Over The Wire: Bandit

Write Up

eXit

January 28, 2017

Contents

Chapter 1

Executive Summary

Being able to traverse through each level of Bandit, on the website OverTheWire. The goals of this project are to:

- Enhance skills using the bash terminal
- Further develop understanding of linux

Chapter 2

Bandit Level: 0

2.1 Introduction

Using puTTy to establish a ssh tunnel into the OverTheWire server.

2.2 Login Details

Connecting to: bandit.labs.overthewire.org

Username: bandit0

Password: bandit0

2.3 Method

Using the ls (list segment) command to identify what files are within the current directory. Identifying that the file is located under the name of readme. Using the cat (concatenate) command to output the contents of the file, the password was located for the next level within.

```
bandit0@melinda:~$ ls
readme
bandit0@melinda:~$ cat readme
boJ9jbbUNNfktd78OOpsqOltutMc3MY1
bandit0@melinda:~$
```

2.4 Output

Password: boJ9jbbUNNfktd78OOpsqOltutMc3MY1

Chapter 3

Bandit Level: 1

3.1 Scenario

To aquire the password from the ”-” file.

3.2 Login

Username: bandit1

Password: boJ9jbbUNNfktd78OOpsqOltutMc3MY1

3.3 Method

Using the ls (list segment) command to identify what files are within the current directory. Identifying that the file is located under the name of readme. Using the cat (concatenate) command to output the contents of the file, but using the pathing in addition to re the ”-” file. As ”-” is seens as a non special character which causes a problem with the kernel.

```
bandit1@melinda:~$ ls
-
bandit1@melinda:~$ cat ./-
CV1DtqXWVFXTvM2F0k09SHz0YwRINYA9
bandit1@melinda:~$
```

3.4 Output

Password: CV1DtqXWVFXTvM2F0k09SHz0YwRINYA9

Chapter 4

Bandit Level: 2

4.1 Scenario

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

4.2 Login Details

Username: bandit2
Password: CV1DtqXWVFXTvM2F0k09SHz0YwRINYA9

4.3 Method

Using the ls (list segment) command to identify what files are within the current directory. Identifying that the file is located under the name of readme. Using the cat (concatenate) command to output the contents of the file, but using the tab function to input \(space) after each string of text. Making the file name "spaces\in\this\filename\".

```
bandit2@melinda:~$ ls
spaces in this filename
bandit2@melinda:~$ cat spaces\ in\ this\ filename
UmHadQclWmgdLOKQ3YNgjWxGoRMb5luK
bandit2@melinda:~$
```

4.4 Output

Password: UmHadQclWmgdLOKQ3YNgjWxGoRMb5luK

Chapter 5

Bandit Level: 3

5.1 Scenario

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

5.2 Login Details

Username: bandit3

Password: UmHadQclWmgdLOKQ3YNgjWxGoRMb5luK

5.3 Method

Using the ls (list segment) command to identify what files are within the current directory. Identifying that there is another directory called "inhere", we cd (change directory) into it. Using ls doesn't show anything within the folder, however if we use the command ls -la (list, -l(long listed version), a(all files including ones with a dot prefix). we see that there is a .hidden file.

Using the cat command we can now do the following cat ./.hidden

```
bandit3@melinda:~/inhere$ ls -la
total 12
drwxr-xr-x 2 root    root    4096 Nov 14  2014 .
drwxr-xr-x 3 root    root    4096 Nov 14  2014 ..
-rw-r----- 1 bandit4 bandit3   33 Nov 14  2014 .hidden
bandit3@melinda:~/inhere$ ls -a
.  ..  .hidden
bandit3@melinda:~/inhere$ cat ./.hidden
pIwrPrtPN36QITSp3EQaw936yaFoFgAB
bandit3@melinda:~/inhere$
```

5.4 Output

Password: pIwrPrtPN36QITSp3EQaw936yaFoFgAB

Chapter 6

Bandit Level: 4

6.1 Scenario

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.

6.2 Login Details

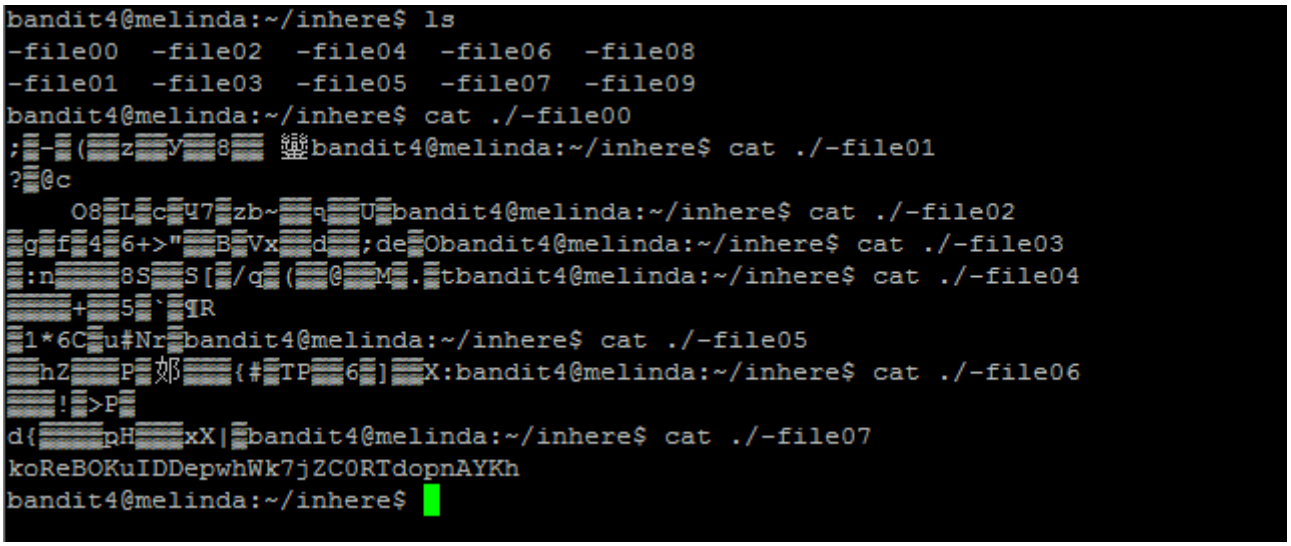
Username: bandit4
Password: pIwrPrtPN36QITSp3EQaw936yaFoFgAB

6.3 Method

Using the ls (list segment) command to identify what files are within the current directory. Identifying that there is another directory called "inhere", we cd (change directory) into it. Using ls we can see that there are multiple files within. Using the command ls -la we can see that they are read write, and that everyone can read it.

Using the file command we can then determine what the file is, which in this case is data. Using cat and the file path and name of the file we get unreadable information.

From the information we are given from the scenario, the password is located within a human-readable format. If we cat all the files we find that "-file07" provides the password.



6.4 Output

Password: koReBOKuIDDepwhWk7jZC0RTdopnAYKh

Chapter 7

Bandit Level: 5

7.1 Scenario

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

7.2 Login Details

Username: bandit5
Password: koReBOKuIDDepwhWk7jZC0RTdopnAYKh

7.3 Method

Using the ls (list segment) command to identify what files are within the current directory. Identifying that there is another directory called "inhere", we cd (change directory) into it. Using ls we can see that there are multiple files within. Using the command ls -la we can see that they are read write, and that everyone can read it.

From the scenario we can see that is is in human readable format, within a file size of 1033 bytes and non executable. We can use the find command with the additional commands to search based on the criteria of the scenario.

The command we can use is find -type f (for regular files) and -size 1033c (search based on the size which will be 1033c (c for bytes)).

which will output the ./maybehere07/.file2. Traversing to maybehere07 directory and using the file path while using the cat command the password is shown.

```
bandit5@melinda:~$ cd inhere/
bandit5@melinda:~/inhere$ ls
maybehere00  maybehere04  maybehere08  maybehere12  maybehere16
maybehere01  maybehere05  maybehere09  maybehere13  maybehere17
maybehere02  maybehere06  maybehere10  maybehere14  maybehere18
maybehere03  maybehere07  maybehere11  maybehere15  maybehere19
bandit5@melinda:~/inhere$ find -type f -size 1033c
./maybehere07/.file2
bandit5@melinda:~/inhere$ cd ./maybehere07
bandit5@melinda:~/inhere/maybehere07$ ls
-file1  -file2  -file3  spaces file1  spaces file2  spaces file3
bandit5@melinda:~/inhere/maybehere07$ cat file2
cat: file2: No such file or directory
bandit5@melinda:~/inhere/maybehere07$ cat ./
-file1        -file3        .file2        spaces file1  spaces file3
-file2        .file1        .file3        spaces file2
bandit5@melinda:~/inhere/maybehere07$ cat ./
-file1        -file3        .file2        spaces file1  spaces file3
-file2        .file1        .file3        spaces file2
bandit5@melinda:~/inhere/maybehere07$ cat ./.
./      ../     .file1  .file2  .file3
bandit5@melinda:~/inhere/maybehere07$ cat ./.file2
DXjZPULLxYr17uwoI01bNLQbtFemEgo7
```

7.4 Output

Password: DXjZPULLxYr17uwoI01bNLQbtFemEgo7

Chapter 8

Bandit Level: 6

8.1 Scenario

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

8.2 Login Details

Username: bandit6
Password: DXjZPULLxYr17uwoI01bNLQbtFemEgo7

8.3 Method

Logging in and using the ls command, no files were presented. In the scenario the password is ”somewhere” on the server. changing the directory to the home folder, and making use of the find command we can search the server for the file based on the criteria given.

We can construct the command: find / -readable -user bandit7 -group bandit6 -size 33c

```
bandit6@melinda:/home$ find / -readable -user bandit7 -group bandit6 -size 33c
```

Using the command we can then traverse through the finds to see which file we have permissions to read.ls

```
find: `/run/user/11023': Permission denied
find: `/run/user/6004': Permission denied
find: `/run/user/5013': Permission denied
find: `/run/user/17000': Permission denied
find: `/run/user/11018': Permission denied
find: `/run/user/11017': Permission denied
find: `/run/user/5012': Permission denied
find: `/run/user/14002': Permission denied
find: `/run/user/11020': Permission denied
find: `/run/user/15006': Permission denied
find: `/run/user/14008': Permission denied
find: `/run/user/14007': Permission denied
find: `/run/user/0': Permission denied
find: `/run/shm': Permission denied
find: `/tmp': Permission denied
find: `/lost+found': Permission denied
find: `/var/lib/sudo': Permission denied
find: `/var/lib/php5': Permission denied
find: `/var/lib/cron-apt/_-_etc_-_cron-apt_-_config': Permission denied
find: `/var/lib/mysql': Permission denied
/var/lib/dpkg/info/bandit7.password
find: `/var/cache/ldconfig': Permission denied
find: `/var/log': Permission denied
find: `/var/lock': Permission denied
```

then using the cat command on the file /var/lib/dpkg/info/bandit7.password we get the password

```
bandit6@melinda:/home$ cat /var/lib/dpkg/info/
Display all 2610 possibilities? (y or n)
bandit6@melinda:/home$ cat /var/lib/dpkg/info/bandit7.password
HKBPTKQnIay4Fw76bEy8PVxKEDQRKTzs
bandit6@melinda:/home$
```

8.4 Output

Password: HKBPTKQnIay4Fw76bEy8PVxKEDQRKTzs

Chapter 9

Bandit Level: 7

9.1 Scenario

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

9.2 Login Details

Username: bandit7
Password: HKBPTKQnIay4Fw76bEy8PVxKEDQRKTzs

9.3 Method

Connecting to the level 7, the scenario states that the password is located in the data.txt file which is located within the home directory. Using cat we see that the file has many different passwords, making it tedious to look through for the password which is next to the word "millionth". However using the cat data.txt and grep command we can search for the line that the keyword is on.

Command: cat data.txt — grep millionth. We need to use a pipe command to be able to run both commands together.

```
mustering       O8KTPj7je3wCvevszGtZ0nzxLoiDKycm
vindicator's    t7lJ6MvYeSWO9W9zhMmCSRzqz1LpYuVI
insurances      qHqaKl3cMzvX1mRCTur3dSImP9YsKyZm
tramming        rmxHqsfcEX3CH362cWbKfjgWaixY2XiG
isolationist    HwHu7J8Ce3lwML77nfQlRMiiew09RTTo
Cheever P83n41SRorZMnERKTMQ8dmqt5oqffZxk
hays    sBOOqNmfCXnPsnXqwhjVsR5ktFRyTlkW
crested qQGaFFdSW1RZeUsOboHRR4vSYc6McMIC
popping rttdKjsrnryfeg4S1lRdWMN4TtL4kVsd
Nouakchott's    F3TB8wUGuiYBX8AyzWpOh8lpX2j2cfvZ
vexing  jChei242gqqUZjHqL6857jsAJDZVWcdj
straightness    PgzpXFsJbJy68doDy0jVnij2Ob9gPFms
tipsiest        6tvS0ruh7U2W0I9r8FIK1RRetS9LADi0
mammal's        ba29TY0wT8Rn8WiQL9jjbqRyUCfeWrm9
altitude's      PejbFQMZr1STSccUajDEu96EYbjd240x
due     3keAzmeZ6a9TZjHEQP4KUX0ygdapqeaE
colonizers      7iNnYZFBjNZLJFIKVGK3h101flUOU8vy
Tamworth        DbvvNC0lcTKZitF8xGn5fHf3ff9E2vnY
lept    TL4Z79TIeJWiTJ032wXq5hVatT9t6SZF
trustier        TbuPhs0kw1mdIrtOOmPItkbAbaox6Rk9
laments WCdfGAgxaFqfN47NDQUJpUg17wCOZ8qs
bandit7@melinda:~$ cat data.txt | grep millionth
millionth       cvX2JJa4CFALtqS87jk27qwqGhBM9plV
bandit7@melinda:~$
```

9.4 Output

Password: cvX2JJa4CFALtqS87jk27qwqGhBM9plV

Chapter 10

Bandit Level: 8

10.1 Scenario

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

10.2 Login Details

Username: bandit8
Password: cvX2JJa4CFALtqS87jk27qwqGhBM9plV

10.3 Method

Level 8 requires us to list the only unique value within the file. Using the list command we see that we are dealing with only one file, but using the cat command alone will take time to find the password. Using the sort function based on the unique id within the file. In the scenario we see that the password is the only unique in the file. Using the pipe command we can find the unique value, sort the text file and output the results to the command prompt.

Command: cat data.txt — sort — uniq -u

```
bandit8@melinda:~$ cat data.txt | sort | uniq -u
UsvVyFSfZZWbi6wgC7dAFyFuR6jQQUhR
bandit8@melinda:~$
```

10.4 Output

Password: UsvVyFSfZZWbi6wgC7dAFyFuR6jQQUhR

Chapter 11

Bandit Level: 9

11.1 Scenario

The password for the next level is stored in the file data.txt in one of the few human-readable strings, beginning with several = characters.

11.2 Login Details

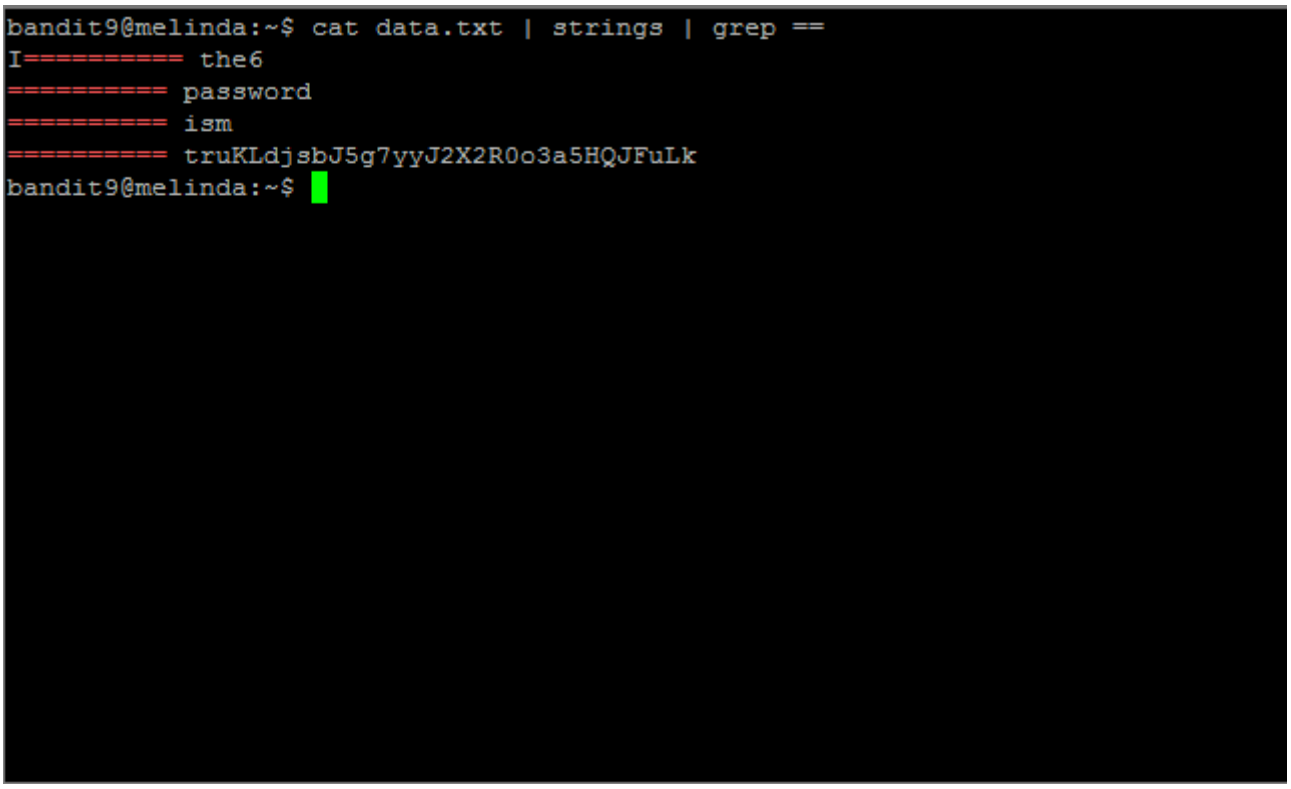
Username: bandit9
Password: UsvVyFSfZZWbi6wgC7dAFyFuR6jQQUhR

11.3 Method

List the files that are located in the home directory, which shows the data.txt file. From the scenario we know that the file contains binary data making it unreadable if we just cat the file. We need to concatinate the file with other commands.

Cat to output the file, strings to find any human readable string and grep to locate the sentances that contain the '='. In our command however we are going to use two or more '=' as the scenario specifies that it has more than one '='.

Command: cat data.txt — strings — grep ==



11.4 Output

Password: truKLdjsbJ5g7yyJ2X2R0o3a5HQJFuLk

Chapter 12

Bandit Level: 10

12.1 Scenario

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

12.2 Login Details

Username: bandit10
Password: truKLdjsbJ5g7yyJ2X2R0o3a5HQJFuLk

12.3 Method

Logging into the level, we need to establish what files we are working with. Like previously done the first point of call is listing everything in the directory. After this we can do cat data.txt. The file contains a long string of text which we can assume is the base64 encode string.

```
bandit10@melinda:~$ ls
data.txt
bandit10@melinda:~$ cat data.txt
VGhlIHBhc3N3b3JkIGlzIElGdWt3S0dzRlc4TU9xM0lSRnFyeEUxaHhUTkViVVBSCg==
bandit10@melinda:~$
```

We can use the base64 function along with –decode to unencrypt the string into the password.

Command: cat data.txt — base64 –decode

```
bandit10@melinda:~$ ls
data.txt
bandit10@melinda:~$ cat data.txt
VGhlIHBhc3N3b3JkIGlzIElGdWt3S0dzRlc4TU9xM0lSRnFyeEUxaHhUTkViVVBSCg==
bandit10@melinda:~$ cat data.txt | base64 --decode
The password is IFukwKGsFW8MOq3IRFqrxE1hxTNEbUPR
bandit10@melinda:~$
```

12.4 Output

Password: IFukwKGsFW8MOq3IRFqrxE1hxTNEbUPR

Chapter 13

Bandit Level: 11

13.1 Scenario

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

13.2 Login Details

Username: bandit11
Password: IFukwKGsFW8MOq3IRFqrxE1hxTNEbUPR

13.3 Method

Level 11 focuses on decrypting the password from a rotation of 13 encryption method. This is where by the letters have been shifted by 13. For example 'A' becomes 'N'. This system is a type of caeser cipher.

To decrypt this we can use the tr command

Command:cat data.txt — tr 'n-za-mN-ZA-M' 'a-zA-Z'

```
bandit11@melinda:~$ ls
data.txt
bandit11@melinda:~$ cat data.txt
Gur cnffjbeq vf 5Gr8L4qetPEsPk8htqjhRK8XSP6x2RHh
bandit11@melinda:~$ cat data.txt | tr 'n-za-mN-ZA-M' 'a-zA-Z'
The password is 5Te8Y4drgCRfCx8ugdwuEX8KFC6k2EUu
bandit11@melinda:~$
```

13.4 Output

Password: 5Te8Y4drgCRfCx8ugdwuEX8KFC6k2EUu

Chapter 14

Bandit Level: 12

14.1 Scenario

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!)

14.2 Login Details

Username: bandit12
Password: 5Te8Y4drgCRfCx8ugdwuEX8KFC6k2EUu

14.3 Method

This level focuses on decompressing many different versions of the file, until the ASCII file has become readable.

To start we need to make a new directory within the tmp file, secondly we need to then create a hexdump of the data.txt file, with the xxd command.

```
--[ Tools ]--

 For your convenience we have installed a few usefull tools which you can find
 in the following locations:

    * peda (https://github.com/longld/peda.git) in /usr/local/peda/
    * gdbinit (https://github.com/gdbinit/Gdbinit) in /usr/local/gdbinit/
    * pwntools (https://github.com/Gallopsled/pwntools) in /usr/src/pwntools/
    * radare2 (http://www.radare.org/) should be in $PATH

--[ More information ]--

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

  For questions or comments, contact us through IRC on
  irc.overthewire.org.


bandit12@melinda:~$ ls
data.txt
bandit12@melinda:~$ mkdir /tmp/hellog0d
bandit12@melinda:~$
```

Using the file command on the hexdumped file we can then see it is a gzipped file and was called data2.bin, so what we have done so far is correct.

```
bandit12@melinda:/tmp/hellog0d$ xxd -r ~/data.txt data.txt
bandit12@melinda:/tmp/hellog0d$
```

We know that it is a gzip file, so we can decompress the file using the zcat command, which decompressed and outputs in the same manner as the cat command.

```
bandit12@melinda:/tmp/hellog0d$ zcat data.txt > data2.bin
bandit12@melinda:/tmp/hellog0d$
```

Using the file command agian we can determine what compression type it is, in this case its a bzip2 file format, which zcat doesn't deal with. However the bzip2 command with "-d" to decompress it.

```
bandit12@melinda:/tmp/hellog0d$ file data2.bin
data2.bin: bzip2 compressed data, block size = 900k
bandit12@melinda:/tmp/hellog0d$ bzip2 -d data2.bin
bzip2: Can't guess original name for data2.bin -- using data2.bin.out
bandit12@melinda:/tmp/hellog0d$
```

Using the file command again we can see it is a gzip file, repeating what we did with the zcat command we can get data4.bin recovered.

```
bandit12@melinda:/tmp/hellog0d$ file data2.bin.out
data2.bin.out: gzip compressed data, was "data4.bin", from Unix, last modified:
Fri Nov 14 10:32:20 2014, max compression
bandit12@melinda:/tmp/hellog0d$ zcat data2.bin.out > data4.bin
bandit12@melinda:/tmp/hellog0d$ ls
data.txt  data2.bin.out  data4.bin
bandit12@melinda:/tmp/hellog0d$
```

The next file format is a POSIX tar archive, meaning we can use the tar command with the following additional commands "-xvf" to decompress it.

```
bandit12@melinda:/tmp/hellog0d$ file data4.bin
data4.bin: POSIX tar archive (GNU)
bandit12@melinda:/tmp/hellog0d$ tar -xvf data4.bin data5.bin
data5.bin
bandit12@melinda:/tmp/hellog0d$ ls
data.txt  data2.bin.out  data4.bin  data5.bin
bandit12@melinda:/tmp/hellog0d$
```

The data5.bin file is a POSIX file, so repeating the same steps again we get data6.bin

```
bandit12@melinda:/tmp/hellog0d$ file data5.bin
data5.bin: POSIX tar archive (GNU)
bandit12@melinda:/tmp/hellog0d$ tar -xvf data5.bin data6.bin
data6.bin
bandit12@melinda:/tmp/hellog0d$ file data6.bin
data6.bin: bzip2 compressed data, block size = 900k
bandit12@melinda:/tmp/hellog0d$
```

Using file again, we can see that data6.bin is a bzip2 file format, doing the same command that we used earlier for the bzip2 format we can get data6.bin.out (data7.bin) extracted. Which is a POSIX tar archive file.

```
bandit12@melinda:/tmp/hellog0d$ file data6.bin.out
data6.bin.out: POSIX tar archive (GNU)
bandit12@melinda:/tmp/hellog0d$ tar -xvf data6.bin.out
data8.bin
bandit12@melinda:/tmp/hellog0d$
```

Data8.bin is a gzip file so we can use the zcat command to extract data9/bin, finally the data9.bin file is an ASCII text file meaning we can cat the file and the password is within.

```
bandit12@melinda:/tmp/hellog0d$ file data8.bin
data8.bin: gzip compressed data, was "data9.bin", from Unix, last modified: Fri
Nov 14 10:32:20 2014, max compression
bandit12@melinda:/tmp/hellog0d$ zcat data8.bin > data9.bin
bandit12@melinda:/tmp/hellog0d$ file data9.bin
data9.bin: ASCII text
bandit12@melinda:/tmp/hellog0d$ cat data9.bin
The password is 8ZjyCRiBWFYkneahHwxCv3wb2a1ORpYL
bandit12@melinda:/tmp/hellog0d$
```

14.4 Output

Password: 8ZjyCRiBWFYkneahHwxCv3wb2a1ORpYL

Chapter 15

Bandit Level: 13

15.1 Scenario

The password for the next level is stored in \etc \bandit_pass \bandit14 file and can only be read by user bandit14. For this level, you dont 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.

15.2 Login Details

Username: bandit13
Password: 8ZjyCRiBWFYkneahHwxCv3wb2a1ORpYL

15.3 Method

This level is a harder one, as it doesn't focus on anything to do with decompression or any previous levels. So lets hit the curve with our "reconnaissance".

So logging in we need to see what we are provided with to give us a clue on where to go next. Using the ls command (list) we have an ssh private key file. Which indicates we can connect to the IP address via ssh.

Using the man page, we need to find what tack command is for using a private key file, which is ”-i”.

Constructing the ssh command, the aim is to connect as user bandit14 and as we are logged in we are connecting back to our server which is localhost. Type yes to continue with the connection, and we are logged in as bandit14.

```
bandit13@melinda:~$ ssh -i sshkey.private bandit14@localhost
Could not create directory '/home/bandit13/.ssh'.
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is 05:3a:1c:25:35:0a:ed:2f:cd:87:1c:f6:fe:69:e4:f6.
Are you sure you want to continue connecting (yes/no)? yes
```

Now that we are logged in as bandit14, we can retrieve the password from the \etc \bandit_pass\bandit14 file.

```
bandit14@melinda:~$ cat /etc/bandit_pass/bandit14
4wcYUJFw0k0XLShlDzztnTBHiqxU3b3e
bandit14@melinda:~$
```

15.4 Output

Password: 4wcYUJFw0k0XLShlDzztnTBHiqxU3b3e

Chapter 16

Bandit Level: 14

16.1 Scenario

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

16.2 Login Details

Username: bandit14
Password: 4wcYUJFw0k0XLShlDzztnTBHiqxU3b3e

16.3 Method

Building on the knowledge we have developed with the previous challenge we can use our knowledge and apply it to this challenge.

So from the challenge scenario we need to connect to port 30000 and submit our password.

Similar to the last challenge we need to connect to the localhost again, but this time with telnet.

There are two methods of doing this challenge

16.4 Method 1

Using the telnet command and connecting to localhost and port 30000, we can then input our password to obtain level 15's password.

```
bandit14@melinda:~$ telnet localhost 30000
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
4wcYUJFw0k0XLShlDzztnTBHiqxU3b3e
Correct!
BfMYroe26WYalil77FoDi9qh59eK5xNr

Connection closed by foreign host.
bandit14@melinda:~$
```

16.5 Method 2

Instead of using the telnet command we can use the netcat command (nc) and connect to the locahost and port 30000 and entering the password in this way.

```
bandit14@melinda:~$ nc localhost 30000
4wcYUJFw0k0XLShlDzztnTBHiqxU3b3e
Correct!
BfMYroe26WYalil77FoDi9qh59eK5xNr

bandit14@melinda:~$
```

16.6 Output

Password: BfMYroe26WYalil77FoDi9qh59eK5xNr

Chapter 17

Bandit Level: 15

17.1 Scenario

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.

17.2 Login Details

Username: bandit15

Password: BfMYroe26WYalil77FoDi9qh59eK5xNr

17.3 Method

The challenge is similar to the previous one but requires you to use SSL. Using openssl and its connect-able client s_client allows the user to connect to a server similar to telnet and netcat.

Using the provided link for the OpenSSL cookbook we can see how the command is constructed. (When connecting, the heartbleed bug appears and denies us access due to an error. Adding "-ign_eof" to the end of the command for it to work)

```
bandit15@melinda:~$ openssl s_client -connect localhost:30001 -ign_eof
CONNECTED(00000003)
```

Once the command has gone through, we can then input the command used for level 15 to get the password for level 16.

```
BfMYroe26WYalil77FoDi9qh59eK5xNr
Correct!
cluFn7wTiGryunymYOu4RcffSxQluehd

read:errno=0
```

17.4 Output

Password: cluFn7wTiGryunymYOu4RcffSxQluehd

Chapter 18

Bandit Level: 16

18.1 Scenario

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 dont. There is only 1 server that will give the next credentials, the others will simply send back to you whatever you send to it.

18.2 Login Details

Username: bandit16
Password: cluFn7wTiGryunymYOu4RcffSxQluehd

18.3 Method

From the scenario we can see that its between 31000 and 32000 and requires SSL. This requires scanning to figure out which one is accepting SSL.

To scan the ports we can use NMAP, on the range of 31000 to 32000. We also want to check to see if any servers are running on the range of ports given or if it will just

echo back. We can manual test the ip range with the use of piping echo with the connection to see if the text is returned or if we get an error. However I have done it with a service version detection via "-sV" of nmap.

```
bandit16@melinda:~$ nmap -A localhost -p 31000-32000 -sV

Starting Nmap 6.40 ( http://nmap.org ) at 2016-05-28 16:35 UTC
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00062s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE VERSION
31046/tcp open  echo
31518/tcp open  msdtc   Microsoft Distributed Transaction Coordinator (error)
31691/tcp open  echo
31790/tcp open  msdtc   Microsoft Distributed Transaction Coordinator (error)
31960/tcp open  echo
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at http://nmap.
org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 41.37 seconds
bandit16@melinda:~$
```

Using the previous ssh command we did in the last challenge to connect to the servers, we can connect to both 31518 and 31970 as we can see that the rest are echoing back what we type.

```
---
New, TLSv1/SSLv3, Cipher is DHE-RSA-AES256-SHA
Server public key is 2048 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
SSL-Session:
    Protocol  : SSLv3
    Cipher    : DHE-RSA-AES256-SHA
    Session-ID: 124F9769DE97C9223D589FE6D255CE40A44F8FE31E62F409889F9A30998755AD
    Session-ID-ctx:
    Master-Key: AC1FDF530EB0881C6C6630884BEC264E5A420C04006FBB605FEE4D2490929D68
CCAD804151DC051B16A270469E51D1C2
    Key-Arg   : None
    PSK identity: None
    PSK identity hint: None
    SRP username: None
    Start Time: 1464453525
    Timeout   : 300 (sec)
    Verify return code: 18 (self signed certificate)
---
cluFn7wTiGryunymYOu4RcffSxQluehd
cluFn7wTiGryunymYOu4RcffSxQluehd
```

After connecting to port 31518 we see that it is just echoing back what we have typed, leaving 31970 as the last remaining possible answer. Repeating the command as before but with the 31970 port at the end, we get a private key which we need to save. As this is the password for the next level.

```
Expansion: NONE
SSL-Session:
    Protocol  : SSLv3
    Cipher    : DHE-RSA-AES256-SHA
    Session-ID: 99A3999A554170B21B7782CEC1DA1029B6F032E4B9110BD96719AC36411BEB9B
    Session-ID-ctx:
    Master-Key: 67C5CCCEBCCD01B3B21E64D710589AB3DC8E75B736FBFA594FDA763D70CEE17B
30418D051B9ACE553B5E7A0EE8F8A8DD
    Key-Arg   : None
    PSK identity: None
    PSK identity hint: None
    SRP username: None
    Start Time: 1464453616
    Timeout   : 300 (sec)
    Verify return code: 18 (self signed certificate)
---
cluFn7wTiGryunymYOu4RcffSxQluehd
Correct!
-----BEGIN RSA PRIVATE KEY-----
MIIEogIBAAKCAQEAvmOkuifmMg6HL2YPIOjon6iWfbp7c3jx34YkYWqUH57SUdyJ
imZzeyGC0gtZPGujUSxiJSWI/oTqexh+cAMTSMlOJf7+BrJObArnxd9Y7YT2bRPQ
Ja6Lzb558YW3FZl87ORiO+rW4LCDCNd2lUvLE/GL2GWyuKN0K5iCd5TbtJzEkQTu
DSt2mcNn4rhAL+JFr56o4T6z8WWAW18BR6yGrMq7Q/kALHYW3OekePQAzL0VUYbW
JGTi65CxbCnzc/w4+mqQyvmzpWtMAzJTzAzQxNbkR2MBGySxDLrjg0LWN6sK7wNX
```

Now that we have a private key we would like to store it, as we can't store in the home directory we need to store it in the tmp file. Paste the private key into your desired text editor, in this case its nano.

Now that we have our private key within a file we can connect to the next level using the ssh command and private key. (If you connect with it after you make the file, it will give you a warning saying UNPROTECTED PRIVATE KEY FILE). To get around this we need to change the permissions of the file via the chmod command to give the owner (thats you) the ability to read and write access. Once done connect and we are in bandit level 17.

```
bandit16@melinda:/tmp/keyforlevel1711$ chmod 600 sshkey.private
bandit16@melinda:/tmp/keyforlevel1711$ ssh -i ./sshkey.private bandit17@localhos
t
Could not create directory '/home/bandit16/.ssh'.
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is 05:3a:1c:25:35:0a:ed:2f:cd:87:1c:f6:fe:69:e4:f6.
Are you sure you want to continue connecting (yes/no)? yes
Failed to add the host to the list of known hosts (/home/bandit16/.ssh/known_hos
ts).

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

Please note that wargame usernames are no longer level<X>, but wargamename<X>
e.g. vortex4, semtex2, ...
```

18.4 Output

Password: —BEGIN RSA PRIVATE KEY— MIIEogIBAAKCAQEAvmOkuifmMg6HL2YPIOjo
imZzeyGC0gtZPGujUSxiJSWI/oTqexh+cAMTSMlOJf7+BrJObArnxd9Y7YT2bRPQ
Ja6Lzb558YW3FZl87ORiO+rW4LCDCNd2lUvLE/GL2GWyuKN0K5iCd5TbtJzEkQTu
DSt2mcNn4rhAL+JFr56o4T6z8WWAW18BR6yGrMq7Q/kALHYW3OekePQAzL0VUYbW
JGTi65CxbCnzc/w4+mqQyvmzpWtMAzJTzAzQxNbkR2MBGySxDLrjg0LWN6sK7wNX
x0YVztz/zbIkPjfkU1jHS+9EbVNj+D1XFOJuaQIDAQABAoIBABagpxpM1aoLWfvD
KHcj10nqcoBc4oE11aFYQwik7xfW+24pRNuDE6SFthOar69jp5RlLwD1NhPx3iBl J9nOM8OJ0V
d8WErY0gPxun8pbJLmxkAtWNhpMvfe0050vk9TL5wqbu9AlbssgTcCXkMQnPw9nC
YNN6DDP2lbcBrvgT9YCNL6C+ZKufD52yOQ9qOkwFTEQpjtF4uNtJom+asvlpmS8A
vLY9r60wYSvmZhNqBUrj7lyCtXMIu1kkd4w7F77k+DjHoAXyxcUp1DGL51sOmama
+TOWWgECgYEA8JtPxP0GRJ+IQkX262jM3dEIkza8ky5moIwUqYdsx0NxHgRRhORT

8c8hAuRBb2G82so8vUHk/fur85OEfc9TncnCY2crpoqsghifKLxrLgtT+qDpfZnx SatLdt8GfQ85yA
HCctNi/FwjulhttFx/rHYKhLidZDFYeiE/v45bN4yFm8x7R/b0iE7KaszX+Exdvt SghaT-
dcG0Knyw1bpJVyusavPzpaJMjdJ6tcFhVAbAjm7enCIvGCSx+X3l5SiWg0A R57hJglezIiVjv3aGw
Ttiek7xRVxUl+iU7rWkGAXFpMLFteQEsRr7PJ/lemmEY5eTDAFMLy9FL2m9oQWCg
R8VdwSk8r9FGLS+9aKcV5PI/WEKlwgXinB3OhYimtiG2Cg5JCqIZFHxD6MjEGOiu
L8ktHMPvodBwNsSBULpG0QKBgBAplTfC1HOnWiMGOU3KPwYWt0O6CdTkmJOmL8Ni
blh9elyZ9FsGxsgtRBXRsqXuz7wtsQAgLHxbdLq/ZJQ7YfzOKU4ZxEnabvXnvWkU
YOdjHdSOoKvDQNWu6ucyLRAWFuISeXw9a/9p7ftpxm0TSgyvmfLF2MIAEwyzRqaM
77pBAoGAMmjmIJdjp+Ez8duyn3ieo36yrttF5NSsJLAbxFpdlc1gvtGCWW+9Cq0b
dxviW8+TFVEBl1O4f7HVm6EpTscdDxU+bCXWkfjuRb7Dy9GOtt9JPsX8MBTakzh3
vBgsyi/sN3RqRBcGU40fOoZyfAMT8s1m/uYv52O6IgeuZ/ujbjY= ——END RSA PRI-
VATE KEY——

Chapter 19

Bandit Level: 17

19.1 Scenario

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.

19.2 Login Details

Username: bandit17
Password: RSA Key Generated

19.3 Method

The scenario gives us a clue in what is needed, the two files in the home directory have ASCII text within, we need to know the difference between us. The commands we can use are cat and diff to compare the two files.

```
bandit17@melinda:~$ ls
passwords.new  passwords.old
bandit17@melinda:~$ diff passwords.new passwords.old
42c42
< kfBf3eYk5BPBRzwjqutbbfE887SVc5Yd
---
> BS8bqB1kqkinKJjuxL6k072Qq9NRwQpR
bandit17@melinda:~$ ^C
bandit17@melinda:~$
```

The scenario indicates that the password is within the passwords.new as we used that one first in our command string it's the one we want.

19.4 Output

Password: kfBf3eYk5BPBRzwjqutbbfE887SVc5Yd

Chapter 20

Bandit Level:18

20.1 Scenario

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

20.2 Login Details

Username: bandit18
Password: kfBf3eYk5BPBRzwjqutbbfE887SVc5Yd

20.3 Method

This one was harder and required me to think a little harder than most challenges done before. Due to the problem of not being able to stay logged in to read the readme file, I thought about trying to connect through my cmd but due to ssh not being installed and the hassle of me installing it (I know it wouldn't take long but hay), I thought why not log in as level 0 and ssh in to level 18 this way.

So logging into bandit0 we can begin to ssh into bandit18, however we still faced

with the problem with getting disconnected. SSH will also allow us to run commands along with connecting to the server, simply by adding it to the end of the command string. Using the cat command to output the text of readme, we get the password just before disconnecting.

```
bandit0@melinda:~$ ssh bandit18@localhost cat readme
Could not create directory '/home/bandit0/.ssh'.
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is 05:3a:1c:25:35:0a:ed:2f:cd:87:1c:f6:fe:69:e4:f6.
Are you sure you want to continue connecting (yes/no)? yes
Failed to add the host to the list of known hosts (/home/bandit0/.ssh/known_host
s).

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

Please note that wargame usernames are no longer level<X>, but wargamename<X>
e.g. vortex4, semtex2, ...

Note: at this moment, blacksun is not available.

bandit18@localhost's password:
IueksS7Ubh8G3DCwVzrTd8rAVOwq3M5x
bandit0@melinda:~$
```

20.4 Output

Password: IueksS7Ubh8G3DCwVzrTd8rAVOwq3M5x

Chapter 21

Bandit Level:19

21.1 Scenario

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 to setuid binary.

21.2 Login Details

Username: bandit18
Password: kfBf3eYk5BPBRzwjqutbbfE887SVc5Yd

21.3 Method

This challenge is using the binary file as an enviroment to run commands. This is to be able to act as bandit20 via bandit19. Similar in the way we SSH between two users before.

Opening the SSH tunnel we have a file called bandit20-do. This is the environment we can use to masquerade as bandit20.

If we run the file it provides us with an example on how to use it.

```
bandit19@melinda:~$ ./bandit20-do
Run a command as another user.
  Example: ./bandit20-do id
bandit19@melinda:~$
```

From the example we can see that we can run commands alongside, such as the id command, but also we can use whoami, cat, ls,etc...

Knowing we can run commands we can output the password located in the "etc/bandit_pass" file. Resulting in the password we need for level 20.

```
bandit19@melinda:~$ ./bandit20-do cat /etc/bandit_pass/bandit20
GbKksEFF4yrVs6il55v6gwY5aVje5f0j
bandit19@melinda:~$
```

21.4 Output

Password: GbKksEFF4yrVs6il55v6gwY5aVje5f0j

Chapter 22

Bandit Level:20

22.1 Scenario

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: To beat this level, you need to login twice: once to run the setuid command, and once to start a network daemon to which the setuid will connect.

NOTE 2: Try connecting to your own network daemon to see if it works as you think

22.2 Login Details

Username: bandit20
Password: GbKksEFF4yrVs6il55v6gwY5aVje5f0j

22.3 Method

Logging into the level we see that we have another environment to use similar to the last challenge, running it "./suconnect" gives us information about the program and how it works.

This challenge requires two shells to be used in combination with each other. One shell to load the netcat listening host, and the other to connect to it.

Because we want to connect locally we can just run nc and a port.

```
bandit20@melinda:~$ nc -l 2000
```

Now that we have a listening host we can connect to it with the ./suconnect environment in the second shell to generate our password for the next level.

```
bandit20@melinda:~$ ./suconnect 2000
```

Now we need to go to the previous shell and send the bandit20 level password through netcat, for ./suconnect to read.

```
bandit20@melinda:~$ nc -l 2000
GbKksEFF4yrVs6il55v6gwY5aVje5f0j
```

Shell 2 will read the password in and confirm that the passwords match and send level 21's password back to shell 1.

```
bandit20@melinda:~$ ./suconnect 2000
Read: GbKksEFF4yrVs6il55v6gwY5aVje5f0j
Password matches, sending next password
```

```
bandit20@melinda:~$ nc -l 2000
GbKksEFF4yrVs6il55v6gwY5aVje5f0j
gE269g2h3mw3pwgrj0Ha9Uoqen1c9DGr
bandit20@melinda:~$
```

22.4 Output

Password: gE269g2h3mw3pwgrj0Ha9Uoqen1c9DGr

Chapter 23

Bandit Level:21

23.1 Scenario

A program is running automatically at regular intervals from cron, the time-based job scheduler. Look in /etc/cron.d/ for the configuration and see what command is being executed.

23.2 Login Details

Username: bandit21
Password: gE269g2h3mw3pwgrj0Ha9Uoqen1c9DGr

23.3 Method

Cron jobs are linux's utility method of running programs through a scheduler. This automises the process's and services.

if we change directory to the /etc/cron.d folder, and run ls -la we can see what cron jobs are being run.

```
-r--r-----   1 root root   46 Nov 14  2014 behemoth4_cleanup
-rw-r--r--   1 root root  355 May 25  2013 cron-apt
-rw-r--r--   1 root root   61 Nov 14  2014 cronjob_bandit22
-rw-r--r--   1 root root   62 Nov 14  2014 cronjob_bandit23
-rw-r--r--   1 root root   61 May  3  2015 cronjob_bandit24
-rw-r--r--   1 root root   62 May  3  2015 cronjob_bandit24_root
-r--r-----   1 root root   47 Nov 14  2014 leviathan5_cleanup
-rw-------   1 root root  233 Nov 14  2014 manpage3_resetpw_job
-rw-r--r--   1 root root   51 Nov 14  2014 melinda-stats
-rw-r--r--   1 root root   54 Nov 14  2014 natas-session-toucher
-rw-r--r--   1 root root   49 Nov 15  2014 natas-stats
-r--r-----   1 root root   44 Aug  3  2015 natas25_cleanup
-r--r-----   1 root root   47 Aug  3  2015 natas25_cleanup~
-r--r-----   1 root root   47 Nov 14  2014 natas26_cleanup
-r--r-----   1 root root   43 Nov 15  2014 natas27_cleanup
-rw-r--r--   1 root root  510 Oct 29  2014 php5
-rw-r--r--   1 root root   63 Jul  8  2015 semtex0-32
-rw-r--r--   1 root root   63 Jul  8  2015 semtex0-64
-rw-r--r--   1 root root   64 Jul  8  2015 semtex0-ppc
-rw-r--r--   1 root root   35 Nov 14  2014 semtex5
-rw-r--r--   1 root root  396 Nov 10  2013 sysstat
-rw-r--r--   1 root root   29 Nov 14  2014 vortex0
-rw-r--r--   1 root root   30 Nov 14  2014 vortex20
bandit21@melinda:/etc/cron.d$
```

We can see that cronjob_bandit22 is running (our objective for this challenge is to get the password for bandit22).

If we cat cronjob_bandit22, we can see what is being run.

```
bandit21@melinda:/etc/cron.d$ cat cronjob_bandit22
* * * * * bandit22 /usr/bin/cronjob_bandit22.sh &> /dev/null
bandit21@melinda:/etc/cron.d$
```

We can see that it is running an sh command, via the directory /usr/bin/cronjob_bandit22.sh.

If we can this file path,we see that it is dumping the something within the tmp directory.

```
bandit21@melinda:/etc/cron.d$ cat /usr/bin/cronjob_bandit22.sh
#!/bin/bash
chmod 644 /tmp/t7O6lds9S0RqQh9aMcz6ShpAoZKF7fgv
cat /etc/bandit_pass/bandit22 > /tmp/t7O6lds9S0RqQh9aMcz6ShpAoZKF7fgv
bandit21@melinda:/etc/cron.d$
```

If we further then read the directory that the cronjob is dumping to we get the password.

```
bandit21@melinda:/etc/cron.d$ cat /tmp/t7O6lds9S0RqQh9aMcz6ShpAoZKF7fgv
Yk7owGAcWjwMVRwrTesJEwB7WVOiILLI
```

23.4 Output

Password: Yk7owGAcWjwMVRwrTesJEwB7WVOiILLI

Chapter 24

Bandit Level:22

24.1 Scenario

A program is running automatically at regular intervals from cron, the time-based job scheduler. Look in /etc/cron.d/ for the configuration and see what command is being executed.

NOTE: Looking at shell scripts written by other people is a very useful skill. The script for this level is intentionally made easy to read. If you are having problems understanding what it does, try executing it to see the debug information it prints.

24.2 Login Details

Username: bandit22
Password: Yk7owGAcWjwMVRwrTesJEwB7WVOiILLI

24.3 Method

Firstly we need to change directory to /etc/cron.d. Once we have done that we can ls -la to list the contents of the directory. Once we have done that we can identify

that there is cronjob_bandit23 running which is our objective.

We can output the cronjob by doing cat cronjob_banit23

```
-r--r-----   1 root root   46 Nov 14  2014 behemoth4_cleanup
-rw-r--r--   1 root root  355 May 25  2013 cron-apt
-rw-r--r--   1 root root   61 Nov 14  2014 cronjob_bandit22
-rw-r--r--   1 root root   62 Nov 14  2014 cronjob_bandit23
-rw-r--r--   1 root root   61 May  3  2015 cronjob_bandit24
-rw-r--r--   1 root root   62 May  3  2015 cronjob_bandit24_root
-r--r-----   1 root root   47 Nov 14  2014 leviathan5_cleanup
-rw-------   1 root root  233 Nov 14  2014 manpage3_resetpw_job
-rw-r--r--   1 root root   51 Nov 14  2014 melinda-stats
-rw-r--r--   1 root root   54 Nov 14  2014 natas-session-toucher
-rw-r--r--   1 root root   49 Nov 15  2014 natas-stats
-r--r-----   1 root root   44 Aug  3  2015 natas25_cleanup
-r--r-----   1 root root   47 Aug  3  2015 natas25_cleanup~
-r--r-----   1 root root   47 Nov 14  2014 natas26_cleanup
-r--r-----   1 root root   43 Nov 15  2014 natas27_cleanup
-rw-r--r--   1 root root  510 Oct 29  2014 php5
-rw-r--r--   1 root root   63 Jul  8  2015 semtex0-32
-rw-r--r--   1 root root   63 Jul  8  2015 semtex0-64
-rw-r--r--   1 root root   64 Jul  8  2015 semtex0-ppc
-rw-r--r--   1 root root   35 Nov 14  2014 semtex5
-rw-r--r--   1 root root  396 Nov 10  2013 sysstat
-rw-r--r--   1 root root   29 Nov 14  2014 vortex0
-rw-r--r--   1 root root   30 Nov 14  2014 vortex20
bandit22@melinda:/etc/cron.d$
```

We can see that cronjob_bandit23 is running a shell script. Which we can output as well.

```
bandit22@melinda:/etc/cron.d$ cat cronjob_bandit23
* * * * * bandit23 /usr/bin/cronjob_bandit23.sh  &> /dev/null
bandit22@melinda:/etc/cron.d$
```

Using the cat /usr/bin/cronjob_banti23.sh we can see that it is a script which takes a parameter which in this case is our username.

```
bandit22@melinda:/etc/cron.d$ cat /usr/bin/cronjob_bandit23.sh
#!/bin/bash

myname=$(whoami)
mytarget=$(echo I am user $myname | md5sum | cut -d ' ' -f 1)

echo "Copying passwordfile /etc/bandit_pass/$myname to /tmp/$mytarget"

cat /etc/bandit_pass/$myname > /tmp/$mytarget
bandit22@melinda:/etc/cron.d$
```

If we then write out the command we get the directory where our file is stored.

```
bandit21@melinda:/etc/cron.d$ cat /tmp/t7O6lds9S0RqQh9aMcz6ShpAoZKF7fgv
Yk7owGAcWjwMVRwrTesJEwB7WVOiILLI
```

Finally we can output the contents of the folder that was then given to us via the shell script and output the password.

```
bandit22@melinda:/etc/cron.d$ cat /tmp/8ca319486bfbbc3663ea0fbe81326349
jc1udXuA1tiHqjIsL8yaapX5XIAI6i0n
bandit22@melinda:/etc/cron.d$
```

24.4 Output

Password: jc1udXuA1tiHqjIsL8yaapX5XIAI6i0n

Chapter 25

Bandit Level:23

25.1 Scenario

A program is running automatically at regular intervals from cron, the time-based job scheduler. Look in /etc/cron.d/ for the configuration and see what command is being executed.

NOTE: This level requires you to create your own first shell-script. This is a very big step and you should be proud of yourself when you beat this level!

NOTE 2: Keep in mind that your shell script is removed once executed, so you may want to keep a copy around

25.2 Login Details

Username: bandit23
Password: jc1udXuA1tiHqjIsL8yaapX5XIAI6i0n

25.3 Method

Level 23, requires us to deal with cron jobs again... Changing to the /etc/cron.d directory again, and list all cronjobs running we see that cronjob_bandit24 is currently

being run.

```
-r--r-----   1 root root   46 Nov 14  2014 behemoth4_cleanup
-rw-r--r--   1 root root  355 May 25  2013 cron-apt
-rw-r--r--   1 root root   61 Nov 14  2014 cronjob_bandit22
-rw-r--r--   1 root root   62 Nov 14  2014 cronjob_bandit23
-rw-r--r--   1 root root   61 May  3  2015 cronjob_bandit24
-rw-r--r--   1 root root   62 May  3  2015 cronjob_bandit24_root
-r--r-----   1 root root   47 Nov 14  2014 leviathan5_cleanup
-rw-------   1 root root  233 Nov 14  2014 manpage3_resetpw_job
-rw-r--r--   1 root root   51 Nov 14  2014 melinda-stats
-rw-r--r--   1 root root   54 Nov 14  2014 natas-session-toucher
-rw-r--r--   1 root root   49 Nov 15  2014 natas-stats
-r--r-----   1 root root   44 Aug  3  2015 natas25_cleanup
-r--r-----   1 root root   47 Aug  3  2015 natas25_cleanup~
-r--r-----   1 root root   47 Nov 14  2014 natas26_cleanup
-r--r-----   1 root root   43 Nov 15  2014 natas27_cleanup
-rw-r--r--   1 root root  510 Oct 29  2014 php5
-rw-r--r--   1 root root   63 Jul  8  2015 semtex0-32
-rw-r--r--   1 root root   63 Jul  8  2015 semtex0-64
-rw-r--r--   1 root root   64 Jul  8  2015 semtex0-ppc
-rw-r--r--   1 root root   35 Nov 14  2014 semtex5
-rw-r--r--   1 root root  396 Nov 10  2013 sysstat
-rw-r--r--   1 root root   29 Nov 14  2014 vortex0
-rw-r--r--   1 root root   30 Nov 14  2014 vortex20
```

Doing the same method as we have done and developed from the previous two challenges we cat the cronjob to understand what it is doing.

```
bandit23@melinda:/etc/cron.d$ cat cronjob_bandit24
* * * * * bandit24 /usr/bin/cronjob_bandit24.sh &> /dev/null
```

Now we can cat the contents of the /usr/bin/cronjob_bandit24.sh to see what is being run. In this case it is another script

```
bandit23@melinda:/etc/cron.d$ cat /usr/bin/cronjob_bandit24.sh
#!/bin/bash

myname=$(whoami)

cd /var/spool/$myname
echo "Executing and deleting all scripts in /var/spool/$myname:"
for i in * .*;
do
    if [ "$i" != "." -a "$i" != ".." ];
    then
        echo "Handling $i"
        timeout -s 9 60 "./$i"
        rm -f "./$i"
    fi
done
```

From this script we can gleem some information from it, such as the directory it

changes to (/var/spool/$myname(bandit23)).

The next part is the fun part, because a cronjob is scheduled to run at intervals. We would like to create a script and place it in the /var/spool/$myname(bandit24) location. The aim is to take the contents of the directory and output it to a file for us to read.

First we need to create a directory for us to write a script to place into the bandit24 folder. This directory needs to be made in the tmp folder.

```
bandit23@melinda:/etc/cron.d$ mkdir /tmp/script23password
bandit23@melinda:/etc/cron.d$
```

Now we need to change directory to the tmp directory for us to create our script (this is because the tmp folder is the only location we can create folders on this server, as every other part of the server is locked down).

```
bandit23@melinda:/etc/cron.d$ cd /tmp/script23password
bandit23@melinda:/tmp/script23password$
```

Now that we are in our directory, we can begin to create our script, using nano (can use vi, vim, or any other command line text editor) we can create our program to be run.

```
bandit23@melinda:/tmp/script23password$ nano listprogram.sh
bandit23@melinda:/tmp/script23password$
```

The contents of this file is shown in the below image. (#!/bin/bash (indicates what interpreter to be run), cat /etc/bandit_pass/bandit24 ¿¿ /tmp/script23password/dump.txt (Outputs the password for bandit24 into the dump.txt file within the directory.).

```
#!/bin/bash
cat /etc/bandit_pass/bandit24 >> /tmp/script23password/dump.txt
```

We then need to change the permissions on this file so that it can be executed by the cronjob. To do the we need to set it to chmod 777 for read, write and execution permissions for everyone.

```
bandit23@melinda:/tmp/script23password$ chmod 777 listprogram.sh
```

In addition to the file we need the folder to have read, write permissions. To do this we chmod recursively with mode 777 for the directory.

```
bandit23@melinda:/tmp/script23password$ chmod -R 777 /tmp/script23password
```

Now that we have made our script, we can copy it over to the bandit24 directory, which was taken from the cronjobs script.

```
bandit23@melinda:/tmp/script23password$ cp listprogram.sh /var/spool/bandit24/
bandit23@melinda:/tmp/script23password$
```

After the cronjob has cycled through we can list the contents of the folder to see if it has generated our text file.

```
bandit23@melinda:/tmp/script23password$ ls
dump.txt  listprogram.sh
```

Once our dump.txt file has been generated output the contents of the file using the cat command (or opening it in a text editor such as nano, vi or vim), and we have our password for the next level.

```
UoMYTrfrBFHyQXmg6gzctqAwOmw1IohZ
```

25.4 Output

Password: UoMYTrfrBFHyQXmg6gzctqAwOmw1IohZ

Chapter 26

Bandit Level:24

26.1 Scenario

A daemon is listening on port 30002 and will give you the password for bandit25 if given the password for bandit24 and a secret numeric 4-digit pincode. There is no way to retrieve the pincode except by going through all of the 10000 combinations, called brute-forcing.

26.2 Login Details

Username: bandit24
Password: UoMYTrfrBFHyQXmg6gzctqAwOmw1IohZ

26.3 Method

From what we can see, a program which is being run in the background of the system. This daemon is listening on the port 30002 which is looking for our password for the currently level plus a four digit pin.

To do this we need to brute force the pin code to be attached to our password.

To do this we need to make a script which has a loop in it.

Firstly we need to make a directory in the tmp file.

```
bandit24@melinda:/tmp/bandit24to25$ mkdir /tmp/bandit24to25
```

Change to that directory in the temporary folder.

```
bandit24@melinda:/tmp/bandit24to25$ cd /tmp/bandit24to25
```

Thirdly we need to create our shell script to be run, to do this we use our text editor nano/vi/vim. (Exactly like the start of the previous challenge),

```
bandit24@melinda:/tmp/bandit24to25$ nano brute.sh
```

Once we have created and opened our shell script, we need to add the contents to it.

```
#!/bin/bash

passwd="UoMYTrfrBFHyQXmg6gzctqAwOmw1IohZ"

for i in {0..9}{0..9}{0..9}{0..9}
do
        echo $passwd' '$i | nc localhost 30002 >> dump.txt &
done
```

(Line by line description) 1: #!/bin/bash = Tells the interpreter what language the shell script has been written in 2: passwd =”password” = Holds our current level password in a variable 3: for i in 0..90..90..90..9 = a for loop, using the variable i and the cycle of the four digits in the range of 0-9. 5: echo $passwd’ ’ $i |nc localhost 30002 ⟩⟩dumptxt & = send the password variable (line2) and the i variable (line 3) to the netcat parameters and output the contents to the dump.txt file. The ’&’ makes the command run in the background.

Once the script has ended, we can then sort the dump.txt file and look for a uniq -u (like we have done in a previous challenge)

This will then output the password for the next level.

```
bandit24@melinda:/tmp/bandit24to25$ sort dump.txt | uniq -u

Correct!
The password of user bandit25 is uNG9O58gUE7snukf3bvZ0rxhtnjzSGzG
```

26.4 Output

Password: uNG9O58gUE7snukf3bvZ0rxhtnjzSGzG

26.5 Addendum

You can use telnet instead of netcat for the script

Chapter 27

Bandit Level:25

27.1 Scenario

Logging in to bandit26 from bandit25 should be fairly easy The shell for user bandit26 is not /bin/bash, but something else. Find out what it is, how it works and how to break out of it.

27.2 Login Details

Username: bandit25
Password: uNG9O58gUE7snukf3bvZ0rxhtnjzSGzG

27.3 Method

This challenge is quite fun, as it uses more tricks than anything.

When we log in we see that we have a ssh private key file within our directory, however if we try to connect to bandit26 we automatically get kicked.

So how to we get around it. Firstly we need to cat /etc/passwd and grep for the username (bandit26). Once we have done that we can see that there is a file /usr/bin/showtext.

We can then cat the directory and see what is inside. As we can see there is the #!/bin/sh command but more important the exit 0 which terminates our connection.

```
bandit26:x:11026:11026:bandit level 26:/home/bandit26:/usr/bin/showtext
bandit25@melinda:~$ cat /usr/bin/showtext
#!/bin/sh

more ~/text.txt
exit 0
```

So after reading around there is a cool trick of resizing the command prompt to anything below six lines. This prevents the exit 0 from running, due to the fact that the command above exit 0 in the script has the more command outputting the text file. The more command also waits for any further command to be executed.

After resizing the window we can ssh into bandit26 using the command ssh -i ¡privatekeyfile¿ bandit26@localhost

Once we ssh'd in we need to use vi as a terminal. Vi can be used as a terminal due to its set commands.

```
 _                     _ _ _   ___   __
| |                   | (_) | |__ \ / /
| |__   __ _ _ __   __| |_| |_   ) / /_
| '_ \ / _` | '_ \ / _` | | __| / / '_ \
| |_) | (_| | | | | (_| | | |_ / /| (_) |
:set shell sh=/bin/bash
```

Once we have set the shell, we can then use ":sh" to create our shell. Now we are in bandit26. Lastly we need to output the password which is located in /etc/bandit_pass/bandit26.

```
bandit26@melinda:~$ cat /etc/bandit_pass/bandit26
5czgV9L3Xx8JPOyRbXh6lQbmIOWvPT6Z
bandit26@melinda:~$
```

Alternatively to logging in with the shell. We can use -e within vi to out put the text of the password file.

Command: ":e /etc/bandit_pass/bandit26

Once you click enter the password is outputted.

27.4 Output

Password: 5czgV9L3Xx8JPOyRbXh6lQbmIOWvPT6Z