I am doing this challenge with PicoCTF, that help me to solve real-life cybersecurity problems. With the basic knowledge I have acquired through my university studies. I'm not able to do a complete challenge alone. However, I'm doing this sheet to document my progess and to lead me in my path, because I hope it'll help me in the feature.

REMEMBER: the "wget" command will help you to download the file

GENERAL SKILS

10 points: python wrangling.

In this challenge we have to download three file: psw.txt flag.txt ende.py. The last one contains the code that we have to execute.

```
jack-mald-picoctf@webshell:~$ ls
README.txt  ende.py  flag.txt.en  warm
jack-mald-picoctf@webshell:~$ python ende.py -d flag.txt.en
Please enter the password:ac9bd0ffac9bd0ffac9bd0ffac9bd0ff
picoCTF{4p0110_1n_7h3_h0us3_ac9bd0ff}
jack-mald-picoctf@webshell:~$
```

So as you can see the program was encrypted, we just need to type "python [nomefile] -d [filetxt]" to decrypt.
At the end I we have put in the terminal the password and we have finish.

15 points: nice netcat

first thing we should do is to learn something about netcat.
It allows users to create and manipulate network connections. To connect you have to type " nc [site] [port]"

```
jack-mald-picoctf@webshell:~$ nc mercury.picoctf.net 21135
112
105
99
111
67
84
70
123
103
48
48
100
95
107
49
116
116
121
33
95
110
49
99
51
95
107
49
116
116
121
33
95
97
102
100
53
```

As you can see there are many numbers in the terminal. These numbers are ASCII code that once converted give you the flag.

FLAG: picoCTF{g00d_k1tty!_n1c3_k1tty!_afd5fda4}

20 points: static ain't always noise

This time we have to download two file: ltdis.sh and static. What is a file with .sh extension? It is a shell script that is files that contain a series of commands or instructions that are meant to be executed by a unix-

like shell. To be run, you have to change the permissions using “chmod +x [file.sh]” and after remember that it is not a normal file therefore to execute it you have to use “./file.sh static”.

```
jack-mald-picoctf@webshell:~$ chmod +x ltdis.sh
jack-mald-picoctf@webshell:~$ ./ltdis.sh
Attempting disassembly of  ...
objdump: 'a.out': No such file
objdump: section '.text' mentioned in a -j option, but not found in any input file
Disassembly failed!
Usage: ltdis.sh <program-file>
Bye!
jack-mald-picoctf@webshell:~$ ./ltdis.sh static
Attempting disassembly of static ...
Disassembly successful! Available at: static.ltdis.x86_64.txt
Ripping strings from binary with file offsets...
Any strings found in static have been written to static.ltdis.strings.txt with file offset
jack-mald-picoctf@webshell:~$ ls
README.txt  ende.py  flag.txt.en  ltdis.sh  static  static.ltdis.strings.txt  static.ltdis.x8
6_64.txt  warm
```

First of all, as you can see now there is another file named “static.ltdis.strings.txt” and in the end if we watch what there is inside you can find the flag.

```
jack-mald-picoctf@webshell:~$ cat static.ltdis.strings.txt
    238 /lib64/ld-linux-x86-64.so.2
    361 libc.so.6
    36b puts
    370 __cxa_finalize
    37f __libc_start_main
    391 GLIBC_2.2.5
    39d _ITM_deregisterTMCloneTable
    3b9 __gmon_start__
    3c8 _ITM_registerTMCloneTable
    660 AWAVI
    667 AUATL
    6ba []A\A]A^A_
    6e8 Oh hai! Wait what? A flag? Yes, it's around here somewhere!
    7c7 ;*3$"
   1020 picoCTF{d15a5m_t34s3r_6f8c8200}
```

“6e8 -> amusing”.

picoCTF{d15a5m_t34s3r_6f8c8200}

20 points: tab, tab, attack

This was truly easy. Why? You only have to download a file named “Addadshashanammu.zip”. However, when you unzip, it has a lot of subdirectories so you can type the main directory and type tab.
In the end execute the file, and you have already done.

```
jack-mald-picoctf@webshell:~/Addadshashanammu/Almurbalarammi/Ashalmimilkala/Assurnabitashpi/Maelkashishi/Onnissiralis/Ularradallaku$ ./fang-of-haynekhtnamet
*ZAP!* picoCTF{l3v3l_up!_t4k3_4_r35t!_2bcfb2ab}
jack-mald-picoctf@webshell:~/Addadshashanammu/Almurbalarammi/Ashalmimilkala/Assurnabitashpi/Maelkashishi/Onnissiralis/Ularradallaku$
```

picoCTF{l3v3l_up!_t4k3_4_r35t!_2bcfb2ab}

30 points: magikarp ground mission

This challenge is focused on the use of the ssh, but what is it?
It is a network protocol that allows secure communication and remote access to a computer or server over an unsecured network. You use “ssh ctf-player@venus.picoctf.net -p 56093” to connect. Once you are

connected you can move in the directory by using “cd” and hey look the flag is in the file, but is fragmented.

```
ctf-player@pico-chall$ ls
1of3.flag.txt  instructions-to-2of3.txt
ctf-player@pico-chall$ cat instrutions-to-2of3.txt
cat: instrutions-to-2of3.txt: No such file or directory
ctf-player@pico-chall$ cat instructions-to-2of3.txt
Next, go to the root of all things, more succinctly `/`
ctf-player@pico-chall$ cat 1of3.flag.txt
picoCTF{xxsh_
ctf-player@pico-chall$ cd ..
ctf-player@pico-chall$ ls
3of3.flag.txt  drop-in
ctf-player@pico-chall$ cat3of3.flag.txt
-bash: cat3of3.flag.txt: command not found
ctf-player@pico-chall$ cat 3of3.flag.txt
71be5264}
ctf-player@pico-chall$ ^C
ctf-player@pico-chall$ cd ..
ctf-player@pico-chall$ ls
ctf-player
ctf-player@pico-chall$ cat ctf-player
cat: ctf-player: Is a directory
ctf-player@pico-chall$ cd ctf-player
ctf-player@pico-chall$ ls
3of3.flag.txt  drop-in
ctf-player@pico-chall$ ./drop-in
-bash: ./drop-in: Is a directory
ctf-player@pico-chall$ cd drop-in
ctf-player@pico-chall$ ls
1of3.flag.txt  instructions-to-2of3.txt
ctf-player@pico-chall$ cd /
ctf-player@pico-chall$ ls
2of3.flag.txt  bin  boot  dev  etc  home  instructions-to-3of3.txt  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
ctf-player@pico-chall$ cat 2of3.flag.txt
0ut_0f_\/\/4t3r_
```

50 points: warmed up

This challenge was so easy, I didn’t even want to write here. But there was 0x3D in hexadecimal notation and we had to convert it in decimal notation.

3* 16^1 + 13*16^0 = 61

picoCTF{61}

100 points: Strings it

This challenge was a bit different. You have to use the command strings.

Strings is used to extract sequences of readable characters from a binary file. Strings [options][file].

In my case I already know that the flags start with the letter ‘p’ therefore I have only to type this command strings eseguibile | grep '^P'

```
jack-mald-picoctf@webshell:~$ strings strings | grep '^p'
puts
pp6DsuJVD5M8STb8BSUZD2WSVewAXSjZYjyumV
pIXGqPUtqu4pxcNHoXWBy7n9uWM2UJuebb
przTtfKvZym9bc5B1JmXAFhFLYvPdnRnYLuzWMNAy
pUMovwdp4wpMmvwlSqnT3gapyXKF8KPNHejx82j9myXTfeLSkIQSkQ
phEzb8kS0VIhwY5Gp3wLvpwApSbzQtURt7
pxMo1Olkcw408UMwL6Kbh2hEM0cpMQ4WgKRJd
porwcE4TIOlo53PYX3cUqXHUD03
pHRT16yNG8IxJ5foc
p6t2ZJktd5lHl8rYGnPxY3tCDqfZvGAPpK2W6ZrLMxS
pdg7hCaaiBxgdm0oMQXKqt9y29zpRoOrRE6MIjVzQfWs1IGMVyzjON8bqxov7bHTih6r4ie
prYP28X2BEARBVWS6Lk8L2ksGdRjKGReQ1qd6nByDmJ8
porRtniEgFayrNSCLQbbvtJtpIykCVrds4e
pOhuwxlgWM7gEWd8NgOVcpb
psr8JL5UuhECQGkVVJcF0u54EKjTBVlc9y7
pfRkjb68Vge04EKTBb42Rg7AbHRIHDyErkrZ1FDIpTfH
phLSMjSYqeXOn5pA8aHPnkb0
pT8t2Ui1HEQvt354WfZqCTqZb3cZbNZsOKmCOzqk
pilK6Px81zXHdYDjsx6Qei7BmXMcUdCm3fanxhnvgRQY
pl86ocIGnjGW813fiWRHDvE
p8QbwedTix8DAzsni1GWWzv698tGmLWRDzhVjct
pmdzNnUD9WUkC5q2ZUhStPJ2IOot4vUZhNiO0CutubRONb
p20S58xogm6L1CPIoEAwfDpythy5l5ZZuudUB2CFg8fLg
pTdvcL3r4LHWekLuY
prpxuqs6om72AfZLw8OL6nRaHWhm6ONLP2KBZGeRCFf
poelAXXRI2jD3DLVzfxyckgQVIPtTBPhIus
pjnXtMoo3aB5TTImUyhOIQVQ86ab
pqZL5A8nGfASPPEjk5YVragK27
pQRhLMSCP712TwOcD6onzlpsuw
picoCTF{5tRIng5_1T_d66c7bb7}
```

100 points: Bases

To do this chellenge you have to now what is base 64 is a way to encrypt msg. So to decript out msg we ha to use the command base64 –decode

```
jack-mald-picoctf@webshell:~$ echo "bDNhcm5fdGgzX3IwcDM1" | base64 --decode
l3arn_th3_r0p35jack-mald-picoctf@webshell:~$
```

PicoCTF{l3arn_th3_r0p35}

100 points: First Grep

To do this chellenge you have to use grep like we had used it before. We have a file named "file" (what an original name). So, to search in this file you have to digit this command "grep 'picoCtf' file"

```
jack-mald-picoctf@webshell:~$ grep "picoCTF" file
picoCTF{grep_is_good_to_find_things_f77e0797}
jack-mald-picoctf@webshell:~$
```

100 points: Codebook

We have the same challenge that we have had before. However we have two file, one is a python file and the second is a txt file.

Use theis command and it should be easy "python code.py -d codebook.txt"

```
jack-mald-picoctf@webshell:~$ cat codebook.txt
azbycxdwevfugthsirjqkplomn
jack-mald-picoctf@webshell:~$ python code.py -d codebook.txt
picoCTF{c0d3b00k_455157_d9aa2df2}
```

100 points: convertme.py

For this channel we have only to create a new file with the command "cat > flag.txt" and after use the command of the challenge above. Look at the photo :)

```
jack-mald-picoctf@webshell:~$ cat > flag.txt
ls^C
jack-mald-picoctf@webshell:~$ ls
Addadshashanammu  README.txt  convertme.py  flag.txt
jack-mald-picoctf@webshell:~$ cat flag.txt
jack-mald-picoctf@webshell:~$ python convertme.py -d flag.txt
If 53 is in decimal base, what is it in binary base?
Answer: 110101
That is correct! Here's your flag: picoCTF{4ll y0ur b4535 722f6b39}
```

100 points: fixme1.py

This challenge give you a program that has a problem that you must find. The problem is the indentation of the line to change it use "nano fixme1.py" save the change and run the code.

```
jack-mald-picoctf@webshell:~$ python fixme1.py -d flag.txt
  File "/home/jack-mald-picoctf/fixme1.py", line 20
    print('That is correct! Here\'s your flag: ' + flag)
IndentationError: unexpected indent
jack-mald-picoctf@webshell:~$ nano fixme1.py
jack-mald-picoctf@webshell:~$ python fixme1.py -d flag.txt
That is correct! Here's your flag: picoCTF{1nd3nt1ty_cr1515_09ee727a}
```

100 points: glitchcat

This challenge give you this code "$ nc saturn.picoctf.net 55826" you have only to connect and decript the flag that I show you below

```
jack-mald-picoctf@webshell:~$ nc saturn.picoctf.net 55826
'picoCTF{gl17ch_m3_n07_' + chr(0x39) + chr(0x63) + chr(0x34) + chr(0x32) + chr(0x61) + chr(0x34) + chr(0x35) + chr(0x64) + '}'
```

100 points: PW Crack 1

This challenge is a little bit thrilled than the other. When u run the python file it request you a psw, Anyway the psw, that you obv don't know, is in the code of the python file.

```
 Rr1wQ  nVT_nPRVWjack-mald-picoctf@webshell:~$ cat level1.py
### THIS FUNCTION WILL NOT HELP YOU FIND THE FLAG --LT ########################
def str_xor(secret, key):
    #extend key to secret length
    new_key = key
    i = 0
    while len(new_key) < len(secret):
        new_key = new_key + key[i]
        i = (i + 1) % len(key)
    return "".join([chr(ord(secret_c) ^ ord(new_key_c)) for (secret_c,new_key_c) in zip(secret,new_key)])
#############################################################################


flag_enc = open('level1.flag.txt.enc', 'rb').read()



def level_1_pw_check():
    user_pw = input("Please enter correct password for flag: ")
    if( user_pw == "1e1a"):
        print("Welcome back... your flag, user:")
        decryption = str_xor(flag_enc.decode(), user_pw)
        print(decryption)
        return
    print("That password is incorrect")



level_1_pw_check()
jack-mald-picoctf@webshell:~$ python level1.py -d level1.flag.txt.enc
Please enter correct password for flag: A
That password is incorrect
jack-mald-picoctf@webshell:~$ cat level1.flag.txt.enc
A
 Rr1wQ  nVT_nPRVWjack-mald-picoctf@webshell:~$ python level1.py -d level1.flag.txt.enc
Please enter correct password for flag: 1e1a
Welcome back... your flag, user:
picoCTF{545h_r1ng1ng_fa343060}
jack-mald-picoctf@webshell:~$
```

100 points: Serpentine

From now the challenge seems a little bit funnier.

When you run the python file you look three choice but anyone of these are correct to print the flag. So you have to look the code and to find the solution. In this case the solution is to add to the choice b the function "print_flag"

```
jack-mald-picoctf@webshell:~$ python serpentine.py -d flag.txt

    Y
  .-^-.
 /     \      .- ~ ~ -.
()     ()    /   _ _   `.                    _ _ _
 \_   _/    /  /     \   \                 . ~  _ _  ~ .
   | |     /  /       \   \              .' .~       ~-. `.
   | |    /  /         )   )            /  /             `.`.
   \ \_ _/  /         /   /            /  /                `'
    \_ _ _.'         /   /            (  (
                    /   /             \  \
                   /   /               \  \
                  /   /                 )  )
                 (   (                 /  /
                  `.  `.              .'  /
                    `.   ~ - - - - ~   .'
                       ~ . _ _ _ _ . ~

Welcome to the serpentine encourager!


a) Print encouragement
b) Print flag
c) Quit

What would you like to do? (a/b/c) b

Oops! I must have misplaced the print_flag function! Check my source code!


picoCTF{7h3_r04d_l355_7r4v3l3d_8e47d128}
a) Print encouragement
b) Print flag
c) Quit
```

100 points: First Find

Here you have a zip file and you have to unzip it. Now you have to find a file named "uber-secret.txt". To complete this challenge write this command "`find / -type f -name "uber-secret.txt"`" (it's black

because I want you to remember it)

```
jack-mald-picoctf@webshell:~/files$ find / -type f -name "uber-secret.txt"
find: '/etc/polkit-1/localauthority': Permission denied
find: '/etc/ssl/private': Permission denied
/home/jack-mald-picoctf/files/adequate_books/more_books/.secret/deeper_secrets/deepest_secrets/uber-secret.txt
find: '/proc/tty/driver': Permission denied
```

100 points: Big Zip

Here you have a big zip with a lot of subdirectories and file.txt. I use this command to search in all the directories. ‘grep -r 'picoCTF' /’ this command search recursively in all the files.

```
jack-mald-picoctf@webshell:~/big-zip-files$ grep -r 'picoCTF' /
grep: /dev/console: Permission denied
grep: /etc/.pwd.lock: Permission denied
grep: /etc/default/cacerts: Permission denied
grep: /etc/gshadow: Permission denied
grep: /etc/profile.d/debuginfod.csh: Permission denied
grep: /etc/profile.d/debuginfod.sh: Permission denied
grep: /etc/security/opasswd: Permission denied
grep: /etc/shadow: Permission denied
grep: /etc/gshadow-: Permission denied
grep: /etc/shadow-: Permission denied
grep: /etc/polkit-1/localauthority: Permission denied
grep: /etc/ssl/private: Permission denied
/home/jack-mald-picoctf/README.txt:Welcome to the picoCTF webshell!
/home/jack-mald-picoctf/README.txt:picoCTF challenges.
/home/jack-mald-picoctf/README.txt:  Extensive brute-forcing is not necessary to solve picoCTF challenges.
/home/jack-mald-picoctf/README.txt:- If you change your username through the picoCTF website, you will
grep: /home/jack-mald-picoctf/Addadshashanammu/Almurbalarammi/Ashalmimilkala/Assurnabitashpi/Maelkashishi/Onnissiralis/Ularradallaku/fang-of-haynekhtnamet: binary file matches
grep: /home/jack-mald-picoctf/files.zip: binary file matches
/home/jack-mald-picoctf/files/adequate_books/more_books/.secret/deeper_secrets/deepest_secrets/uber-secret.txt:picoCTF{f1nd_15_f457_ab443fd1}
/home/jack-mald-picoctf/big-zip-files/folder_pmbymkjcya/folder_cawigcwvgv/folder_ltdayfmktr/folder_fnpfclfyee/whzxrpivpqld.txt:information on the record will last a billion years. Genes and brains and books encode picoCTF{g
r3p_15_m4g1c_ef8790dc}
```

100 points: chrono

You have to luch a ssh connection to their server and after you have to move in the file to find the flag. But it was easy if you use the command that we had used above. ‘grep -r 'picoCTF' /’

```
picoplayer@challenge:/$ grep -r 'picoCTF' /
grep: /etc/.pwd.lock: Permission denied
grep: /etc/gshadow: Permission denied
grep: /etc/security/opasswd: Permission denied
grep: /etc/shadow: Permission denied
grep: /etc/ssh/ssh_host_ecdsa_key: Permission denied
grep: /etc/ssh/ssh_host_ed25519_key: Permission denied
grep: /etc/ssh/ssh_host_rsa_key: Permission denied
grep: /etc/ssh/ssh_host_dsa_key: Permission denied
/etc/crontab:# picoCTF{Sch3DUL7NG_T45K3_L1NUX_d83baed1}
```

100 points: permission

This challenge was few difficult. Firstly you have to connect via ssh.
But after you have to see the permission that you have, “ls -l” and on the root directory you have not permission to read it“drwx------”. However you can write “sudo -l” to view all the permissions that you have. You have all the permissions in the “/usr/bin/vi”. So write “sudo /usr/bin/vi” and vim will open.
When vim is open write “:! id” and how you can see all the permissions are abled for you. So move in the root file and after read the flag.

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

User picoplayer may run the following commands on challenge:
    (ALL) /usr/bin/vi
picoplayer@challenge:/$ sudo /usr/bin/vi

uid=0(root) gid=0(root) groups=0(root)

Press ENTER or type command to continue
total 0
drwxr-xr-x   1 root   root     51 Sep 30 20:14 .
drwxr-xr-x   1 root   root     51 Sep 30 20:14 ..
-rwxr-xr-x   1 root   root      0 Sep 30 20:14 .dockerenv
lrwxrwxrwx   1 root   root      7 Mar  8  2023 bin -> usr/bin
drwxr-xr-x   2 root   root      6 Apr 15  2020 boot
d---------   1 root   root     27 Aug  4 21:35 challenge
drwxr-xr-x   5 root   root    340 Sep 30 20:14 dev
drwxr-xr-x   1 root   root     66 Sep 30 20:14 etc
drwxr-xr-x   1 root   root     24 Aug  4 21:32 home
lrwxrwxrwx   1 root   root      7 Mar  8  2023 lib -> usr/lib
lrwxrwxrwx   1 root   root      9 Mar  8  2023 lib32 -> usr/lib32
lrwxrwxrwx   1 root   root      9 Mar  8  2023 lib64 -> usr/lib64
lrwxrwxrwx   1 root   root     10 Mar  8  2023 libx32 -> usr/libx32
drwxr-xr-x   2 root   root      6 Mar  8  2023 media
drwxr-xr-x   2 root   root      6 Mar  8  2023 mnt
drwxr-xr-x   2 root   root      6 Mar  8  2023 opt
dr-xr-xr-x 2349 nobody nogroup   0 Sep 30 20:14 proc
drwx------   1 root   root     23 Aug  4 21:35 root
drwxr-xr-x   1 root   root     66 Sep 30 20:14 run
lrwxrwxrwx   1 root   root      8 Mar  8  2023 sbin -> usr/sbin
drwxr-xr-x   2 root   root      6 Mar  8  2023 srv
dr-xr-xr-x  13 nobody nogroup   0 Sep 30 20:14 sys
```

```
                    VIM - Vi IMproved

                    version 8.1.3741
                 by Bram Moolenaar et al.
       Modified by team+vim@tracker.debian.org
     Vim is open source and freely distributable

            Sponsor Vim development!
type  :help sponsor<Enter>    for information

type  :q<Enter>               to exit
type  :help<Enter>  or  <F1>  for on-line help
type  :help version8<Enter>   for version info
:! cat root/.flag.txt
```

100 points: repetitions

In this challenge you have a base34 phrase and you have to decrypt it.
Use the following code “base64 -d enc_flag | base64 -d | base64 -d | base64 -d | base64 -d | base64 -d” in this necessary to repeat as you can see.

100 points: useless

This challenge steal me a lot of time because you have to run a programm called useless but when you write “./useless” the programm tell you that you have to read the code. In the code there is a specif lines that tell u to read the manual. But as I remember there is a command called man in linux. So I write man useless and the flag war there.

```
picoplayer@challenge:~$ ./useless div 1 2
The quotient is: 0
picoplayer@challenge:~$ ./useless mul 1 2
The product is: 2
picoplayer@challenge:~$ ./useless m 1 2
Read the manual
picoplayer@challenge:~$ ./useless div 4 0
./useless: line 20: 4 / 0 : division by 0 (error token is "0 ")
picoplayer@challenge:~$ man useless

useless
     useless, -- This is a simple calculator script

SYNOPSIS
     useless, [add sub mul div] number1 number2

DESCRIPTION
     Use the useless, macro to make simple calulations like addition,subtraction, multiplication and division.

Examples
     ./useless add 1 2
       This will add 1 and 2 and return 3

     ./useless mul 2 3
       This will return 6 as a product of 2 and 3

     ./useless div 6 3
       This will return 2 as a quotient of 6 and 3

     ./useless sub 6 5
       This will return 1 as a remainder of substraction of 5 from 6

Authors
     This script was designed and developed by Cylab Africa
```

200 points: plumbing

In this thrilled challenge you have to learn the use of the pipe operator, but he didn't know that we have already had it. When you connect via nc you use a program is automatically lunch so you have to write this command to find in this large output "nc jupiter.challenges.picoctf.org 14291 | grep 'picoCTF'".

```
Again, I really don't think this is a flag
This is defintely not a flag
Again, I really don't think this is a flag
This is defintely not a flag
Again, I really don't think this is a flag
Not a flag either
Again, I really don't think this is a flag
This is defintely not a flag
^C
jack-mald-picoctf@webshell:~$ nc jupiter.challenges.picoctf.org 14291 | grep 'picoCTF'
picoCTF{digital_plumb3r_ea8bfec7}
```

300 points: flag_shop

This is the best challenge I've done so far. initially you have to connect via netcat then you will run a program and you have to understand how to get the flag.
The program focuses on the representation of integers.
Int numbers in c have a representability of -2147483648 to 2147483647 so you just need to go beyond this positive range to go into negative numbers (you have to see the challenge and the source code to understand).

```
These knockoff Flags cost 900 each, enter desired quantity
2386192

The final cost is: -2147394496

Your current balance after transaction: 2147395596

Welcome to the flag exchange
We sell flags

1. Check Account Balance

2. Buy Flags

3. Exit

 Enter a menu selection
2
Currently for sale
1. Defintely not the flag Flag
2. 1337 Flag
2
1337 flags cost 100000 dollars, and we only have 1 in stock
Enter 1 to buy one1
YOUR FLAG IS: picoCTF{m0n3y_bag5_68d16363}
Welcome to the flag exchange
We sell flags
```

SUMMARY

I am going to summarize what I've learned.

Python

This command is used to run python code.
SYNTAX: python [filename] -d [file.txt]
The instruction running a python file with a option -d to decrypt the file and put the flag in the file.txt

Netcat

Is used for involving TCP, UDP conncetions. It deals with both IPv4 and IPv6.
SYNTAX: nc [nomesite] [port]
SYNTAX: echo "file_dainviare" | nc localhost 2220

Chmod

Is used to modify the access permissions of files and directories. This determine who can read, write or execute a file or directory.
SYNTAX: ls -l (to see the permission that you have)
SYNTAX: chmod [options] mode file/directory
[options] -> 0: No permission, 1: Execute, 2: Write, 4: Read (you can add this permissions)
chmod 755 file.txt
-> u (owner), g (group), o (others), a (all) with + to add permissions or - to remove.
chmod a+x file.txt

Ssh

Allow secure communication over an unsecured network
SYNTAX: ssh username@remote_server_ip -p 1234

Grep

Recursive search: **SYNTAX:** grep -r 'pattern' directory (if u put / starts from the root directory)
Search a pattern in multiple files: **SYNTAX**: grep 'pattern' file1 file 2 file 3
SYNTAX: grep -v "pattern" does not find "pattern"
You can use the regular expression too

Find

Search files and directories.
SYNTAX: find [path] [expression]
find [/] [- type f] [-name "*.txt"]
[-type d] [-size +100k]
[! -executable]
[-user/-group]

Create file

1. touch filename
2. cat > filename
3. Nano filename; vi filename; vim filename; gedit filename;
4. Echo "content" >> filename

Create directory

SYNTAX: mkdire nomedirectory

Copiare file

SYNTAX: cp file [path_new_directory]

Rinomina file

SYNTAX: mv vecchio_nome.txt nuovo_nome.txt

Vim

In vim you can also write a command by typing ":!"

Base 64

Is a encoding scheme that represents binary data in an ASCII string format

SYNTAX: base64 [options] [filename]

base64 -d enc_flag

Ls

1. To see files and directories hidden

SYNTAX: ls -a

Cat

1. To read a file with spaces

SYNTAX: cat "file with spaces"

2. To read file that starts with "-"

SYNTAX: cat ./-file00

3. To create SYNTAX: cat > file.txt To read

File

Determine the type of a file.

SYNTAX: file [options] filename

file a1 a2 a3

Sort

Search a line that is unique (occurs only once)

SYNTAX: sort [file.txt] | uniq -u

Strings

This command allows you to read a file that is not legible

XXD

Allow you to do a conversion from hexadecimal to binary

SYNTAX: xxd -r new_data.txt > data

UNZIP

SYNTAX: (.gz) gzip -d file.gz

SYNTAX: (.bz2) bzip2 -d file.bz2

SYNTAX: (.tar) tar xf file.tar

OPENSSL

You can connect and read the information about the connection

SYNTAX: openssl s_client -connect host:port

NMAP

Network scanning and host discorvery.

SYNTAX: nmap -p 3000-3100 localhost

DIFF

Compare two files line by line and show the differences between them.

SYNTAX: diff file1.txt file2.txt