I spent the whole of week 3 familiarising myself with the OverTheWire platform, through completing the first challenge: Bandit. This set of challenges helped me get up to speed with the linux commands I would need for the future challenges, in the form of exercises where I had to retrieve a password to gain access to the next level.

The first few levels were ceremonious events in which my knowledge of linux commands from my comp sci studies gave me an advantage. However soon enough, I realised some holes in my knowledge of some commands, and realised that there were many commands I hadn't paid much thought to.

Bandit Level 12

One challenge that took me a fair amount of time was level 12 in which I was given a .txt file which was a hexdump of a file that was repeatedly compressed. I delved straight into using the command 'file' to figure out what sort of file this was.

```
00000070: a000 c87a 81a3 464d a8d3 43c5 1068 0346  ...z..FM..C..h.F
00000080: 8343 40d0 3400 0340 66a6 8068 0cd4 f500  .C@.4..@f..h....
00000090: 69ea 6800 0f50 68f2 4d00 680d 06ca 0190  i.h..Ph.M.h.....
000000a0: 0000 69a1 a1a0 1ea0 194d 340d 1ea1 b280  ..i......M4.....
000000b0: f500 3406 2340 034d 3400 0000 3403 d400  ..4.#@.M4...4...
000000c0: 1a07 a832 3400 f51a 0003 43d4 0068 0d34  ...24.....C..h.4
000000d0: 6868 f51a 3d43 2580 3e58 061a 2c89 6bf3  hh..=C%.>X..,.k.
000000e0: 0163 08ab dc31 91cd 1747 599b e401 0b06  .c...1...GY.....
000000f0: a8b1 7255 a3b2 9cf9 75cc f106 941b 347a  ..rU....u.....4z
00000100: d616 55cc 2ef2 9d46 e7d1 3050 b5fb 76eb  ..U....F..0P..v.
00000110: 01f8 60c1 2201 33f0 0de0 4aa6 ec8c 914f  ..`."3...J....O
00000120: cf8a aed5 7b52 4270 8d51 6978 c159 8b5a  ....{RBp.Qix.Y.Z
00000130: 2164 fb1f c26a 8d28 b414 e690 bfdd b3e1  !d...j.(........
00000140: f414 2f9e d041 c523 b641 ac08 0c0b 06f5  ../..A.#.A......
00000150: dd64 b862 1158 3f9e 897a 8cae 32b0 1fb7  .d.b.X?..z..2...
00000160: 3c82 af41 20fd 6e7d 0a35 2833 41bd de0c  <..A .n}.5(3A...
00000170: 774f ae52 a1ac 0fb2 8c36 ef58 537b f30a  wO.R.....6.XS{..
00000180: 1510 cab5 cb51 4231 95a4 d045 b95c ea09  .....QB1...E.\..
00000190: 9fa0 4d33 ba43 22c9 b5be d0ea eeb7 ec85  ..M3.C".........
000001a0: 59fc 8bf1 97a0 87a5 0df0 7acd d555 fc11  Y.........z..U..
000001b0: 223f fdc6 2be3 e809 c974 271a 920e acbc  "?..+....t'.....
000001c0: 0de1 f1a6 393f 4cf5 50eb 7942 86c3 3d7a  ....9?L.P.yB..=z
000001d0: fe6d 173f a84c bb4e 742a fc37 7b71 508a  .m.?.L.Nt*.7{qP.
000001e0: a2cc 9cf1 2522 8a77 39f2 716d 34f9 8620  ....%".w9.qm4..
000001f0: 4e33 ca36 eec0 cd4b b3e8 48e4 8b91 5bea  N3.6...K..H...[.
00000200: 01bf 7d21 0b64 82c0 3341 3424 e98b 4d7e  ..}!.d..3A4$..M~
00000210: c95c 1b1f cac9 a04a 1988 43b2 6b55 c6a6  .\.....J..C.kU..
00000220: 075c 1eb4 8ecf 5cdf 4653 064e 84da 263d  .\....\.FS.N..&=
00000230: b15b bcea 7109 5c29 c524 3afc d715 4894  .[..q.\).$:...H.
00000240: 7426 072f fc28 ab05 9603 b3fc 5dc9 14e1  t&./.(......]...
00000250: 4242 393c 7320 98f7 681d 3d02 0000       BB9<s ..h.=...
bandit12@bandit:/tmp/sonali$ file data.txt
data.txt: ASCII text
bandit12@bandit:/tmp/sonali$ xxd -r data.txt newdata
bandit12@bandit:/tmp/sonali$ file newdata
```

I used the manual to find out what xxd does, as it was a suggested command on the challenge page. It seemed like I could use it to decode this ASCII text into binary, so I used the command **xxd -r data.txt** to convert the file to binary. The file however did not convert. Doh - I forgot to save this into a new file! I saved this into a file called 'newdata' and checked to see what this new file contained.

In my haste, I checked to see what the file looked like but the words contained randomised symbols - I had further decompressing to do.

```
d[illegible]A4$[illegible]M~[illegible]gJ[illegible]kUR\[illegible]SN[illegible]=[illegible]
```

The new file was a gzip file, so I looked up gzip on the manual by typing in **man zip** and it said that gzip is a compression tool that uses the extension .gz and decompresses a file using -d.

I used the command **mv newdata newdata.gz** to add the .gz extension to the newdata file. I could then decompress this using **gzip -d newdata.gz**.

```
bandit12@bandit:/tmp/sonali$ bzip2 -d data7.bz2
bandit12@bandit:/tmp/sonali$ file data7
data7: POSIX tar archive (GNU)
bandit12@bandit:/tmp/sonali$ tar -xvf data7
data8.bin
bandit12@bandit:/tmp/sonali$ file data8.bin
data8.bin: gzip compressed data, was "data9.bin", last modified: Thu May  7 18:14:30 2020, max compression, from Unix
bandit12@bandit:/tmp/sonali$ mv data8.bin data8.gz
bandit12@bandit:/tmp/sonali$ gzip -d data8.gz
bandit12@bandit:/tmp/sonali$ file data8
data8: ASCII text
bandit12@bandit:/tmp/sonali$ cat data8
The password is 8ZjyCRiBWFYkneahHwxCv3wb2a1ORpYL
bandit12@bandit:/tmp/sonali$
```

I then check the contents of the newdata file again, and it is a bzip file. Similarly, I looked it up on the manual and found that it has a .bz2 file extension so I moved newdata to the new file extension and decompressed it. As seen in the image, this process took 8 iterations until I finally revealed that my file contained ASCII text. This made me hopeful that I could simply view the contents (while crossing my fingers that it would work).

It surprisingly did!

This challenge made me ponder about how I could also vigorously compress my passwords like the challenge did. Until recently, my passwords were merely jotted down in random notes so compressing it and utilising rot13 to rotate the text within the password would ensure that my passwords are more secure (or at least more secure than being accessible in plain sight of my desktop).

Bandit Level 19

This level took 10 minutes but I wanted to blog about it to keep note of this seemingly small command.

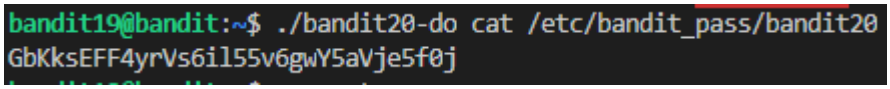
I had to gain access to the level 20 password but I do not have permissions for it. So what did I do? This is where the **setuid binary** was useful.

I used the command **ls -l** to view the file permissions, and when an executable file's setuid permission is set, users may execute that program with a level of access that matches the user who owns the file.

```
-rwsr-x--- 1 bandit20 bandit19 7296 May  7  2020 bandit20-do
```

The setuid permission is displayed as an “s” in the “user execute” bit position. The user is bandit20. So if we execute **./bandit20-do** it will run as user bandit20.

So I do this command to get the password:



Overall Reflection

Skills

While this week was not too strenuous, my skills have already improved as I am more efficiently using the linux manual to find commands that are needed for the challenge at hand. I have also learnt more about the various ways in which I can gain access to files which are encrypted, and finding files which are placed within many, and finding the one which is human readable. I am also more confident in extracting data (e..g a password) from a file that may seem too cluttered with text. Whilst I would have previously given up on finding data from a huge file, I now can filter the contents of the file to find a line containing a specific word, or sorting the lines to find ones that are repeated.

In level 14, I was asked to connect to localhost on port 3000 in order to proceed to the next level. Here my knowledge of networks from the network programming course, allowed me to skip over the pre-readings which explained what telnet is. Since I already knew that telnet can be used to connect to another host, I used the command **telnet localhost 3000** to access the next level.

Challenges

I found level 12 challenging as it took me longer than the other tasks. I decompressed the file but it was still not showing me readable ASCII text. This made me panic for a short while but then I worked in slow steps, progressively checking what type of file I had, and decompressing it many times - not giving up. My thought process is outlined above.It took about 20 minutes but my efforts had paid off when I was able to SSH into the next level.

Reflection

It is scary to think that a hacker could extract my password from my computer in a similar fashion by simply filtering the file for the word "password" or "passwd". This has made me exercise more caution in the way I store my passwords.

One of the things I learnt from this week's challenges was that the manual page is really useful and was the key factor in enabling me to solve the challenge. I often resort to googling commands on the fly, and have been doing this a lot especially in my current machine learning course where I am unfamiliar with the modules used in modelling regression. I saved so much time using the linux manual and these exercises brought me up to scratch with many commands as it refreshed my memory on compression tool commands, and commands like 'tr' which allowed me to complete a challenge in which the text was rotated 13 times - a form of the Caesar cipher.