

# Penetration Test Report of Findings



**OWASP CrAPI**

December 15, 2023

*Version 1*

# Table of Contents

Statement of Confidentiality .....2

Engagement Contacts .....3

Executive Summary .....4

    Approach .....4

    Scope .....5

    Assessment Overview and Recommendations .....5

Network Penetration Test Assessment Summary ..... 6

    Summary of Findings .....6

    Network Enumeration/Vulnerability Discovery .....7

Enumeration conducted by Wireshark:..... 11

Testing the collected information .....16

Remediation Summary ..... 23

Technical Findings Details .....24

## Statement of Confidentiality

The contents of this document have been developed as a part of final examination for CTC 458 Network Security through Penetration Testing class at California State University Dominguez Hills. The content of this report is subject to be made public as part of a portfolio, as well as to be used for educational purposes. The template of the document was provided by HackTheBox.

The contents of this document do not constitute legal advice. The assessment conducted is purely for educational purposes and will be submitted as a demonstration of work for the class’s final exam.

## Engagement Contacts

Inlanefreight Contacts		
Primary Contact	Title	Primary Contact Email
Hardipinder Singh Ubhi	Professor	hubhi1@csudh.edu

Assessor Contact		
Assessor Name	Title	Assessor Contact Email
Achraf El Khatib	Student	ashraf.khatib@outlook.com

## Executive Summary

As part of the final examination for CTC 458 Network Security through Penetration Testing class, the students are subjected to conduct a penetration test on OWASP's CrAPI open source Project which is hosted by the Assessor. The project is an intentionally vulnerable application meant for educational purposes on exploring and learning about API security.

## Approach

The testing is conducted within a Virtual Machine hosted by Microsoft's Hyper-V platform. The virtual machine will be running Kali Linux as the Operation System, and CrAPI would be hosted using Docker's "docker-compose" locally. Testing was performed with the goal of uncovering as many misconfigurations and vulnerabilities as possible. Testing was performed via a host that was provisioned specifically for this assessment. Each weakness identified was documented and investigated to determine exploitation possibilities and escalation potential. The tools used were: "Nmap", "Burp Suite", "OWASP ZAP", "Foxy Proxy"

## Scope

The scope of this assessment was one internal network range and the INLANEFREIGHT.LOCAL Active Directory domain.

### In-Scope Assets

Host/URL/IP Address	Description
127.0.0.1:8888	CrAPI
127.0.0.1:8085	MailHog

Table 1: Scope Details

## Assessment Overview and Recommendations

During the assessment of crAPI, multiple vulnerabilities have been found and tested, ranging from low to high risk. Some of the risks are as follow:

1. Poor implementation of headers leading to security risks where a user is able to manipulate the user's tokens getting access to another user's account.
2. Tokens aren't validated to verify the user's legitimacy for the request allowing for request calls to be manipulated and modified.
3. Some sections are vulnerable to SQL injection leaking information that can be abused such as the leakage of the coupon that's showcased in the report.
4. Request calls that can be manipulated to leak sensitive information such as locations, emails, tokens, etc.

In conclusion, the server application has been found to in a high-risk vulnerable state, and is not ready for a live publication without proper handling of the issues found.

# Network Penetration Test Assessment Summary

The testing has been conducted by launching the program using docker compose, which enabled the ability to test the server without affecting a live version.

## Summary of Findings

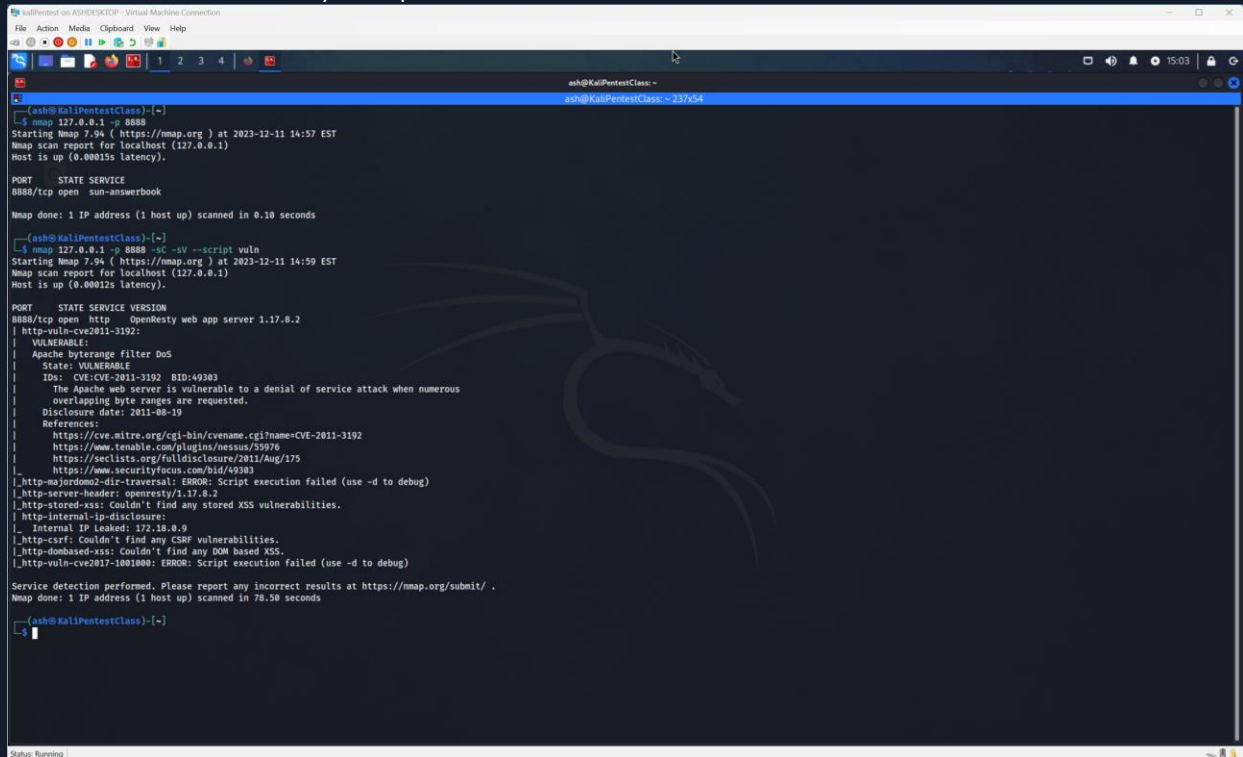
During the testing, many vulnerabilities

Finding Severity			
High	Medium	Low	Total
3	3	1	7

Table 2: Severity Summary

# Network Enumeration/Vulnerability Discovery

## 1. Enumeration conducted by Nmap:



```
ash@kali:~$ nmap 127.0.0.1 -p 8888
Starting Nmap 7.94 ( https://nmap.org ) at 2023-12-11 14:57 EST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00015s latency).

PORT      STATE SERVICE
8888/tcp  open  sun-answerbook

Nmap done: 1 IP address (1 host up) scanned in 0.10 seconds

ash@kali:~$ nmap 127.0.0.1 -p 8888 -sV --script vuln
Starting Nmap 7.94 ( https://nmap.org ) at 2023-12-11 14:59 EST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00012s latency).

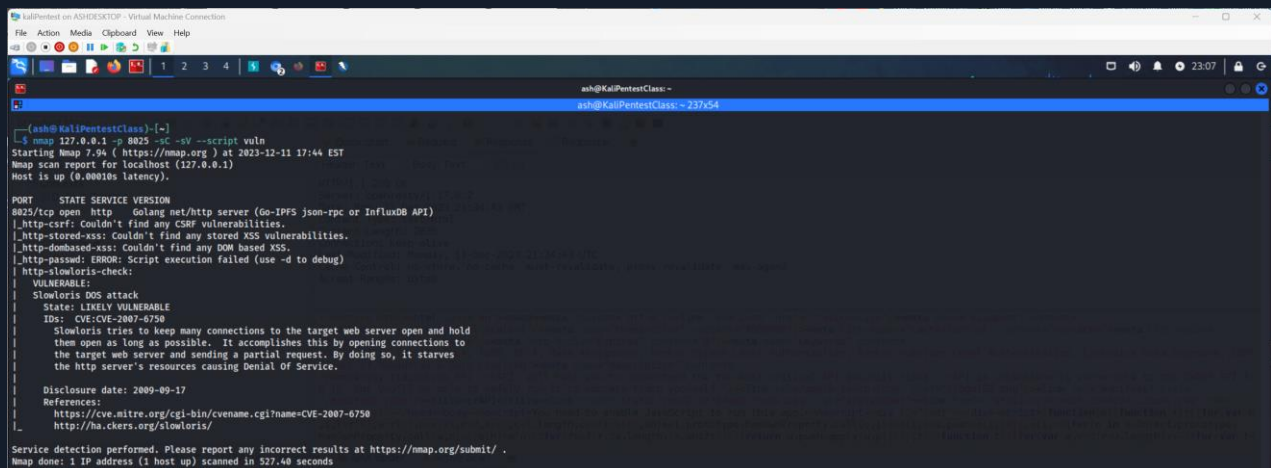
PORT      STATE SERVICE VERSION
8888/tcp  open  http      OpenResty web app server 1.17.8.2
|_ http-vuln-cve2011-3192:
|_   VULNERABLE:
|_   Apache byterange filter DoS
|_   State: VULNERABLE
|_   IDS: CVE:2011-3192 BID:40383
|_   The Apache web server is vulnerable to a denial of service attack when numerous
|_   overlapping byte ranges are requested.
|_   Disclosure date: 2011-08-19
|_   References:
|_   - https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-3192
|_   - https://www.tenable.com/plugins/nessus/55976
|_   - https://seclists.org/fulldisclosure/2011/Aug/175
|_   - https://www.securityfocus.com/bid/49383
|_ http-majordomo2-dir-traversal: ERROR: Script execution failed (use -d to debug)
|_ http-server-header: openresty/1.17.8.2
|_ http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_ http-internal-ip-disclosure:
|_   Internal IP Leaked: 172.18.0.9
|_ http-csrf: Couldn't find any CSRF vulnerabilities.
|_ http-dombased-xss: Couldn't find any DOM based XSS.
|_ http-vuln-cve2017-1801800: ERROR: Script execution failed (use -d to debug)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 78.58 seconds

ash@kali:~$
```

picture 1: Nmap for port 8888

The Nmap scan gave insight on an open vulnerability concerning CVE E-2011-3192 for port 8888 which indicates a vulnerability for a denial-of-service attack (DoS)



```
ash@kali:~$ nmap 127.0.0.1 -p 8025 -sV --script vuln
Starting Nmap 7.94 ( https://nmap.org ) at 2023-12-11 17:44 EST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00010s latency).

PORT      STATE SERVICE
8025/tcp  open  GoLang net/http server (Go-IPFS json-rpc or InfluxDB API)

|_ http-csrf: Couldn't find any CSRF vulnerabilities.
|_ http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_ http-dombased-xss: Couldn't find any DOM based XSS.
|_ http-passwd: ERROR: Script execution failed (use -d to debug)
|_ http-slowloris-check:
|_   VULNERABLE:
|_   Slowloris DoS attack
|_   State: LIKELY VULNERABLE
|_   IDS: CVE:2007-6750
|_   Slowloris tries to keep many connections to the target web server open and hold
|_   them open as long as possible. It accomplishes this by opening connections to
|_   the target web server and sending a partial request. By doing so, it starves
|_   the http server's resources causing Denial Of Service.
|_   Disclosure date: 2009-09-17
|_   References:
|_   - https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-6750
|_   - http://ha.ckers.org/slowloris/

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 527.48 seconds

ash@kali:~$
```

picture 2: Nmap scan for port 8025

The Nmap scan gave insight on an open vulnerability concerning CVE-2007-6750 for port 8025 which indicates a vulnerability for a denial-of-service attack (DoS)

## 2. Enumeration conducted by OWASP ZAP:

Cloud Metadata Potentially Exposed

URL: <http://localhost:8888/latest/meta-data/>

Risk: High

Confidence: Low

Parameter:

Attack: 169.254.169.254

Evidence:

CWE ID: 0

WASC ID: 0

Description:  
The Cloud Metadata Attack attempts to abuse a misconfigured NGINX server in order to access the instance metadata maintained by cloud service providers such as AWS, GCP and Azure. All of these providers provide metadata via an internal unroutable IP address '169.254.169.254' - this can be exposed by incorrectly configured NGINX servers and accessed by using

Other Info:  
Based on the successful response status code cloud metadata may have been returned in the response. Check the response data to see if any cloud metadata has been returned. The meta data returned can include information that would allow an attacker to completely compromise the system.

Solution:  
Do not trust any user data in NGINX configs. In this case it is probably the use of the \$host variable which is set from the 'Host' header and can be controlled by an attacker.

Reference:  
<https://www.nginx.com/blog/trust-no-one-perils-of-trusting-user-input/>

Alert Tags:

Key	Value
OWASP_2021_A05	<a href="https://owasp.org/Top10/A05_2021-Security_Misconfiguration/">https://owasp.org/Top10/A05_2021-Security_Misconfiguration/</a>
OWASP_2017_A06	<a href="https://owasp.org/www-project-top-ten/2017/A6_2017-Security_Misconfiguration.html">https://owasp.org/www-project-top-ten/2017/A6_2017-Security_Misconfiguration.html</a>

picture 3: Automated scan by ZAP (1)

The following flag showcases that users within the NGINX configs would establish a threat to the service as it's misconfigured.

.env Information Leak

URL: <http://localhost:8888/.env>

Risk: Medium

Confidence: Medium

Parameter:

Attack:

Evidence: HTTP/1.1 200 OK

CWE ID: 215

WASC ID: 13

Description:  
One or more .env files seems to have been located on the server. These files often expose infrastructure or administrative account credentials, API or APP keys, or other sensitive configuration information. |

Other Info:

Solution:  
Ensure the .env file is not accessible.

Reference:  
[https://www.google.com/search?q=db\\_password+filetype%3Aenv](https://www.google.com/search?q=db_password+filetype%3Aenv)  
<https://mobile.twitter.com/svblxyz/status/1045013939904532482>

Alert Tags:

Key	Value
OWASP_2021_A05	<a href="https://owasp.org/Top10/A05_2021-Security_Misconfiguration/">https://owasp.org/Top10/A05_2021-Security_Misconfiguration/</a>
WSTG-v42-CONF-05	<a href="https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_...">https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_...</a>
OWASP_2017_A06	<a href="https://owasp.org/www-project-top-ten/2017/A6_2017-Security_Misco...">https://owasp.org/www-project-top-ten/2017/A6_2017-Security_Misco...</a>

Cancel Save

picture 4: Automated scan by ZAP (2)

The following flag showcases that with the .env file being leaked it exposes sensitive information.



**Edit Alert**

Content Security Policy (CSP) Header Not Set

URL: http://localhost:8888

Risk: Medium

Confidence: High

Parameter:

Attack:

Evidence:

CWE ID: 693

WASC ID: 15

Description:

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources

Other Info:

Solution:

Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.

Reference:

[https://developer.mozilla.org/en-US/docs/Web/Security/CSP/Introducing\\_Content\\_Security\\_Policy](https://developer.mozilla.org/en-US/docs/Web/Security/CSP/Introducing_Content_Security_Policy)  
[https://cheatsheetseries.owasp.org/cheatsheets/Content\\_Security\\_Policy\\_Cheat\\_Sheet.html](https://cheatsheetseries.owasp.org/cheatsheets/Content_Security_Policy_Cheat_Sheet.html)  
<http://www.w3.org/TR/CSP/>

Alert Tags:

Key	Value
OWASP_2021_A05	<a href="https://owasp.org/Top10/A05_2021-Security_Misconfiguration/">https://owasp.org/Top10/A05_2021-Security_Misconfiguration/</a>
OWASP_2017_A06	<a href="https://owasp.org/www-project-top-ten/2017/A6_2017-Securi...">https://owasp.org/www-project-top-ten/2017/A6_2017-Securi...</a>

Cancel Save

picture 5: Automated scan by ZAP (3)

This flag showcases that the content security policy header is currently not setup

**Edit Alert**

Missing Anti-clickjacking Header

URL: http://localhost:8888

Risk: Medium

Confidence: Medium

Parameter: x-frame-options

Attack:

Evidence:

CWE ID: 1021

WASC ID: 15

Description:

The response does not include either Content-Security-Policy with 'frame-ancestors' directive or X-Frame-Options to protect against 'Clickjacking' attacks.

Other Info:

Solution:

Modern Web browsers support the Content-Security-Policy and X-Frame-Options HTTP headers. Ensure one of them is set on all web pages returned by your site/app.  
 If you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN,

Reference:

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options>

Alert Tags:

Key	Value
OWASP_2021_A05	<a href="https://owasp.org/Top10/A05_2021-Security_Misconfiguration/">https://owasp.org/Top10/A05_2021-Security_Misconfiguration/</a>
WSTG-v42-CLNT-09	<a href="https://owasp.org/www-project-web-security-testing-guide/v42/4-W...">https://owasp.org/www-project-web-security-testing-guide/v42/4-W...</a>
OWASP_2017_A06	<a href="https://owasp.org/www-project-top-ten/2017/A6_2017-Security_Mi...">https://owasp.org/www-project-top-ten/2017/A6_2017-Security_Mi...</a>

Cancel Save

picture 6: Automated scan by ZAP (4)

The following flag showcases that there's a missing anti-clickjacking header

**Edit Alert**

Server Leaks Version Information via "Server" HTTP Response Header Field

URL: http://localhost:8888

Risk: Low

Confidence: High

Parameter:

Attack:

Evidence: openresty/1.17.8.2

CWE ID: 200

WASC ID: 13

Description:  
The web/application server is leaking version information via the "Server" HTTP response header. Access to such information may facilitate attackers identifying other vulnerabilities your web/application server is subject to.

Other Info:

Solution:  
Ensure that your web server, application server, load balancer, etc. is configured to suppress the "Server" header or provide generic details.

Reference:  
<http://httpd.apache.org/docs/current/mod/core.html#servertokens>  
[http://msdn.microsoft.com/en-us/library/ff648552.aspx#ht\\_uriscan\\_007](http://msdn.microsoft.com/en-us/library/ff648552.aspx#ht_uriscan_007)  
<http://blogs.msdn.com/b/varunm/archive/2013/04/23/remove-unwanted-http-response-headers.aspx>

Alert Tags:

Key ^	Value
OWASP_2017_A06	<a href="https://owasp.org/www-project-top-ten/2017/A6_2017-Security_Misconfiguration...">https://owasp.org/www-project-top-ten/2017/A6_2017-Security_Misconfiguration...</a>
OWASP_2021_A05	<a href="https://owasp.org/Top10/A05_2021-Security_Misconfiguration/">https://owasp.org/Top10/A05_2021-Security_Misconfiguration/</a>
WSTG-V42-INFO-02	<a href="https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Application...">https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Application...</a>

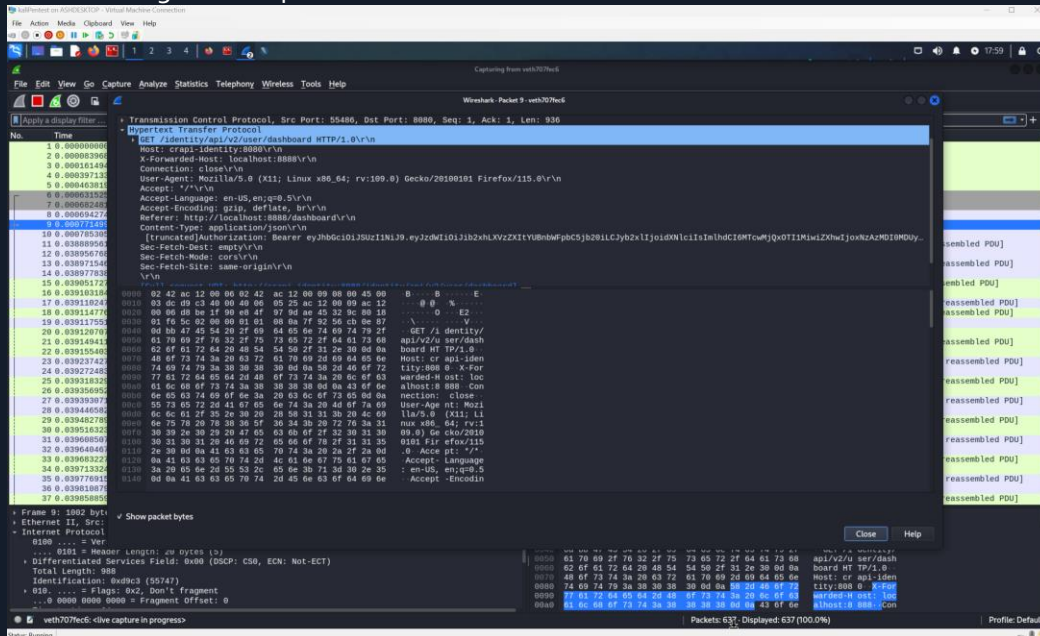
Cancel Save

picture 7: Automated scan by ZAP (5)

The following flag showcases an alert that the server leaks its version information via “server” HTTP response header field.

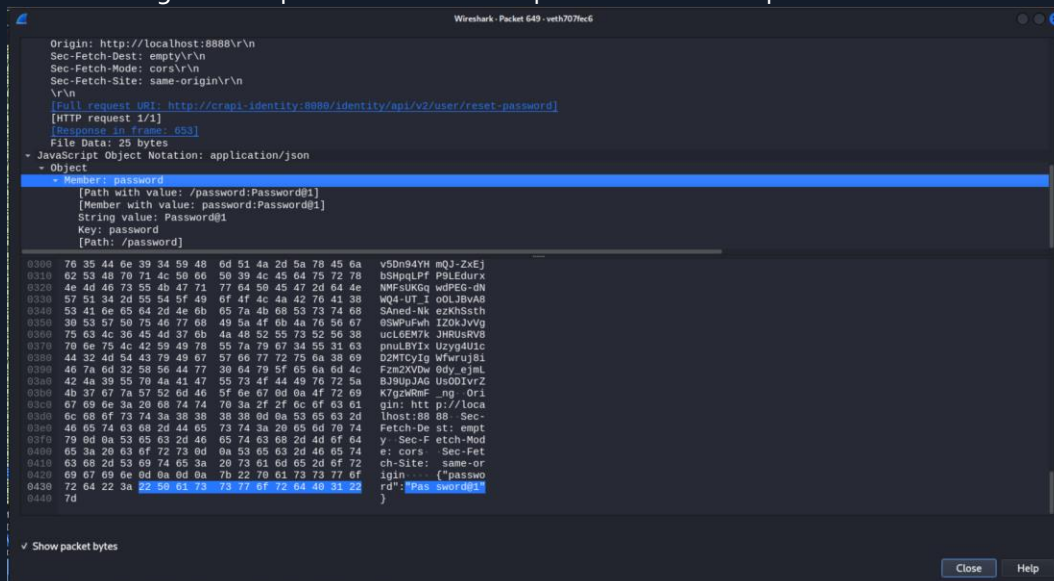
## Enumeration conducted by Wireshark:

1. The following PCAP capture the token ID for the account was able to be leaked



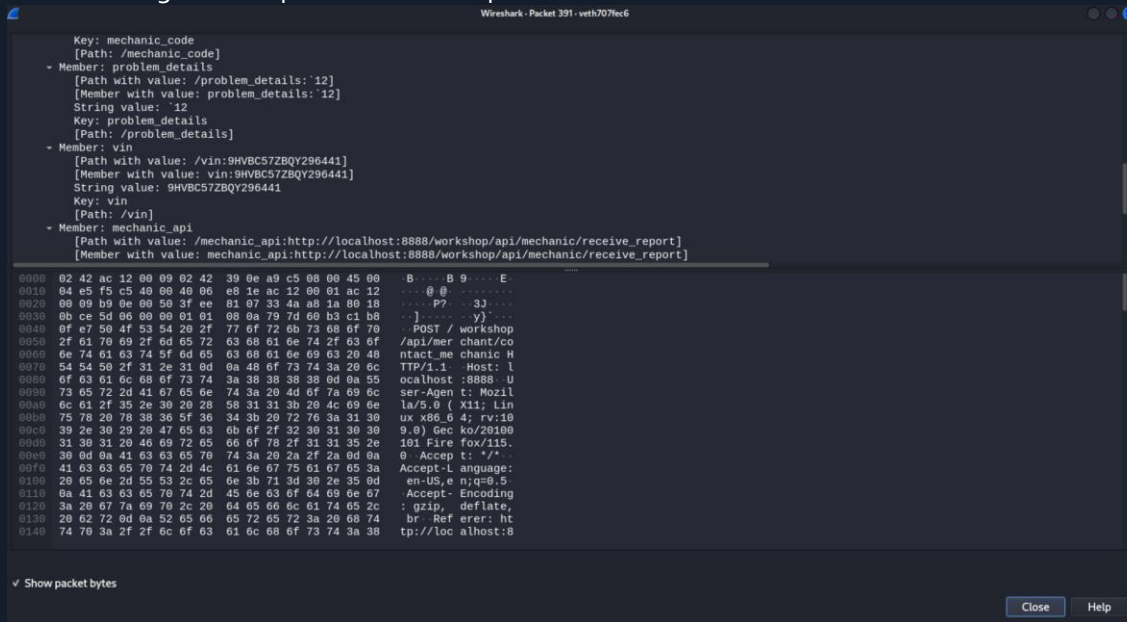
picture 8: PCAP capture (1)

2. The following PCAP capture showcases the password after the password reset was conducted



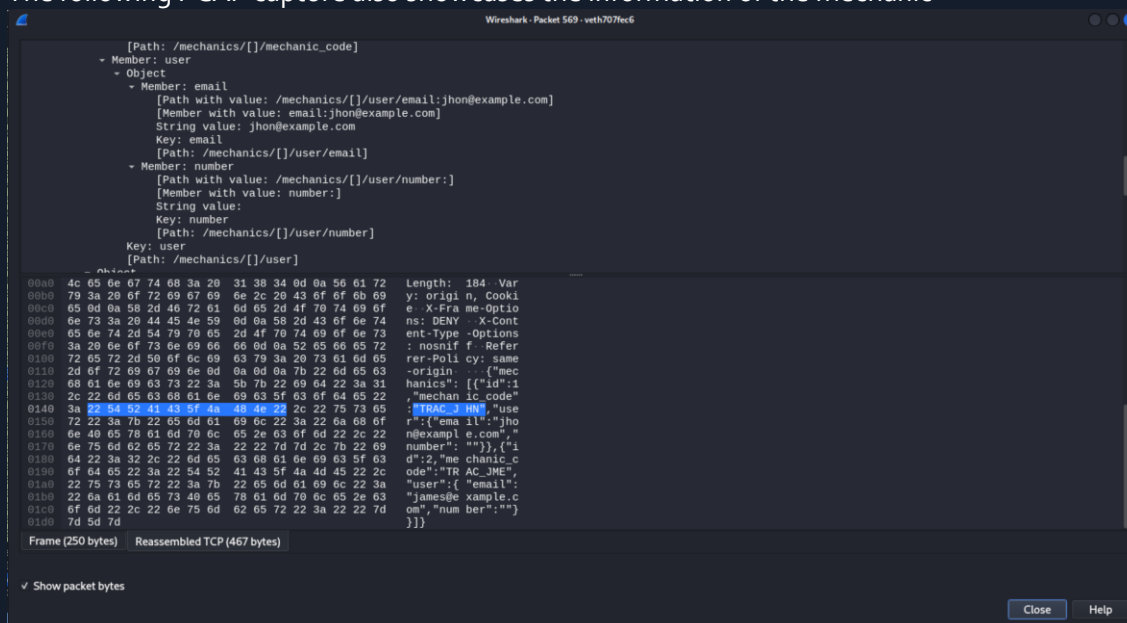
picture 9: PCAP capture (2)

3. The following PCAP capture showcases personal information of the user that would be leaked



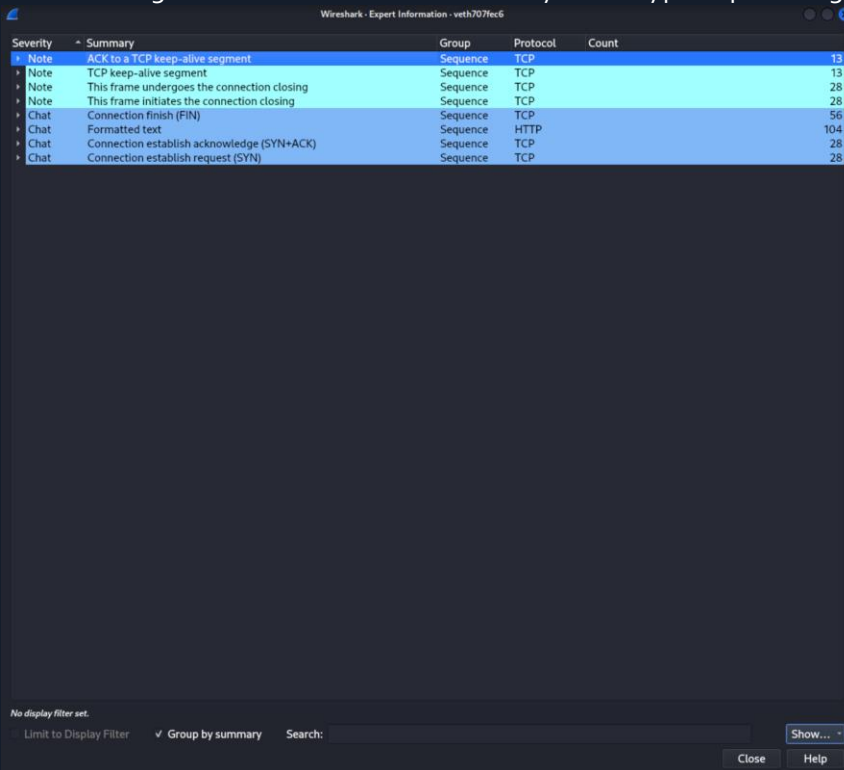
picture 10: PCAP capture (3)

4. The following PCAP capture also showcases the information of the mechanic



picture 11: PCAP capture (4)

5. The following table showcases the a summary of the type of packets gathered



Severity	Summary	Group	Protocol	Count
Note	ACK to a TCP keep-alive segment	Sequence	TCP	13
Note	TCP keep-alive segment	Sequence	TCP	13
Note	This frame undergoes the connection closing	Sequence	TCP	28
Note	This frame initiates the connection closing	Sequence	TCP	28
Chat	Connection finish (FIN)	Sequence	TCP	56
Chat	Formatted text	Sequence	HTTP	104
Chat	Connection establish acknowledge (SYN+ACK)	Sequence	TCP	28
Chat	Connection establish request (SYN)	Sequence	TCP	28

No display filter set.

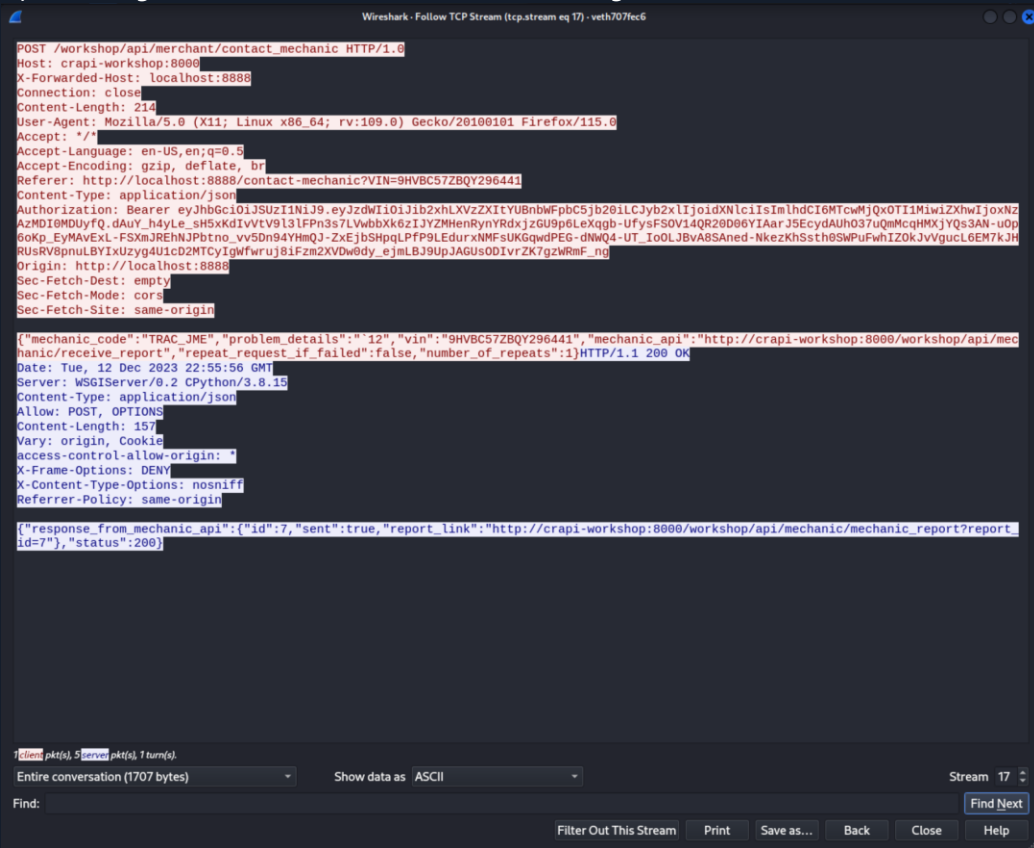
☐ Limit to Display Filter    ☒ Group by summary    Search:            

picture 12: Expert Information

From the following we can conclude:

1. There's been 13 attempts for an ACK to TCP.
2. There's been 13 attempts for TCP keep-alive segments.
3. The frame has been closed 28 times.
4. The frame has undergone closure 28 times.
5. 104 HTTP packets have been captured which would showcase the information in plain text.
6. The connection request has been sent 28 times.
7. The connection has established an acknowledge (SYN+ACK) 28 times.
8. The connection has finished (FIN) 56 times.

6. By following the TCP stream we found the following:



picture 13: TCP stream

From the TCP stream we can conclude the following:

1. The token of the user.
2. Personal information such as the vin number.
3. The date of the request.
4. The report ID.
5. The report's link.
6. The mechanic's API.

7. The following are property details from the capture.

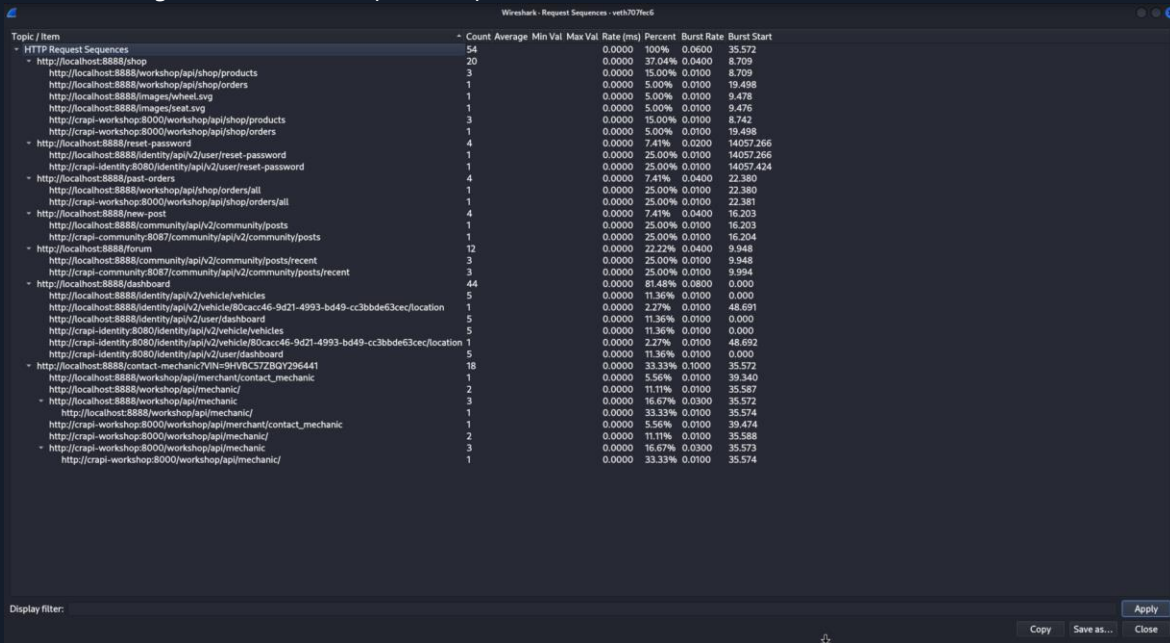
Measurement	Captured	Displayed	Marked
Packets	676	676 (100.0%)	—
Time span, s	18238.914	18238.914	—
Average pps	0.0	0.0	—
Average packet size, B	1180	1180	—
Bytes	797823	797823 (100.0%)	0
Average bytes/s	43	43	—
Average bits/s	349	349	—

picture 14: properties details

From the following properties we can conclude the following:

1. The capture had an average capture speed of 43 bytes/s.
2. The capture collected 676 packets.
3. The average packet size is 1180 B.

## 8. The following are the HTTP request sequences



Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
HTTP Request Sequences	54				0.0000	100%	0.0600	35.572
http://localhost:8888/shop	20				0.0000	37.04%	0.0400	8.709
http://localhost:8888/workshop/api/shop/products	3				0.0000	15.00%	0.0100	8.709
http://localhost:8888/workshop/api/shop/orders	1				0.0000	5.00%	0.0100	19.498
http://localhost:8888/images/wheel.svg	1				0.0000	5.00%	0.0100	9.478
http://localhost:8888/images/wheel.svg	1				0.0000	5.00%	0.0100	9.478
http://localhost:8888/workshop/api/shop/products	3				0.0000	15.00%	0.0100	8.742
http://localhost:8888/workshop/api/shop/orders	1				0.0000	5.00%	0.0100	19.498
http://localhost:8888/reset-password	4				0.0000	7.41%	0.0200	14057.266
http://localhost:8888/reset-password	1				0.0000	25.00%	0.0100	14057.266
http://localhost:8888/reset-password	1				0.0000	25.00%	0.0100	14057.424
http://localhost:8888/past-orders	4				0.0000	7.41%	0.0400	22.380
http://localhost:8888/workshop/api/shop/orders/all	1				0.0000	25.00%	0.0100	22.380
http://localhost:8888/workshop/api/shop/orders/all	1				0.0000	25.00%	0.0100	22.381
http://localhost:8888/new-post	4				0.0000	7.41%	0.0400	16.203
http://localhost:8888/community/api/v2/community/posts	1				0.0000	25.00%	0.0100	16.203
http://localhost:8888/community/api/v2/community/posts	1				0.0000	25.00%	0.0100	16.204
http://localhost:8888/forum	12				0.0000	22.22%	0.0400	9.948
http://localhost:8888/forum	3				0.0000	25.00%	0.0100	9.948
http://localhost:8888/forum	3				0.0000	25.00%	0.0100	9.994
http://localhost:8888/dashboard	44				0.0000	81.48%	0.0800	0.000
http://localhost:8888/dashboard	5				0.0000	11.36%	0.0100	0.000
http://localhost:8888/dashboard	1				0.0000	2.27%	0.0100	48.691
http://localhost:8888/dashboard	5				0.0000	11.36%	0.0100	0.000
http://localhost:8888/dashboard	5				0.0000	11.36%	0.0100	0.000
http://localhost:8888/dashboard	1				0.0000	2.27%	0.0100	48.692
http://localhost:8888/dashboard	5				0.0000	11.36%	0.0100	0.000
http://localhost:8888/contact-mechanic?VIN=9HVB5C72BQ1296441	18				0.0000	33.33%	0.1000	35.572
http://localhost:8888/workshop/api/mechanic/contact_mechanic	1				0.0000	5.56%	0.0100	39.340
http://localhost:8888/workshop/api/mechanic	2				0.0000	11.11%	0.0100	35.587
http://localhost:8888/workshop/api/mechanic	3				0.0000	16.67%	0.0300	35.572
http://localhost:8888/workshop/api/mechanic	1				0.0000	33.33%	0.0100	35.574
http://localhost:8888/workshop/api/mechanic/contact_mechanic	1				0.0000	5.56%	0.0100	39.474
http://localhost:8888/workshop/api/mechanic	2				0.0000	11.11%	0.0100	35.588
http://localhost:8888/workshop/api/mechanic	3				0.0000	16.67%	0.0300	35.573
http://localhost:8888/workshop/api/mechanic	1				0.0000	33.33%	0.0100	35.574

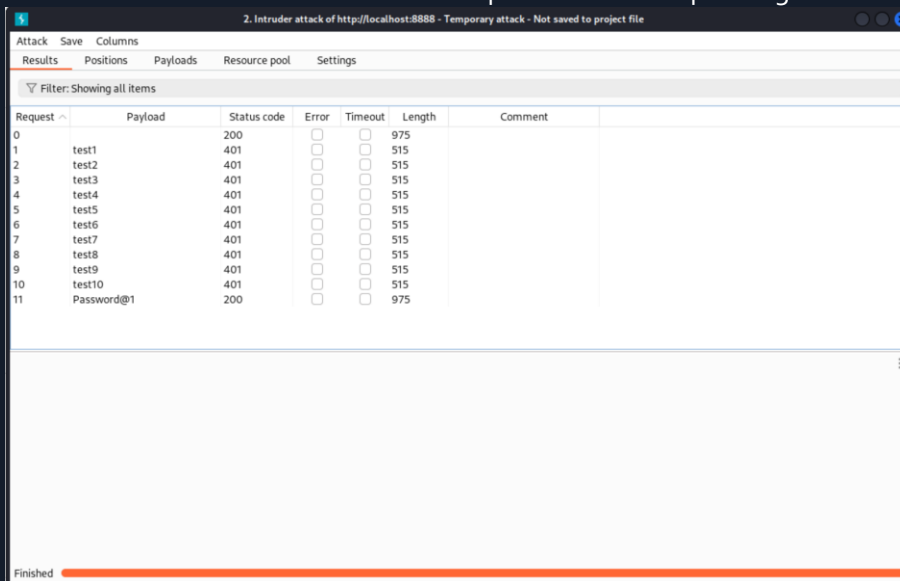
picture 15: request sequence HTTP

Within the request sequences the following is noticed:

1. Packets for shop orders.
2. Packets for the password reset.
3. Packets for past orders.
4. Packets for new posts
5. Packets for the forum
6. Packets for dashboard
7. Packets for the mechanic
8. The count for each packet
9. The rate for each packet

## Testing the collected information

1. Testing the login section with burpsuite:
  - a. After conducting a simulated dictionary attack on the password section by utilizing burpsuite's intruder feature it was concluded that there's no protection set in place against it.

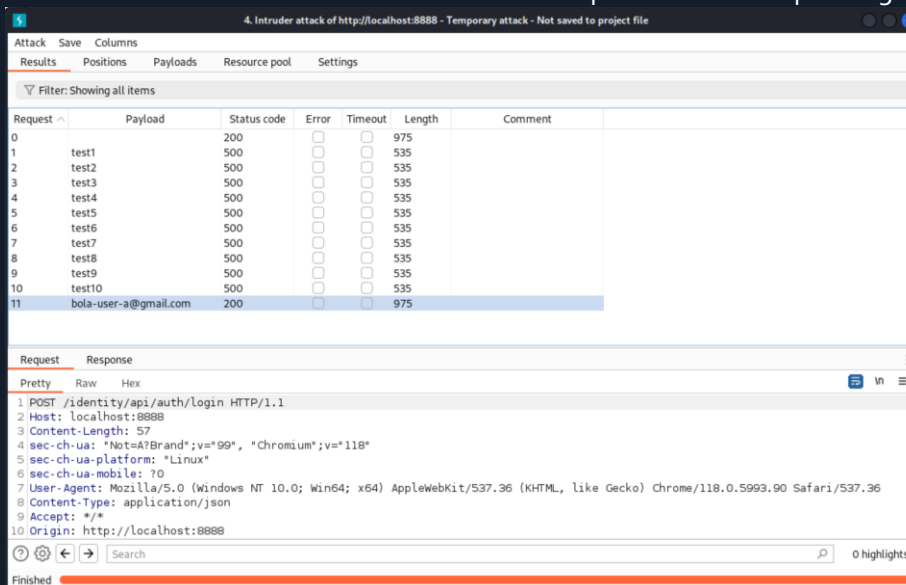


The screenshot shows the Burp Suite Intruder interface for an attack on the password section. The table lists 11 requests, with the first 10 being test payloads and the 11th being the password 'Password@1'. All test requests returned a 401 status code, while the password request returned a 200 status code, indicating a successful login.

Request	Payload	Status code	Error	Timeout	Length	Comment
0		200	<input type="checkbox"/>	<input type="checkbox"/>	975	
1	test1	401	<input type="checkbox"/>	<input type="checkbox"/>	515	
2	test2	401	<input type="checkbox"/>	<input type="checkbox"/>	515	
3	test3	401	<input type="checkbox"/>	<input type="checkbox"/>	515	
4	test4	401	<input type="checkbox"/>	<input type="checkbox"/>	515	
5	test5	401	<input type="checkbox"/>	<input type="checkbox"/>	515	
6	test6	401	<input type="checkbox"/>	<input type="checkbox"/>	515	
7	test7	401	<input type="checkbox"/>	<input type="checkbox"/>	515	
8	test8	401	<input type="checkbox"/>	<input type="checkbox"/>	515	
9	test9	401	<input type="checkbox"/>	<input type="checkbox"/>	515	
10	test10	401	<input type="checkbox"/>	<input type="checkbox"/>	515	
11	Password@1	200	<input type="checkbox"/>	<input type="checkbox"/>	975	

picture 16: Intruder Attack (1)

- b. After conducting another simulated dictionary attack on the username section by utilizing burpsuite's intruder feature it was concluded that there's no protection set in place against it.



The screenshot shows the Burp Suite Intruder interface for an attack on the username section. The table lists 11 requests, with the first 10 being test payloads and the 11th being the username 'bola-user-a@gmail.com'. All test requests returned a 500 status code, while the username request returned a 200 status code, indicating a successful login.

Request	Payload	Status code	Error	Timeout	Length	Comment
0		200	<input type="checkbox"/>	<input type="checkbox"/>	975	
1	test1	500	<input type="checkbox"/>	<input type="checkbox"/>	535	
2	test2	500	<input type="checkbox"/>	<input type="checkbox"/>	535	
3	test3	500	<input type="checkbox"/>	<input type="checkbox"/>	535	
4	test4	500	<input type="checkbox"/>	<input type="checkbox"/>	535	
5	test5	500	<input type="checkbox"/>	<input type="checkbox"/>	535	
6	test6	500	<input type="checkbox"/>	<input type="checkbox"/>	535	
7	test7	500	<input type="checkbox"/>	<input type="checkbox"/>	535	
8	test8	500	<input type="checkbox"/>	<input type="checkbox"/>	535	
9	test9	500	<input type="checkbox"/>	<input type="checkbox"/>	535	
10	test10	500	<input type="checkbox"/>	<input type="checkbox"/>	535	
11	bola-user-a@gmail.com	200	<input type="checkbox"/>	<input type="checkbox"/>	975	

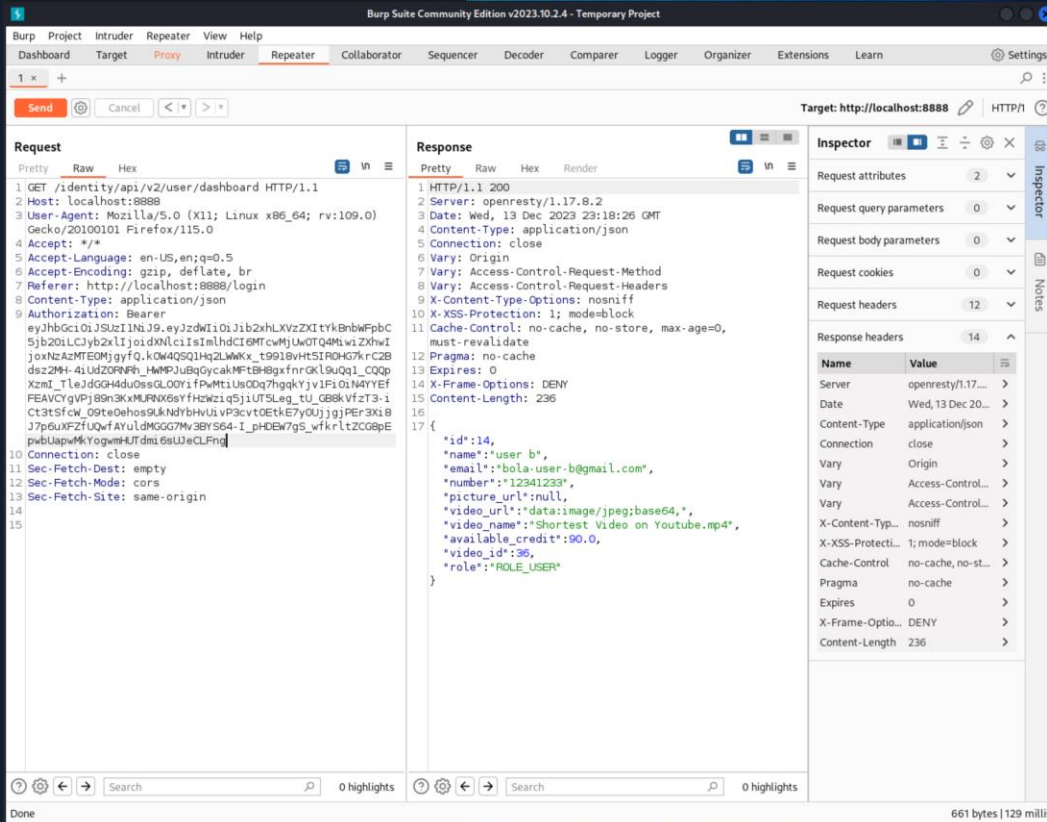
Below the table, the 'Request' tab is selected, showing the raw HTTP request details:

```
1 POST /identity/api/auth/login HTTP/1.1
2 Host: localhost:8888
3 Content-Length: 57
4 sec-ch-ua: "NotA?Brand";v="99", "Chromium";v="118"
5 sec-ch-ua-platform: "Linux"
6 sec-ch-ua-mobile: ?0
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.90 Safari/537.36
8 Content-Type: application/json
9 Accept: */*
10 Origin: http://localhost:8888
```

picture 17: intruder Attack (2)

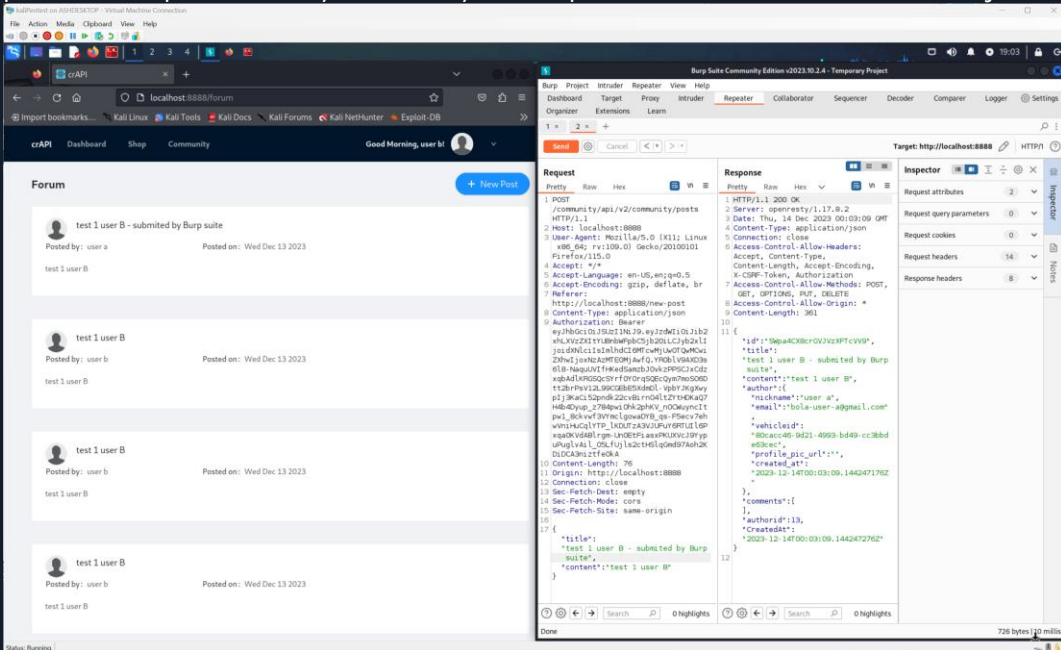


- c. After conducting a simulated attack by capturing user's login token, it was determined that the login function doesn't verify the token's ID and so making it vulnerable for account hijacking.



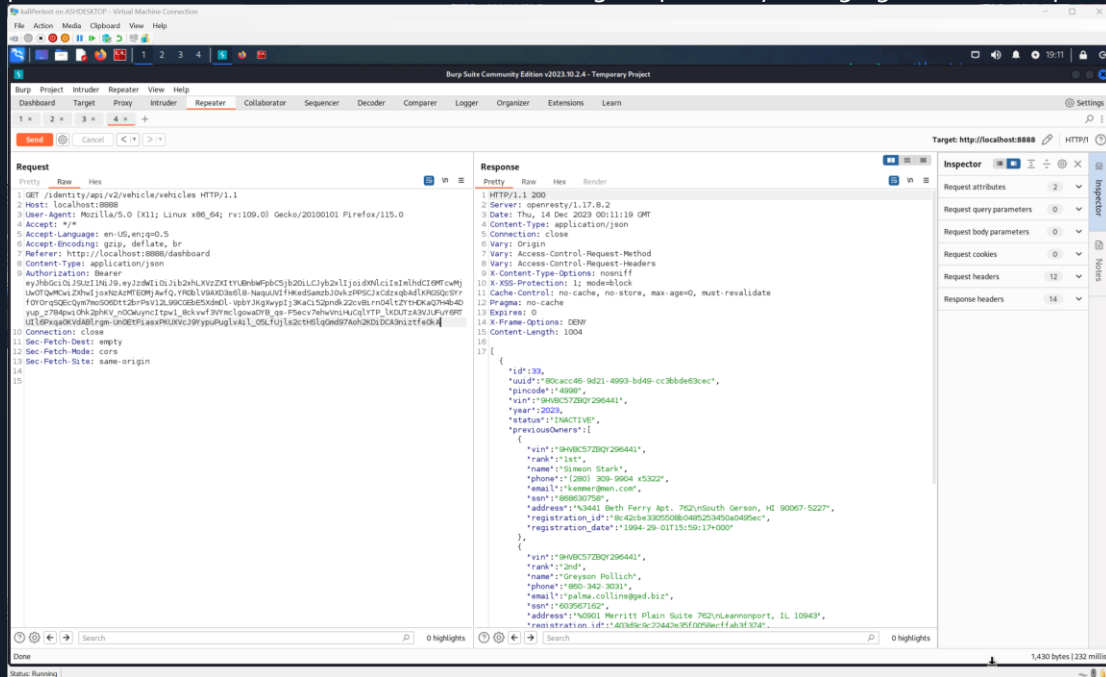
picture 18: Repeater attack (1)

- d. After conducting a test on the community section using the repeater, it was determined that there's no protection in place to verify the validity of who posted the comment and can be hijacked.



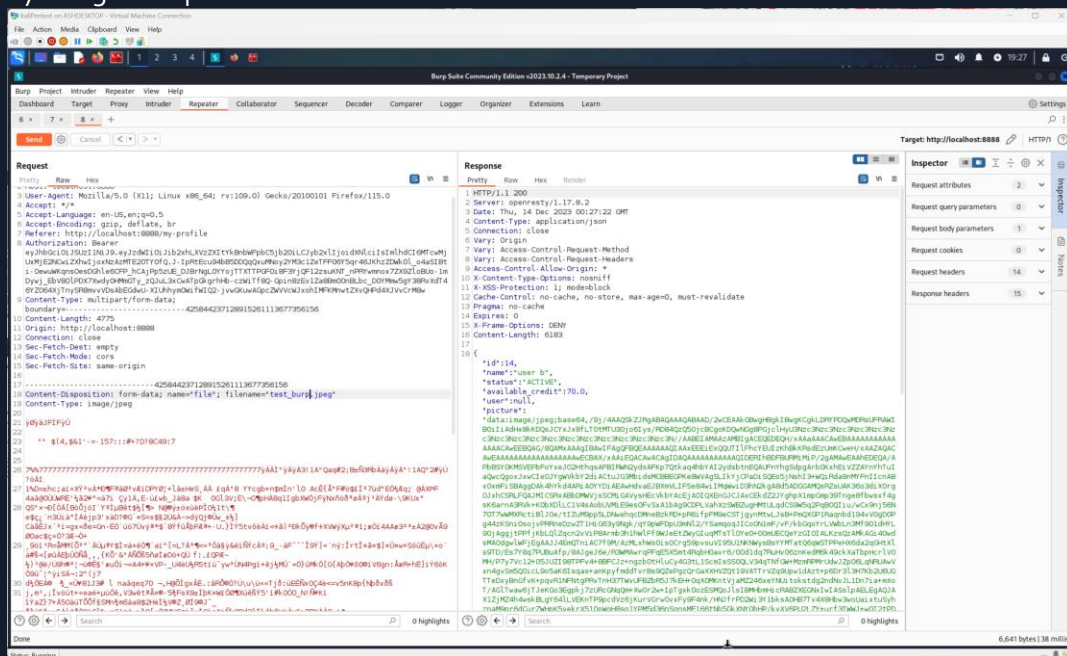
picture 19: Repeater attack (2)

- e. After testing the vehicle API section, it was determined that personal information regarding the owner and previous owners as well as sensitive information about the vehicle where abouts, vin number previous owner's addresses can be collected using a repeater by changing the token requesting the call.



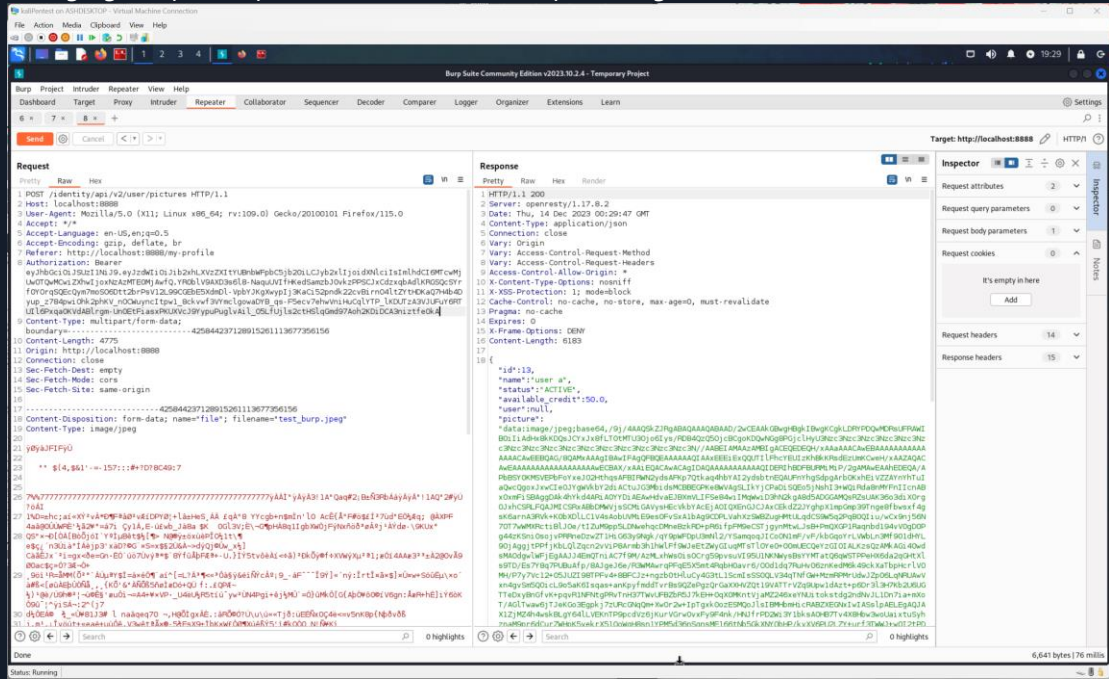
picture 20: Repeater attack (3)

- f. After testing the profile picture section, it was determined that the name of the picture can be changed by using the repeater



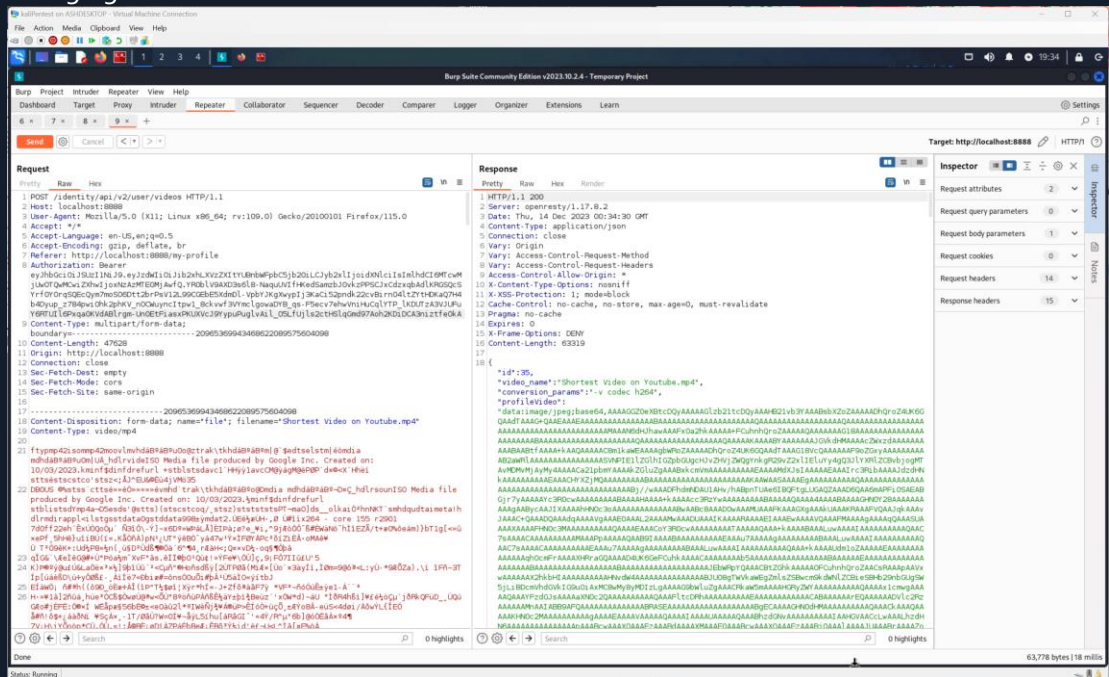
picture 21: Repeater attack (4)

## changing the profile picture of another user by utilizing the same method



picture 22: Repeater attack (5)

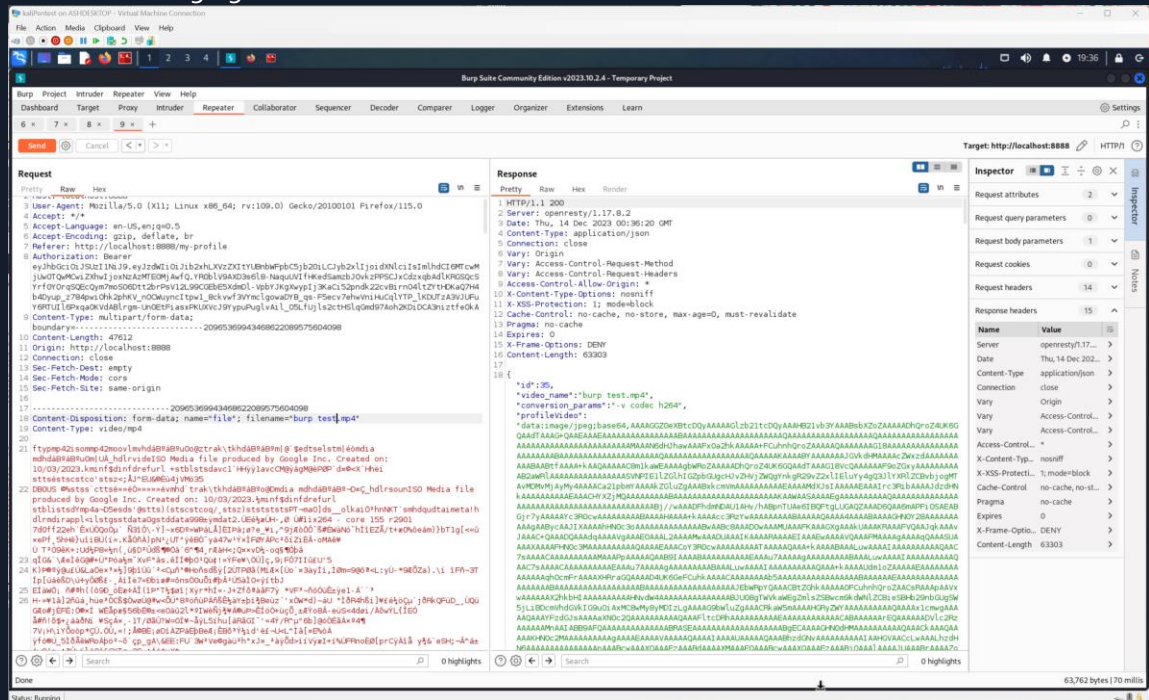
- g. After conducting a test on the video section, it was determined that the user's video can be changed by changing Bearer token



picture 23: Repeater attack (6)

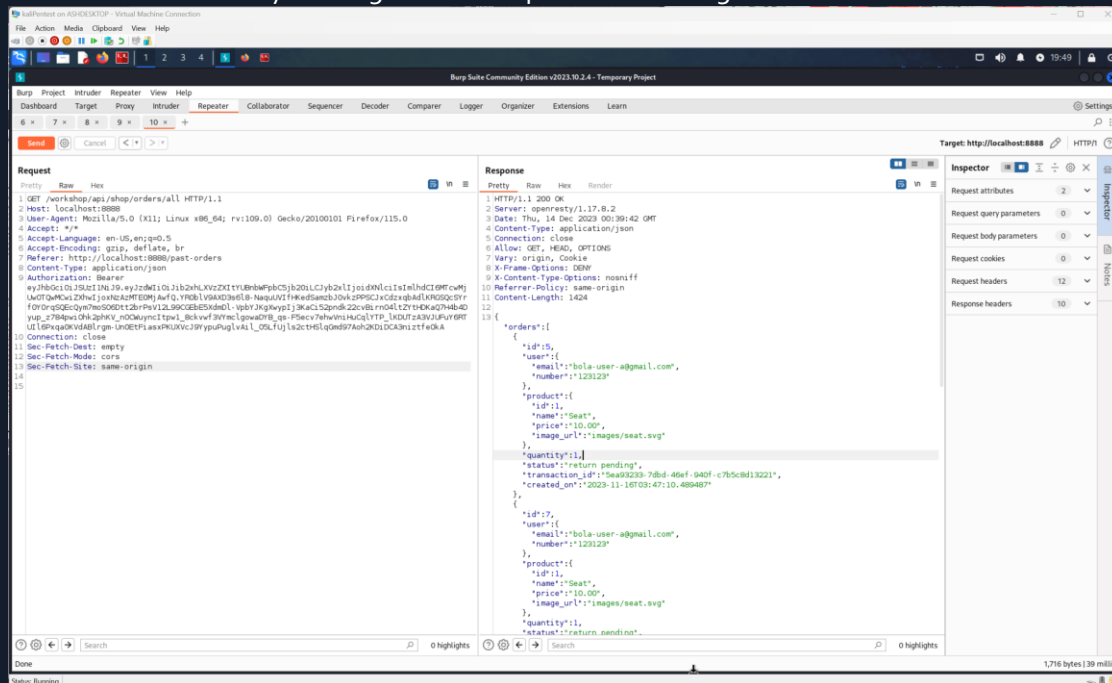


## As well as changing the video's name



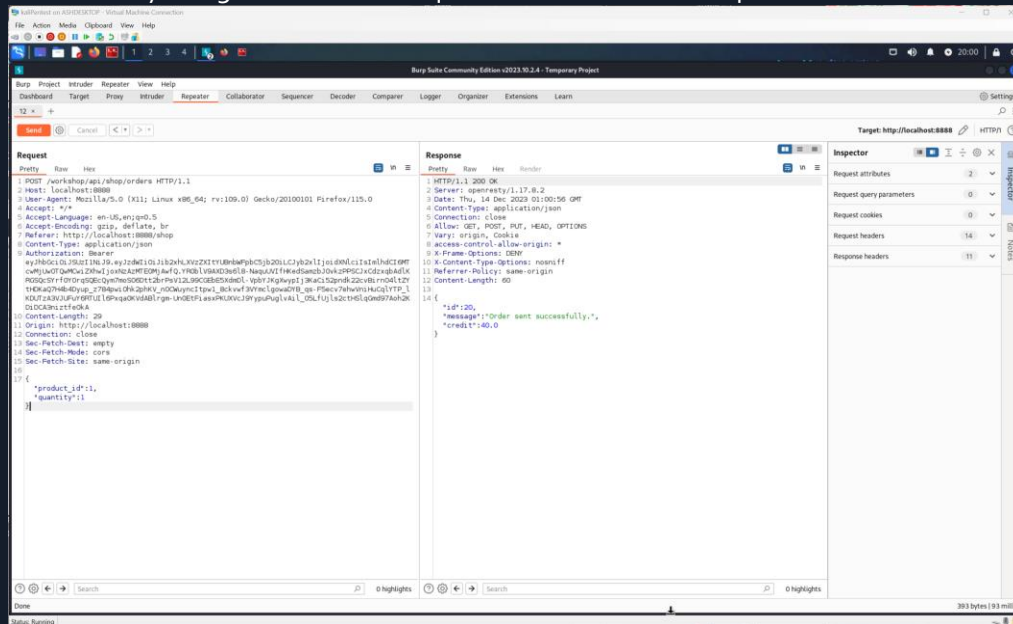
picture 24: repeater attack (7)

- h. After conducting a test on the shop requests, its also determined that there's no verifications in place to verify the legitimacy of the request.
- Which would allow anyone to grab a user's past orders using their token.



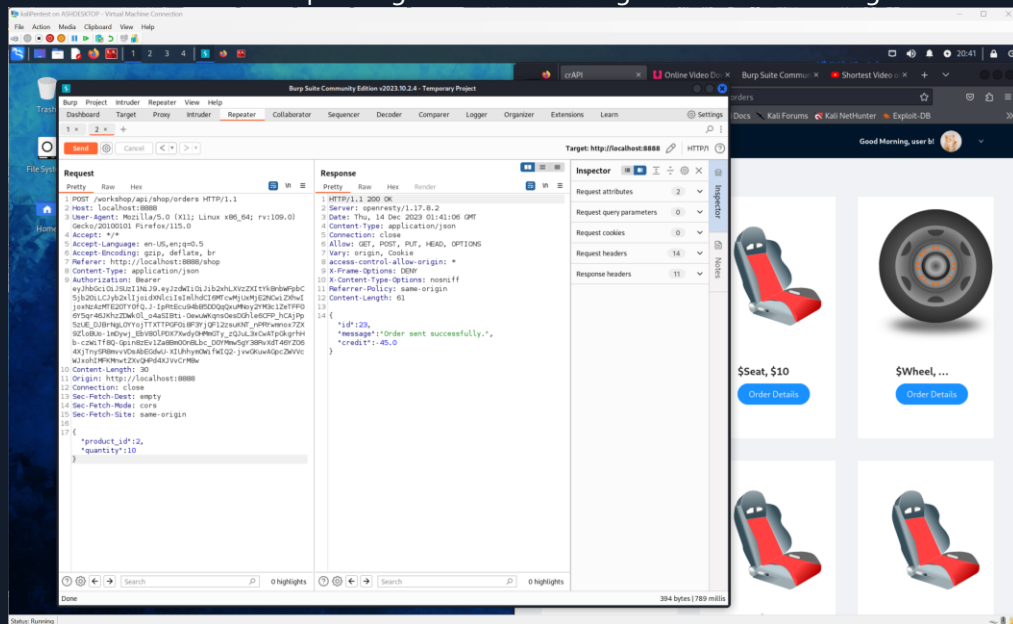
picture 25: repeater attack (8)

As well as by using the order call request we are able to manipulate the token to order for another user.



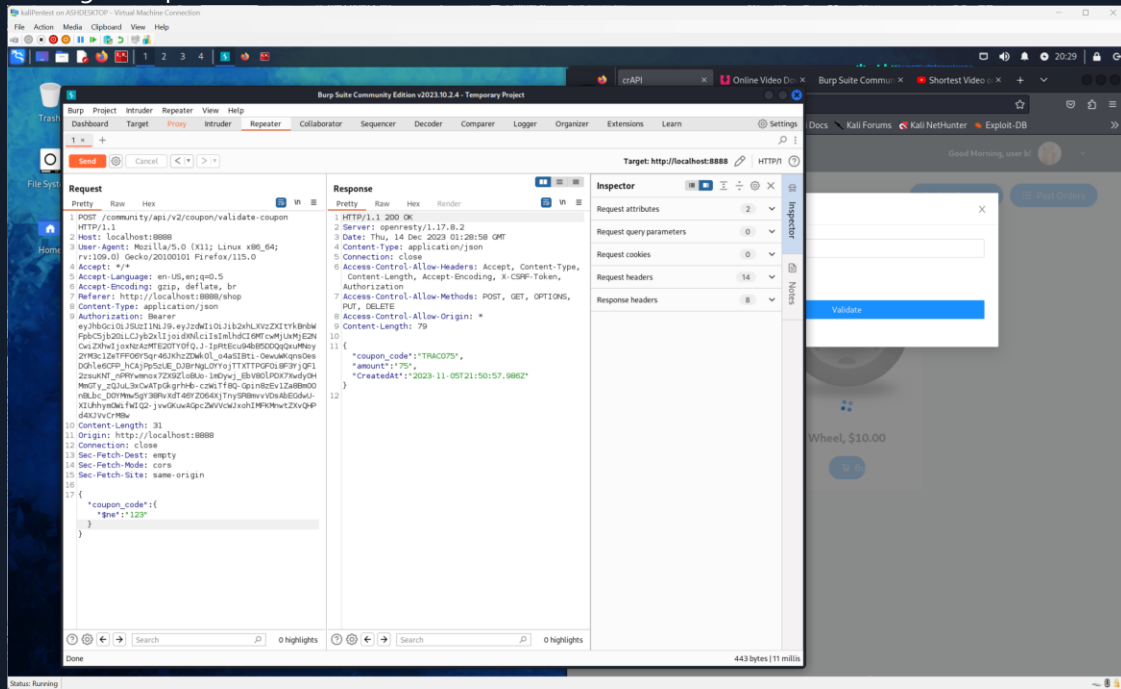
picture 26: Repeater attack (9)

which also includes manipulating the orders leading the balance to a negative one.



picture 27: Repeater attack (10)

- i. After testing the coupon section of the shop, it was determined that it's vulnerable to SQL injection leaking a coupon code for use



picture 28: SQL injection using repeater

## Remediation Summary

As a result of this assessment there are several remedies to made in order to strengthen the application server:

1. All user-controlled input must be validated for invalid characters.
2. All user-controlled input must be validated for legitimacy.
3. The API must be changed to a safer version.
4. Applying least privilege to the application accounts.
5. Disable server-side JavaScript execution.
6. Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.
7. Modern Web browsers support the Content-Security-Policy and X-Frame-Options HTTP headers. Ensure one of them is set on all web pages returned by your site/app.  
If you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. Alternatively consider implementing Content Security Policy's "frame-ancestors" directive.
8. Ensure that your web server, application server, load balancer, etc. is configured to suppress the "Server" header or provide generic details.
9. Manually confirm that the timestamp data is not sensitive, and that the data cannot be aggregated to disclose exploitable patterns.
10. Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages.  
If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.

## Technical Findings Details

CVE-ID	Description
CVE-2011-3192	The byte range filter in the Apache HTTP Server 1.3.x, 2.0.x through 2.0.64, and 2.2.x through 2.2.19 allows remote attackers to cause a denial of service (memory and CPU consumption) via a Range header that expresses multiple overlapping ranges, as exploited in the wild in August 2011, a different vulnerability than CVE-2007-0086
CVE-2007-6750	The Apache HTTP Server 1.x and 2.x allows remote attackers to cause a denial of service (daemon outage) via partial HTTP requests, as demonstrated by Slowloris, related to the lack of the mod_reqtimeout module in versions before 2.2.15.

table 3: CVE details

1. Cloud Metadata Potentially exposed: The Cloud Metadata Attack attempts to abuse a misconfigured NGINX server in order to access the instance metadata maintained by cloud service providers such as AWS, GCP and Azure.  
All of these providers provide metadata via an internal unroutable IP address '169.254.169.254' - this can be exposed by incorrectly configured NGINX servers and accessed by using this IP address in the Host header field.
2. .env Information Leak: One or more .env files seems to have been located on the server. These files often expose infrastructure or administrative account credentials, API or APP keys, or other sensitive configuration information.
3. Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.
4. The response does not include either Content-Security-Policy with 'frame-ancestors' directive or X-Frame-Options to protect against 'ClickJacking' attacks.
5. The web/application server is leaking version information via the "Server" HTTP response header. Access to such information may facilitate attackers identifying other vulnerabilities your web/application server is subject to.