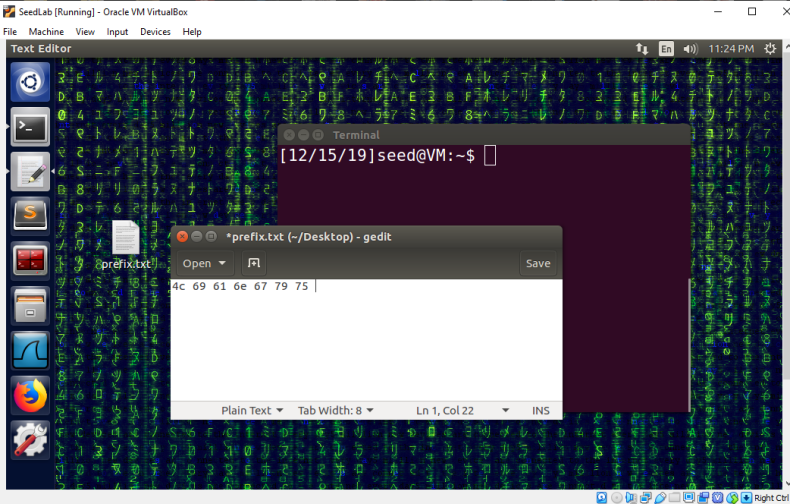
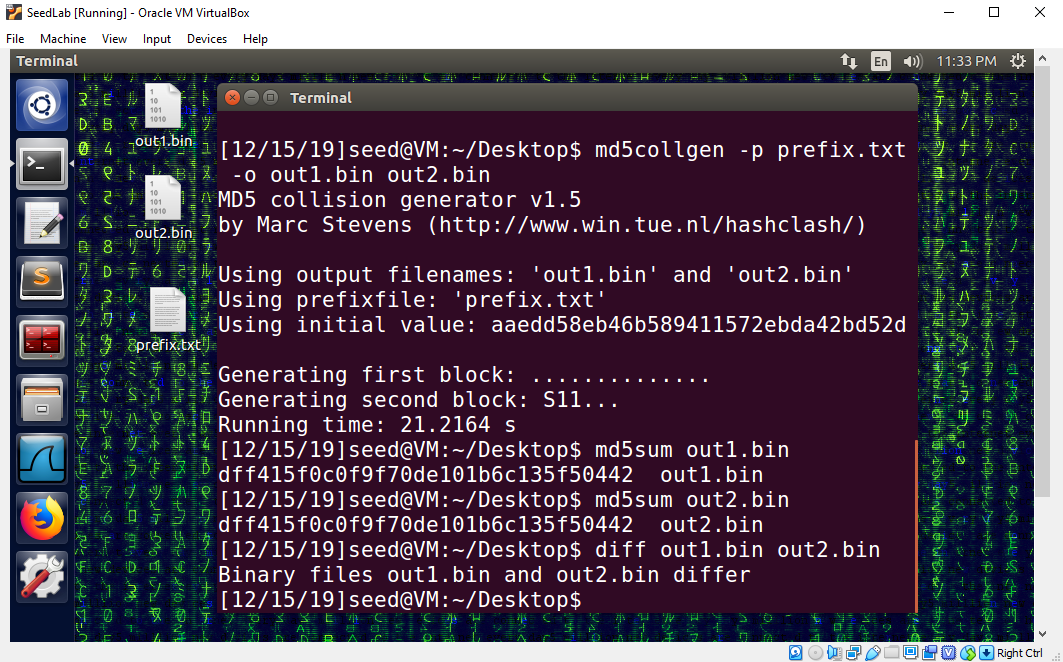
**MD5 Collision Attack Lab**

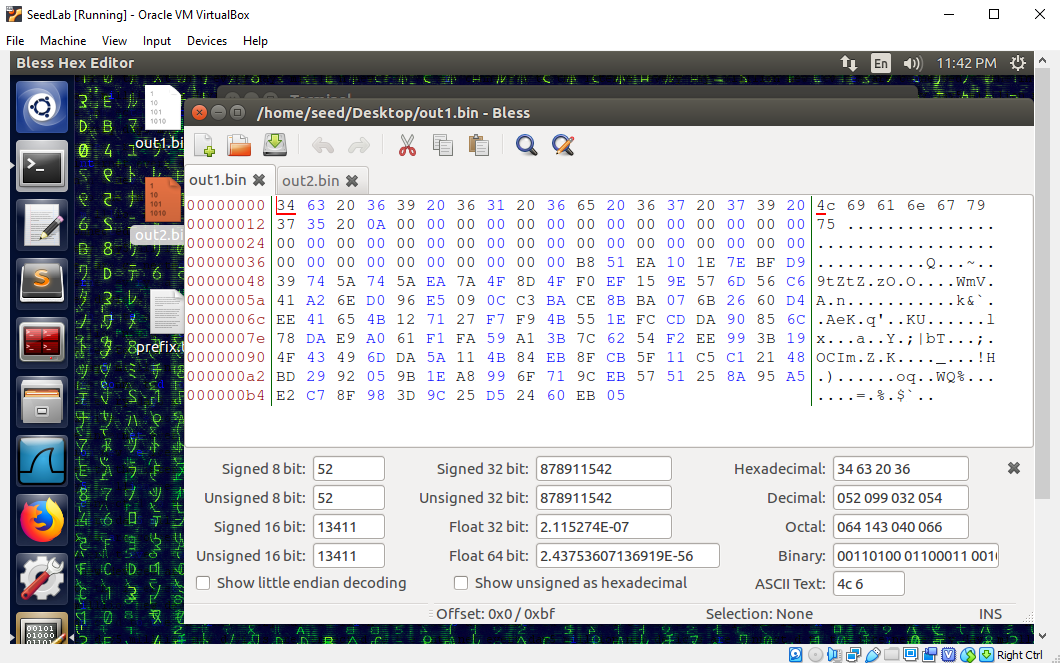
Liangyu W

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

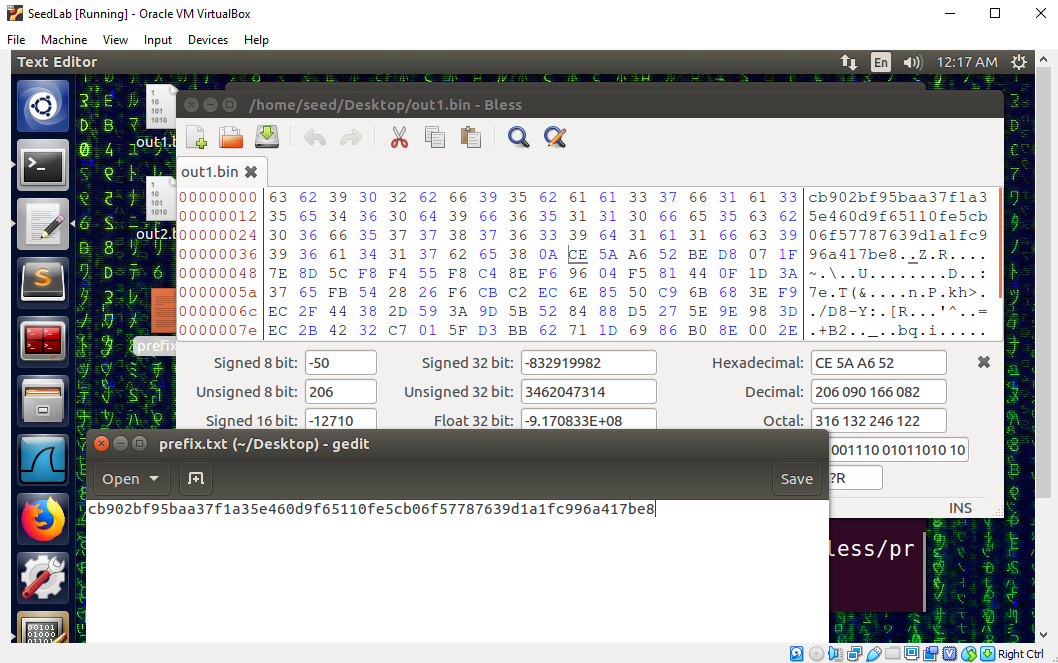


We create a prefix file prefix.txt then run the md5collgen program to generate two files out1.bin and out2.bin with the same md5 hash.

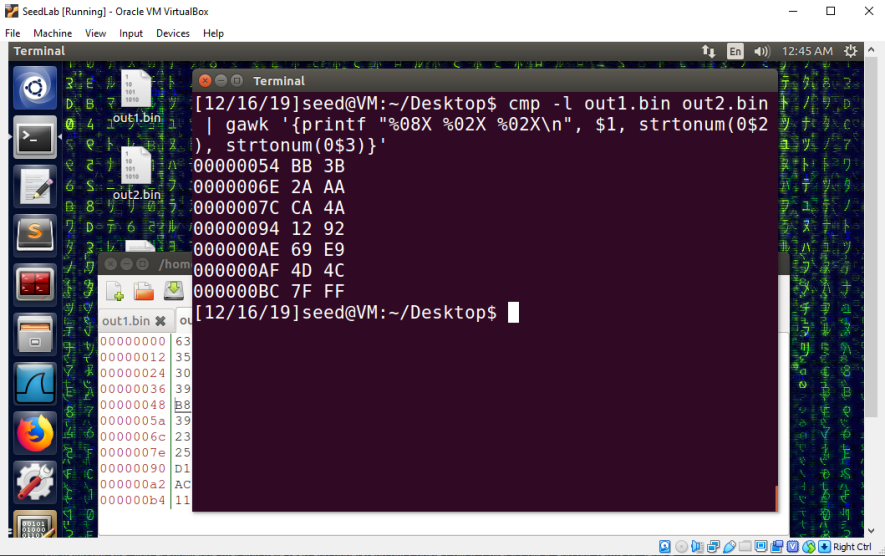




By using the Bless hex editor, we see that when content of the prefix file is not a multiple of 64 bytes, zeros are padded. This is because md5 processes blocks of 64 bytes.



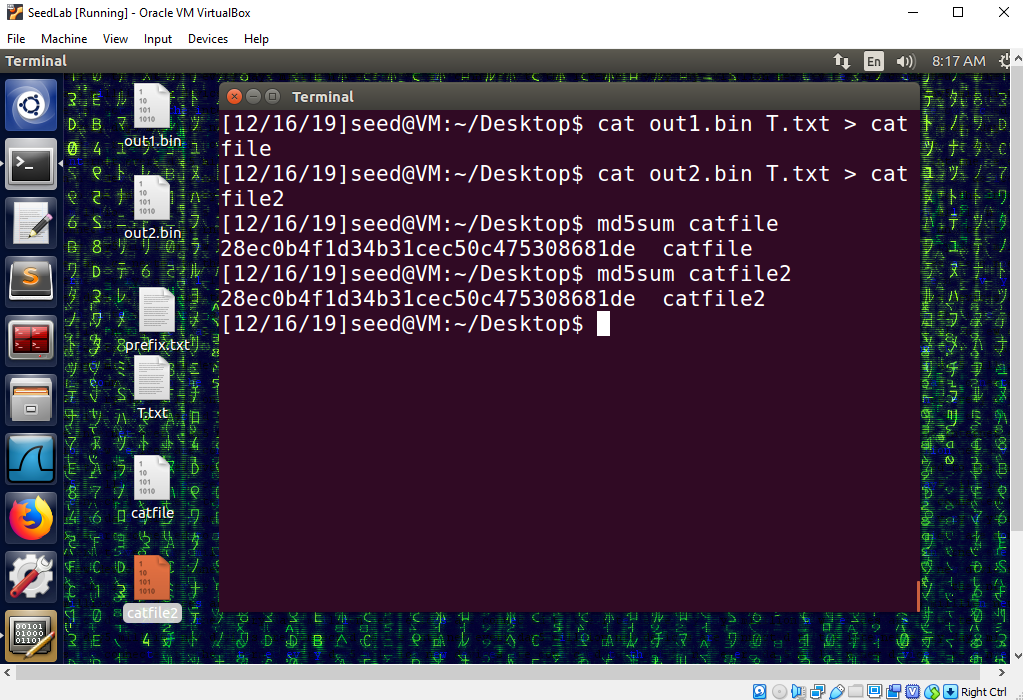
If we use a prefix of exactly 64 bytes, then the md5collgen generated files have no zero padding.



Using the cmp command, we can see all the differences between the two generated files.

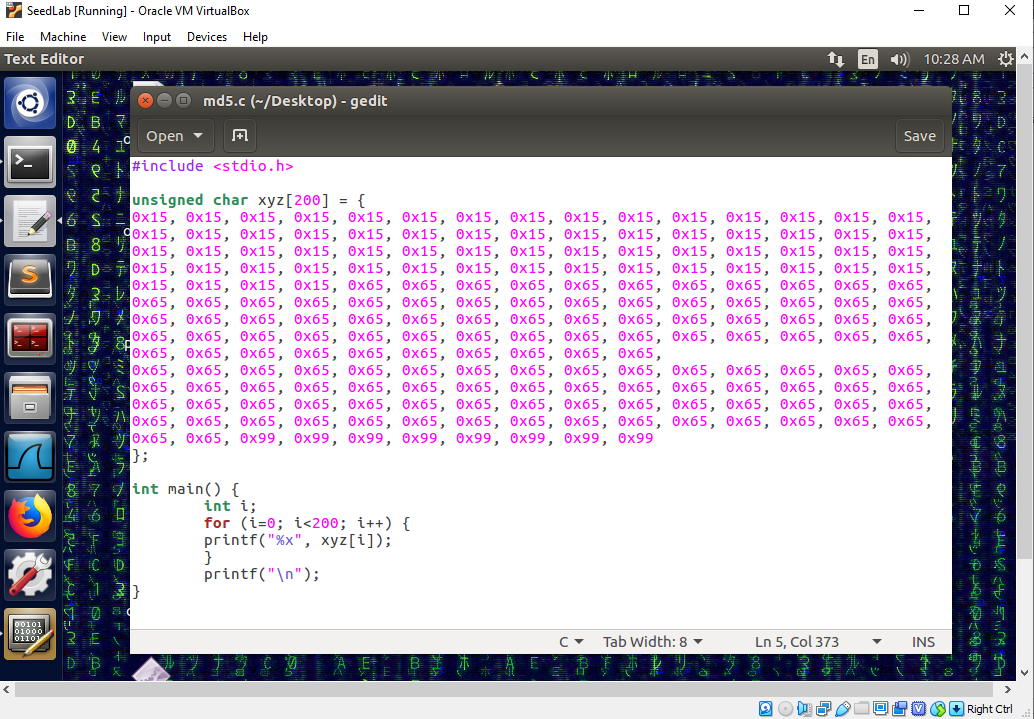
**Task 2: Understanding MD5’s Property**

We create a text file T.txt filled with random bytes

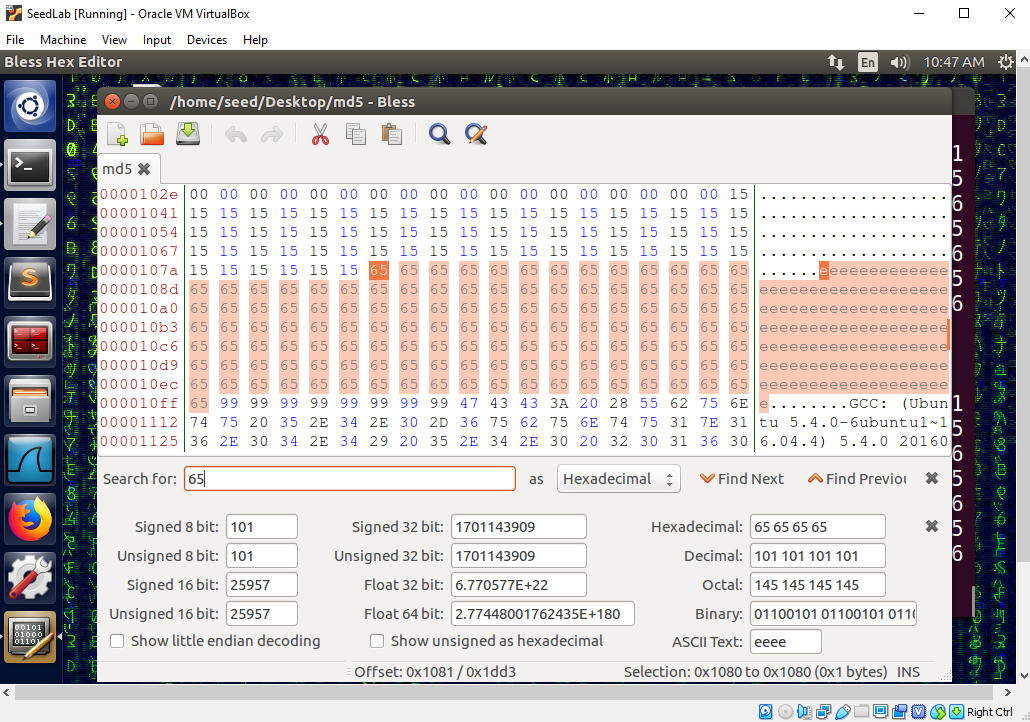


Then used the cat command to concatenate T.txt to out1.bin and out2.bin, then ran md5sum on the two generated files. The two generated files also have identical MD5 hashes, thus confirming the common Hash function property that if M and N have the same MD5 hash, then for any input T, MD5(M‖T) = MD5(N‖T), where‖ represents concatenation.

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

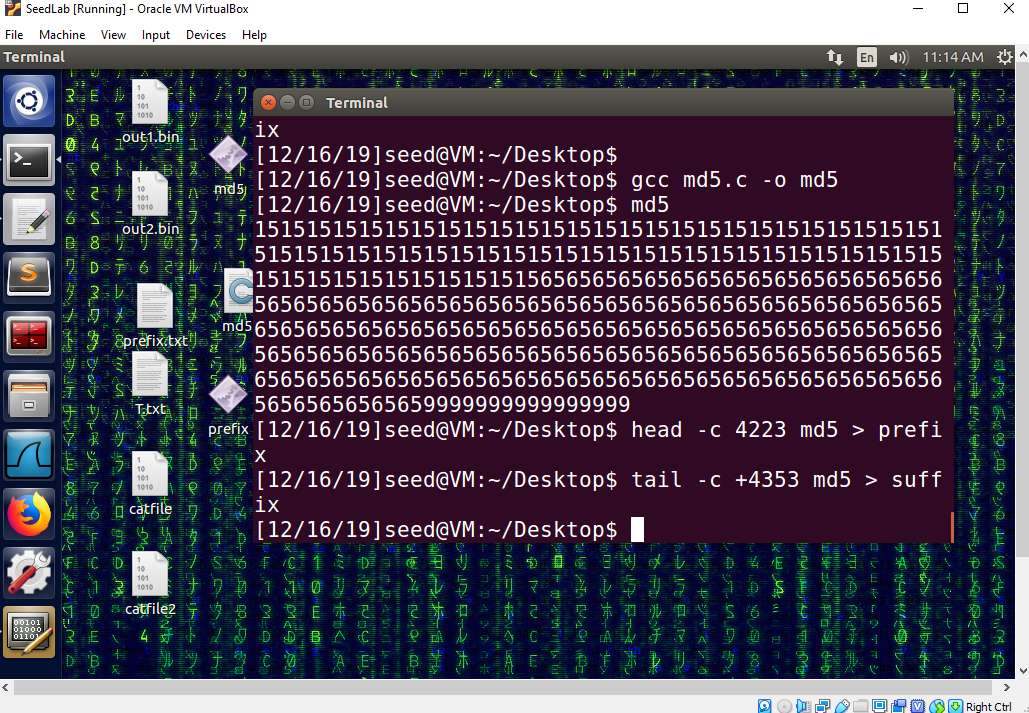


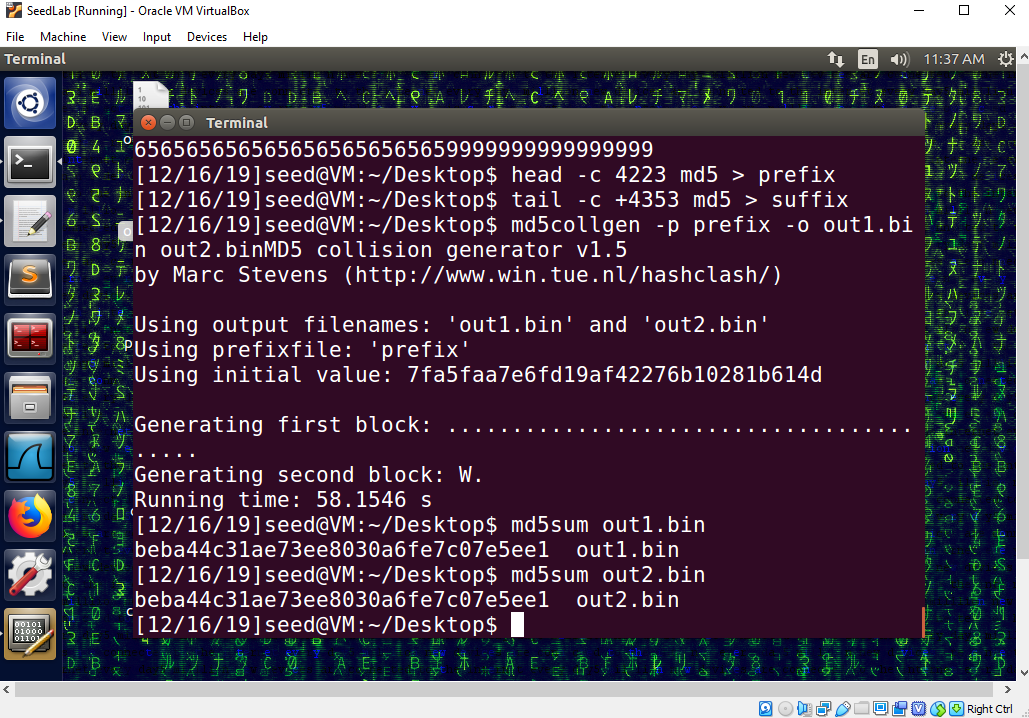
In the xyz array, we fill prefix with 0x15, the 128-byte region with 0x65 and suffix with 0x99.



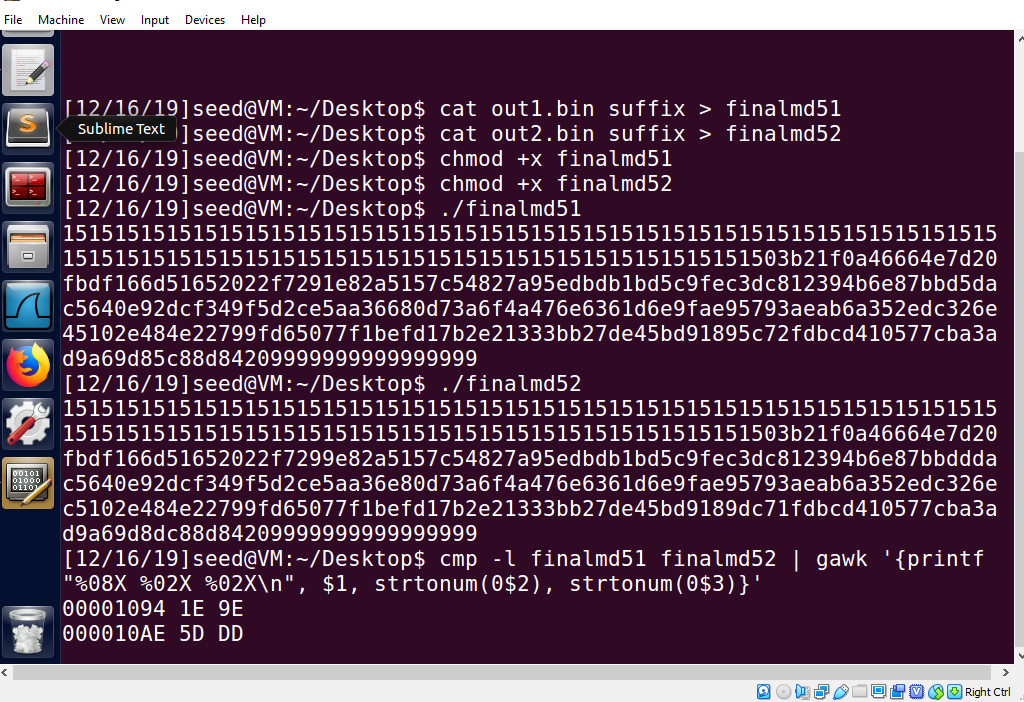
We find that the xyz array content is stored at offset 0x1040 - 0x1107 in the compiled file. The prefix of the array is stored at 0x1040 - 0x107f. The suffix of the array is stored from offset 0x1100 to the end of the file. 107f converted to decimal is 4223, 0x1100 converted to decimal is 4352.

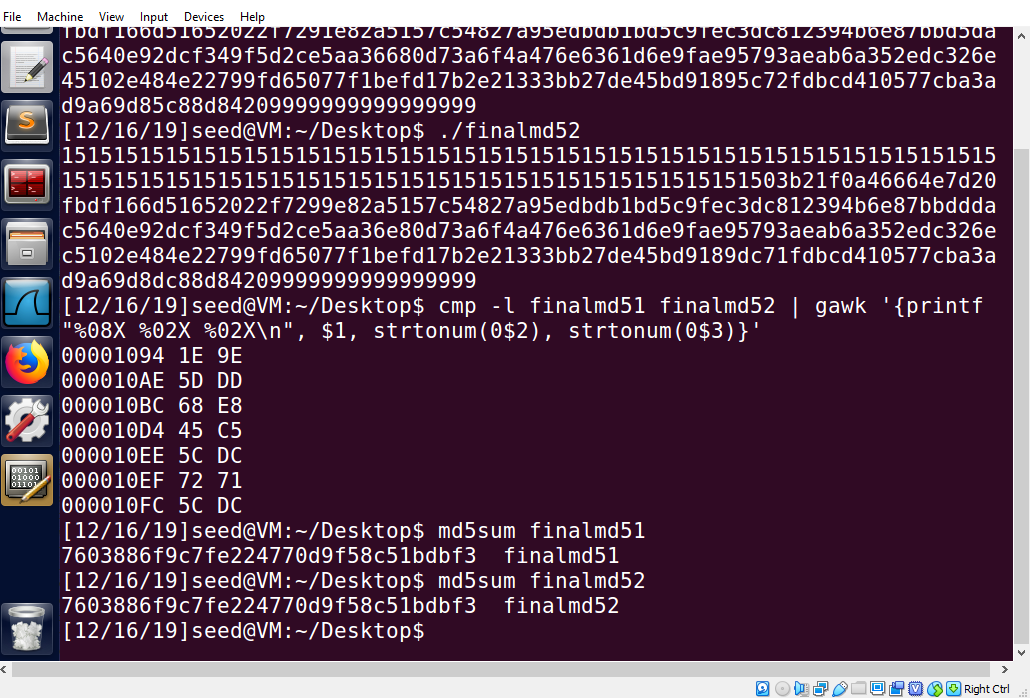
So we use the head command to slice and copy the first part of the binary file into prefix file. Then use the tail command to slice and copy the last part of the binary file into suffix file.





We ran md5collgen on prefix file and generated two binary files out1.bin and out2.bin with identical md5 hashes.

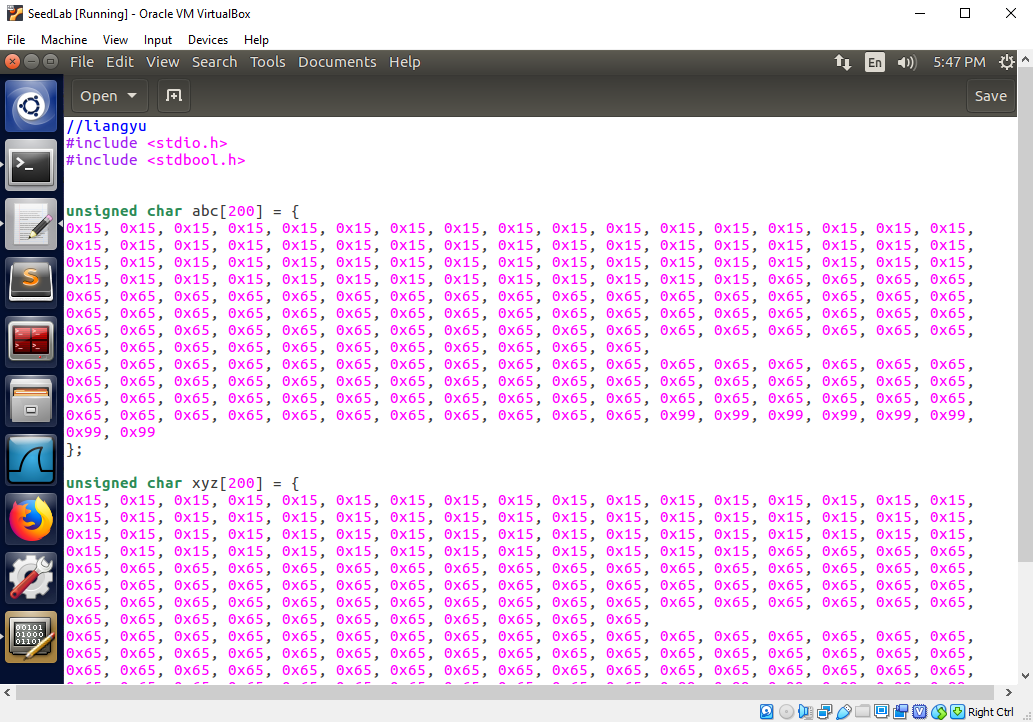


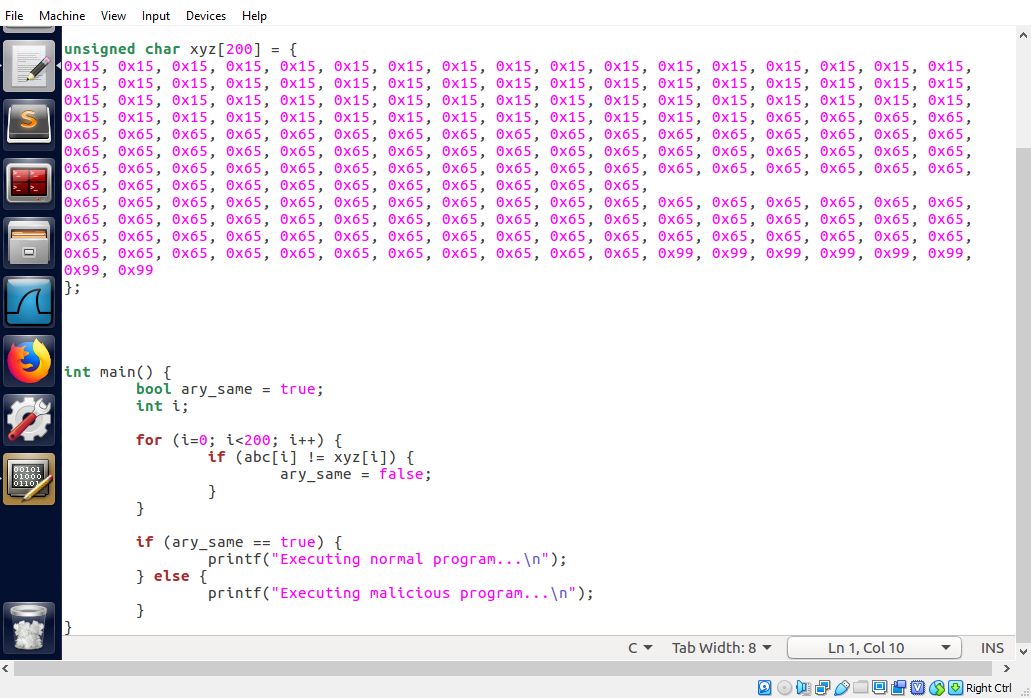


We then concatenate out1.bin and out2.bin to the suffix file, producing two executable files that have the same md5 hashes but produce different outputs.

**Task 4: Making the Two Programs Behave Differently**

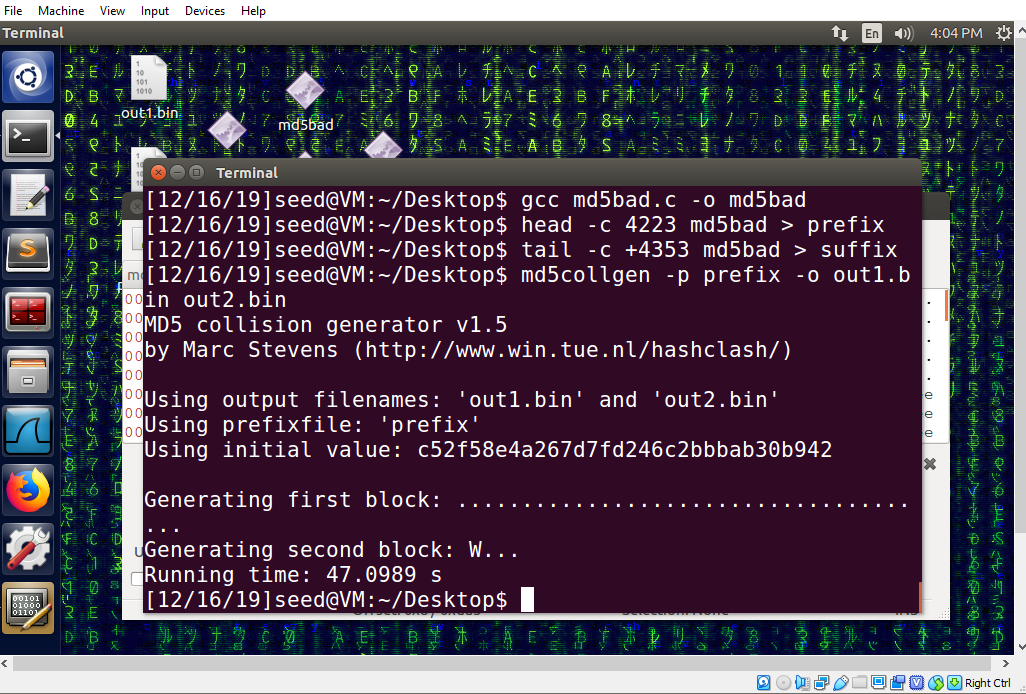
We create a C program called md5bad as follows:





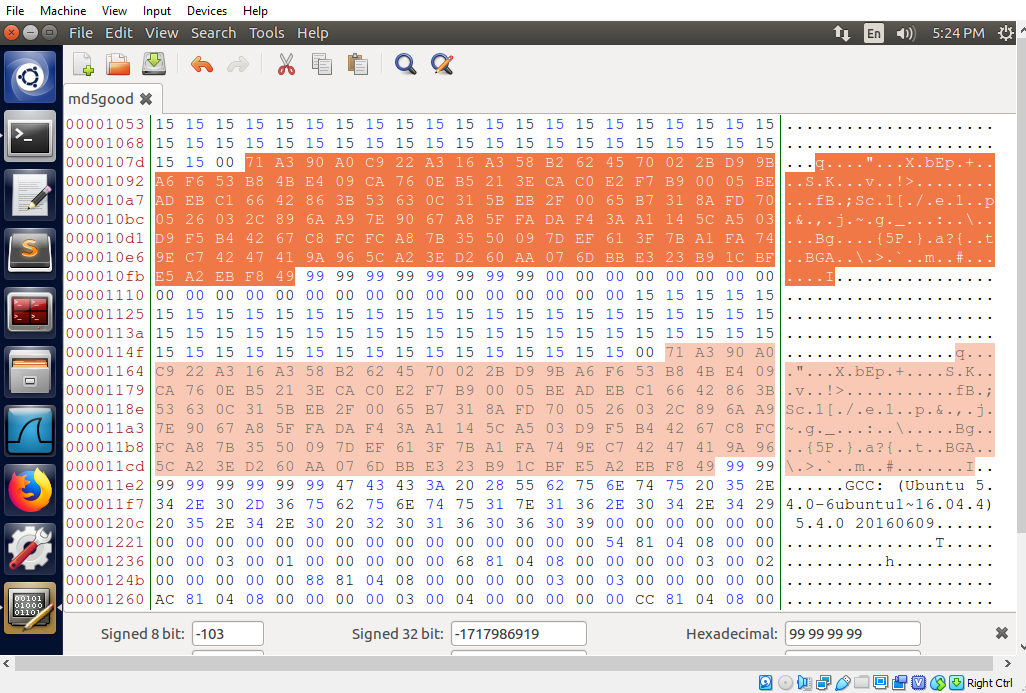
Using Bless, we can see that the two arrays’ contents are stored in the compiled executable file from offset 0x1040 to 0x1107 and from 0x1120 to 0x11e7. The prefix section of the first array is stored at offset 0x1040 to 0x107f (4223 in decimal). The suffix section of the first array is stored at offset 0x1100 (4352 in decimal).

We use the head command to slice and copy the first part of the compiled executable file into prefix file. Then use the tail command to slice and copy the last part of the file into suffix file. Then run md5collgen on prefix file to generate out1.bin and out2.bin.

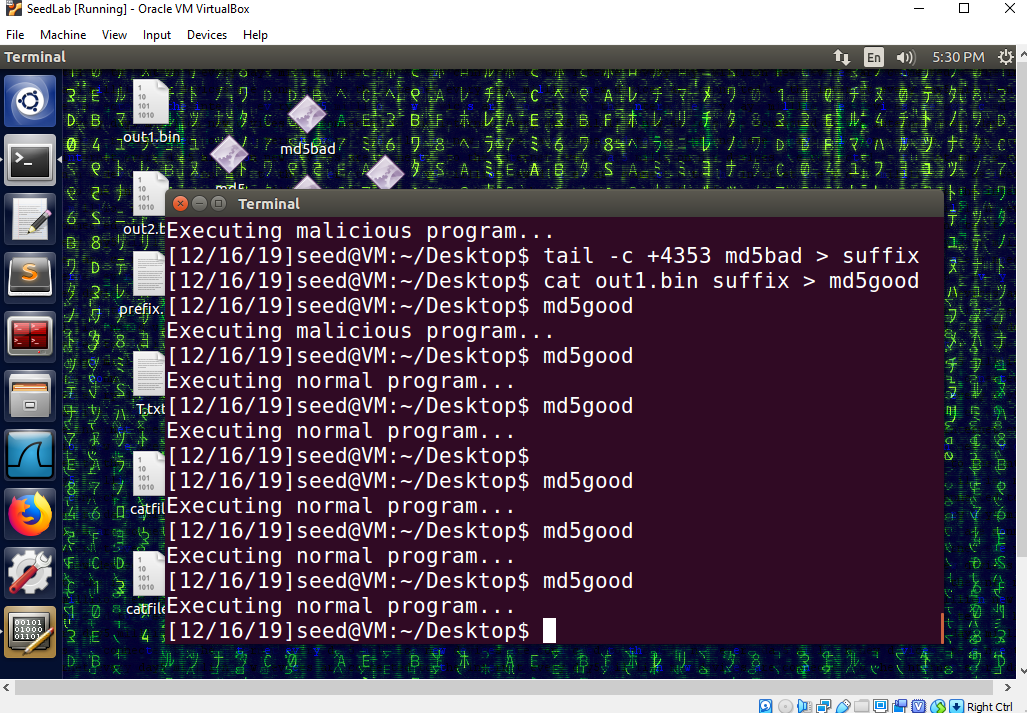


We use command “cat out1.bin suffix > md5good” to generate a file called md5good.

We copy and paste the section generated by md5collgen to the content of the second array.

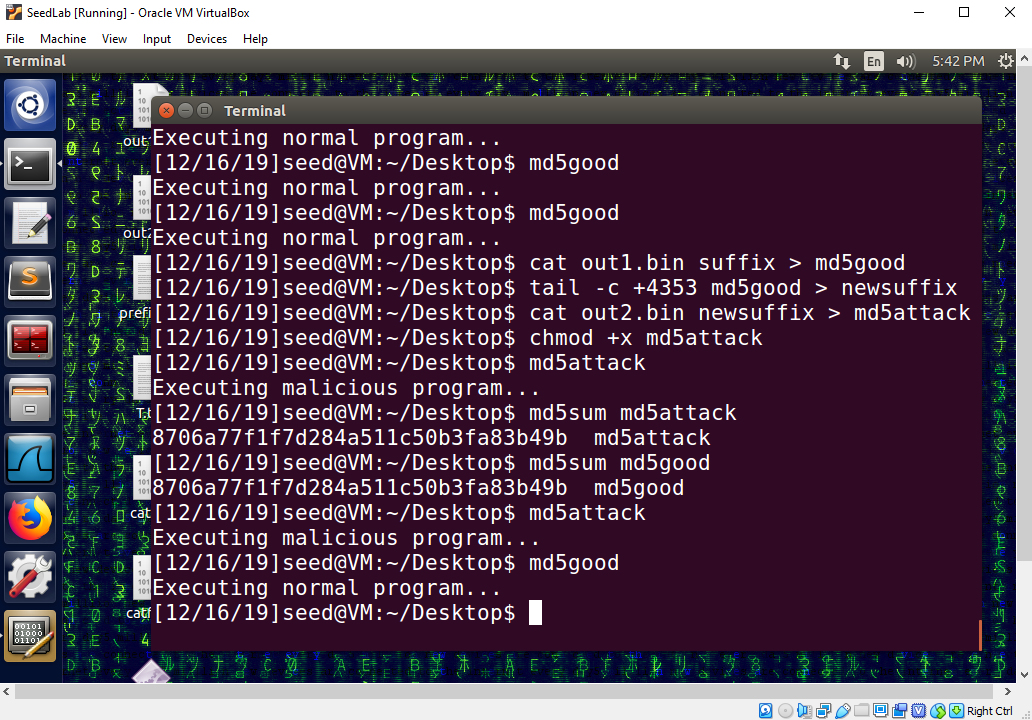


If we now run md5good program, it executes benign code.



Now we use the tail command to slice the md5good program to create a new suffix.

Then use the cat command to concatenate the new suffix to the out2.bin.



Now we have two programs that behave differently but have the exact same md5 hash.