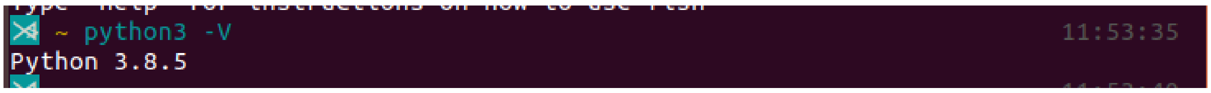
CTF 01 Salt, Capability Report

2018312567 조명하

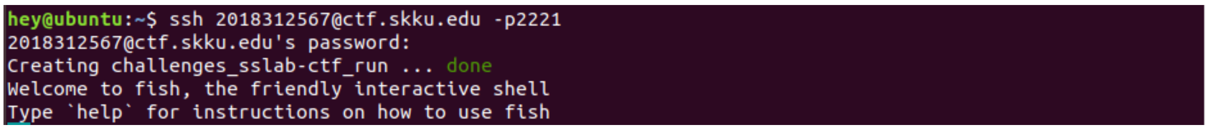
0. Environment

OS: Ubuntu 18.04. LST on VMware

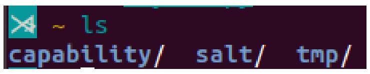
Language: Python 3.8.5


```
~ python3 -V                                                    11:53:35
Python 3.8.5
```

1. Connecting to ctf.skku.edu


```
hey@ubuntu:~$ ssh 2018312567@ctf.skku.edu -p2221
2018312567@ctf.skku.edu's password:
Creating challenges_sslab-ctf_run ... done
Welcome to fish, the friendly interactive shell
Type `help` for instructions on how to use fish
```

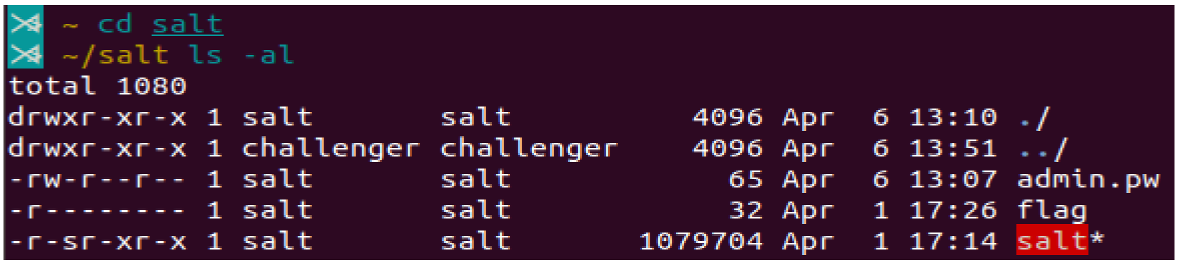
2. Checking Directory Structure


```
~ ls
capability/  salt/  tmp/
```

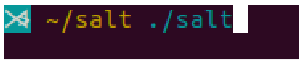
There are three directories.

3. Solving *Salt* Problem

Using command *cd*, go to '**salt/**' and check its structure

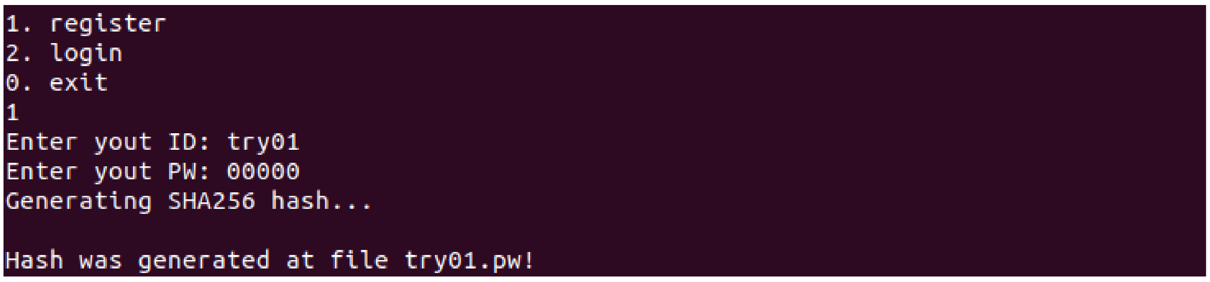

```
~ cd salt
~/salt ls -al
total 1080
drwxr-xr-x 1 salt       salt          4096 Apr  6 13:10 ./
drwxr-xr-x 1 challenger challenger    4096 Apr  6 13:51 ../
-rw-r--r-- 1 salt       salt            65 Apr  6 13:07 admin.pw
-r-------- 1 salt       salt            32 Apr  1 17:26 flag
-r-sr-xr-x 1 salt       salt       1079704 Apr  1 17:14 salt*
```

I can only execute **salt**, read **admin.pw**. No right about **flag**.



```
~/salt ./salt
```

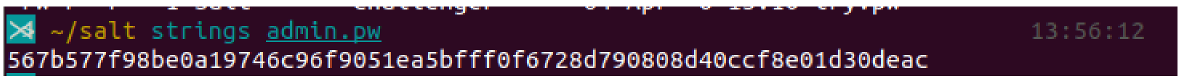
Execute **salt.** After entering the command, it prints menu.



```
1. register
2. login
0. exit
1
Enter yout ID: try01
Enter yout PW: 00000
Generating SHA256 hash...

Hash was generated at file try01.pw!
```

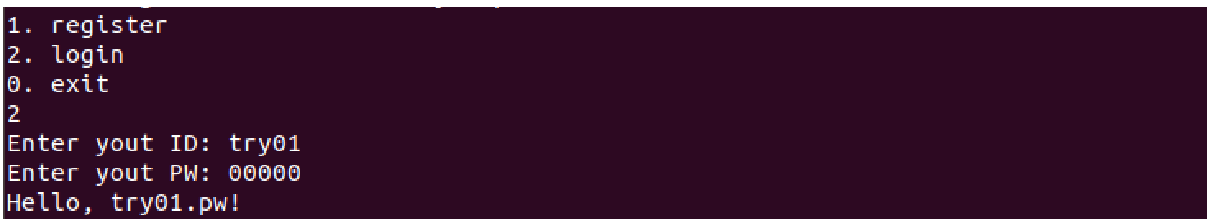
I choose **1. register** and set ID, PW as **try01, 00000**, then it makes **try01.pw**

I guess the file **admin.pw** has a hash value of the combination of admin's password and salt.



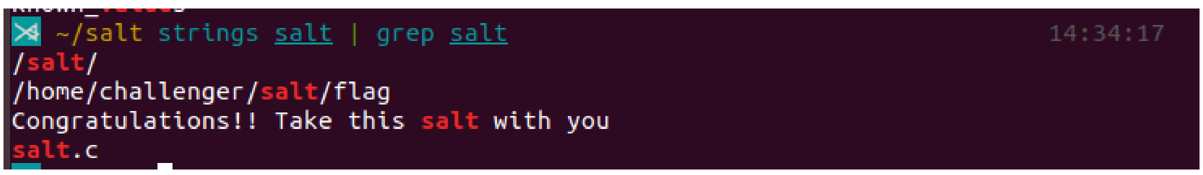
```
~/salt strings admin.pw                                              13:56:12
567b577f98be0a19746c96f9051ea5bfff0f6728d790808d40ccf8e01d30deac
```

So, using command ***strings***, I read **admin.pw**. It seems that it contains the result of h(admin's password, salt).



```
1. register
2. login
0. exit
2
Enter yout ID: try01
Enter yout PW: 00000
Hello, try01.pw!
```

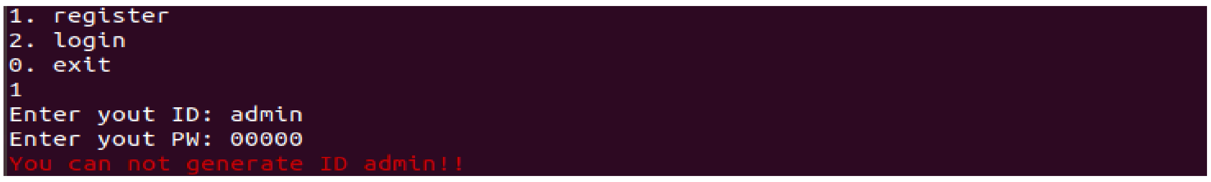
Back to the program salt, when I login as try01 and enter correct password, it prints '**Helllo, try01.pw**!' Maybe if I login as admin normally, it prints out the file flag or at least give me some useful information.



```
~/salt strings salt | grep salt                                      14:34:17
/salt/
/home/challenger/salt/flag
Congratulations!! Take this salt with you
salt.c
```

And I found a suspicious string ***"Congratulations!! Take this salt with you"***. Maybe I could get the salt value if I login as admin.

Since there are only two function, register and login in the program **salt**, I have only two options. **To register** admin's new password, or **to login** as admin.

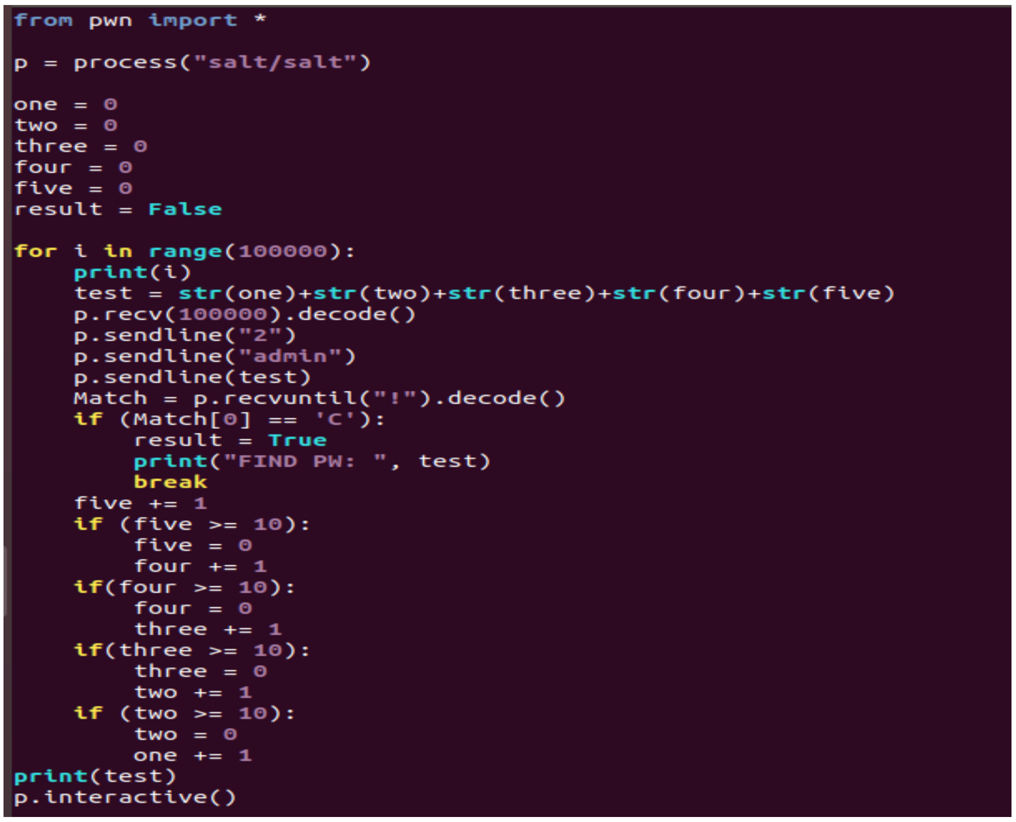


```
1. register
2. login
0. exit
1
Enter yout ID: admin
Enter yout PW: 00000
You can not generate ID admin!!
```

But when I try to register as admin, it prohibits my action. So, I have to find admin's password, and then login normally as admin.

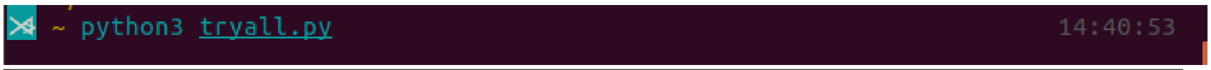
There are two hints. The password hashing algorithm uses a 4-byte salt. Admin's password is a 5-digit number.

This system saves hash value of password plus salt, and it is hard to decrypt hash value into plaintext in salt system. So, I have to try all possible passwords. To do that, I made a automation program **tryall.py** in the root directory(because I have no right to write on salt directory.)

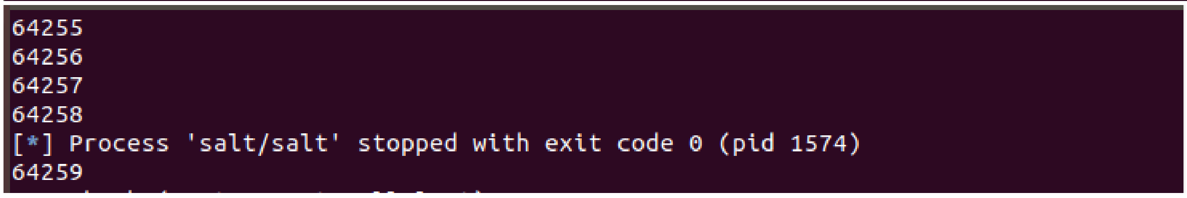


```python
from pwn import *

p = process("salt/salt")

one = 0
two = 0
three = 0
four = 0
five = 0
result = False

for i in range(100000):
    print(i)
    test = str(one)+str(two)+str(three)+str(four)+str(five)
    p.recv(100000).decode()
    p.sendline("2")
    p.sendline("admin")
    p.sendline(test)
    Match = p.recvuntil("!").decode()
    if (Match[0] == 'C'):
        result = True
        print("FIND PW: ", test)
        break
    five += 1
    if (five >= 10):
        five = 0
        four += 1
    if(four >= 10):
        four = 0
        three += 1
    if(three >= 10):
        three = 0
        two += 1
    if (two >= 10):
        two = 0
        one += 1
print(test)
p.interactive()
```

It automatically try to login as admin 100000 times, changing password from 00000 to 99999 until it finds correct password. Because I found a string starts with "Congratulations~", I guess if it finds a correct password, the program would print that string, so it stops when the program prints a string starts with "C". And print out the password tested on that time.

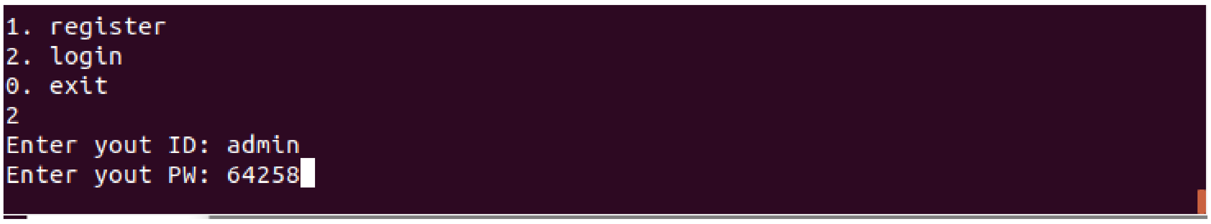


```
~ python3 tryall.py                                                14:40:53
```



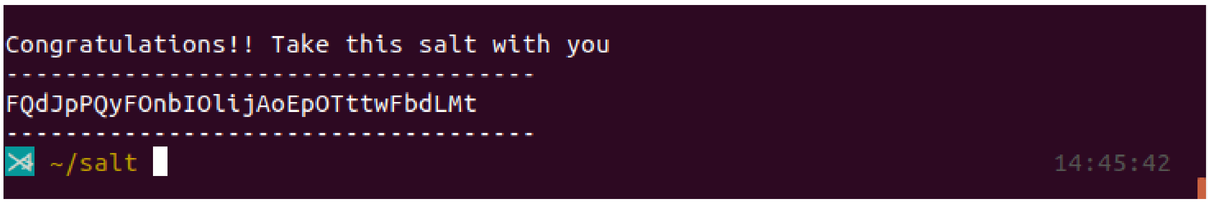
```
64255
64256
64257
64258
[*] Process 'salt/salt' stopped with exit code 0 (pid 1574)
64259
```

I execute it, and it stopped after trying **64258.**



```
1. register
2. login
0. exit
2
Enter yout ID: admin
Enter yout PW: 64258
```

So I login as **admin, 64258**

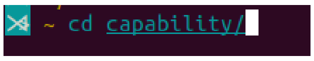


```
Congratulations!! Take this salt with you
------------------------------------
FQdJpPQyFOnbIOlijAoEpOTttwFbdLMt
------------------------------------
~/salt                                                           14:45:42
```

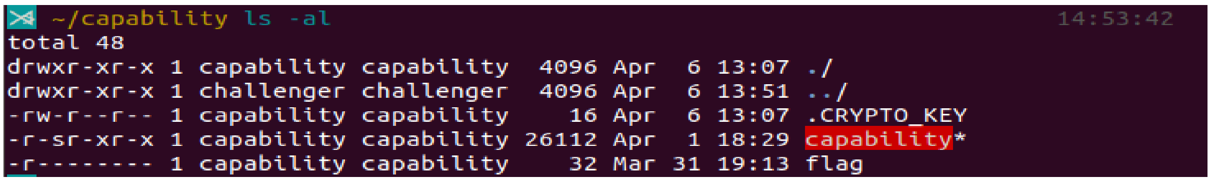
And it shows the salt value!

4. Solving Capability Problem

Go to directory capability and check its structure



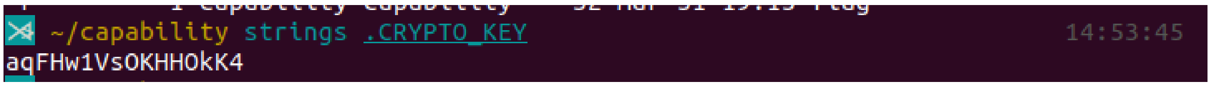
```
~ cd capability/
```



```
~/capability ls -al                                                14:53:42
total 48
drwxr-xr-x 1 capability capability  4096 Apr  6 13:07 ./
drwxr-xr-x 1 challenger challenger  4096 Apr  6 13:51 ../
-rw-r--r-- 1 capability capability    16 Apr  6 13:07 .CRYPTO_KEY
-r-sr-xr-x 1 capability capability 26112 Apr  1 18:29 capability*
-r-------- 1 capability capability    32 Mar 31 19:13 flag
```

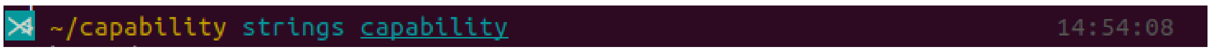
I can read and execute **capability,** can only read **.CRYPTO_KEY,** have no right on **flag**

Read **.CRYPTO_KEY** using ***strings***, but cannot understand

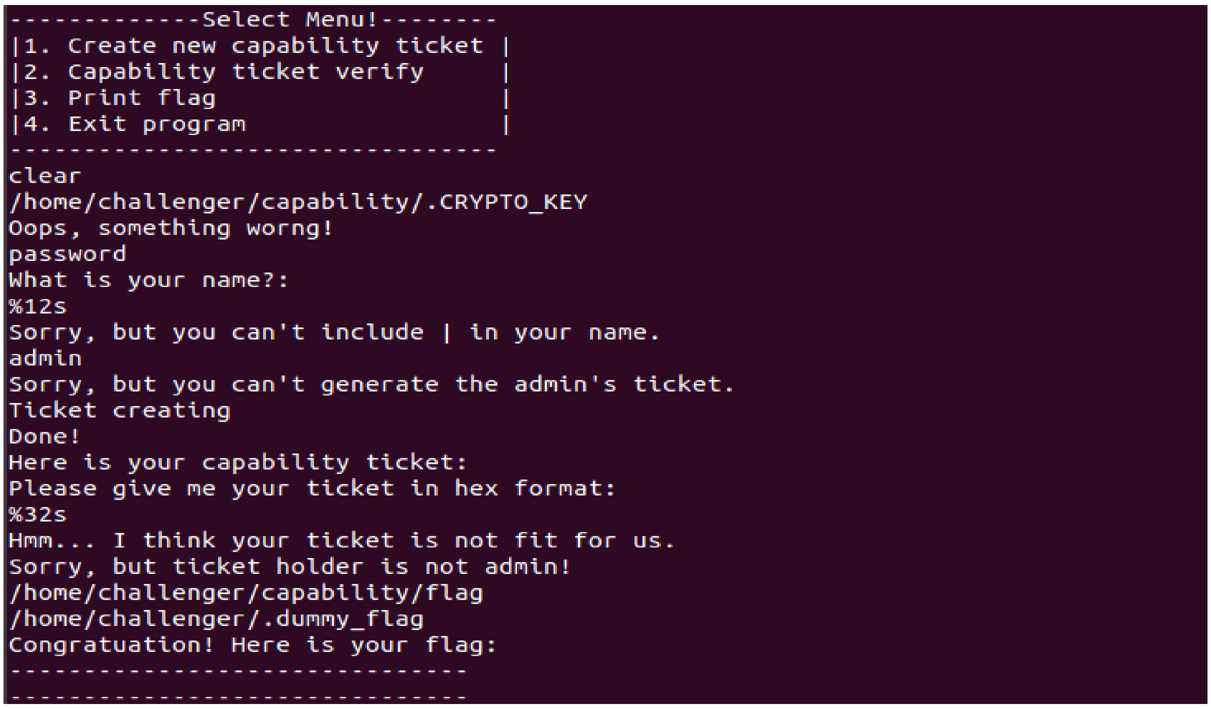


```
~/capability strings .CRYPTO_KEY                                   14:53:45
aqFHw1VsOKHHOkK4
```

Read **capability** using *strings,* and found following contents



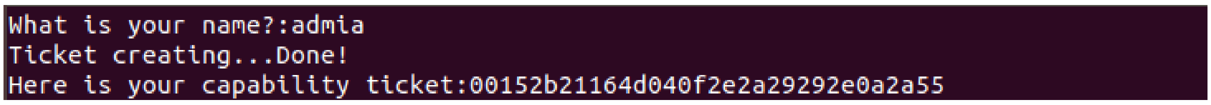
```
~/capability strings capability                                              14:54:08
```



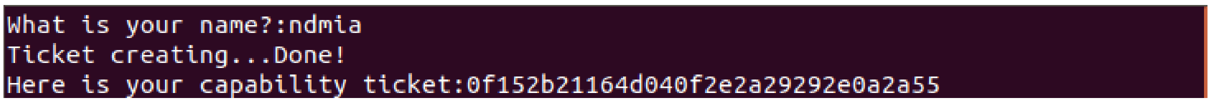
```
------------Select Menu!--------
|1. Create new capability ticket |
|2. Capability ticket verify     |
|3. Print flag                   |
|4. Exit program                 |
--------------------------------
clear
/home/challenger/capability/.CRYPTO_KEY
Oops, something worng!
password
What is your name?:
%12s
Sorry, but you can't include | in your name.
admin
Sorry, but you can't generate the admin's ticket.
Ticket creating
Done!
Here is your capability ticket:
Please give me your ticket in hex format:
%32s
Hmm... I think your ticket is not fit for us.
Sorry, but ticket holder is not admin!
/home/challenger/capability/flag
/home/challenger/.dummy_flag
Congratuation! Here is your flag:
-------------------------------
-------------------------------
```

I guess that there are four functions. It takes input string as name, and it cannot include '|' and cannot be 'admin'. If I entering normal name, it seems to create capability ticket. But if the ticket holder is not **admin**, it prohibits certain actions. So, I need to figure out how admin's ticket looks like, and verify my ticket is same as admin's ticket to get flag.

I find that 'admia', 'ndmia', 'ndmin''s capability ticket share some parts. For example, 'admia' and 'ndmia''s capability ticket are exactly same except the first byte '00' and '0f'. This means that 'admia' first character, 'a' turns into '00'.

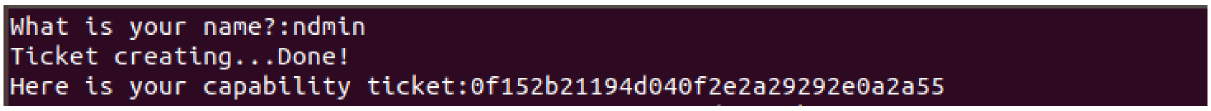


```
What is your name?:admia
Ticket creating...Done!
Here is your capability ticket:00152b21164d040f2e2a29292e0a2a55
```



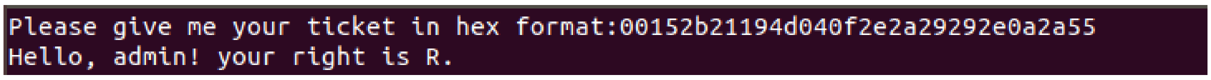
```
What is your name?:ndmia
Ticket creating...Done!
Here is your capability ticket:0f152b21164d040f2e2a29292e0a2a55
```

And 'ndmin' would be exactly same except the first byte '0f', and I can get 'admin's capability ticket by changing '0f' into '00'. So 'admin's ticket is **00152b21194d040f2e2a29292e0a2a55**



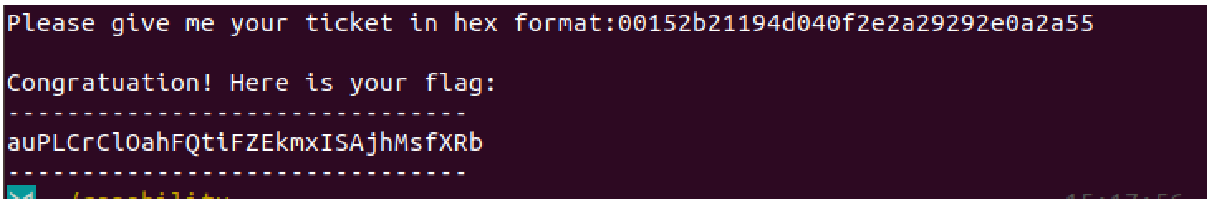
```
What is your name?:ndmin
Ticket creating...Done!
Here is your capability ticket:0f152b21194d040f2e2a29292e0a2a55
```

So in menu 2, verifying this ticket and check my right is R



```
Please give me your ticket in hex format:00152b21194d040f2e2a29292e0a2a55
Hello, admin! your right is R.
```

In menu 3, I enter admin's ticket and get my flag.



```
Please give me your ticket in hex format:00152b21194d040f2e2a29292e0a2a55

Congratuation! Here is your flag:
-------------------------------
auPLCrClOahFQtiFZEkmxISAjhMsfXRb
-------------------------------
```