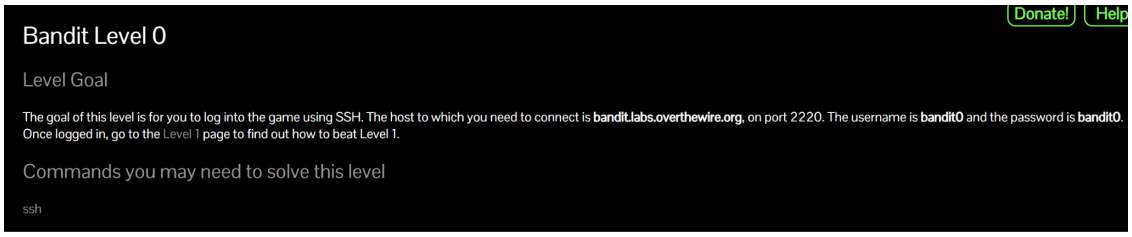
OverTheWire-Bandit

Bayu Putra Ibana

Level 0

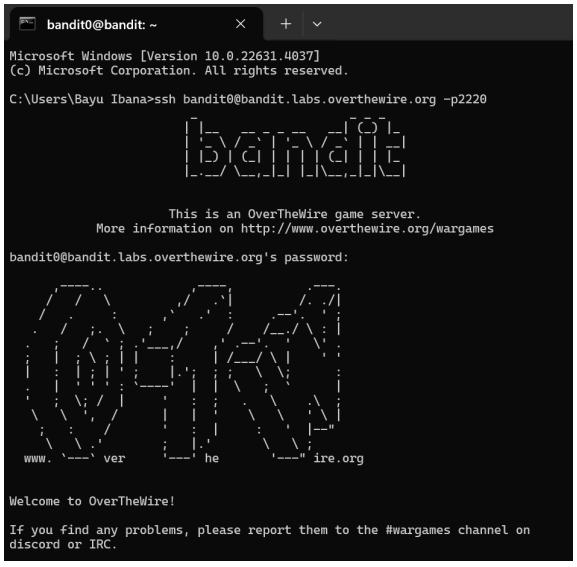


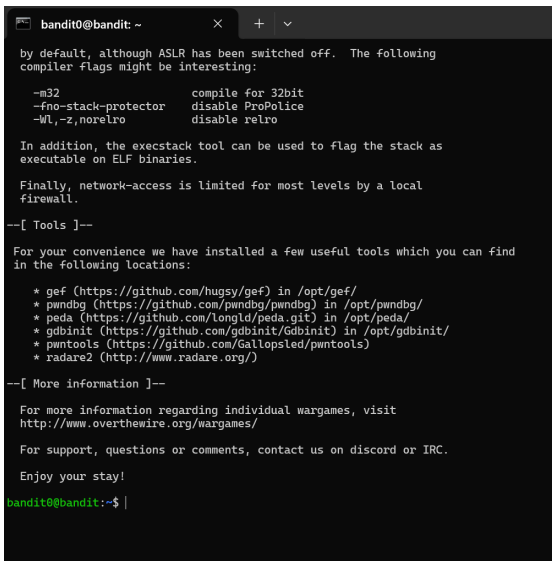
Step 1: Open the Powershell

Step 2: Connect to bandit.labs.overthewire.org on port 2220 then enter username bandit0

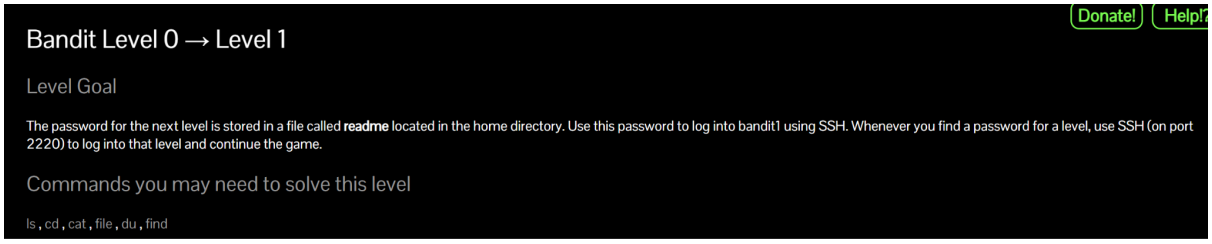
```
ssh bandit0@bandit.labs.overthewire.org -p2220
```

with the password (bandit0)

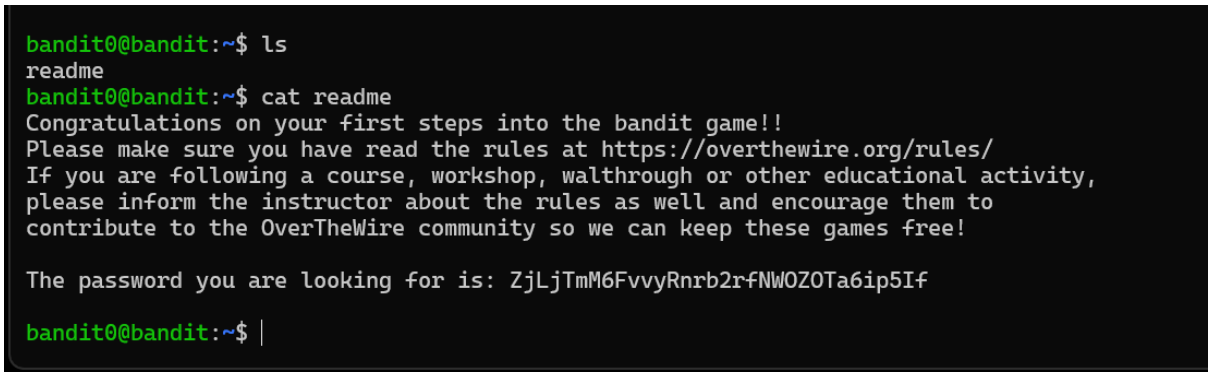




Level 0-1



Step 1 :



Type

“ls”

This command is used to view the list of the contents of the directory.

Step 2:There is a readme file in the directory, type the

“cat readme “

command, which is used to display the content of files in the terminal, in this case we are displaying the contents of the “readme” file which contains the password for the next level.

The Password is => **ZjLjTmM6FvvyRnrb2rfNWOZOTa6ip5If**

Level 1-2

Bandit Level 1 → Level 2

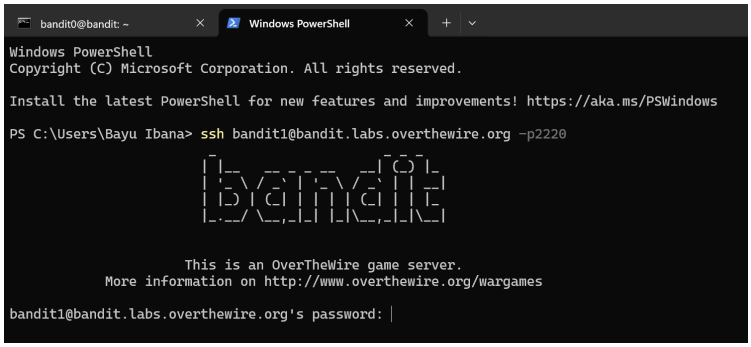
Level Goal

The password for the next level is stored in a file called - located in the home directory

Commands you may need to solve this level

ls , cd , cat , file , du , find

Step 1:

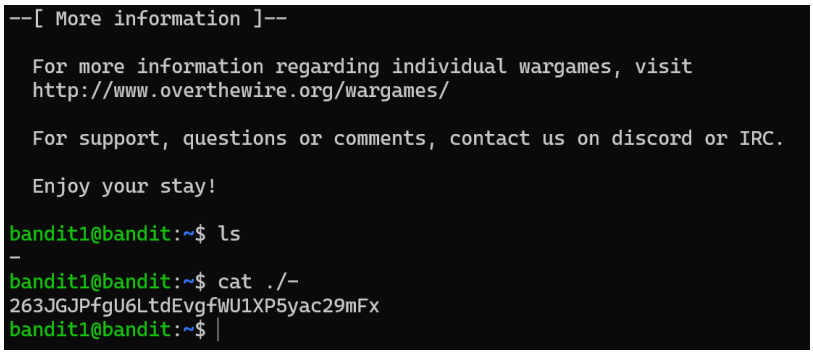


Open another tab in the powershell and type:

```
ssh bandit1@bandit.labs.overthewire.org -p2220
```

then enter the password that we got from the last level to continue. This step is done repeatedly every time we enter a new level.

Step 2



Type

`“ls”`

the file “-” will be displayed in the directory

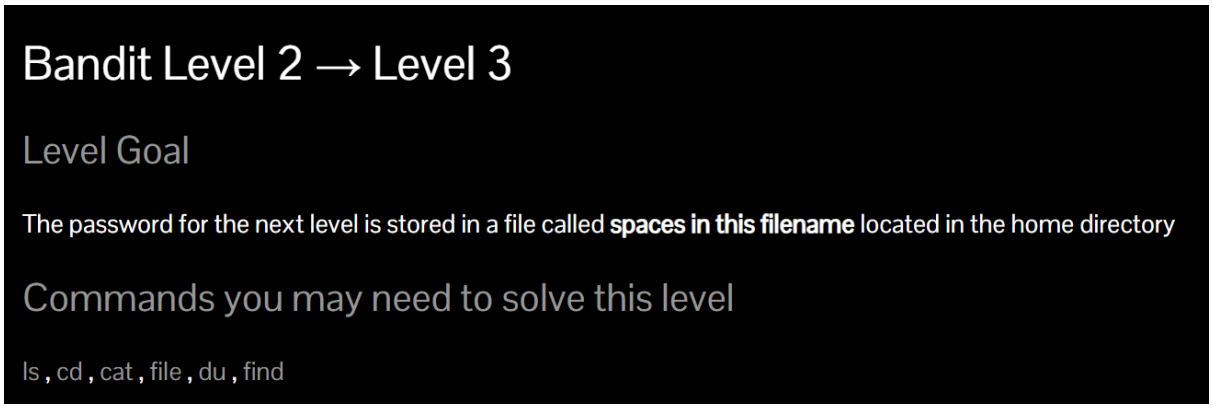
Step 3

Type

`“cat ./-”`

use the cat command again to display the contents of the file, however since it is named “-” we need to add “./” in the command to open the file. the file also contains the password for the next level: **263JGJPfgU6LtdEvgfWU1XP5yac29mFx**

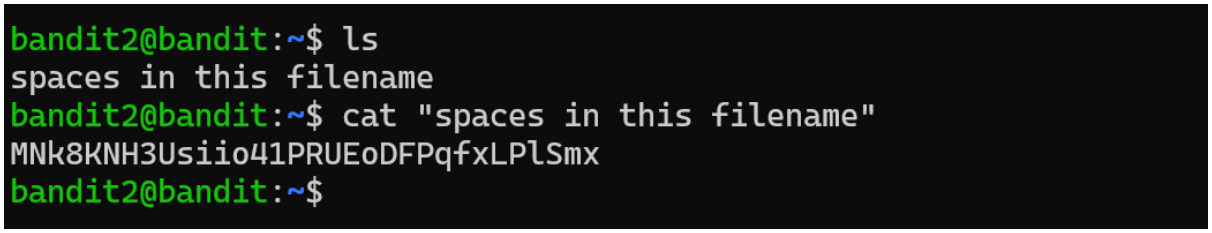
Level 2-3



Step 1
Open another Powershell tab, repeat the process in the previous level, this time connect to :

```
ssh bandit2@bandit.labs.overthewire.org -p2220
```

Step 2



Type

```
“ls”
```

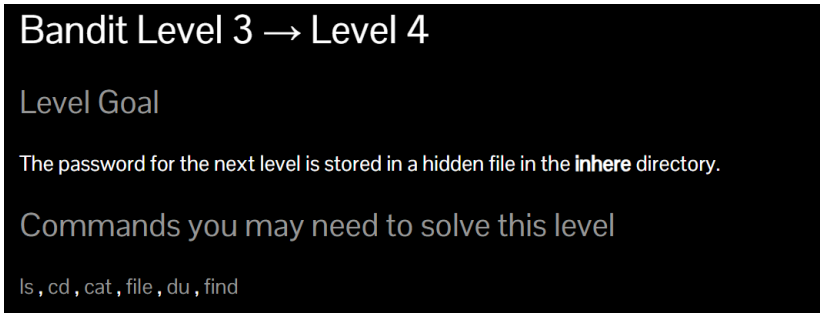
the file “spaces in this filename” will be displayed in the directory.

Step 3
Type

```
“ cat “spaces in this filename” ”
```

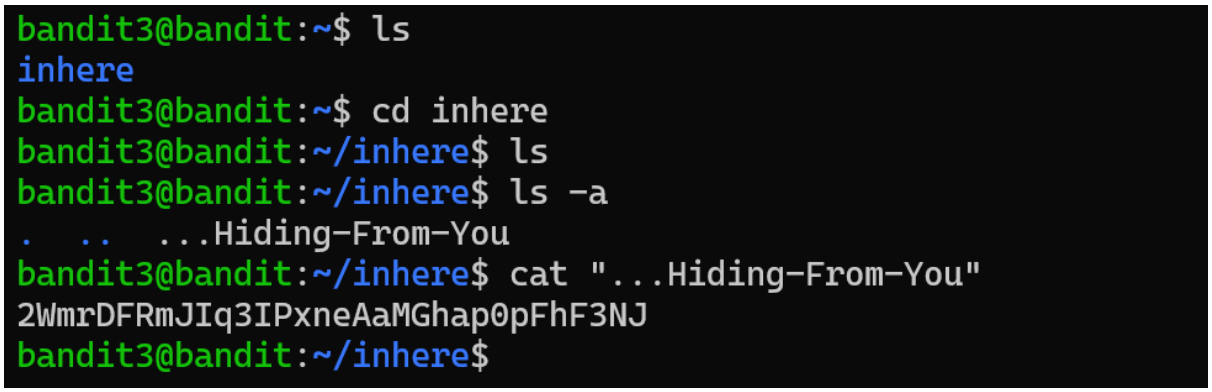
This time we are opening the file “spaces in the filename”, which according to the name has spaces in it. Because of this, we must use “ “ after cat to view the contents of the file. It will then reveal the password for the next level: **MNk8KNH3Usiio41PRUEoDFPqfxLPlSmx**

Level 3-4



Step 1
Open another Powershell tab, repeat the process in the previous level, this time connect to :

```
ssh bandit3@bandit.labs.overthewire.org -p2220
```



Step 2
Type

```
“ls”
```

The File “inhere” contains files inside of it, so we must change the directory to view its contents. We will use the “cd” command. type:

```
“cd inhere”
```

Step 3
However, when we type “ls” again, there are no files shown, so we try using “ls -a”, this way, hidden files in the directory will be revealed/not ignored.

Step 4
Then once the file is revealed we then can open it using:

```
cat “...Hiding-From-You”
```

The password for the next level : **2WmrDFRmJIq3IPxneAaMGhap0pFhF3NJ**

Level 4-5

Bandit Level 4 → Level 5

Level Goal

The password for the next level is stored in the only human-readable file in the **inhere** directory. Tip: if your terminal is messed up, try the "reset" command.

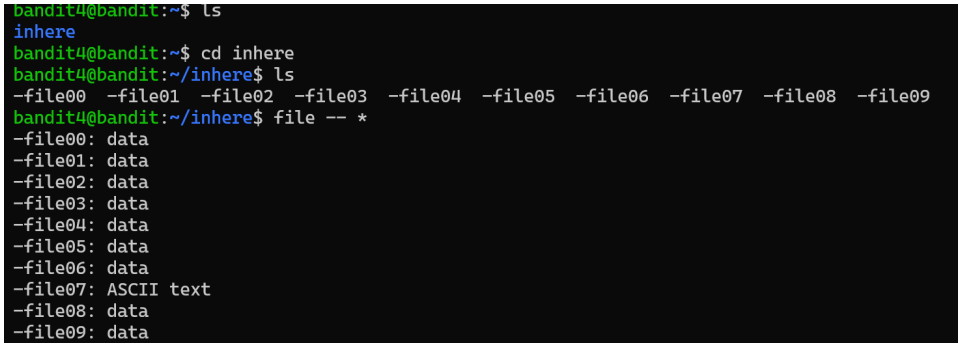
Commands you may need to solve this level

ls , cd , cat , file , du , find

Step 1

Open another Powershell tab, repeat the process in the previous level, this time connect to :

```
ssh bandit4@bandit.labs.overthewire.org -p2220
```



Step 2

Type "ls" again and change the directory to "inhere" again just like the previous level.

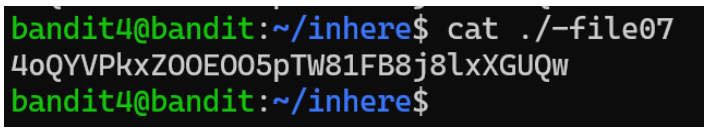
Step 3

Type "ls" to reveal the contents of "inhere" which will result in multiple files. however we are tasked with finding a file with "human-readable text" which is also called an ASCII text.
to do that, type

```
"file -- *"
```

The "file" command is used to determine the type of all the files in the directory. using this will help find the file with ASCII text.
Once done, it will reveal that every file has a type of data except for -file07 which has human-readable text, so let's view the content of the file.

Step 4



Type

```
"cat ./-file07"
```

Just like the previous levels, we use the "cat" command to view the contents of files.
The password for the next level : **4oQYVPkxZOOEOO5pTW81FB8j8lxXGUQw**

Level 5-6

Bandit Level 5 → Level 6

Level Goal

The password for the next level is stored in a file somewhere under the **inhere** directory and has all of the following properties:

- human-readable
- 1033 bytes in size
- not executable

Step 1

Open another Powershell tab, repeat the process in the previous level, this time connect to :

```
ssh bandit5@bandit.labs.overthewire.org -p2220
```

Step 2

```
bandit5@bandit:~$ ls
inhere
bandit5@bandit:~$ cd inhere
```

use the “ls” command and change the directory to “inhere”

Step 3

```
bandit5@bandit:~/inhere/maybehere07$ file -- *
-file1:       ASCII text, with very long lines (3662)
-file2:       ASCII text, with very long lines (2487)
-file3:       data
spaces file1: ASCII text, with very long lines (4129)
spaces file2: ASCII text, with very long lines (9063)
spaces file3: data
```

use the “**file – ***” command again to find files with ASCII text

Step 4

```
bandit5@bandit:~/inhere/maybehere07$ find . -size 1033c ! -executable
./.file2
```

Find the last 2 conditions for the file we are looking for, which is 1033 bytes in size and not executable, using

```
find . -size 1033c ! -executable
```

Step 5

Now that we found the file, which is “file2”,

```
bandit5@bandit:~/inhere/maybehere07$ cat ./.file2
HWasnPhtq9AVKe0dmk45nxy20cvUa6EG
```

use the “cat” command to view the contents of the file, which is the password for the next level : **HWasnPhtq9AVKe0dmk45nxy20cvUa6EG**

Level 6-7

Bandit Level 6 → Level 7

Level Goal

The password for the next level is stored **somewhere on the server** and has all of the following properties:

- owned by user bandit7
- owned by group bandit6
- 33 bytes in size

Step 1
Open another Powershell tab, repeat the process in the previous level, this time connect to :

```
ssh bandit6@bandit.labs.overthewire.org -p2220
```

Step 2

```
bandit6@bandit:~$ ls
bandit6@bandit:~$
```

ls does not reveal any files, so use **ls -a** to reveal hidden files

```
bandit6@bandit:~$ ls
bandit6@bandit:~$ ls -a
.  ..  .bash_logout  .bashrc  .profile
bandit6@bandit:~$ cat .bashrc
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
case $- in
    *i*) ;;
      *) return;;
esac

# don't put duplicate lines or lines starting with space in the history.
# See bash(1) for more options
HISTCONTROL=ignoreboth

# append to the history file, don't overwrite it
shopt -s histappend

# for setting history length see HISTSIZE and HISTFILESIZE in bash(1)
HISTSIZE=1000
HISTFILESIZE=2000

# check the window size after each command and, if necessary,
# update the values of LINES and COLUMNS.
shopt -s checkwinsize
```

there are hidden files that are revealed, but there seems to be nothing that is helpful
so i try and use find to find it manually based on the properties:

Step 3
The password is in a **file** which is owned by **bandit7**, in the bandit6 **group** and is **33 bytes** in size. we can use a similar solution to the previous level:

```
find . -type f -user bandit7 -group bandit6 -size 33c
```

however “find .” does not seem to work because the file is not found in the directory. so switch it to “find /” which searches the root directory, so enter the command:

```
find / -type f -user bandit7 -group bandit6 -size 33c
```

```
bandit6@bandit:~$ find / -type f -user bandit7 -group bandit6 -size 33c
find: ‘/drifter/drifter14_src/axTLS’: Permission denied
find: ‘/root’: Permission denied
find: ‘/snap’: Permission denied
find: ‘/tmp’: Permission denied
find: ‘/proc/tty/driver’: Permission denied
find: ‘/proc/2383822/task/2383822/fdinfo/6’: No such file or directory
find: ‘/proc/2383822/fdinfo/5’: No such file or directory
find: ‘/home/bandit31-git’: Permission denied
find: ‘/home/ubuntu’: Permission denied
find: ‘/home/bandit5/inhere’: Permission denied
find: ‘/home/bandit30-git’: Permission denied
find: ‘/home/drifter8/chroot’: Permission denied
find: ‘/home/drifter6/data’: Permission denied
find: ‘/home/bandit29-git’: Permission denied
find: ‘/home/bandit28-git’: Permission denied
find: ‘/home/bandit27-git’: Permission denied
find: ‘/lost+found’: Permission denied
find: ‘/etc/polkit-1/rules.d’: Permission denied
find: ‘/etc/multipath’: Permission denied
find: ‘/etc/stunnel’: Permission denied
find: ‘/etc/xinetd.d’: Permission denied
find: ‘/etc/credstore.encrypted’: Permission denied
```

Step 4

There are a lot of files, but they all have permission denied so we cannot access them, however by scrolling down you can notice one file that does not have its permissions denied.

```
find: ‘/var/lib/amazon’: Permission denied
/var/lib/dpkg/info/bandit7.password
find: ‘/var/lib/apt/lists/partial’: Permission denied
```

since there is permission, let’s view the contents of the file using **cat**

```
bandit6@bandit:~$ cat /var/lib/dpkg/info/bandit7.password
morbNTDkSW6jIlUc0ymOdMaLnOlFVAaj
```

The password for the next level : **morbNTDkSW6jIlUc0ymOdMaLnOlFVAaj**

Level 7-8

Bandit Level 7 → Level 8

Level Goal

The password for the next level is stored in the file **data.txt** next to the word **millionth**

Step 1
Open another Powershell tab, repeat the process in the previous level, this time connect to :

```
ssh bandit7@bandit.labs.overthewire.org -p2220
```

Step 2

```
bandit7@bandit:~$ ls
data.txt
```

check the directory, and there is the data.txt

Step 3

```
bandit7@bandit:~$ cat data.txt
```

cat to view the txt file.

```
gondola HAr2v629EBfAkbDijIcnipoGDzFyuuxv
coronary        cgcSIIKdhGasyu3fc8fFL5AfrLsUmQsD
hence   lebN1bFDn0AMtiTfJLt5Dne7FB4lTVvn
baseness        LmEn1ggqfkHXahBXK1NMC7XC9YDD4SeB
tire    5E7wMsKjT1YKIYMPI1fv7TlQP3dJGsSB
dismal  4VjV4LnEz9SFJle6IpA7qcYFlccPzlWl
compressors     ZYUIpLtvu6V4QgTTmxOVs0BMd0Ind0hP
thunder's       EKQprwO7PBli28Egqm1NkPf6hilRYR6e
toddy   5yHkW4YgmyKcpCMJMEWZE59Z94zLSeXb
impatient       xw1Js46f6PDlcv5ZCodESqxzRxKXNzR6
pirouette's     Be9qHe9qO6NbWp28izwrqh2rEn8UlJTH
```

The result will be a lot of passwords next to random words. it would be very time consuming if we were to manually look for the word millionth.

Step 4
to look for lines, specific words or patterns in a file, we can use **grep** command.

```
grep "millionth" data.txt
```

```
bandit7@bandit:~$ grep "millionth" data.txt
millionth       dfwvzFQi4mU0wfNbFOe9RoWskMLg7eEc
```

the password, which is next to the word “millionth” is found.

the password for the next level: **dfwvzFQi4mU0wfNbFOe9RoWskMLg7eEc**

Level 8-9

Step 1
Open another Powershell tab, repeat the process in the previous level, this time connect to :

```
ssh bandit8@bandit.labs.overthewire.org -p2220
```

Step 2

```
bandit8@bandit:~$ ls
data.txt
bandit8@bandit:~$ cat data.txt
aMKlTMrptUxxTypCHocCTrqYRkR2gT8h
PRerp5EfTVxJHKuCZDXfAfRyCQSdPjMi
0BKVRLEJQcpNx8wnSPxDLFnFKlQafKK6
6Boy6esAjnIxCYn8uI6KZ7VD7zysDM8i
tgHSfEXcbYCejWXfsWDO4VXXbqtTVcqS
KZJOZECxhLxDhxDbGzdNy8m0uplzvP11
w6x5XtaoRWDqMCsYxgZIWuOKVdiGByAu
0kJ7XHD4gVtNSZIpqyP1V45sfz9OBLFo
Wr4hWlUhGCKJpGDCeio8C1pLVt7DZm3X
Su9w1lri9UACf53cL1evAMKXVgI0nfqe
6Boy6esAjnIxCYn8uI6KZ7VD7zysDM8i
CgUjZiluCoMEvzNAge1Nbv3g9tpLQQj2
ysKmfYcysVfnViisRBcXzgjjXMDgnKKv
1VKPEkd0bCtIRwMFVQfY7InulwOFyDsn
ylbAYB5vBiEAmViEQOBwITUwjSZkwC7Q
8pePxslMzXqA2mi87wFjxd44qDRdrPiW
ylbAYB5vBiEAmViEQOBwITUwjSZkwC7Q
```

When data.txt is opened, there are many lines of text and we are supposed to find a line that does NOT repeat, or the line that does not repeat.

Step 3

```
bandit8@bandit:~$ sort data.txt
0BKVRLEJQcpNx8wnSPxDLFnFKlQafKK6
0BKVRLEJQcpNx8wnSPxDLFnFKlQafKK6
0BKVRLEJQcpNx8wnSPxDLFnFKlQafKK6
0BKVRLEJQcpNx8wnSPxDLFnFKlQafKK6
0BKVRLEJQcpNx8wnSPxDLFnFKlQafKK6
0BKVRLEJQcpNx8wnSPxDLFnFKlQafKK6
0BKVRLEJQcpNx8wnSPxDLFnFKlQafKK6
0BKVRLEJQcpNx8wnSPxDLFnFKlQafKK6
0BKVRLEJQcpNx8wnSPxDLFnFKlQafKK6
0BKVRLEJQcpNx8wnSPxDLFnFKlQafKK6
0eJPctF8gK96ykGBBaKydhJgxSpTlJtz
0eJPctF8gK96ykGBBaKydhJgxSpTlJtz
0eJPctF8gK96ykGBBaKydhJgxSpTlJtz
0eJPctF8gK96ykGBBaKydhJgxSpTlJtz
0eJPctF8gK96ykGBBaKydhJgxSpTlJtz
0eJPctF8gK96ykGBBaKydhJgxSpTlJtz
0eJPctF8gK96ykGBBaKydhJgxSpTlJtz
```

The **sort** command can be used to sort the lines of texts in data.txt

Step 4
To find the line of text that appears only once, the **uniq** command can be used:
for finding unique lines, the command is uniq -u.
However, **sort** and **uniq** must be used together in a pipeline. Running them separately won't give you the correct output because **uniq** will not be able to identify adjacent duplicates.
so the command we will be using is:

```
sort data.txt | uniq -u
```

the " | " or piping is used for inputting the sorted data into the uniq -u command

```
bandit8@bandit:~$ sort data.txt | uniq -u
4CKMh1JI91bUIZZPXDqGanal4xvAg0JM
```

That line is the line of text we are looking for , which is the password for the next level:
4CKMh1JI91bUIZZPXDqGanal4xvAg0JM

Level 9-10

Bandit Level 9 → Level 10

Level Goal

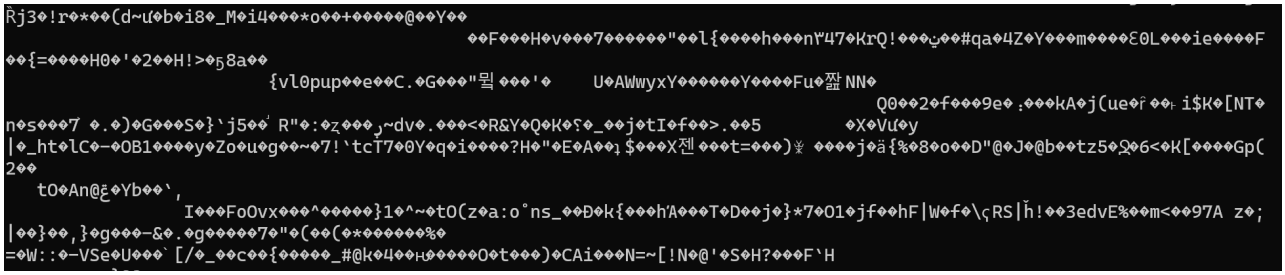
The password for the next level is stored in the file **data.txt** in one of the few human-readable strings, preceded by several '=' characters.

Step 1

Open another Powershell tab, repeat the process in the previous level, this time connect to :

```
ssh bandit9@bandit.labs.overthewire.org -p2220
```

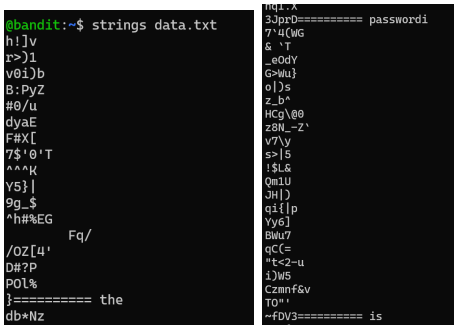
Step 2



When data.txt is opened using cat, this is what we get. we need to find human-readable strings.

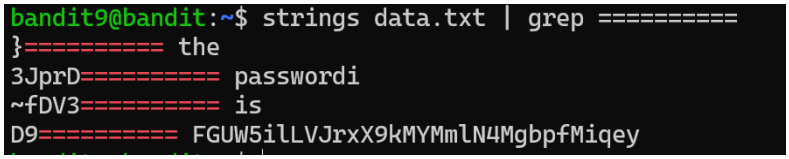
Step 3

to find human readable strings we can use the **strings** command.



There are a lot of human readable strings, and like the goal of the question says, the password is preceded by multiple “=” strings. just like some of the strings in the screenshot above. so we have to narrow the strings down to the ones that are preceded with “=”

Step 4



by piping the **strings** command and the **grep** ====== commands we get the password: **FGUW5ilLVJrxX9kMYMmlN4MgbpfMiqey**

Level 10-11

Bandit Level 10 → Level 11

Level Goal

The password for the next level is stored in the file **data.txt**, which contains base64 encoded data

Step 1
Open another Powershell tab, repeat the process in the previous level, this time connect to :

```
ssh bandit10@bandit.labs.overthewire.org -p2220
```

Step 2

```
bandit10@bandit:~$ ls
data.txt
bandit10@bandit:~$ cat data.txt
VGhlIHBhc3N3b3JkIGlzIGR0UjE3M2ZaS2IwUlJzREZTR3NnMlJXbnBOVmozcVJyCg==
```

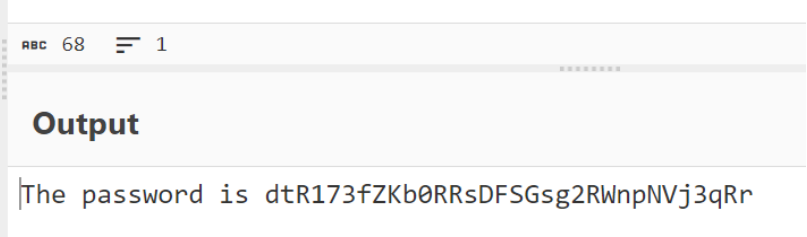
when we open data.txt there is a line of text inside text.txt, however according to the level goal, the data contains base64 encoded data, which means that we need to decode this to reveal the password.
Base64 is an encoding method that converts binary data into a text format using a set of 64 characters.

Step 3

```
bandit10@bandit:~$ base64 -d data.txt
The password is dtR173fZKb0RRsDFSGsg2RWnpNVj3qRr
```

```
base64 -d data.txt
```

To decode a Base64 string, we can use the **base64 -d** command in the terminal to decode the line.
or you can use an external tool/websites like cyberchef to decode from base64



The password for the next level: **dtR173fZKb0RRsDFSGsg2RWnpNVj3qRr**

Level 11-12

Bandit Level 11 → Level 12

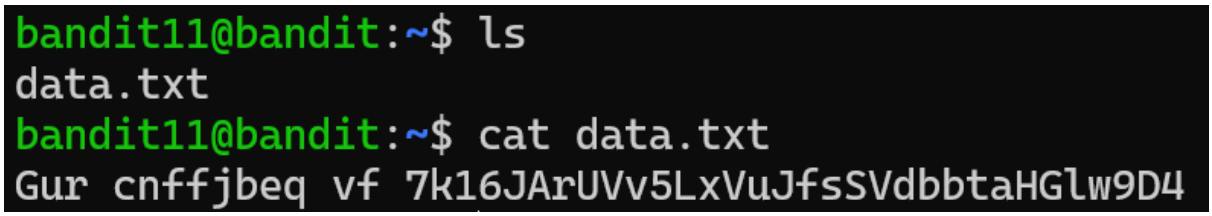
Level Goal

The password for the next level is stored in the file **data.txt**, where all lowercase (a-z) and uppercase (A-Z) letters have been rotated by 13 positions

Step 1
Open another Powershell tab, repeat the process in the previous level, this time connect to :

```
ssh bandit11@bandit.labs.overthewire.org -p2220
```

Step 2

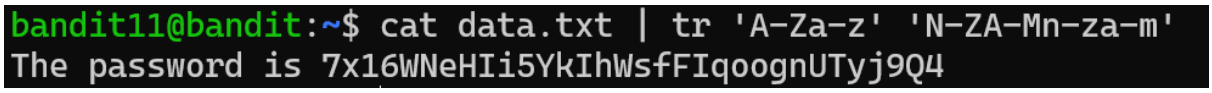


When the data.txt is opened, there is a line of text that has been encoded with rot13 which is a simple letter substitution cipher that replaces a letter with the 13th letter after it in the Latin alphabet.

Step 3
to apply decoding to the line of text, there is no built-in command that does it, so we need to use the **tr** which replaces letters or numbers. we do it while also piping it with cat data.txt :

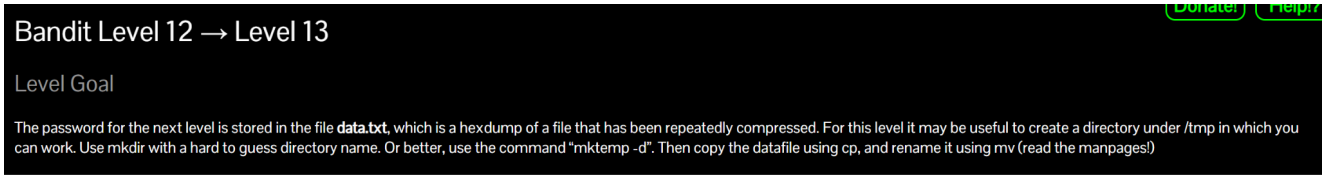
```
cat data.txt | tr 'A-Za-z' 'N-ZA-Mn-za-m'
```

'A-Za-z' 'N-ZA-Mn-za-m' is used specifically for rot13 decoding,
where ‘A-Za-z’ is the alphabet, and ‘N-ZA-Mn-za-m’ is the alphabet shifted by 13 positions.



From that, we get the password for the next level: **7x16WNeHIi5YkIhWsfFIqoognUTyj9Q4**

Level 12-13



Step 1
connect to :

```
ssh bandit12@bandit.labs.overthewire.org -p2220
```

Step 2:

```
bandit12@bandit:~$ ls
data.txt
bandit12@bandit:~$ cat data.txt
00000000: 1f8b 0808 dfcd eb66 0203 6461 7461 322e  .......f..data2.
00000010: 6269 6e00 013e 02c1 fd42 5a68 3931 4159  bin..>...BZh91AY
00000020: 2653 59ca 83b2 c100 0017 7fff dff3 f4a7  &SY.............
00000030: fc9f fefe f2f3 cffe f5ff ffdd bf7e 5bfe  ............~[.
00000040: faff dfbe 97aa 6fff f0de edf7 b001 3b56  ......o.......;V
00000050: 0400 0034 d000 0000 0069 a1a1 a000 0343  ...4.....i.....C
00000060: 4686 4341 a680 068d 1a69 a0d0 0068 d1a0  F.CA.....i...h..
00000070: 1906 1193 0433 5193 d4c6 5103 4646 9a34  .....3Q...Q.FF.4
00000080: 0000 d320 0680 0003 264d 0346 8683 d21a  ... ....&M.F....
00000090: 0686 8064 3400 0189 a683 4fd5 0190 001e  ...d4.....O.....
000000a0: 9034 d188 0343 0e9a 0c40 69a0 0626 4686  .4...C...@i..&F.
000000b0: 8340 0310 d340 3469 a680 6800 0006 8d0d  .@...@4i..h.....
000000c0: 0068 0608 0d1a 64d3 469a 1a68 c9a6 8030  .h....d.F..h...0
000000d0: 9a68 6801 8101 3204 012a ca60 51e8 1cac  .hh...2..*.`Q...
```

Inspecting the file dara.txt, which according to the level desc. , is a hexdump file that has been repeatedly compressed. It says that we have to make a directory , then copy and rename the file.

Step 3:

```
bandit12@bandit:~$ mktemp -d
/tmp/tmp.WyNppQ1FI7
```

Make a temporary directory with

mktemp -d.

Step 4:

```
bandit12@bandit:~$ cp data.txt /tmp/tmp.WyNppQ1FI7
bandit12@bandit:~$ cd /tmp/tmp.WyNppQ1FI7
bandit12@bandit:/tmp/tmp.WyNppQ1FI7$ ls
data.txt
```

Copy the data.txt file into the temporary directory that was just made.

cp data.txt /tmp/tmp.WyNppQ1FI7

Then, **ls** to check.

Step 5:

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ file data.txt
data.txt: ASCII text
```

the temporary directory's name is different because I had to restart the session.

When the file is checked, it is an ASCII text, to work with the actual hexdump, we need to convert it to binary first:

xxd -r data.txt hexdump

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ xxd -r data.txt hexdump
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ ls
data.txt  hexdump
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$
```

Step 6:
Inspecting the file with the **file command:**

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ file hexdump
hexdump: gzip compressed data, was "data2.bin", last modified: Thu Sep 19
 07:08:15 2024, max compression, from Unix, original size modulo 2^32 574
```

It is a gzip compressed data, and remembering the question's description, this file is compressed multiple times, so i will decompress it:
This time with gzip.

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ mv hexdump hexdump.gz
```

Rename the file to have the .gz extension.

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ gunzip hexdump.gz
```

Use gunzip /gzip decode to decompress the binary file.

Step 7:
After inspecting the file again with file:

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ file hexdump
hexdump: bzip2 compressed data, block size = 900k
```

It is still compressed but with bzip2, I will decompress it again:

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ mv hexdump hexdump.bz2
```

Rename the file with the bzip2 file extension, which is **.bz2**

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ bzip2 -d hexdump.bz2
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ ls
data.txt  hexdump
```

Decompress the file, with the command bzip2 -d .

Step 8:
Inspect the file again and this time it is compressed with gzip again:

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ file hexdump
hexdump: gzip compressed data, was "data4.bin", last modified: Thu Sep 19
 07:08:15 2024, max compression, from Unix, original size modulo 2^32 204
80
```

Repeat the previous process:

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ mv hexdump hexdump.gz
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ gunzip hexdump.gz
```

Step 9:
After inspecting the file again,

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ file hexdump
hexdump: POSIX tar archive (GNU)
```

It's a tar archive, I will decompress it again.

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ mv hexdump hexdump.tar
```

Rename it with the appropriate extension for tar archives.

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ tar -xvf hexdump.tar
data5.bin
```

After extracting with the tar -xvf command, we get the file: data5.bin

I used the same command on the file, and then i got another file:

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ tar -xvf data5.bin
data6.bin
```

Step 10:
I inspected the file again, and found out it is also compressed.

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ file data6.bin
data6.bin: bzip2 compressed data, block size = 900k
```

I repeated the process of decompressing bzip2 compressed data:

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ bzip2 -d data6.bin
bzip2: Can't guess original name for data6.bin -- using data6.bin.out
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ ls
data5.bin  data6.bin.out  data.txt  hexdump.tar
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ file data6.bin.out
data6.bin.out: POSIX tar archive (GNU)
```

The process outputs another file, and after checking it is another tar archive.

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ tar -xvf data6.bin.out
data8.bin
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ file data8.bin
data8.bin: gzip compressed data, was "data9.bin", last modified: Thu Sep
19 07:08:15 2024, max compression, from Unix, original size modulo 2^32 4
9
```

Another gzip compressed file.

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ gunzip data8.bin
gzip: data8.bin: unknown suffix -- ignored
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ mv data8.bin data8.gz
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ gunzip data8.gz
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ ls
data5.bin  data6.bin.out  data8  data.txt  hexdump.tar
```

I needed to change the extension before decoding with gzip. Then the result is the file data8.

Step 11:
Inspect the file:

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ file data8
data8: ASCII text
```

It is finally an ascii text, view the contents of the file with the cat command:

```
bandit12@bandit:/tmp/tmp.Y5zm9R4GXn$ cat data8
The password is FO5dwFsc0cbaIiH0h8J2eUks2vdTDwAn
```

We finally got the password for the next level:
FO5dwFsc0cbaIiH0h8J2eUks2vdTDwAn

Level 13-14

Bandit Level 13 → Level 14

Level Goal

The password for the next level is stored in **/etc/bandit_pass/bandit14 and can only be read by user bandit14**. For this level, you don't get the next password, but you get a private SSH key that can be used to log into the next level. **Note: localhost** is a hostname that refers to the machine you are working on

Step 1
connect to :

```
ssh bandit13@bandit.labs.overthewire.org -p2220
```

Step 2

```
bandit13@bandit:~$ ls
sshkey.private
```

There is a file, **sshkey.private**.

```
bandit13@bandit:~$ cat sshkey.private
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAxkkOE83W2cOT7IWhFc9aPaaQmQDdgzuXCv+ppZHa++buSkN+
gg0tcr7Fw8NLGa5+Uzec2rEg0WmeevB13AIoYp0MZyETq46t+jk9puNwZwIt9XgB
ZufGtZEwWbFWw/vVLNwOXBe4UWStGRWzgPpEeSv5Tb1VjLZIBdGphTIK22Amz6Zb
ThMsiMnyJafEwJ/T8PQO3myS91vUHEuoOMAzoUID4kN0MEZ3+XahyK0HJVq68KsV
ObefXG1vvA3GAJ29kxJaqvRfgYnqZryWN7w3CHjNU4c/2Jkp+n8L0SnxaNA+WYA7
jiPyTF0is8uzMlYQ4l1Lzh/8/MpvhCQF8r22dwIDAQABAoIBAQC6dWBjhyEOzjeA
J3j/RWmap9M5zfJ/wb2bfidNpwbB8rsJ4sZIDZQ7XuIh4LfygoAQSS+bBw3RXvzE
pvJt3SmU8hIDuLsCjL1VnBY5pY7Bju8g8aR/3FyjyNAqx/TLfzlLYfOu7i9Jet67
xAh0tONG/u8FB5I3LAI2Vp6OviwvdWeC4nOxCthldpuPKNLA8rmMMVRTKQ+7T2VS
nXmwYckKUcUgzoVSpiNZaS0zUDypdpy2+tRH3MQa5kqN1YKjvF8RC47woOYCktsD
o3FFpGNFec9Taa3Msy+DfQQhHKZFKIL3bJDONtmrVvtYK40/yeU4aZ/HA2DQzwhe
ol1AfiEhAoGBAOnVjosBkm7sblK+n4IEwPxs8sOmhPnTDUy5WGrpSCrXOmsVIBUf
laL3ZGLx3xCIwtCnEucB9DvN2HZkupc/h6hTKUYLqXuyLD8njTrbRhLgbC9QrKrS
M1F2fSTxVqPtZDlDMwjNR04xHA/fKh8bXXyTMqOHNJTHHNhbh3McdURjAoGBANkU
1hqfnW7+aXncJ9bjysr1ZWbqOE5Nd8AFgfwaKuGTTVX2NsUQnCMWdOp+wFak40JH
PKWkJNdBG+ex0H9JNQsTK3X5PBMAS8AfX0GrKeuwKWA6erytVTqjOfLYcdp5+z9s
8DtVCxDuVsM+i4X8UqIGOlvGbtKEVokHPFXP1q/dAoGAcHg5YX7WEehCgCYTzpO+
xysX8ScM2qS6xuZ3MqUWAxUWkh7NGZvhe0sGy9iOdANzwKw7mUUFViaCMR/t54W1
GC83sOs3D7n5Mj8x3NdO8xFit7dT9a245TvaoYQ7KgmqpSg/ScKCw4c3eiLava+J
3btnJeSIU+8ZXq9XjPRpKwUCgYA7z6LiOQKxNeXH3qHXcnHok855maUj5fJNpPbY
iDkyZ8ySF8GlcFsky8Yw6fWCqfG3zDrohJ5l9JmEsBh7SadkwsZhvecQcS9t4vby
9/8X4jS0P8ibfcKS4nBP+dT81kkkg5Z5MohXBORA7VWx+ACohcDEkprsQ+w32xeD
qT1EvQKBgQDKm8ws2ByvSUVs9GjTilCajFqLJ0eVYzRPaY6f++Gv/UVfAPV4c+S0
kAWpXbv5tbkkzbS0eaLPTKgLzavXtQoTtKwrjpolHKIHUz6Wu+n4abfAIRFubOdN
/+aLoRQ0yBDRbdXMsZN/jvY44eM+xRLdRVyMmdPtP8belRi2E2aEzA==
-----END RSA PRIVATE KEY-----
```

This is a private key that can be used to log in to bandit14.
Using this command :

```
ssh -i sshkey.private bandit14@bandit.labs.overthewire.org -p2220
```

The -i is included because the private key is being used.

Step 3

```
bandit13@bandit:~$ ssh -i sshkey.private bandit14@bandit.labs.overthewire.org -p2220
The authenticity of host '[bandit.labs.overthewire.org]:2220 ([127.0.0.1]:2220)' can't be established.
ED25519 key fingerprint is SHA256:C2ihUBV7ihnV1wUXRb4RrEcLfXC5CXlhmAAM/urerLY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

Connect to bandit14, then from here we will find the password, and according to the level's description it is stored inside /etc/bandit_pass/bandit14 , and only bandit14 can read it.
Since i have already logged in as bandit14, i will read the file:

```
bandit14@bandit:~$ cat /etc/bandit_pass/bandit14
MU4VWeTyJk8ROof1qqmcBPaLh7lDCPvS
```

The password for the next level:
MU4VWeTyJk8ROof1qqmcBPaLh7lDCPvS

Level 14-15

Bandit Level 14 → Level 15

Level Goal

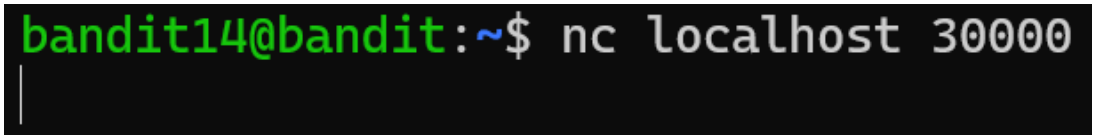
The password for the next level can be retrieved by submitting the password of the current level to **port 30000 on localhost.**

Step 1
Connect to:

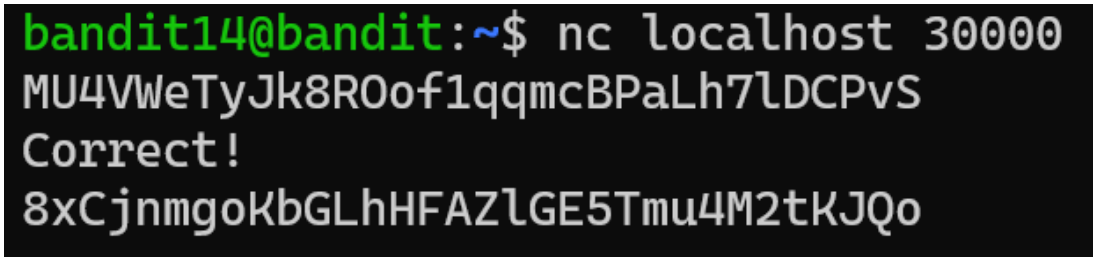
```
ssh bandit14@bandit.labs.overthewire.org -p2220
```

Step 2
The password for the next level is on the port 30000 on localhost.
To connect with the localhost, I used the **nc** command.

```
nc localhost 30000
```



After entering/submitting the password for the current level just like the description hinted, this happened:



It was successful, and we got the password for the next level:

8xCjnmgoKbGLhHFAZlGE5Tmu4M2tKJQo