Ethical Hacking and Penetration Testing

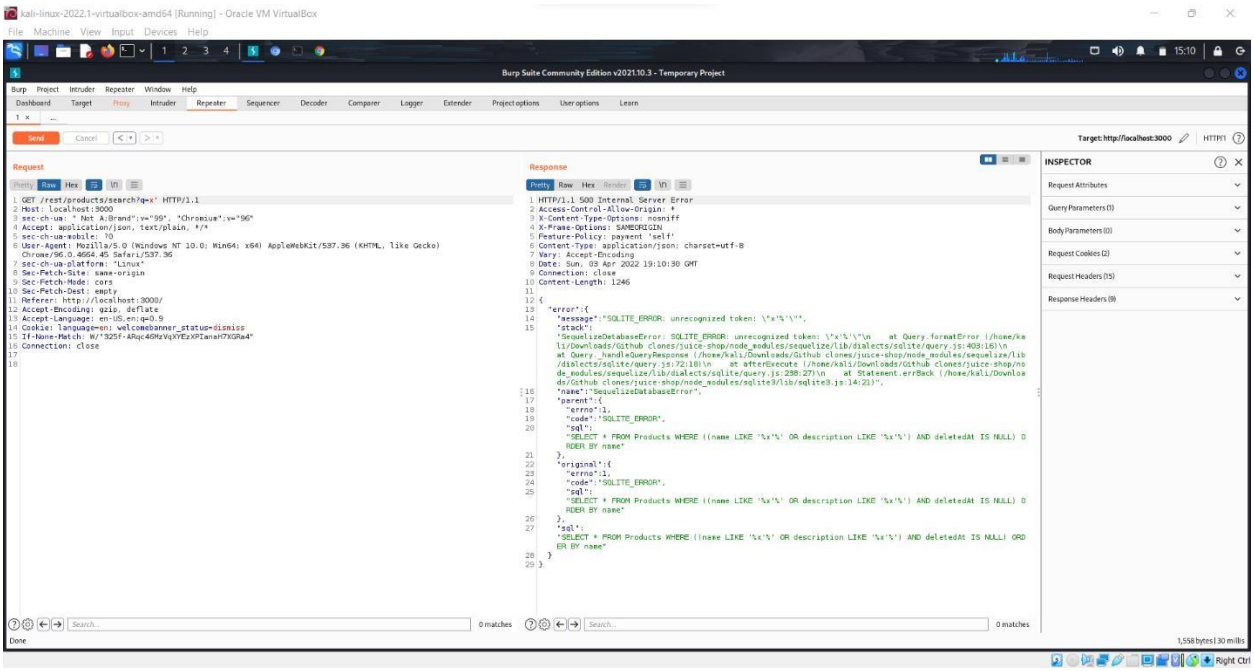
Assignment 3

Web Exploitation (Juice-Shop)

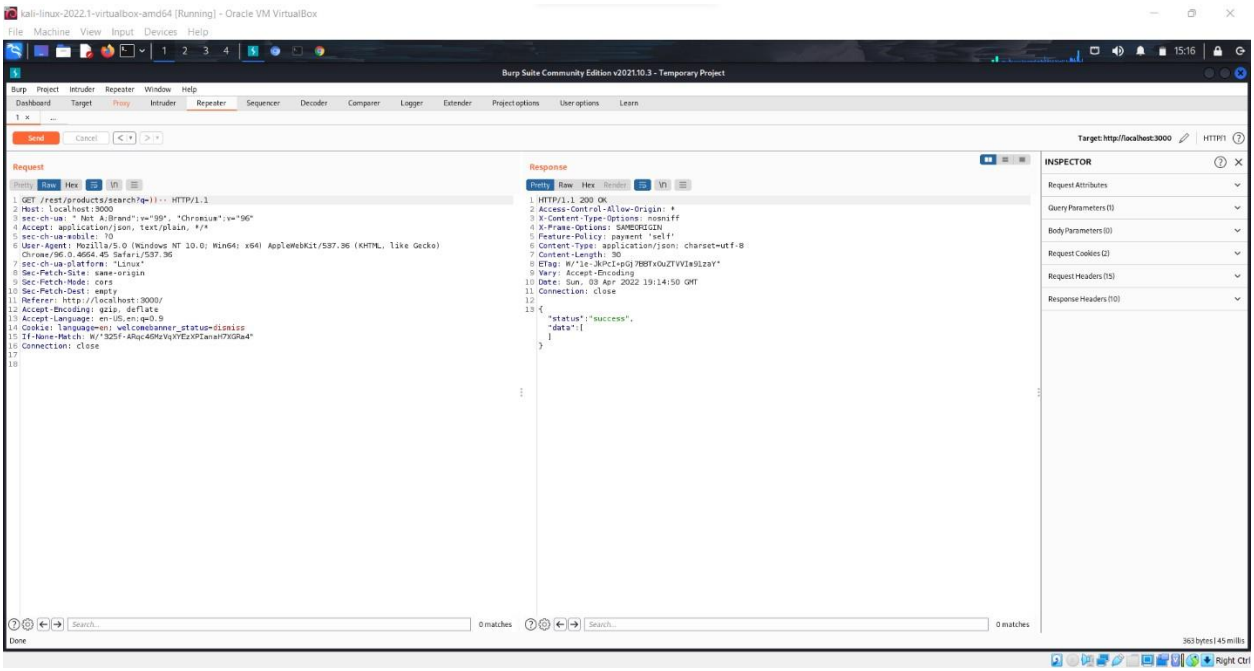
Name: Karim Salem

Attack number 1

1- First of all, I opened owasp juice shop website and started navigating while intercepting the packets on burpsuite to find an interesting request that I can send to the repeater and notice its responses searching for a request that has an SQL injection vulnerability, I found that the search GET req input is not filtered and it interacts with the database retrieving the queries and the errors that SQLITE produces when an unexpected input is written



2- Discovered that it’s vulnerable to SQL injection after I wrote a query to close the name and comment the rest of the query and it succeeded



3- After browsing for a query in SQLITE to find the DB Schma I found this

Getting the structure of a table using the SQL statement

You can find the structure of a table by querying it from the `sqlite_schema` table as follows:

```
SELECT sql
FROM sqlite_schema
```

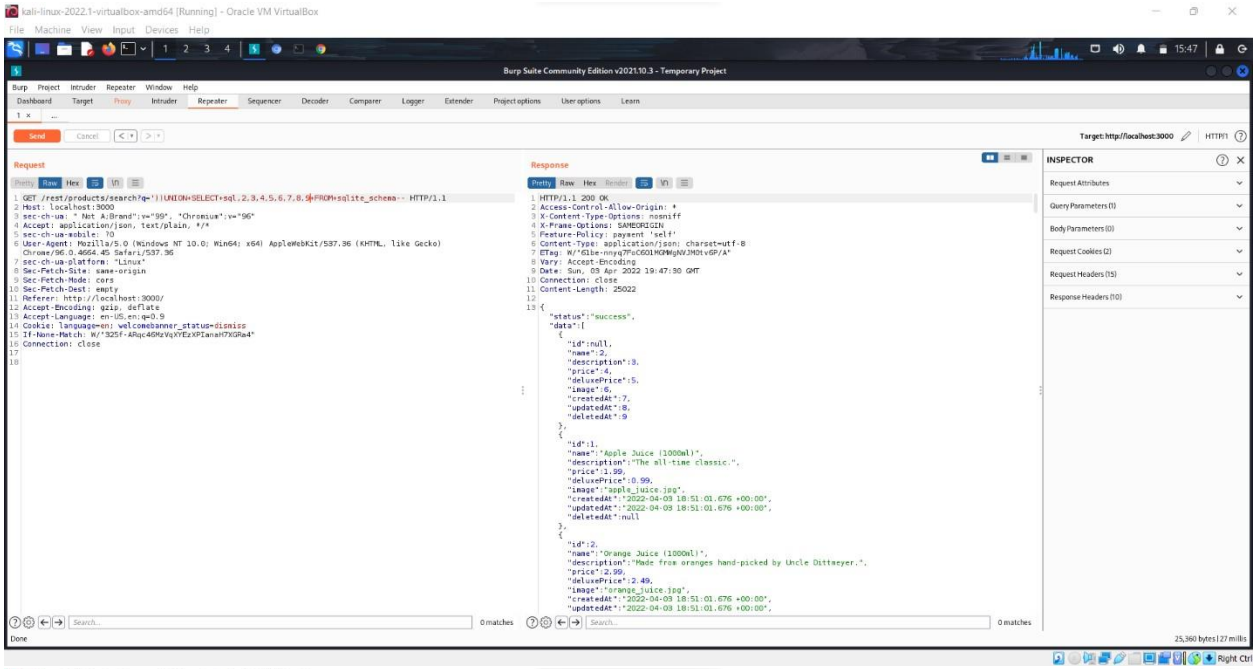
4- I used the union operator to combine queries on burpsuite and combined the above mentioned query, but I found an error because the result columns on the left and the right don't match

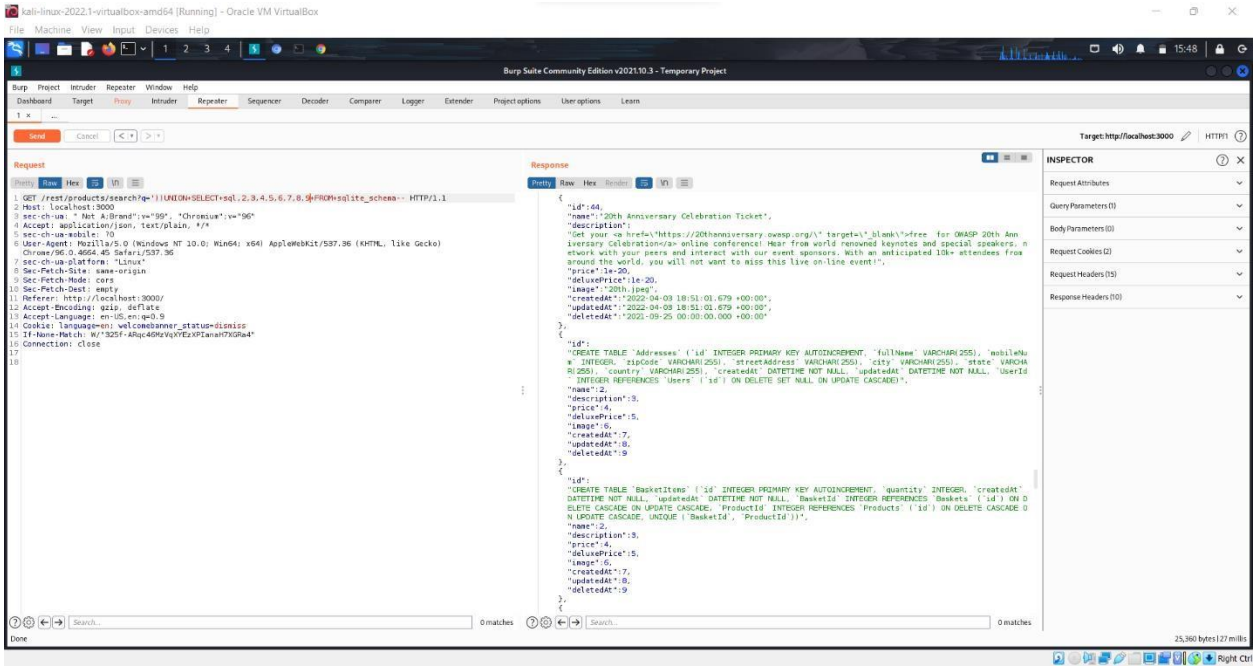
Getting the structure of a table using the SQL statement

You can find the structure of a table by querying it from the `sqlite_schema` table as follows:

```
SELECT sql
FROM sqlite_schema
```

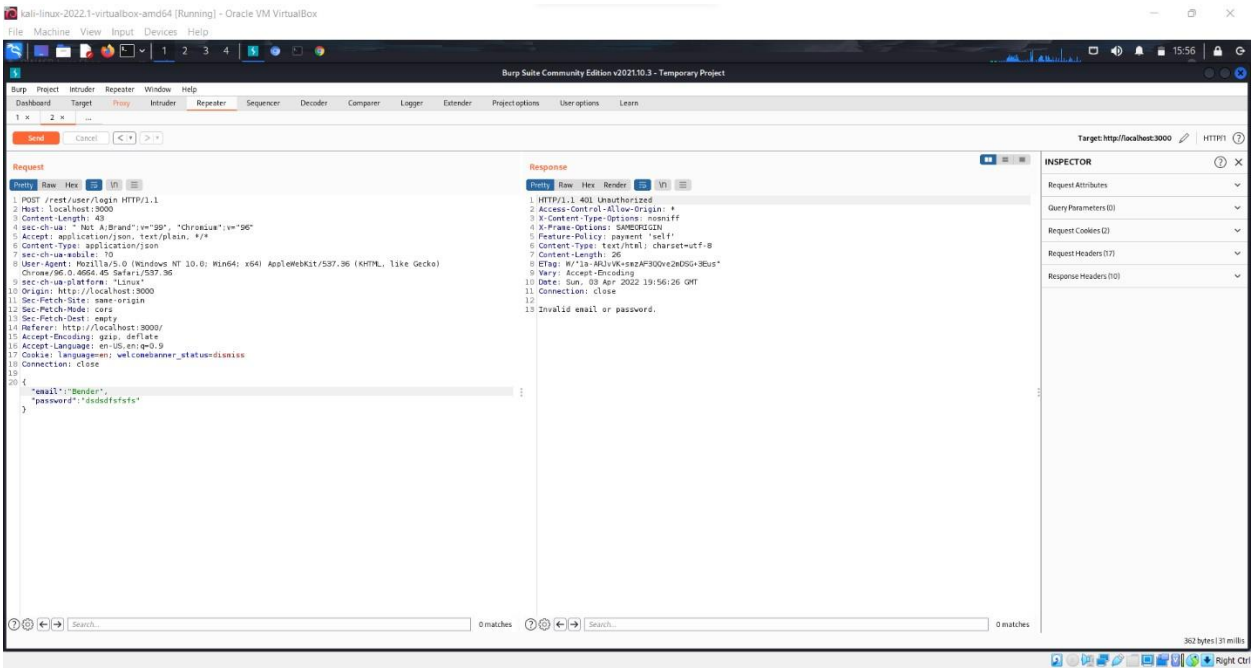
5- I solved the error just by trying and incrementing the number of the columns till I succedded and retrieved the entire DB schema of the site


kali-linux-2022.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
Burp Suite Community Edition v2021.10.3 - Temporary Project
Request
GET /rest/products/search?q='))UNION+SELECT+sql,2,3,4,5,6,7,8,9+FROM+sqlite_schema-- HTTP/1.1
Response
HTTP/1.1 200 OK
INSPECTOR

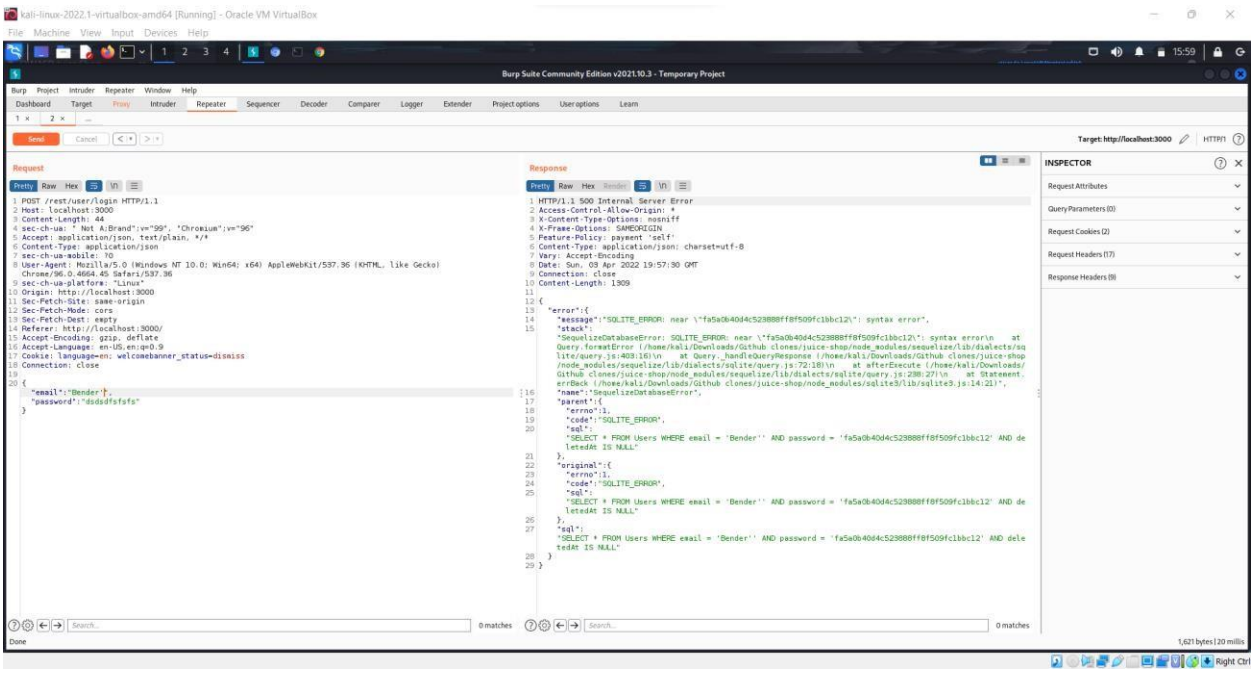

kali-linux-2022.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
Burp Suite Community Edition v2021.10.3 - Temporary Project
Request
GET /rest/products/search?q='))UNION+SELECT+sql,2,3,4,5,6,7,8,9+FROM+sqlite_schema-- HTTP/1.1
Response
INSPECTOR

Attack number 2

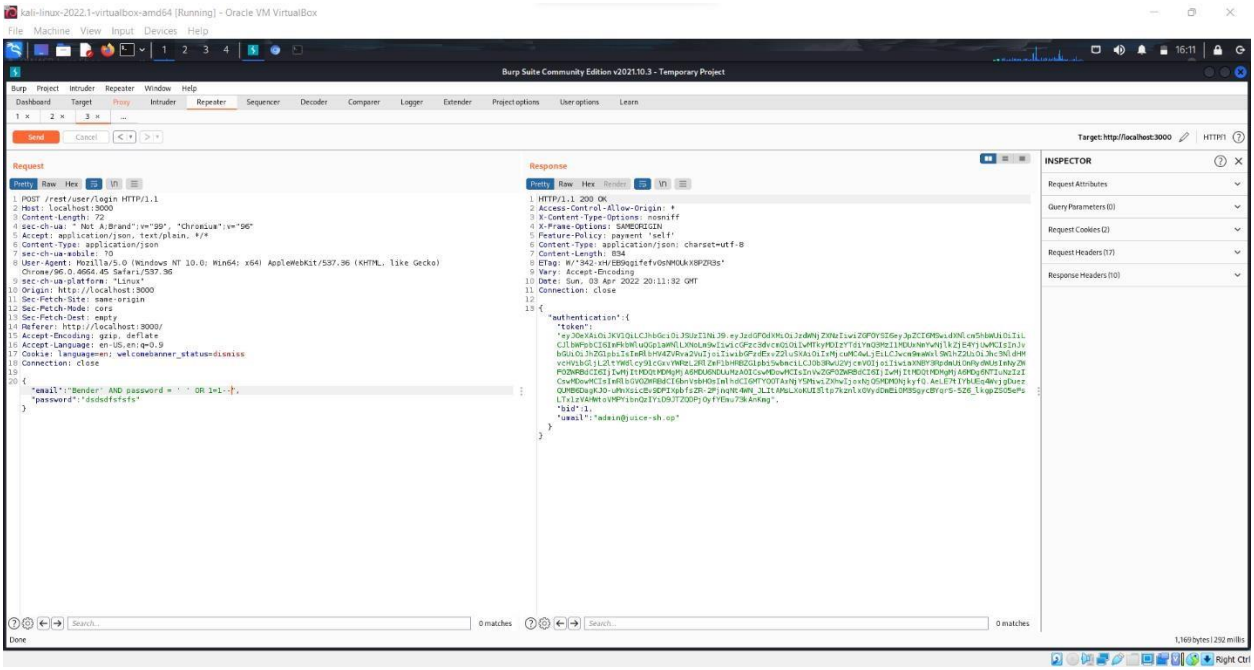
1- I opened the login webpage and typed username Bender and any random password, then I forwarded the packets from burpsuite proxy till I found a req that holds my input and sent it to the repeater



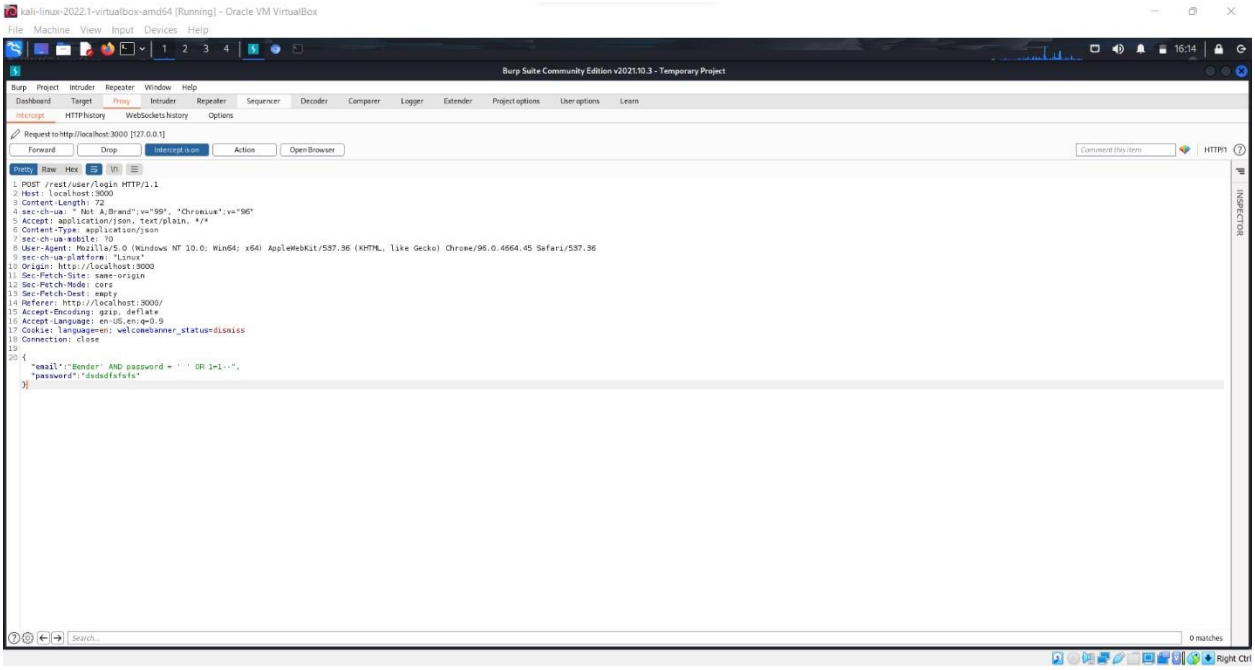
2- Added extra ‘ after the username to know the syntax of the query

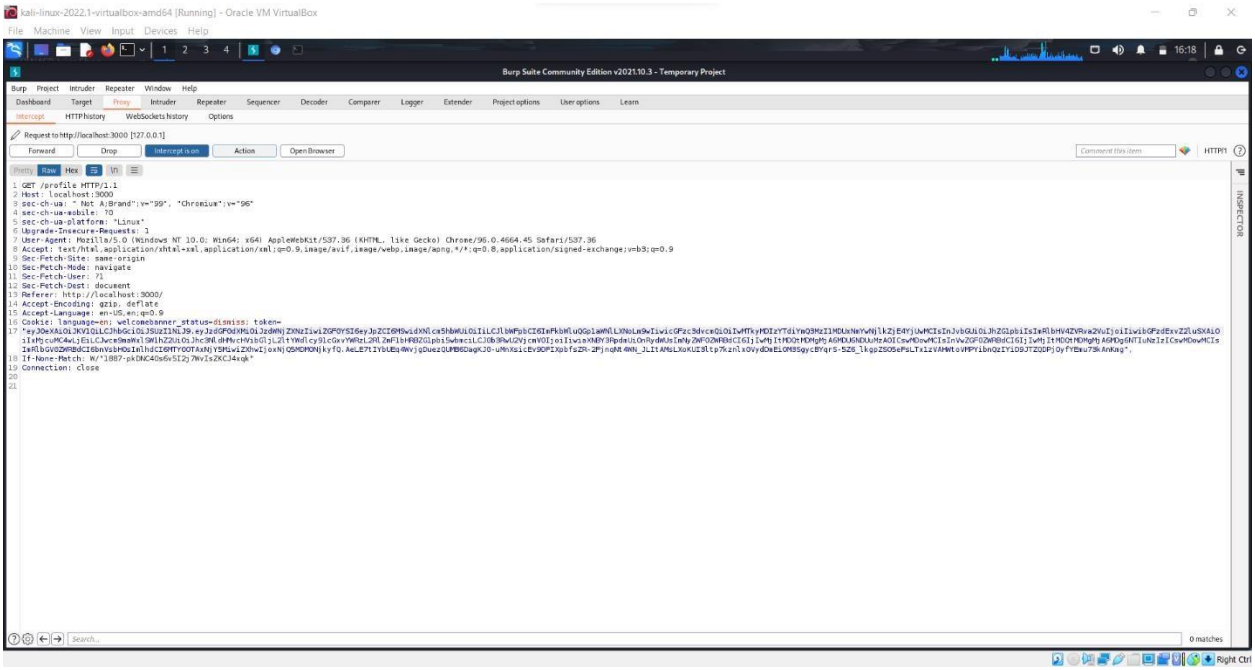


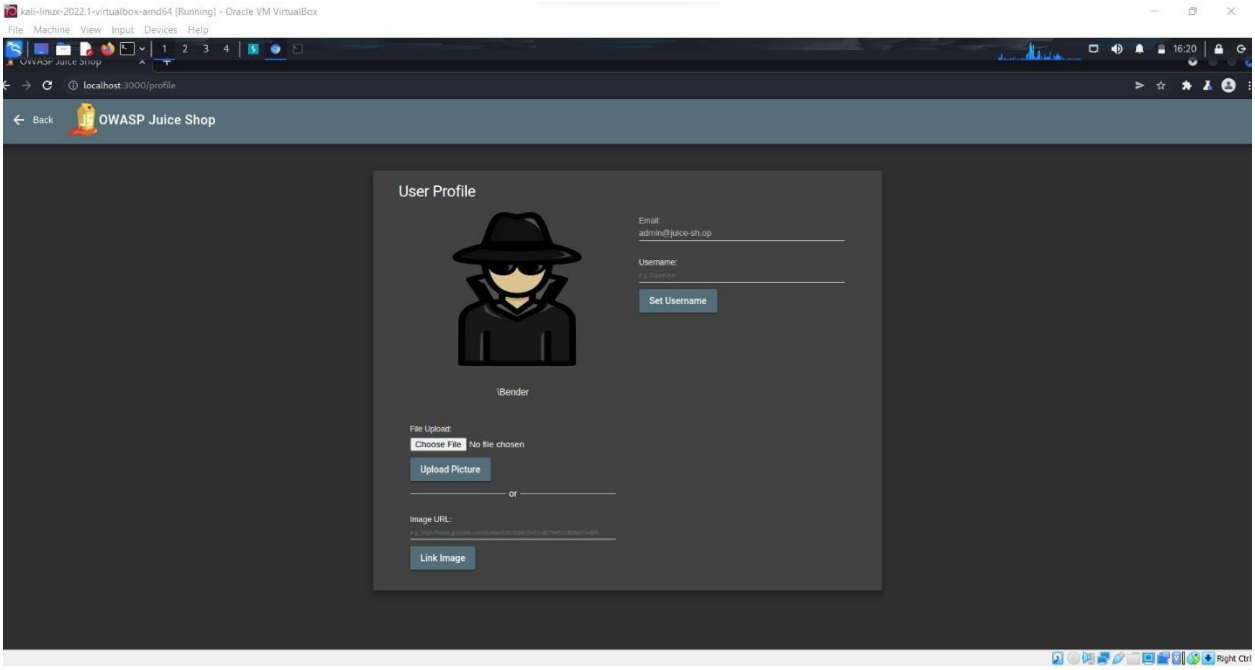
3- Manipulated my input to bypass that query and successfully logging in



4- I copied the req from the repeater to the proxy and started forwarding packets till I reached a req that requires a token, I copied the token from the repeater res and continued forwarding till it's done and I successfully logged in

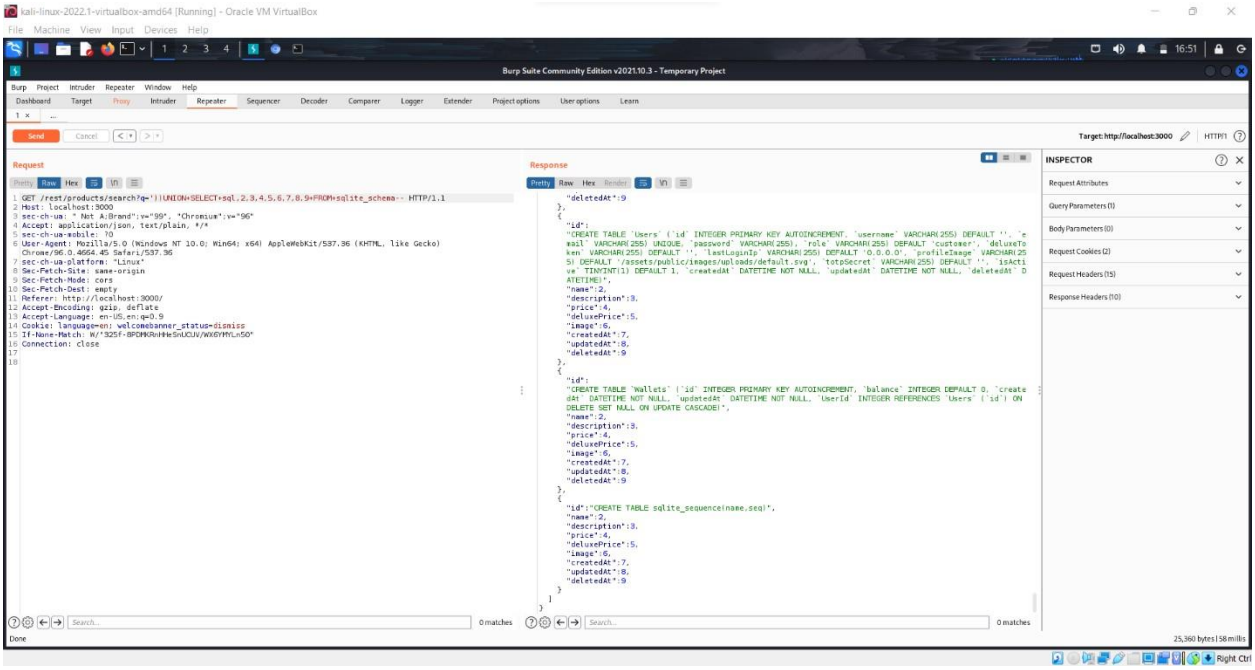

kali-linux-2022.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Burp Suite Community Edition v2021.10.3 - Temporary Project
Request to http://localhost:3000 [127.0.0.1]
Forward
Drop
Intercept is on
Action
Open Browser
POST /rest/user/login HTTP/1.1
Host: localhost:3000
Content-Length: 72
sec-ch-ua: " Not A;Brand";v="99", "Chromium";v="96"
Accept: application/json, text/plain, */*
Content-Type: application/json
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/96.0.4664.45 Safari/537.36
sec-ch-ua-platform: "Linux"
Origin: http://localhost:3000
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:3000/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: language=en; welcomebanner_status=dismiss
Connection: close
{
"email":"Bender' AND password = ' ' OR 1=1--",
"password":"dsdsdfsfsfs"
}
0 matches
Right Ctrl


kali-linux-2022.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Burp Suite Community Edition v2021.10.3 - Temporary Project
Request to http://localhost:3000 [127.0.0.1]
Forward
Drop
Intercept is on
Action
Open Browser
GET /profile HTTP/1.1
Host: localhost:3000
sec-ch-ua: " Not A;Brand";v="99", "Chromium";v="96"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Linux"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/96.0.4664.45 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://localhost:3000/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: language=en; welcomebanner_status=dismiss; token=
If-None-Match: W/"1887-pkDNC4Os6x5I2y7NvIs2KCl4xqk"
Connection: close
0 matches
Right Ctrl

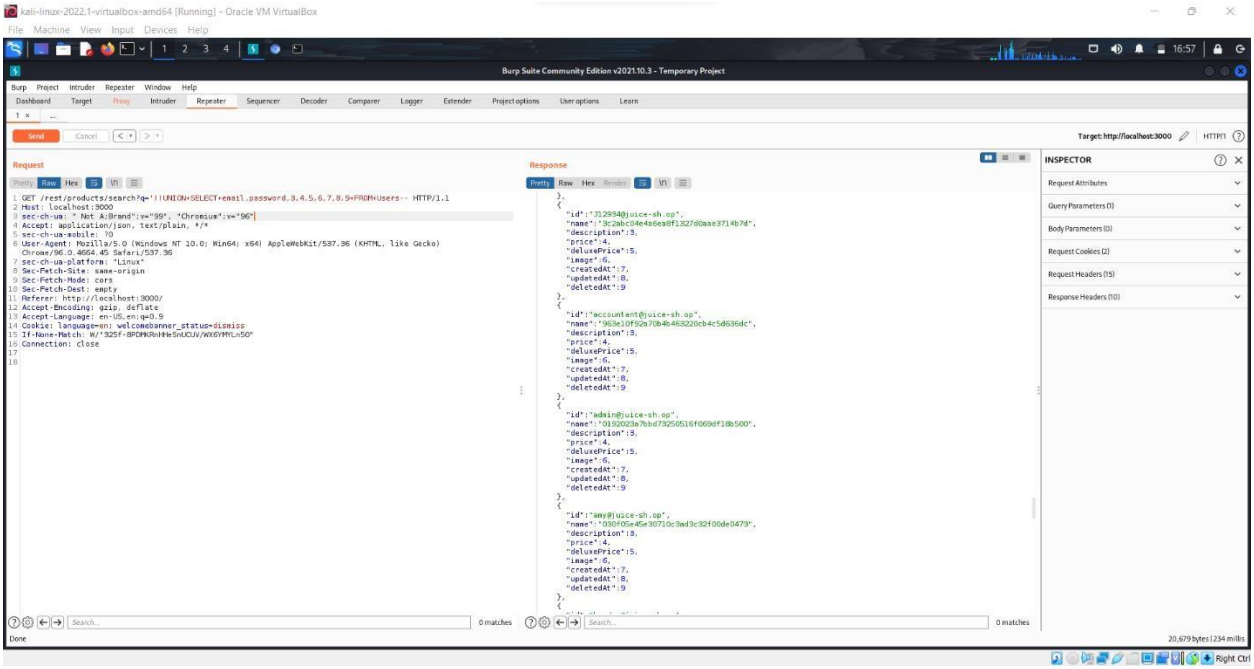


Attack number 3

1- Following attack number 1 we knew that there's a table inside my DB that has the name Users, so using the same SQL injection vulnerability we used in attack number 1 we'll try to retrieve data from this table by manipulating the input of the search GET req, first I got the Users table schema from attack number 1 to know the attributes names

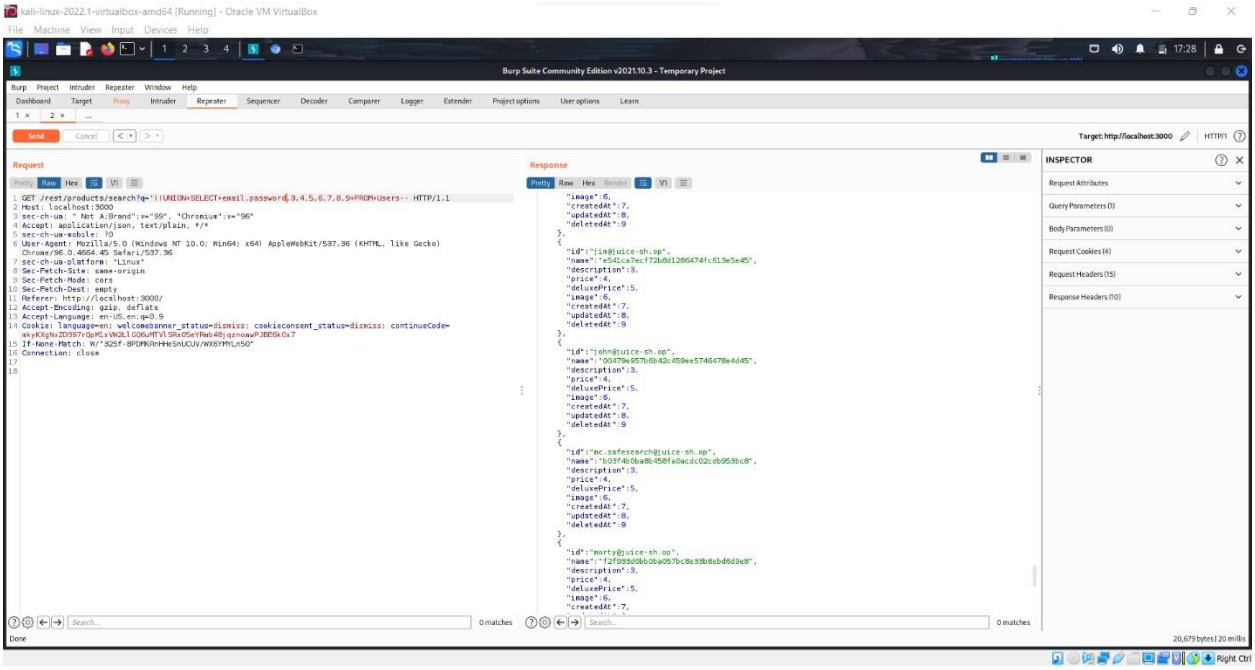


2- Then I manipulated the input to select all the emails and passwords of the users, the emails are shown in the “id”, and the hashes of the passwords are shown in the “name”, you can get the password of any specific user using any passwords hash cracker tool (e.g ntlm hash)

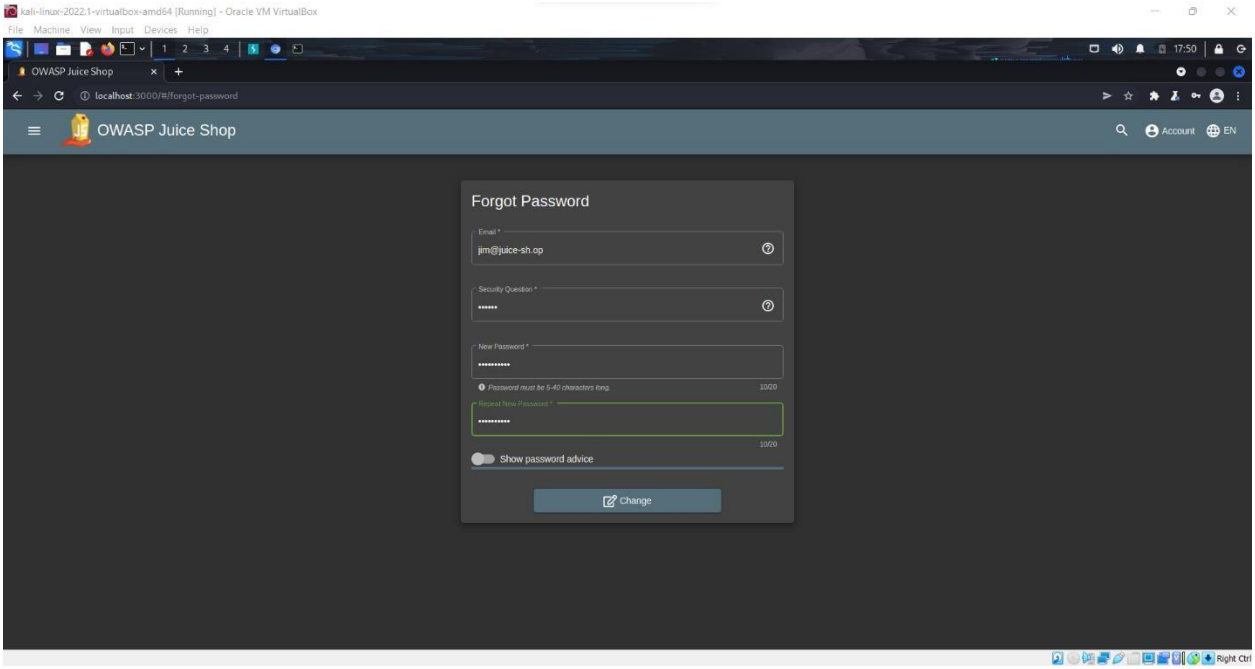


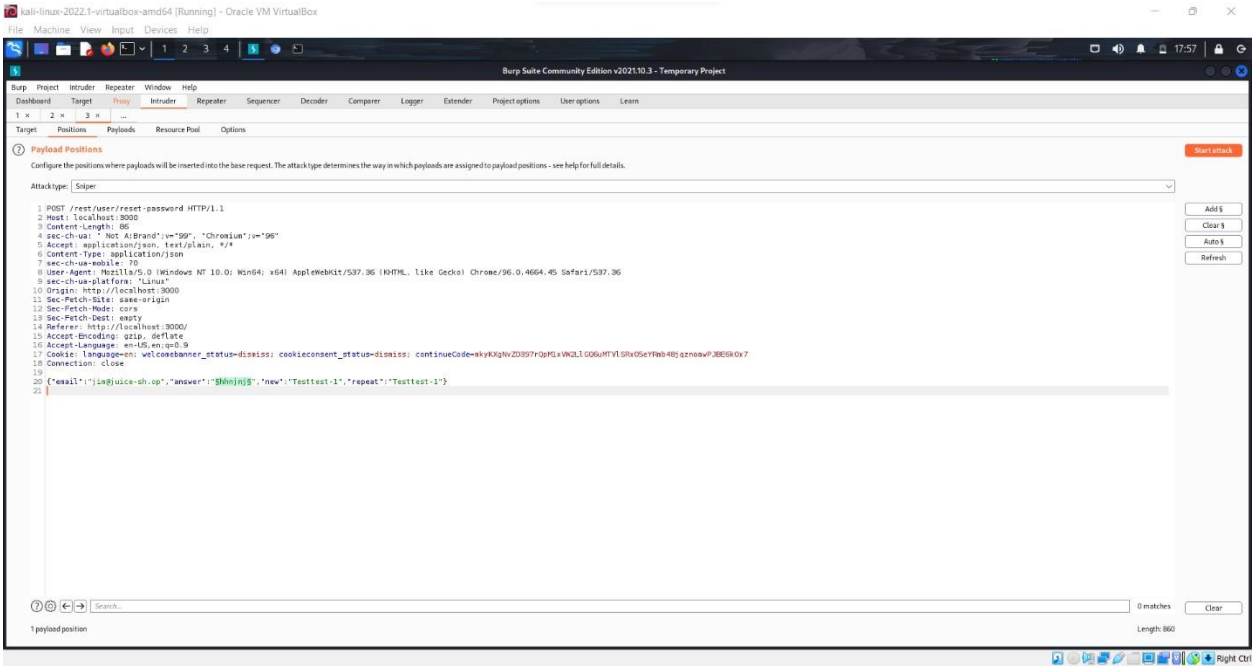
Attack number 4

1- Firstly, we get Jim’s email from the attack number 3

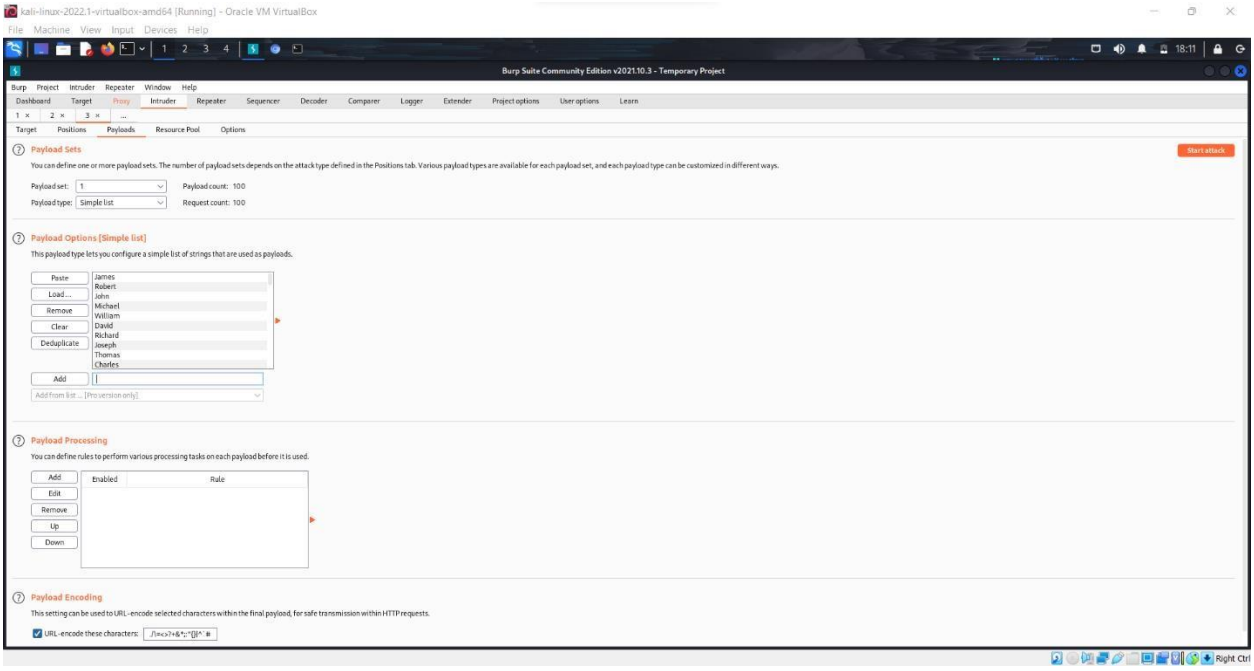


2- Then I went to the login webpage then entered Jim's email and pressed on forget password, I entered his mail and other dummy data to get the request, then I sent the request to the intruder and configured the position of the payload by adding the §

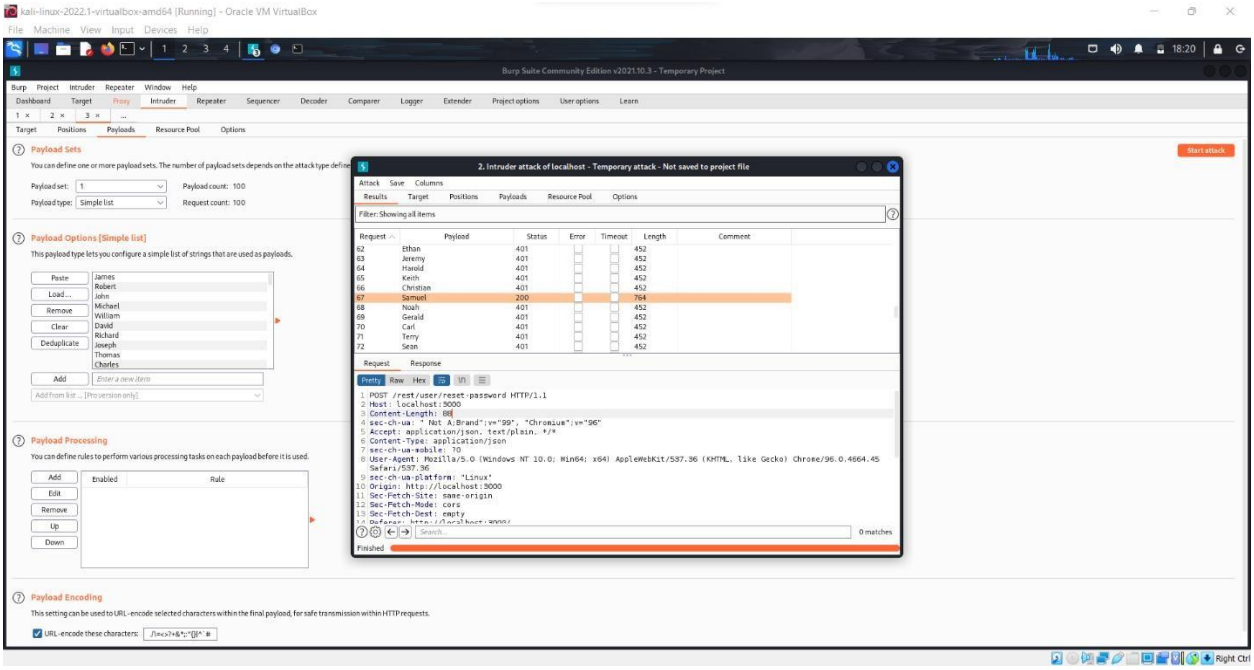




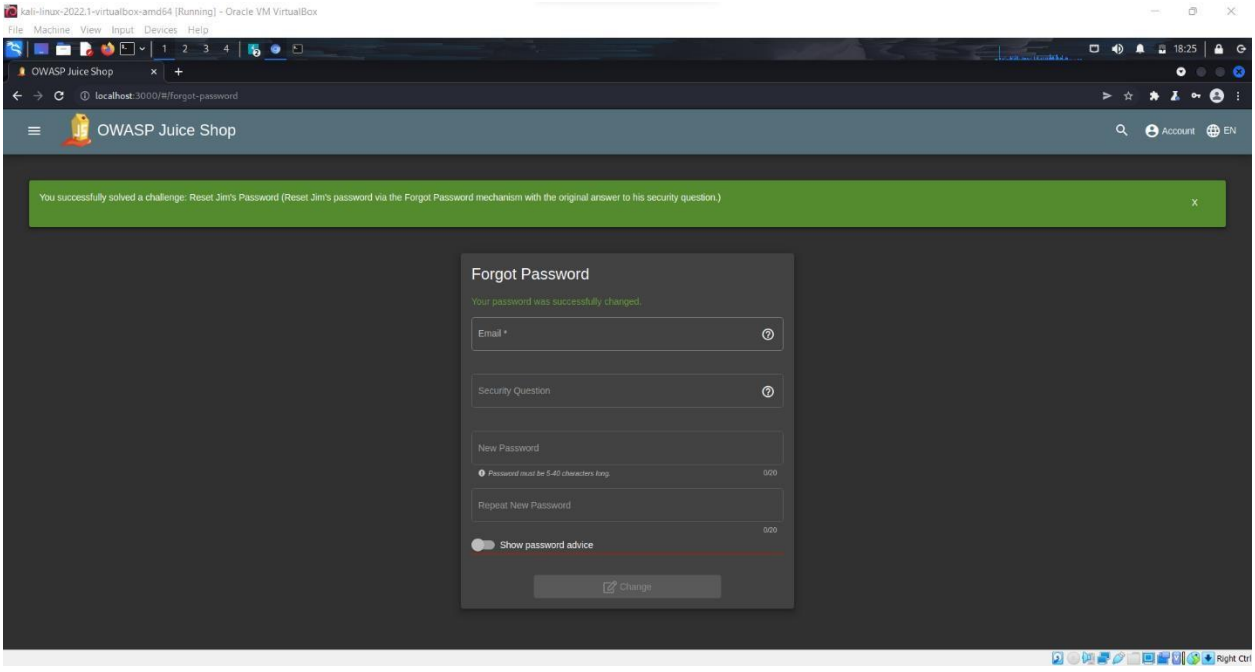
3- Then I downloaded a wordlist from google and copied it to the payload options



4- Then I searched for the name that gave a status 200, that correct name was “Samuel”

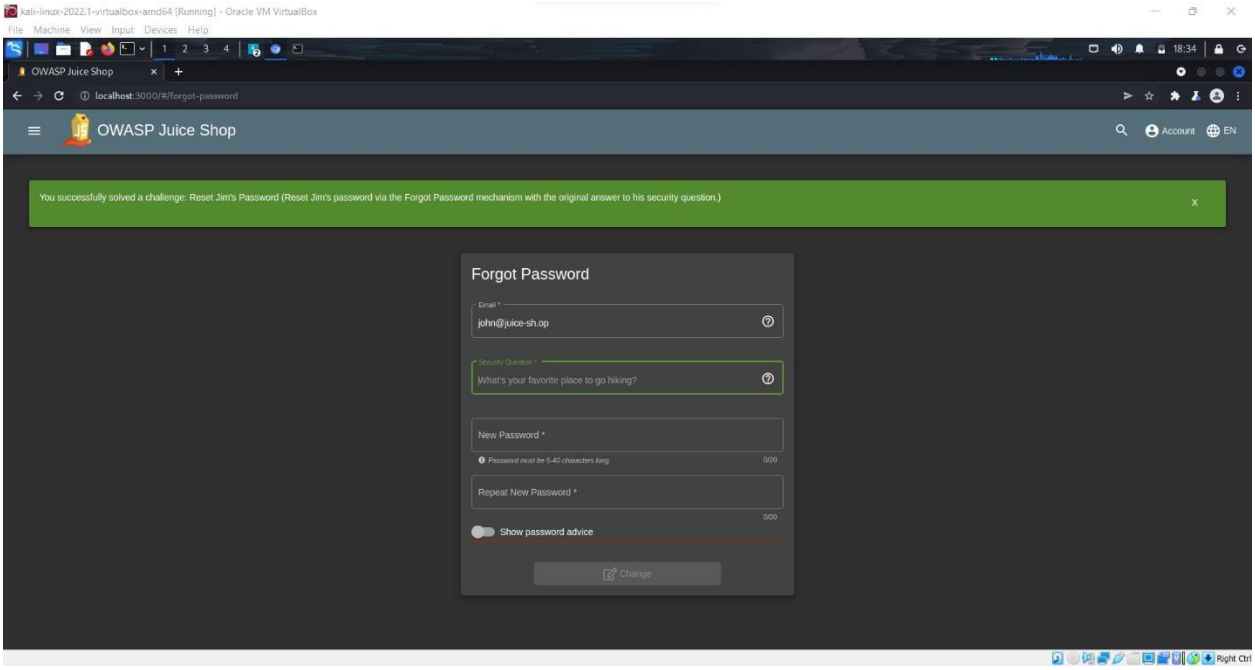


5- I went back to the forget password tab and entered the correct email and answer to the security question and was able to change the password successfully

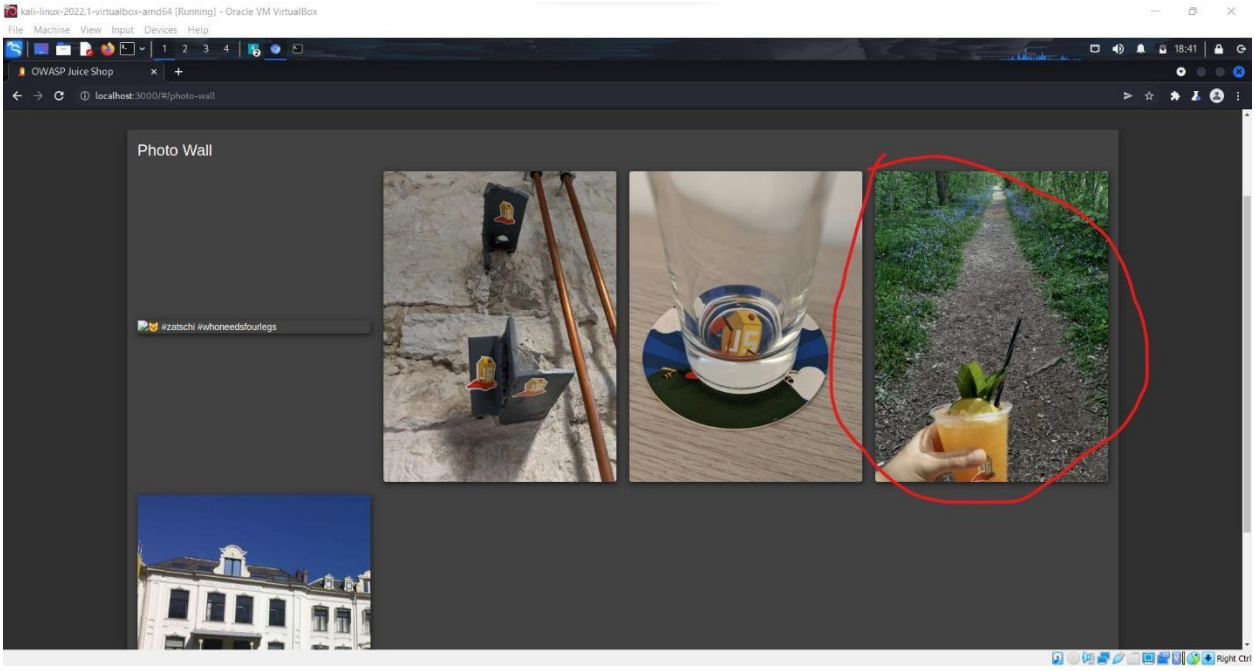


Attack number 5

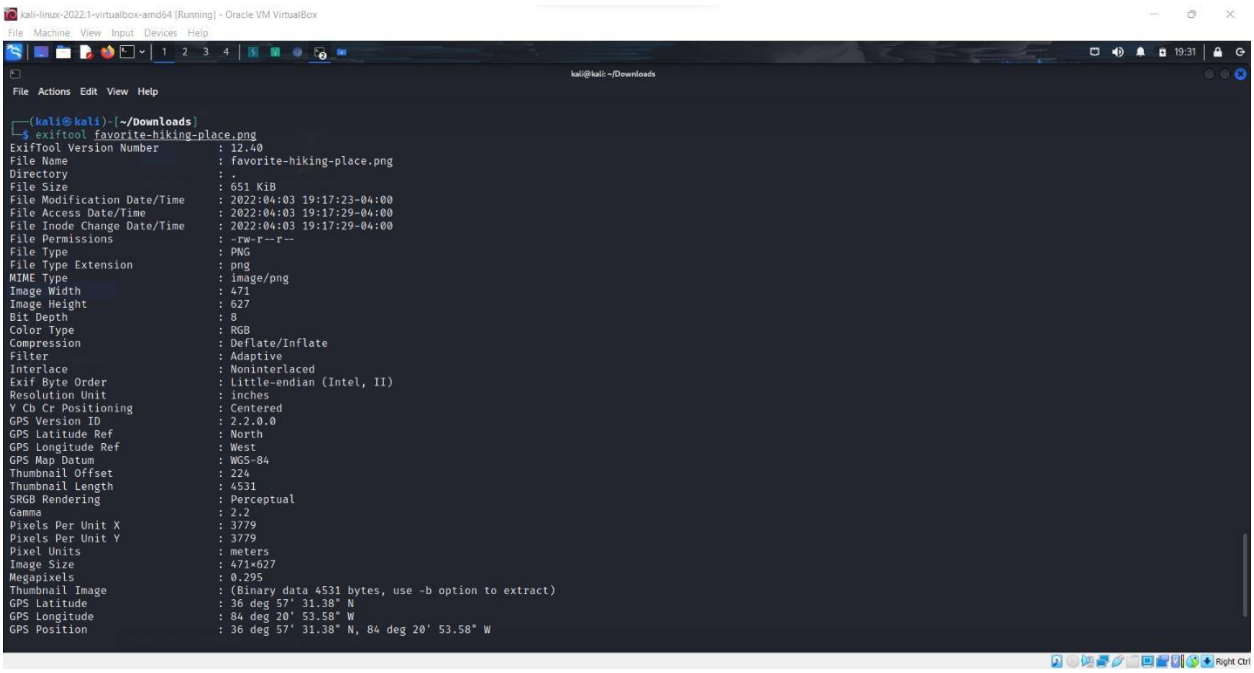
1- Same as attack number 4, I got the email of john from the attack number 3, then I went to the forget password tab to see what his security question is



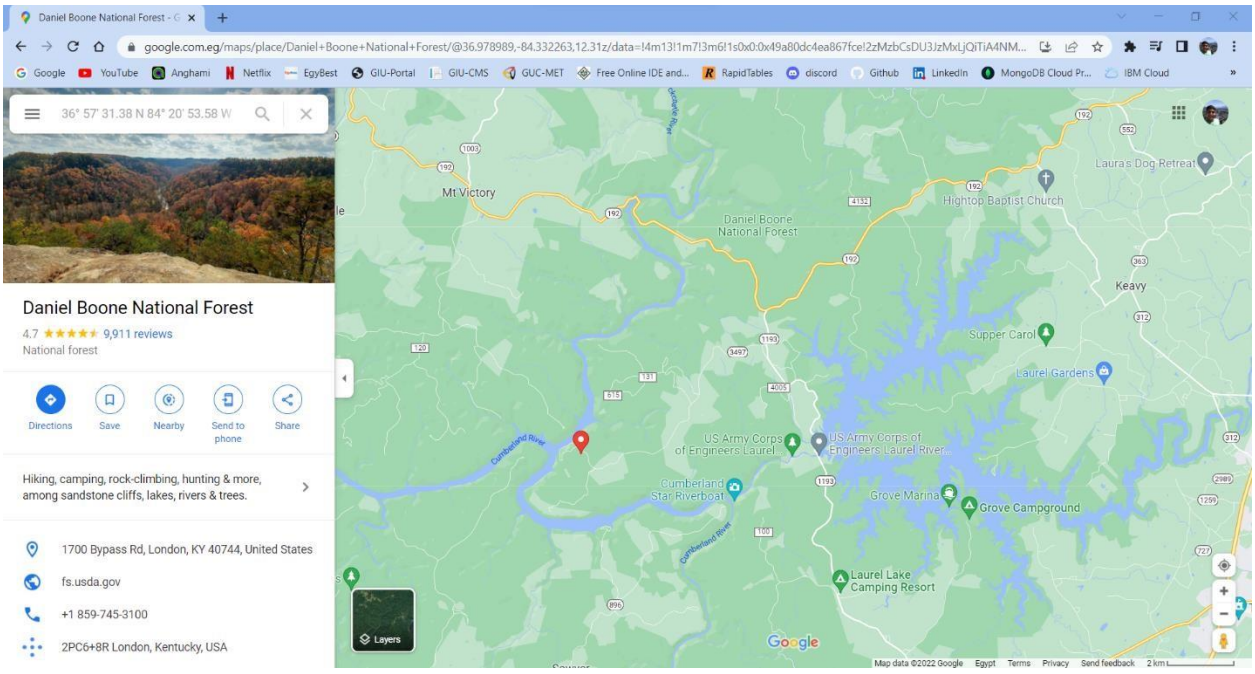
2- Then I went to the photo wall and found the photo of the tweet that is saved by John with a caption “I love going hiking here”



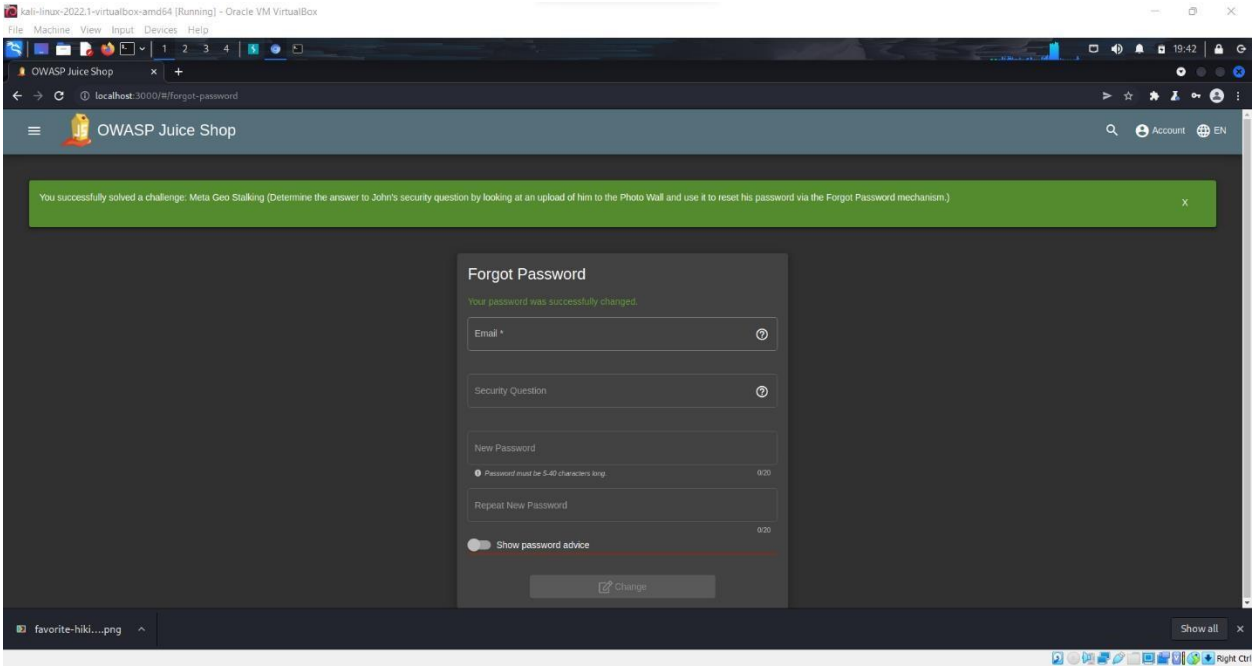
3- I downloaded the photo and used exiftool to see its metadata, the meta data contains the GPS position of this photo



4- I opened google maps and searched for that position and found out that the place's name is "Daniel Boone National Forest"



5- I went back to the forget password tab and entered the correct email and answer to the security question and was able to change the password successfully

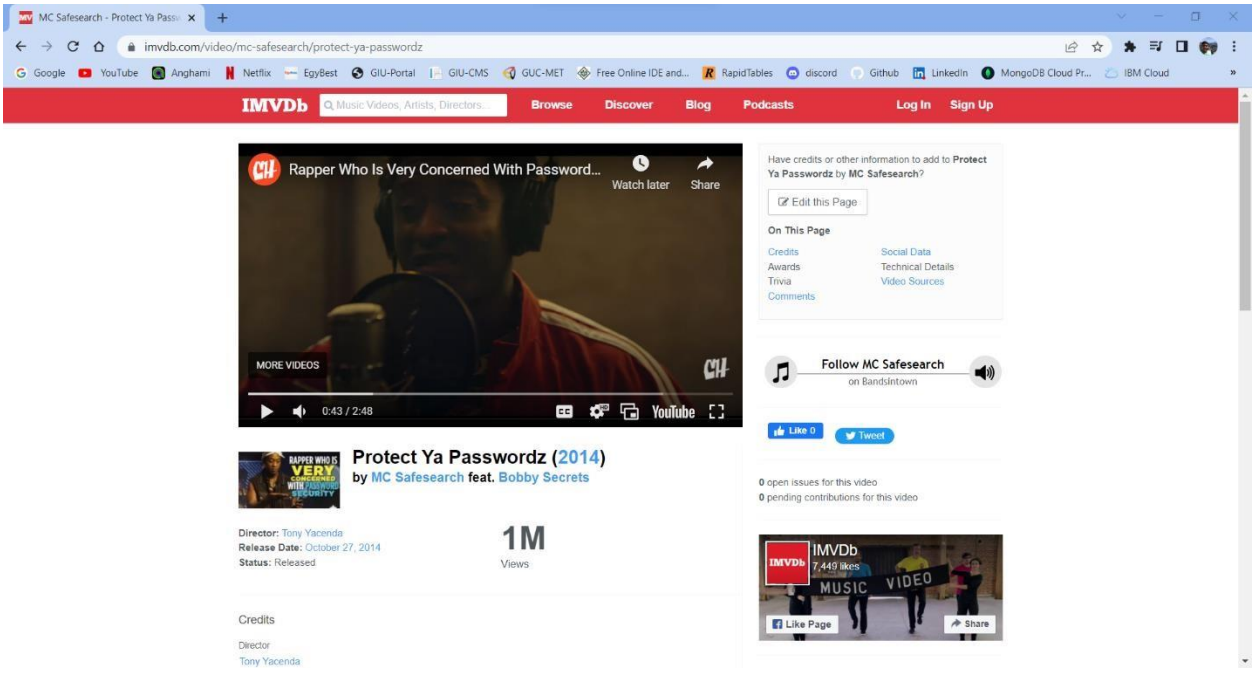


Attack number 6

1- Same as attack number 4 and 5, I got the email of MC SafeSearch from the attack number 3, his email is mc.safesearch@juice-sh.op

2- Next, I searched for his name on google and listened to a song for him (the first URL that appears in my google search), in the lyrics he mentioned “I say

use the first name of your favorite pet, mine is mr.noodles, no matter if you know cause I trickingly replaced some vowels with zeros", so I tried different combinations of mr.n00dles till I get the correct password is "Mr. N00dles"



3- Then I go to the login webpage and entered the correct email and password and was able to login successfully without any need of injections or bypass