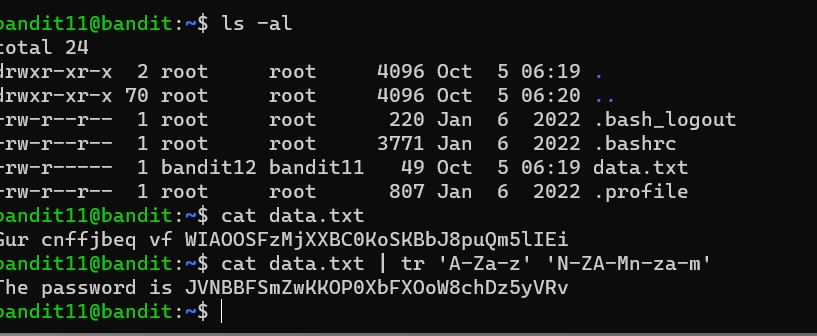
LEVEL 11🡪 LEVEL 12

*#CHALLENGE*

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.

*#SOLUTION*



1. First, we list all the directories by using ‘**ls-al’**.
2. Then on getting data.txt we list it contents using ‘**cat’** command.
3. But it is encoded in **Rot13**, to decode it we use ‘**tr**’ command.
4. tr accepts two sets of characters, usually with the same length, and replaces the characters of the first sets with the corresponding characters from the second set. Here it replaces all letters in the first set that is A-Z to N-Z then A-M same with lower case. Each letter is replaced by its 13 letter.

*#FLAG*

JVNBBFSmZwKKOP0XbFXOoW8chDz5yVRv

*#REFERENCE:*

<https://linuxize.com/post/linux-tr-command/?source=post_page-----df6e59deda05-------------------------------->

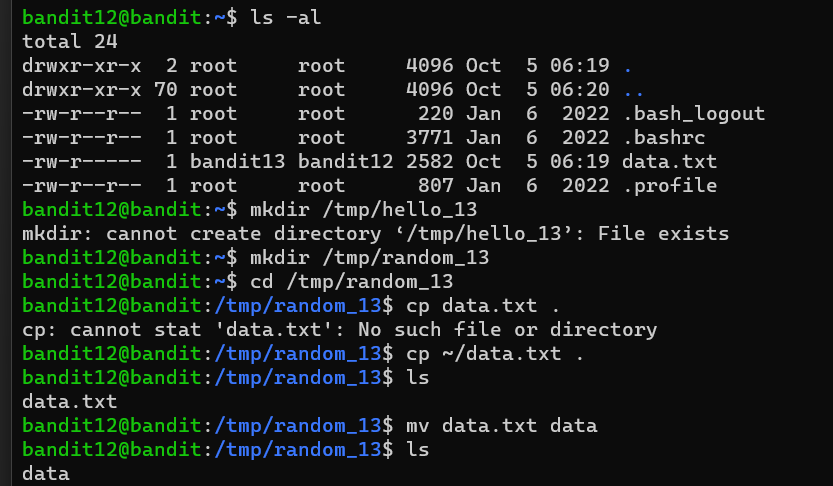
LEVEL 12🡪 LEVEL 13

*#CHALLENGE*

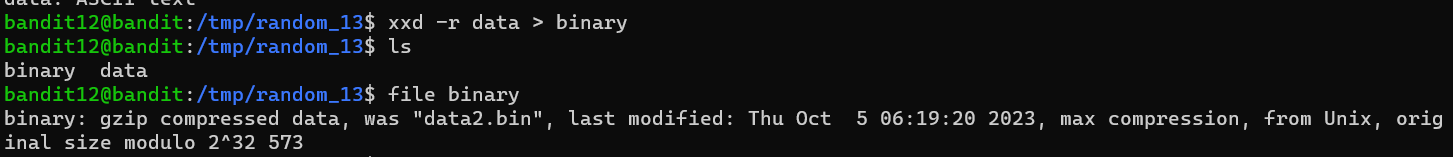
The password for the next level is stored in the file **data.txt**, which is a hexdump of a file that has been repeatedly compressed. For this level it may be useful to create a directory under /tmp in which you can work using mkdir. For example: mkdir /tmp/myname123. Then copy the datafile using cp and rename it using mv (read the manpages!)

*#SOLUTION*

1. As stated in the question file data.txt is a hexdump. For that we must convert the data to binary form, for that we first need to create a temporary working directory in the **/tmp** directory as we do not have permission to create new files in the current location. We can do this using the **mkdir** command. To move into the new directory, we can use the **cd** command. We now need to move data.txt to this new location. We can do this using the **cp** command. And then we rename the file to remove the **.txt** extension with **mv** command as we know the file is not a text file.

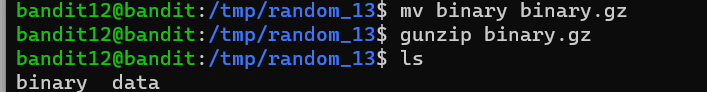


1. Now that the data is in the new directory, we can now use **xxd** to convert the data into its binary equivalent. After this we can use the **file** command to see what type of data is stored in the file.



1. **Gzip Decompression**

We can see that the file was compressed using qzip so we can decompress the data using the **gunzip** or **gzip -d** command. When trying to decompress a gzip file it is important that the file has the correct extension. Thus, for this we rename the file using **mv** command to **.gz** extension.



1. **Bzip Decompression**

Using the **file** command, we can again look at the type of the data that is stored in the file .We see that the data is compressed using **bzip2.** For decompressing a bzip2 file we can use the **bunzip2** command or **bzip2 -d.**

A computer screen shot of white text

Description automatically generated

1. Using the **file** command, we can again look at the type of the data that is stored in the file. We can see that the file was compressed thus we repeat **step 3.**

**NOTE:** to decompress we need to change the extension of the file to **.gz**.

A black screen with white text

Description automatically generated

1. **Tar Archive**

Using the **file** command, we can again look at the type of the data that is stored in the file. We see that the data is saved in a **tar** archive. For extracting a tar file we use the**tar** command. The **-x** flag is used to extract one or more items from the file. The **-f** flag is used to specify the filename.

A screen shot of a computer

Description automatically generated

1. It looks like the password file has recursively been compressed using “**tar”**, “**gzip**” and “**bzip2**”. We keep repeating the above steps tell we get the password file.

A screen shot of a computer

Description automatically generated

*#FLAG*

wbWdlBxEir4CaE8LaPhauuOo6pwRmrDw

*#REFERENCE:*

<https://david-varghese.medium.com/overthewire-bandit-level-12-level-13-2ec761a88907>

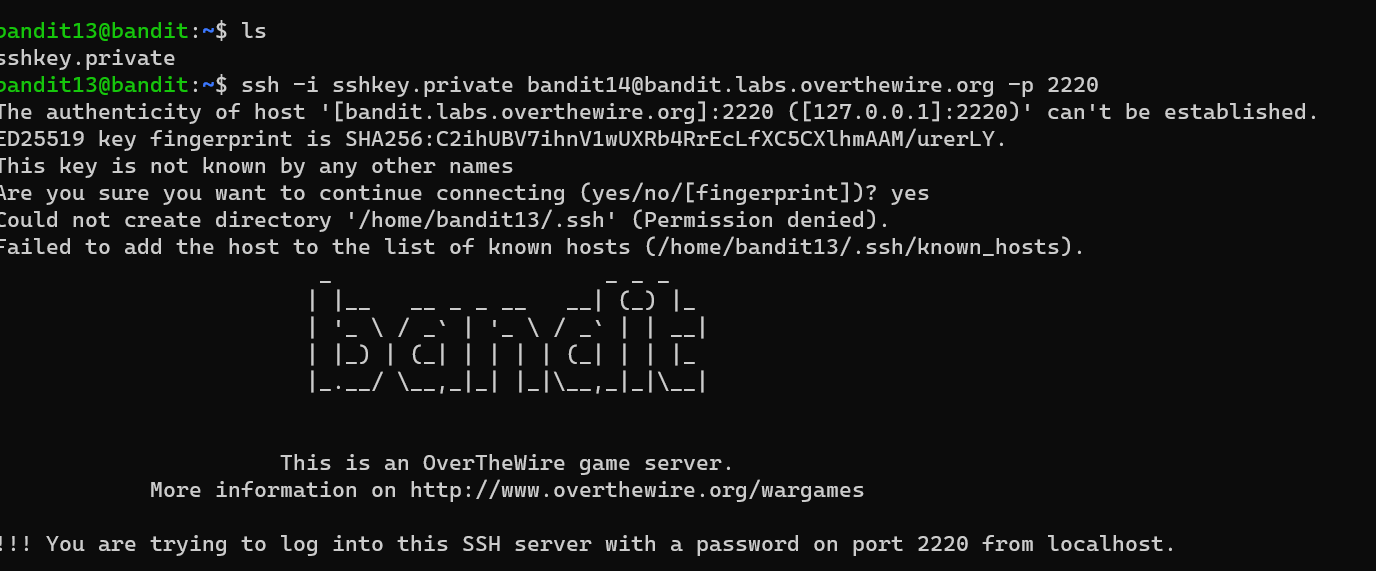
<https://phoenixnap.com/kb/tar-command-in-linux>

<https://askanydifference.com/difference-between-gzip-and-bzip2-with-table/>

<https://mayadevbe.me/posts/overthewire/bandit/level13>

LEVEL 13🡪 LEVEL 14

*#CHALLENGE*

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. **

*#SOLUTION*

*A black background with white text

Description automatically generated*

1. First, we list the contents of the directory using **ls.**

1. We have an SSH private key. We can use the SSH command with the**-i**flag to use the private key.
2. We have logged in as bandit14 we can confirm this by looking at your prompt. Get the password using **cat** command. Here localhost refers to **bandit.labs.overthewire.org**.

*#FLAG*

fGrHPx402xGC7U7rXKDaxiWFTOiF0ENq

*#REFERENCE:*

<https://askubuntu.com/questions/817626/whats-the-meaning-of-i-in-ssh>

LEVEL 14🡪 LEVEL 15

*#CHALLENGE*

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

*#SOLUTION*

A black screen with white text

Description automatically generated

1. From the question we know that there is a service that is running on port 30,000. We can try to connect to the service using **netcat**command.
2. Next we need to submit the password.

*#FLAG*

jN2kgmIXJ6fShzhT2avhotn4Zcka6tnt

*#REFERENCE:*

<https://mayadevbe.me/posts/overthewire/bandit/level15/>

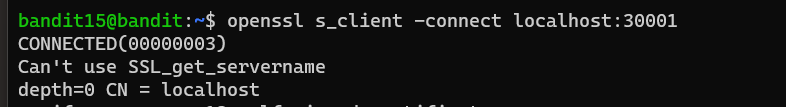
LEVEL 15🡪 LEVEL 16

*#CHALLENGE*

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

**Helpful note: Getting “HEARTBEATING” and “Read R BLOCK”? Use -ign\_eof and read the “CONNECTED COMMANDS” section in the manpage. Next to ‘R’ and ‘Q’, the ‘B’ command also works in this version of that command…**

*#SOLUTION*

**

*A screen shot of a computer

Description automatically generated*

1. We know that we must connect to a service on port 30,001 using SSL encryption. The simplest way to achieve this is using the **openssl**command along with **s\_client** which allows to connect to services on our machine using SSL.
2. Next, we need to submit the password.

*#FLAG*

JQttfApK4SeyHwDlI9SXGR50qclOAil1

*#REFERENCE:*

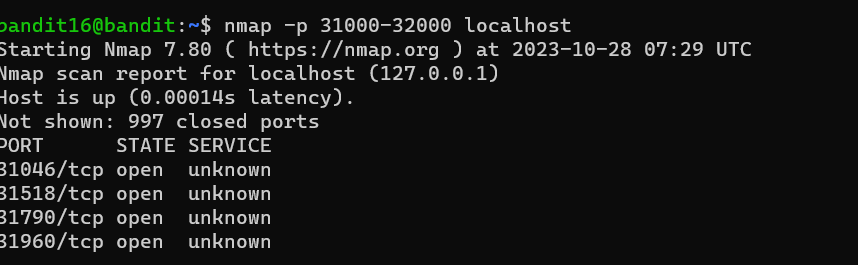
<https://david-varghese.medium.com/overthewire-bandit-level-15-level-16-aea1b72721b0>

LEVEL 16🡪 LEVEL 17

*#CHALLENGE*

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 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.

*#SOLUTION*

1. **We know there that the service that we need is running in the range 31,000–32,000. We can find all services in that range using **nmap**command.
2. We know that the service uses SSL encryption. So, we need to use the **opens sl**and **s\_client** command to connect to the port and pass the password of the current user.

*A screen shot of a computer

Description automatically generated*

1. We need to save the key in a file to be used with SSH. As we don't have permission in the current working directory to make a file. We create a folder in the **/tmp** directory and work from there. We create a new directory using **mkdir** command. After copying the key to a file, we tried to use it to login to the next level. However, a permission error was given. After changing the permission to 600 it worked as it is the required permission for RSA private keys. Thus we get logged in to the next level.

A computer screen shot of a black screen

Description automatically generated.

*#FLAG*

We don’t get a password we get RSA private key and thus we get logged in to the next level.

*#REFERENCE:*

<https://upcloud.com/resources/tutorials/use-ssh-keys-authentication?source=post_page-----c137701b3af1-------------------------------->

<https://phoenixnap.com/kb/nmap-commands>

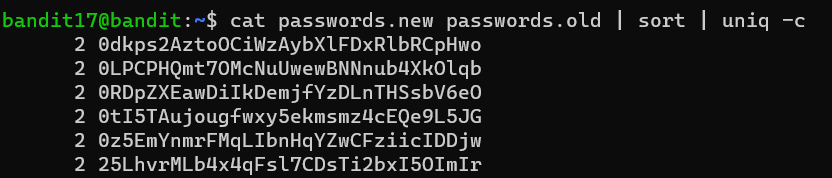
LEVEL 17🡪 LEVEL 18

*#CHALLENGE*

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**

*#SOLUTION*





1. First we concatenated the two files.
2. Using piping sorted the given concatenated file.
3. Now we use the **uniq -c** command to get the no. of times each line was repeated.

*#FLAG*

hga5tuuCLF6fFzUpnagiMN8ssu9LFrdg

*#REFERENCE:*

<https://www.geeksforgeeks.org/basic-linux-commands/>

LEVEL 18🡪 LEVEL 19

*#CHALLENGE*

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.

*#SOLUTION*



As every time **.bashrc** file closes ssh connection we can use the cat readme command along with the ssh command so that the .bashrc file is not triggered.

*#FLAG*

awhqfNnAbc1naukrpqDYcF95h7HoMTrC

*#REFERENCE:*

<https://www.geeksforgeeks.org/basic-linux-commands/>

LEVEL 19🡪 LEVEL 20

*#CHALLENGE*

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.

*#SOLUTION*

**

1. We have been told there is a binary file that is present in the home directory which somehow can help us to access the password of bandit20.
2. We can see that the file is called **bandit20-do** and when we list the details of the file, we can see that the binary file can be executed by the current user (bandit19) and it is owned by bandit20
3. To run an executable file, we just need to specify its name along with the location. The file is in the current working directory so we can use **./<filename>** to access the file
4. The file tells us that it allows us to run a command as another user. Let’s see an example of running a command as another user using the id command.
5. We observe that when we use the binary file, we are assigned the uid for bandit20 as well which means we can run commands as if we are bandit20
6. Now that we know we can run commands as bandit20 so let’s use the binary to access the password of user bandit20.

*#FLAG*

VxCazJaVykI6W36BkBU0mJTCM8rR95XT

*#REFERENCE*

<https://en.wikipedia.org/wiki/Setuid?source=post_page-----73ea3291d26b-------------------------------->

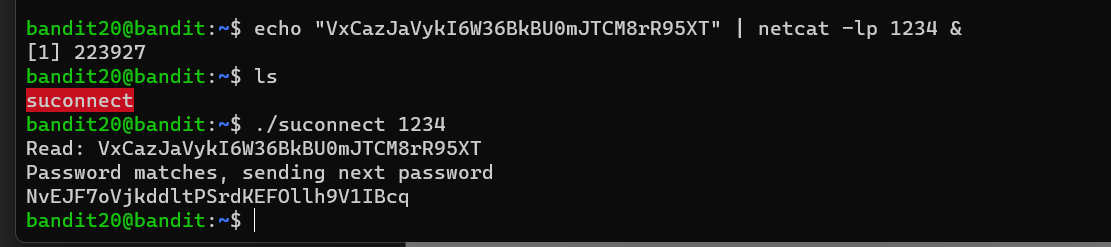
LEVEL 20🡪 LEVEL 21

*#CHALLENGE*

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 command line 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

*#SOLUTION*



1. first task that we need to do is setup a listener on any port on the system that will return the previous level password if we connect to that port using the binary file. We can setup an listener using the **netcat**command.
2. The **-l** flag is used to setup a listener and the **-p** flag is used to specify the port the listener should listen on. As we have not specified IP Address the listener is going to run on localhost. The **“&”** at the end of the command is used to specify that we want the command to run in the background.
3. Now that we have the listener setup, we can use the binary file to connect on the same port.

*#FLAG*

NvEJF7oVjkddltPSrdKEFOllh9V1IBcq

*#REFERENCE*

<https://linuxize.com/post/netcat-nc-command-with-examples/?source=post_page-----cc769247866a-------------------------------->