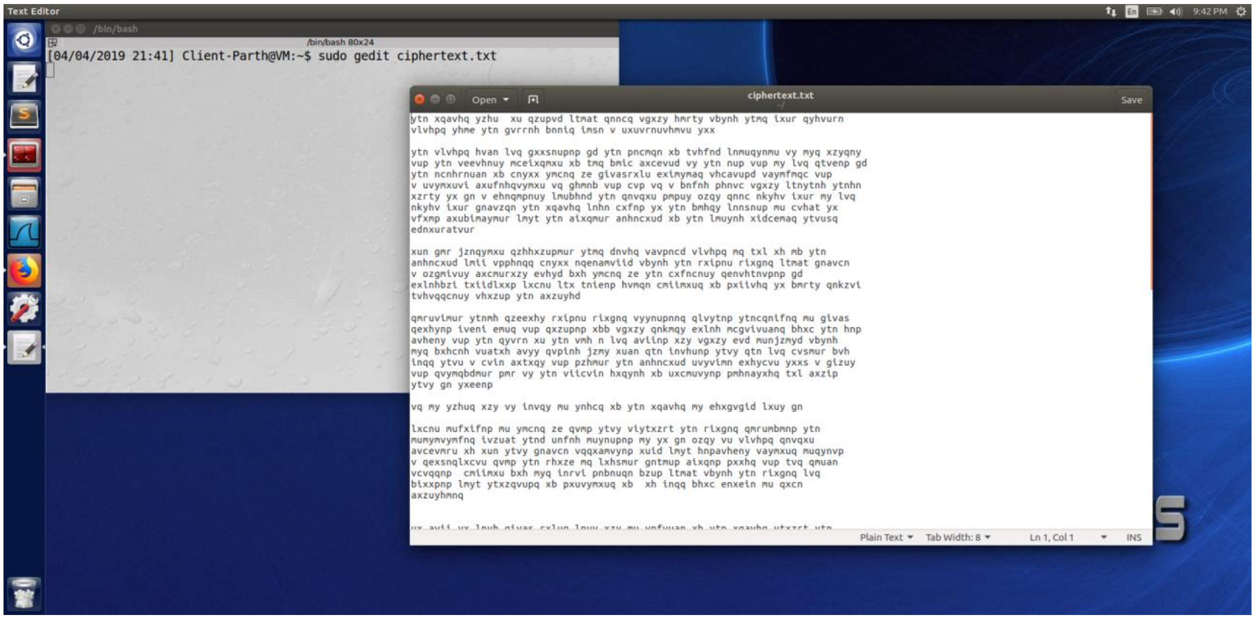
Secret-Key Encryption Lab

Task 1: Frequency Analysis Against Monoalphabetic Substitution Cipher

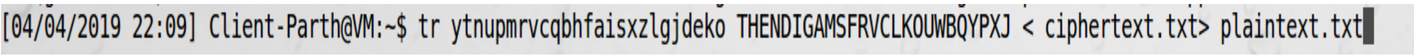


```
[04/04/2019 21:41] Client-Parth@VM:~$ sudo gedit ciphertext.txt
```

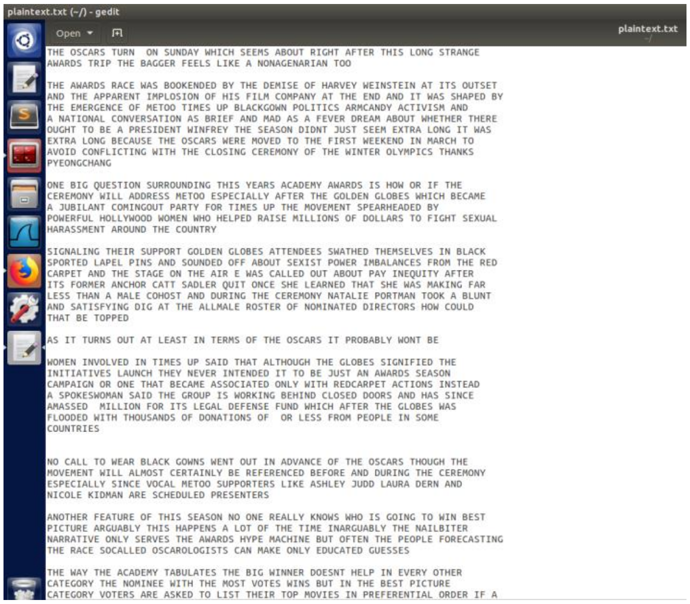
```
ytn xqavhq yzhu  xu qzupvd ltmat qnncq vgxzy hmrty vbynh ytmq ixur qyhvurn
vlvhpq yhme ytn gvrrnh bnniq imsn v uxuvrnuvhmvu yxx

ytn vlvhpq hvan lvq gxxsnupnp gd ytn pncmqn xb tvhfnd lnmuqynmu vy myq xzyqny
vup ytn veevhnuy mceixqmxu xb tmq bmic axcevud vy ytn nup vup my lvq qtvenp gd
ytn ncnhrnuan xb cnyxx ymcnq ze givasrxlu eximymaq vhcavupd vaymfmqc vup
v uvymxuvi axufnhqvymxu vq ghmnb vup cvp vq v bnfnh phnvc vgxzy ltnytnh ytnhn
xzrty yx gn v ehnqmpnuy lmubhnd ytn qnvqxu pmpuy ozqy qnnc nkyhv ixur my lvq
nkyhv ixur gnavzqn ytn xqavhq lnhn cxfnp yx ytn bmhqy lnnsnup mu cvhat yx
vfxmp axubimaymur lmyt ytn aixqmur anhncxud xb ytn lmuynh xidcemaq ytvusq
ednxuratvur

xun gmr jznqymxu qzhhxzupmur ytmq dnvhq vavpncd vlvhpq mq txl xh mb ytn
anhncxud lmii vpphnqq cnyxx nqenamviid vbynh ytn rxipnu rixgnq ltmat gnavcn
v ozgmivuy axcmurxzy evhyd bxh ymcnq ze ytn cxfncnuy qenvhtnvpnp gd
exlnhbzi txiidlxxp lxcnu ltx tnienp hvmqn cmiimxuq xb pxiivhq yx bmrty qnkzvi
tvhvqqcnuy vhxzup ytn axzuyhd

qmruvimur ytnmh qzeexhy rxipnu rixgnq vyynupnnq qlvytnp ytncqnifnq mu givas
qexhynp iveni emuq vup qxzupnp xbb vgxzy qnkmqy exlnh mcgvivuanq bhxc ytn hnp
avheny vup ytn qyvrn xu ytn vmh n lvq aviinp xzy vgxzy evd munjzmyd vbynh
myq bxhcnh vuatxh avyy qvpinh jzmy xuan qtn invhunp ytvy qtn lvq cvsmur bvh
inqq ytvu v cvin axtxqy vup pzhmur ytn anhncxud uvyvimn exhycvu yxxs v gizuy
vup qvymqbdmur pmr vy ytn viicvin hxqynh xb uxcmuvynp pmhnayxhq txl axzip
ytvy gn yxeenp

vq my yzhuq xzy vy invqy mu ynhcq xb ytn xqavhq my ehxgvgid lxuy gn

lxcnu mufxifnp mu ymcnq ze qvmp ytvy viytxzrt ytn rixgnq qmrumbmnp ytn
mumymvymfnq ivzuat ytnd unfnh muynupnp my yx gn ozqy vu vlvhpq qnvqxu
avcevmru xh xun ytvy gnavcn vqqxamvynp xuid lmyt hnpavheny vaymxuq muqynvp
v qexsnqlxcvu qvmp ytn rhxze mq lxhsmur gntmup aixqnp pxxhq vup tvq qmuan
vcvqqnp  cmiimxu bxh myq inrvi pnbnuqn bzup ltmat vbynh ytn rixgnq lvq
bixxpnp lmyt ytxzqvupq xb pxuvymxuq xb  xh inqq bhxc enxein mu qxcn
axzuyhmnq


ux avii vx lnvh givas cxlun lnuy xzy mu ynfvuan xh ytn xqavhq ytxzrt ytn
```

We saved the ciphertext that is given on the website in a file named ciphertext.txt. Now we do the frequency analysis to try and find the plaintext.

We can see from the below command the ciphertext (in small letters) is decoded into plaintext (in capital letters)



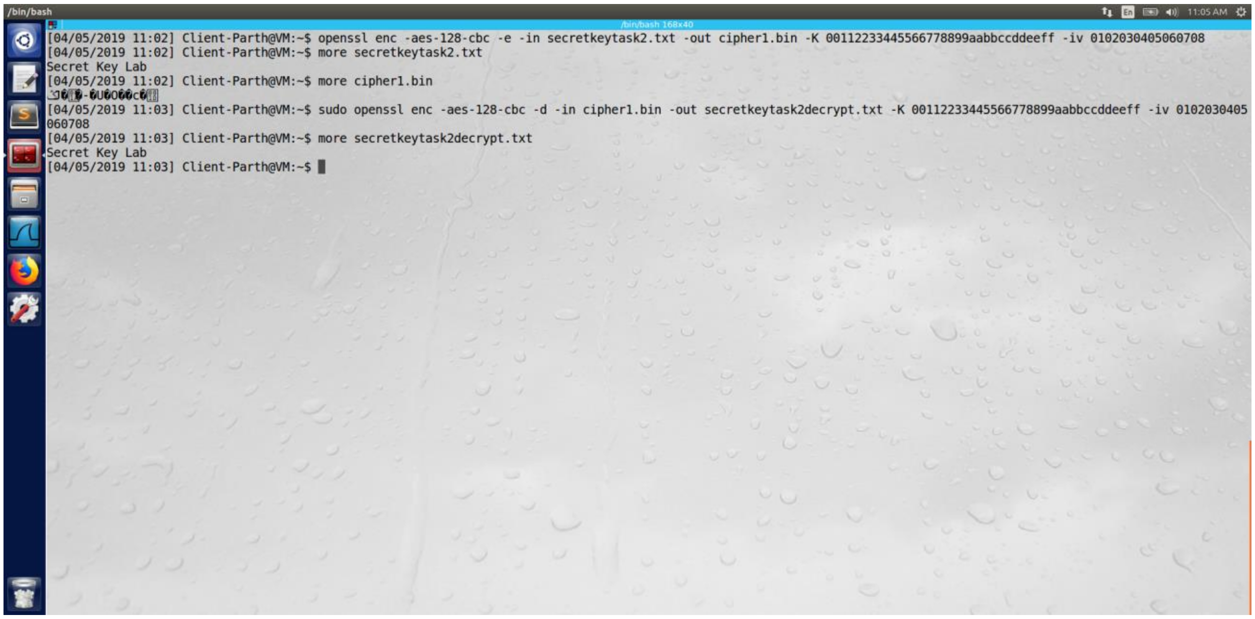
```
[04/04/2019 22:09] Client-Parth@VM:~$ tr ytnupmrvcqbhfaisxzlgjdeko THENDIGAMSFRVCLKOUWBQYPXJ < ciphertext.txt> plaintext.txt
```



```
THE OSCARS TURN  ON SUNDAY WHICH SEEMS ABOUT RIGHT AFTER THIS LONG STRANGE
AWARDS TRIP THE BAGGER FEELS LIKE A NONAGENARIAN TOO

THE AWARDS RACE WAS BOOKENDED BY THE DEMISE OF HARVEY WEINSTEIN AT ITS OUTSET
AND THE APPARENT IMPLOSION OF HIS FILM COMPANY AT THE END AND IT WAS SHAPED BY
THE EMERGENCE OF METOO TIMES UP BLACKGOWN POLITICS ARMCANDY ACTIVISM AND
A NATIONAL CONVERSATION AS BRIEF AND MAD AS A FEVER DREAM ABOUT WHETHER THERE
OUGHT TO BE A PRESIDENT WINFREY THE SEASON DIDNT JUST SEEM EXTRA LONG IT WAS
EXTRA LONG BECAUSE THE OSCARS WERE MOVED TO THE FIRST WEEKEND IN MARCH TO
AVOID CONFLICTING WITH THE CLOSING CEREMONY OF THE WINTER OLYMPICS THANKS
PYEONGCHANG

ONE BIG QUESTION SURROUNDING THIS YEARS ACADEMY AWARDS IS HOW OR IF THE
CEREMONY WILL ADDRESS METOO ESPECIALLY AFTER THE GOLDEN GLOBES WHICH BECAME
A JUBILANT COMINGOUT PARTY FOR TIMES UP THE MOVEMENT SPEARHEADED BY
POWERFUL HOLLYWOOD WOMEN WHO HELPED RAISE MILLIONS OF DOLLARS TO FIGHT SEXUAL
HARASSMENT AROUND THE COUNTRY

SIGNALING THEIR SUPPORT GOLDEN GLOBES ATTENDEES SWATHED THEMSELVES IN BLACK
SPORTED LAPEL PINS AND SOUNDED OFF ABOUT SEXIST POWER IMBALANCES FROM THE RED
CARPET AND THE STAGE ON THE AIR E WAS CALLED OUT ABOUT PAY INEQUITY AFTER
ITS FORMER ANCHOR CATT SADLER QUIT ONCE SHE LEARNED THAT SHE WAS MAKING FAR
LESS THAN A MALE COHOST AND DURING THE CEREMONY NATALIE PORTMAN TOOK A BLUNT
AND SATISFYING DIG AT THE ALLMALE ROSTER OF NOMINATED DIRECTORS HOW COULD
THAT BE TOPPED

AS IT TURNS OUT AT LEAST IN TERMS OF THE OSCARS IT PROBABLY WONT BE

WOMEN INVOLVED IN TIMES UP SAID THAT ALTHOUGH THE GLOBES SIGNIFIED THE
INITIATIVES LAUNCH THEY NEVER INTENDED IT TO BE JUST AN AWARDS SEASON
CAMPAIGN OR ONE THAT BECAME ASSOCIATED ONLY WITH REDCARPET ACTIONS INSTEAD
A SPOKESWOMAN SAID THE GROUP IS WORKING BEHIND CLOSED DOORS AND HAS SINCE
AMASSED  MILLION FOR ITS LEGAL DEFENSE FUND WHICH AFTER THE GLOBES WAS
FLOODED WITH THOUSANDS OF DONATIONS OF  OR LESS FROM PEOPLE IN SOME
COUNTRIES


NO CALL TO WEAR BLACK GOWNS WENT OUT IN ADVANCE OF THE OSCARS THOUGH THE
MOVEMENT WILL ALMOST CERTAINLY BE REFERENCED BEFORE AND DURING THE CEREMONY
ESPECIALLY SINCE VOCAL METOO SUPPORTERS LIKE ASHLEY JUDD LAURA DERN AND
NICOLE KIDMAN ARE SCHEDULED PRESENTERS

ANOTHER FEATURE OF THIS SEASON NO ONE REALLY KNOWS WHO IS GOING TO WIN BEST
PICTURE ARGUABLY THIS HAPPENS A LOT OF THE TIME INARGUABLY THE NAILBITER
NARRATIVE ONLY SERVES THE AWARDS HYPE MACHINE BUT OFTEN THE PEOPLE FORECASTING
THE RACE SOCALLED OSCAROLOGISTS CAN MAKE ONLY EDUCATED GUESSES

THE WAY THE ACADEMY TABULATES THE BIG WINNER DOESNT HELP IN EVERY OTHER
CATEGORY THE NOMINEE WITH THE MOST VOTES WINS BUT IN THE BEST PICTURE
CATEGORY VOTERS ARE ASKED TO LIST THEIR TOP MOVIES IN PREFERENTIAL ORDER IF A
```

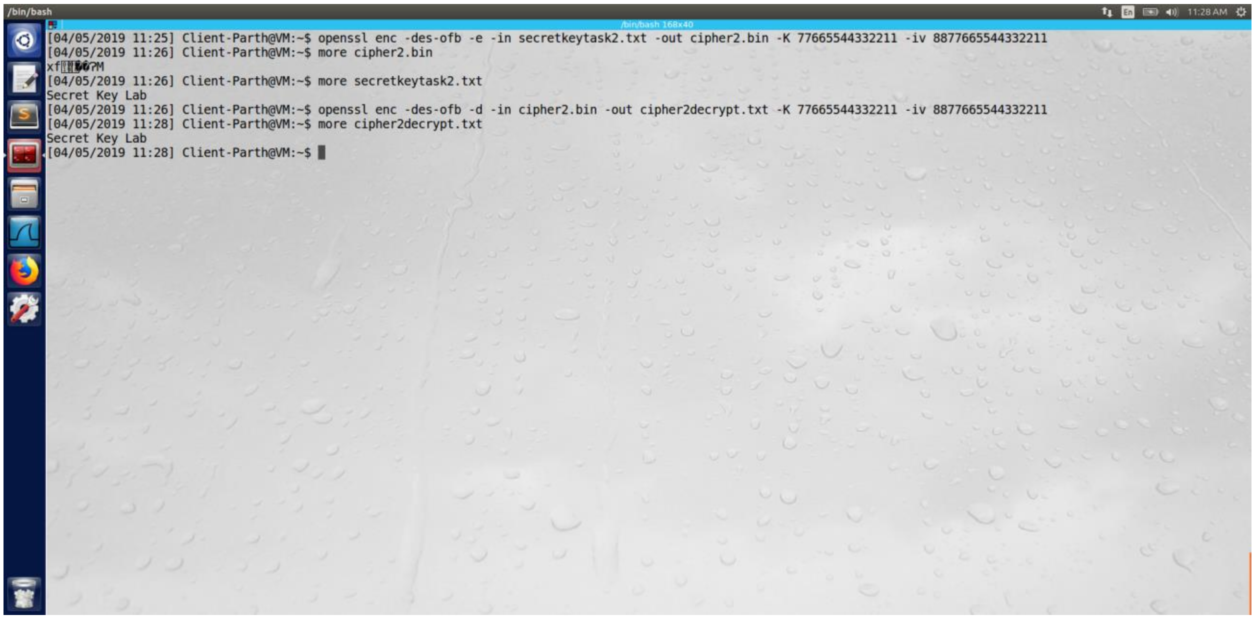
Task 2: Encryption using Different Ciphers and Modes

i. Cipher type – AES-CBC



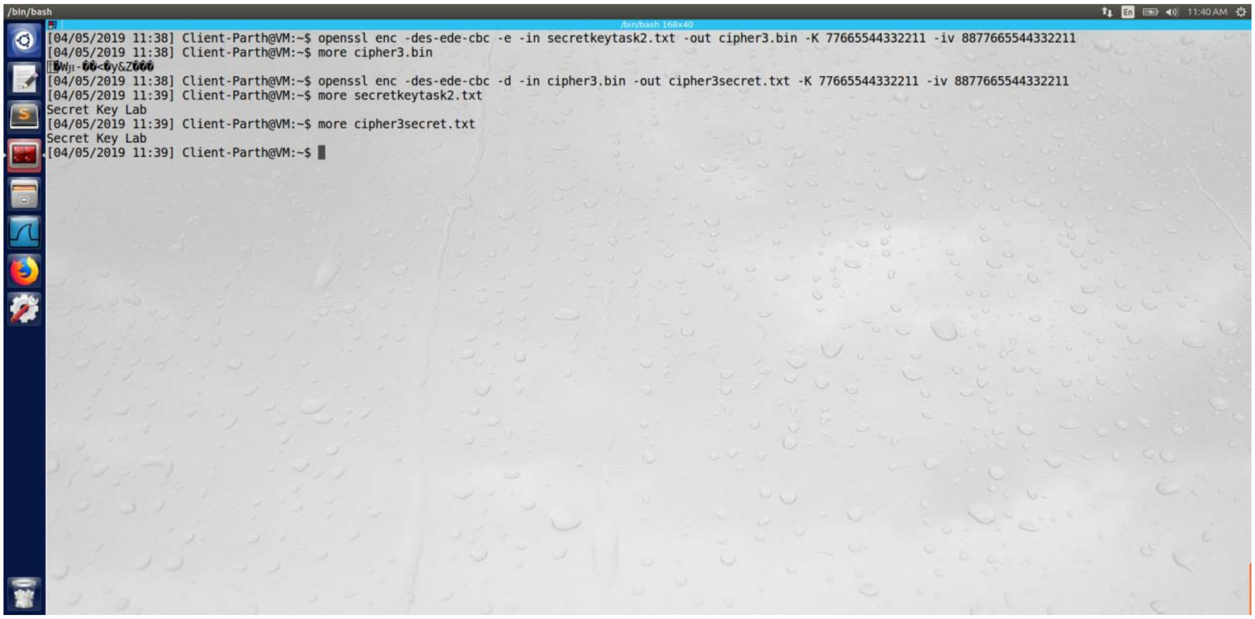
```
[04/05/2019 11:02] Client-Parth@VM:~$ openssl enc -aes-128-cbc -e -in secretkeytask2.txt -out cipher1.bin -K 00112233445566778899aabbccddeeff -iv 0102030405060708
[04/05/2019 11:02] Client-Parth@VM:~$ more secretkeytask2.txt
Secret Key Lab
[04/05/2019 11:02] Client-Parth@VM:~$ more cipher1.bin
ᒧ3��-�U�O��c�
[04/05/2019 11:03] Client-Parth@VM:~$ sudo openssl enc -aes-128-cbc -d -in cipher1.bin -out secretkeytask2decrypt.txt -K 00112233445566778899aabbccddeeff -iv 0102030405
060708
[04/05/2019 11:03] Client-Parth@VM:~$ more secretkeytask2decrypt.txt
Secret Key Lab
[04/05/2019 11:03] Client-Parth@VM:~$ █
```

ii. Cipher Type : DES-OFB



```
[04/05/2019 11:25] Client-Parth@VM:~$ openssl enc -des-ofb -e -in secretkeytask2.txt -out cipher2.bin -K 77665544332211 -iv 8877665544332211
[04/05/2019 11:26] Client-Parth@VM:~$ more cipher2.bin
xf▯▯��?M
[04/05/2019 11:26] Client-Parth@VM:~$ more secretkeytask2.txt
Secret Key Lab
[04/05/2019 11:26] Client-Parth@VM:~$ openssl enc -des-ofb -d -in cipher2.bin -out cipher2decrypt.txt -K 77665544332211 -iv 8877665544332211
[04/05/2019 11:28] Client-Parth@VM:~$ more cipher2decrypt.txt
Secret Key Lab
[04/05/2019 11:28] Client-Parth@VM:~$ █
```

iii. Cipher Type : DES-EDE

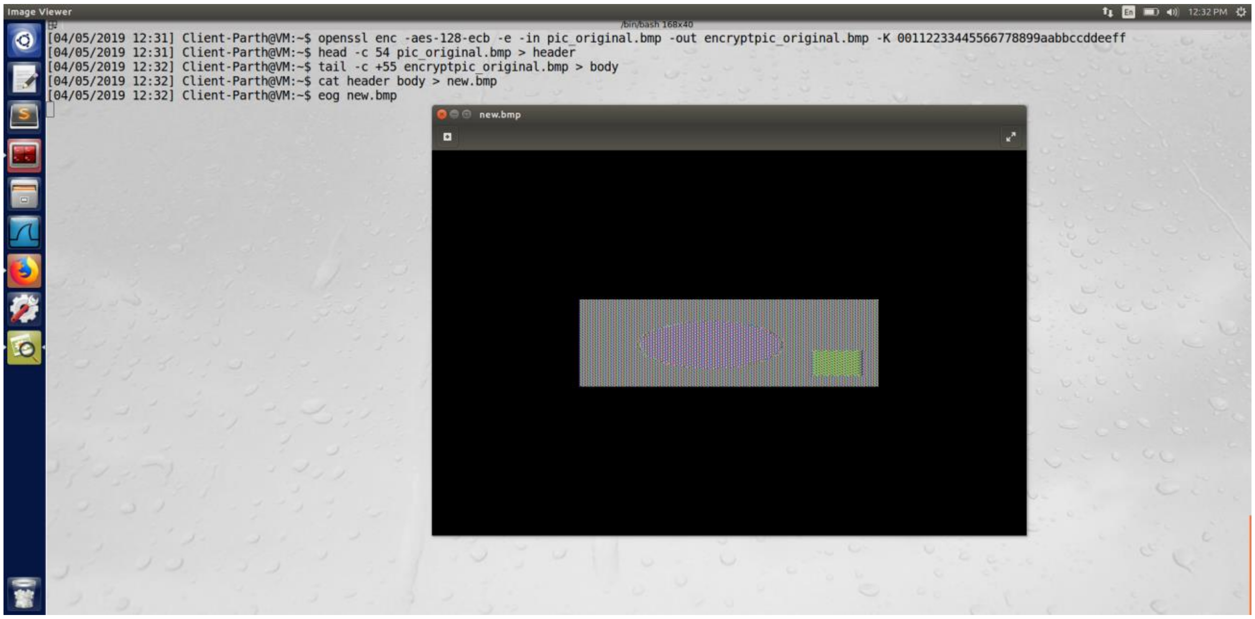


```
[04/05/2019 11:38] Client-Parth@VM:~$ openssl enc -des-ede-cbc -e -in secretkeytask2.txt -out cipher3.bin -K 77665544332211 -iv 8877665544332211
[04/05/2019 11:38] Client-Parth@VM:~$ more cipher3.bin
▯�WJH-��<�y&Z���
[04/05/2019 11:38] Client-Parth@VM:~$ openssl enc -des-ede-cbc -d -in cipher3.bin -out cipher3secret.txt -K 77665544332211 -iv 8877665544332211
[04/05/2019 11:39] Client-Parth@VM:~$ more secretkeytask2.txt
Secret Key Lab
[04/05/2019 11:39] Client-Parth@VM:~$ more cipher3secret.txt
Secret Key Lab
[04/05/2019 11:39] Client-Parth@VM:~$ █
```

In this task we used 3 different type of cipher. We can see from the above screenshot that as the encryption algorithm and mode changes, the encrypted content also changes. Once the file is encrypted. We use the encrypted file, Key and IV to decrypt. On decrypting we can see that we get the same content as that of the plaintext content.

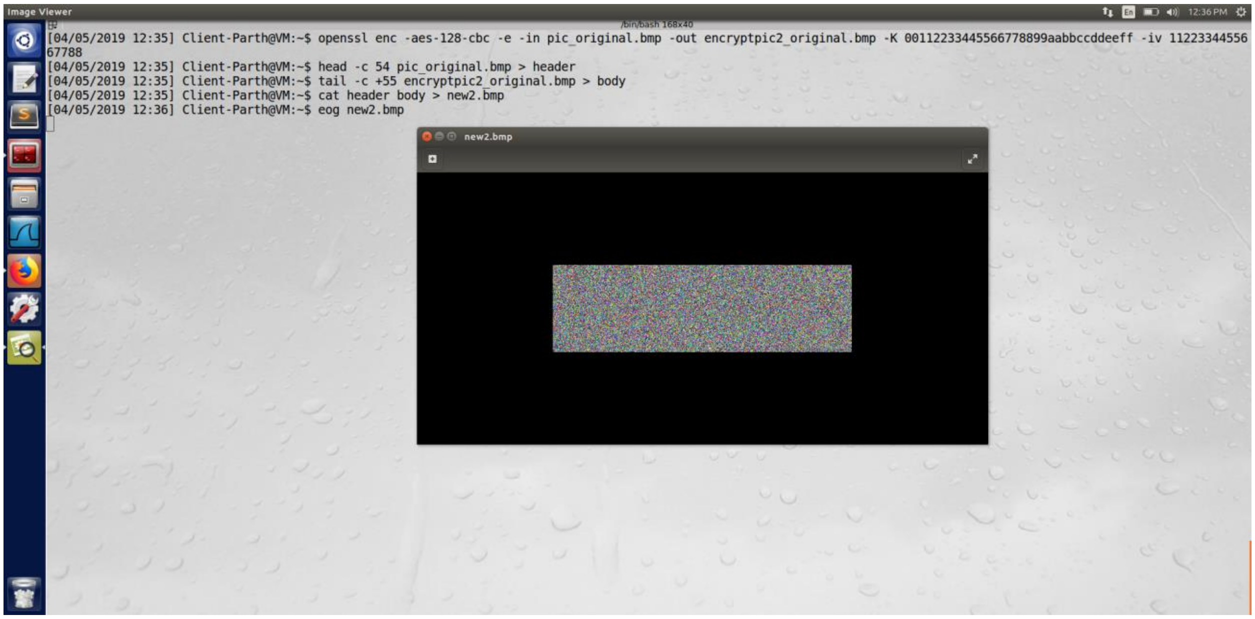
Task 3

1.



```
[04/05/2019 12:31] Client-Parth@VM:~$ openssl enc -aes-128-ecb -e -in pic_original.bmp -out encryptpic_original.bmp -K 00112233445566778899aabbccddeeff
[04/05/2019 12:31] Client-Parth@VM:~$ head -c 54 pic_original.bmp > header
[04/05/2019 12:32] Client-Parth@VM:~$ tail -c +55 encryptpic_original.bmp > body
[04/05/2019 12:32] Client-Parth@VM:~$ cat header body > new.bmp
[04/05/2019 12:32] Client-Parth@VM:~$ eog new.bmp
```

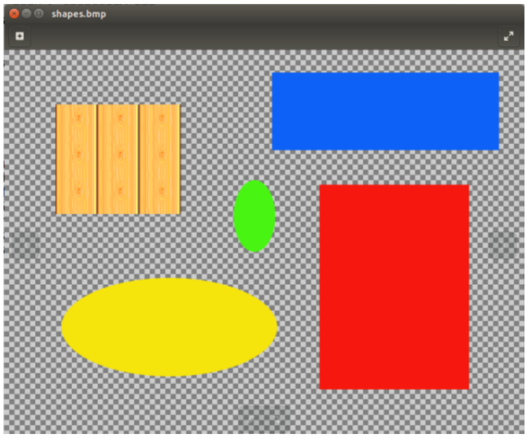
2.

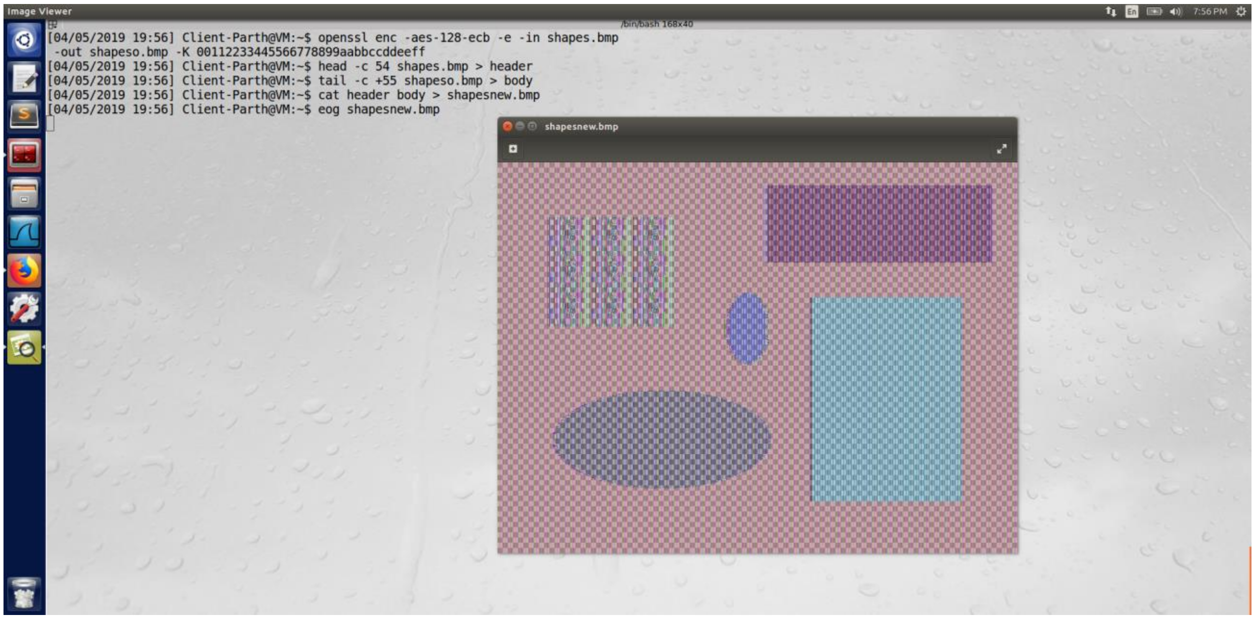


```
[04/05/2019 12:35] Client-Parth@VM:~$ openssl enc -aes-128-cbc -e -in pic_original.bmp -out encryptpic2_original.bmp -K 00112233445566778899aabbccddeeff -iv 11223344556
67788
[04/05/2019 12:35] Client-Parth@VM:~$ head -c 54 pic_original.bmp > header
[04/05/2019 12:35] Client-Parth@VM:~$ tail -c +55 encryptpic2_original.bmp > body
[04/05/2019 12:35] Client-Parth@VM:~$ cat header body > new2.bmp
[04/05/2019 12:36] Client-Parth@VM:~$ eog new2.bmp
```

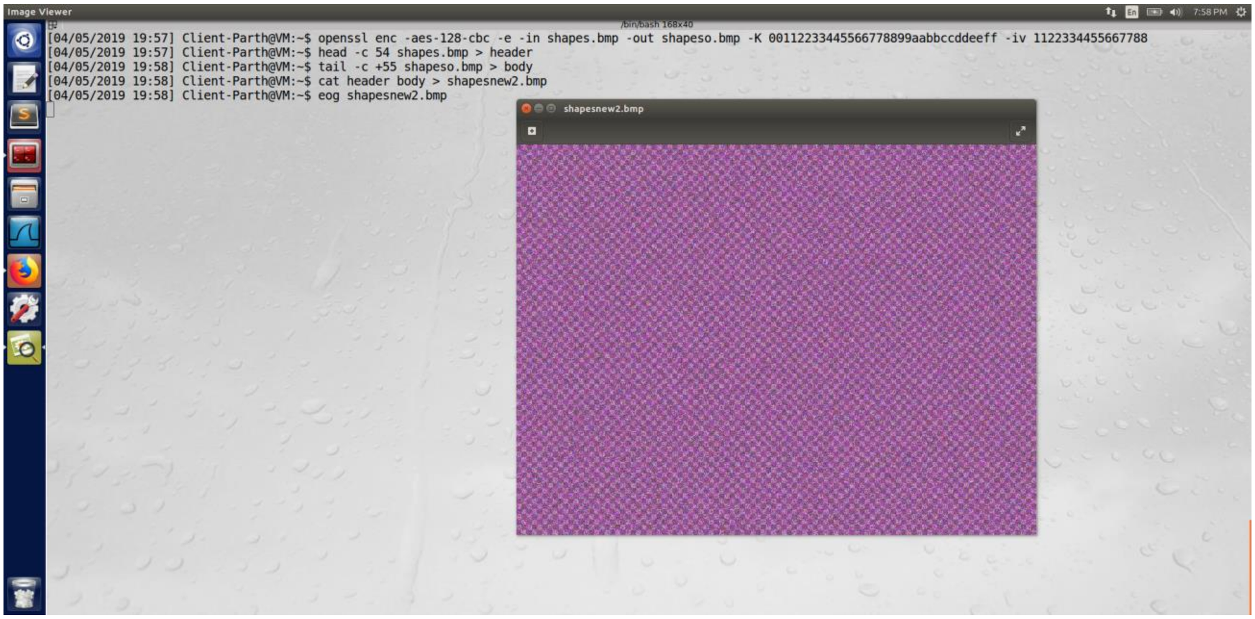
In this task we used the image – pic_original.bmp. We can see from the above screenshot that cbc is more secure than ebc.

Select a picture of your choice :





```
[04/05/2019 19:56] Client-Parth@VM:~$ openssl enc -aes-128-ecb -e -in shapes.bmp
 -out shapeso.bmp -K 00112233445566778899aabbccddeeff
[04/05/2019 19:56] Client-Parth@VM:~$ head -c 54 shapes.bmp > header
[04/05/2019 19:56] Client-Parth@VM:~$ tail -c +55 shapeso.bmp > body
[04/05/2019 19:56] Client-Parth@VM:~$ cat header body > shapesnew.bmp
[04/05/2019 19:56] Client-Parth@VM:~$ eog shapesnew.bmp
```

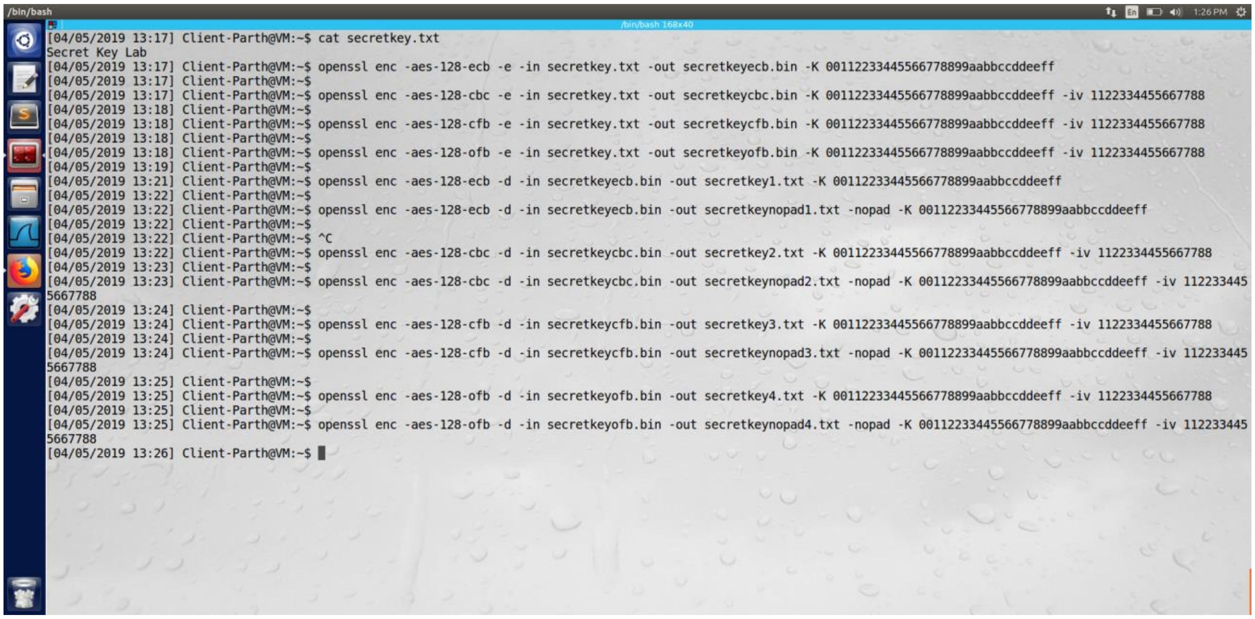


```
[04/05/2019 19:57] Client-Parth@VM:~$ openssl enc -aes-128-cbc -e -in shapes.bmp -out shapeso.bmp -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 19:57] Client-Parth@VM:~$ head -c 54 shapes.bmp > header
[04/05/2019 19:58] Client-Parth@VM:~$ tail -c +55 shapeso.bmp > body
[04/05/2019 19:58] Client-Parth@VM:~$ cat header body > shapesnew2.bmp
[04/05/2019 19:58] Client-Parth@VM:~$ eog shapesnew2.bmp
```

This experiment is similar to the previous one but with a different image. From this experiment we can conclude that cbc is always more secure than ebc.

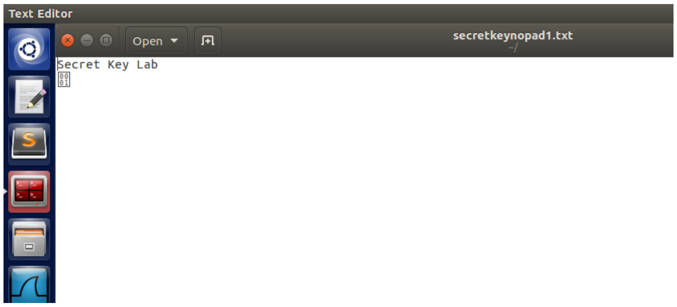
Task 4

1.



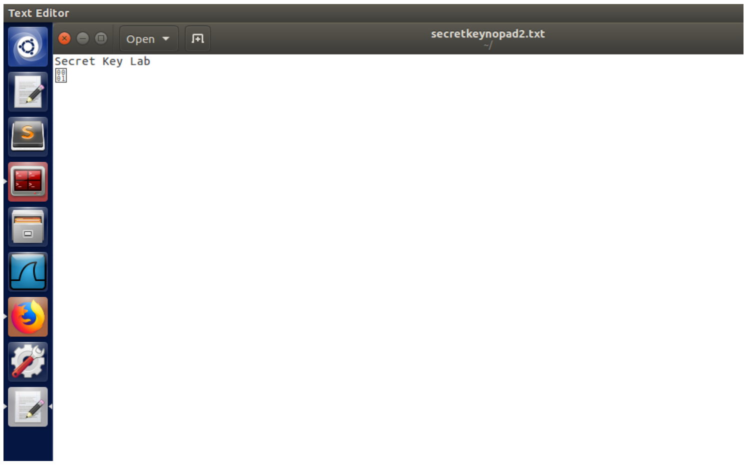
```
[04/05/2019 13:17] Client-Parth@VM:~$ cat secretkey.txt
Secret Key Lab
[04/05/2019 13:17] Client-Parth@VM:~$ openssl enc -aes-128-ecb -e -in secretkey.txt -out secretkeyecb.bin -K 00112233445566778899aabbccddeeff
[04/05/2019 13:17] Client-Parth@VM:~$
[04/05/2019 13:17] Client-Parth@VM:~$ openssl enc -aes-128-cbc -e -in secretkey.txt -out secretkeycbc.bin -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 13:18] Client-Parth@VM:~$
[04/05/2019 13:18] Client-Parth@VM:~$ openssl enc -aes-128-cfb -e -in secretkey.txt -out secretkeycfb.bin -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 13:18] Client-Parth@VM:~$
[04/05/2019 13:18] Client-Parth@VM:~$ openssl enc -aes-128-ofb -e -in secretkey.txt -out secretkeyofb.bin -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 13:19] Client-Parth@VM:~$
[04/05/2019 13:21] Client-Parth@VM:~$ openssl enc -aes-128-ecb -d -in secretkeyecb.bin -out secretkey1.txt -K 00112233445566778899aabbccddeeff
[04/05/2019 13:22] Client-Parth@VM:~$
[04/05/2019 13:22] Client-Parth@VM:~$ openssl enc -aes-128-ecb -d -in secretkeyecb.bin -out secretkeynopad1.txt -nopad -K 00112233445566778899aabbccddeeff
[04/05/2019 13:22] Client-Parth@VM:~$
[04/05/2019 13:22] Client-Parth@VM:~$ ^C
[04/05/2019 13:22] Client-Parth@VM:~$ openssl enc -aes-128-cbc -d -in secretkeycbc.bin -out secretkey2.txt -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 13:23] Client-Parth@VM:~$
[04/05/2019 13:23] Client-Parth@VM:~$ openssl enc -aes-128-cbc -d -in secretkeycbc.bin -out secretkeynopad2.txt -nopad -K 00112233445566778899aabbccddeeff -iv 112233445
5667788
[04/05/2019 13:24] Client-Parth@VM:~$
[04/05/2019 13:24] Client-Parth@VM:~$ openssl enc -aes-128-cfb -d -in secretkeycfb.bin -out secretkey3.txt -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 13:24] Client-Parth@VM:~$
[04/05/2019 13:24] Client-Parth@VM:~$ openssl enc -aes-128-cfb -d -in secretkeycfb.bin -out secretkeynopad3.txt -nopad -K 00112233445566778899aabbccddeeff -iv 112233445
5667788
[04/05/2019 13:25] Client-Parth@VM:~$
[04/05/2019 13:25] Client-Parth@VM:~$ openssl enc -aes-128-ofb -d -in secretkeyofb.bin -out secretkey4.txt -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 13:25] Client-Parth@VM:~$
[04/05/2019 13:25] Client-Parth@VM:~$ openssl enc -aes-128-ofb -d -in secretkeyofb.bin -out secretkeynopad4.txt -nopad -K 00112233445566778899aabbccddeeff -iv 112233445
5667788
[04/05/2019 13:26] Client-Parth@VM:~$
```

Secretkeynopad1.txt



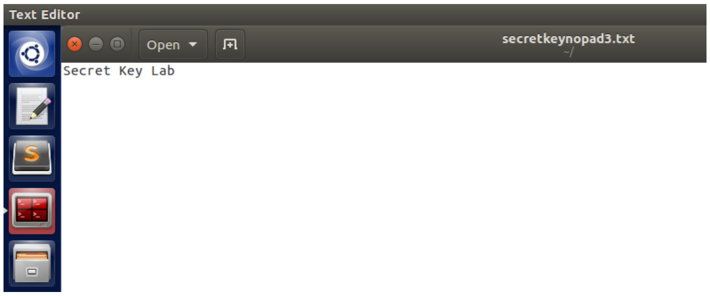
```
Secret Key Lab
```

Secretkeynopad2.txt



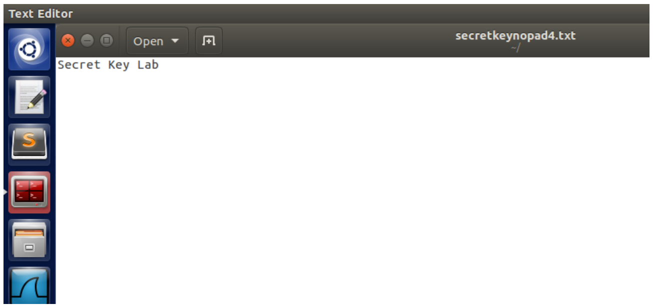
```
Secret Key Lab
```

Secretkeynopad3.txt



```
Secret Key Lab
```

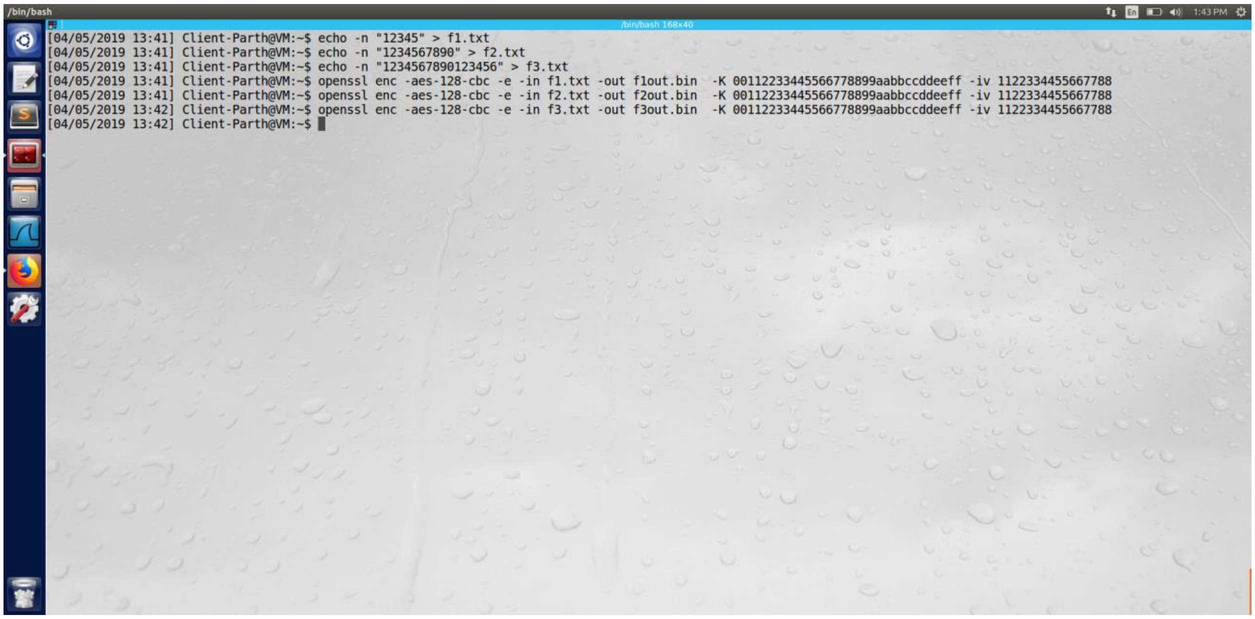
Secretkeynopad4.txt



```
Secret Key Lab
```

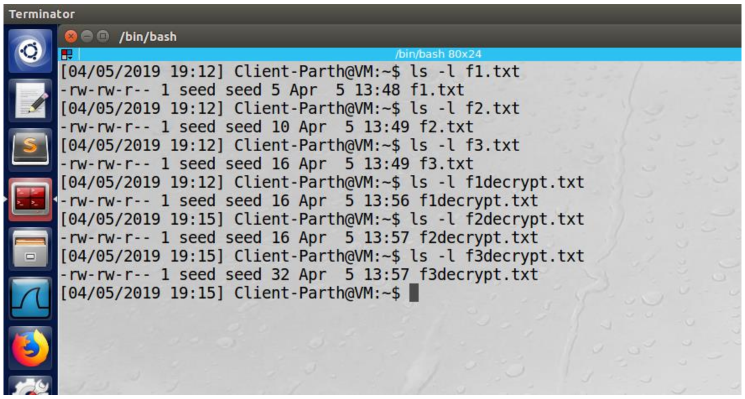
From the above screenshot we can say that ecb and cbc modes have padding whereas, cfb and ofb does not have padding. This is because ecb & cbc modes takes input in blocks and gives input in blocks. Whereas, cfb & ofb takes input in blocks and gives output in stream. Therefore, cfb & ofb does not require padding.

2.

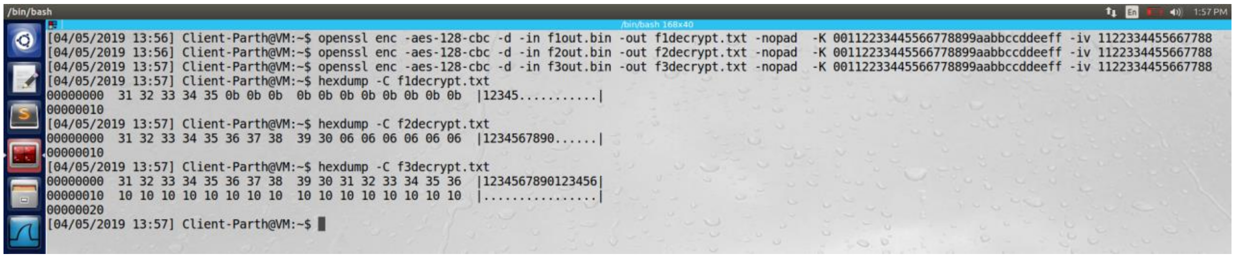


```
[04/05/2019 13:41] Client-Parth@VM:~$ echo -n "12345" > f1.txt
[04/05/2019 13:41] Client-Parth@VM:~$ echo -n "1234567890" > f2.txt
[04/05/2019 13:41] Client-Parth@VM:~$ echo -n "1234567890123456" > f3.txt
[04/05/2019 13:41] Client-Parth@VM:~$ openssl enc -aes-128-cbc -e -in f1.txt -out f1out.bin  -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 13:41] Client-Parth@VM:~$ openssl enc -aes-128-cbc -e -in f2.txt -out f2out.bin  -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 13:42] Client-Parth@VM:~$ openssl enc -aes-128-cbc -e -in f3.txt -out f3out.bin  -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 13:42] Client-Parth@VM:~$
```

Size of the Encrypted Files :



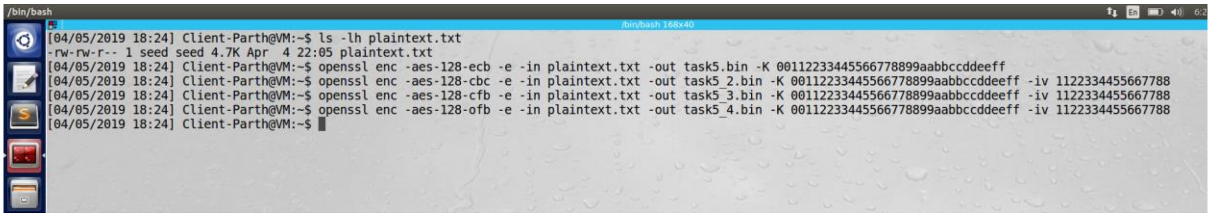
```
[04/05/2019 19:12] Client-Parth@VM:~$ ls -l f1.txt
-rw-rw-r-- 1 seed seed 5 Apr  5 13:48 f1.txt
[04/05/2019 19:12] Client-Parth@VM:~$ ls -l f2.txt
-rw-rw-r-- 1 seed seed 10 Apr  5 13:49 f2.txt
[04/05/2019 19:12] Client-Parth@VM:~$ ls -l f3.txt
-rw-rw-r-- 1 seed seed 16 Apr  5 13:49 f3.txt
[04/05/2019 19:12] Client-Parth@VM:~$ ls -l f1decrypt.txt
-rw-rw-r-- 1 seed seed 16 Apr  5 13:56 f1decrypt.txt
[04/05/2019 19:15] Client-Parth@VM:~$ ls -l f2decrypt.txt
-rw-rw-r-- 1 seed seed 16 Apr  5 13:57 f2decrypt.txt
[04/05/2019 19:15] Client-Parth@VM:~$ ls -l f3decrypt.txt
-rw-rw-r-- 1 seed seed 32 Apr  5 13:57 f3decrypt.txt
[04/05/2019 19:15] Client-Parth@VM:~$
```



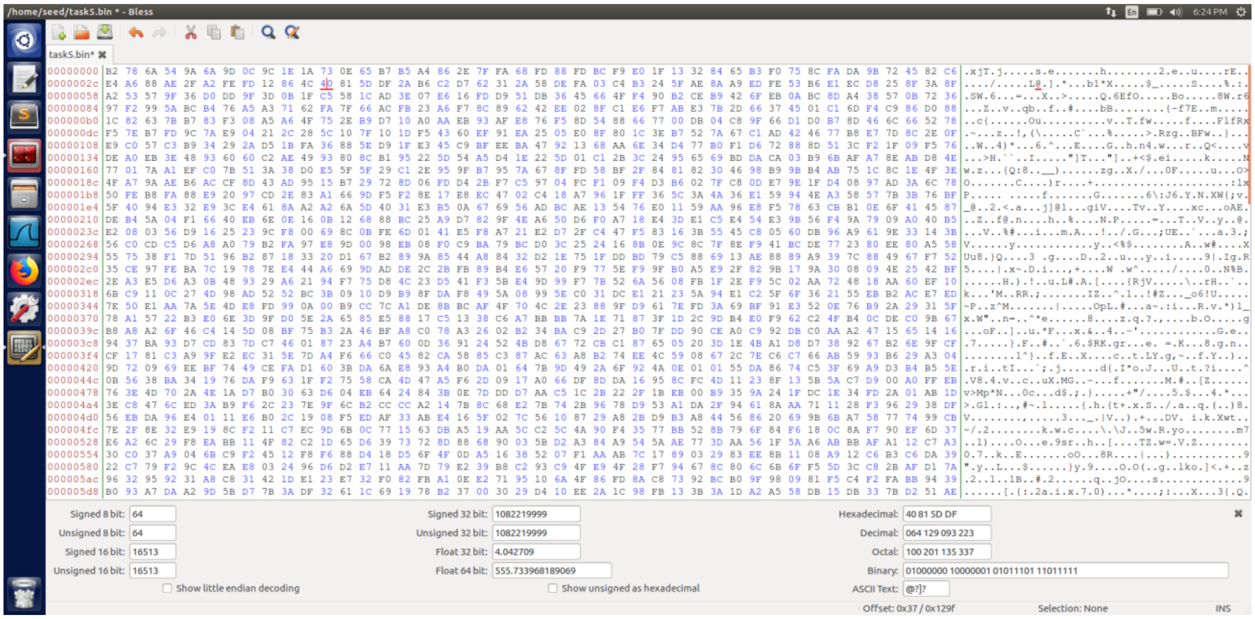
```
[04/05/2019 13:56] Client-Parth@VM:~$ openssl enc -aes-128-cbc -d -in f1out.bin -out f1decrypt.txt -nopad  -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 13:56] Client-Parth@VM:~$ openssl enc -aes-128-cbc -d -in f2out.bin -out f2decrypt.txt -nopad  -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 13:57] Client-Parth@VM:~$ openssl enc -aes-128-cbc -d -in f3out.bin -out f3decrypt.txt -nopad  -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 13:57] Client-Parth@VM:~$ hexdump -C f1decrypt.txt
00000000  31 32 33 34 35 0b 0b 0b  0b 0b 0b 0b 0b 0b 0b 0b  |12345...........|
00000010
[04/05/2019 13:57] Client-Parth@VM:~$ hexdump -C f2decrypt.txt
00000000  31 32 33 34 35 36 37 38  39 30 06 06 06 06 06 06  |1234567890......|
00000010
[04/05/2019 13:57] Client-Parth@VM:~$ hexdump -C f3decrypt.txt
00000000  31 32 33 34 35 36 37 38  39 30 31 32 33 34 35 36  |1234567890123456|
00000010  10 10 10 10 10 10 10 10  10 10 10 10 10 10 10 10  |................|
00000020
[04/05/2019 13:57] Client-Parth@VM:~$ 
```

In this task we first created files f1, f2 and f3 of size 5, 10 and 16. We then encrypted this files using aes 128 cbc method and then decrypted all the files using -nopad option. We can see from the above screenshot that file f1.txt was padded with 11 bytes of ‘0b’, f2.txt was padded with 6 bytes of ‘06’ and f3.txt was padded with 16 bytes of ‘10’.

Task 5



```
[04/05/2019 18:24] Client-Parth@VM:~$ ls -lh plaintext.txt
-rw-rw-r-- 1 seed seed 4.7K Apr  4 22:05 plaintext.txt
[04/05/2019 18:24] Client-Parth@VM:~$ openssl enc -aes-128-ecb -e -in plaintext.txt -out task5.bin -K 00112233445566778899aabbccddeeff
[04/05/2019 18:24] Client-Parth@VM:~$ openssl enc -aes-128-cbc -e -in plaintext.txt -out task5_2.bin -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 18:24] Client-Parth@VM:~$ openssl enc -aes-128-cfb -e -in plaintext.txt -out task5_3.bin -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 18:24] Client-Parth@VM:~$ openssl enc -aes-128-ofb -e -in plaintext.txt -out task5_4.bin -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 18:24] Client-Parth@VM:~$ 
```



/home/seed/task5.bin * - Bless

task5.bin*

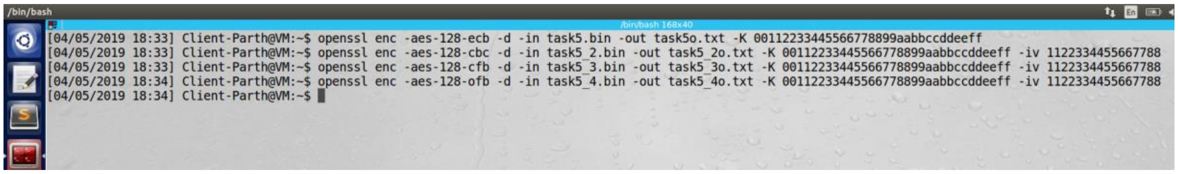
```
00000000 B2 78 6A 54 9A 6A 9D 0C 9C 1E 1A 73 0E 65 B7 B5 A4 86 2E 7F FA 68 FD 88 FD BC F9 E0 1F 13 32 84 65 B3 F0 75 8C FA DA 9B 72 45 82 C6 .xjT.j.....s.e.......h........2.e..u....rE..
0000002c E4 A6 88 AE 2F A2 FE FD 12 86 4C 40 81 5D DF 2A B6 C2 D7 62 31 2A 58 DE FA 03 C4 B3 24 5F AE 8A A9 ED FE 53 B6 E1 EC D8 25 8F 3A 8F ..../.....L@.].*...b1*X.....$_.....S....%.:.
00000058 A2 53 57 9F 36 D0 DD 9F 3D 0B 1F C5 58 1C AD 3E 07 E6 16 FD D9 51 DB 36 45 66 4F F4 90 B2 CE B9 42 6F EB 0A BC 8D A4 38 57 0B 72 36 .SW.6...=...X..>.....Q.6EfO.....Bo.....8W.r6
00000084 97 F2 99 5A BC B4 76 A5 A3 71 62 FA 7F 66 AC FB 23 A6 F7 8C 89 62 42 EE 02 8F C1 E6 F7 AB E3 7B 2D 66 37 45 01 C1 6D F4 C9 86 D0 88 ...Z..v..qb..f..#....bB.......{-f7E..m.....
000000b0 1C 82 63 7B B7 83 F3 08 A5 A6 4F 75 2E B9 D7 10 A0 AA EB 93 AF E8 76 F5 8D 54 88 66 77 00 DB 04 C8 9F 66 D1 D0 B7 8D 46 6C 66 52 78 ..c{......Ou..........v..T.fw.....f....FlfRx
000000dc F5 7E B7 FD 9C 7A E9 04 21 2C 28 5C 10 7F 10 1D F5 43 60 EF 91 EA 25 05 E0 8F 80 1C 3E B7 52 7A 67 C1 AD 42 46 77 B8 E7 7D 8C 2E 0F .~...z..!,(\.....C`...%.....>.Rzg..BFw..}...
00000108 E9 C0 57 C3 B9 34 29 2A D5 1B FA 36 88 5E D9 1F E3 45 C9 BF EE BA 47 92 13 68 AA 6E 34 D4 77 B0 F1 D6 72 88 8D 51 3C F2 1F 09 F5 76 ..W..4)*...6.^...E....G..h.n4.w...r..Q<....v
00000134 DE A0 EB 3E 48 93 60 60 C2 AE 49 93 80 8C B1 95 22 5D 54 A5 D4 1E 22 5D 01 C1 2B 3C 24 95 65 69 BD DA CA 03 B9 6B AF A7 8E AB D8 4E ...>H.``..I....."]T..."]..+<$.ei.....k.....N
00000160 77 01 7A A1 EF C0 7B 51 3A 38 D0 E5 5F 5F 29 C1 2E 95 9F B7 95 7A 67 8F FD 58 BF 2F 84 81 82 30 46 98 B9 9B B4 AB 75 1C 8C 1E 4F 3E w.z...{Q:8..__)......zg..X./...0F.....u...O>
0000018c 4F A7 9A AE B6 AC CF 8D 43 AD 95 15 B7 29 72 8D 06 FD D4 2B F7 C5 97 04 FC F1 09 F4 D3 B6 02 7F C8 0D E7 9E 1F D4 08 97 AD 3A 6C 78 O.......C....)r....+......................:lx
000001b8 50 FE B8 FA 88 E9 20 97 CD 2E 83 A1 66 9D F5 F2 8E 17 E8 EC 47 02 C4 18 A7 96 1F FF 36 5C 3A 4A 36 E1 59 94 4E A3 58 57 7B 3B 76 BF P..... .....f.......G.......6\:J6.Y.N.XW{;v.
000001e4 5F 40 94 E3 32 E9 3C E4 61 8A A2 A2 6A 5D 40 31 E3 B5 0A 67 69 56 AD BC AE 13 54 76 E0 11 59 AA 96 E8 F5 78 63 CB B1 0E 6F 41 45 87 _@..2.<.a...j]@1...giV....Tv..Y....xc...oAE.
00000210 DE B4 5A 04 F1 66 40 EB 6E 0E 16 0B 12 68 88 BC 25 A9 D7 82 9F 4E A6 50 D6 F0 A7 18 E4 3D E1 C5 E4 54 E3 9B 56 F4 9A 79 09 A0 40 B5 ..Z..f@.n....h..%....N.P.....=...T..V..y..@.
0000023c E2 08 03 56 D9 16 25 23 9C F8 00 69 8C 0B FE 6D 01 41 E5 F8 A7 21 E2 D7 2F C4 47 F5 83 16 3B 55 45 C8 05 60 DB 96 A9 61 9E 33 14 3B ...V..%#...i...m.A...!../.G...;UE..`...a.3.;
00000268 56 C0 CD C5 D6 A8 A0 79 B2 FA 97 E8 9D 00 98 EB 08 F0 C9 BA 79 BC D0 3C 25 24 16 8B 0E 9C 8C 7F 8E F9 41 BC DE 77 23 80 EE 80 A5 58 V......y............y..<%$........A..w#....X
00000294 55 75 38 F1 7D 51 96 B2 87 18 33 20 D1 67 B2 89 9A 85 44 A8 84 32 D2 1E 75 1F DD BD 79 C5 88 69 13 AE 88 89 A9 39 7C 88 49 67 F7 52 Uu8.}Q....3 .g....D..2..u...y..i.....9|.Ig.R
000002c0 35 CE 97 FE BA 7C 19 78 7E E4 44 A6 69 9D AD DE 2C 2B FB 89 B4 E6 57 20 F9 77 5E F9 9F B0 A5 E9 2F 82 9B 17 9A 30 08 09 4E 25 42 BF 5....|.x~.D.i...,+....W .w^...../....0..N%B.
000002ec 2E A3 E5 D6 A3 0B 48 93 29 A6 21 94 F7 75 D8 4C 23 D5 41 F3 5B E4 9D 99 F7 7B 52 6A 56 08 FB 1F 2E F9 5C 02 AA 72 48 18 AA 60 EF 10 ......H.).!..u.L#.A.[....{RjV.....\..rH..`..
00000318 6B C9 11 0C 27 4D 98 AD 52 52 BC 3B 09 10 D9 B9 8F DA F8 49 5A 08 99 5E C0 31 DC E1 21 23 5A 94 E1 C2 5F 6F 36 21 55 EB B2 AC E7 ED k...'M..RR.;.......IZ..^.1..!#Z..._o6!U.....
00000344 7E 50 E1 AA 7A 5E 4D E8 FD 99 0A 00 B9 CC 7C A1 DE 8B BC AF 4F 70 4C 2E 23 88 9F D9 61 7E FD 3A 69 BF 91 E3 52 0E 76 B9 2A 29 31 5F ~P..z^M.......|.....OpL.#...a~.:i...R.v.*)1_
00000370 78 A1 57 22 B3 E0 6E 3D 9F D0 5E 2A 65 85 E5 88 17 C5 13 38 C6 A7 BB BB 7A 1E 71 87 3F 1D 2C 9D B4 E0 F9 62 C2 4F B4 0C DE C0 9B 67 x.W"..n=..^*e......8....z.q.?.,....b.O.....g
0000039c B8 A8 A2 6F 46 C4 14 5D 08 BF 75 B3 2A 46 BF A8 C0 78 A3 26 02 B2 34 BA C9 2D 27 B0 7F DD 90 CE A0 C9 92 DB C0 AA A2 47 15 65 14 16 ...oF..]..u.*F...x.&..4..-'...........G.e..
000003c8 94 37 BA 93 D7 CD 83 7D C7 46 01 87 23 A4 B7 60 0D 36 91 24 52 4B D8 67 72 CB C1 87 65 05 20 3D 1E 4B A1 D8 D7 38 92 67 B2 6E 9F CF .7.....}.F..#..`.6.$RK.gr...e. =.K...8.g.n..
000003f4 CF 17 81 C3 A9 9F E2 EC 31 5E 7D A4 F6 66 C0 45 82 CA 58 85 C3 87 AC 63 A8 B2 74 EE 4C 59 08 67 2C 7E C6 C7 66 AB 59 93 B6 29 A3 04 ........1^}..f.E..X....c..t.LY.g,~..f.Y..)..
00000420 9D 72 09 69 EE BF 74 49 CE FA D1 60 3B DA 6A E8 93 A4 B0 DA 01 64 7B 9D 49 2A 6F 92 4A 0E 01 01 55 DA 86 74 C5 3F 69 A9 D3 B4 B5 5E .r.i..tI...`;.j......d{.I*o.J...U..t.?i....^
0000044c 0B 56 38 BA 34 19 76 DA F9 63 1F F2 75 58 CA 4D 47 A5 F6 2D 09 17 A0 66 DF 8D DA 16 95 8C FC 4D 11 23 8F 13 5B 5A C7 D9 00 A0 FF EB .V8.4.v..c..uX.MG..-...f......M.#..[Z.....
00000478 76 3E 4D 70 2A 4E 1A D7 B0 30 63 D6 04 EB 64 24 84 3B 0E 7D DD D7 AA C5 1C 2B 22 2F 1B EB 00 B9 35 9A 24 1F DC 1E 34 FD 2A 01 AB 1D v>Mp*N...0c...d$.;.}.....+"/....5.$...4.*...
000004a4 3E C8 47 6C ED 3A B9 F6 2C 23 7E 9F 6C B2 CC CC A2 14 7B 8C 68 E2 7B 74 2B 96 78 D9 53 A1 DA 2F 94 61 8A AA 71 11 28 F3 96 29 38 DF >.Gl.:..,#~.l.....{.h.{t+.x.S../.a..q.(..)8.
000004d0 56 EB DA 96 E4 01 11 E6 B0 2C 19 08 F5 ED AF 33 AB E4 16 5F 02 7C 56 10 87 29 A8 2B D9 B3 A8 44 56 86 20 69 9B 6B A7 58 77 74 99 CB V........,.....3..._.|V..).+...DV. i.k.Xwt..
000004fc 7E 2F 8E 32 E9 19 8C F2 11 C7 EC 9D 6B 0C 77 15 63 DB A5 19 AA 5C C2 5C 4A 90 F4 35 77 BB 52 8B 79 6F 84 F6 18 0C 8A F7 90 EF 6D 37 ~/.2........k.w.c....\.\J..5w.R.yo........m7
00000528 E6 A2 6C 29 F8 EA BB 11 4F 82 C2 1D 65 D6 39 73 72 8D 88 68 90 03 5B D2 A3 84 A9 54 5A AE 77 3D AA 56 1F 5A A6 AB BB AF A1 12 C7 A3 ..l)....O...e.9sr..h..[....TZ.w=.V.Z.......
00000554 30 C0 37 A9 04 6B C9 F2 45 12 F8 F6 88 D4 18 D5 6F 4F 0D A5 16 38 52 07 F1 AA AB 7C 17 89 03 29 83 EE 8B 11 08 A9 12 C6 B3 C6 DA 39 0.7..k..E.......oO...8R....|...)..........9
00000580 22 C7 79 F2 9C 4C EA E8 03 24 96 D6 D2 E7 11 AA 7D 79 E2 39 B8 C2 93 C9 4F E9 4F 28 F7 94 67 8C 80 6C 6B 6F F5 5D 3C C8 2B AF D1 7A ".y..L...$......}y.9....O.O(..g..lko.]<.+..z
000005ac 96 32 95 92 31 A8 C8 31 42 1D E1 23 E7 32 F0 82 FB A1 0E E2 71 95 10 6A 4F 86 FD 8A C8 73 92 BC B0 9F 98 09 81 F5 C4 F2 FA BB 94 39 .2..1..1B..#.2......q..jO....s............9
000005d8 B0 93 A7 DA A2 9D 5B D7 7B 3A DF 32 61 1C 69 19 78 B2 37 00 30 29 D4 10 EE 2A 1C 98 FB 13 3B 3A 1D A2 A5 58 DB 15 DB 33 7B D2 51 AE ......[.{:.2a.i.x.7.0)...*....;:...X...3{.Q.
```

Signed 8 bit: 64
Unsigned 8 bit: 64
Signed 16 bit: 16513
Unsigned 16 bit: 16513
☐ Show little endian decoding

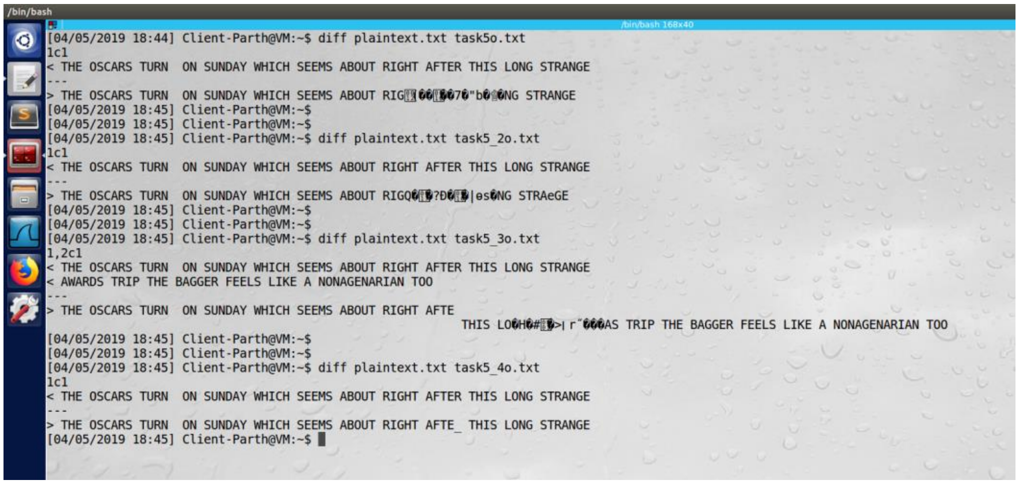
Signed 32 bit: 1082219999
Unsigned 32 bit: 1082219999
Float 32 bit: 4.042709
Float 64 bit: 555.733968189069
☐ Show unsigned as hexadecimal

Hexadecimal: 40 81 5D DF
Decimal: 064 129 093 223
Octal: 100 201 135 337
Binary: 01000000 10000001 01011101 11011111
ASCII Text: @?]?

Offset: 0x37 / 0x129f   Selection: None   INS



```
[04/05/2019 18:33] Client-Parth@VM:~$ openssl enc -aes-128-ecb -d -in task5.bin -out task5o.txt -K 00112233445566778899aabbccddeeff
[04/05/2019 18:33] Client-Parth@VM:~$ openssl enc -aes-128-cbc -d -in task5_2.bin -out task5_2o.txt -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 18:33] Client-Parth@VM:~$ openssl enc -aes-128-cfb -d -in task5_3.bin -out task5_3o.txt -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 18:34] Client-Parth@VM:~$ openssl enc -aes-128-ofb -d -in task5_4.bin -out task5_4o.txt -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 18:34] Client-Parth@VM:~$ 
```



```
[04/05/2019 18:44] Client-Parth@VM:~$ diff plaintext.txt task5o.txt
1c1
< THE OSCARS TURN  ON SUNDAY WHICH SEEMS ABOUT RIGHT AFTER THIS LONG STRANGE
---
> THE OSCARS TURN  ON SUNDAY WHICH SEEMS ABOUT RIG▯��▯�7�"b��NG STRANGE
[04/05/2019 18:45] Client-Parth@VM:~$
[04/05/2019 18:45] Client-Parth@VM:~$
[04/05/2019 18:45] Client-Parth@VM:~$ diff plaintext.txt task5_2o.txt
1c1
< THE OSCARS TURN  ON SUNDAY WHICH SEEMS ABOUT RIGHT AFTER THIS LONG STRANGE
---
> THE OSCARS TURN  ON SUNDAY WHICH SEEMS ABOUT RIGQ�▯?D�▯|es�NG STRAeGE
[04/05/2019 18:45] Client-Parth@VM:~$
[04/05/2019 18:45] Client-Parth@VM:~$
[04/05/2019 18:45] Client-Parth@VM:~$ diff plaintext.txt task5_3o.txt
1,2c1
< THE OSCARS TURN  ON SUNDAY WHICH SEEMS ABOUT RIGHT AFTER THIS LONG STRANGE
< AWARDS TRIP THE BAGGER FEELS LIKE A NONAGENARIAN TOO
---
> THE OSCARS TURN  ON SUNDAY WHICH SEEMS ABOUT RIGHT AFTE
                                                          THIS LO�H�#▯>| r˝���AS TRIP THE BAGGER FEELS LIKE A NONAGENARIAN TOO
[04/05/2019 18:45] Client-Parth@VM:~$
[04/05/2019 18:45] Client-Parth@VM:~$
[04/05/2019 18:45] Client-Parth@VM:~$ diff plaintext.txt task5_4o.txt
1c1
< THE OSCARS TURN  ON SUNDAY WHICH SEEMS ABOUT RIGHT AFTER THIS LONG STRANGE
---
> THE OSCARS TURN  ON SUNDAY WHICH SEEMS ABOUT RIGHT AFTE_ THIS LONG STRANGE
[04/05/2019 18:45] Client-Parth@VM:~$
```

Before conducting the task I assumed that for cfb that particular byte and the next byte will be changed. For ofb that particular byte might have been changed. And for ecb there would be no changes.

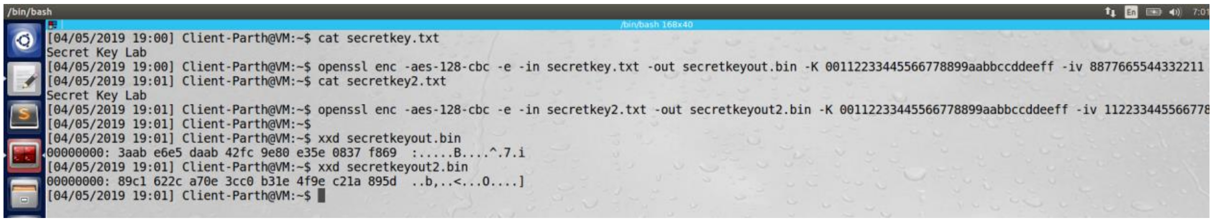
The result about how much can be recovered from decrypted file can be seen from the screenshot above.

We first encrypted the same file using different encryption and then we corrupted the 55th byte with '@' . We can see from the above screenshot that after corrupting the aes-ofb, it does not corrupt anything. The rest all were corrupted.

Task 6

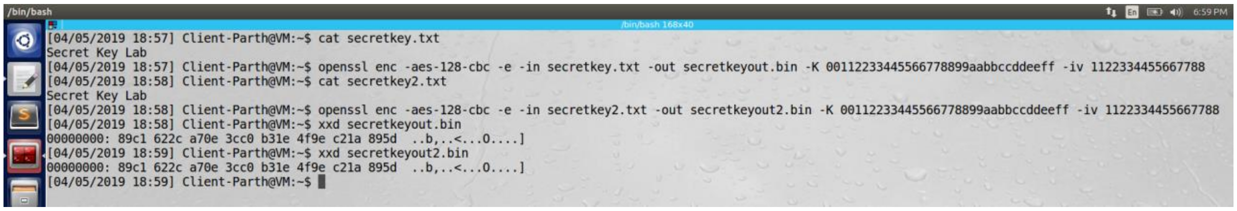
6.1

1.



```
[04/05/2019 19:00] Client-Parth@VM:~$ cat secretkey.txt
Secret Key Lab
[04/05/2019 19:00] Client-Parth@VM:~$ openssl enc -aes-128-cbc -e -in secretkey.txt -out secretkeyout.bin -K 00112233445566778899aabbccddeeff -iv 8877665544332211
[04/05/2019 19:01] Client-Parth@VM:~$ cat secretkey2.txt
Secret Key Lab
[04/05/2019 19:01] Client-Parth@VM:~$ openssl enc -aes-128-cbc -e -in secretkey2.txt -out secretkeyout2.bin -K 00112233445566778899aabbccddeeff -iv 112233445566778
[04/05/2019 19:01] Client-Parth@VM:~$
[04/05/2019 19:01] Client-Parth@VM:~$ xxd secretkeyout.bin
00000000: 3aab e6e5 daab 42fc 9e80 e35e 0837 f869  :.....B....^.7.i
[04/05/2019 19:01] Client-Parth@VM:~$ xxd secretkeyout2.bin
00000000: 89c1 622c a70e 3cc0 b31e 4f9e c21a 895d  ..b,..<...O....]
[04/05/2019 19:01] Client-Parth@VM:~$
```

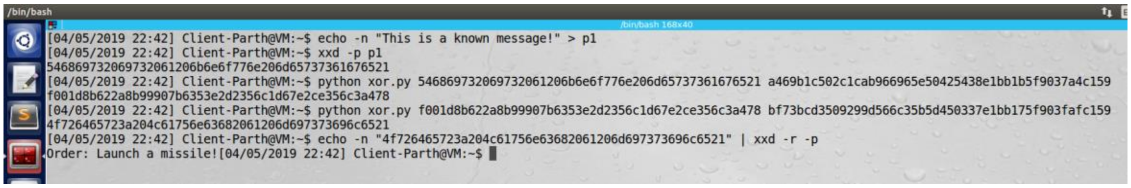
2.



```
[04/05/2019 18:57] Client-Parth@VM:~$ cat secretkey.txt
Secret Key Lab
[04/05/2019 18:57] Client-Parth@VM:~$ openssl enc -aes-128-cbc -e -in secretkey.txt -out secretkeyout.bin -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 18:58] Client-Parth@VM:~$ cat secretkey2.txt
Secret Key Lab
[04/05/2019 18:58] Client-Parth@VM:~$ openssl enc -aes-128-cbc -e -in secretkey2.txt -out secretkeyout2.bin -K 00112233445566778899aabbccddeeff -iv 1122334455667788
[04/05/2019 18:58] Client-Parth@VM:~$ xxd secretkeyout.bin
00000000: 89c1 622c a70e 3cc0 b31e 4f9e c21a 895d  ..b,..<...O....]
[04/05/2019 18:59] Client-Parth@VM:~$ xxd secretkeyout2.bin
00000000: 89c1 622c a70e 3cc0 b31e 4f9e c21a 895d  ..b,..<...O....]
[04/05/2019 18:59] Client-Parth@VM:~$
```

We can see from the 2 screenshot above that if we use different IV for the same plaintext then we get different ciphertext but if we use the same IV for 2 similar plaintext we will get the same ciphertext which is a weakness.

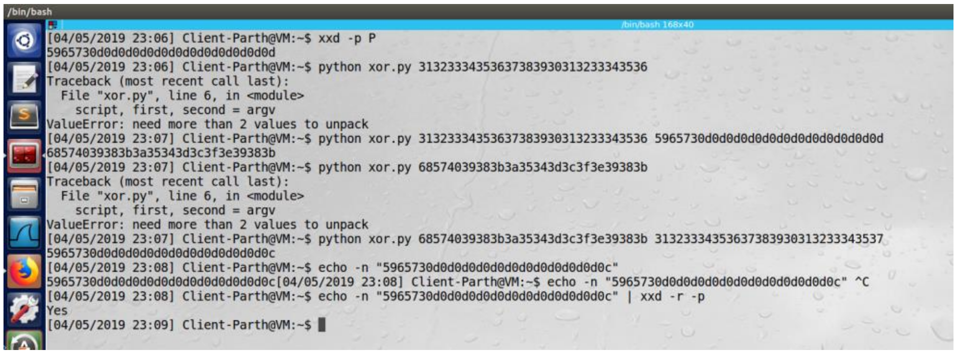
6.2



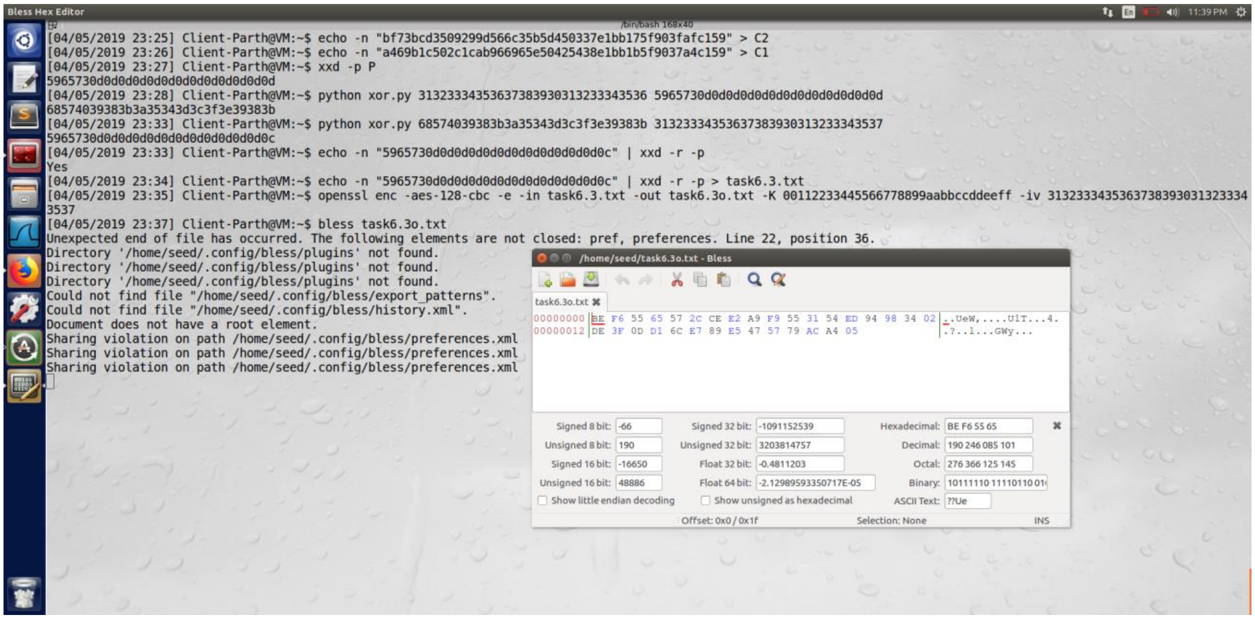
```
[04/05/2019 22:42] Client-Parth@VM:~$ echo -n "This is a known message!" > p1
[04/05/2019 22:42] Client-Parth@VM:~$ xxd -p p1
546869732069732061206b6e6f776e206d65737361676521
[04/05/2019 22:42] Client-Parth@VM:~$ python xor.py 546869732069732061206b6e6f776e206d65737361676521 a469b1c502c1cab966965e50425438e1bb1b5f9037a4c159
f001d8b622a8b99907b6353e2d2356c1d67e2ce356c3a478
[04/05/2019 22:42] Client-Parth@VM:~$ python xor.py f001d8b622a8b99907b6353e2d2356c1d67e2ce356c3a478 bf73bcd3509299d566c35b5d450337e1bb175f903fafc159
4f726465723a204c61756e63682061206d697373696c6521
[04/05/2019 22:42] Client-Parth@VM:~$ echo -n "4f726465723a204c61756e63682061206d697373696c6521" | xxd -r -p
Order: Launch a missile![04/05/2019 22:42] Client-Parth@VM:~$ █
```

In this task we first save the "This is a known message!" to p1. We then convert that to hex using command 'xxd'. After which we xor the hex of p1 with c1. Whatever result we get we xor that with c2. Now, the result that we got is in hex. From the above screenshot we can see that we converted that hex value to ascii and the output we got is "Order : Launch a missile".

6.3



```
[04/05/2019 23:06] Client-Parth@VM:~$ xxd -p P
5965730d0d0d0d0d0d0d0d0d0d0d0d0d
[04/05/2019 23:06] Client-Parth@VM:~$ python xor.py 31323334353637383930313233343536
Traceback (most recent call last):
  File "xor.py", line 6, in <module>
    script, first, second = argv
ValueError: need more than 2 values to unpack
[04/05/2019 23:07] Client-Parth@VM:~$ python xor.py 31323334353637383930313233343536 5965730d0d0d0d0d0d0d0d0d0d0d0d0d
68574039383b3a35343d3c3f3e39383b
[04/05/2019 23:07] Client-Parth@VM:~$ python xor.py 68574039383b3a35343d3c3f3e39383b
Traceback (most recent call last):
  File "xor.py", line 6, in <module>
    script, first, second = argv
ValueError: need more than 2 values to unpack
[04/05/2019 23:07] Client-Parth@VM:~$ python xor.py 68574039383b3a35343d3c3f3e39383b 31323334353637383930313233343537
5965730d0d0d0d0d0d0d0d0d0d0d0d0c
[04/05/2019 23:08] Client-Parth@VM:~$ echo -n "5965730d0d0d0d0d0d0d0d0d0d0d0d0c"
5965730d0d0d0d0d0d0d0d0d0d0d0d0c[04/05/2019 23:08] Client-Parth@VM:~$ echo -n "5965730d0d0d0d0d0d0d0d0d0d0d0d0c" ^C
[04/05/2019 23:08] Client-Parth@VM:~$ echo -n "5965730d0d0d0d0d0d0d0d0d0d0d0d0c" | xxd -r -p
Yes
[04/05/2019 23:09] Client-Parth@VM:~$ █
```



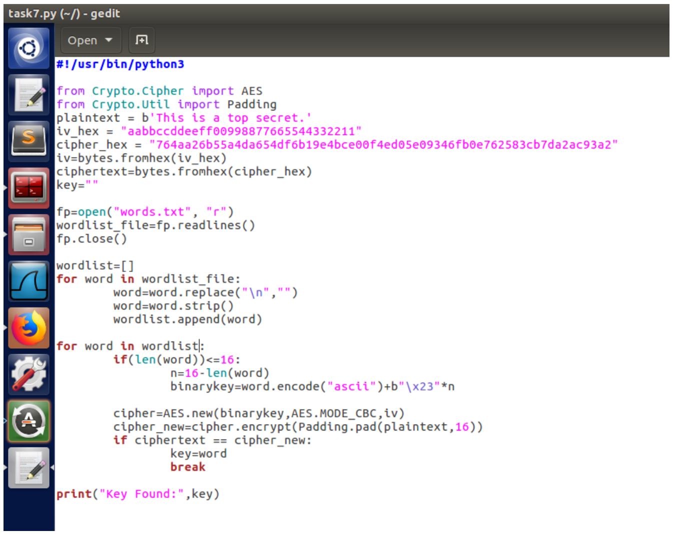
```
[04/05/2019 23:25] Client-Parth@VM:~$ echo -n "bf73bcd3509299d566c35b5d450337e1bb175f903fafc159" > C2
[04/05/2019 23:26] Client-Parth@VM:~$ echo -n "a469b1c502c1cab966965e50425438e1bb1b5f9037a4c159" > C1
[04/05/2019 23:27] Client-Parth@VM:~$ xxd -p P
5965730d0d0d0d0d0d0d0d0d0d0d0d0d
[04/05/2019 23:28] Client-Parth@VM:~$ python xor.py 31323334353637383930313233343536 5965730d0d0d0d0d0d0d0d0d0d0d0d0d
68574039383b3a35343d3c3f3e39383b
[04/05/2019 23:33] Client-Parth@VM:~$ python xor.py 68574039383b3a35343d3c3f3e39383b 31323334353637383930313233343537
5965730d0d0d0d0d0d0d0d0d0d0d0d0c
[04/05/2019 23:33] Client-Parth@VM:~$ echo -n "5965730d0d0d0d0d0d0d0d0d0d0d0d0c" | xxd -r -p
Yes
[04/05/2019 23:34] Client-Parth@VM:~$ echo -n "5965730d0d0d0d0d0d0d0d0d0d0d0d0c" | xxd -r -p > task6.3.txt
[04/05/2019 23:35] Client-Parth@VM:~$ openssl enc -aes-128-cbc -e -in task6.3.txt -out task6.3o.txt -K 00112233445566778899aabbccddeeff -iv 3132333435363738393031323334
3537
[04/05/2019 23:37] Client-Parth@VM:~$ bless task6.3o.txt
Unexpected end of file has occurred. The following elements are not closed: pref, preferences. Line 22, position 36.
Directory '/home/seed/.config/bless/plugins' not found.
Directory '/home/seed/.config/bless/plugins' not found.
Directory '/home/seed/.config/bless/plugins' not found.
Could not find file "/home/seed/.config/bless/export_patterns".
Could not find file "/home/seed/.config/bless/history.xml".
Document does not have a root element.
Sharing violation on path /home/seed/.config/bless/preferences.xml
Sharing violation on path /home/seed/.config/bless/preferences.xml
Sharing violation on path /home/seed/.config/bless/preferences.xml
```

```
00000000 BE F6 55 65 57 2C CE E2 A9 F9 55 31 54 ED 94 98 34 02 ..UeW,....U1T...4.
00000012 DE 3F 0D D1 6C E7 89 E5 47 57 79 AC A4 05             .?..l...GWy...
```

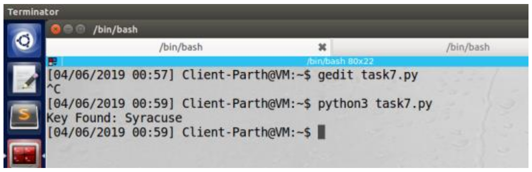
Signed 8 bit: -66
Unsigned 8 bit: 190
Signed 16 bit: -16650
Unsigned 16 bit: 48886
Signed 32 bit: -1091152539
Unsigned 32 bit: 3203814757
Float 32 bit: -0.4811203
Float 64 bit: -2.12989593350717E-05
Hexadecimal: BE F6 55 65
Decimal: 190 246 085 101
Octal: 276 366 125 145
Binary: 10111110 11110110 01
ASCII Text: ??Ue
☐ Show little endian decoding
☐ Show unsigned as hexadecimal
Offset: 0x0 / 0x1f
Selection: None
INS

In this task we created a file P. Converted the content to hex and then xored that with IV (in hex) . The result of that we xored with the next IV (in hex). And then we convert the hex value into ascii as seen from the above screen shot. Thus we can conclude that IV's cannot be similar or predictable.

Task 7



```
#!/usr/bin/python3

from Crypto.Cipher import AES
from Crypto.Util import Padding
plaintext = b'This is a top secret.'
iv_hex = "aabbccddeeff00998877665544332211"
cipher_hex = "764aa26b55a4da654df6b19e4bce00f4ed05e09346fb0e762583cb7da2ac93a2"
iv=bytes.fromhex(iv_hex)
ciphertext=bytes.fromhex(cipher_hex)
key=""

fp=open("words.txt", "r")
wordlist_file=fp.readlines()
fp.close()

wordlist=[]
for word in wordlist_file:
        word=word.replace("\n","")
        word=word.strip()
        wordlist.append(word)

for word in wordlist:
        if(len(word))<=16:
                n=16-len(word)
                binarykey=word.encode("ascii")+b"\x23"*n

        cipher=AES.new(binarykey,AES.MODE_CBC,iv)
        cipher_new=cipher.encrypt(Padding.pad(plaintext,16))
        if ciphertext == cipher_new:
                key=word
                break

print("Key Found:",key)
```



```
[04/06/2019 00:57] Client-Parth@VM:~$ gedit task7.py
^C
[04/06/2019 00:59] Client-Parth@VM:~$ python3 task7.py
Key Found: Syracuse
[04/06/2019 00:59] Client-Parth@VM:~$ 
```

In this task we are given the ciphertext, plaintext and IV. Using the dictionary wordlist the key found was: Syracuse.