

Benjamin Haedt

Lab 8 – SKE

CSCI 476 – Comp security

2 April 2023

### Task 1

```
[04/02/23] seed@VM:~/.../08_ske$ ls  
ciphertext.txt  pic_original.bmp  words.txt  
freq_counter.py  sample_xor.py  
[04/02/23] seed@VM:~/.../08_ske$ tr 'aet' 'XGE' < ciphertext.txt > out.txt  
[04/02/23] seed@VM:~/.../08_ske$
```

I ran the freq\_counter.py on the ciphertext.txt file.

```
[4]+  Stopped                 sudo python3 freq_counter.py ciphertext.txt  
[04/02/23] seed@VM:~/.../08_ske$ sudo python3 freq_counter.py < ciphertext.txt  
  
letter frequencies (all):  
y : 81 (13.28%)  
l : 52 (8.52%)  
t : 50 (8.20%)  
f : 50 (8.20%)  
x : 46 (7.54%)  
e : 42 (6.89%)  
n : 37 (6.07%)  
w : 36 (6.00%)
```

I used frequency analysis to make predictions about what each letter would be. I determined that the first letter, y, which occurred the most, would be e. Once I set y = e I guessed that the most common trigram tzy could be THE, so I did that. At each step I kind of guessed more and more, with each step I could see the next piece of the puzzle and continue making words and assumptions based on sentence structure and basic English language knowledge (that was some good writing though! Haha). One of these, I was able to guess fe = is, since it appeared in a certain spot in the sentence that made it seemed like that would be a good fit and work with the context. I can't go back through all the steps I did exactly how I did it now. I did record each step down below and also took a screenshot of the ciphered text all uncyphered. I did have to back track on a few letters early on, but eventually went in the right direction. I originally thought tzy was 'are', but nope. Weeeee! That was fun.

Y = e

I guessed that the first word could be the, so that makes tzy = THE

I guess l = a, fe = is, v = w, x = n, m = d, g = c, c = p, qs = of, w = r, u = u, a = g, b = m, r = b, n = l, h = v, l = x, p = y, k = k

```
writing csv data to: 3grams.csv
...done!
[04/02/23]seed@VM:~/.../08_ske$ tr 'y' 'E' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'y' 'E' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tz' 'ar' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzy' 'are' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzye' 'AREN' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzyel' 'AREN'T < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzyelw' 'ARENTH' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzyelwlxm' 'ARENTHING' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzyelwlxm' 'THE' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzy' 'THE' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzyl' 'THEA' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylc' 'THEAB' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylcx' 'THEABR' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylc' 'THEAB' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylcf' 'THEABIS' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylfe' 'THEAIS' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylfev' 'THEAISW' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylfevxm' 'THEAISWN' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylfevxmg' 'THEAISWNDC' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylfevxmgc' 'THEAISWNDCP' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylfevxmgcq' 'THEAISWNDCPOF' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylfevxmgcqsw' 'THEAISWNDCPOFR' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylfevxmgcqswua' 'THEAISWNDCPOFRug' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylfevxmgcqswua' 'THEAISWNDCPOFRug' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylfevxmgcqswuab' 'THEAISWNDCPOFRUGM' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylfevxmgcqswuab' 'THEAISWNDCPOFRUGMB' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylfevxmgcqswuabrn' 'THEAISWNDCPOFRUGMBL' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylfevxmgcqswuabrn' 'THEAISWNDCPOFRUGMBLV' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylfevxmgcqswuabrnhip' 'THEAISWNDCPOFRUGMBLVXY' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$ tr 'tzylfevxmgcqswuabrnhip' 'THEAISWNDCPOFRUGMBLVXYK' < ciphertext.txt > out.txt
[04/02/23]seed@VM:~/.../08_ske$
```

## Task 2

I made a text file to encrypt.

```
File to Encrypt.txt
~/code/08_ske
```

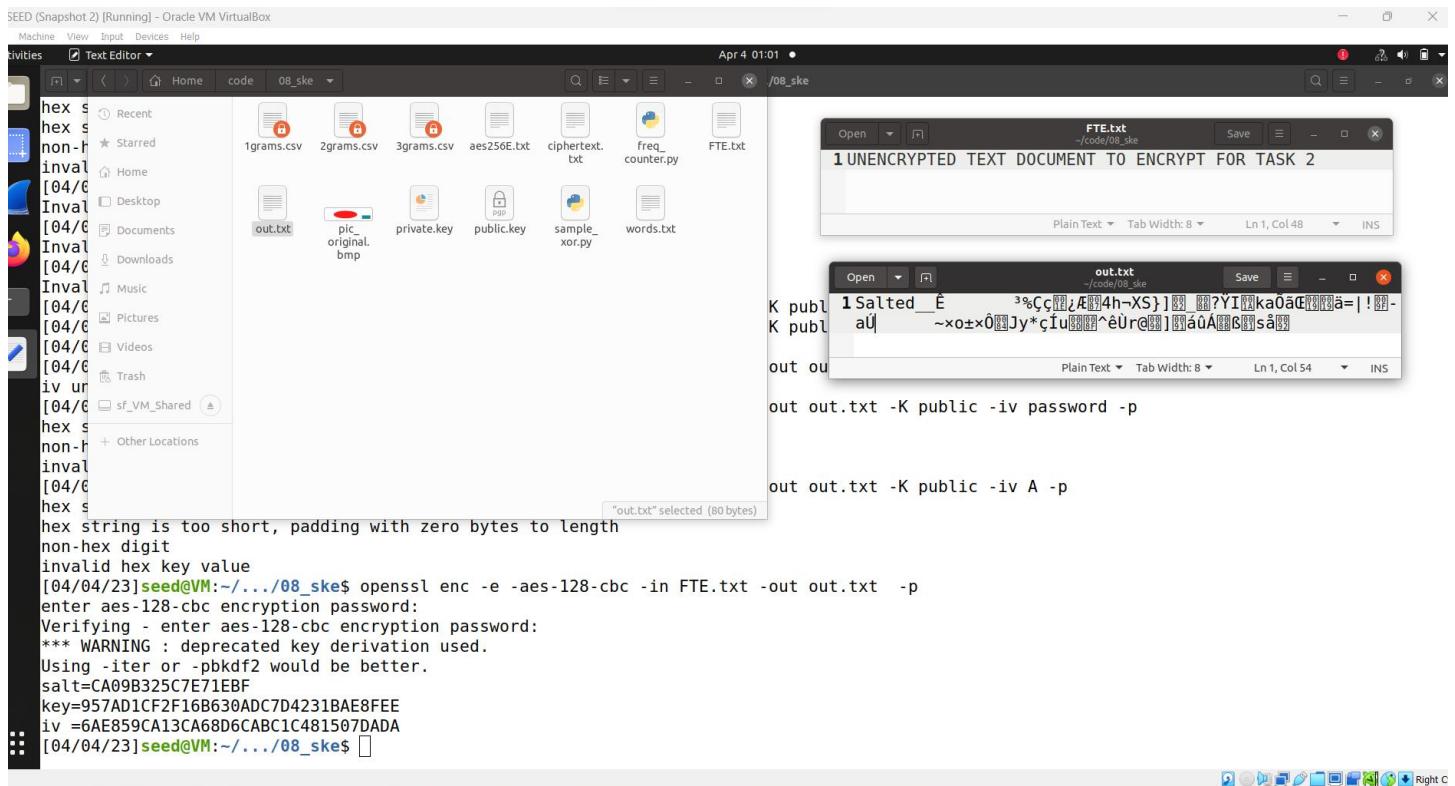
1 UNENCRYPTED TEXT DOCUMENT TO ENCRYPT FOR TASK 2

```
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -aes-256-cbc -in FTE.txt -out aes256E.txt
enter aes-256-cbc encryption password:
Verifying - enter aes-256-cbc encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
```

I encrypted FTE.txt using aes256 with basic options and used the password 'password' and my key.

I think the requirements for this task is to create a private and public key, so I did so below and generated both a private and public key.

```
[04/04/23]seed@VM:~/.../08_ske$ openssl genrsa -aes256 -passout pas
s:password -out private.key 2048
Generating RSA private key, 2048 bit long modulus (2 primes)
.....+++++
.....+++++
e is 65537 (0x010001)
[04/04/23]seed@VM:~/.../08_ske$ openssl rsa -in private.key -passin
pass:password -pubout -out public.key
writing RSA key
[04/04/23]seed@VM:~/.../08_ske$
```



Using the public key, I encrypted the text file FTE.txt.

Using the private key I decrypted the text file.

Next I created new keys using -aes-128-cfb, then encrypted a file. Then decrypted the encrypted txt file out.txt

Next I attempted to decrypt a file that I encrypted using the private key with the public key

And it doesn't decrypt the encrypted file as it shouldn't.

For the last encryption, I used -aes-128-ofb for my cipher type, I don't know why but it says I need to enter a password when I use a public key to encrypt a file, then that password is the same one used to decrypt the encrypted txt.

Then I encrypted FTE.txt with the public key and then decrypted using the private key.

Machine View Input Devices Help

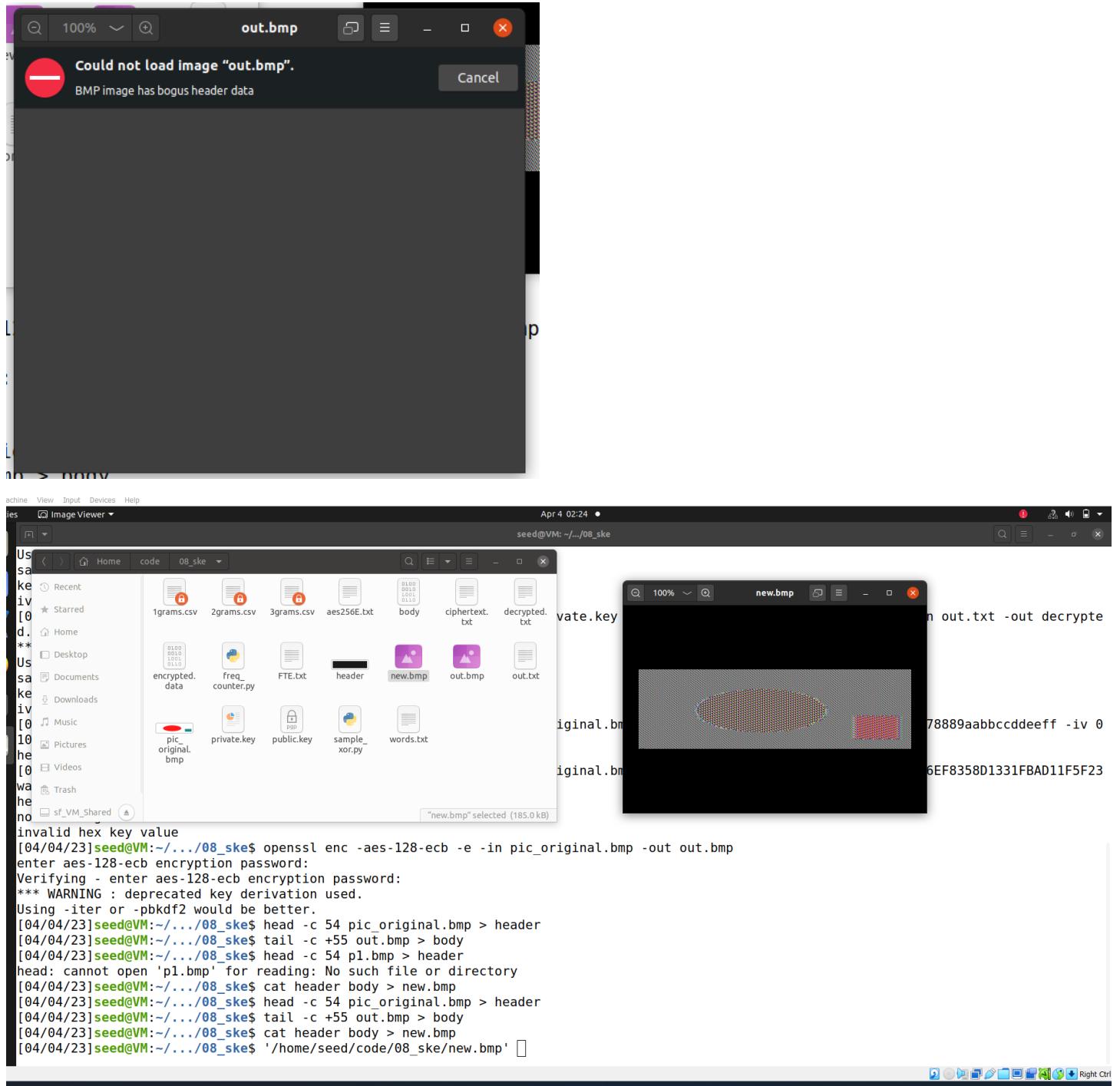
Activities Files Home code 08\_ske

04/04/23 seed@VM:~/08\_ske\$ iv = f  
[04/04/23] seed@VM:~/08\_ske\$ enter  
Verif  
\*\*\* W  
Using  
[04/04/23] seed@VM:~/08\_ske\$ enter  
\*\*\* W  
Using  
[04/04/23] seed@VM:~/08\_ske\$ enter  
Gener  
...+  
e is  
[04/04/23] seed@VM:~/08\_ske\$ ls  
1grams.csv 2grams.csv 3grams.csv aes256E.txt cipherhex.txt decrypted.txt freq\_counter.py  
FTE.txt out.txt pic\_original.bmp private.key public.key sample\_xor.py words.txt  
"decrypted.txt" selected (48 bytes)  
[04/04/23] seed@VM:~/08\_ske\$ openssl enc -e -aes-128-cfb -in FTE.txt -out out.txt  
enter aes-128-cfb encryption password:  
Verifying - enter aes-128-cfb encryption password:  
\*\*\* WARNING : deprecated key derivation used.  
Using -iter or -pbkdf2 would be better.  
[04/04/23] seed@VM:~/08\_ske\$ openssl enc -d -aes-128-cfb -in out.txt -out decrypted.txt  
enter aes-128-cfb decryption password:  
\*\*\* WARNING : deprecated key derivation used.  
Using -iter or -pbkdf2 would be better.  
[04/04/23] seed@VM:~/08\_ske\$ openssl genrsa -aes-128-ofb -out private.key 2048  
Generating RSA private key, 2048 bit long modulus (2 primes)  
.....++++  
e is 65537 (0x010001)  
Enter pass phrase for private.key:  
Verifying - Enter pass phrase for private.key:  
[04/04/23] seed@VM:~/08\_ske\$

Hopefully this fulfills Task 2 requirements. I could continue using different ciphertexts and generating new keys, but this proves that I can use public and private keys to encrypt and decrypt, and also just use a simple password to encrypt a file.

### Task 3

Encrypted the image with ecb, cannot open the output from encrypting it due to header information also being encrypted. So I followed the directions to get the right header information and then applied it to a image named out.bmp. It works.



Yes, when we use ecb to encrypt the image, we can still clearly see what the image is, a oval and a rectangle.

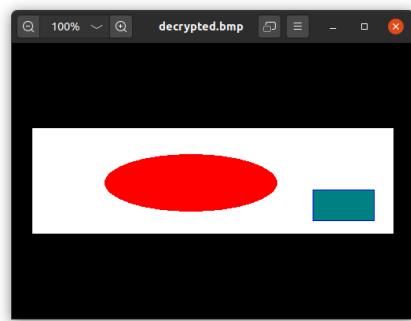
Next I used cbc to encrypt the image and then applied the header information to that picture. This is clearly a better encryption option for images, since it completely distorts the image.

seed@VM: ~/08\_ske

```
[04/04/23]seed@VM:~/08_ske$ cat header body > new.bmp
[04/04/23]seed@VM:~/08_ske$ openssl enc -aes-128-cbc -e -in pic_original.bmp -out out.bmp
enter aes-128-cbc encryption password:
Verifying - enter aes-128-cbc encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/04/23]seed@VM:~/08_ske$ openssl enc -aes-128-cbc -e -in pic_original.bmp -out out.bmp
enter aes-128-cbc encryption password:
Verifying - enter aes-128-cbc encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/04/23]seed@VM:~/08_ske$ head -c 54 pic_original.bmp > header
[04/04/23]seed@VM:~/08_ske$ tail -c +55 out.bmp > body
[04/04/23]seed@VM:~/08_ske$ cat header body > new.bmp
[04/04/23]seed@VM:~/08_ske$ 
```

Also decrypted the image that was encrypted using cbc

```
** WARNING : deprecated key derivation used.
sing -iter or -pbkdf2 would be better.
04/04/23]seed@VM:~/08_ske$ openssl enc -aes-128-cbc -e -in pic_original.bmp -out out.bmp
nter aes-128-cbc encryption password:
erifying - enter aes-128-cbc encryption password:
** WARNING : deprecated key derivation used.
sing -iter or -pbkdf2 would be better.
04/04/23]seed@VM:~/08_ske$ head -c 54 pic_original.bmp > header
04/04/23]seed@VM:~/08_ske$ tail -c +55 out.bmp > body
04/04/23]seed@VM:~/08_ske$ cat header body > new.bmp
04/04/23]seed@VM:~/08_ske$ openssl enc -aes-128-cbc -d -in out.bmp -out decrypted.bmp
nter aes-128-cbc decryption password:
** WARNING : deprecated key derivation used.
sing -iter or -pbkdf2 would be better.
04/04/23]seed@VM:~/08_ske$ 
```



### Task 3.2

For some reason, I have to use my own image and do the exact same thing I did for the previous image, so here we go...

Here is custom image using cbc to encrypt

```

[04/04/23]seed@VM:~/.../08_ske$ head -c 54 pic_original
[04/04/23]seed@VM:~/.../08_ske$ tail -c +55 out.bmp > body
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -aes-128-cbc -e -in custom.bmp -out out.bmp
enter aes-128-cbc encryption password:
Verifying - enter aes-128-cbc encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -aes-128-cbc -e -in custom.bmp -out out.bmp
enter aes-128-cbc encryption password:
Verifying - enter aes-128-cbc encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/04/23]seed@VM:~/.../08_ske$ head -c 54 custom.bmp > header
[04/04/23]seed@VM:~/.../08_ske$ tail -c +55 out.bmp > body
[04/04/23]seed@VM:~/.../08_ske$ cat header body > new.bmp
[04/04/23]seed@VM:~/.../08_ske$ 

```

And here is the image using ECB

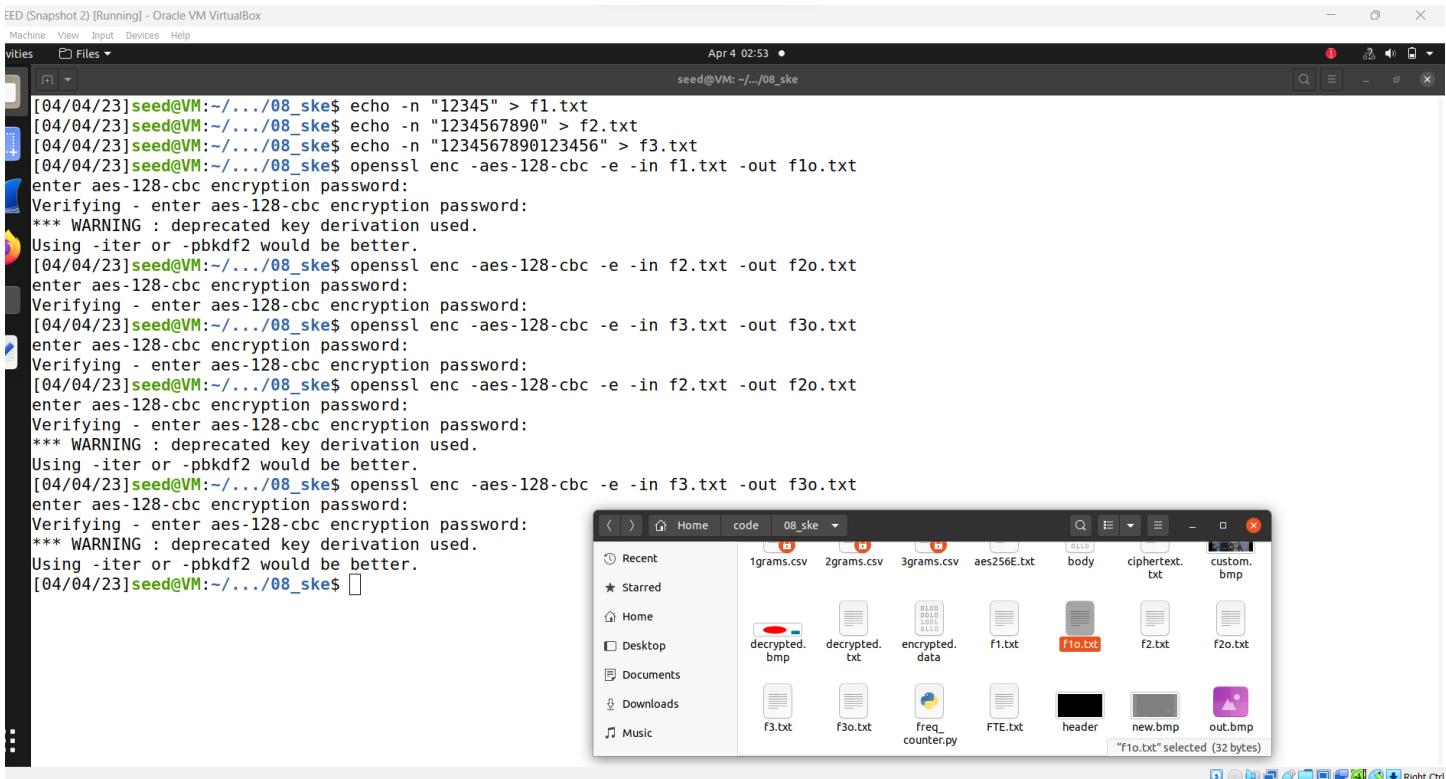
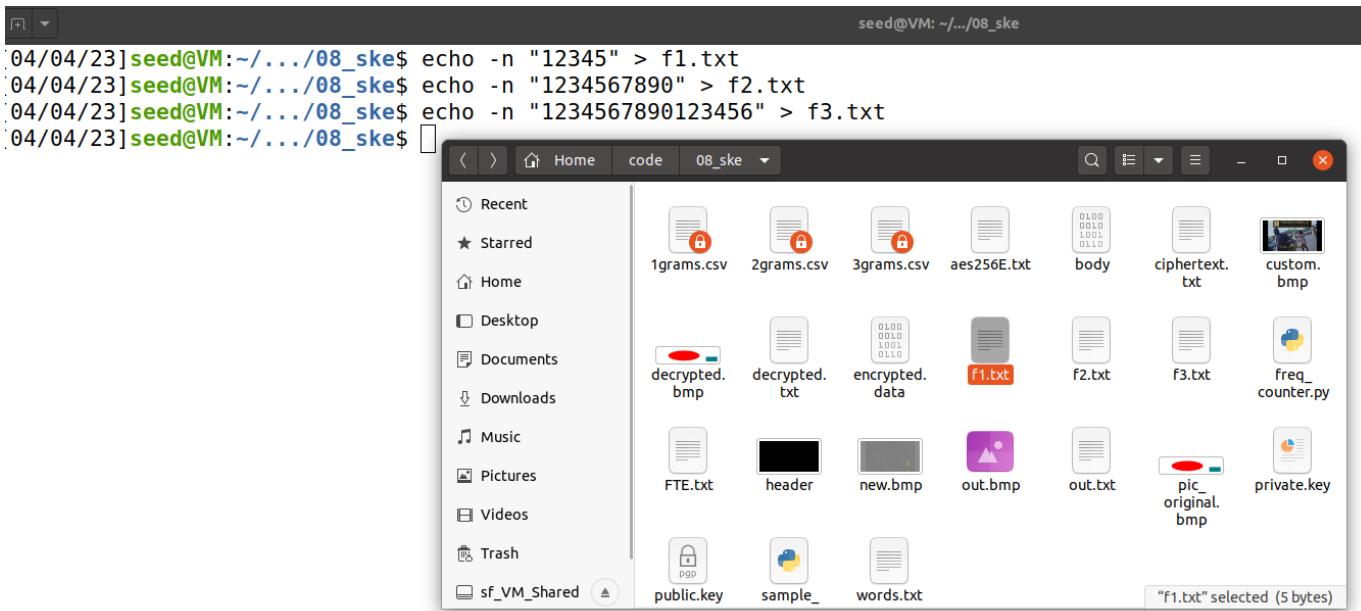
```

[04/04/23]seed@VM:~/.../08_ske$ head -c 54 pic_original
[04/04/23]seed@VM:~/.../08_ske$ tail -c +55 out.bmp > body
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -aes-128-ecb -e -in custom.bmp -out out.bmp
enter aes-128-ecb encryption password:
Verifying - enter aes-128-ecb encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/04/23]seed@VM:~/.../08_ske$ head -c 54 custom.bmp > header
[04/04/23]seed@VM:~/.../08_ske$ tail -c +55 out.bmp > body
[04/04/23]seed@VM:~/.../08_ske$ cat header body > new.bmp
[04/04/23]seed@VM:~/.../08_ske$ 

```

The ECB encryption on this file worked a lot better than the picture provided.

## Task 4.1



The size of the encrypted files increased. Since the first two were under 16 bytes, when encrypted they had enough room once encrypted and padded to increase the size to 16 bytes , but once we encrypted the 16 byte file, it needed more room to encrypt and pad so it now is 48 bytes long.

```
[04/04/23]seed@VM:~/.../08_ske$ hexdump -C f1.txt |12345|
00000000 31 32 33 34 35
00000005
[04/04/23]seed@VM:~/.../08_ske$ hexdump -C f1o.txt |Salted.....u*...|
00000000 53 61 6c 74 65 64 5f 5f f3 e0 9d 85 75 2a ee a8 |Qpe]....c)duP..e|
00000010 51 70 65 5d a1 c3 dc 05 63 29 64 75 50 ec 0c 65
00000020
[04/04/23]seed@VM:~/.../08_ske$ xxd f1.txt 12345
00000000: 3132 3334 35
[04/04/23]seed@VM:~/.../08_ske$ hexdump -C f2.txt
00000000 31 32 33 34 35 36 37 38 39 30 |1234567890|
0000000a
[04/04/23]seed@VM:~/.../08_ske$ hexdump -C f2o.txt |Salted.....|
00000000 53 61 6c 74 65 64 5f 5f 06 1d 92 10 07 1c d5 d3 |...u.=....!.I...8|
00000010 da b8 75 f5 3d ab a6 e4 bf 21 0e 49 d8 88 e0 38
00000020
[04/04/23]seed@VM:~/.../08_ske$ xxd f2.txt
00000000: 3132 3334 3536 3738 3930 1234567890
[04/04/23]seed@VM:~/.../08_ske$ xxd f2o.txt
00000000: 5361 6c74 6564 5f5f 061d 9210 071c d5d3 Salted.....
00000010: dab8 75f5 3dab a6e4 bf21 0e49 d888 e038 ...u.=....!.I...8
[04/04/23]seed@VM:~/.../08_ske$ hexdump -C f3.txt |1234567890123456|
00000000 31 32 33 34 35 36 37 38 39 30 31 32 33 34 35 36 |1234567890123456|
00000010
[04/04/23]seed@VM:~/.../08_ske$ hexdump -C f3o.txt |Salted_Cv.0.P.b|
00000000 53 61 6c 74 65 64 5f 5f 43 76 a6 30 a2 50 18 62 |...?D..@.N!7....|
00000010 36 13 0d 3f 44 05 89 40 ac 4e 21 37 d2 0f 7f e4 |x.n.w.XJ.e2.!..C|
00000020 78 f0 6e ed 77 ae 58 4a cd 65 32 9f 21 b7 87 43
00000030
[04/04/23]seed@VM:~/.../08_ske$ xxd f3.txt
00000000: 3132 3334 3536 3738 3930 3132 3334 3536 1234567890123456
[04/04/23]seed@VM:~/.../08_ske$ xxd f3o.txt |Salted_Cv.0.P.b|
00000000: 5361 6c74 6564 5f5f 4376 a630 a250 1862 |...?D..@.N!7....|
00000010: 3613 0d3f 4405 8940 ac4e 2137 d20f 7fe4 |x.n.w.XJ.e2.!..C|
00000020: 78f0 6eed 77ae 584a cd65 329f 21b7 8743
[04/04/23]seed@VM:~/.../08_ske$ █
```

```
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-cbc -d -in flo.txt -out ff1.txt
enter aes-128-cbc decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/04/23]seed@VM:~/.../08_ske$ hexdump ff1.txt
00000000 3231 3433 0b35 0b0b 0b0b 0b0b 0b0b
0000010
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-cbc -d -in f2o.txt -out ff2.txt
enter aes-128-cbc decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/04/23]seed@VM:~/.../08_ske$ hexdump ff2.txt
00000000 3231 3433 3635 3837 3039 0606 0606
0000010
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-cbc -d -in f3o.txt -out ff3.txt
enter aes-128-cbc decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/04/23]seed@VM:~/.../08_ske$ hexdump ff3.txt
00000000 17e5 505c 783b 1134 60d6 9a2f 4e04 1b49
00000010 9758 e6d6 c1a0 3d84 2ed1 f3c0 5de8 1976
0000020
[04/04/23]seed@VM:~/.../08_ske$
```

Here we can see the padded values that were added to each txt file as it was encrypted.

The terminal window shows the command line operations for encrypting files flo.txt, f2o.txt, and f3o.txt using AES-128-CBC mode with no padding. The hexedit windows show the resulting files ff1.txt, ff2.txt, and ff3.txt. Each file contains the original data followed by a sequence of FF bytes, indicating padding. The padding length varies: ff1.txt has 10 FF bytes, ff2.txt has 10 FF bytes, and ff3.txt has 20 FF bytes.

```
p: command not found.
[04/04/23]seed@VM:~/.../08_ske$ hexdump ff1.txt
00000000 3231 3433 0b35 0b0b 0b0b 0b0b 0b0b
0000010
[04/04/23]seed@VM:~/.../08_ske$ hexdump ff2.txt
00000000 3231 3433 3635 3837 3039 0606 0606
0000010
[04/04/23]seed@VM:~/.../08_ske$ hexdump ff3.txt
00000000 17e5 505c 783b 1134 60d6 9a2f 4e04 1b49
00000010 9758 e6d6 c1a0 3d84 2ed1 f3c0 5de8 1976
0000020
[04/04/23]seed@VM:~/.../08_ske$
```

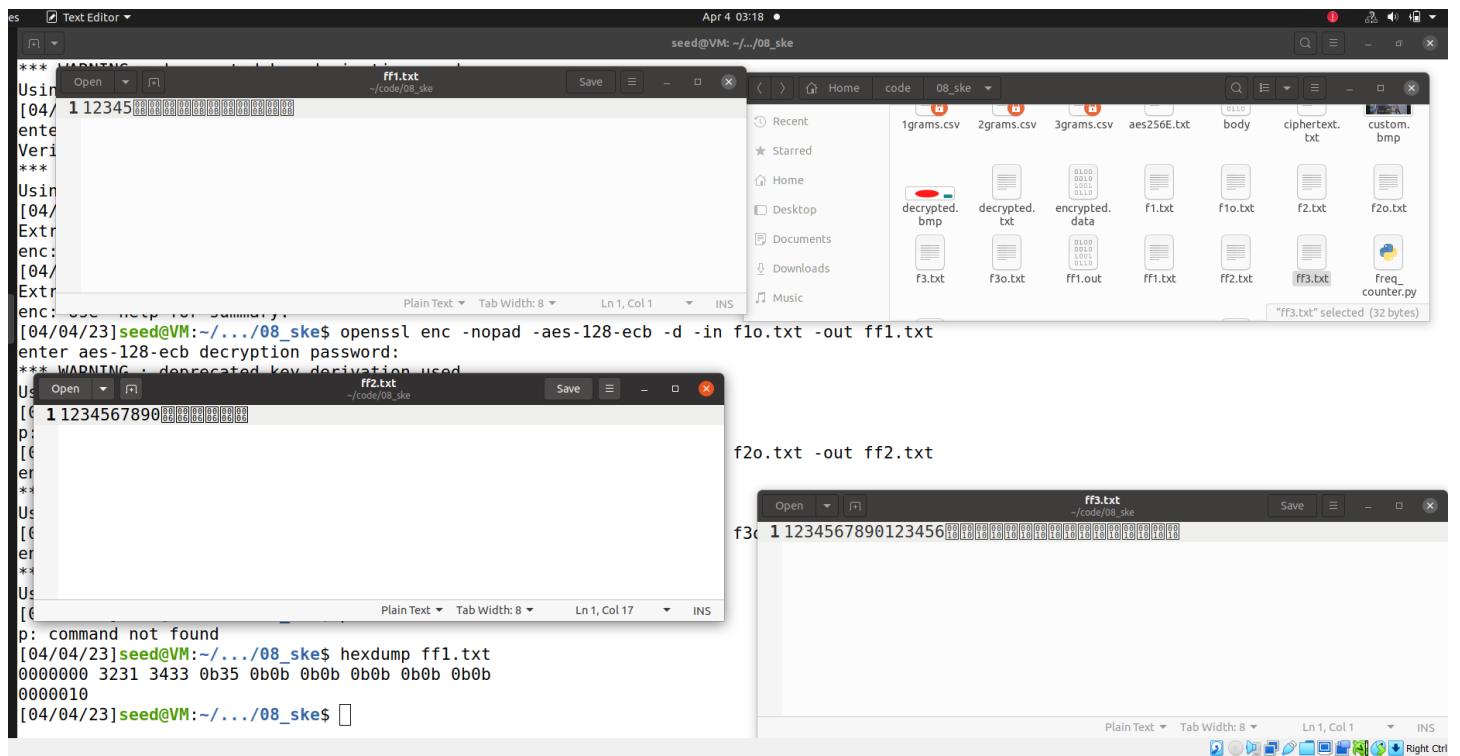
## Task 4.2

**Using ecb to encrypt and decrypt. ECB uses padding.**

```

[04/04/23]seed@VM:~/.../08_ske$ openssl enc -aes-128-ecb -e -in f3.txt -out f3o.txt
enter aes-128-ecb encryption password:
Verifying - enter aes-128-ecb encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -aes-128-ecb -d in flo.txt -out ff1.txt
Extra arguments given.
enc: Use -help for summary.
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-ecb -d in flo.txt -out ff1.txt
Extra arguments given.
enc: Use -help for summary.
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-ecb -d -in flo.txt -out ff1.txt
enter aes-128-ecb decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/04/23]seed@VM:~/.../08_ske$ p
p: command not found
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-ecb -d -in f2o.txt -out ff2.txt
enter aes-128-ecb decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-ecb -d -in f3o.txt -out ff3.txt
enter aes-128-ecb decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/04/23]seed@VM:~/.../08_ske$ p
p: command not found
[04/04/23]seed@VM:~/.../08_ske$ hexdump ff1.txt
00000000 3231 3433 0b35 0b0b 0b0b 0b0b 0b0b
00000010

```



**Using cfb, it does not use padding**

```
seed@VM:~/.../08_ske$ 
00000000 3231 3433 0b35 0b0b 0b0b 0b0b 0b0b 0b0b
0000010
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -aes-128-cfb -e -in f1.txt -out f1o.txt
enter aes-128-cfb encryption password:
Verifying - enter aes-128-cfb encryption password:
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -aes-128-cfb -e -in f2.txt -out f2o.txt
enter aes-128-cfb encryption password:
Verifying - enter aes-128-cfb encryption password:
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -aes-128-cfb -e -in f3.txt -out f3o.txt
enter aes-128-cfb encryption password:
Verifying - enter aes-128-cfb encryption password:
[04/04/23]seed@VM:~/.../08_ske$ hexdump ff1.txt
00000000 3231 3433 0b35 0b0b 0b0b 0b0b 0b0b 0b0b
0000010
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-ecb -d -in f3o.txt -out ff3.txt
enter aes-128-ecb decryption password:
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-ecb -d -in f2o.txt -out ff2.txt
enter aes-128-ecb decryption password:
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-ecb -d -in f1o.txt -out ff1.txt
enter aes-128-ecb decryption password:
[04/04/23]seed@VM:~/.../08_ske$ hexdump ff1.txt
00000000 3231 3433 0b35 0b0b 0b0b 0b0b 0b0b 0b0b
0000010
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-cfb -d -in f3o.txt -out ff3.txt
enter aes-128-cfb decryption password:
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-cfb -d -in f2o.txt -out ff2.txt
enter aes-128-cfb decryption password:
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-cfb -d -in f1o.txt -out ff1.txt
enter aes-128-cfb decryption password:
[04/04/23]seed@VM:~/.../08_ske$ hexdump ff1.txt
00000000 3231 3433 0b35 0b0b 0b0b 0b0b 0b0b 0b0b
0000010
[04/04/23]seed@VM:~/.../08_ske$ 
```

```
seed@VM:~/.../08_ske$ 
00000000 3231 3433 0b35 0b0b 0b0b 0b0b 0b0b 0b0b
0000010
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-ecb -d -in f3o.txt -out ff3.txt
enter aes-128-ecb decryption password:
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-ecb -d -in f2o.txt -out ff2.txt
enter aes-128-ecb decryption password:
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-ecb -d -in f1o.txt -out ff1.txt
enter aes-128-ecb decryption password:
[04/04/23]seed@VM:~/.../08_ske$ hexdump ff1.txt
00000000 3231 3433 0b35 0b0b 0b0b 0b0b 0b0b 0b0b
0000010
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-cfb -d -in f3o.txt -out ff3.txt
enter aes-128-cfb decryption password:
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-cfb -d -in f2o.txt -out ff2.txt
enter aes-128-cfb decryption password:
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-cfb -d -in f1o.txt -out ff1.txt
enter aes-128-cfb decryption password:
[04/04/23]seed@VM:~/.../08_ske$ hexdump ff1.txt
00000000 3231 3433 0b35 0b0b 0b0b 0b0b 0b0b 0b0b
0000010
[04/04/23]seed@VM:~/.../08_ske$ 
```

Using OFB, it does not use padding.

```

[04/04/23] seed@VM:~/.../08_ske$ openssl enc -aes-128-ofb -e -in f1.txt -out ff1.txt
enter aes-128-ofb encryption password:
Verifying - enter aes-128-ofb encryption password:
[04/04/23] seed@VM:~/.../08_ske$ openssl enc -aes-128-ofb -e -in f2.txt -out ff2.txt
enter aes-128-ofb encryption password:
Verifying - enter aes-128-ofb encryption password:
[04/04/23] seed@VM:~/.../08_ske$ openssl enc -aes-128-ofb -e -in f3.txt -out ff3.txt
enter aes-128-ofb encryption password:
Verifying - enter aes-128-ofb encryption password:
[04/04/23] seed@VM:~/.../08_ske$ hexdump ff1.txt
00000000 3231 3433 0b35 0b0b 0b0b 0b0b 0b0b
00000010
[04/04/23] seed@VM:~/.../08_ske$ hexdump ff2.txt
00000000 3231 3433 0b35 0b0b 0b0b 0b0b 0b0b
00000010
[04/04/23] seed@VM:~/.../08_ske$ hexdump ff3.txt
00000000 3231 3433 3635 3837 3039 3231 3433 3635
00000010 1010 1010 1010 1010 1010 1010 1010 1010
00000020
[04/04/23] seed@VM:~/.../08_ske$ hexdump f1.txt
00000000 3231 3433 3635 3837 3039 3231 3433 3635
00000010
[04/04/23] seed@VM:~/.../08_ske$ hexdump f2.txt
00000000 3231 3433 3635 3837 3039 3231 3433 3635
00000010
[04/04/23] seed@VM:~/.../08_ske$ hexdump f3.txt
00000000 3231 3433 3635 3837 3039 3231 3433 3635
00000010
[04/04/23] seed@VM:~/.../08_ske$ 

```

0000010

```

[04/04/23] seed@VM:~/.../08_ske$ openssl enc -aes-128-ofb -e -in f1.txt -out f1o.txt
enter aes-128-ofb encryption password:
Verifying - enter aes-128-ofb encryption password:
[04/04/23] seed@VM:~/.../08_ske$ openssl enc -aes-128-ofb -e -in f2.txt -out f2o.txt
enter aes-128-ofb encryption password:
Verifying - enter aes-128-ofb encryption password:
[04/04/23] seed@VM:~/.../08_ske$ openssl enc -aes-128-ofb -e -in f3.txt -out f3o.txt
enter aes-128-ofb encryption password:
Verifying - enter aes-128-ofb encryption password:
[04/04/23] seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-ofb -d -in f1o.txt -out ff1.txt
enter aes-128-ofb decryption password:
[04/04/23] seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-ofb -d -in f2o.txt -out ff2.txt
enter aes-128-ofb decryption password:
[04/04/23] seed@VM:~/.../08_ske$ openssl enc -nopad -aes-128-ofb -d -in f3o.txt -out ff3.txt
enter aes-128-ofb decryption password:
[04/04/23] seed@VM:~/.../08_ske$ hexdump ff1.txt
00000000 3231 3433 0b35 0b0b 0b0b 0b0b 0b0b
00000010
[04/04/23] seed@VM:~/.../08_ske$ 

```

## OFB doesn't add padding

```

[04/04/23] seed@VM:~/.../08_ske$ openssl enc -aes-128-ofb -e -in f3.txt -out f3o.txt
enter aes-128-ofb encryption password:
Verifying - enter aes-128-ofb encryption password:
[04/04/23] seed@VM:~/.../08_ske$ openssl enc -aes-128-ofb -d -in f3o.txt -out ff3.txt
enter aes-128-ofb decryption password:
[04/04/23] seed@VM:~/.../08_ske$ hexdump ff3.txt
00000000 3231 3433 3635 3837 3039 3231 3433 3635
00000010 1010 1010 1010 1010 1010 1010 1010 1010
00000020
[04/04/23] seed@VM:~/.../08_ske$ hexdump f3.txt
00000000 3231 3433 3635 3837 3039 3231 3433 3635
00000010
[04/04/23] seed@VM:~/.../08_ske$ 

```

## CFB does not add padding

```
0000010
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -aes-128-cfb -d -in f3o.txt -out ff3.txt
enter aes-128-cfb decryption password:
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -aes-128-cfb -e -in f3.txt -out f3o.txt
enter aes-128-cfb encryption password:
Verifying - enter aes-128-cfb encryption password:
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -aes-128-cfb -d -in f3o.txt -out ff3.txt
enter aes-128-cfb decryption password:
[04/04/23]seed@VM:~/.../08_ske$ hexdump f3.txt
00000000 3231 3433 3635 3837 3039 3231 3433 3635
0000010
[04/04/23]seed@VM:~/.../08_ske$ hexdump ff3.txt
00000000 3231 3433 3635 3837 3039 3231 3433 3635
0000010 1010 1010 1010 1010 1010 1010 1010 1010
0000020
[04/04/23]seed@VM:~/.../08_ske$
```

## ECB adds padding

```
0000020
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -aes-128-ecb -e -in f3.txt -out f3o.txt
enter aes-128-ecb encryption password:
Verifying - enter aes-128-ecb encryption password:
[04/04/23]seed@VM:~/.../08_ske$ openssl enc -aes-128-ecb -d -in f3o.txt -out ff3.txt
enter aes-128-ecb decryption password:
[04/04/23]seed@VM:~/.../08_ske$ hexdump ff3.txt
00000000 3231 3433 3635 3837 3039 3231 3433 3635
0000010 1010 1010 1010 1010 1010 1010 1010 1010
0000020
[04/04/23]seed@VM:~/.../08_ske$ hexdump f3.txt
00000000 3231 3433 3635 3837 3039 3231 3433 3635
0000010
[04/04/23]seed@VM:~/.../08_ske$
```

**CFB and OFB do not use padding since it pseudo generates random XOR with the plain text to generate the ciphertext, since we can use as many bits of random streams as we want, when don't need padding for these.**

## Task 5

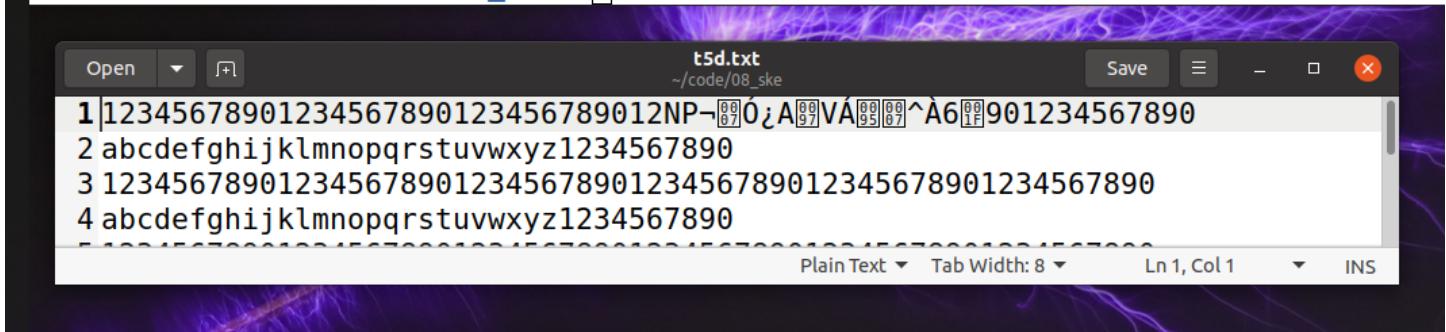
### Task 5.1

How much information can you recover by decrypting the corrupted file, if the encryption mode is ECB, CBC, CFB, or OFB, respectively? I think we can recover all of the information from CFB and OFB since they do not use padding.

### Task 5.2

Using ECB we can still most of the original data but the bit we changed couldn't decrypt properly

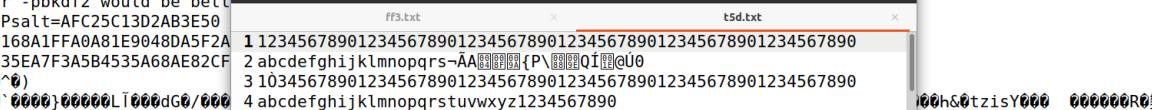
```
[04/11/23]seed@VM:~/.../08_ske$ ghex t5e.txt
[04/11/23]seed@VM:~/.../08_ske$ openssl enc -d -aes-128-cfb -in t5e.txt -out t5d
.txt
enter aes-128-cfb decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/11/23]seed@VM:~/.../08_ske$ openssl enc -e -aes-128-ecb -in task5.txt -out t
5e.txt
enter aes-128-ecb encryption password:
Verifying - enter aes-128-ecb encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/11/23]seed@VM:~/.../08_ske$ ghex t5e.txt
[04/11/23]seed@VM:~/.../08_ske$ openssl enc -d -aes-128-ecb -in t5e.txt -out t5d
.txt
enter aes-128-ecb decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/11/23]seed@VM:~/.../08_ske$ 
```



### Task 5.3

Using CBC, we can still most of the original data but the bit we changed couldn't decrypt properly.

```
enter aes-128-cbc encryption password:  
Verifying - enter aes-128-cbc encryption password:  
*** WARNING : deprecated key derivation  
Using -iter or -pbkdf2 would be better  
Salted_00\c>Psalt=AFAC25C13D2AB3E50  
key=5F2307DF6168A1FFA0A81E9048DA5F2A  
iv=D79AC3EFE35EA7F3A5B4535A68AE82CF  
0F00Z" A;DK^@  
07000400y0000`0000,00000L000dG0/000  
#00J0@M80000L0p6!0000D[]U000>0N00  
030000-000))0)0D00I$#  
I-j000|$,03000  
l7],v$ne0rx0000u089006)0D20020Kv00  
0R\000;00e0000000000/00p00&qm00V00>00  
t000 hwpL0,0,00000Z000cn00000  
E000Y0000000-02,#/0@E<YV  
[04/06/23]seed@VM:~/.../08_ske$ openssl enc -e -aes-128-cbc -in task5.txt -out t5.txt -K 5F2307DF6168A1FFA0A81E9048DA5F2A -iv D79AC3EFE35EA7F3A5B4535A68AE82CF  
[04/06/23]seed@VM:~/.../08_ske$ ghex t5.txt  
[04/06/23]seed@VM:~/.../08_ske$ openssl enc -d -aes-128-cbc -in t5.txt -out t5d.txt -K 5F2307DF6168A1FFA0A81E9048DA5F2A -iv D79AC3EFE35EA7F3A5B4535A68AE82CF  
[04/06/23]seed@VM:~/.../08_ske$
```



Using ECB we can still most of the original data but the bit we changed couldn't decrypt properly

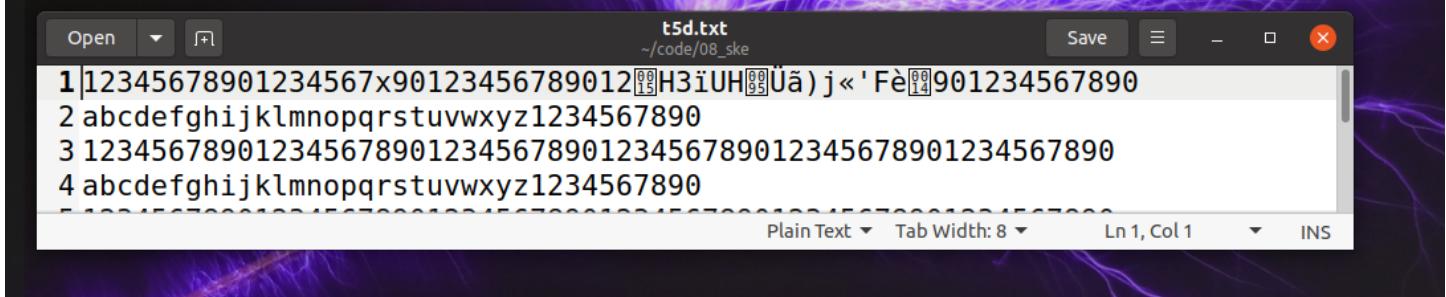
```
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.../08_ske$ openssl enc -e -aes-128-cfb -in task5.txt -out t5.txt
enter aes-128-cfb encryption password:
Verifying - enter aes-128-cfb encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.../08_ske$ ghex t5.txt
[04/06/23]seed@VM:~/.../08_ske$ openssl enc -d -aes-128-cfb -in t5.txt -out ff1.txt
enter aes-128-cfb decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.../08_ske$ openssl enc -e -aes-128-ofb -in task5.txt -out t5.txt
enter aes-128-ofb encryption password:
Verifying - enter aes-128-ofb encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.../08_ske$ ghex t5.txt
[04/06/23]seed@VM:~/.../08_ske$ openssl enc -d -aes-128-ofb -in t5.txt -out ff2.txt
enter aes-128-ofb decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.../08_ske$ openssl enc -e -aes-128-ecb -in task5.txt -out t5.txt
enter aes-128-ecb encryption password:
Verifying - enter aes-128-ecb encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.../08_ske$ ghex t5.txt
[04/06/23]seed@VM:~/.../08_ske$ openssl enc -d -aes-128-ecb -in t5.txt -out t5d.txt
enter aes-128-ecb decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.../08_ske$ 
```

Using OFB we can see that when we changed one byte in the encrypted .txt file and then unencrypted it, it only didn't decrypt that byte we changed.

```
[04/11/23] seed@VM:~/.../08_ske$ openssl enc -e -aes-128-ofb -in task5.txt -out t5e.txt
enter aes-128-ofb encryption password:
Verifying - enter aes-128-ofb encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/11/23] seed@VM:~/.../08_ske$ ghex t5e.txt
[04/11/23] seed@VM:~/.../08_ske$ openssl enc -d -aes-128-ofb -in t5e.txt -out t5d.txt
enter aes-128-ofb decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/11/23] seed@VM:~/.../08_ske$
```

Using CFB we can see that changing one bit changed in the encrypted .txt, it changes multiple bytes, but we are still able to see most of the original data.

```
seed@VM: ~/.../08_ske
enter aes-128-cfb decryption password:
[04/11/23]seed@VM:~/.../08_ske$ openssl enc -e -aes-128-cfb -in task5.txt -out t5e.txt
enter aes-128-cfb encryption password:
Verifying - enter aes-128-cfb encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/11/23]seed@VM:~/.../08_ske$ openssl enc -d -aes-128-cfb -in t5e.txt -out t5d.txt
enter aes-128-cfb decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/11/23]seed@VM:~/.../08_ske$ ghex t5e.txt
[04/11/23]seed@VM:~/.../08_ske$ openssl enc -d -aes-128-cfb -in t5e.txt -out t5d.txt
enter aes-128-cfb decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/11/23]seed@VM:~/.../08_ske$
```



### Task 5.3

Using CBC, the bit we changed didn't decrypt but the ones we didn't change decrypted just fine.

```

*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.08_ske$ openssl enc -e -aes-128-ofb -in task5.txt
enter aes-128-ofb encryption password:
Verifying - enter aes-128-ofb encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.08_ske$ ghex t5.txt
[04/06/23]seed@VM:~/.08_ske$ openssl enc -d
enter aes-128-ofb decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.08_ske$ openssl enc -e
enter aes-128-ecb encryption password:
Verifying - enter aes-128-ecb encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.08_ske$ ghex t5.txt
[04/06/23]seed@VM:~/.08_ske$ openssl enc -d
enter aes-128-ecb decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.08_ske$ openssl enc -e -aes-128-cbc -in task5.txt -out t5.txt
enter aes-128-cbc encryption password:
Verifying - enter aes-128-cbc encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.08_ske$ ghex t5.txt
[04/06/23]seed@VM:~/.08_ske$ openssl enc -d -aes-128-cbc -in t5.txt -out t5d.txt
enter aes-128-cbc decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.08_ske$ 

```

## Task 5.4

Using CFB we can see that changing one bit changed in the encrypted .txt, it changes multiple bytes, but we are still able to see most of the original data.

```

[04/06/23]seed@VM:~/.08_ske$ openssl enc -d -aes-128-cbc -in t5.txt -out t5d.txt
enter aes-128-cbc decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.08_ske$ openssl enc -e -aes-128-cfb -in task5.txt
enter aes-128-cfb encryption password:
Verifying - enter aes-128-cfb encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.08_ske$ ghex t5.txt
[04/06/23]seed@VM:~/.08_ske$ openssl enc -d -aes-128-cfb -in t5.txt -out t5d.txt
enter aes-128-cfb decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.08_ske$ ghex t5.txt
[04/06/23]seed@VM:~/.08_ske$ openssl enc -e -aes-128-cfb -in t5d.txt -out t5d.txt
enter aes-128-cfb encryption password:
Verifying - enter aes-128-cfb encryption password:
[04/06/23]seed@VM:~/.08_ske$ ghex t5.txt
[04/06/23]seed@VM:~/.08_ske$ openssl enc -d -aes-128-cfb -in t5d.txt -out t5d.txt
enter aes-128-cfb decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.08_ske$ ghex t5.txt
[04/06/23]seed@VM:~/.08_ske$ openssl enc -d -aes-128-cfb -in t5d.txt -out t5d.txt
enter aes-128-cfb decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[04/06/23]seed@VM:~/.08_ske$ 

```

In conclusion for Task 5, using CFB and CBC, when we change one byte, it corrupts 2 blocks of data. When we use ECB and OFB, it only corrupts 1 blocks of the data.

## Task 6.

### Task 6.1

```
Verifying - enter aes-128-cfb encryption password:  
[04/06/23]seed@VM:~/.../08_ske$ ghex t5.txt  
[04/06/23]seed@VM:~/.../08_ske$ openssl enc -d -aes-128-cfb -in t5.txt -  
enter aes-1  
Verifying - There was a problem opening the file "/home/seed/code/08_ske/t6e.txt".  
*** WARNING : The file you opened has some invalid characters. If you continue editing this file you could corrupt this document.  
You can also choose another character encoding and try again.  
Using -iter  
Character Encoding: Current Locale (UTF-8) ▾  
[04/06/23]seed@VM:~/.../08_ske$ 1$Salted_ 9F9D$ 2|NB|1D|äZ|a46;|B1N|FA|F9:X%|9C|9E|`|A6  
enter aes-1  
[04/06/23]seed@VM:~/.../08_ske$ 1$Salted_ 9F9D$ 2|NB|1D|äZ|a46;|B1N|FA|F9:X%|9C|9E|`|A6  
Verifying - enter aes-128-cfb encryption password:  
[04/06/23]seed@VM:~/.../08_ske$ 1$Salted_ 9F9D$ 2|NB|1D|äZ|a46;|B1N|FA|F9:X%|9C|9E|`|A6  
Using -iter  
Character Encoding: Current Locale (UTF-8) ▾  
[04/06/23]seed@VM:~/.../08_ske$ 1$Salted_ 9F9D$ 2|NB|1D|äZ|a46;|B1N|FA|F9:X%|9C|9E|`|A6  
Verifying - enter aes-128-cfb encryption password:  
[04/06/23]seed@VM:~/.../08_ske$ 1$Salted_ 9F9D$ 2|NB|1D|äZ|a46;|B1N|FA|F9:X%|9C|9E|`|A6  
Using -iter or -pbkdf2 would be better.  
[04/06/23]seed@VM:~/.../08_ske$
```

Here is with a different iv, the encrypted text is different

```
Using -ite  
[04/06/23]seed@VM:~/.../08_ske$ t6e.txt  
[04/06/23]seed@VM:~/.../08_ske$ Ff1.txt t5d.txt t6e.txt  
enter aes-1  
There was a problem opening the file "/home/seed/code/08_ske/t6e.txt".  
[04/06/23]seed@VM:~/.../08_ske$ The file you opened has some invalid characters. If you continue editing this file you could corrupt this document.  
You can also choose another character encoding and try again.  
enter aes-1  
*** WARNING : The file you opened has some invalid characters. If you continue editing this file you could corrupt this document.  
Character Encoding: Current Locale (UTF-8) ▾  
[04/06/23]seed@VM:~/.../08_ske$ 1$Salted_ 9F9D$ 2|NB|1D|äZ|a46;|B1N|FA|F9:X%|9C|9E|`|A6  
enter aes-1  
[04/06/23]seed@VM:~/.../08_ske$ 1$Salted_ 9F9D$ 2|NB|1D|äZ|a46;|B1N|FA|F9:X%|9C|9E|`|A6  
Verifying - enter aes-128-cbc encryption password:  
[04/06/23]seed@VM:~/.../08_ske$ 1$Salted_ 9F9D$ 2|NB|1D|äZ|a46;|B1N|FA|F9:X%|9C|9E|`|A6  
Verify fail  
bad password read  
140696079897920:error:2807106B:UI routines:UI_process:processing error:crypto/ui/ui_lib.c:545:while reading strings  
[04/06/23]seed@VM:~/.../08_ske$ openssl enc -e -aes-128-cbc -in task6.txt -out t6e.txt -iv D79AC3EFE35EA7F3A5B4535A68AE82CF  
enter aes-128-cbc encryption password:  
Verifying - enter aes-128-cbc encryption password:  
*** WARNING : deprecated key derivation used.  
Using -iter or -pbkdf2 would be better.  
[04/06/23]seed@VM:~/.../08_ske$
```

```
Using -ite  
[04/06/23]seed@VM:~/.../08_ske$ t6e.txt  
[04/06/23]seed@VM:~/.../08_ske$ Ff1.txt t5d.txt t6e.txt  
enter aes-1  
There was a problem opening the file "/home/seed/code/08_ske/t6e.txt".  
[04/06/23]seed@VM:~/.../08_ske$ The file you opened has some invalid characters. If you continue editing this file you could corrupt this document.  
You can also choose another character encoding and try again.  
Verifying -  
Character Encoding: Current Locale (UTF-8) ▾  
[04/06/23]seed@VM:~/.../08_ske$ 1$Salted_ 9A0  
bad password  
140696079897920:error:2807106B:UI routines:UI_process:processing error:crypto/ui/ui_lib.c:545:while reading strings  
[04/06/23]seed@VM:~/.../08_ske$ 1$Salted_ 9A0  
-iv D79AC3EFE35EA7F3A5B4535A68AE82CF  
enter aes-1  
[04/06/23]seed@VM:~/.../08_ske$ 1$Salted_ 9A0  
Verifying - enter aes-128-cbc encryption password:  
[04/06/23]seed@VM:~/.../08_ske$ 1$Salted_ 9A0  
*** WARNING : deprecated key derivation used.  
Using -iter or -pbkdf2 would be better.  
[04/06/23]seed@VM:~/.../08_ske$ openssl enc -e -aes-128-cbc -in task6.txt -out t6e.txt -iv D79AC3EFE35EA7F3A5B4535A68AE82CF  
enter aes-128-cbc encryption password:  
Verifying - enter aes-128-cbc encryption password:  
*** WARNING : deprecated key derivation used.  
Using -iter or -pbkdf2 would be better.  
[04/06/23]seed@VM:~/.../08_ske$
```

**Even with the same IV and password, the encrypted text was different.**

**Question:** *Can they decrypt other (different) encrypted messages if the same IV is always used? Not in my experiments. The only time we could get the same encrypted file contents unencrypted the same is if we use both the same key and iv*