level 0:

```
ssh -p [port number] username@hostname
ssh -p 2220 bandit0@bandit0.labs.overthewire.org
```

Bandit Level 0 → 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.

solution
since we're already in home directory ~, use ls -a list all files, there was only 1 file readme
use cat readme to print the file's content
ZjLjTmM6FvvyRnrb2rfNWOZOTa6ip5If

Bandit Level 1 → Level 2

Level Goal

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

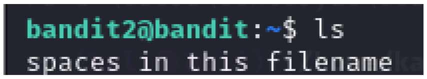
solution
use ls will see only one file named -
use command cat ./- to print out the content of the file. The ./ prefix explicitly tells the shell to treat - as a file in the current directory. Without the ./ prefix, cat - waits for input from user and treat that input as a placeholder for stdin.

263JGJPfgU6LtdEvgfWU1XP5yac29mFx

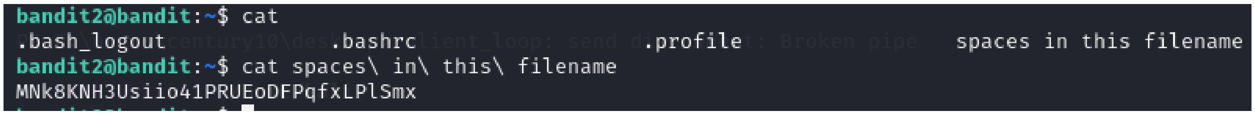
Bandit Level 2 → Level 3

Level Goal

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



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



```
bandit2@bandit:~$ cat
.bash_logout           .bashrc           .profile                spaces in this filename
bandit2@bandit:~$ cat spaces\ in\ this\ filename
MNk8KNH3Usiio41PRUEoDFPqfxLPlSmx
```

Using the command cat spaces\ in\ this\ filename, we get the content of the specified file:
MNk8KNH3Usiio41PRUEoDFPqfxLPlSmx
We an escape spaces with backlashes \
Or a shortcut is typing in cat and the first letter of the file's name, for example "s", then hit Tab.

Bandit Level 3 → Level 4

Level Goal

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

solution
First, I used the ls command to list out all the directories in home, and there was only one directory "inhere". Now, I list out all the files in this directory with ls -a command to make sure I can see all the hidden files. There was only one hidden file, so using cat I could get the content of it.

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

Alternatively, we can use ls -aR to list out everything including directories and files from the ~ directory:

```
bandit3@bandit:~$ ls
inhere
bandit3@bandit:~$ ls -aR
.:
.  ..  .bash_logout  .bashrc  inhere  .profile

./inhere:
.  ..  ...Hiding-From-You
```

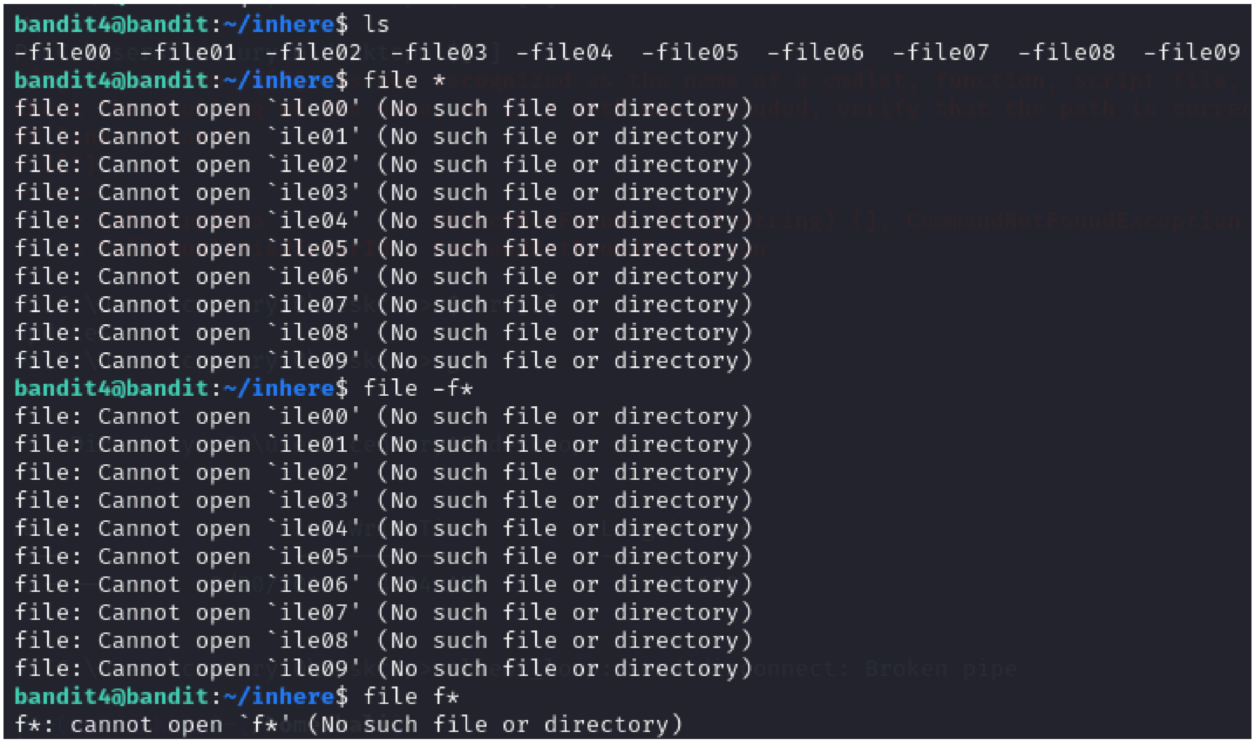
Bandit Level 4 → Level 5

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.

solution

Use the file command to determine the type of each file. Let’s see what happened when I tried to run the file * command with the wildcard *



```
bandit4@bandit:~/inhere$ ls
-file00  -file01  -file02  -file03  -file04  -file05  -file06  -file07  -file08  -file09
bandit4@bandit:~/inhere$ file *
file: Cannot open `ile00' (No such file or directory)
file: Cannot open `ile01' (No such file or directory)
file: Cannot open `ile02' (No such file or directory)
file: Cannot open `ile03' (No such file or directory)
file: Cannot open `ile04' (No such file or directory)
file: Cannot open `ile05' (No such file or directory)
file: Cannot open `ile06' (No such file or directory)
file: Cannot open `ile07' (No such file or directory)
file: Cannot open `ile08' (No such file or directory)
file: Cannot open `ile09' (No such file or directory)
bandit4@bandit:~/inhere$ file -f*
file: Cannot open `ile00' (No such file or directory)
file: Cannot open `ile01' (No such file or directory)
file: Cannot open `ile02' (No such file or directory)
file: Cannot open `ile03' (No such file or directory)
file: Cannot open `ile04' (No such file or directory)
file: Cannot open `ile05' (No such file or directory)
file: Cannot open `ile06' (No such file or directory)
file: Cannot open `ile07' (No such file or directory)
file: Cannot open `ile08' (No such file or directory)
file: Cannot open `ile09' (No such file or directory)
bandit4@bandit:~/inhere$ file f*
f*: cannot open `f*' (No such file or directory)
```

Similar to the last level, all the filenames start with dashes, so we need to prefix them with ./ so the shell interprets them as filenames.

```
bandit4@bandit:~/inhere$ file ./*
./-file00: data
./-file01: data
./-file02: data
./-file03: data
./-file04: data
./-file05: data
./-file06: data
./-file07: ASCII text
./-file08: data
./-file09: data
```

From the output, we can see that ./-file07 is the only human-readable file identified as ASCII text. Then we use the cat command to read the file.

```
bandit4@bandit:~/inhere$ cat ./-file07
4oQYVPkxZOOEOO5pTW81FB8j8lxXGUQw
bandit4@bandit:~/inhere$
```

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:

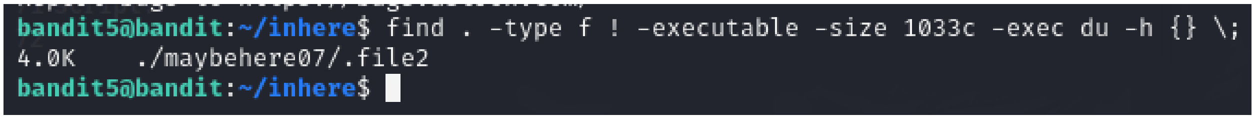
- human-readable
- 1033 bytes in size
- not executable

Solution

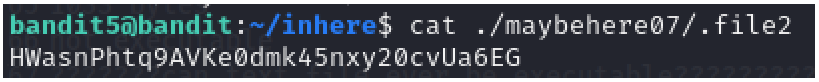
Find . (dot indicates the current directory)
find [path] [expression]

HWasnPhtq9AVKe0dmk45nxy20cvUa6EG

To find human-readable file, use du -h command
At the end of the find command, we need the -exec flag to execute the du -h command.



```
bandit5@bandit:~/inhere$ find . -type f ! -executable -size 1033c -exec du -h {} \;
4.0K    ./maybehere07/.file2
bandit5@bandit:~/inhere$ 
```



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

Bandit Level 6 → Level 7

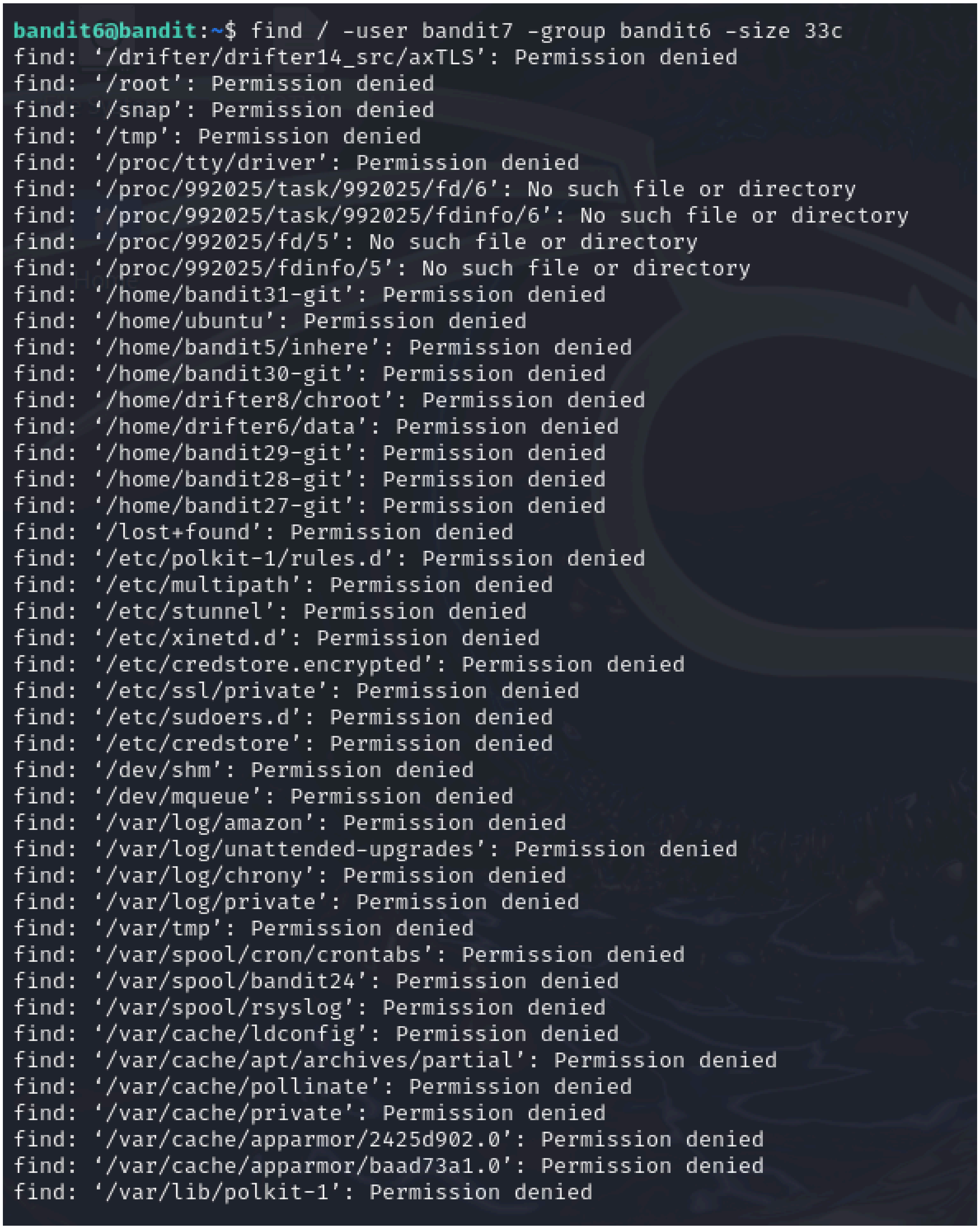
Level Goal

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

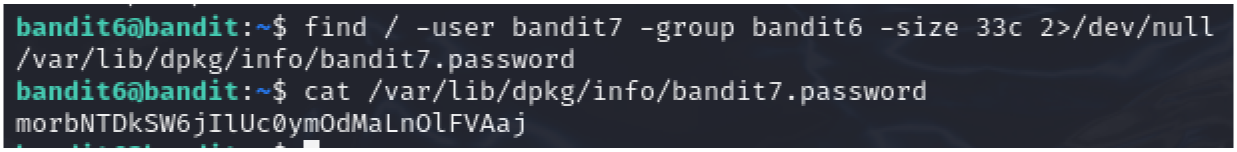
Solution
find command: recurses through all directories by default, including dot-hidden ones
find / : means applying the *find* command from the home directory.

If I just put find / -user bandit7 -group bandit6 -size 33c, it would give me lots of paths to permission denied directories. For example:



```
bandit6@bandit:~$ find / -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/992025/task/992025/fd/6': No such file or directory
find: '/proc/992025/task/992025/fdinfo/6': No such file or directory
find: '/proc/992025/fd/5': No such file or directory
find: '/proc/992025/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
find: '/etc/ssl/private': Permission denied
find: '/etc/sudoers.d': Permission denied
find: '/etc/credstore': Permission denied
find: '/dev/shm': Permission denied
find: '/dev/mqueue': Permission denied
find: '/var/log/amazon': Permission denied
find: '/var/log/unattended-upgrades': Permission denied
find: '/var/log/chrony': Permission denied
find: '/var/log/private': Permission denied
find: '/var/tmp': Permission denied
find: '/var/spool/cron/crontabs': Permission denied
find: '/var/spool/bandit24': Permission denied
find: '/var/spool/rsyslog': Permission denied
find: '/var/cache/ldconfig': Permission denied
find: '/var/cache/apt/archives/partial': Permission denied
find: '/var/cache/pollinate': Permission denied
find: '/var/cache/private': Permission denied
find: '/var/cache/apparmor/2425d902.0': Permission denied
find: '/var/cache/apparmor/baad73a1.0': Permission denied
find: '/var/lib/polkit-1': Permission denied
```

So there is a way to make things clearer, eliminate all the errors, we can use 2>/dev/null.
2>/dev/null: send all stderr error to a trash can Eli5
This works really well if we have tons of errors on the screen.



```
bandit6@bandit:~$ find / -user bandit7 -group bandit6 -size 33c 2>/dev/null
/var/lib/dpkg/info/bandit7.password
bandit6@bandit:~$ cat /var/lib/dpkg/info/bandit7.password
morbNTDkSW6jIlUc0ymOdMaLnOlFVAaj
```

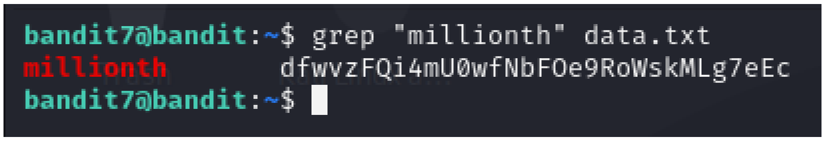
Then, we can identify the exact path that doesn't give errors. It's fortunate that there is only one place that isn't giving any errors. From here, we simply "print" out the content of the file via cat command.
morbNTDkSW6jIlUc0ymOdMaLnOlFVAaj

Bandit Level 7 → Level 8

Level Goal

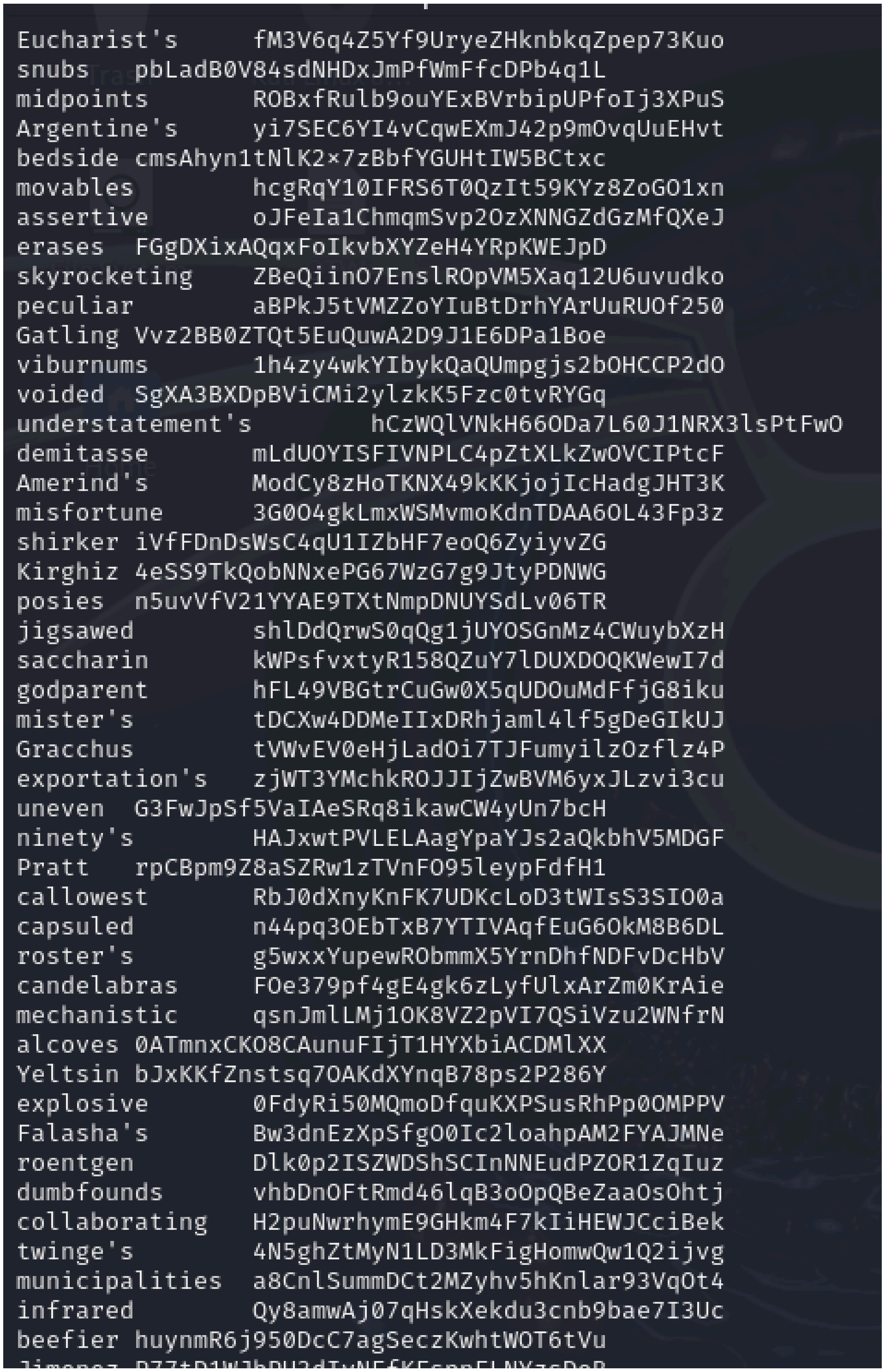
The password for the next level is stored in the file **data.txt** next to the word **millionth**

Solution
grep command prints lines that match patterns



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

So immediately, I thought of using grep to find where "millionth" is in the data.txt. Fortunately for me, it also prints out the next word to "millionth", which is the password. Well, from here, I thought that grep command also prints out the next string to the string I'm finding, which I didn't think grep could do before that. Then, out of curiosity, I "cat" out the file. Turns out it was all planned before. This is a snippet of the content:



```
Eucharist's       fM3V6q4Z5Yf9UryeZHknbkqZpep73Kuo
snubs   pbLadB0V84sdNHDxJmPfWmFfcDPb4q1L
midpoints         ROBxfRulb9ouYExBVrbipUPfoIj3XPuS
Argentine's       yi7SEC6YI4vCqwEXmJ42p9mOvqUuEHvt
bedside cmsAhyn1tNlK2×7zBbfYGUHtIW5BCtxc
movables          hcgRqY10IFRS6T0QzIt59KYz8ZoGO1xn
assertive         oJFeIa1ChmqmSvp2OzXNNGZdGzMfQXeJ
erases  FGgDXixAQqxFoIkvbXYZeH4YRpKWEJpD
skyrocketing      ZBeQiinO7EnslROpVM5Xaq12U6uvudko
peculiar          aBPkJ5tVMZZoYIuBtDrhYArUuRUOf250
Gatling Vvz2BB0ZTQt5EuQuwA2D9J1E6DPa1Boe
viburnums         1h4zy4wkYIbykQaQUmpgjs2bOHCCP2dO
voided  SgXA3BXDpBViCMi2ylzkK5Fzc0tvRYGq
understatement's        hCzWQlVNkH66ODa7L60J1NRX3lsPtFwO
demitasse         mLdUOYISFIVNPLC4pZtXLkZwOVCIPtcF
Amerind's         ModCy8zHoTKNX49kKKjojIcHadgJHT3K
misfortune        3G0O4gkLmxWSMvmoKdnTDAA6OL43Fp3z
shirker iVfFDnDsWsC4qU1IZbHF7eoQ6ZyiyvZG
Kirghiz 4eSS9TkQobNNxePG67WzG7g9JtyPDNWG
posies  n5uvVfV21YYAE9TXtNmpDNUYSdLv06TR
jigsawed          shlDdQrwS0qQg1jUYOSGnMz4CWuybXzH
saccharin         kWPsfvxtyR158QZuY7lDUXDOQKWewI7d
godparent         hFL49VBGtrCuGw0X5qUDOuMdFfjG8iku
mister's          tDCXw4DDMeIIxDRhjaml4lf5gDeGIkUJ
Gracchus          tVWvEV0eHjLadOi7TJFumyilzOzflz4P
exportation's     zjWT3YMchkROJJIjZwBVM6yxJLzvi3cu
uneven  G3FwJpSf5VaIAeSRq8ikawCW4yUn7bcH
ninety's          HAJxwtPVLELAagYpaYJs2aQkbhV5MDGF
Pratt   rpCBpm9Z8aSZRw1zTVnFO95leypFdfH1
callowest         RbJ0dXnyKnFK7UDKcLoD3tWIsS3SIO0a
capsuled          n44pq3OEbTxB7YTIVAqfEuG6OkM8B6DL
roster's          g5wxxYupewRObmmX5YrnDhfNDFvDcHbV
candelabras       FOe379pf4gE4gk6zLyfUlxArZm0KrAie
mechanistic       qsnJmlLMj1OK8VZ2pVI7QSiVzu2WNfrN
alcoves 0ATmnxCKO8CAunuFIjT1HYXbiACDMlXX
Yeltsin bJxKKfZnstsq7OAKdXYnqB78ps2P286Y
explosive         0FdyRi50MQmoDfquKXPSusRhPp0OMPPV
Falasha's         Bw3dnEzXpSfgO0Ic2loahpAM2FYAJMNe
roentgen          Dlk0p2ISZWDShSCInNNEudPZOR1ZqIuz
dumbfounds        vhbDnOFtRmd46lqB3oOpQBeZaaOsOhtj
collaborating     H2puNwrhymE9GHkm4F7kIiHEWJCciBek
twinge's          4N5ghZtMyN1LD3MkFigHomwQw1Q2ijvg
municipalities    a8CnlSummDCt2MZyhv5hKnlar93VqOt4
infrared          Qy8amwAj07qHskXekdu3cnb9bae7I3Uc
beefier huynmR6j950DcC7agSeczKwhtWOT6tVu
```

This looks like a dictionary where you have a keyword then an explanation next to it in the same line. So this fact concludes that using grep command is the best choice here.

dfwvzFQi4mU0wfNbFOe9RoWskMLg7eEc

Bandit Level 8 → Level 9

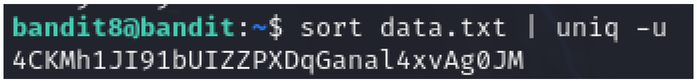
Level Goal

The password for the next level is stored in the file **data.txt** and is the only line of text that occurs only once.

Solution

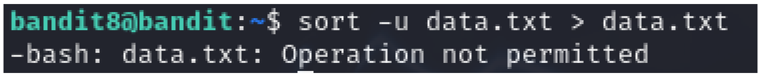
We want to use a command that can show the number of times of each line of text in the file.

So immediately, I thought of the uniq command with the -u –unique flag that only prints out unique lines. But 'uniq' does not detect repeated lines unless they are adjacent. We have to sort the input first.

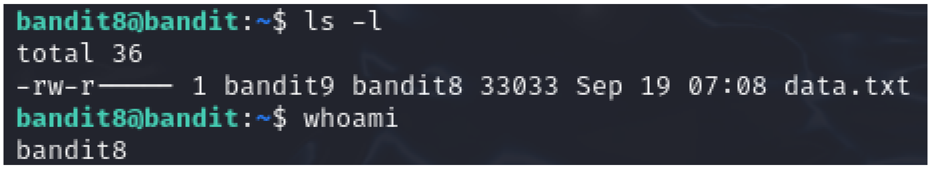


```
bandit8@bandit:~$ sort data.txt | uniq -u
4CKMh1JI91bUIZZPXDqGanal4xvAg0JM
```

4CKMh1JI91bUIZZPXDqGanal4xvAg0JM
Alternatively, we could also do sort -u filename > filename if we have the permission to modify the content of data.txt



```
bandit8@bandit:~$ sort -u data.txt > data.txt
-bash: data.txt: Operation not permitted
```



```
bandit8@bandit:~$ ls -l
total 36
-rw-r----- 1 bandit9 bandit8 33033 Sep 19 07:08 data.txt
bandit8@bandit:~$ whoami
bandit8
```

Here I as bandit8 only had group permission which is read only. This explains why the method above didn’t work.

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.

Solution

FGUW5ilLVJrxX9kMYMmlN4MgbpfMiqey

Only with the command “strings data.txt”, I got a list of results back, four of which were human-readable strings after a sequence of “=”. I just took a quick look through all four possibilities, and this one seemed most potential:
D9========== FGUW5ilLVJrxX9kMYMmlN4MgbpfMiqey
I’m pretty sure this is supposed to be trickier or require the player to use a more complicated command. But that just worked.

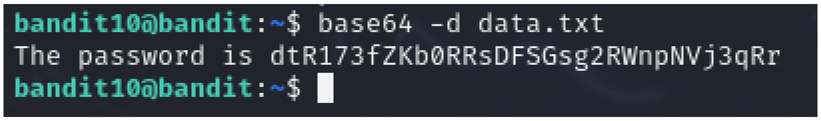
Bandit Level 10 → Level 11

Level Goal

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

Solution

Use the -d flag of base64 command to decode data in the file data.txt:



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

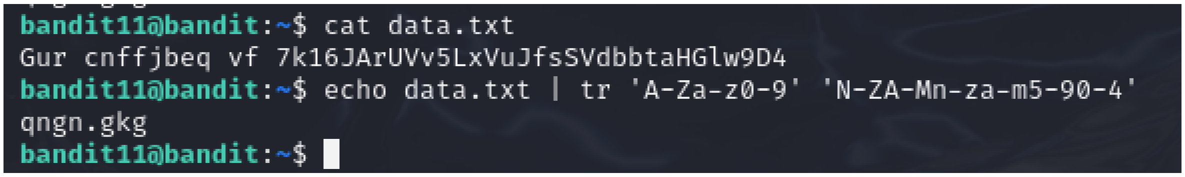
dtR173fZKb0RRsDFSGsg2RWnpNVj3qRr

Bandit Level 11 → Level 12 – ROT13

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

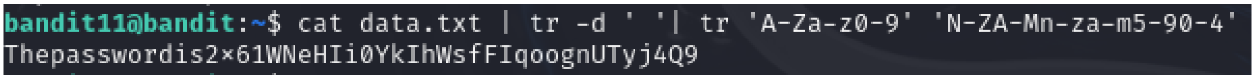
Solution



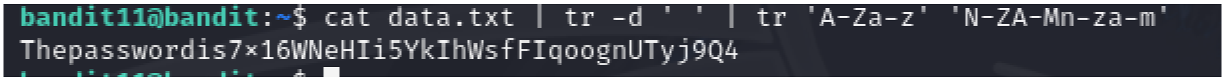
```
bandit11@bandit:~$ cat data.txt
Gur cnffjbeq vf 7k16JArUVv5LxVuJfsSVdbbtaHGlw9D4
bandit11@bandit:~$ echo data.txt | tr 'A-Za-z0-9' 'N-ZA-Mn-za-m5-90-4'
qngn.gkg
bandit11@bandit:~$ 
```

The file content includes not only letters in lower and upper cases, it also includes numbers.

2x61WNeHIi0YkIhWsfFIqoognUTyj4Q9 This is not the correct answer.



```
bandit11@bandit:~$ cat data.txt | tr -d ' '| tr 'A-Za-z0-9' 'N-ZA-Mn-za-m5-90-4'
Thepasswordis2×61WNeHIi0YkIhWsfFIqoognUTyj4Q9
```



```
bandit11@bandit:~$ cat data.txt | tr -d ' ' | tr 'A-Za-z' 'N-ZA-Mn-za-m'
Thepasswordis7×16WNeHIi5YkIhWsfFIqoognUTyj9Q4
```

Quick fix: the numbers were not rotated, only the alphabets
7x16WNeHIi5YkIhWsfFIqoognUTyj9Q4

Bandit Level 12 → Level 13

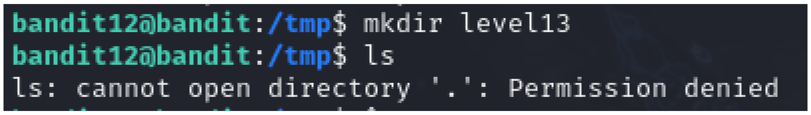
Level Goal

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!)

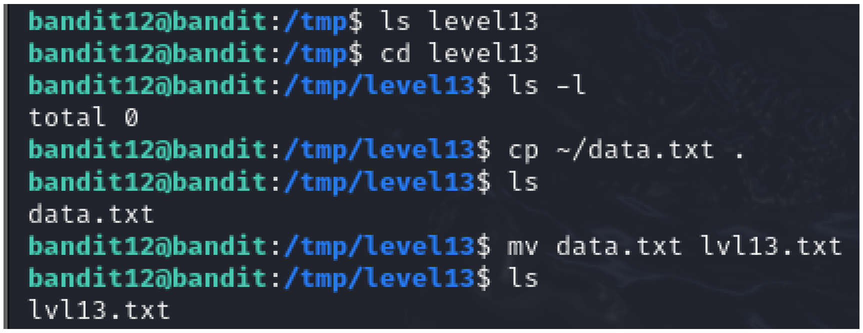
Solution

What’s the difference btw /tmp and /var/tmp?

Interesting fact: you can’t open /tmp but you can create a folder in it and open it !!!Source



```
bandit12@bandit:/tmp$ mkdir level13
bandit12@bandit:/tmp$ ls
ls: cannot open directory '.': Permission denied
```



```
bandit12@bandit:/tmp$ ls level13
bandit12@bandit:/tmp$ cd level13
bandit12@bandit:/tmp/level13$ ls -l
total 0
bandit12@bandit:/tmp/level13$ cp ~/data.txt .
bandit12@bandit:/tmp/level13$ ls
data.txt
bandit12@bandit:/tmp/level13$ mv data.txt lvl13.txt
bandit12@bandit:/tmp/level13$ ls
lvl13.txt
```