**Over The Wire Bandit Level 12-13 Walk-Through**

This is a long and frustrating level because you might end up running in circles before you figure out what is going on. You will be mostly working with the tar, gzip, bzip2, xxd, mv, file commands so it’s a good idea to figure out what each of them do before you start. When you are done with this level you will realize that the instructions told you everything you needed to know right at the start.

Bandit Level 12 → Level 13

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!)

Commands you may need to solve this level

grep, sort, uniq, strings, base64, tr, tar, gzip, bzip2, xxd, mkdir, cp, mv, file

(<https://overthewire.org/wargames/bandit/bandit13.html>)

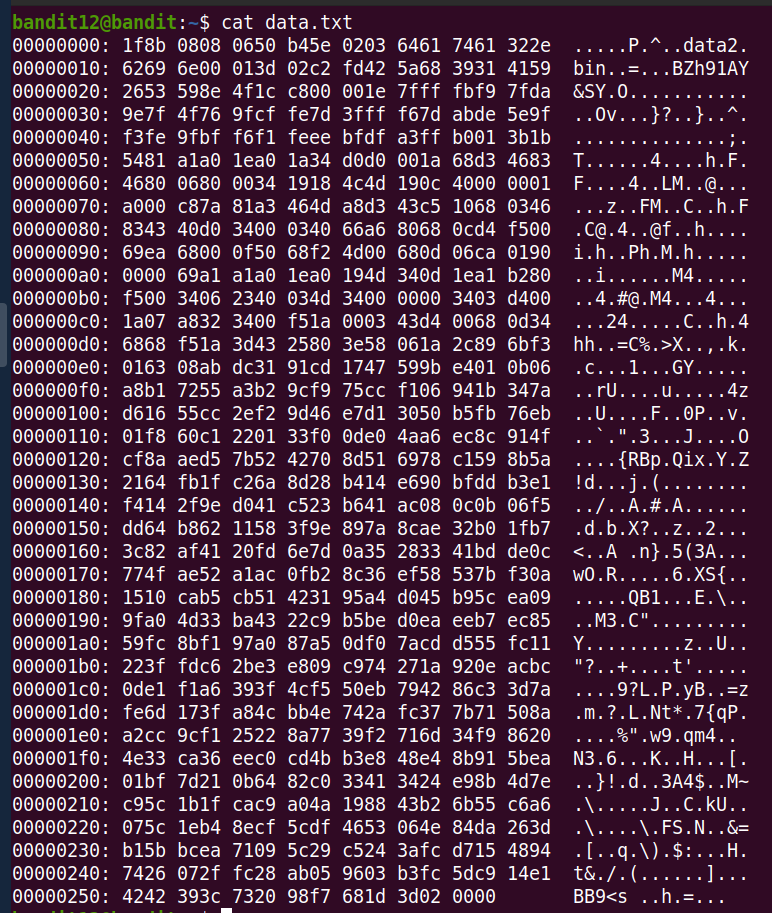
First, we need to follow the initial instructions and create your own temporary directory in the bandit’s tmp directory and copy the data.txt file into that directory. Then we change into it.

bandit12@bandit:~$ mkdir /tmp/user

bandit12@bandit:~$ cp data.txt /tmp/user

bandit12@bandit:/$ cd tmp/user

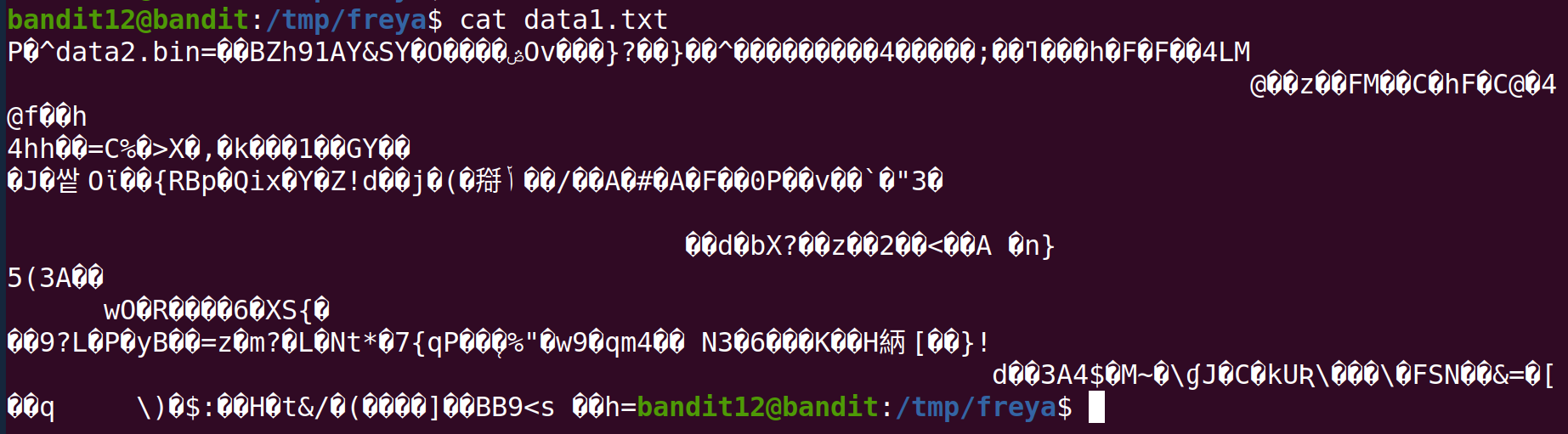
When we cat the data.txt file we will see something like this:



As the instructions said, this is a hexdump of a file we need to decompress. This means that to get to the file we need, we need to reverse the hexdump. For this we need to use the xxd command with the -r flag. We also need to save the output in a new file data1.txt.

bandit12@bandit:/tmp/user$ xxd -r data.txt data1.txt

When you cat data1.txt you will see that the file isn’t human-readable:



At this point we need to figure out what format this file is and we can use the file command for that.

bandit12@bandit:/tmp/user$ file data1.txt

data1.txt: gzip compressed data, was "data2.bin", last modified: Thu May 7 18:14:30 2020, max compression, from Unix

Again, as the instructions warned us, the file is a compressed archive, specifically a gzip archive. However, as only files with the .gz suffix can be decompressed, we first need to rename the file to the correct format using the mv command.

bandit12@bandit:/tmp/user$ mv data1.txt data1.gz

Then, we can use the gzip -d command to decompress the file. Checking with ls we see that we now have a file called data1:

bandit12@bandit:/tmp/user$ gzip -d data1.gz

bandit12@bandit:/tmp/user$ ls

data1 data.txt

When we look at what type of file data1 is we find this:

bandit12@bandit:/tmp/user$ file data1

data1: bzip2 compressed data, block size = 900k

Bzip2 is a different type of file archive and again we need to use the -d flag to decompress the file. Bzip2 can decompress without any suffix, so we don’t need to rename the file this time.

bandit12@bandit:/tmp/user$ bzip2 -d data1

bzip2: Can't guess original name for data1 -- using data1.out

Running file on data1.out will tell us that, again, it is a gzip file. This means first adding the .gz suffix and then decompressing with the gzip -d command.

bandit12@bandit:/tmp/user$ file data1.out

data1.out: gzip compressed data, was "data4.bin", last modified: Thu May 7 18:14:30 2020, max compression, from Unix

bandit12@bandit:/tmp/user$ mv data1.out data1.gz

bandit12@bandit:/tmp/user$ gzip -d data1.gz

This gets us to data1 again and this time file reveals that it is a tar archive:

bandit12@bandit:/tmp/user$ ls

data1 data.txt

bandit12@bandit:/tmp/user$ file data1

data1: POSIX tar archive (GNU)

To decompress a tar archive we need to use the tar command together with the -xf flags:

bandit12@bandit:/tmp/user$ tar -xf data1

We get a file called data5.bin which is again a tar archive. We decompress the file once more.

bandit12@bandit:/tmp/user$ ls

data1 data5.bin data.txt

bandit12@bandit:/tmp/user$ file data5.bin

data5.bin: POSIX tar archive (GNU)

bandit12@bandit:/tmp/user$ tar -xf data5.bin

This time we see data6.bin file which is a bzip2 file. Without needing to change the suffix, we decompress it with the bzip2 command.

bandit12@bandit:/tmp/user$ ls

data1 data5.bin data6.bin data.txt

bandit12@bandit:/tmp/user$ file data6.bin

data6.bin: bzip2 compressed data, block size = 900k

bandit12@bandit:/tmp/user$ bzip2 -d data6.bin

bzip2: Can't guess original name for data6.bin -- using data6.bin.out

The new file, data6.bin.out, is a tar archive which we decompress with the tar command.

bandit12@bandit:/tmp/user$ ls

data1 data5.bin data6.bin.out data.txt

bandit12@bandit:/tmp/user$ file data6.bin.out

data6.bin.out: POSIX tar archive (GNU)

bandit12@bandit:/tmp/user$ tar -xf data6.bin.out

The latest file is data8.bin which file tells us is a gzip.

bandit12@bandit:/tmp/user$ ls

data1 data5.bin data6.bin.out data8.bin data.txt

bandit12@bandit:/tmp/user$ file data8.bin

data8.bin: gzip compressed data, was "data9.bin", last modified: Thu May 7 18:14:30 2020, max compression, from Unix

A gzip file needs a .gz suffix before decompressing, so let’s rename it first and then unzip with gzip -d:

bandit12@bandit:/tmp/user$ mv data8.bin data8.bin.gz

bandit12@bandit:/tmp/user$ gzip -d data8.bin.gz

Now we check the data8.bin file for its format and we see it’s an ASCII file.

bandit12@bandit:/tmp/user$ ls

data1 data5.bin data6.bin.out data8.bin data.txt

bandit12@bandit:/tmp/user$ file data8.bin

data8.bin: ASCII text

We open the file and yes! There’s your password!

bandit12@bandit:/tmp/user$ cat data8.bin

The password is 8ZjyCRiBWFYkneahHwxCv3wb2a1ORpYL