Challenge 1

1. Level 0

We just use ssh to login onto the server . password is given . use -p on ssh to login on port 2220.

1. 0 to 1

cat readme to get password

3) 1 to 2

We cant print dashed file using cat normally. Have to use ./- to print it. Browsed how to print dashed files in resources.

4) 2 to 3

Enclosed the file name in “” as linux uses space to delimit arguments. cat “spaces in the file name”.

5) 3 to 4

Hidden files start with a . Just cd to inhere directory and the ls -la to get all files.

We see .hidden and cat it.

6) 4 to 5

Cd to inhere. Using ls -la we see all the files. Can just cat all of them . Found better solution on browser thats is to use file ./\* to get file type of all files.

7) 5 to 6

Used find command can specify file type using -f , size using -size , ! -executable to specify not executable.

8) 6 to 7

Again use find command specify its a file , specify size , -user and -group to specify user and group respectively. And put 2>/dev/null to remove error part.

9) 7 to 8

Just use less and use / (search command) to find millionth.

10) 8 to 9

Sort data.txt and pipe it into uniq command with -u flag to display only unique string.

11) 9 to 10

Used strings command to get human understandable part of dat.txt and pipe it into grep ==

(min condition for string)

12) 10 to 11

Cat data and Use base64 command with -d flag to decode it.

13) 11 to 12

I used rot13 translator online . Can be done using translate command tr ‘A-Za-z’ ‘N-ZA-Mn-za-m’.

14) 12 to 13

Copy data to new file.

Use xxd -r to reverse hexdump and put reversed hexdump in new file.

Using xxd told us its a gzip compressed file (by observing header bytes).

Renamed file to put .gz on file then used gzip -d to reverse compression.

Again using xxd we see its now a bzip2 compressed file (425a in header bytes).

Rename to put .bz2 in end and use bzip2 -d to decompress.

Again its gzip . decompress

Now when we use xxd we see data5.bin . use tar -xf to unarchive.

Now when we use xxd we see data6.bin . use tar -xf to unarchive.

Again its bzip2 . decompress

There is archive . Use tar -xf to get data8.bin

Finally its gzip decompress to get password.

Had to browse forums to understand decompressing.

15)13 to 14

We modify permission of private sshkey file by chmod 700.

Then use -i flag of ssh to use sshkey.private(file containing private sshkey ) to logon to local host.

Afterwards we can just use /etc/bandit\_pass/bandit14 to get password for level 14.

16)14 to 15

Used nc command to open communication . nc localhost 30000

Afterwards just paste password and enter.

17) 15 to 16

Used openssl s\_client -connect localhost:30001 to establish connection.

Then submitted password to get next level key.

18)16 to 17

Nmap localhost -p 31000-32000 to get server communicating and speaking ssl.

Then like previous used openssl to connect , enter password and get private SSH key.

Save key in file , modify permission then use -i flag of ssh to login to next level.

19)17 to 18

Used diff command suggested in question to compare both files. Get password and login/

20)18 to 19

Use -t to force pseudo terminal allocation.

ssh [bandit18@bandit.labs.overthewire.org](mailto:bandit18@bandit.labs.overthewire.org) -p 2220 -t /bin/sh

Afterwards ust cat the file containig password.

21)19 to 20

Ls -la tells us the permission for bandit20-do . We can see that due to SETUID we can run bandit20-do as bandit 19.(bandit 19 has executing permission). We use ./bandit20-do

To read /etc/bandit\_pass/bandit20 and get password.

22)20 to 21

Echo password of level 20 and pipe it to nc -l -p 1234 &

It will return next level password.

-l flag is used as it has to listen to string and compare it to our input and & is used to run it in background while we now run ./suconnect 1234 to connect to our netcat server , receive inputted password and give next level password.

23)21 to 22

I used ls -la on /etc/cron.d to see /etc/cron.d folder

Read the cronjob\_bandit22.sh file and we see that it creates a file in tmp folder and stores next levels password in it. Read it to get password.

24)22 to 23

I used ls -la on /etc/cron.d to see /etc/cron.d folder

Read the cronjob\_bandit23.sh file and we see that it has 2 variables: myname and mytarget.

Myname = bandit 23 as that's what we get from whoami command(script is run as bandit 23)

Mytarget gives us the file in which the password is stored.

Cat the file to get the password.

25)23 to 24

I used ls -la on /etc/cron.d to see /etc/cron.d folder

Read the cronjob\_bandit24.sh file and we see that when executed it deletes all files in folder

var/spool/bandit24/foo while ensuring current and previous folder is not deleted.

Its executed every minute .(exploring told thats what the 5 starts in cronjob\_bandit24 signify.)

I created a new directory in /tmp and created a new file in it.(call new file kek)

Cd to new directory and create a vim file(call it new.sh) which has a script which prints password to kek.

Adjust permissions such that there is no issue in copying files.

Give executable permission to kek and new.sh.

Give max permission to our temporary directory.

Print kek to get password.

26)24 to 25

We need to create a vim file which will continuously input possible combinations to the

Daemon.

guy=(password of level 24)

We create a loop for i in {0000..9999}

Do

Echo “$guy $i”

Done | nc localhost 30002

Give executable permission to vim file and run it in a /tmp created new directory

It stops when we get combination correct.

I had to check online how to run loops in shell script.

27) 25 to 26

Cat /etc/passwd and search for bandit 26

Instead of default shell we see /usr/bin/showtext

Running ls on it gives us a script.

In this when more command runs on text.txt it results in exit as in we are logged out.

In home directory we find private SSH key for bandit 26. We adjust its permissions to prevent issues while logging in.

We use private sshkey to login ssh -i bandit26.sshkey bandit26@localhost -p 2220

To prevent more from executing minimise screen .

Use v to go into vim and then use vims shell commands :set shell to return to default /bin/bash

Now we can access /etc/bandit\_pass to get password.

Couldnt solve it at all had to check online for complete solution and understand its procedure.

28) 26 to 27

Ls shows us bandit27-do file.

Run command as user bandit27 to access /etc/bandit\_pass/bandit27 and get password.

29)27 to 28

Make random directory in /tmp and use git clone.

Cd to repo and cat readme to get password.

30)28 to 29

Same as previous. However file doesnt have password.

We use git log and git show(commit) to see past commits.

Commit Add missing data has required password.

31)29 to 30

Again file didnt have password.

Git log had nothing so we used git branch -a to see all branches available in this repository.

Switch to other branches.(git switch)

Dev branch had required password.

Had to search online to get hints..

32)30 to 31

Had to use git tag and then git show secret to get password.

33)31 to 32

Clone git and cd to repo

Create key.txt and put required string in it.

However we cant commit and push it due to .gitignore which prevents .txt from being pushed by git

We have to use git add -f key.txt then commit and push again.

Had to search online due to lack of experience with git. Planning to learn using it.

34)32 to 33

Lowercase commands werent working as everything was executed in uppercase.

Use $0 to return to normal shell . Upon returning ls shows uppershell file runs as bandit33

As a result we can see password.

I was able to solve initial half questions by linux knowledge + exploring suggested commands.

However had to use online help in git questions and some escape shell ones.

Challenge 2

Q1) russian roulette

import random

chambers = 6

bullet\_chamber = random.randint(1,7)

for x in range(1,7):

input('press enter to fire:')

if x == bullet\_chamber:

print("you died")

print("press play to start again")

break

else:

print("you survived!")

continue

(i took standard chambers as 6 as in original game. Can take input if needed and change range to (1,chambers+1) and randint to (1,chambers+1)

Q2) in C (selection sort)

#include<stdio.h>

void swap(float \*a,float \*b)

{

float t= \*a; #swap function

\*a=\*b;

\*b=t;

}

int main()

{

int n;

printf("enter number of elements:");

scanf("%d",&n);

float arr[n];

for(int i=0;i<n;i++)

{

printf("enter element :");

scanf("%f",&arr[i]);

}

int min;

for(int i=0;i<n-1;i++)

{

min=i;

for(int j=i+1;j<n;j++) #nested loops to find min element in search area

{ and swap it with our selected element.

if(arr[min]>arr[j])

min=j;

}

if (min != i)

swap(&arr[i],&arr[min]);

}

for(int i=0;i<n;i++)

{

printf("%f ",arr[i]);

}

}

In pytho

file1=open("work.txt","r")

print(file1.read())

file1.close()

I am creating list in which i store tasks . Afterwards i open file write list int it and close it.

3)b) and task 4 are leftn (selection sort)

arr = []

n = int(input("Enter number of elements : "))

for i in range(0, n):

ele = float(input())

arr.append(ele) #creating array of elements.

for i in range (len(arr)-1):

min = i

for j in range(i+1,len(arr)):

if arr[min]> arr[j]:

min =j

arr[i],arr[min] = arr[min],arr[i] #sorting

for i in range (len(arr)):

print(arr[i], end =" ")

Q3)i) for single user

arr = []

n = int(input("Enter number of tasks : "))

for i in range(0, n):

work = (input())

arr.append(work)

arr[i]=str(i)+") " + arr[i]+ " \n"

text\_file = open("work.txt","a")

text\_file.writelines(arr)

text\_file.close()