

Bandit Level 0

Level Goal

The goal of this level is for you to log into the game using SSH. The host to which you need to connect is `bandit.labs.overthewire.org`, on port 2220. The username is `bandit0` and the password is `bandit0`. Once logged in, go to the Level 1 page to find out how to beat Level 1.

The first level of Bandit is extremely simple. We have to login to the host with below details:

Host: `bandit.labs.overthewire.org`

Port: 2220

User: `bandit0`

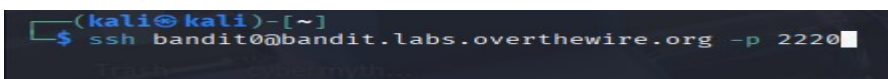
Password: `bandit0`

The password is provided at the beginning of the game to login to the host. Once we proceed further, the primary objective is to discover the passwords somewhere in the host.

In order to connect we are going to use SSH connection. SSH (Secure Shell or Secure Socket Shell) is a secure network protocol to access a computer over an unsecured network such as our current internet connection. SSH provides an encryption between data communications, strong password authentication and public key authentication.

The syntax to access is as follows. We start by typing `ssh`, followed by the username and hostname joined with `@` and port name 2220.

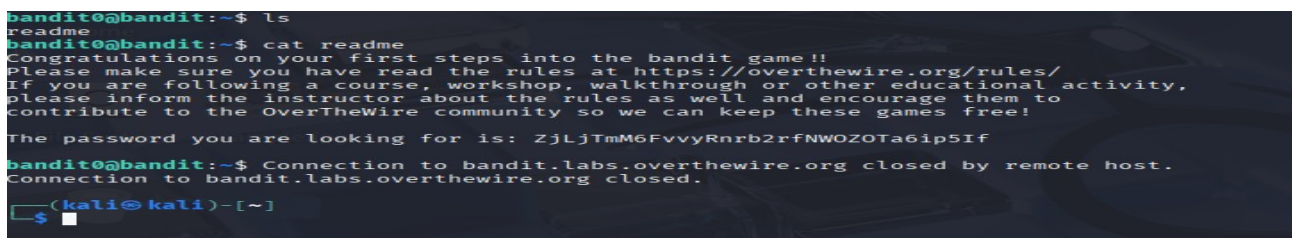
Press enter or click to see image in full size



```
(kali㉿kali)-[~]  
$ ssh bandit0@bandit.labs.overthewire.org -p 2220
```

You will be prompted a password. Type `bandit0` and you are in.

Press enter or click to see image in full size



```
bandit0@bandit:~$ ls  
readme  
bandit0@bandit:~$ cat readme  
Congratulations on your first steps into the bandit game!!  
Please make sure you have read the rules at https://overthewire.org/rules/  
If you are following a course, workshop, walkthrough or other educational activity,  
please inform the instructor about the rules as well and encourage them to  
contribute to the OverTheWire community so we can keep these games free!  
  
The password you are looking for is: ZjLjTmM6FvvyRnrb2rfNWOZ0Ta6ip5If  
bandit0@bandit:~$ Connection to bandit.labs.overthewire.org closed by remote host.  
Connection to bandit.labs.overthewire.org closed.  
(kali㉿kali)-[~]  
$
```

Bandit Level 0 to Level 1

Level Goal

The password for the next level is stored in a file called `readme` located in the home directory. Use this password to log into `bandit1` using SSH. Whenever you find a password for a level, use SSH (on port 2220) to log into that level and continue the game.

Commands you may need to solve this level

`ls` , `cd` , `cat` , `file` , `du` , `find`

At this level we need to locate a password that is concealed in a file named `readme`. We will use this password to proceed to next level.

First, we have to write `ls` command. `ls` command is utilized in order to list files in the present directory. Very conveniently we can observe there is a file named `readme`. But how do we read it? There comes the `cat` command.

`cat` is an abbreviation for concatenate is used to read data from a file and asks the input. Lets do it. Voila! Now we have password for next level.

Password: `ZjLjTmM6FvvyRnrb2rfNWOZOTa6ip5If`

Bandit Level 1 to Level 2

Level Goal

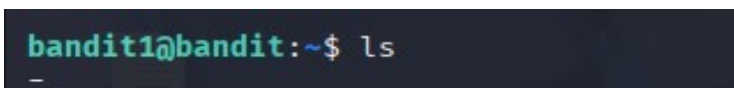
The password for the next level is stored in a file called `-` located in the home directory

Commands you may need to solve this level

`ls` , `cd` , `cat` , `file` , `du` , `find`

Now we know how to go from `bandit0` — `bandit1` lets log in to `bandit1`.

Press enter or click to see image in full size



```
bandit1@bandit:~$ ls
```

When prompted for the password, type in the password we have accumulated above.

According to the level goal, the password for the next level is inside a file named `-` which is in the home directory. Lets again searching with `ls` command. Using the `ls` command we can view the `-` file.

But opening the file using the `cat` command we learned is not possible.

So lets do something else. I would like to attempt by indicating the file with ./cat.

Now we have the password.

Press enter or click to view image in full size

```
bandit1@bandit:~$ ls
bandit1@bandit:~$ cat ./-
263JGJPFgU6LtdEv9fWU1XP5yac29mFx
bandit1@bandit:~$
```

Password for this level is : -263JGJPFgU6LtdEv9fWU1XP5yac29mFx

Bandit Level 2 to Level 3

Level Goal

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

Commands you may need to solve this level

ls , cd , cat , file , du , find

In this level we are need to read the password from a file named spaces in this filename. Then lets connect.

Press enter or click to view image in full size

```
bandit2@bandit:~$ ls
--spaces in this filename--
bandit2@bandit:~$ cat -- '--spaces in this filename--
```

As always lets take a look with the ls command. We can now see the file. In order to read this file we need to put "" at the both ends of the sentence.

Press enter or click to view image in full size

```
bandit2@bandit:~$ ls
--spaces in this filename--
bandit2@bandit:~$ cat -- '--spaces in this filename--'
MNk8KNH3Usiio41PRUEoDFPqfxLPLSmx
bandit2@bandit:~$
```

Now we have the password.

Password for the level: MNk8KNH3Usiio41PRUEoDFPqfxLPLSmx

Bandit Level 3 to Level 4

Level Goal

The password for the next level is stored in a hidden file in the inhere directory.

Commands you may need to solve this level

ls , cd , cat , file , du , find

In this level we need to find the password in a hidden file in the inhere directory. Lets start with connecting.

Press enter or click to view image in full size

```
bandit3@bandit:~$ ls
inhere
bandit3@bandit:~$ cd inhere
```

As we are in, lets again take a look with ls. With ls we can see that inhere directory is here.

Now we need to change our current directory to inhere. For that purpose we are going to use “cd” (change directory) command which is a command for changing directories.

After that with ls code we will take a look but nothing pops up.

Press enter or click to view image in full size

```
bandit3@bandit:~$ ls
inhere
bandit3@bandit:~$ cd inhere
```

To view the hidden directory we have to use ls -la or ls — a command that lists the hidden files in the directory. Now we can view that the hidden file is here. Now let's open it write the following command: cat ./hidden. Then we have the password.

Press enter or click to view image in full size

```
bandit3@bandit:~$ ls
inhere
bandit3@bandit:~$ cd inhere
bandit3@bandit:~/inhere$ ls -la
.. .. Hiding-From-You
bandit3@bandit:~/inhere$ cat ... Hiding-From-You
2WmrDFRmJIq3IPxneAaMGhap0pFhF3NJ
bandit3@bandit:~/inhere$
```

Password for this level: 2WmrDFRmJIq3IPxneAaMGhap0pFhF3NJ

Bandit Level 4 to 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

The password to this level as the game says is kept in the only human readable file in inhere directory. Human readable is the hint for us here. Lets first login and see. I will be avoiding the login section from here since we learnt that. Now let's go to the ls and switch our directory to inhere using cd. To view the files lets use another ls command.

Press enter or click to view image in full size

```
bandit4@bandit:~$ ls
inhere
```

As you can notice there are couple of files here. It is also possible to examine each of them but this is not hackers way :) By human readable challenge provides us with tremendous hint meaning that it can be either ASCII or UNICODE as they are the most widespread data encoding that are human readable.

Then let us try using the file command. File command is to identify they file type and file * is to list all the files in the present working directory. In that way we can view the file format.

Press enter or click to view image in full size

```
inhere
bandit4@bandit:~$ cd inhere
bandit4@bandit:~/inhere$ ls -a
.  .. -file00 -file01 -file02 -file03 -file04 -file05 -file06 -file07 -file08 -file09
bandit4@bandit:~/inhere$ cat ./-file07
```

Good now we know the file we are searching is in file07. Lets cat the file with cat ./-file07 and check the password.

Press enter or click to view image in full size

```
bandit4@bandit:~$ ls
inhere
bandit4@bandit:~$ cd inhere
bandit4@bandit:~/inhere$ ls -a
.  .. -file00 -file01 -file02 -file03 -file04 -file05 -file06 -file07 -file08 -file09
bandit4@bandit:~/inhere$ cat ./-file07
4oQYVPkxZ00E005pTW81FB8j8lxXGUQw
bandit4@bandit:~/inhere$
```

Password for the next level is: 4oQYVPkxZ00E005pTW81FB8j8lxXGUQw

Bandit Level 5 to 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

Commands you may need to solve this level

ls , cd , cat , file , du , find

In this level we have to search a file which should satisfy the following conditions.

human-readable

1033 bytes in size

not executable

Lets connect and see. With our normal steps we connect to the host, and type ls to check inhere is here. Type cd inhere to go to inhere directory. In inhere then type ls-la to check the files.

Press enter or click to see image in full size

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

Now we can notice there are plenty of files there and it will take too long to go through one by one. For this scenario we will be using the find command.

Find is employed in order to locate files and directories. We are going to form a find command that satisfies the challenge requirements.

c — bytes

w — two-byte words

k — Kilobytes

M — Megabytes

G — Gigabytes

So let's write find `-.size 1033c`."

Press enter or click to view image in full size

```
bandit5@bandit:~$ ls
inhere
bandit5@bandit:~$ cd inhere
bandit5@bandit:~/inhere$ ls -la
.  maybehere00  maybehere02  maybehere04  maybehere06  maybehere08  maybehere10  maybehere12  maybehere14  maybehere16  maybehere18
.. maybehere01  maybehere03  maybehere05  maybehere07  maybehere09  maybehere11  maybehere13  maybehere15  maybehere17  maybehere19
bandit5@bandit:~/inhere$ cd maybehere07
bandit5@bandit:~/inhere/maybehere07$ ls
-file1 -file2 -file3 spaces file1 spaces file2 spaces file3
bandit5@bandit:~/inhere/maybehere07$ cat ./file2
cat: ./file2: No such file or directory
bandit5@bandit:~/inhere/maybehere07$ cat ./file2
HWasnPhtq9AVKe0dmk45nxy20cvUa6EG
```

There we see that the only file with that details is file2 belonging to maybehere07. Let's switch to that directory and cat the file. Then comes the passport.

Password for the file: HWasnphtq9AVKe0dmk45nxy20cvUa6EG

Bandit Level 6 to Level 7

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

Commands you may need to solve this level

ls , cd , cat , file , du , find

Lets challenge is instructing us to search for the file which has been stored somewhere in the directory. Some clues we need to search for are:

owned by user bandit7

owned by group bandit6

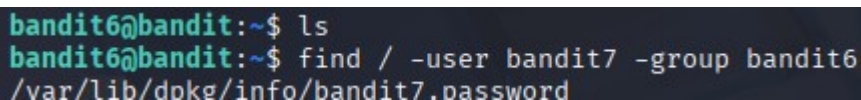
33 bytes in size

So lets use -user, -group and -size options of the find. Lets execute the below command.

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

Type is for file since we want a file, user is for bandit7 group is for bandit6 and size is for 33 bytes.

Press enter or click to view image in full size



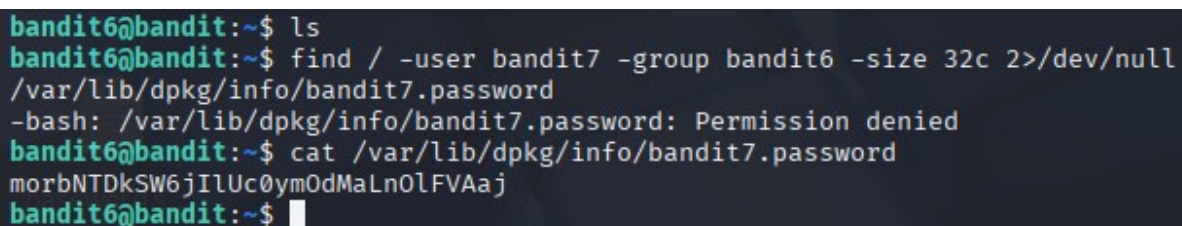
```
bandit6@bandit:~$ ls
bandit6@bandit:~$ find / -user bandit7 -group bandit6
/var/lib/dpkg/info/bandit7.password
```

But the issue with this query is that it will show all files that you dont have permission. So we should include 2>/dev/null command which will suppress the error messages.

```
find / -type f -user bandit7 -group bandit6 -size 33c 2>/dev/nul
```

This one is working and give us a folder path and file. Rest is simple. cd to /var/lib/dpkg/info/ and cat bandit7.password and we get our password

Press enter or click to see image in full size



```
bandit6@bandit:~$ ls
bandit6@bandit:~$ find / -user bandit7 -group bandit6 -size 32c 2>/dev/null
/var/lib/dpkg/info/bandit7.password
-bash: /var/lib/dpkg/info/bandit7.password: Permission denied
bandit6@bandit:~$ cat /var/lib/dpkg/info/bandit7.password
morbNTDkSW6jILUc0ymOdMaLnOLFVAaj
bandit6@bandit:~$
```


Lets proceed to the level 7.

Password for the level: morbNTDkSW6jIlUc0ymOdMaLn0lFVAaj

Bandit Level 7 to Level 8

Level Goal

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

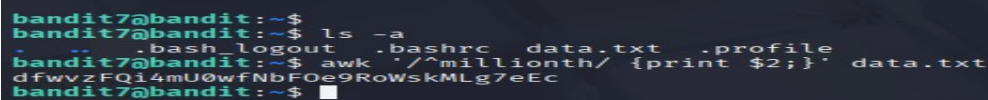
Commands you may need to solve this level

man, grep, sort, uniq, strings, base64, tr, tar, gzip, bzip2, xxd

Our task in this level is to get the password from the file data.txt beside the word millionth.

data.txt is in our working directory as we can notice from simple. ls command.

Press enter or click to view image full size

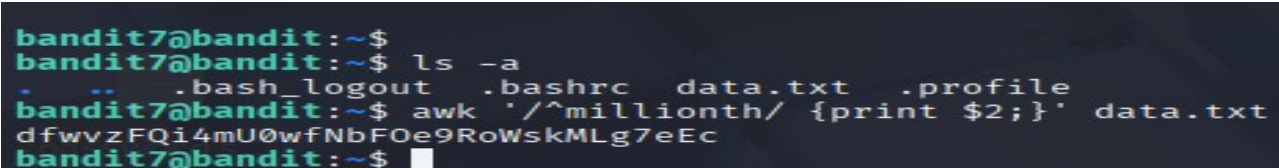


```
bandit7@bandit:~$  
bandit7@bandit:~$ ls -a  
..  .bash_logout  .bashrc  data.txt  .profile  
bandit7@bandit:~$ awk '/^millionth/ {print $2;}\' data.txt  
dfwvzFQi4mU0wfNbFOe9RoWskMLg7eEc  
bandit7@bandit:~$
```

We dont want to cat the entire document and search for word millionth.

We have a command named "grep" for these situations. grep looks for text in the file contents as well as the file names. Next we will use a -w to find the whole word. For instance if we search for on grep it should look for any word that has ten in it like contain. To restrict that we use -w. Next we define the folder path and filename that will be searched. Lastly we will utilize -e to define the word. grep -w data.txt -e 'millionth' is the equation.

Press enter or click to see image in full size



```
bandit7@bandit:~$  
bandit7@bandit:~$ ls -a  
..  .bash_logout  .bashrc  data.txt  .profile  
bandit7@bandit:~$ awk '/^millionth/ {print $2;}\' data.txt  
dfwvzFQi4mU0wfNbFOe9RoWskMLg7eEc  
bandit7@bandit:~$
```

Now we have our equation.

Password for the level: - dfwvzFQi4mU0wfNbFOe9RoWskMLg7eEc

Bandit Level 8 to Level 9

Level Goal

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

Commands you may need to solve this level

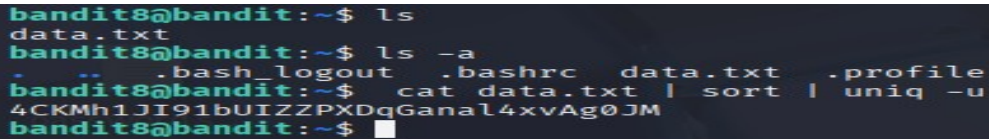
man, grep, sort, uniq, strings, base64, tr, tar, gzip, bzip2, xxd

For this assignment we need to grab some word from the data.txt and the only line of text which appear only once. This was time consuming for me to learn.

Lets sort the lines and delete the duplicates. SORT command is used to sort a file, and arrange in a specific order. Then we will use the uniq command to delete the deplicates.

Lets write the command. sort data.txt | uniq -u.

Press enter or click to view image in full size



```
bandit8@bandit:~$ ls
data.txt
bandit8@bandit:~$ ls -a
.  ..  .bash_logout  .bashrc  data.txt  .profile
bandit8@bandit:~$ cat data.txt | sort | uniq -u
4CKMh1JI91bUIZZPXDqGanal4xvAg0JM
bandit8@bandit:~$
```

This will provide us with the password.

Password for the level: 4CKMh1JI91bUIZZPXDqGanal4xvAg0JM

Bandit Level 9 to 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.

Commands you may need to solve this level

grep, sort, uniq, strings, base64, tr, tar, gzip, bzip2, xxd

Once logged in, we can look for the password within the data.txt in one of the few readable strings, preceded by a number of '=' characters as indicated.

So first we have to resolve the human-readable strings through String command. String command will give us every string type of printable characters in the file.

So now our command will be: "cat data.txt | strings | grep ^="

Press enter or click to see image in full size

```

bandit9@bandit:~$ ls -a
.  ..  .bash_logout  .bashrc  data.txt  .profile
bandit9@bandit:~$ strings data.txt | grep "="
epr~F=K
7?YD=
^y=7: System= sample.jpg
=?t)
=8]'
===== the
If=q
U.=4!
k={7
lTOB=
YZ=* Trash= cybermyth...
D===== password
w===== is
=*{>
=wu,
h=0"
4,=Y
*y=1
4=+0 onclas...
7]J=0
zSKc=
]2m=
===== FGUW5ilLVJrxX9kMYMmlN4MgbpfMiqey
t|HfY=

```

```

bandit9@bandit:~$ ls -a
.  ..  .bash_logout  .bashrc  data.txt  .profile
bandit9@bandit:~$ strings data.txt | grep "="
epr~F=K
7?YD=
^y=7: System= sample.jpg
=?t)
=8]'
===== the
If=q
U.=4!
k={7
lTOB=
YZ=* Trash= cybermyth...
D===== password
w===== is
=*{>
=wu,
h=0"
4,=Y
*y=1
4=+0 onclas...
7]J=0
zSKc=
]2m=
===== FGUW5ilLVJrxX9kMYMmlN4MgbpfMiqey
t|HfY=

```

Our password is here.

Password for this level: FGUW5ilLVJrxX9kMYMmlN4MgbpfMiqey

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

Commands you may need to solve this level

grep, sort, uniq, strings, base64, tr, tar, gzip, bzip2, xxd

To sign up : ssh bandit10@bandit.labs.overthewire.org -p 2220

Password : FGUW5ilLVJrxX9kMYMmlN4MgbpfMiqey

The command **base64 -d date.txt** is used to decode a Base64-encoded file named **date.txt** . Base64 is a binary-to-text encoding scheme that is commonly used to transmit data over the internet, such as email attachments. The **-d** flag tells the **base64** command to decode the file.

```
bandit10@bandit:~$ ls
data.txt
bandit10@bandit:~$ ls -la
total 24
drwxr-xr-x  2 root    root      4096 Aug 15 13:15 .
drwxr-xr-x 150 root    root      4096 Aug 15 13:18 ..
-rw-r--r--  1 root    root       220 Mar 31  2024 .bash_logout
-rw-r--r--  1 root    root     3851 Aug 15 13:09 .bashrc
-rw-r--r--  1 bandit11 bandit10   69 Aug 15 13:15 data.txt
-rw-r--r--  1 root    root      807 Mar 31  2024 .profile
bandit10@bandit:~$ ls -a
.  .. .bash_logout .bashrc data.txt .profile
bandit10@bandit:~$ cat data.txt
VGhlIHBhc3N3b3JkIGlzIGR0UjE3M2ZaS2IwUlJzREZTR3NnMlJXbnBOVmozcVJyCg==
bandit10@bandit:~$ echo VGhlIHBhc3N3b3JkIGlzIGR0UjE3M2ZaS2IwUlJzREZTR3NnMlJXbnBOVmozcVJyCg== |base64 -decode
base64: invalid option -- 'e'
Try 'base64 --help' for more information.
bandit10@bandit:~$ echo VGhlIHBhc3N3b3JkIGlzIGR0UjE3M2ZaS2IwUlJzREZTR3NnMlJXbnBOVmozcVJyCg== |base64 --decode
The password is dtR173fZKb0RRsDFSGsg2RWnpNVj3qRr
```

Our password is here.

Password for this level: dtR173fZKb0RRsDFSGsg2RWnpNVj3qRr

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

Commands you may need to solve this level

grep, sort, uniq, strings, base64, tr, tar, gzip, bzip2, xxd

Here we see another “data.txt” file in the home directory. The webpage for this level says that there are upper and lower case characters in this file which are rotated by 13 positions. So basically each letter in a string is replaced by a character at the 13th position from it. So a will be replaced by n, b will be replaced by o, and so on. This is basically a special type of Ceaser's cipher. After reading around for a bit, I found a StackOverflow link explaining to decrypt this cipher using the ‘tr’ command which stands for translate. This command

translates a set of characters into another set of characters. So we can then run the command “`tr '[-a-z][-A-Z]' '[-n-za-m][-N-ZA-M]'`”. This command will basically rearrange the characters decrypt the data in the file.

```
Enjoy your stay!
bandit11@bandit:~$ ls
data.txt
bandit11@bandit:~$ ls -la
. .bash_logout .bashrc data.txt .profile
bandit11@bandit:~$ cat data.txt | tr '[-a-z][-A-Z]' '[-n-za-m][-N-ZA-M]'
The password is 7x16WNeHIi5YkIhWsfFIqoognUTyj9Q4
bandit11@bandit:~$ exit
logout
Connection to bandit.labs.overthewire.org closed.

(kali@kali)-[~]
$ ssh bandit12@bandit.labs.overthewire.org -p 2220

bandit12@bandit:~$
```

Bandit Level 12 → Level 13

Level Goal

The password for the next level is stored in the file `data.txt`, which is a hexdump of a file that has been repeatedly compressed. For this level it may be useful to create a directory under `/tmp` in which you can work. Use `mkdir` with a hard to guess directory name. Or better, use the command “`mktemp -d`”. Then copy the datafile using `cp`, and rename it using `mv` (read the manpages!)

Commands you may need to solve this level

`grep`, `sort`, `uniq`, `strings`, `base64`, `tr`, `tar`, `gzip`, `bzip2`, `xxd`, `mkdir`, `cp`, `mv`, `file`

In this level, the password is stored in “**data.txt**” which is a hexdump of a file that’s been repeatedly compressed. The webpage for this level suggests that we make a folder in the `/tmp` directory, copy the file in this directory and work on it. So we can create a directory using ‘**`mkdir /tmp/praveen`**’ and copy the file in this location by running the command ‘**`cp data.txt /tmp/praveen`**’. We then rename the file using the command ‘**`mv data.txt hex.txt`**’.

```
bandit12@bandit:~$ ls -al
total 24
drwxr-xr-x  2 root    root    4096 Aug 15 13:15 .
drwxr-xr-x 150 root    root    4096 Aug 15 13:18 ..
-rw-r--r--  1 root    root     220 Mar 31  2024 .bash_logout
-rw-r--r--  1 root    root   3851 Aug 15 13:09 .bashrc
-rw-r-----  1 bandit13 bandit12 2645 Aug 15 13:15 data.txt
-rw-r--r--  1 root    root     807 Mar 31  2024 .profile
bandit12@bandit:~$
bandit12@bandit:~$ mkdir /tmp/praveen
-bash: mkdir /tmp/praveen: No such file or directory
bandit12@bandit:~$ mkdir /tmp/praveen
bandit12@bandit:~$ cp data.txt /tmp/praveen
bandit12@bandit:~$ cd /tmp/praveen
bandit12@bandit:/tmp/praveen$ ls
data.txt
bandit12@bandit:/tmp/praveen$ file data.txt
data.txt: ASCII text
```

Here we need to first decrypt the hex dump file and for that, we use a tool called xxd. xxd is a tool that can make a hex dump and reverse it. So we use `xxd -r data.txt data1`. This should reverse the hex dump file and store its content in the file “new”. Now we know that this file has been repeatedly compressed. On running `file data1`, then use mv command to move data1 to data2.gz we find out that it has been zipped by using gzip. Now before we start decompressing the file, we need to make sure that it has the proper extension else the tools won’t work. So we first change the extension and rename the file data2.gz. We can now unzip the file using `gzip -d data2.gz`.

```
bandit12@bandit:/tmp/praveen$ xxd -r data.txt data1
bandit12@bandit:/tmp/praveen$ ls
data1 data.txt
bandit12@bandit:/tmp/praveen$
bandit12@bandit:/tmp/praveen$ file data1
data1: gzip compressed data, was "data2.bin", last modified: Fri Aug 15 13:15:53 2025, max compression, from Unix, original size modulo 2^32 584
bandit12@bandit:/tmp/praveen$ mv data1 data2.gz
bandit12@bandit:/tmp/praveen$ ls
data2.gz data.txt
bandit12@bandit:/tmp/praveen$ gzip -d data2.gz
bandit12@bandit:/tmp/praveen$ ls
data2 data.txt
bandit12@bandit:/tmp/praveen$ file data2
data2: bzip2 compressed data, block size = 900k
bandit12@bandit:/tmp/praveen$
```

Now after decompressing the file, we check the file type again using the `file` command. We see that it is of file type bzip2. So we use the tool bzip to decompress this file by using the command `bunzip2 data2`.

```
bandit12@bandit:/tmp/praveen$ file data2
data2: bzip2 compressed data, block size = 900k
bandit12@bandit:/tmp/praveen$ mv data2 data3.bz2
bandit12@bandit:/tmp/praveen$ ls
data3.bz2 data.txt
bandit12@bandit:/tmp/praveen$ bzip2 -d data3.bz2
bandit12@bandit:/tmp/praveen$ ls
data3 data.txt
bandit12@bandit:/tmp/praveen$ file data3
data3: gzip compressed data, was "data4.bin", last modified: Fri Aug 15 13:15:53 2025, max compression, from Unix, original size modulo 2^32 20480
bandit12@bandit:/tmp/praveen$ mv data3 data4.gz
bandit12@bandit:/tmp/praveen$ ls
data4.gz data.txt
bandit12@bandit:/tmp/praveen$
```

Now we have another compressed file! We run the `file` command again and see that it is a gzip file. So we rename it again decompress it using gunzip as described above. Now when we check the file type, we find out that it’s a tar file. So we rename the file with .tar extension and then run the command `tar -xvf data4.tar`. The `-x` flag tells tar to extract from the file and `-f` flags tells it that it is a regular file.

```
bandit12@bandit:/tmp/praveen$ gzip -d data4.gz
bandit12@bandit:/tmp/praveen$ ls
data4 data.txt
bandit12@bandit:/tmp/praveen$ file data4.txt
data4.txt: cannot open 'data4.txt' (No such file or directory)
bandit12@bandit:/tmp/praveen$ file data4
data4: POSIX tar archive (GNU)
bandit12@bandit:/tmp/praveen$ tar -xvf data4
data5.bin
bandit12@bandit:/tmp/praveen$ file data5.bin
data5.bin: POSIX tar archive (GNU)
bandit12@bandit:/tmp/praveen$ tar -xvf data5.bin
data6.bin
bandit12@bandit:/tmp/praveen$ file data6.bin
data6.bin: bzip2 compressed data, block size = 900k
bandit12@bandit:/tmp/praveen$ mv data6.bin data7.bz2
bandit12@bandit:/tmp/praveen$ ls
data4 data5.bin data7.bz2 data.txt
bandit12@bandit:/tmp/praveen$
```


On running the `ls` command, we see a new file named “data5.bin”. We check the file type and figure out that it is a .tar file. We know that this file has been compressed several times, so we go through the same process of finding the file, changing the extension in case it’s a gzip file, and then decompress it using the methods described above. We continue doing this until we finally get the text file. We then `cat` out the “data9” file which has our password.

```
bandit12@bandit:/tmp/praveen$ ls
data4 data5.bin data7.bz2 data.txt
bandit12@bandit:/tmp/praveen$ bzip2 -d data7.bz2
bandit12@bandit:/tmp/praveen$ ls
data4 data5.bin data7 data.txt
bandit12@bandit:/tmp/praveen$ file data7
data7: POSIX tar archive (GNU)
bandit12@bandit:/tmp/praveen$ tar -xvf data7
data8.bin
bandit12@bandit:/tmp/praveen$ file data8.bin
data8.bin: gzip compressed data, was "data9.bin", last modified: Fri Aug 15 13:15:53 2025, max compression, from Unix, original size modulo 2^32 49
bandit12@bandit:/tmp/praveen$ mv data8.bin data9.gz
bandit12@bandit:/tmp/praveen$ ls
data4 data5.bin data7 data9.gz data.txt
bandit12@bandit:/tmp/praveen$ gzip -d data9.gz
bandit12@bandit:/tmp/praveen$ ls
data4 data5.bin data7 data9 data.txt
bandit12@bandit:/tmp/praveen$
```

```
bandit12@bandit:/tmp/praveen$ file data8.bin
data8.bin: gzip compressed data, was "data9.bin", last modified: Fri Aug 15 13:15:53 2025, max compression, from Unix, original size modulo 2^32 49
bandit12@bandit:/tmp/praveen$ mv data8.bin data9.gz
bandit12@bandit:/tmp/praveen$ ls
data4 data5.bin data7 data9.gz data.txt
bandit12@bandit:/tmp/praveen$ gzip -d data9.gz
bandit12@bandit:/tmp/praveen$ ls
data4 data5.bin data7 data9 data.txt
bandit12@bandit:/tmp/praveen$ file data9
data9: ASCII text
bandit12@bandit:/tmp/praveen$ cat data9
The password is F05dwFsc0cbaIiH0h8J2eUks2vdTDwAn
bandit12@bandit:/tmp/praveen$
```

This is a password :- F05dwFsc@cbaIiH0h8J2eUks2vdTDwAn

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

Commands you may need to solve this level

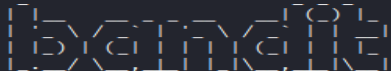
ssh, telnet, nc, openssl, s_client, nmap

Here, to get access to the next level, we will need an SSH key which can be used to get access to level 14. In the home directory, we can find the said ssh key by running `ls`.

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

We can then copy this file to our local system and change its permission to 600, which is required for ssh private keys to work. We can then use this key to login as *bandit14* using the command '`ssh -i id_rsa bandit14@bandit.labs.overthewire.org -p 2220`'. Here '*id_rsa*' file has our private key.

```
-rw-r--r-- 1 root    root      4096 Aug 15 13:10 .  
-rw-r--r-- 1 root    root       220 Mar 31 2024 .bash_logout  
-rw-r--r-- 1 root    root     3851 Aug 15 13:09 .bashrc  
-rw-r--r-- 1 root    root      807 Mar 31 2024 .profile  
-rw-r----- 1 bandit14 bandit13 1679 Aug 15 13:15 sshkey.private  
bandit13@bandit:~$ sudo su  
sudo: /usr/bin/sudo must be owned by uid 0 and have the setuid bit set  
bandit13@bandit:~$ sudo  
sudo: /usr/bin/sudo must be owned by uid 0 and have the setuid bit set  
bandit13@bandit:~$ ssh bandit14@localhost -p 2220 -i sshkey.private  
The authenticity of host '[localhost]: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  
Could not create directory '/home/bandit13/.ssh' (Permission denied).  
Failed to add the host to the list of known hosts (/home/bandit13/.ssh/known_hosts).
```



The logo consists of the words "OverTheWire" in a stylized, outlined font. The letters are interconnected, with the 'O's being particularly large and circular.

This is an OverTheWire game server.
More information on <http://www.overthewire.org/wargames>

!!! You are trying to log into this SSH server with a password on port 2220 from localhost.

```
For more information regarding individual wargames, visit
http://www.overthewire.org/wargames/

For support, questions or comments, contact us on discord or IRC.

Enjoy your stay!

bandit14@bandit:~$
```

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.

Commands you may need to solve this level

ssh, telnet, nc, openssl, s_client, nmap

The webpage for this level says “The password for the next level can be retrieved by submitting the password of the current level to **port 30000 on localhost**.” So we first need to find the password of *bandit14* as we only have a private key and not the actual password for this user. Since we are the owner of the file “**etc/bandit_pass/bandit14**”, we can read the contents of this file and retrieve the password.

```
bandit14@bandit:~$ ls
bandit14@bandit:~$ ls -la
total 24
drwxr-xr-x  3 root root 4096 Aug 15 13:15 .
drwxr-xr-x 150 root root 4096 Aug 15 13:18 ..
-rw-r--r--  1 root root  220 Mar 31 2024 .bash_logout
-rw-r--r--  1 root root 3851 Aug 15 13:09 .bashrc
-rw-r--r--  1 root root  807 Mar 31 2024 .profile
drwxr-xr-x  2 root root 4096 Aug 15 13:15 .ssh
bandit14@bandit:~$ whoami
bandit14
bandit14@bandit:~$ cat /etc/bandit_pass/bandit14
MU4VWeTyJk8R0of1qqmcBPALh7LDCPvS
bandit14@bandit:~$ ls
bandit14@bandit:~$ nc localhost 30000 MU4VWeTyJk8R0of1qqmcBPALh7LDCPvS
nc: port number invalid: MU4VWeTyJk8R0of1qqmcBPALh7LDCPvS
bandit14@bandit:~$ nc localhost 30000
MU4VWeTyJk8R0of1qqmcBPALh7LDCPvS
Correct!
8xCjnmgoKbGLhHFAZlGE5Tmu4M2tKJQo
```

Now that we have *bandit14*'s password, all we have to do is connect to the localhost at port 30000 and supply the password. We can use **netcat** to open a connection to our localhost. **Netcat** is a tool that can be used for a variety of things including port scanning and listening for traffic. We can connect to the localhost by running the command '**nc localhost 30000**' and we get the password for *bandit15*!

Bandit Level 15 → Level 16

Level Goal

The password for the next level can be retrieved by submitting the password of the current level to port 30001 on localhost using SSL/TLS encryption.

Helpful note: Getting “DONE”, “RENEGOTIATING” or “KEYUPDATE”? Read the “CONNECTED COMMANDS” section in the manpage.

Commands you may need to solve this level

ssh, telnet, nc, ncat, socat, openssl, s_client, nmap, netstat, ss

To get the password for the next level, we will have to connect to the localhost over SSL on port 30001 and submit the current user's password. We can do this by using “**openssl**”. We run the command '**openssl s_client -connect localhost:30001**'. This opens up the connection to the localhost. We then provide the password and retrieve *bandit16*'s password.

```

bandit15@bandit:~$ openssl s_client -connect localhost:30001
CONNECTED(00000003)
depth=0 CN = localhost
verify error:num=18:self signed certificate
verify return:1
depth=0 CN = localhost
verify return:1
---
Certificate chain
 0 s:/CN=localhost
  i:/CN=localhost
---
Server certificate
-----BEGIN CERTIFICATE-----
MIICBjCCAW+gAwIBAgIEc/JorzANBgkqhkiG9w0BAQUFADAUMRIwEAYDVQDDAlsb2NhbgHvc3QwHhcNMjEwMzI2MTEwNzE2WjAUMRIwEAYDVQDDAlsb2NhbgHvc3QwZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAL4S7GJ6MMkd9sGjXvizaYVKxlXF2yR6rEfo2lm4WLI/oJqAl2ZImgrFR0tCATWRgSjSuGTFbQRLlXzMYXtGNUOnU3cYncuJYwXFUCCTD7FcqTjS3y7x/YXcLAnZQPL42SEJDTQ53AYXMECP2rzv7z2H6qMZxsVJEUq6FFAzS4GPAgMBAAGjZTBjMBQGA1UdEQQNMAUCWxvY2FsaG9zdDBLBglghkgBhvhCAQ0EPhY8QXV0b21hdGJjYWxseSBnZW5lcmF0ZWQgYnkgTmNhdc4gU2VlIGh0dHBzOi8vbW1hcC5vcmcvbmNhdc8uMA0GCSqGSIb3DQEBBQUAA4GBAAhub5wBj/cjV07NGPA02dntrDCct8De4jXMRM+Ww8UtBu03yQEtFQuzp2BkeuYkXWbLjVTJJ10Y2h13y3eMpEp5wwEgdxWg11z0AyQYwzXqzxmb96mBrfYEzN8S83bvtY7M6ig4psS1FtWe0XQmR27Zet1Exzz/AuNagbJjppQBQq
-----END CERTIFICATE-----
subject=/CN=localhost
issuer=/CN=localhost
---
No client certificate CA names sent
Peer signing digest: SHA512
Server Temp Key: X25519, 253 bits
---
SSL handshake has read 1019 bytes and written 269 bytes
Verification error: self signed certificate

```

```

... 00c0 - e0 cb 98 fe b9 4b e4 f8-ec 6a 59 d9 b8 06 23 77 .....K...jY..
.#w 00d0 - 5a 03 75 a4 34 a9 5c 0a-be 60 2a f0 72 e5 db ec Z.u.4.\...`*.r
...

Start Time: 1758210867
Timeout : 7200 (sec)
Verify return code: 18 (self-signed certificate)
Extended master secret: no
Max Early Data: 0
---
read R BLOCK
8xCjnmgoKbGLhHFAZlGE5Tmu4M2tKJQo
Correct!
kSkvUpMQ7lBYyCM4GBPVCvT1BfWRy0Dx
closed
bandit15@bandit:/etc/bandit_pass$

```

Bandit Level 16 → Level 17

Level Goal

The credentials for the next level can be retrieved by submitting the password of the current level to a port on localhost in the range 31000 to 32000. First find out which of these ports have a server listening on them. Then find out which of those speak SSL/TLS and which don't. There is only 1 server that will give the next credentials, the others will simply send back to you whatever you send to it.

Helpful note: Getting “DONE”, “RENEGOTIATING” or “KEYUPDATE”? Read the “CONNECTED COMMANDS” section in the manpage.

Commands you may need to solve this level

ssh, telnet, nc, ncat, socat, openssl, s_client, nmap, netstat, ss

On this level, we first have to find the correct port in the range 31000 to 32000, connect to it on localhost over SSL, and submit the current user’s password to get the password for the next level. First, we need to find the correct port to connect to. We can do this using **nmap**. Nmap is a port scanner that tells us about all the open ports and the services they are running. So we do a quick nmap scan by running the command ***nmap -p 31000–32000 localhost***. This shows us the following output.

```
bandit16@bandit:~$ ls
bandit16@bandit:~$ ls -la
total 24
drwxr-xr-x  2 root    root    4096 Aug 15 13:15 .
drwxr-xr-x 150 root    root    4096 Aug 15 13:18 ..
-rw-r----- 1 bandit16 bandit16  33 Aug 15 13:15 .bandit15.password
-rw-r--r--  1 root     root     220 Mar 31 2024 .bash_logout
-rw-r--r--  1 root     root    3851 Aug 15 13:09 .bashrc
-rw-r--r--  1 root     root     807 Mar 31 2024 .profile
bandit16@bandit:~$
```

```
bandit16@bandit:~$ nmap -p 31000-32000 localhost
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-09-19 05:43 UTC
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00014s latency).
Not shown: 996 closed tcp ports (conn-refused)
PORT      STATE SERVICE
31046/tcp  open  unknown
31518/tcp  open  unknown
31691/tcp  open  unknown
31790/tcp  open  unknown
31960/tcp  open  unknown

Nmap done: 1 IP address (1 host up) scanned in 0.06 seconds
bandit16@bandit:~$
```

We can then manually connect to each port till we find the correct one. Once we have the correct port, we can use the same openssl tool to connect to the localhost and retrieve the password in the similar fashion. The command is ***openssl s_client -connect localhost:31518***. Here we submit the current user’s password and it then gives us an RSA private key for the next user.

```

kSkvUpMQ7lBYyCM4GBpVcVt1BfWRY0Dx
bandit16@bandit:~$ echo "kSkvUpMQ7lBYyCM4GBpVcVt1BfWRY0Dx" | openssl s_client -connect localhost:31518 -quiet
Can't use SSL_get_servername
depth=0 CN = SnakeOil
verify error:num=18:self-signed certificate
verify return:1
depth=0 CN = SnakeOil
verify return:1
kSkvUpMQ7lBYyCM4GBpVcVt1BfWRY0Dx
bandit16@bandit:~$ echo "kSkvUpMQ7lBYyCM4GBpVcVt1BfWRY0Dx" | openssl s_client -connect localhost:31790 -quiet
Can't use SSL_get_servername
depth=0 CN = SnakeOil
verify error:num=18:self-signed certificate
verify return:1
depth=0 CN = SnakeOil
verify return:1
Correct!
-----BEGIN RSA PRIVATE KEY-----
MIIEogIBAAKCAQEAvm0kuiFmMg6HL2YPIOjon6iWfbp7c3jx34YkYwQYUH57SudyJ
imZzeyGC0gtZPGUjUSxiJ5WI/oTqexh+cAMTSMlOJf7+BrJ0bArnxd9Y7YT2bRPQ
Ja6Lzb5S8YW3FZl87ORiO+rW4LDCNd2lUvLE/GL2GWyuKN0K5iCd5TbtJzEkQTu
DSt2mcNn4rHAL+JFr56o4T6z8WWAW18BR6yGrMq7Q/kALHYW3OeKePQAZL0VUYBW
JGTi65CxbCnzc/w4+mqQyvmzpWtMAZJTzAzQxNbkR2MBGySxDLrjg0LWN6sK7wNX
x0YVztz/zbIkPjfkU1jHS+9EbVnj+D1XFOJuaQIDAQABAoIBABagpxpM1aoLWfvD
KhCj10nqcoBc4oE1laFYQwik7xFW+24pRNuDE6SFthOar69jp5RlLwD1NhPx3iBl
J9nOM8OJ0VTOum43UOS8YxF8WwhXrlyGnc1sskbwpXOUDc9uX4+UESzH22P29ovd
d8WEY0gPxun8pbJLmxkAtWNhpMvfe0050vk9TL5wqbu9AlbssgTcCXkMQnPw9nC
YNN6DDP2lbcBrvgT9YCNL6C+2KufD52yOQ9gQKwFTEQpJtF4uNtJom+asvLpmS8A
vLY9r60wYSvmZhhNqBURj7lYcTXMiutlkkdAw7F77k+dJH0AXyxcUpIDGL51sOmama
+TOWWgEcgyEA8JtPXP0GRJ+IQkX262jM3dEIkza8ky5moIwUqYdsx0NxHgRRhORT
8c8hAuRBB2G82so8vUHK/fur850Efc9TncnCY2crpoqsgHfKLxrlGtT+qDpfZnx
-----END RSA PRIVATE KEY-----

```

```

bandit16@bandit:~$ cd /tmp
bandit16@bandit:/tmp$ mkdir passwd17
bandit16@bandit:/tmp$ cd passwd17
bandit16@bandit:/tmp/passwd17$ nano rsafile
Unable to create directory /home/bandit16/.local/share/nano/: No such file or directory
It is required for saving/loading search history or cursor positions.

bandit16@bandit:/tmp/passwd17$ ls -la
total 1284
drwxrwxr-x  2 bandit16 bandit16  4096 Sep 18 08:12 .
drwxrwx-wt 1629 root      root    1302528 Sep 18 08:12 ..
-rw-rw-r--  1 bandit16 bandit16  1675 Sep 18 08:12 rsafile
bandit16@bandit:/tmp/passwd17$ chmod 600 rsafile
bandit16@bandit:/tmp/passwd17$ ls
rsafile
bandit16@bandit:/tmp/passwd17$ ls -la
total 1284
drwxrwxr-x  2 bandit16 bandit16  4096 Sep 18 08:12 .
drwxrwx-wt 1630 root      root    1302528 Sep 18 08:14 ..
-rw-rw-r--  1 bandit16 bandit16  1675 Sep 18 08:12 rsafile
bandit16@bandit:/tmp/passwd17$ ssh -i rsafile bandit17@localhost -p 2220
The authenticity of host '[localhost]:2220 ([127.0.0.1]:2220)' can't be established.
ED25519 key fingerprint is SHA256:C2ihUBV7ihhVlWUXRb4RrEclFXC5CXlhmAAM/ureryLY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes^[F^[[5~

```

```

bandit16@bandit:~$ cd /tmp
bandit16@bandit:/tmp$ mkdir passwd17
bandit16@bandit:/tmp$ cd passwd17
bandit16@bandit:/tmp/passwd17$ nano rsafile
Unable to create directory /home/bandit16/.local/share/nano/: No such file or directory
It is required for saving/loading search history or cursor positions.

bandit16@bandit:/tmp/passwd17$ ls -la
total 1284
drwxrwxr-x  2 bandit16 bandit16  4096 Sep 18 08:12 .
drwxrwx-wt 1629 root      root    1302528 Sep 18 08:12 ..
-rw-rw-r--  1 bandit16 bandit16  1675 Sep 18 08:12 rsafile
bandit16@bandit:/tmp/passwd17$ chmod 600 rsafile
bandit16@bandit:/tmp/passwd17$ ls
rsafile
bandit16@bandit:/tmp/passwd17$ ls -la
total 1284
drwxrwxr-x  2 bandit16 bandit16  4096 Sep 18 08:12 .
drwxrwx-wt 1630 root      root    1302528 Sep 18 08:14 ..
-rw-rw-r--  1 bandit16 bandit16  1675 Sep 18 08:12 rsafile
bandit16@bandit:/tmp/passwd17$

```


Bandit Level 17 → Level 18

Level Goal

There are 2 files in the homedirectory: passwords.old and passwords.new. The password for the next level is in passwords.new and is the only line that has been changed between passwords.old and passwords.new

NOTE: if you have solved this level and see 'Byebye!' when trying to log into bandit18, this is related to the next level, bandit19

Commands you may need to solve this level

cat, grep, ls, diff

In the home directory of *bandit17*, there are two files with almost the same content except for one line. That line has the password for the next level. We can find this line by using a tool called “diff”. We can run the command ‘*diff passwords.old passwords.new*’. This will give us the password for the next level.

```
bandit17@bandit:~$ ls
passwords.new  passwords.old
bandit17@bandit:~$ diff passwords.new passwords.old
42c42
< x2gLTTjFwMOhQ8oWNbMN362QKxfRqGLO
---
> CgmS55GVLEKTgx8xpW8HuWnHlBKP924b
bandit17@bandit:~$
```

Bandit Level 18 → Level 19

Level Goal

The password for the next level is stored in a file readme in the homedirectory. Unfortunately, someone has modified .bashrc to log you out when you log in with SSH.

Commands you may need to solve this level

ssh, ls, cat

This is an interesting challenge! When you login with *bandit18*'s credentials, we get access, see a “**Byebye !**” message and the session terminates. This is because the configuration in .bashrc is such that it will log you out when you try to log in. After looking around for a while, I figured out that ssh lets you execute a command when trying to open a connection. Amazing! So even if our shell terminates, it will retrieve the results of the command we execute. So Let's try to list files in the home directory first. We run the command

'ssh bandit18@bandit.labs.overthewire.org -p 2220 ls'. Here as you can see, I am trying to execute the **'ls'** command. Let's see what this does.

```
(kali㉿kali)-[~]
$ sshpass -p "x2gLTtjFwMOhQ8oWNbMN362QKxfRqGLO" ssh bandit18@bandit.labs.overthewire.org -p 2220 'cat readme'

      _____
     |   _   _   |
     |  ( ) ( )  |
     |  _   _   |
     | |   |   | |
     | |___|___| |
     |___|___|___|

This is an OverTheWire game server.
More information on http://www.overthewire.org/wargames

backend: gibson-1
cGWpMaXXVwDUNgPAVJbWYuGHVn9z13j8
```

ssh executed the 'ls' command and displayed the "readme" file in the home directory. Sweet! So this means we can also read the contents of this file. We run the command 'ssh **bandit18@bandit.labs.overthewire.org** -p 2220 cat readme' and voila! We get the password for *bandit19*.

Bandit Level 19 → Level 20

Level Goal

To gain access to the next level, you should use the `setuid` binary in the `homedirectory`. Execute it without arguments to find out how to use it. The password for this level can be found in the usual place (`/etc/bandit pass`), after you have used the `setuid` binary.

Now to figure out what this binary does, we have to execute it. So we run the command `./bandit20-do`. On running this command, the binary tells us to “run a command as another user”. So we basically have to execute this binary along with a command of another user. So let’s try to print *bandit20*’s password as that user is the owner of this binary. we run the command `./bandit20-do cat /etc/bandit_pass/bandit20`. This gives us the password for the user *bandit20*.

```
bandit19@bandit:~$ ls -la
total 36
drwxr-xr-x  2 root    root      4096 Aug 15 13:16 .
drwxr-xr-x 150 root    root      4096 Aug 15 13:18 ..
-rwsr-x---  1 bandit20 bandit19 14884 Aug 15 13:16 bandit20-do
-rw-r--r--  1 root    root       220 Mar 31 2024 .bash_logout
-rw-r--r--  1 root    root      3851 Aug 15 13:09 .bashrc
-rw-r--r--  1 root    root       807 Mar 31 2024 .profile
bandit19@bandit:~$ ./bandit20-do
Run a command as another user.
Example: ./bandit20-do id
bandit19@bandit:~$ ./bandit20-do id
uid=11019(bandit19) gid=11019(bandit19) euid=11020(bandit20) groups=11019(bandit19)
bandit19@bandit:~$ ./bandit20-do cat /etc/bandit_passbandit20
cat: /etc/bandit_passbandit20: No such file or directory
bandit19@bandit:~$ ./bandit20-do cat /etc/bandit_pass/bandit20
0qXahG8ZjOVMN9Ghs7iOWsCfZyXOUbYO
bandit19@bandit:~$
```

```
bandit20@bandit:~$
```

Bandit Level 20 → Level 21

Level Goal

There is a `setuid` binary in the `homedirectory` that does the following: it makes a connection to `localhost` on the port you specify as a commandline argument. It then reads a line of text from the connection and compares it to the password in the previous level (`bandit20`). If the password is correct, it will transmit the password for the next level (`bandit21`).

NOTE: Try connecting to your own network daemon to see if it works as you think

Commands you may need to solve this level

`ssh`, `nc`, `cat`, `bash`, `screen`, `tmux`, Unix 'job control' (`bg`, `fg`, `jobs`, `&`, `CTRL-Z`, ...)

In this level the `setuid` binary provided doesn't allow for direct shell commands however it allows for a socket connection on whichever port you choose. Opening a second terminal and connecting it to the `localhost` on any user will allow for a listener using the `nc` command. After the listener is open use the binary with that port number then go back to the listener and submit the previous password to get the new

```
bandit20@bandit:~$ cat /etc/bandit_pass/bandit20
0qXahG8ZjOVMN9Ghs7iOWsCfZyX0UbY0
bandit20@bandit:~$
```

```
bandit20@bandit:~$ ls
suconnect
bandit20@bandit:~$ ./suconnect 2222
Read:
ERROR: This doesn't match the current password!
bandit20@bandit:~$ ./suconnect 2222
Read: 0qXahG8ZjOVMN9Ghs7iOWsCfZyX0UbY0
Password matches, sending next password
bandit20@bandit:~$
```



```
bandit20@bandit:~$ cat /etc/bandit_pass/bandit20
0qXahG8Zj0VMN9Ghs7iOWsCfZyX0UbyO
bandit20@bandit:~$ nc -lvp 2222
Listening on 0.0.0.0 2222
Connection received on localhost 51440

FAIL!
bandit20@bandit:~$ nc -lvp 2222
Listening on 0.0.0.0 2222
Connection received on localhost 53382
0qXahG8Zj0VMN9Ghs7iOWsCfZyX0UbyO
EeoULMCra2q0dSkyj561DX7s1CpBu0Bt
bandit20@bandit:~$ █
```

```
bandit21@bandit:~$
```