mz4"
Vary: Accept-Encoding
Date: Tue, 02 Apr 2024 02:57:15 GMT
Connection: close

{
  "user":{
    "id":22,
    "email":"User1@email.com",
    "lastLoginIp":"192.168.18.9",
    "profileImage":
    "assets/public/images/uploads/22.jpg"
  }
}
```



With the predictable naming scheme, we could access another image in the folder, 20.jpg. Directory traversal to access other areas was not possible, as seen in the Directory Traversal section.



Finding	Severity	Description
	Minor	Predictable image naming scheme and lack of access control allows unintended access to assets. Implement appropriate server-side access control.

Uploading a different type of file in the profile image is blocked by an Illegal file type error. The control could not be bypassed by manipulating the Content-Type: image/jpeg

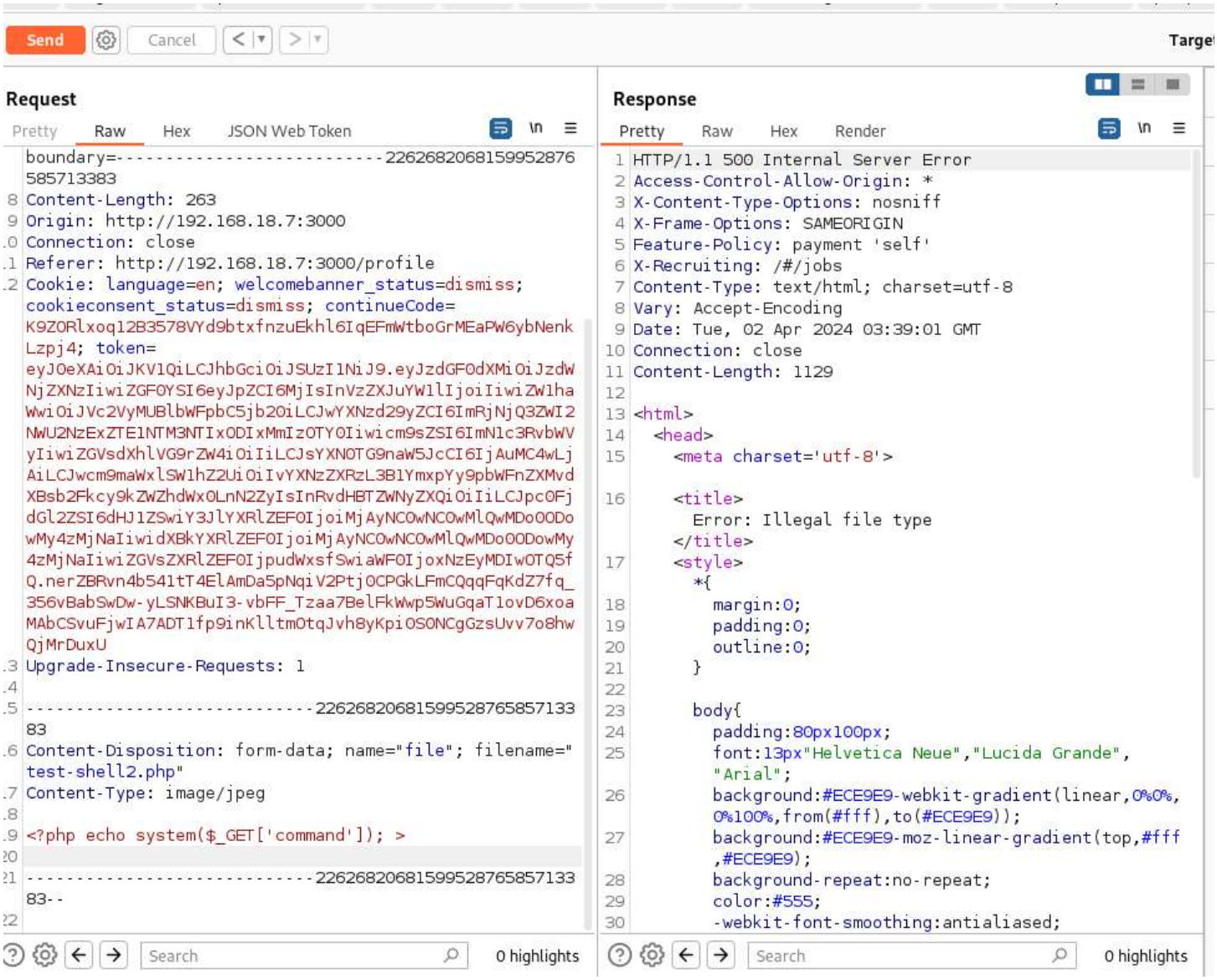
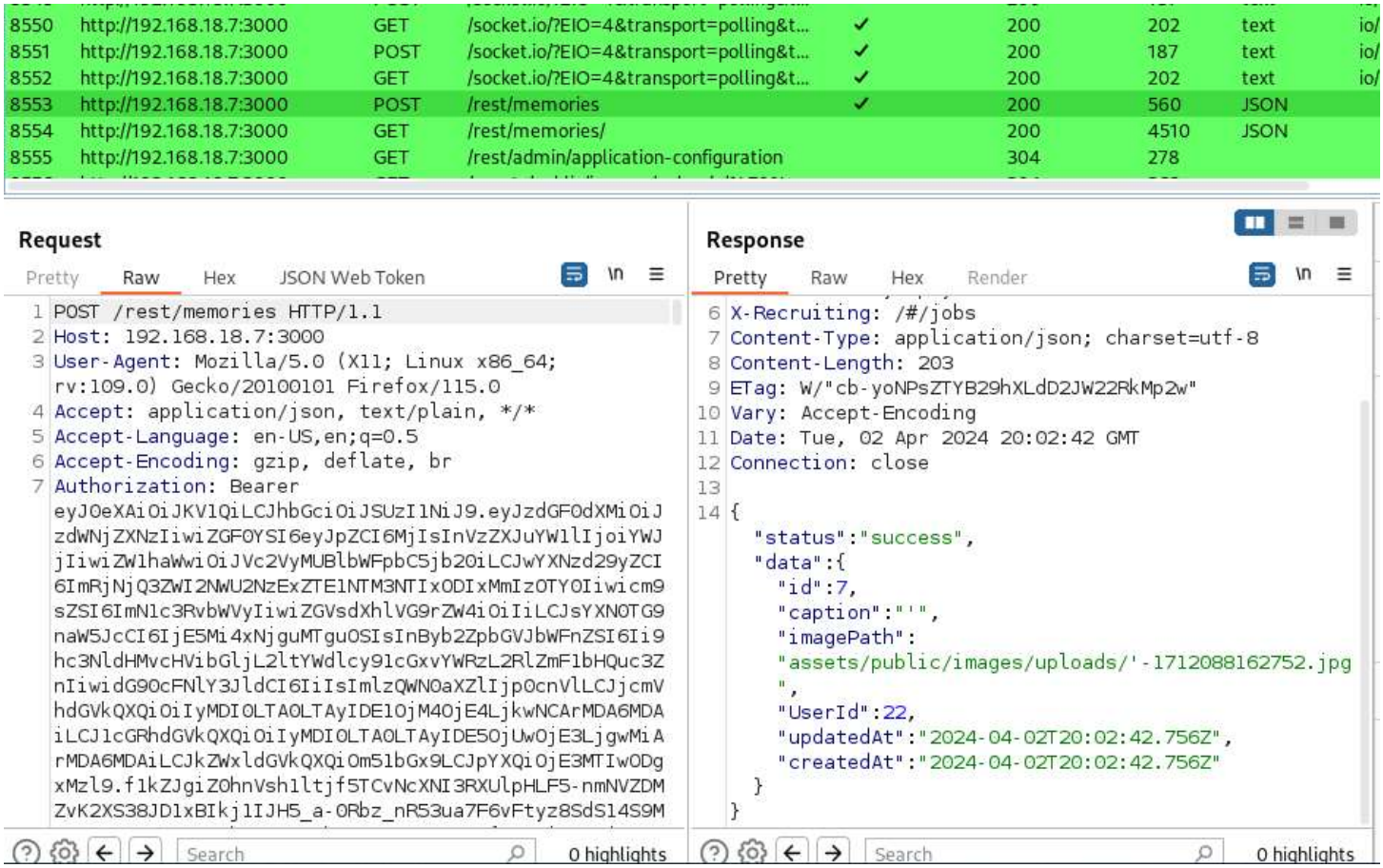


Photo-Wall

Submitting an image to the photo-wall includes a caption and an image in a POST request. A successful upload generates a 200 response.



OS Command Injection

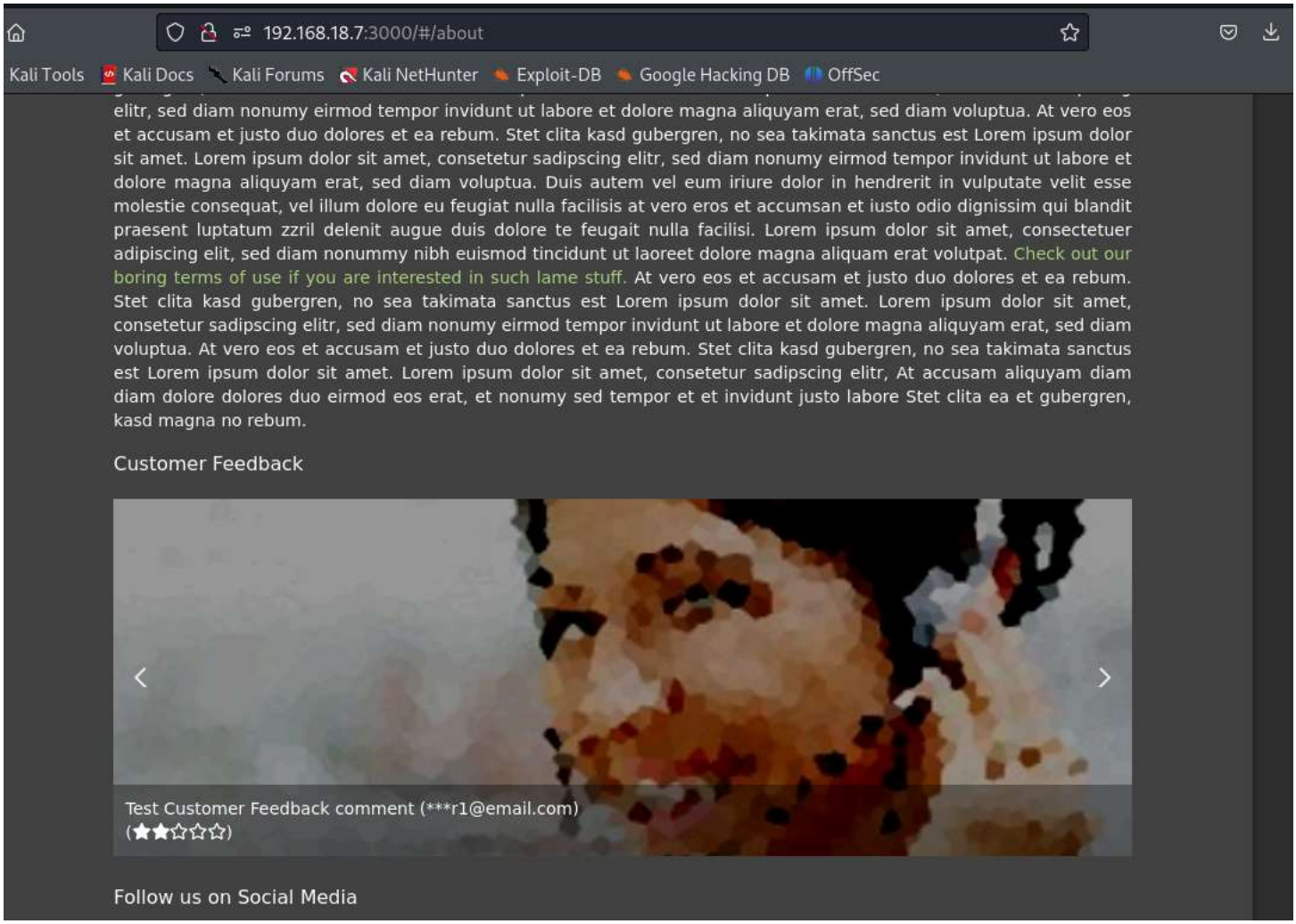
The application was reviewed for potential OS Command Injection where parameters are passed to the web server in an HTTP request. No such locations were found.

No issue:	No OS Command Injection vulnerabilities were found.

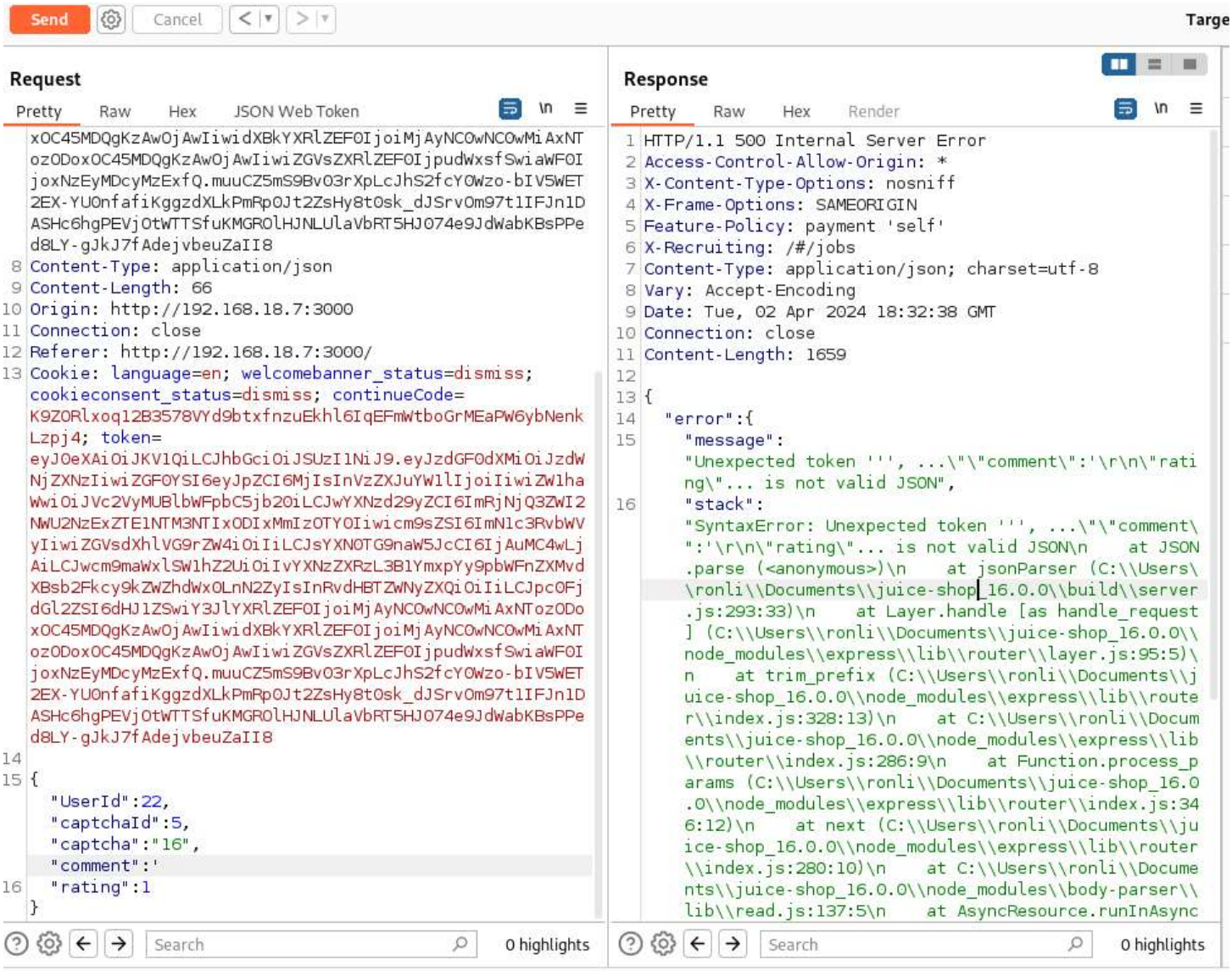
SQL Injection and XSS

Customer Feeback

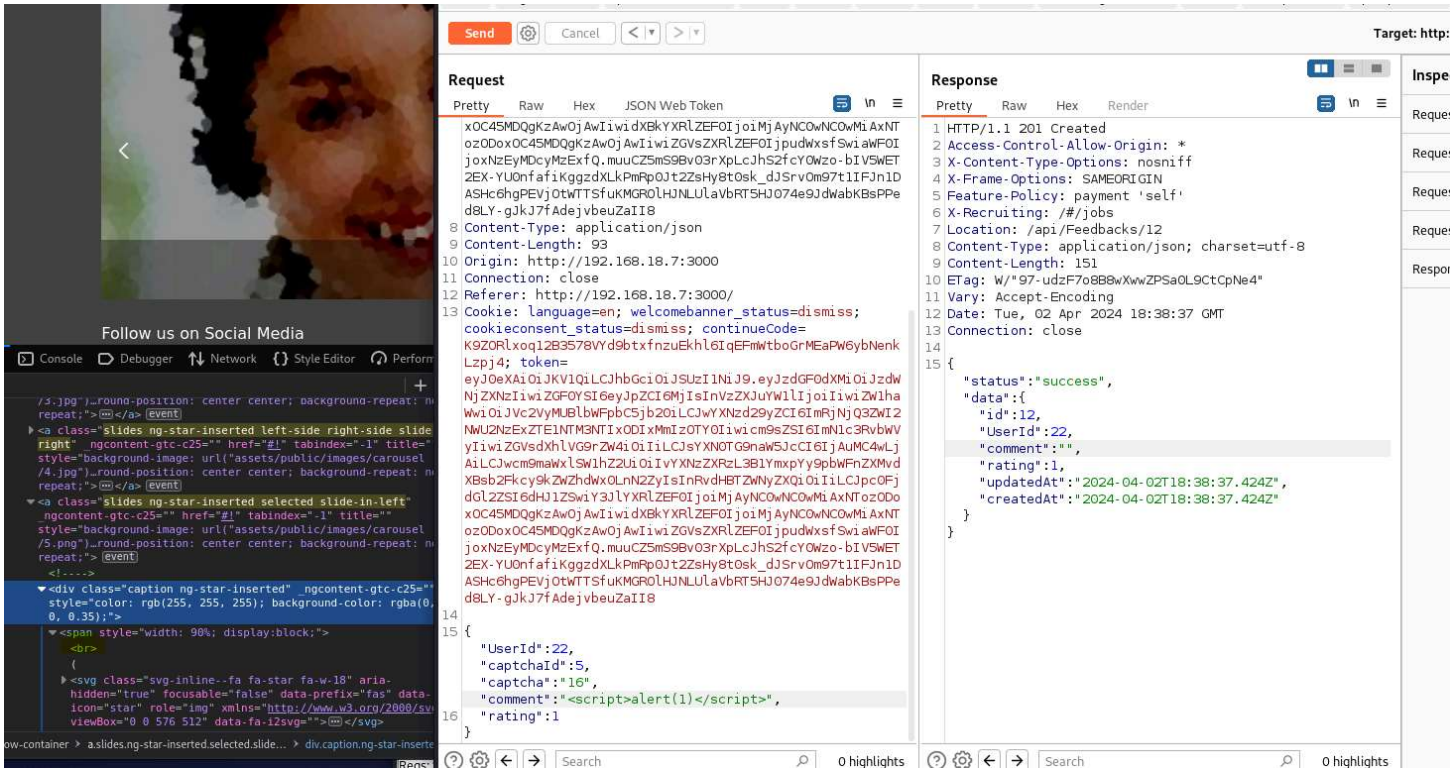
The Comment in the Customer Feedback page is posted directly in the /about directory. A test comment “Test Customer Feedback” was written and we can see it displayed as such.



Sending a single quote in the comment generates an error.

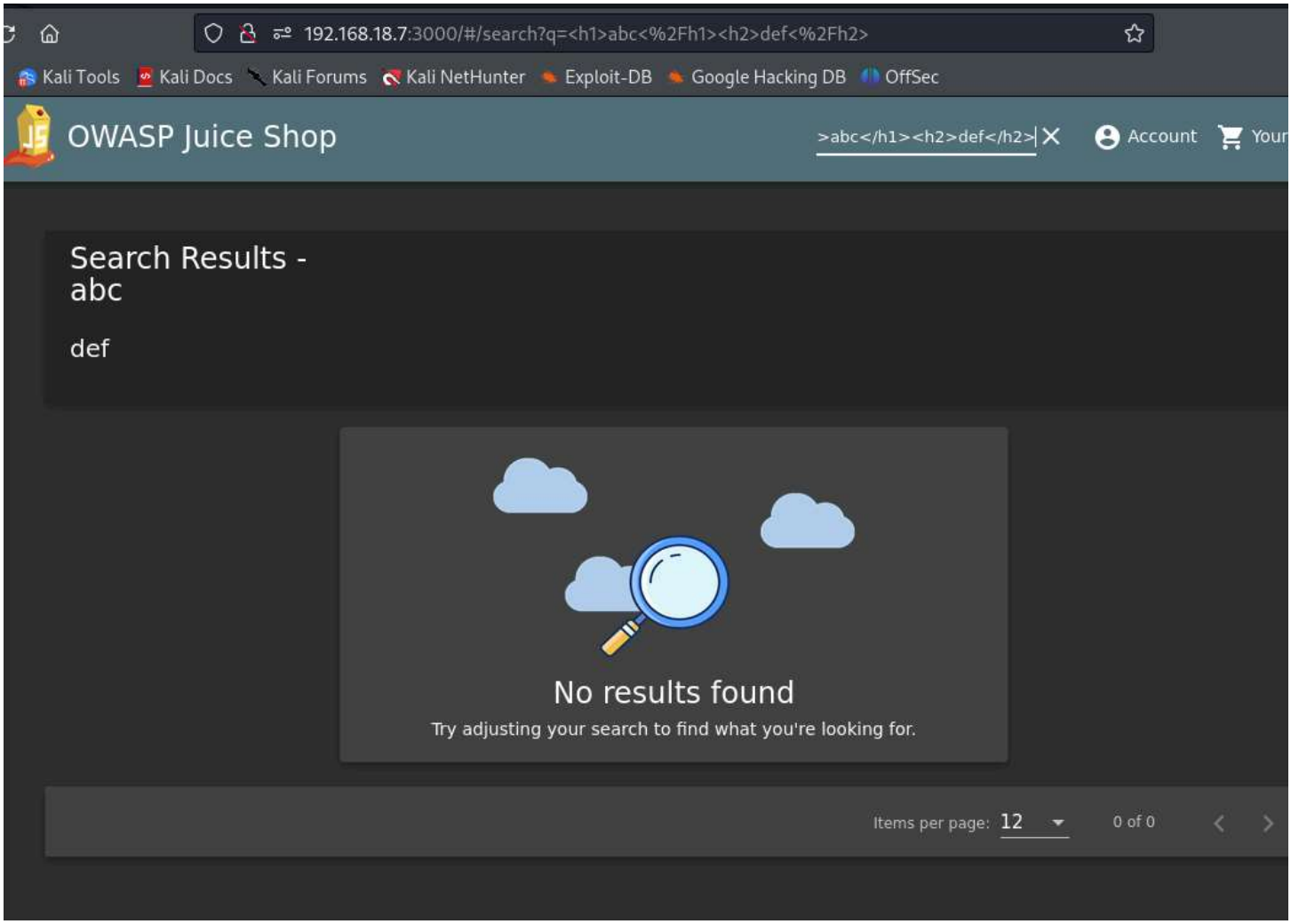


It was possible to enter JavaScript tags however the data from the submitted comment seems to have disappeared when viewing the page, and the text also disappeared in the client-side HTML.



Search Bar

Entering a single quote in the Search Bar did not generate any errors.
HTML tags can be entered in the search bar, however on initial tests it was not possible to enter JavaScript.

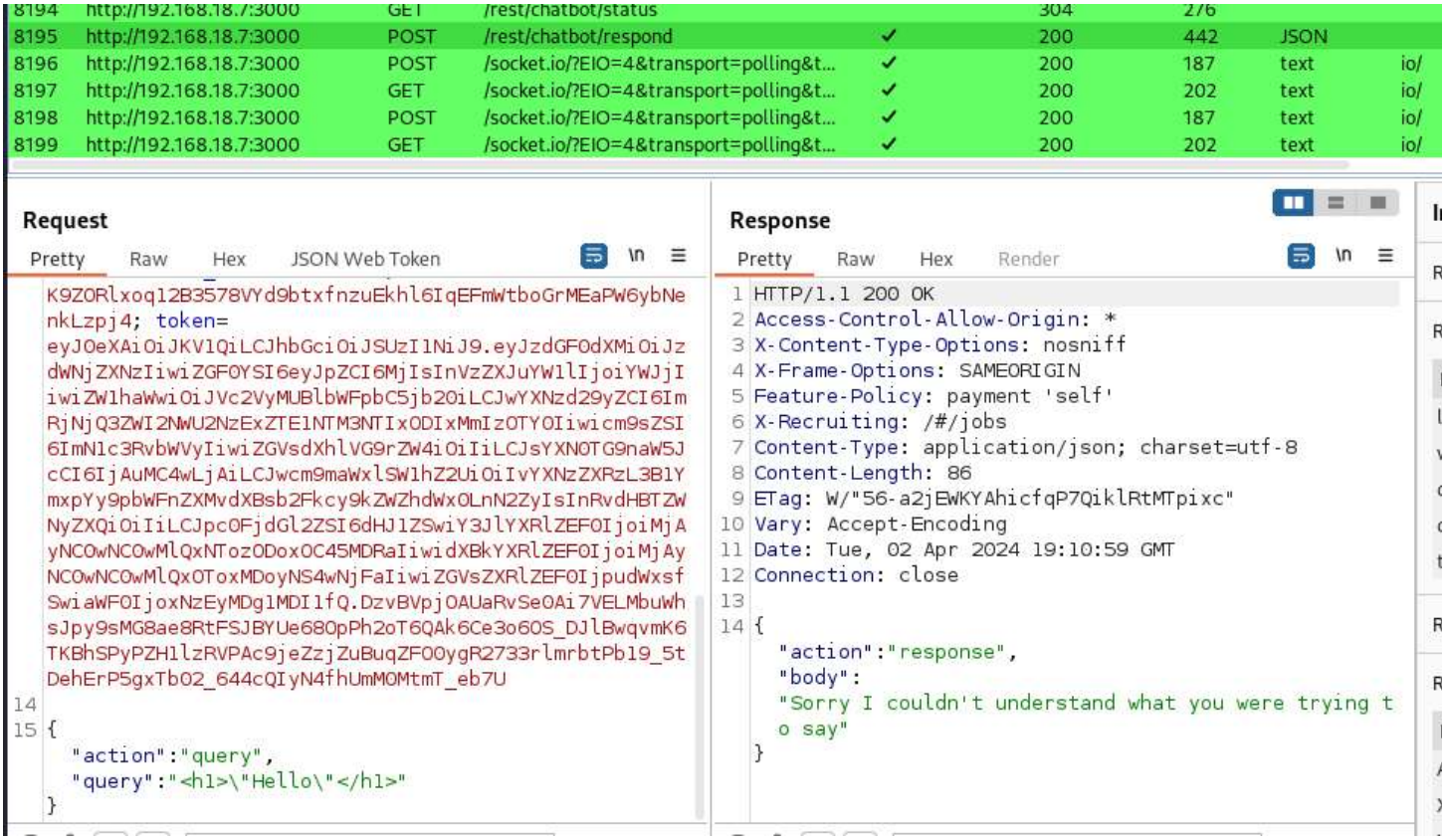


Complaint

The Complaint message is not reflected in other areas of the webpage. Entering a single quote and JavaScript tags seem to generate no errors.

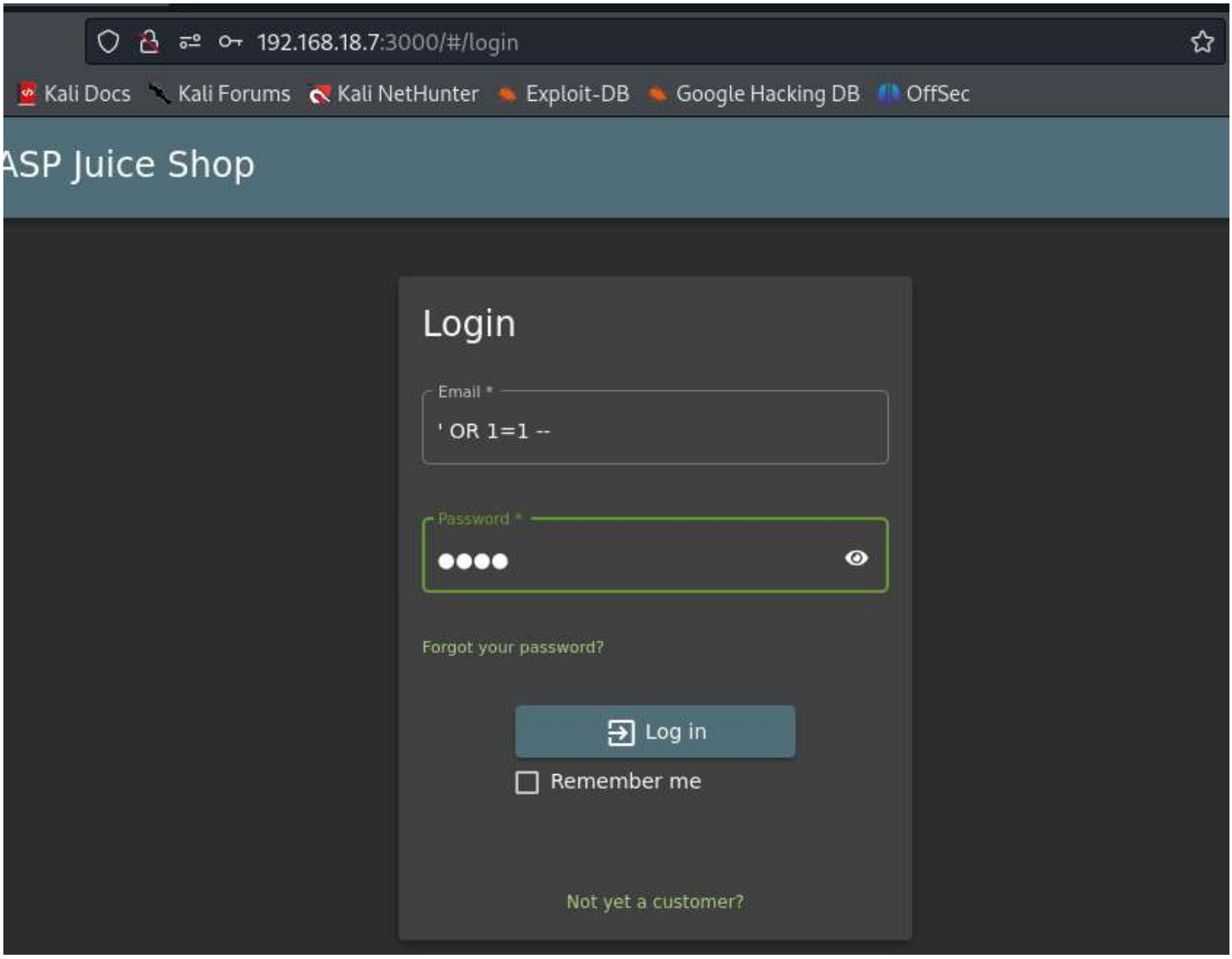
Chat Bot

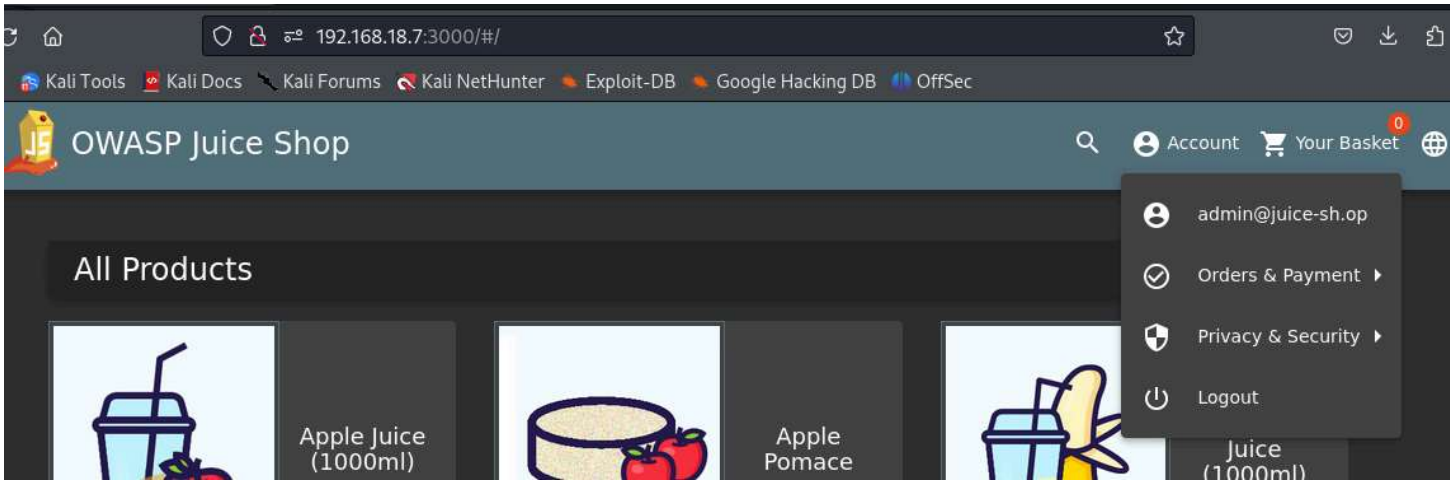
Messages to the chat bot are sent in a POST request as JSON data. Entering a single quote and JavaScript tags seem to generate no errors.



Login

The login page is susceptible to SQL injection. Entering “‘ OR 1=1 – “ in the email field and any text in the password field allows logging in as the administrator without knowing the password.

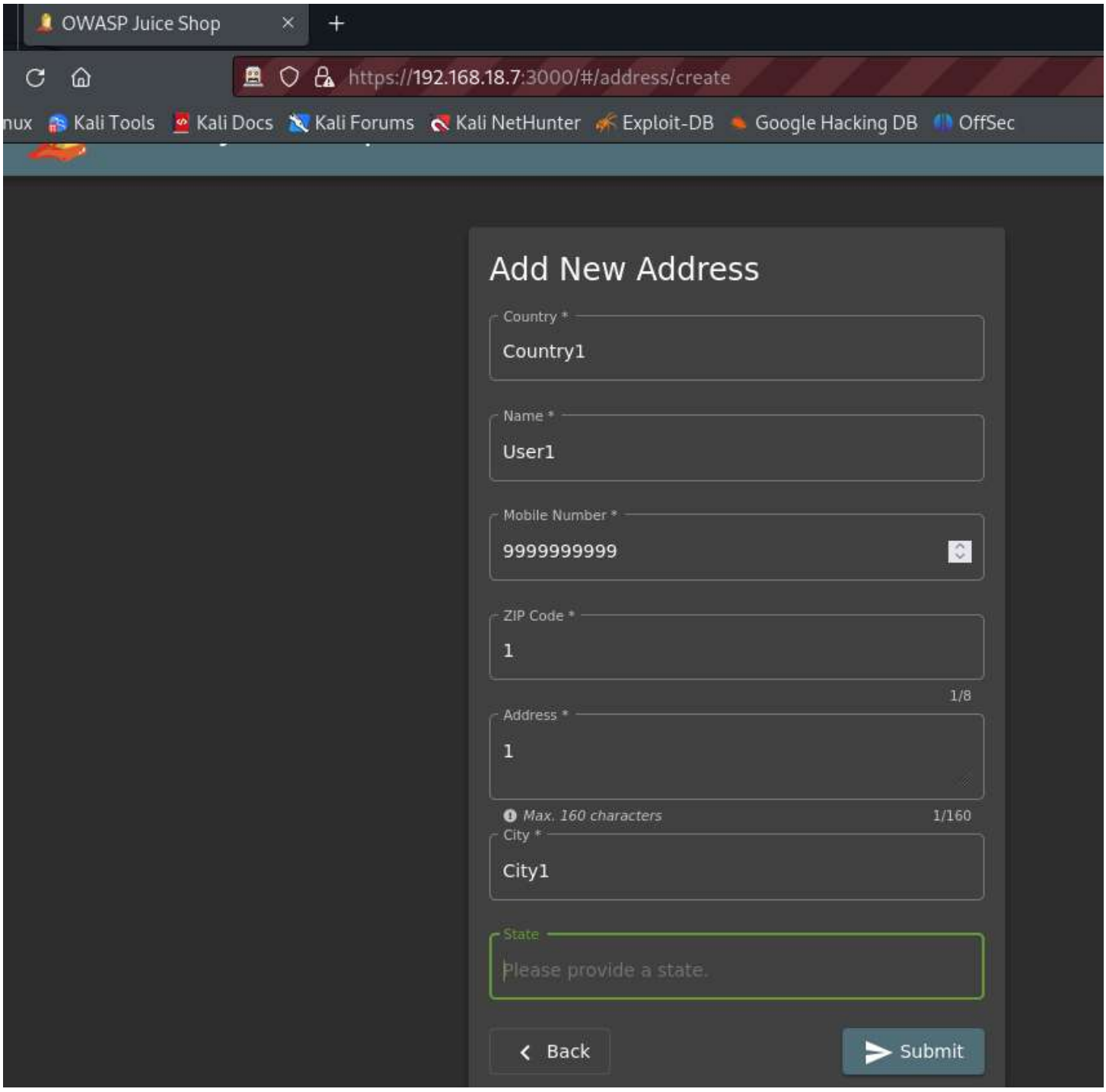




Finding	Severity	Description
	Critical	SQL injection in the Login allows attackers to login as administrator. Parameterized queries should be used to prevent SQL injection.

Input Validation in Other Areas of the Web Application

An address was created with the following information



Misc