

Ahmet Putun

CMPSC443

MD5 Collision Attack Lab

10/13/2019

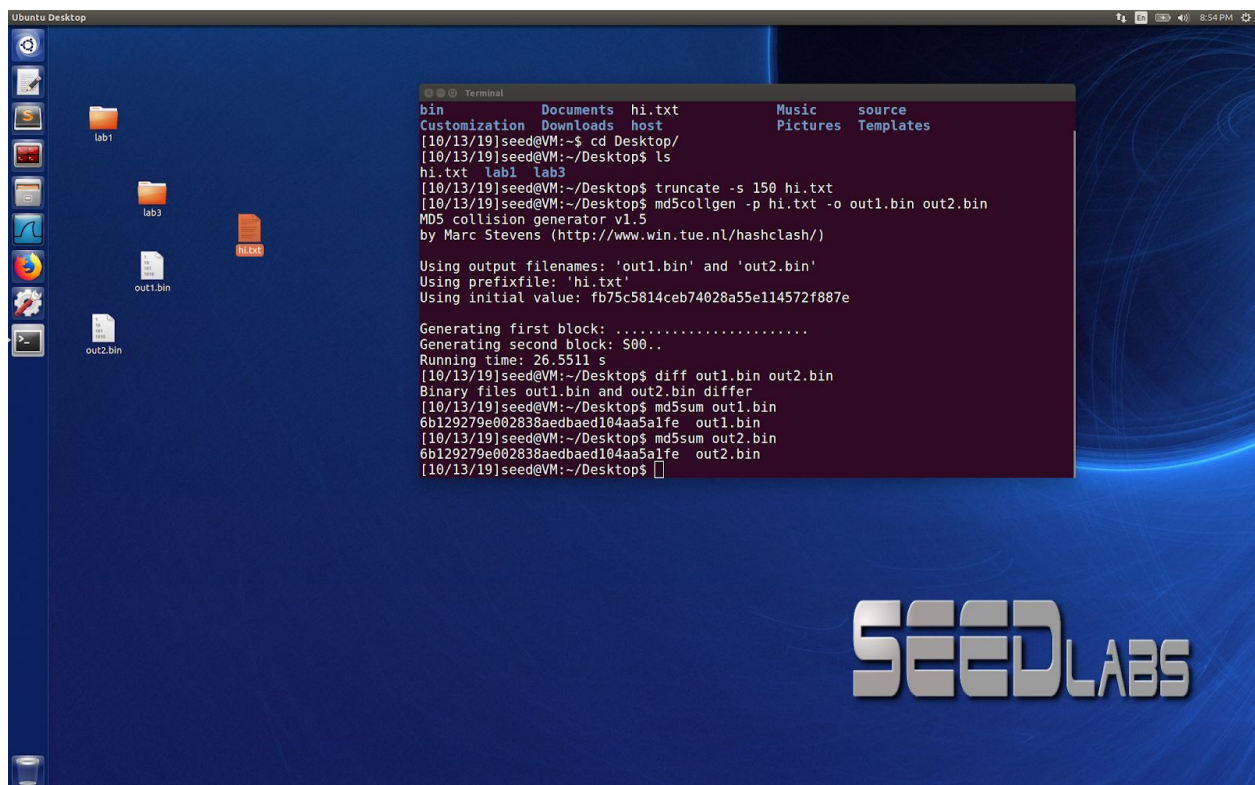
Mr. Sencun Zhu

Task 1: Generating Two Different Files with the Same MD5 Hash

Q1) **a)** *If the length of your prefix file is not a multiple of 64, what is going to happen?*

I created a short text file name hi.txt and truncated it to 150 bytes with the command *truncate -s 150 hi.txt*.

I then run *md5collgen -p hi.txt -o out1.bin out2.bin*



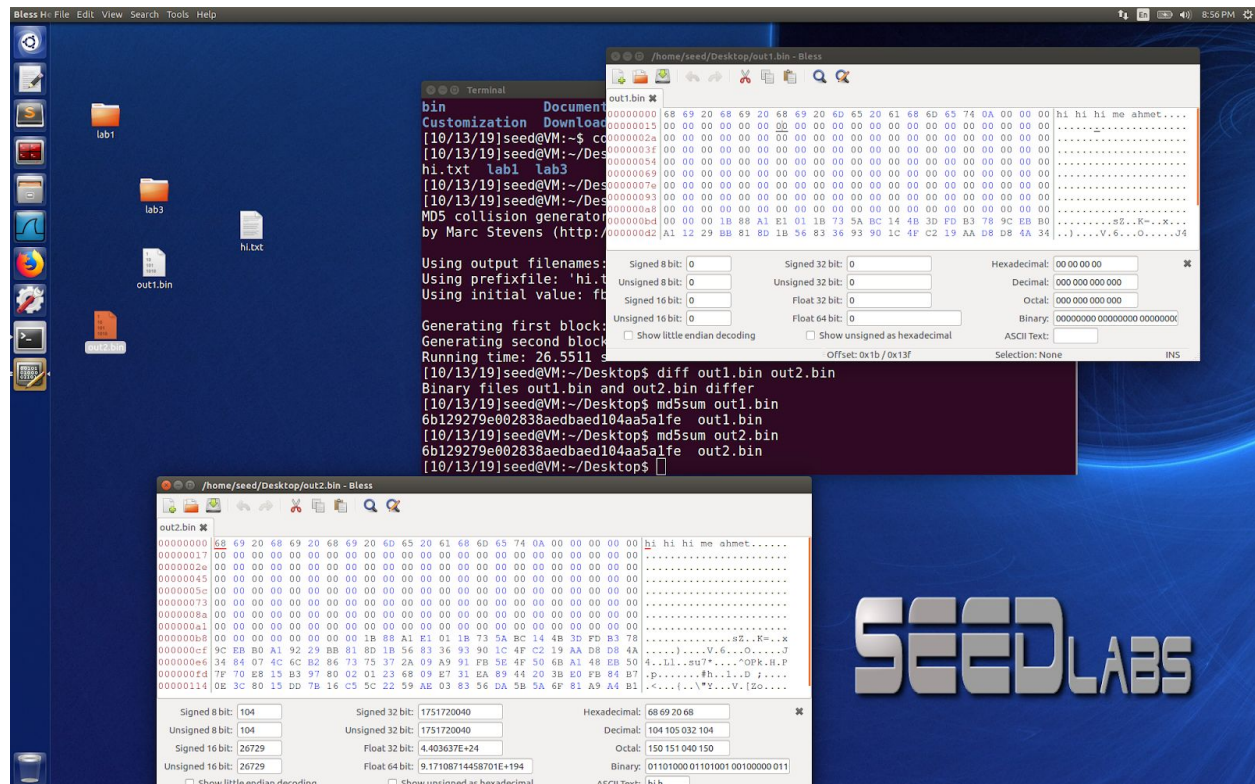
```
bin      Documents  hi.txt      Music      source
Customization Downloads host  Pictures  Templates

[10/13/19]seed@VM:~$ cd Desktop/
[10/13/19]seed@VM:~/Desktop$ ls
hi.txt  lab1  lab3
[10/13/19]seed@VM:~/Desktop$ truncate -s 150 hi.txt
[10/13/19]seed@VM:~/Desktop$ md5collgen -p hi.txt -o out1.bin out2.bin
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Using output filenames: 'out1.bin' and 'out2.bin'
Using prefixfile: 'hi.txt'
Using initial value: fb75c5814ceb74028a55e114572f887e

Generating first block: .....
Generating second block: S00..
Running time: 26.5511 s
[10/13/19]seed@VM:~/Desktop$ diff out1.bin out2.bin
Binary files out1.bin and out2.bin differ
[10/13/19]seed@VM:~/Desktop$ md5sum out1.bin
6b129279e002838aedbaed104aa5a1fe  out1.bin
[10/13/19]seed@VM:~/Desktop$ md5sum out2.bin
6b129279e002838aedbaed104aa5a1fe  out2.bin
[10/13/19]seed@VM:~/Desktop$
```

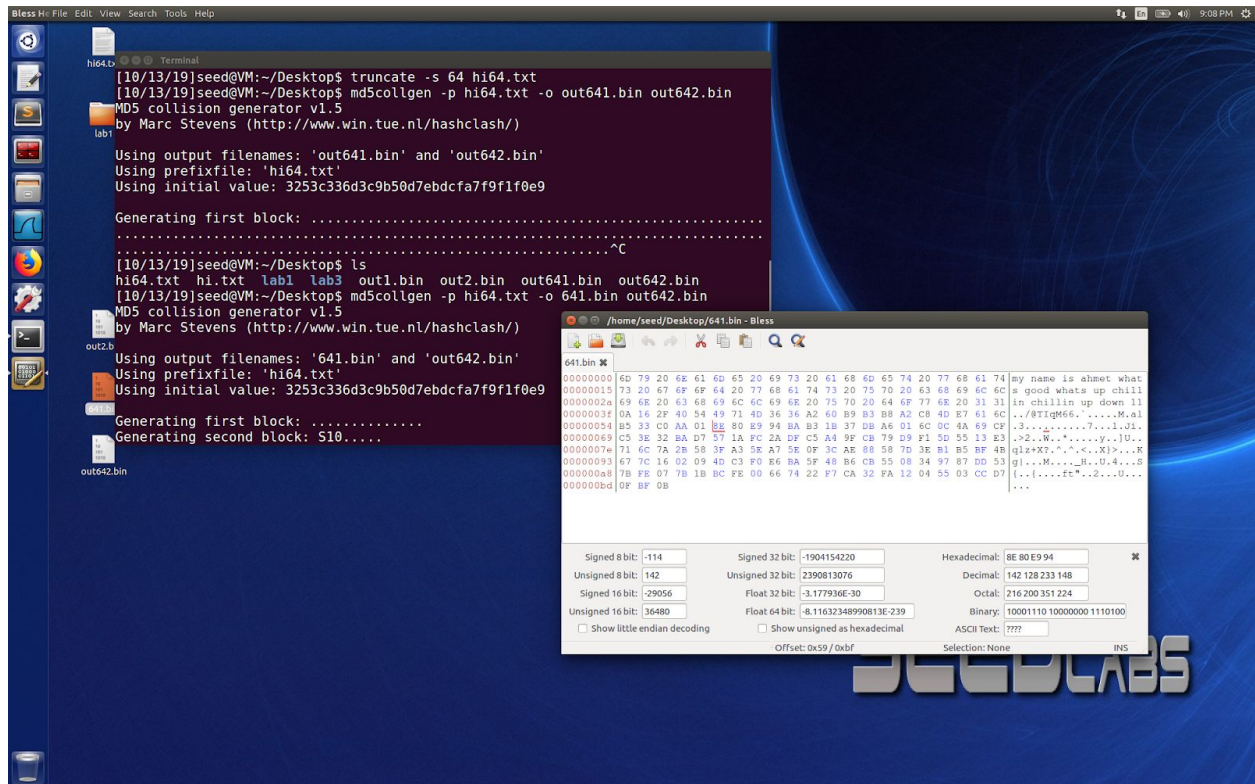
a) Looking at the binary file by the program Bless, it is seen that padding has been implied.



Q1) **b)** Create a prefix file with exactly 64 bytes, and run the collision tool again, and see what happens.

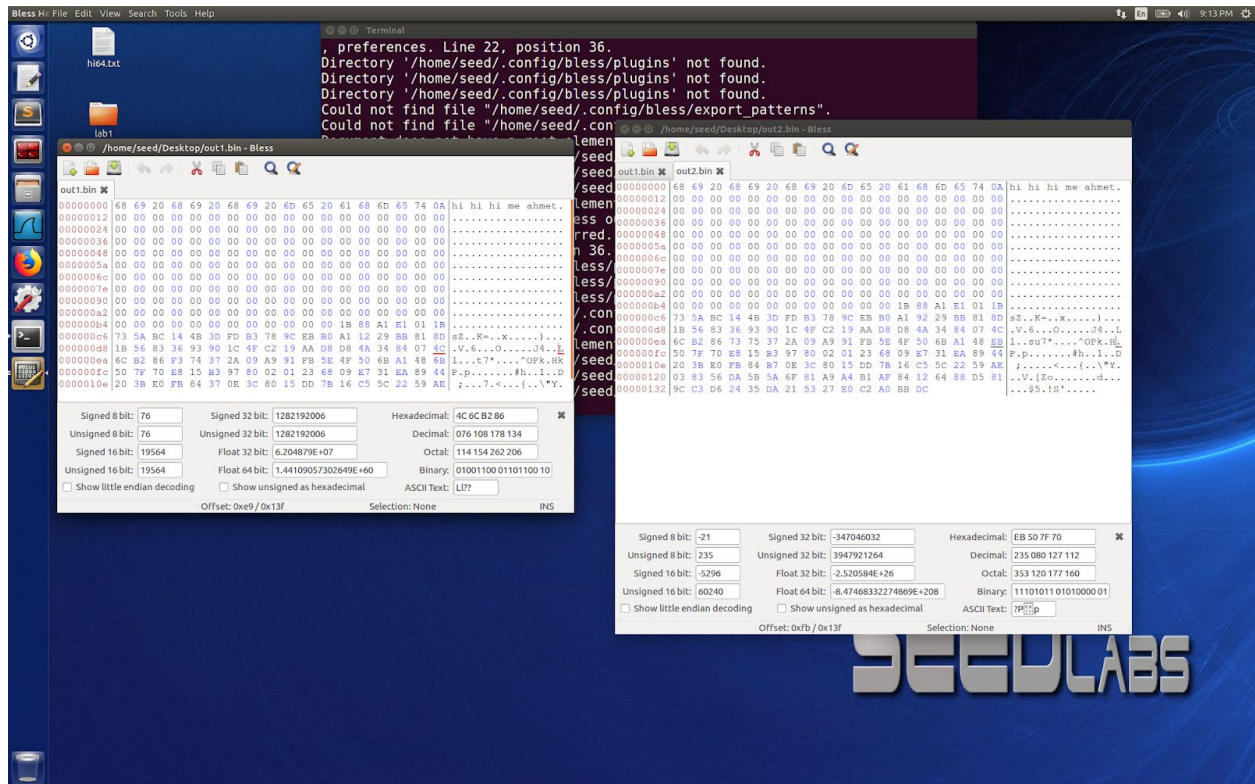
b) For this question, I created a text file that is exactly 64 bytes, truncated it and I then ran `md5collgen -p hi64.txt -o 641.bin 642.bin`.

Looking at the binary files of 641.bin, no 0 padding has been observed.



Q1) **c)** Are the data (128 bytes) generated by md5collgen completely different for the two output files? Please identify all the bytes that are different.

c) No, not all bytes are different. Bytes only differ at certain positions but the differences are not constant. In the image underneath, I provided an image of certain letter address or byte changes.



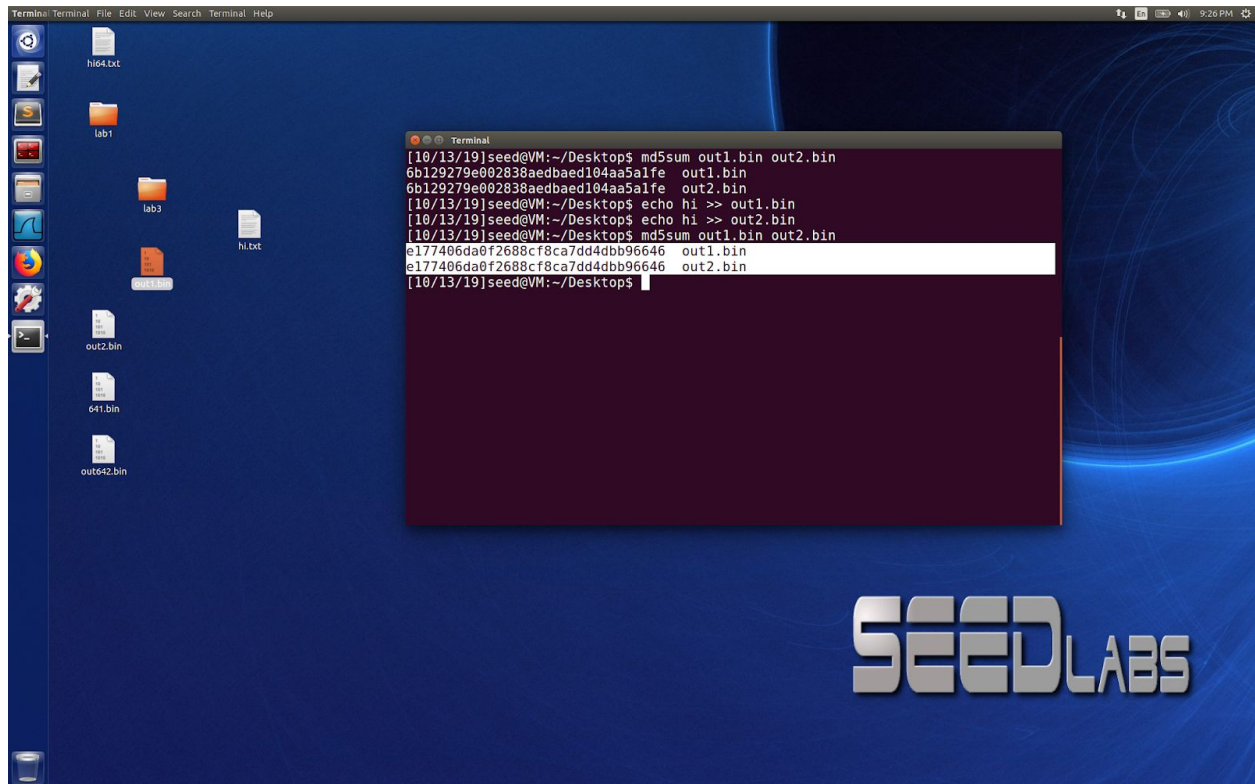
2.2 Task 2: Understanding MD5's Property

When appending the same file to the outputs generated by md5collgen, while it changes the MD5 hash, both the new hashes will be identical and this is because of many hash functions are related to length extension.

So, for testing I again created a text file hi.txt and ran `md5collgen -p hi.txt -o out1.bin out2bin`.

I then verified that hashes are the same by the command `md5sum out1.bin out2.bin`.

To test, I appended a random string to the end of both files and checked the MD5 hashes of the both files by `echo >> out1.bin` and `echo >> out2.bin` and then checked the both files again.



Answer: MD5 hashes remain identical.

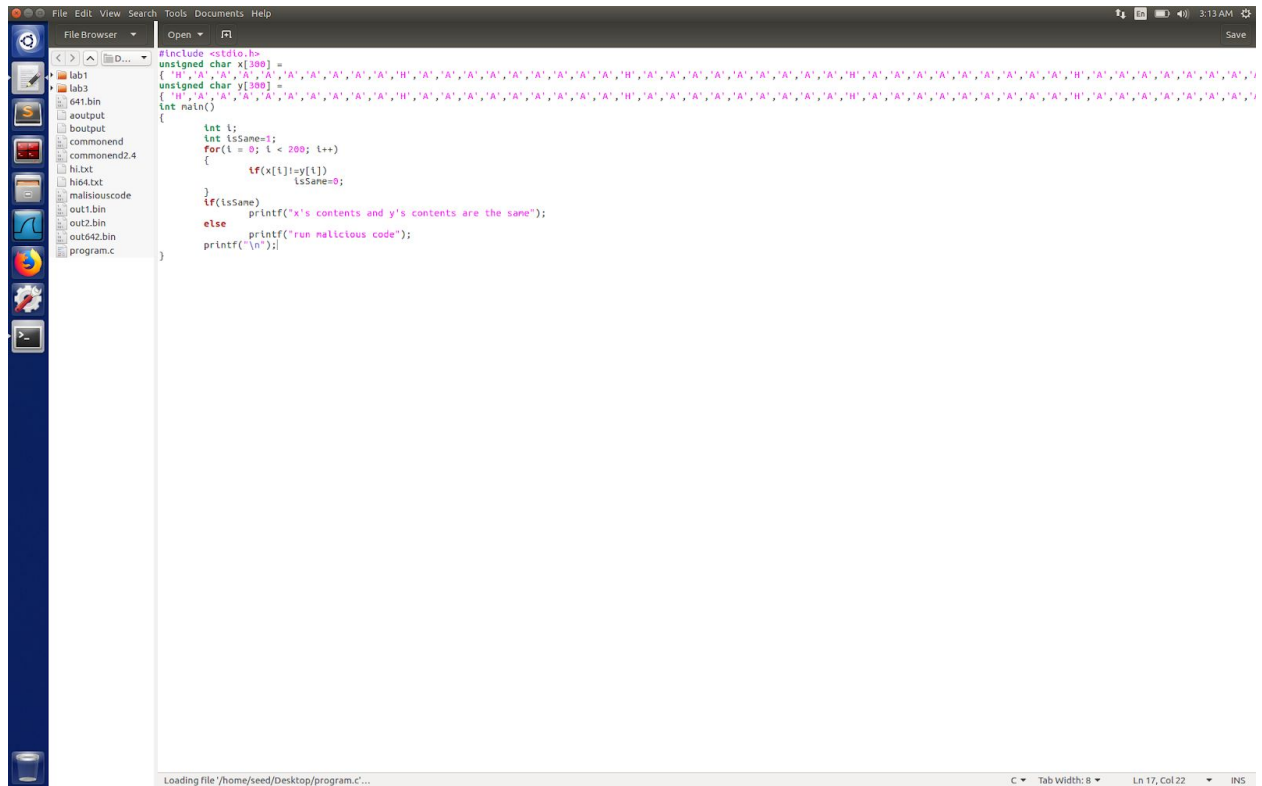
2.3 Task 3: Generating Two Executable Files with the Same MD5 Hash

For this task, I created a .c file with the code provided in the lab, and then I filled the code, array with assigned string letters like 'H' and 'A'.

[illegible]

Using diff, we can see that the output is different.

For this, we have to create two programs which have the same MD5 sum, but behave differently. For the sake of the assignment, in the end our program should print out whether the “bad” or “good” code will execute.

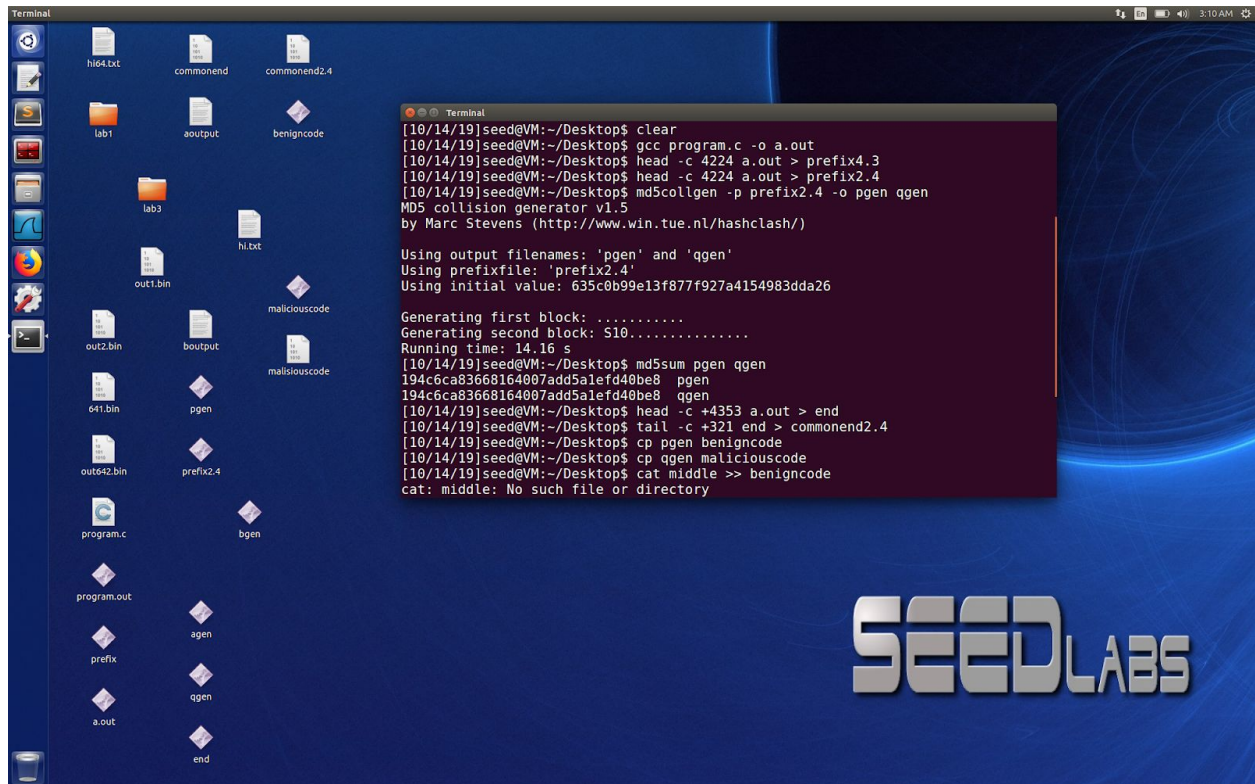


I then compiled the .c file in order to get a binary file like in the previous tasks. Using command

```
gcc program.c -o a.out
```

Referring to the first 4224 bytes of data, running `head -c 4224 a.out > prefix2.4`. Then running `md5collgen -p prefix -o pgen qgen` which will generate two binary files with the same MD5sum but they will differ.

Using the `bleed` application for binary file viewing, we can see the 128 characters generated by `md5sum` are identical. We check this by `md5sum pgen qgen`



After, I ran the similar code for the common middle and end file, `tail -c 321 end > commonend2.4` (depending on your size of middle, offset)

Putting all the files together and verifying if the hashes are the same by running the command `md5sum benigncode maliciouscode`.

Checking the sizes of those files are the same and running an executable code, check screenshot below, we achieved the desired outcome of making 2 programs behave differently.

