**IS LAB ASSIGNMENT#01**

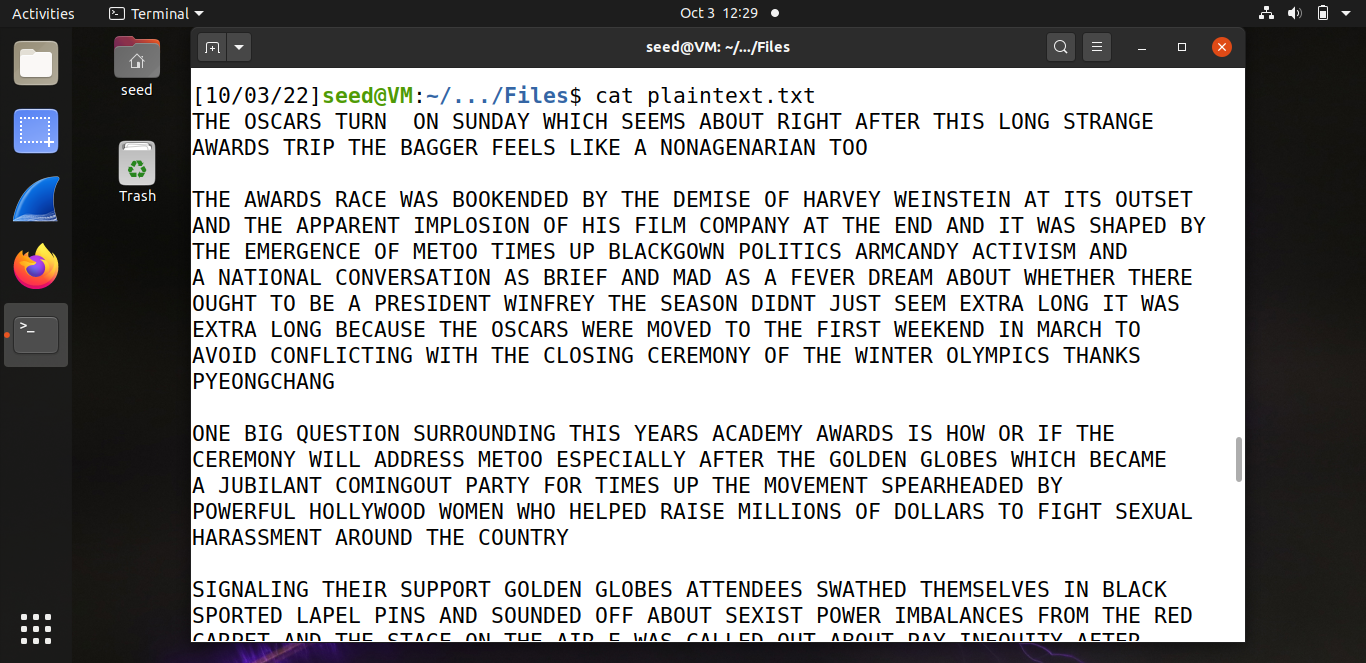
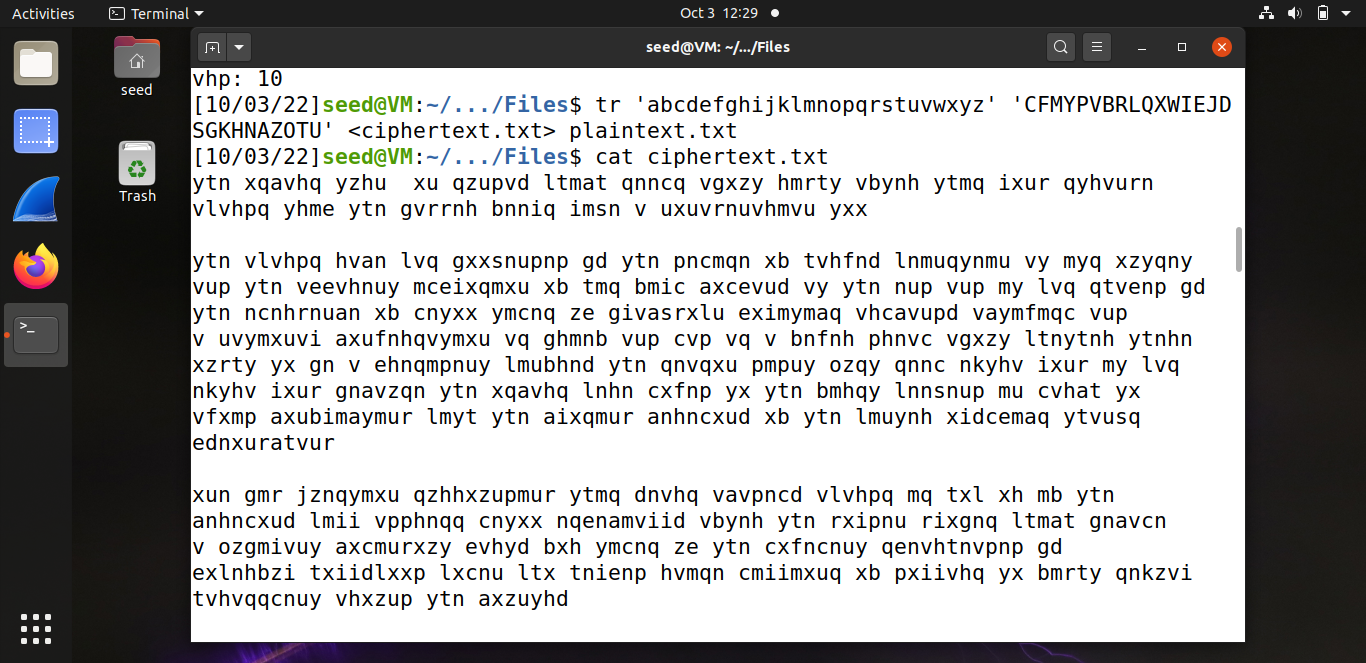
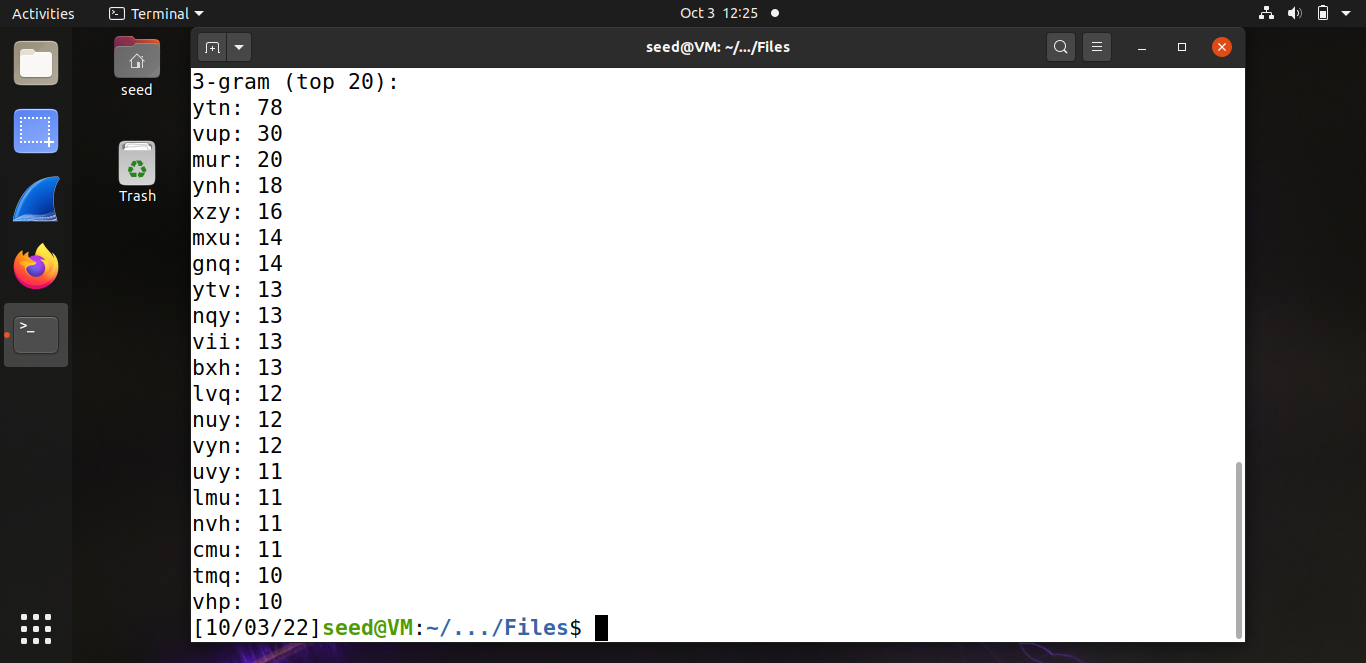
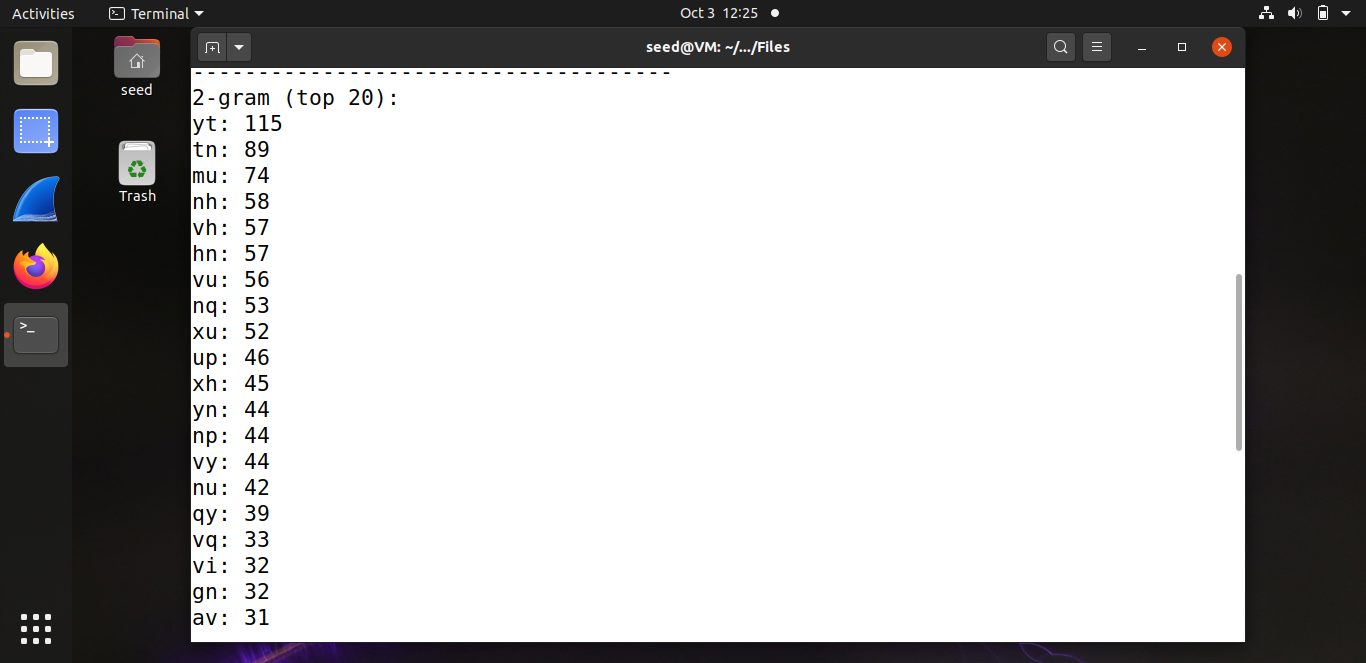
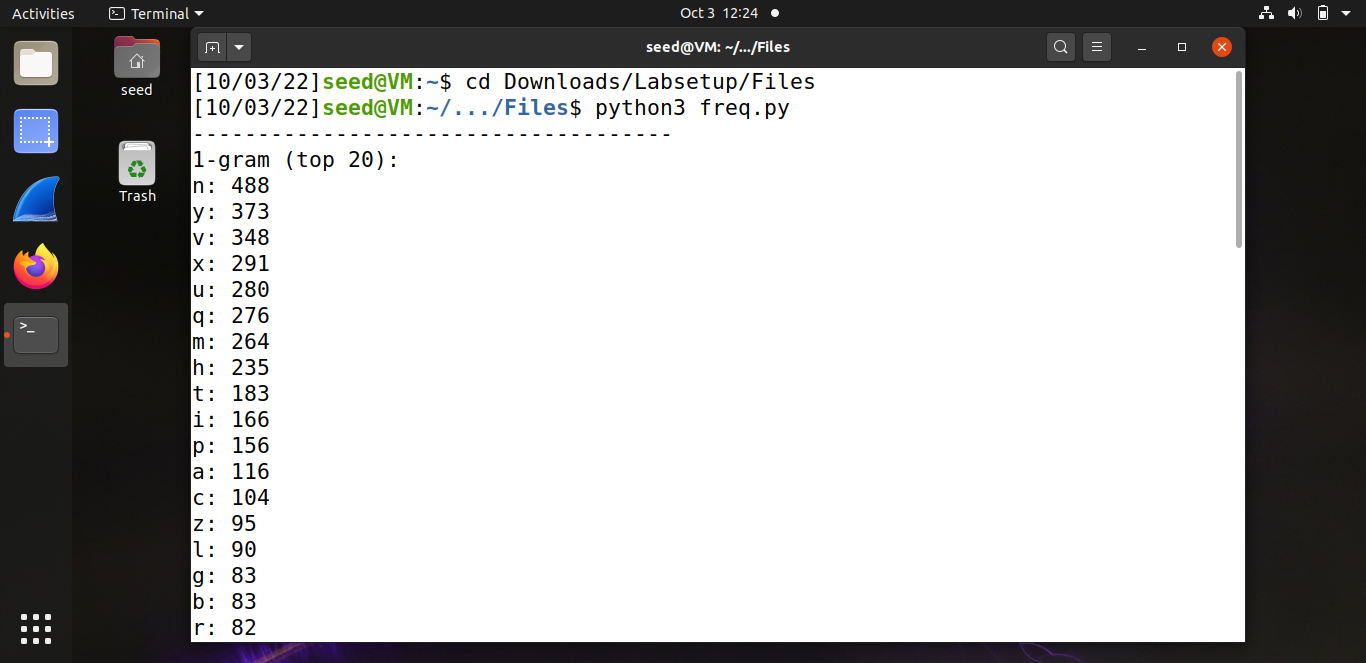
**Name: Saman Khan**

**ID: 19K-0354**

**Section: H**

**TASK#01:**

First, the python file named as “freq.py” will be executed to find all the n-grams. Using these n-grams I have performed frequency analysis which led to the discovery of key. Then I used to key to replace each character with its corresponding key. This decrypted text is saved in the file “plaintext.txt” and displayed using cat command.

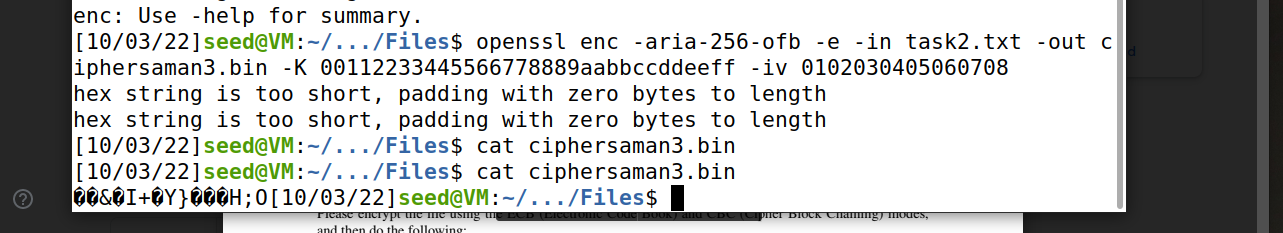
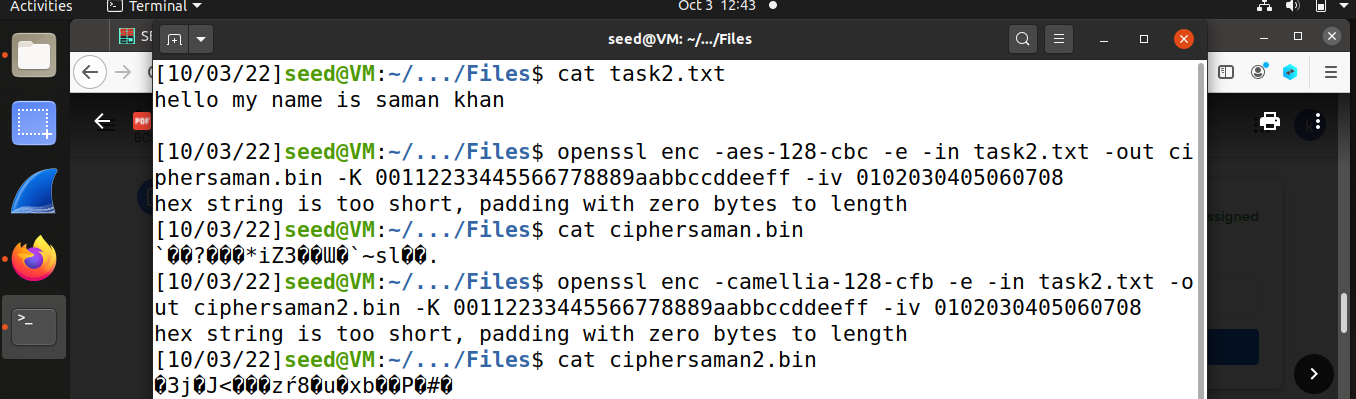
****

**TASK#02:**

In task 2 we were supposed to implement encryption using three different algorithms. I have used the following algorithms for encryption:

1. AES-128-CBC
2. Camellia-128-CFB
3. Aria-256-OFB

The key and IV is same as the one given in lab document.

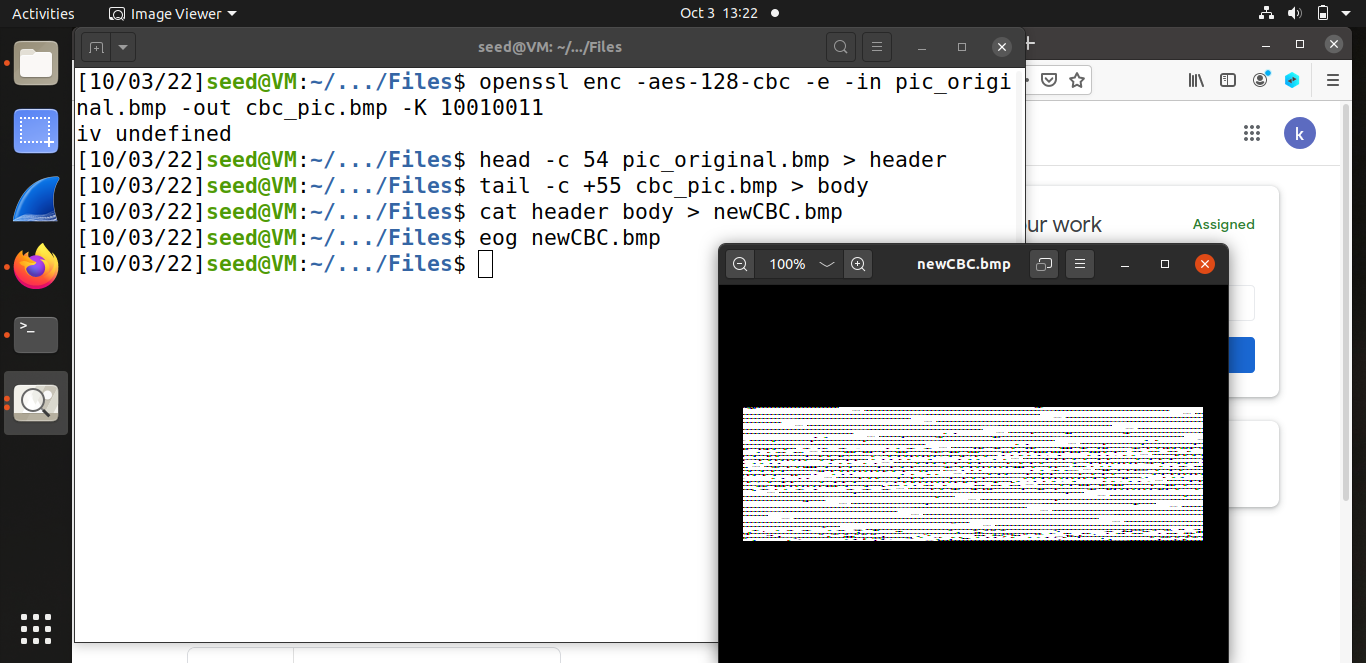
****

It can be seen that the cipher text produced is different for each algorithm even though same plain text, key, and IV are used.

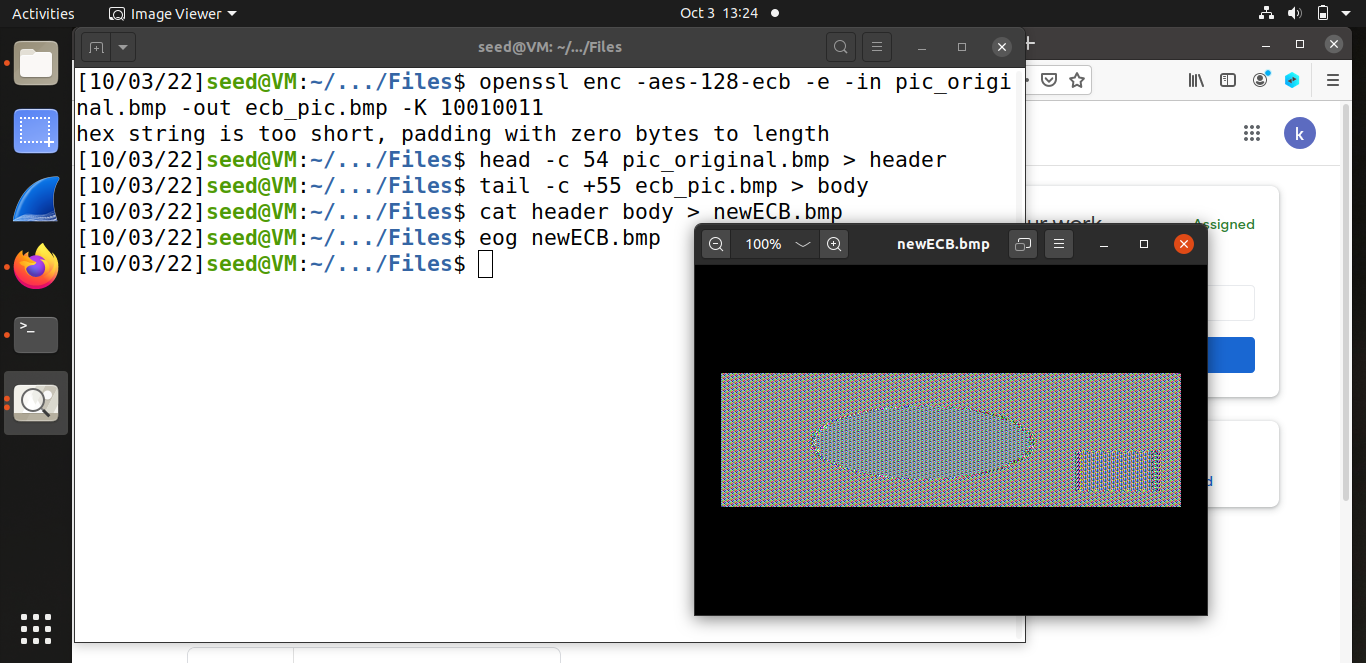
**TASK#03:**

In task 3 we have encrypted the image given in the folder, so people without the encryption keys cannot know what is in the picture. We have used ECB (Electronic Code Book) and CBC (Cipher Block Chaining) for encryption. For the .bmp file, the first 54 bytes contain the header information about the picture, we have to set it correctly, so the encrypted file can be treated as a legitimate .bmp file. We will replace the header of the encrypted picture with that of the original picture. We have used commands (as seen in screenshots) to get the header from p1.bmp, the data from p2.bmp (from offset 55 to the end of the file), and then combine the header and data together into a new file. ECB is less secure, which does not have the randomness. As an output a cryptanalyst can break the cipher or the key and steals the data. Whereas CBC is more secure.

* **CBC:**

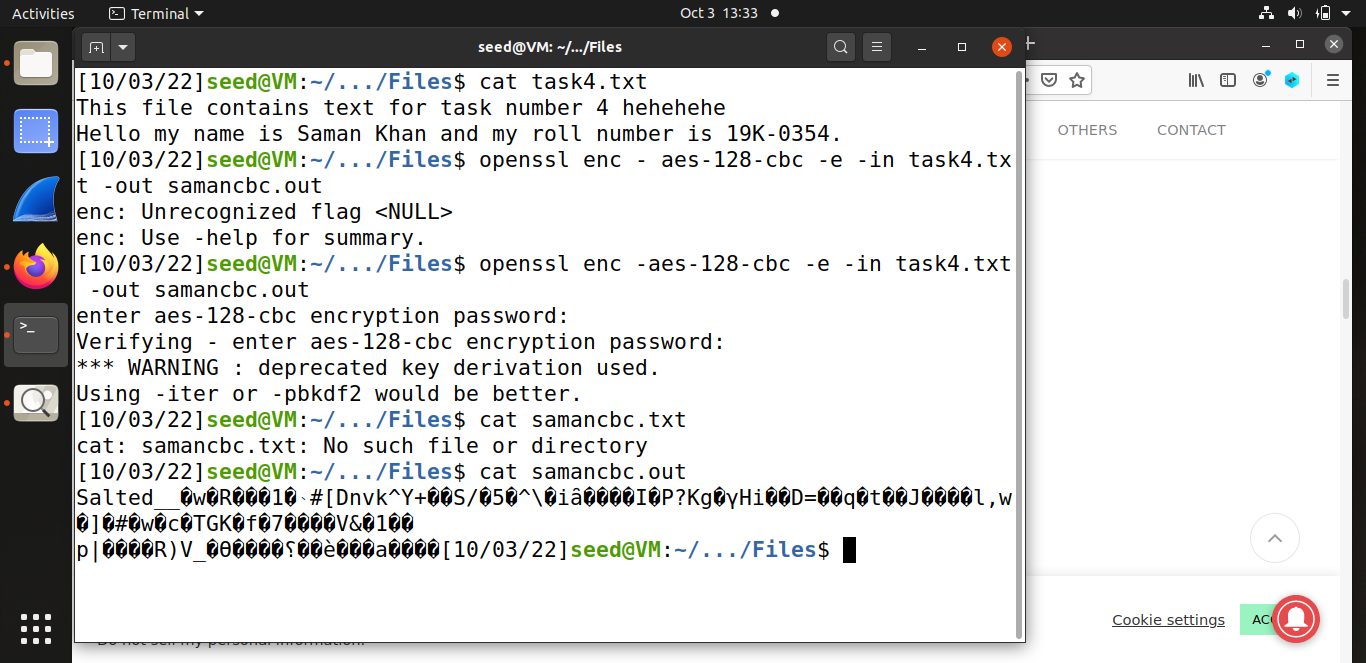
****

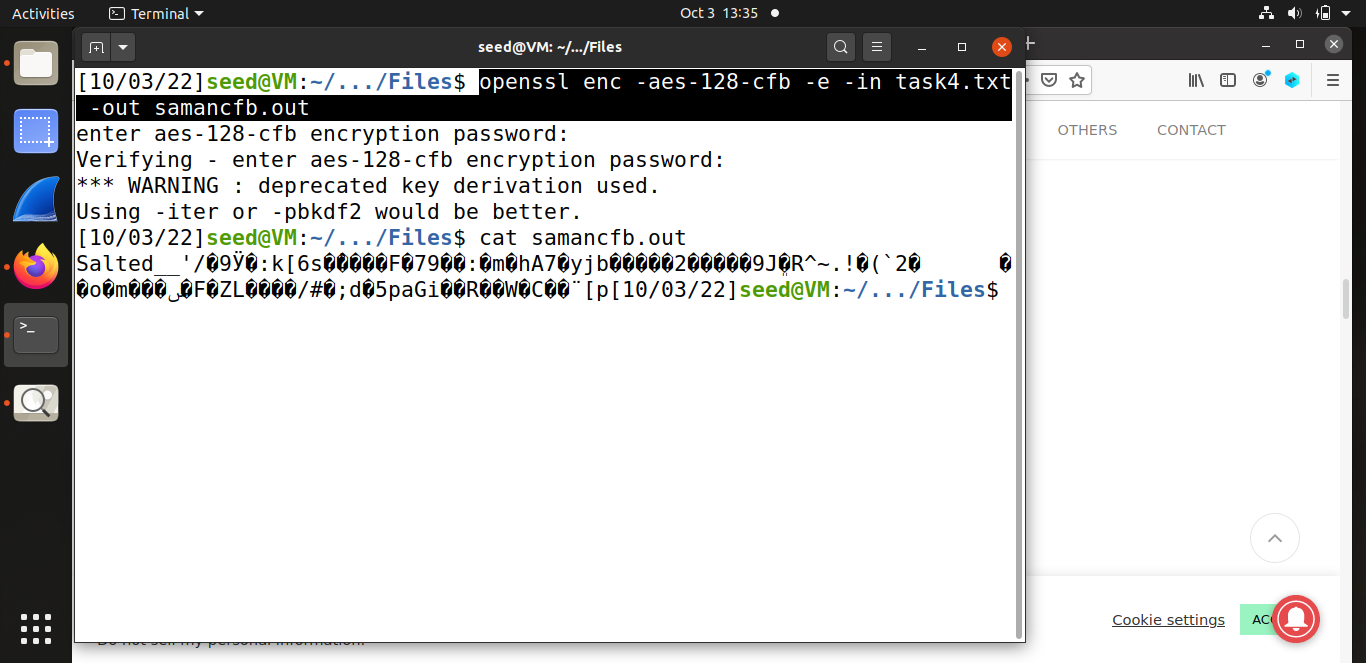
* **ECB:**

****

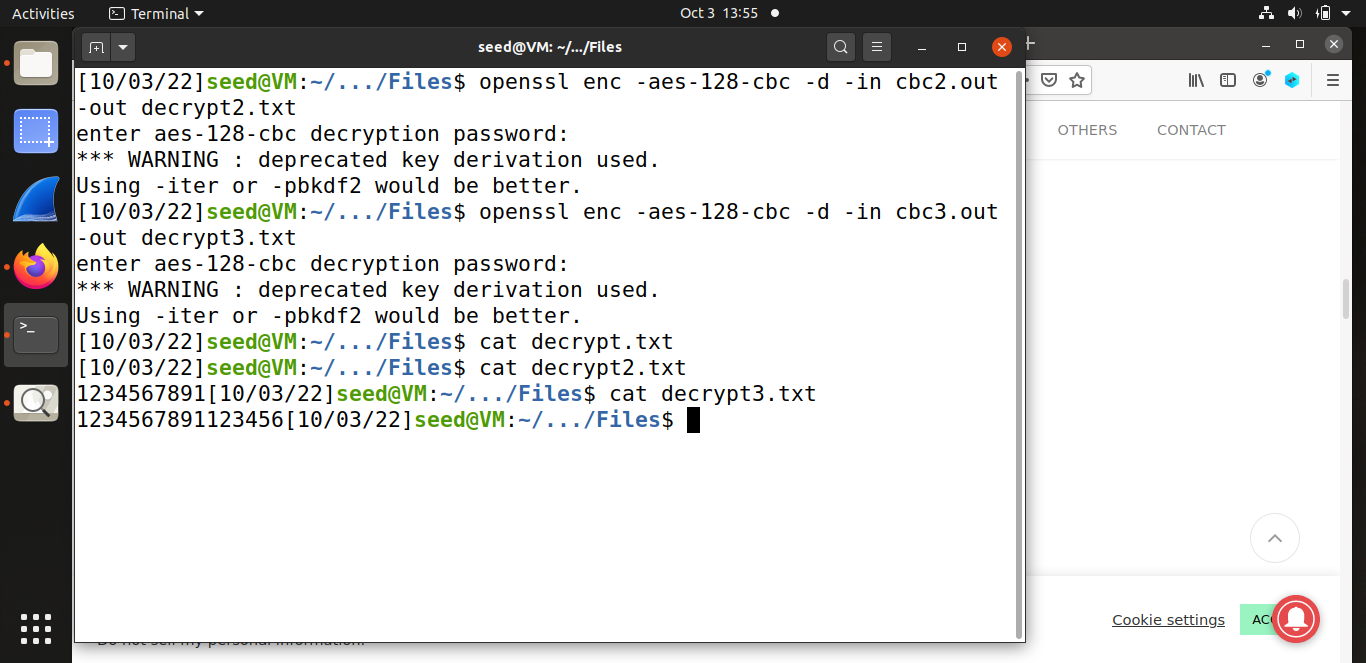
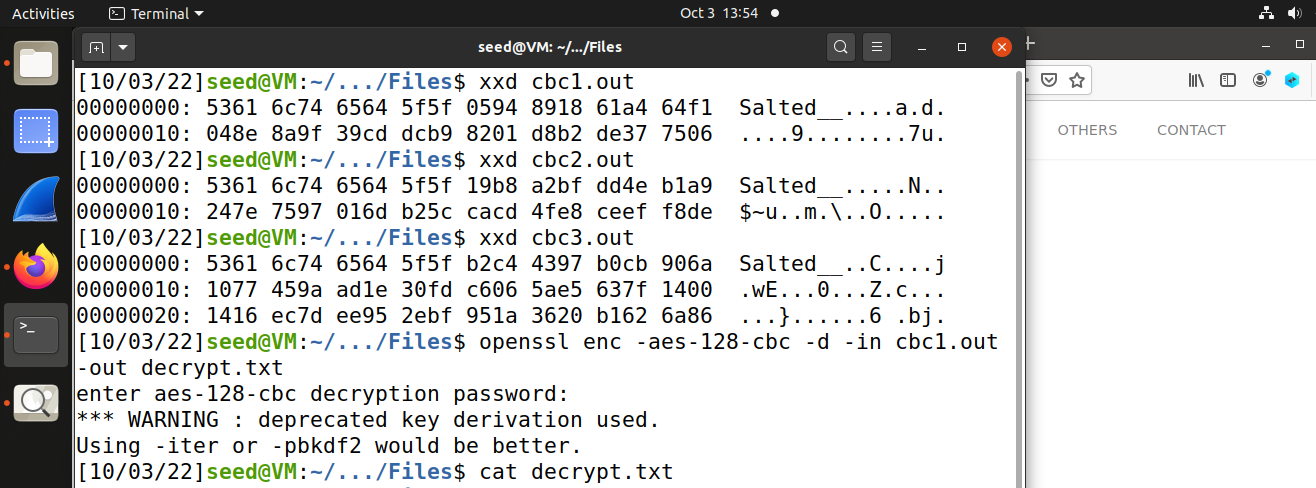
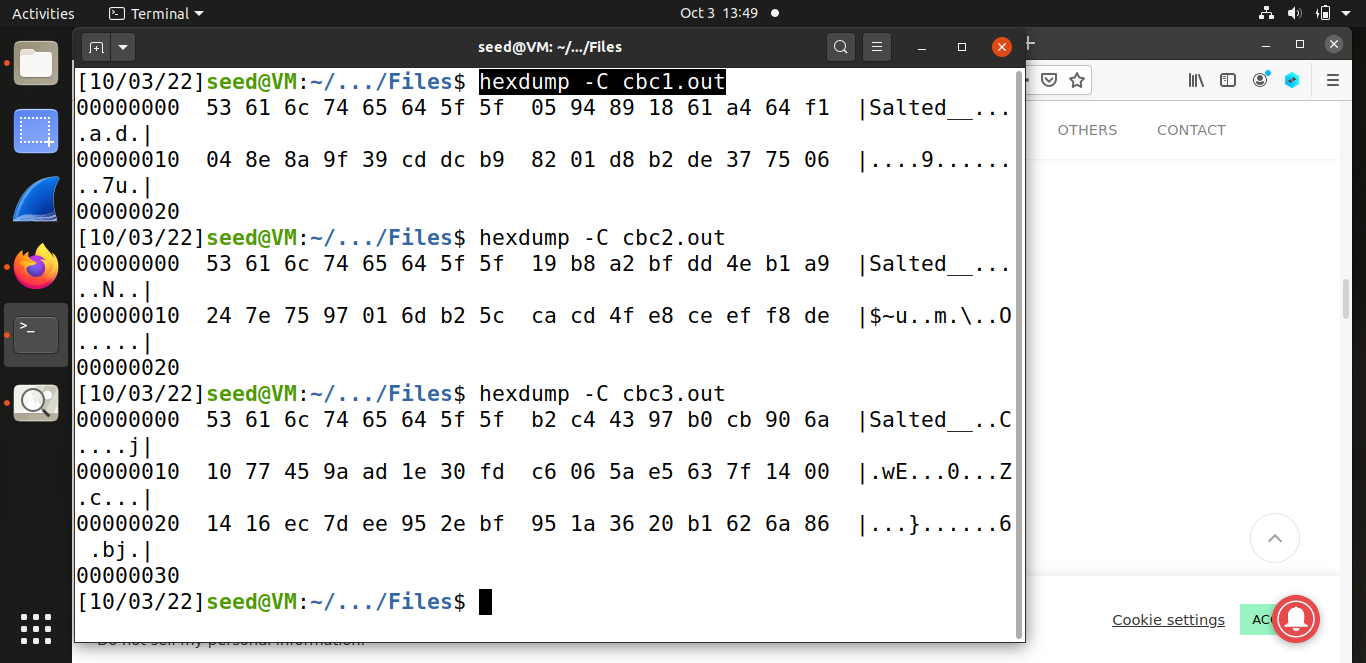
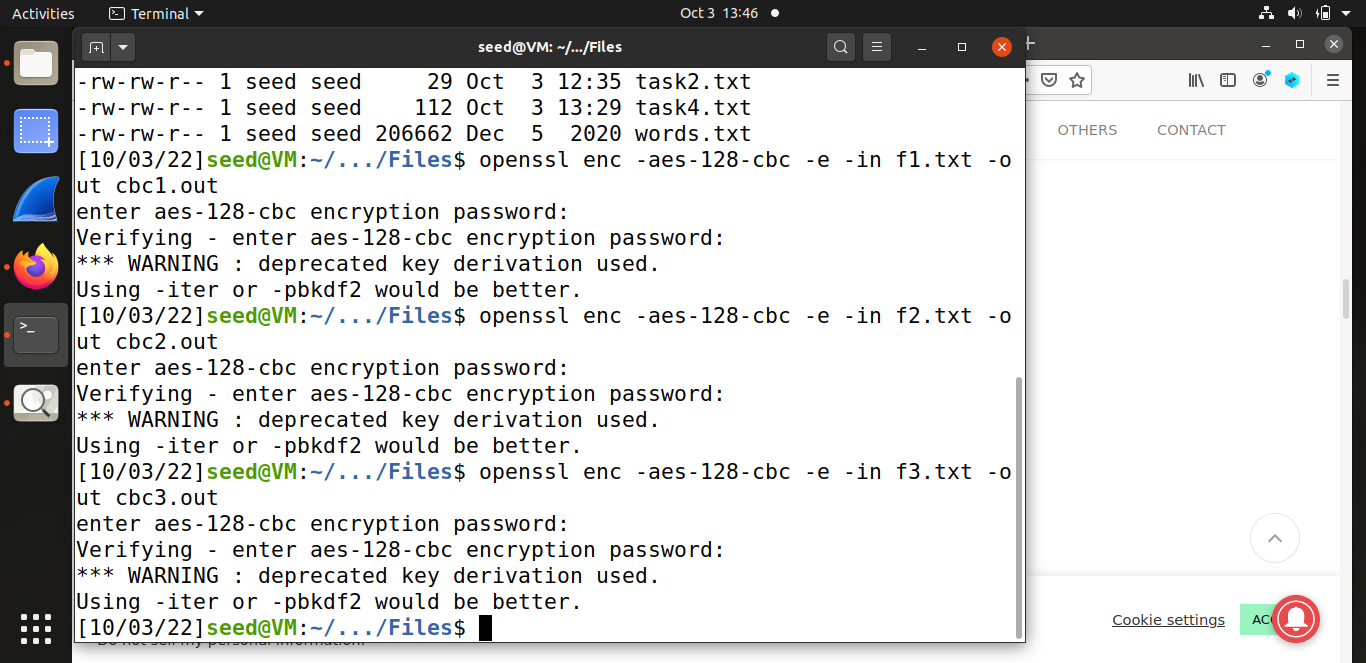
**TASK#04:**

Only the ECB and CBC had padding whereas others do not because the OFB and CFB don’t require padding as it's not applicable because the cipher text is always the same length. Then three files f1.txt, f2.txt, and f3.txt of size 5 bytes, 10 bytes, and 15 bytes as instructed respectively. After performing encryption on them it is observed that padding is only required for ecb and cbc modes because they are block ciphers whereas ofb and cfb donot require padding because they are stream ciphers.

****

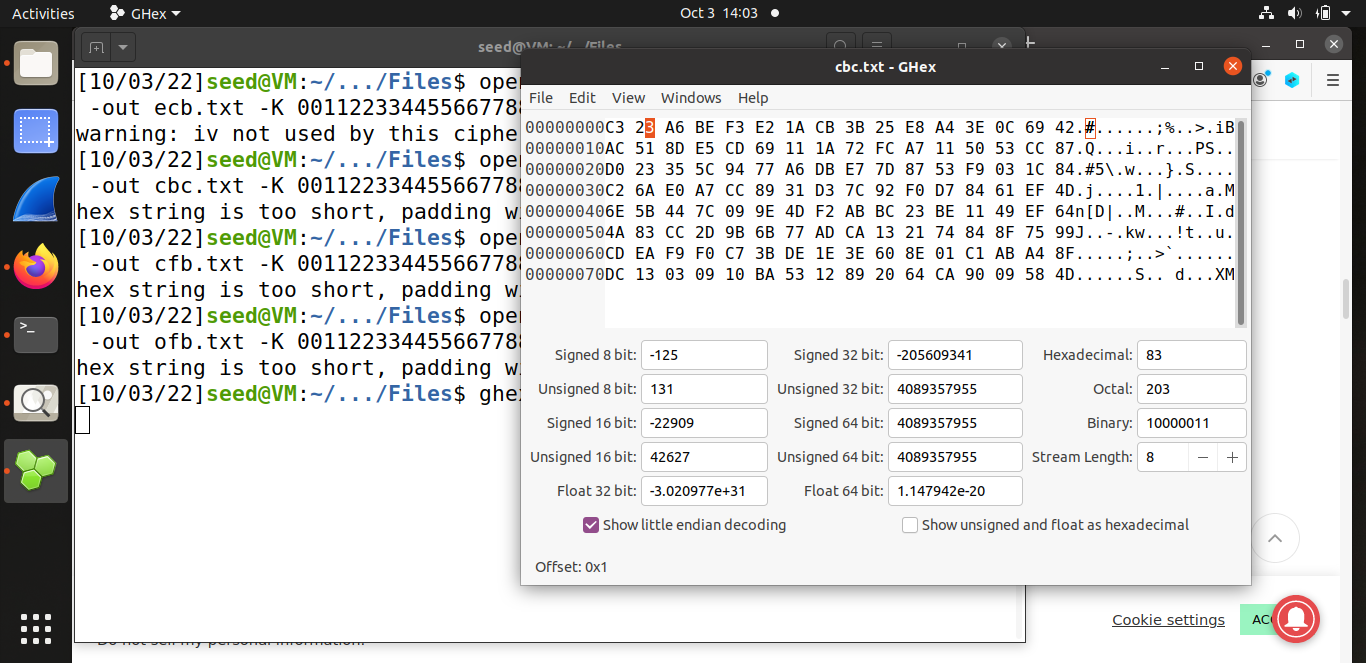
****

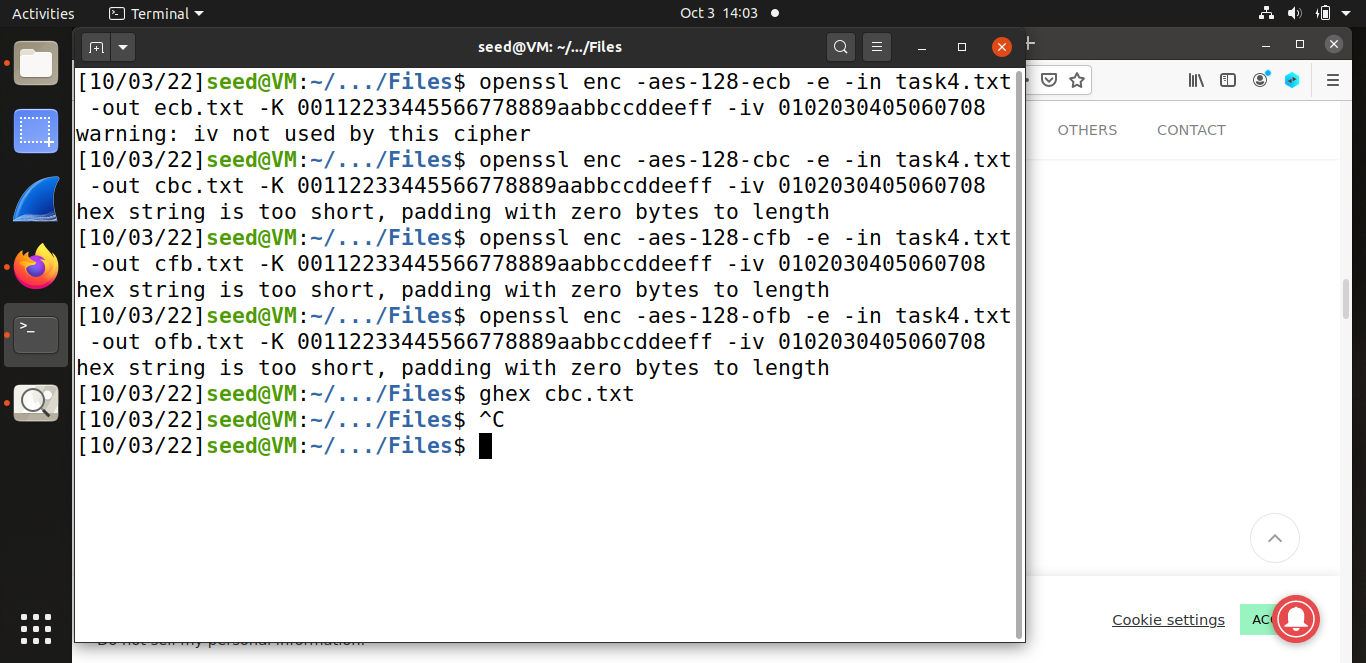
****

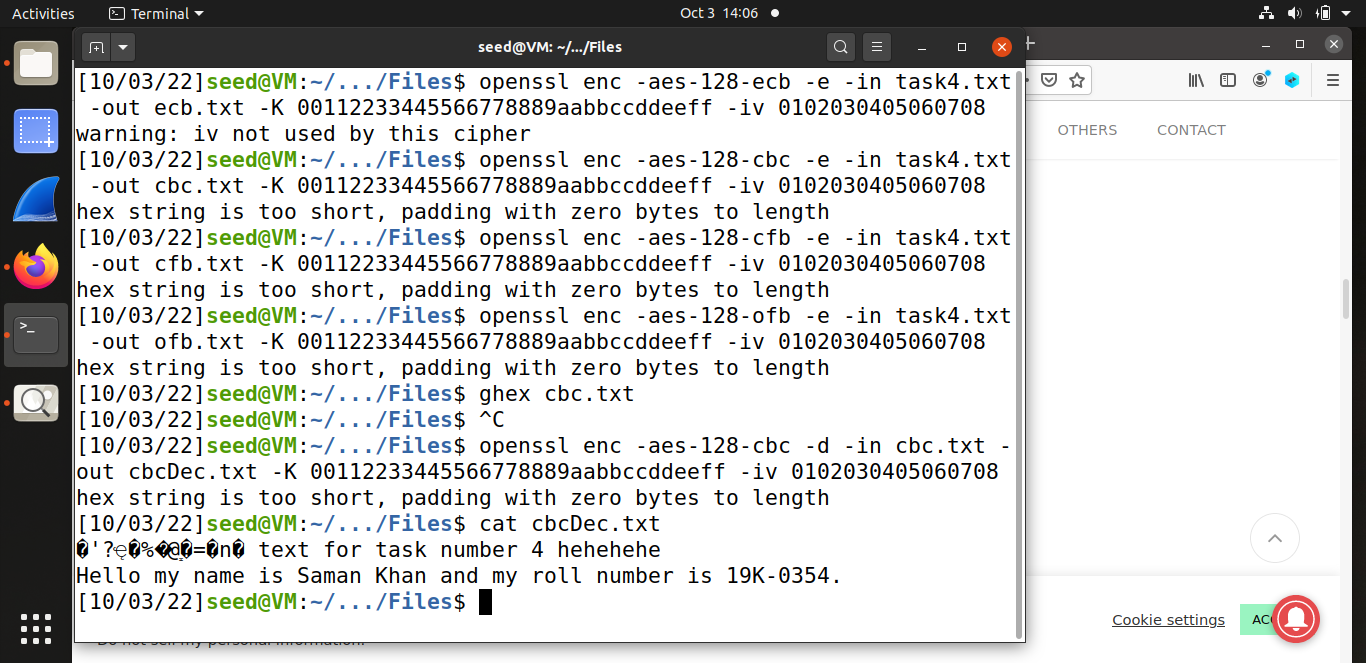
****

**TASK#05:**

Production of corrupted decrypted files. Here the file was first encrypted using aes-128-ecb and then a bit is altered i.e, 55th.

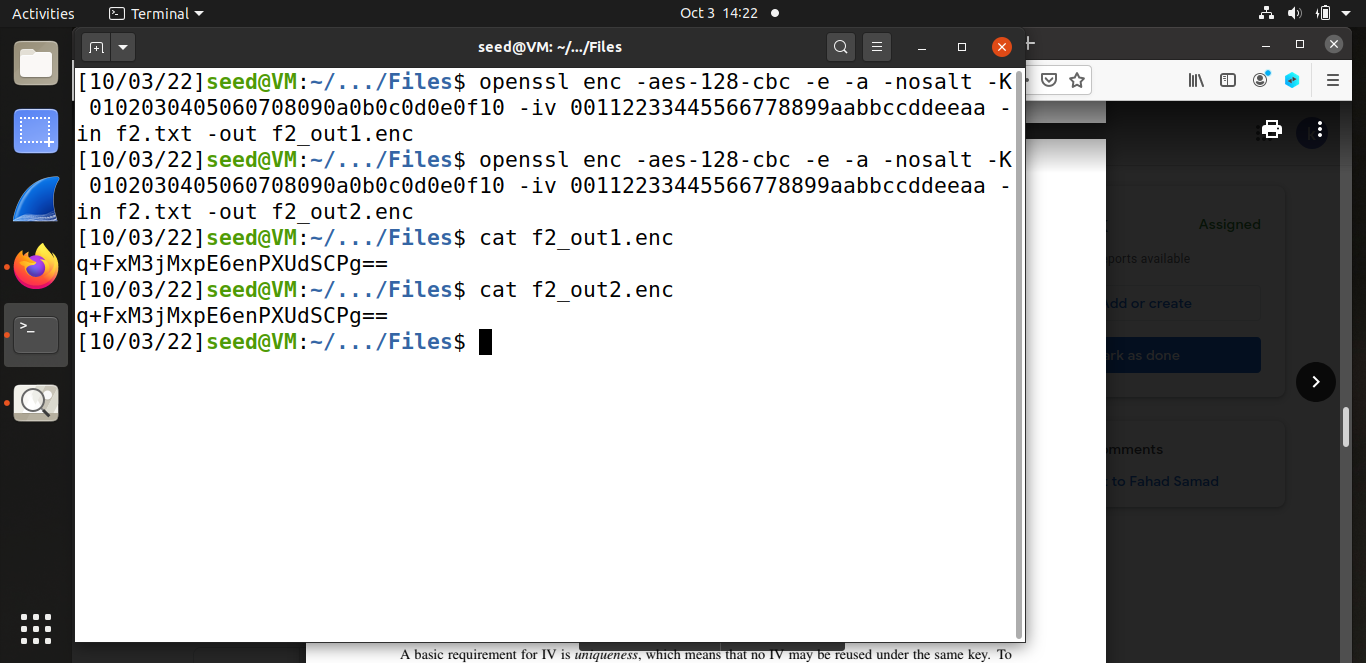
****

****

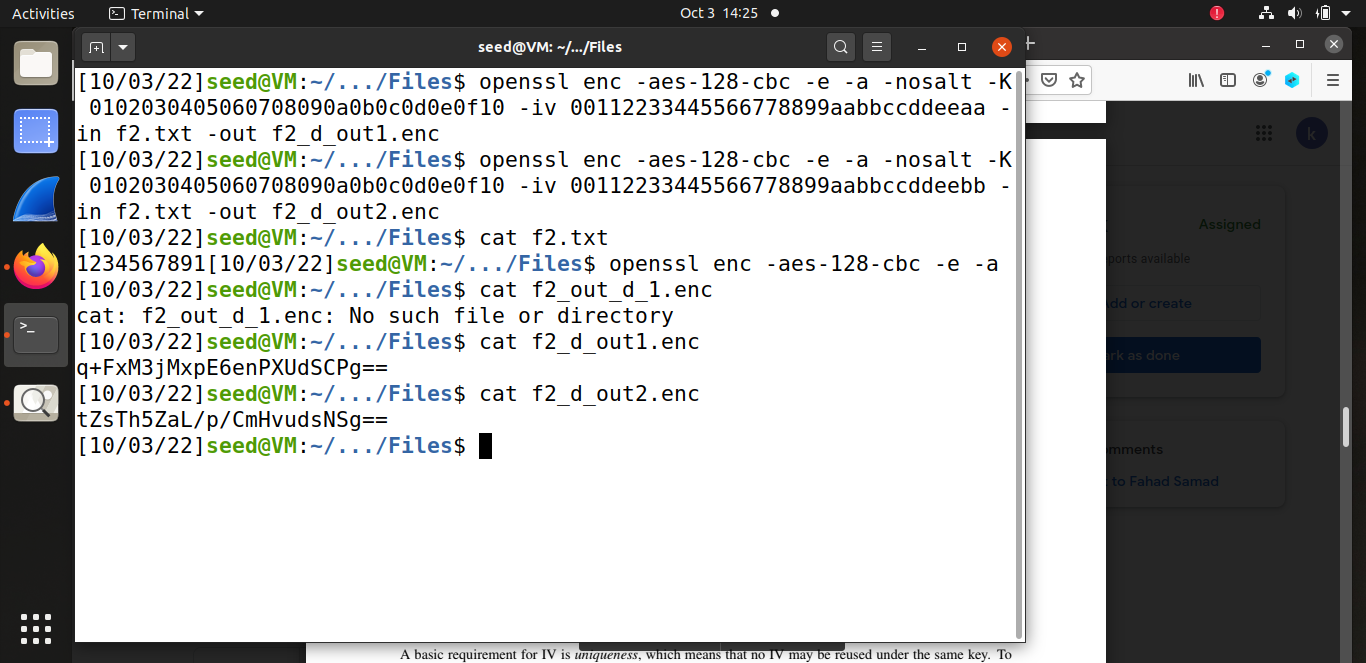
****

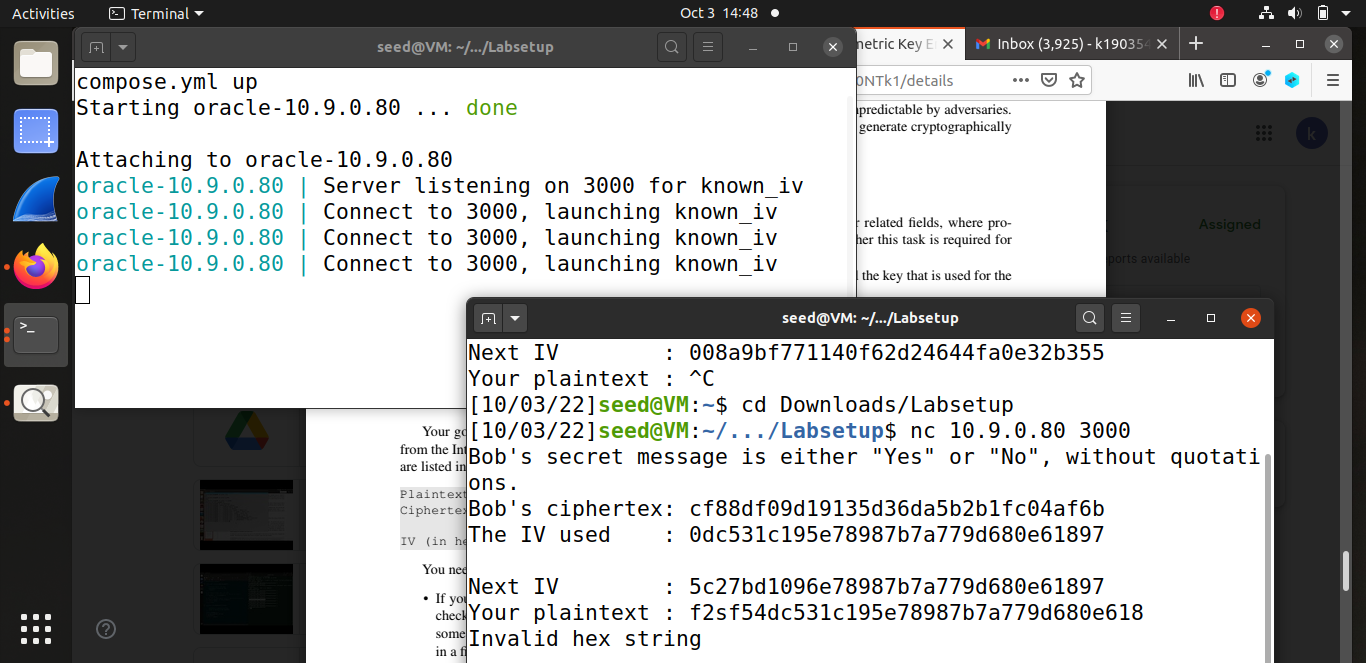
**TASK#06:**

It can be seen in the image that when same IV is used (even when key is different) it has generated the same cipher text.

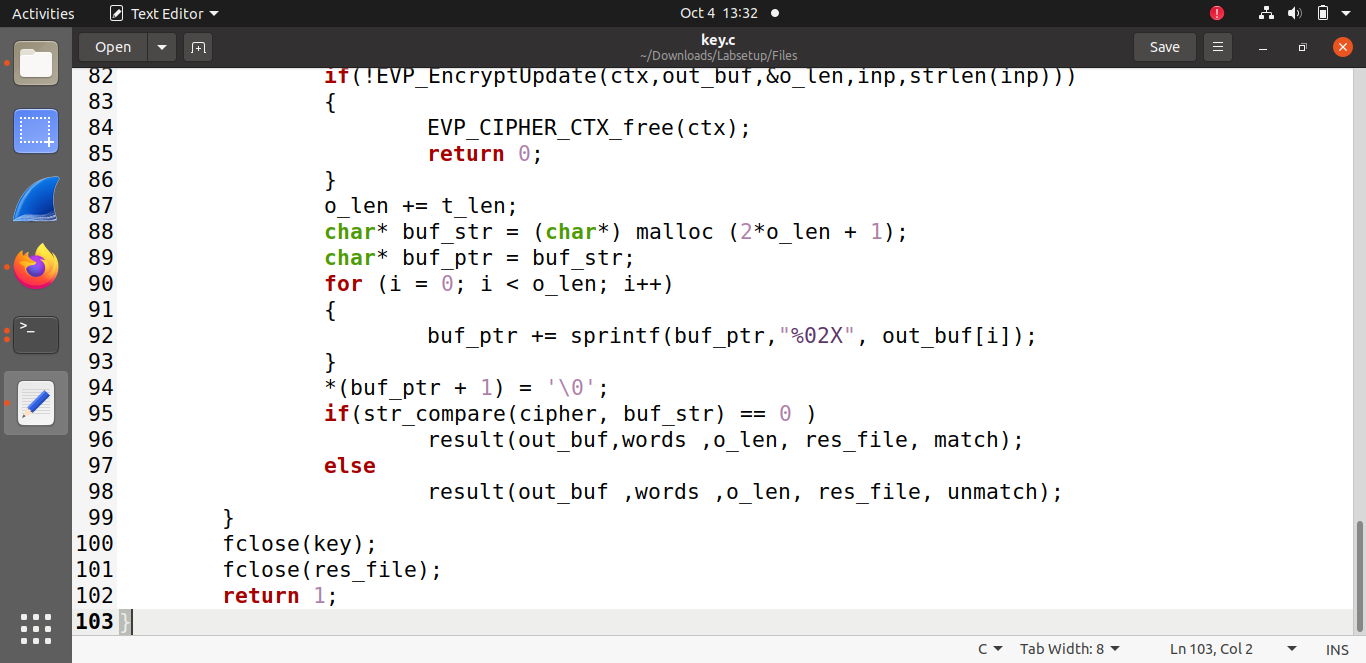
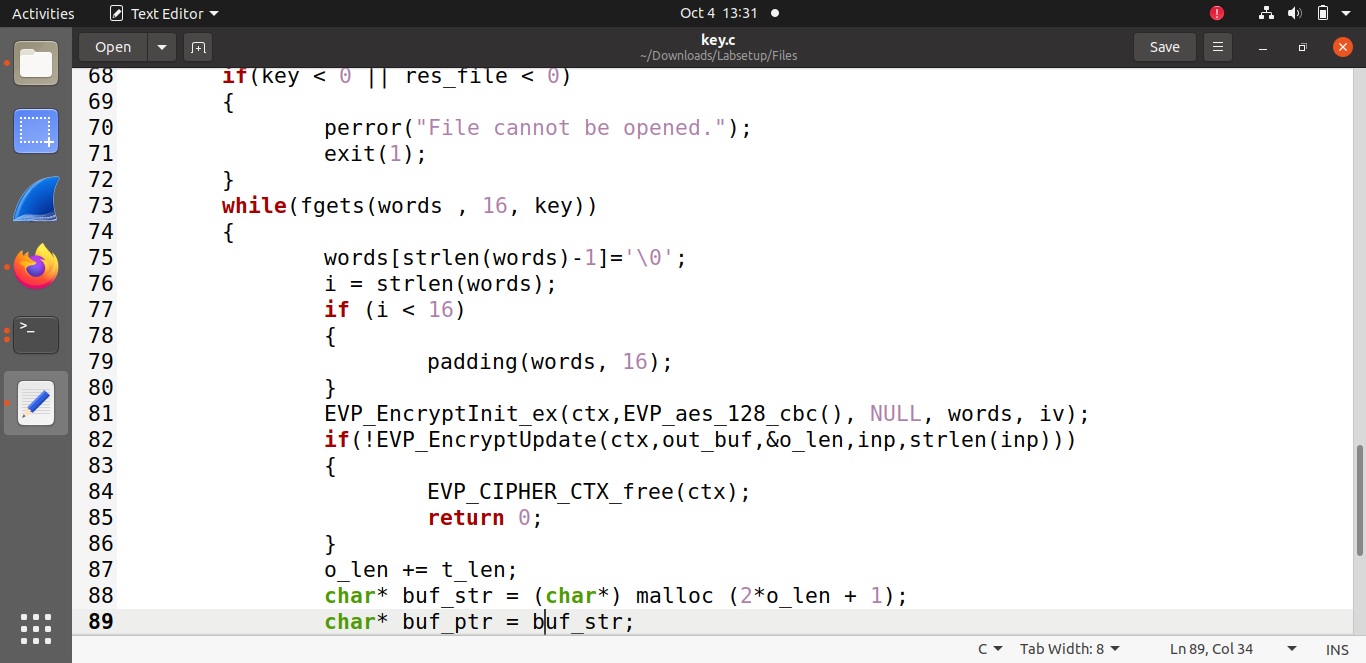
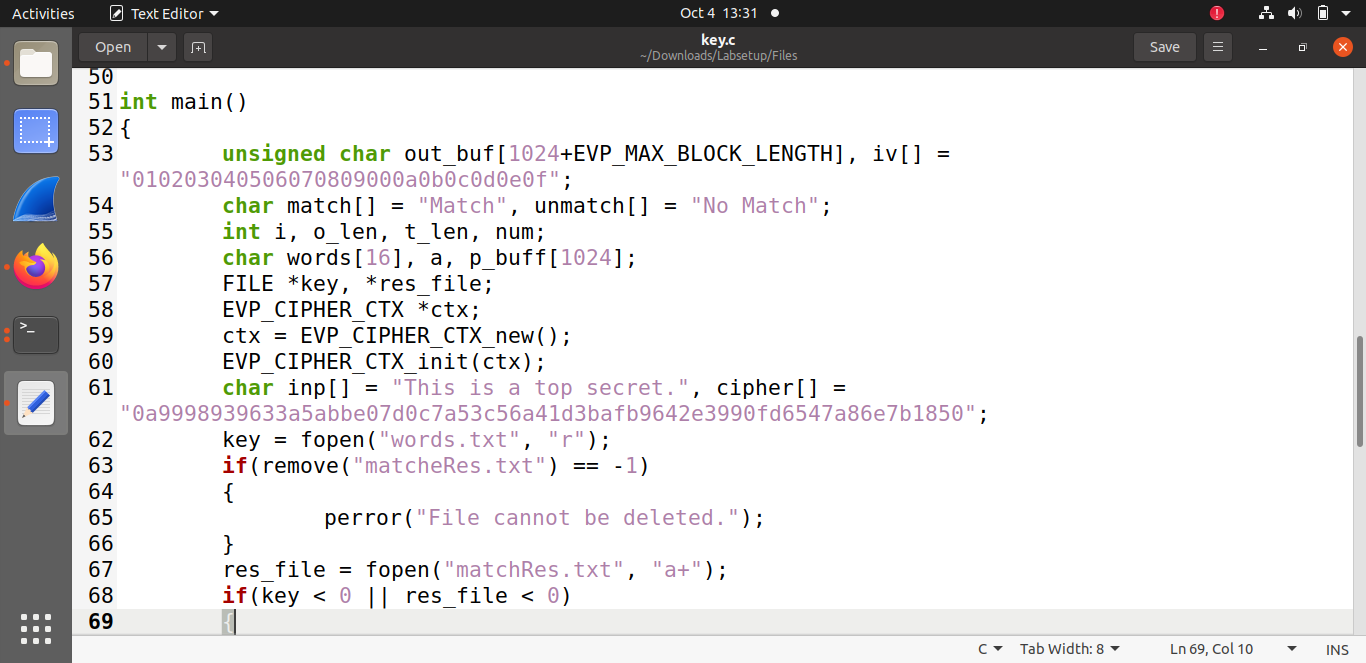
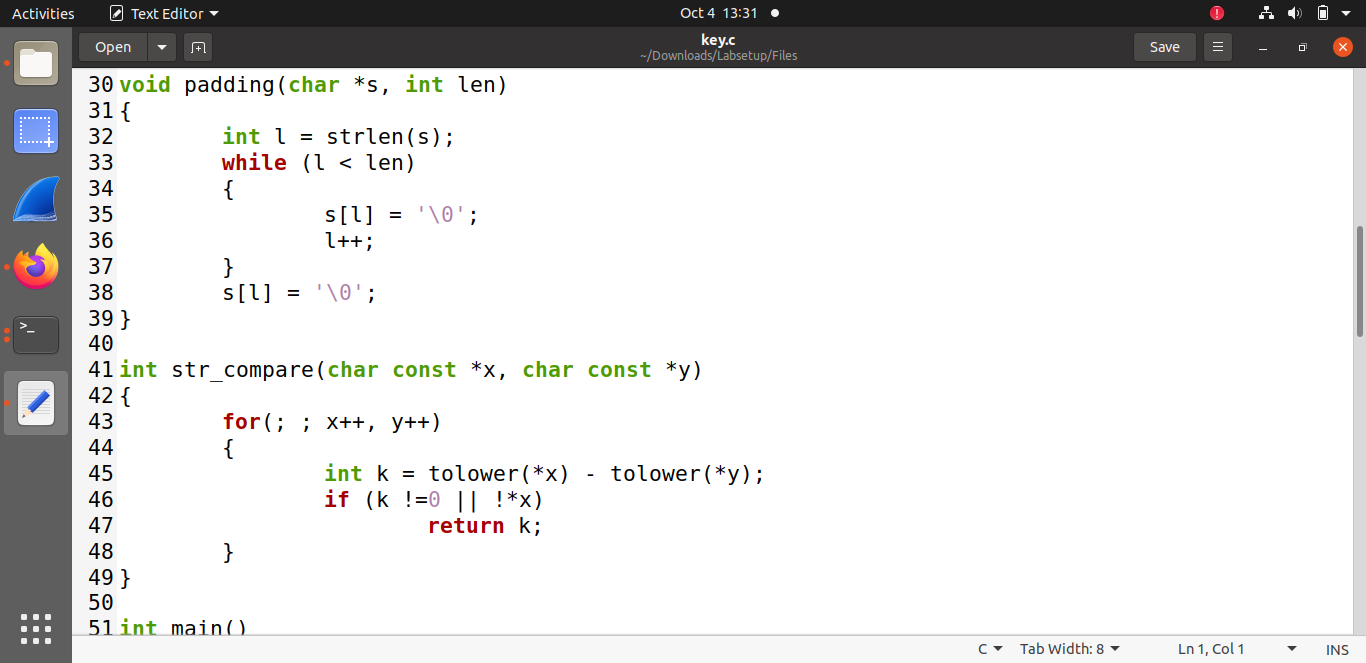
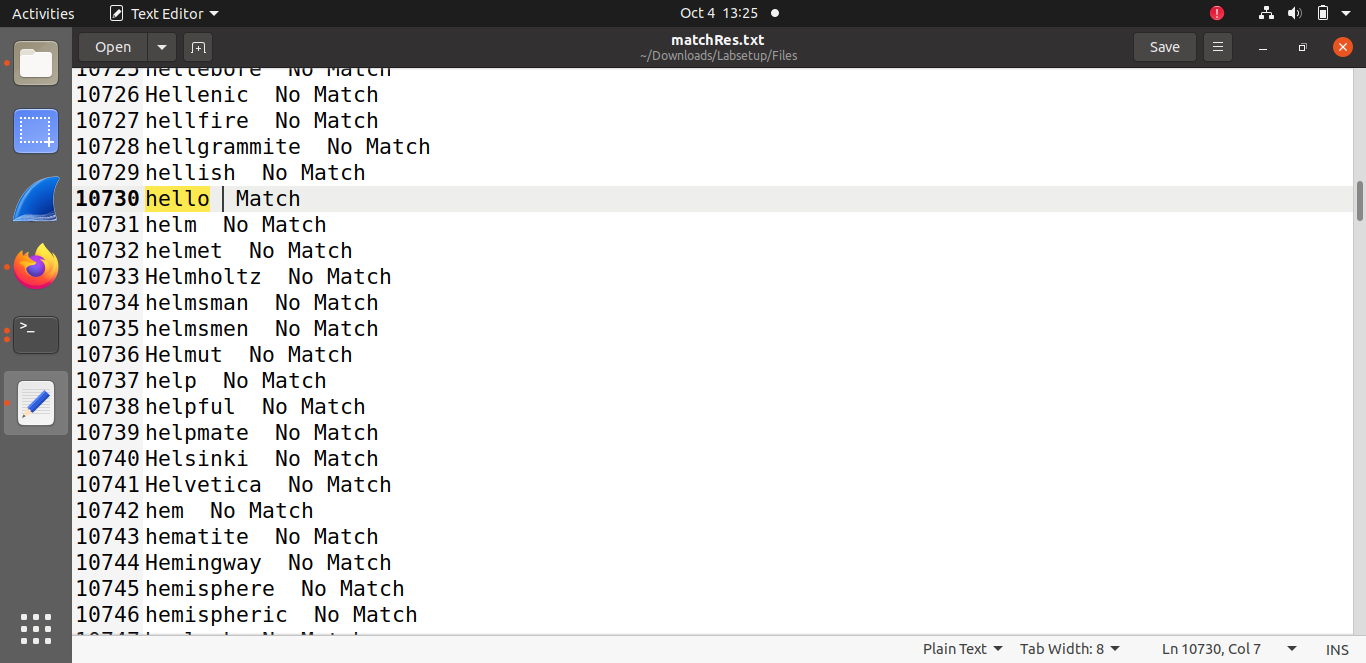
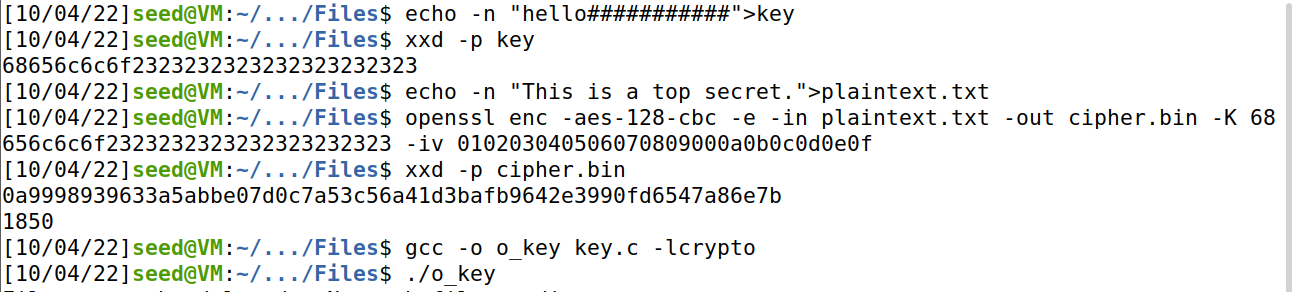
****

Here effect of different IV is observed which is presented in the form of different cipher text as opposed to the scenario above.

****

****

**TASK#07:**

****