Luis Medina

CS4351

Due 10/25/20

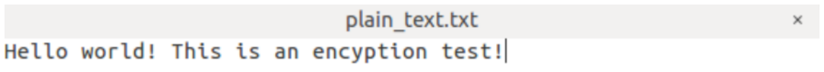
Assignment 3 – Secret Key Crypto

Problem

For this assignment we were to implement and observe encryption and decryption methods by applying them to files such as text and images. By editing these files, we see how errors can affect a file's integrity.

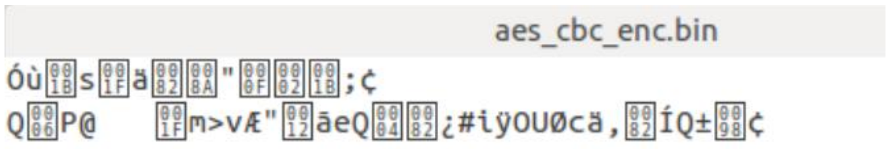
Task 1

1. First, I created a plain text file that I was to encrypt (*image 1*)
2. Next, I encrypted the text file with cbc by using the following command: openssl enc -aes-128-cbc -e -in plain_text.txt -out aes_cbc_enc.bin -K 00112233445566778899aabbccddeeff -iv 0102030405060708
3. Then, I opened this file to view its contents (*image 2*)



```
Hello world! This is an encyption test!
```

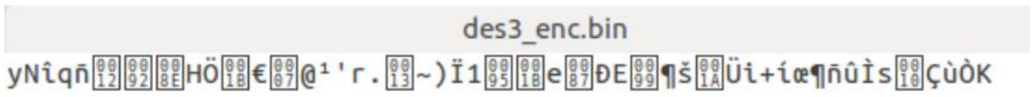
Image 1



```
Óù□s□ä□□"□□□;¢
Q□P@    □m>vÆ"□āeQ□□¿#iÿOUØcă,□ÍQ±□¢
```

Image 2

4. I then repeated this process, but I used des3 encryption with the following command:
5. openssl enc -des3 -e -in plain_text.txt -out des3_enc.bin -K 00112233445566778899aabbccddeeff -iv 0102030405060708
6. Then I opened the file (*image 3*)



```
yNîqñ□□□HÖ□€□@¹'r.□~)Ï1□□e□ÐE□¶š□Üi+íœ¶ñûÌs□ÇùÒK
```

Image 3

7. One last time, I encrypted the text file with a new encryption type: openssl enc -aes128 -e -in plain_text.txt -out aes128_enc.bin -K 00112233445566778899aabbccddeeff -iv 0102030405060708
8. Then I opened the file (*image 4*)

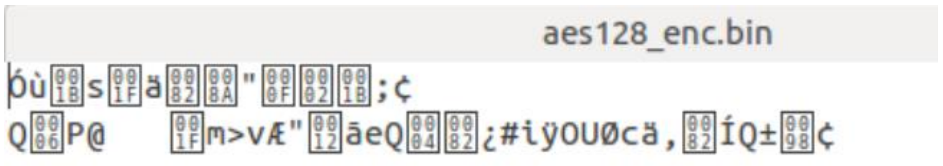


Image 4

Task 2

1. First, I encrypted the bmp picture provided (*image 5*) with ecb mode with the following command:
   openssl enc -aes-128-ecb -e -in pic_original.bmp -out p1.bmp -K 00112233445566778899aabbccddeeff

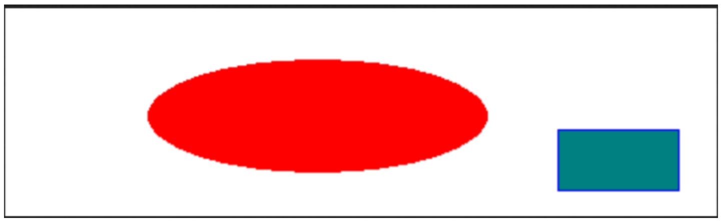


Image 5

2. Using the following commands to copy the original image's header and replaced the encrypted picture's header:
   - head -c 54 pic_original.bmp > header
   - tail -c +55 p1.bmp > body
   - cat header body > newp1.bmp
3. Then I checked the result (*image 6*)
   - Using this encryption method, the original image is still identifiable. It seems like it changed the values of the colors but retained the general construction of it.

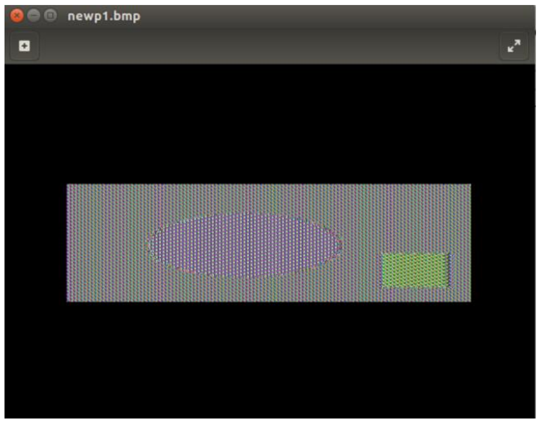


Image 6

4. I then encrypted the provided image with cbc mode by using the following command:
   openssl enc -aes-128-cbc -e -in pic_original.bmp -out p2.bmp -K 00112233445566778899aabbccddeeff -iv 0102030405060708

5. Using the following commands to copy the original image's header and replaced the encrypted picture's header:
   - tail -c +55 p2.bmp > body
   - cat header body > newp2.bmp
6. Then I checked the result (*image 7*)
   - This encryption method seems to be better fit for images. The result did not resemble the original image in any way.

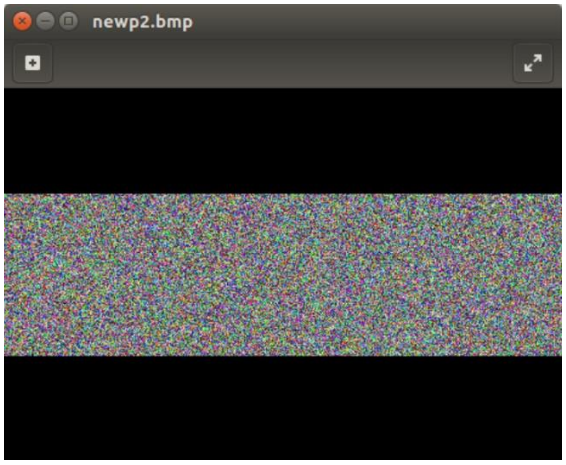


Image 7

7. Next, I selected a new image to encrypt (*image 8*)

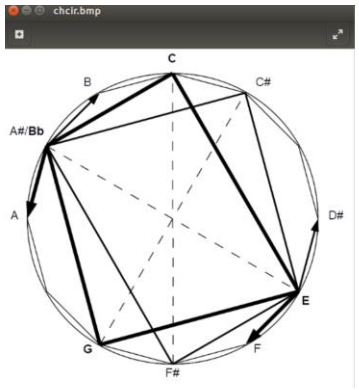


Image 8

8. I encrypted the image using ecb mode with the following command:
   openssl enc -aes-128-ecb -e -in chcir.bmp -out mp1.bmp -K 00112233445566778899aabbccddeeff
9. I again replaced the header on the new picture with the following commands:
   - head -c 54 chcir.bmp > header
   - tail -c +55 mp1.bmp > body
   - cat header body > newchcir.bmp
10. Then I opened the image (*image 9*)

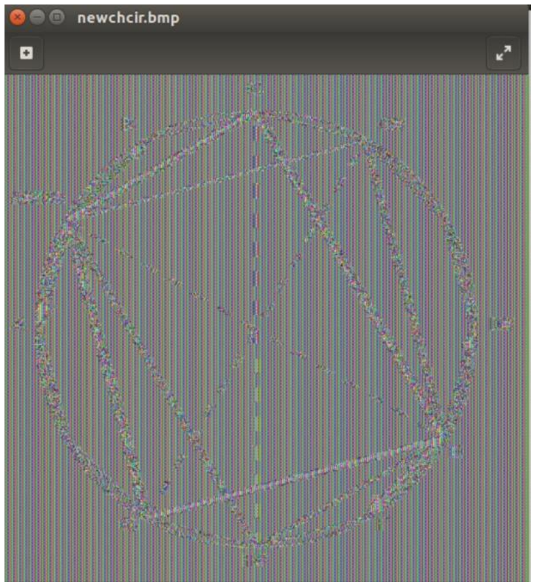


Image 9

11. I encrypted the original image using cbc mode with the following command:
    openssl enc -aes-128-cbc -e -in chcir.bmp -out mp2.bmp -K 00112233445566778899aabbccddeeff -iv 0102030405060708

12. I then replaced the header again:
    - tail -c +55 mp2.bmp > body
    - cat header body > newchcir2.bmp

13. Then I checked the image (*image 10*)
    - The results were similar to what I noticed with the provided image. Which is understandable as I used the same encryption methods.

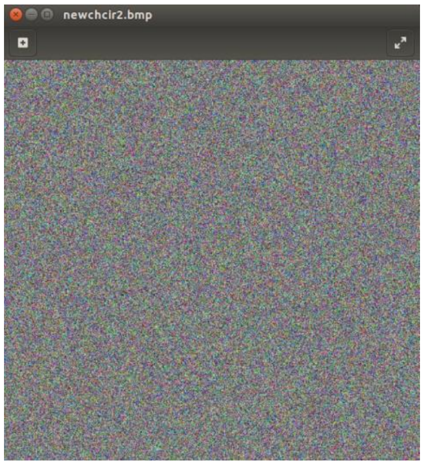
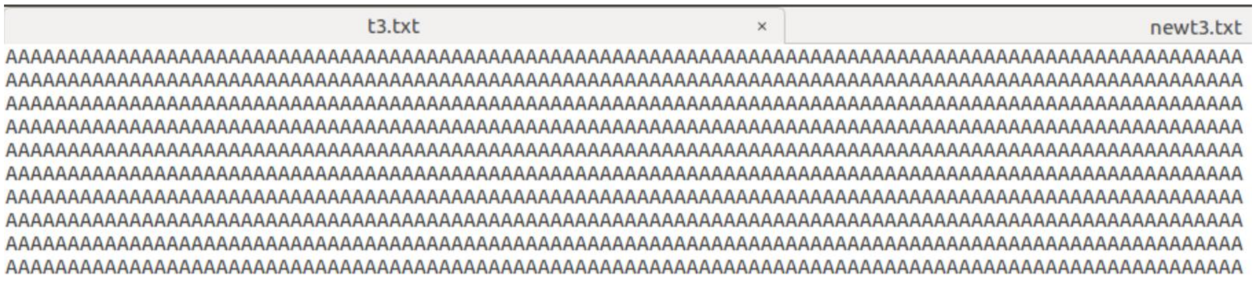


Image 10

Task 3

1. I created a text file that was 1010 byte by inputting 1000 ‘A’ characters (*image 11*)



```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

Image 11

2. I then encrypted this file with the following command:

openssl enc -aes128 -e -in t3.txt -out newt3.txt -K 00112233445566778899aabbccddeeff -iv 0102030405060708

3. I checked the file to verify (*image 12*)

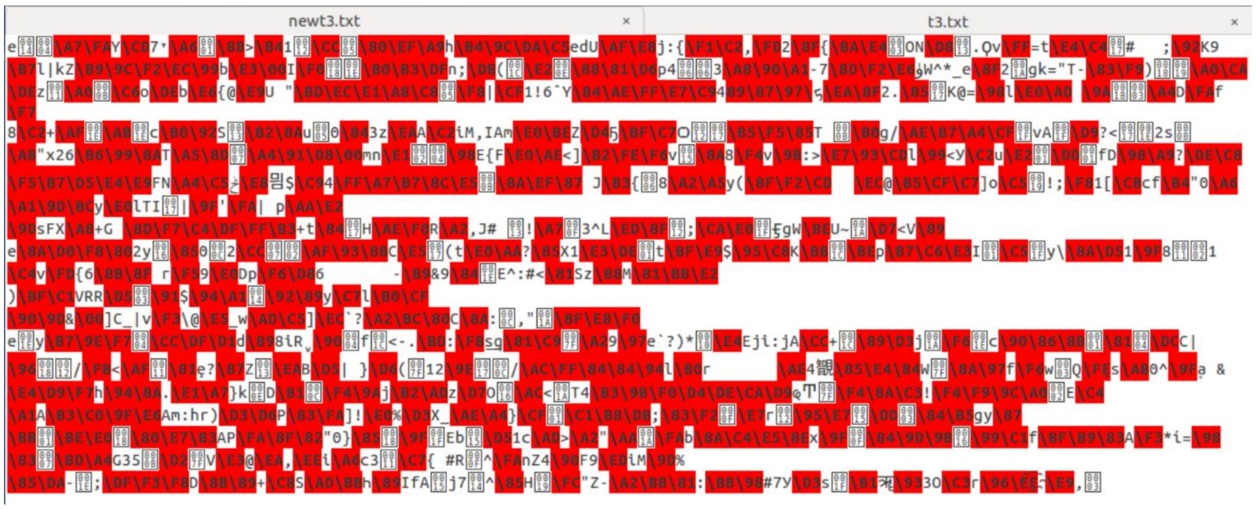
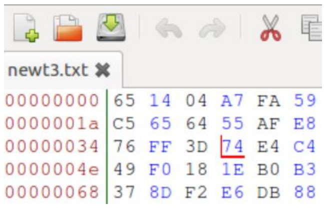


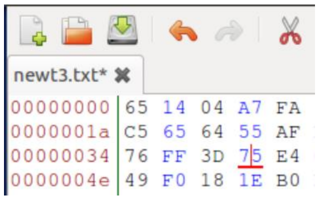
Image 12

4. Next, I opened the file with bless hex editor and changed the value of the 55th byte by one (*image 13 and 14*)



```
00000000 65 14 04 A7 FA 59
0000001a C5 65 64 55 AF E8
00000034 76 FF 3D 74 E4 C4
0000004e 49 F0 18 1E B0 B3
00000068 37 8D F2 E6 DB 88
```

Image 13



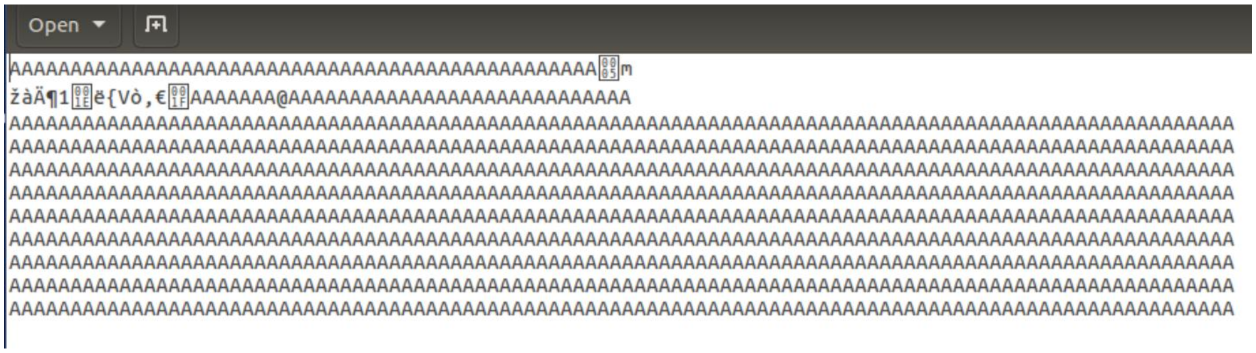
```
00000000 65 14 04 A7 FA
0000001a C5 65 64 55 AF
00000034 76 FF 3D 75 E4
0000004e 49 F0 18 1E B0
```

Image 14

5. I then decrypted the file with the following command:

```
openssl enc -aes128 -d -in newt3.txt -out dect3.txt -K 00112233445566778899aabbccddeeff -iv 0102030405060708
```

6. I then checked the contents (*image 15*)

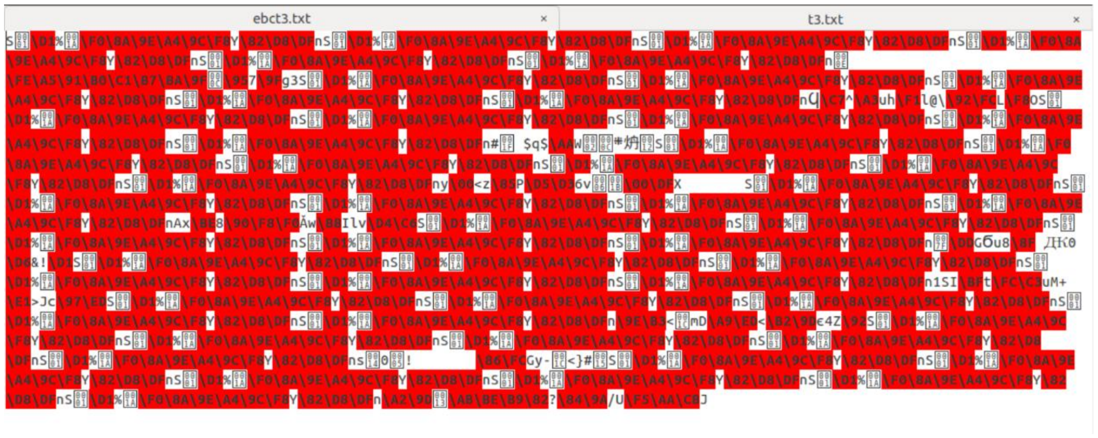


```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA□m
žàÄ¶1□ë{Vò,€□AAAAAAA@AAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

Image 15

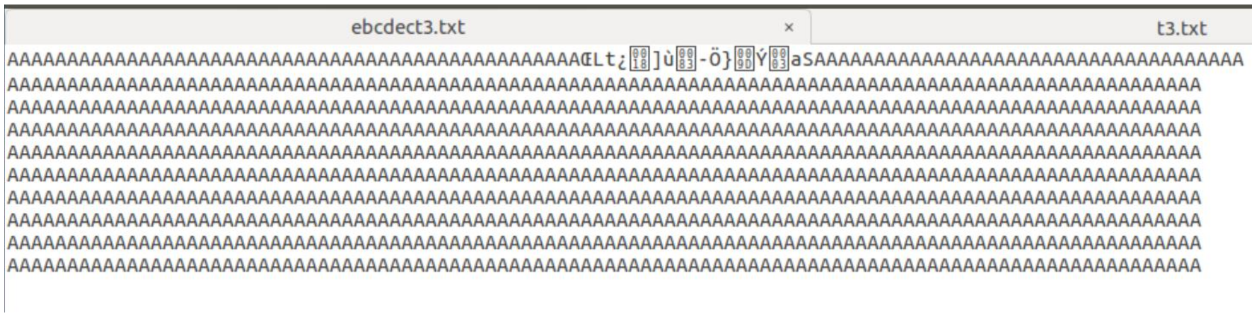
7. I then repeated steps 1 – 7 by using EBC, CBC and CFB instead to then compare the results.

EBC



```
S□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A
\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFn□
\FE\A5\91\B0\C1\B7\8A\9F□\957\9Fg3S□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E
\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnЦ\C7^\A3uh\F1l@\\92\FCL\F8OS□
\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E
\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFn#□ $q$\AAW□□*炳□S□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0
\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C
\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFny\00<z\85P\D5\D36v□□\00\DFX          S□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□
\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E
\A4\9C\F8Y\82\D8\DFnAx\BE8\90\F8\F0Ăw\B8Ilv\D4\C6S□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□
\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFn□\DDGБu8\8F ДК0
\D6&!\D1S□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□
\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFn1SI\BFt\FC\C3uM+
\E1>Jc\97\EDS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□
\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFn\9E\B3<□mD\A9\ED<\B2\9D€4Z\92S□\D1%□\F0\8A\9E\A4\9C
\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8
\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFns□0□!          \86\FCGy-□<}#□S□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E
\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82
\D8\DFnS□\D1%□\F0\8A\9E\A4\9C\F8Y\82\D8\DFn\A2\9D□\A8\8E\B9\82?\84\9A/U\F5\AA\CBJ
```

Image 16 (encrypted file)



```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAŒLt¿□]ù□-Ö}□Ý□aSAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

Image 17 (decrypted with error)

CBC

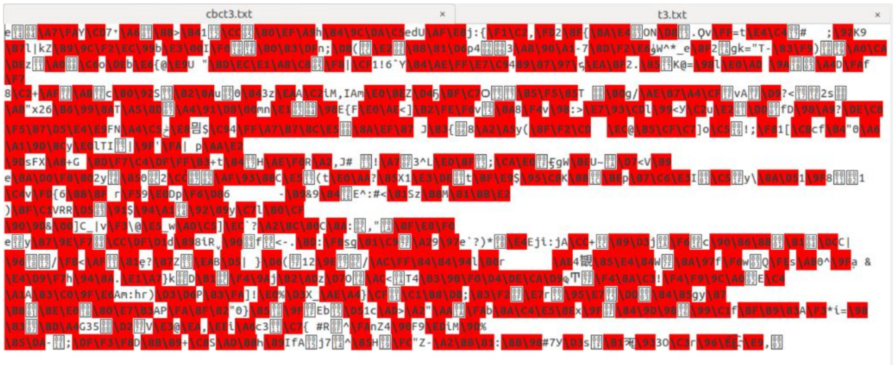
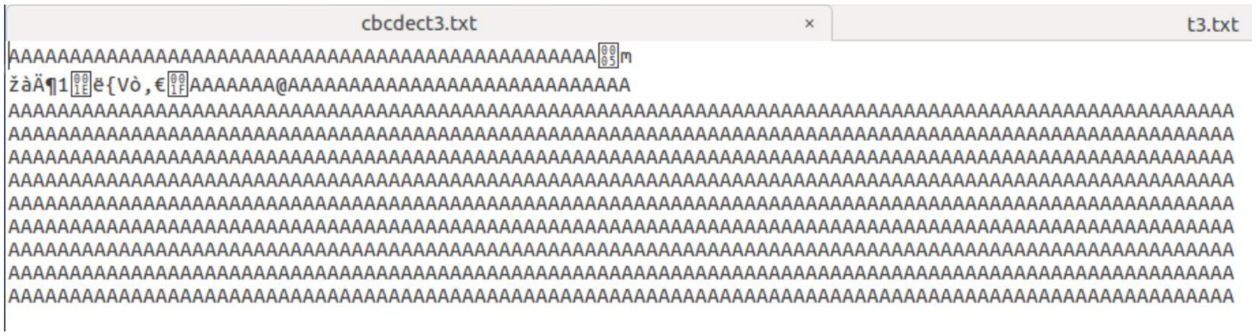


Image 18 (encrypted file)



```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAm
žàÄ¶1ë{Vò,€AAAAAAA@AAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

Image 19 (decrypted with error)

CFB

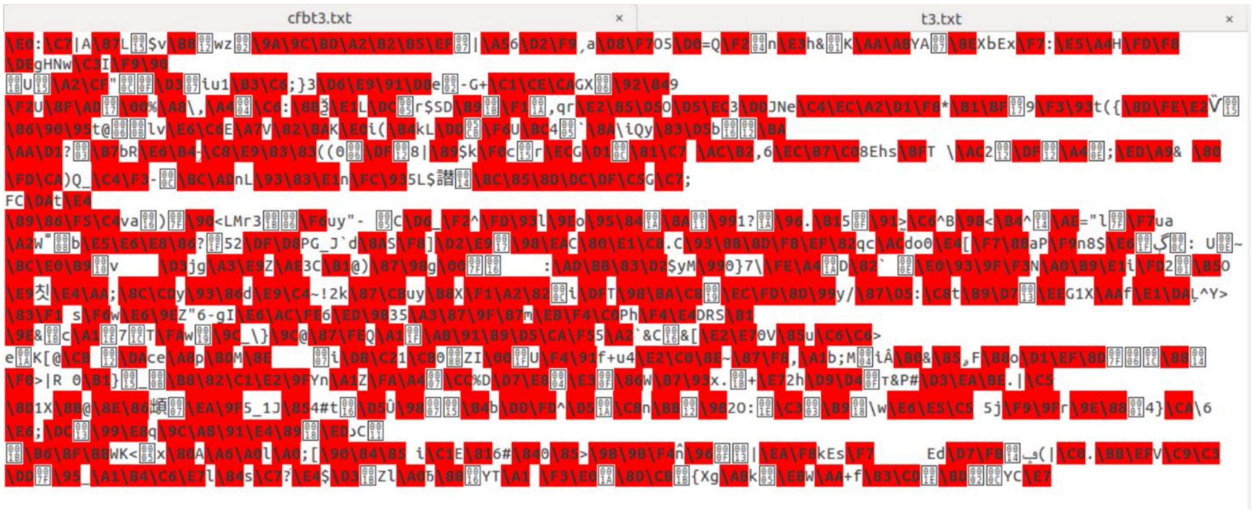
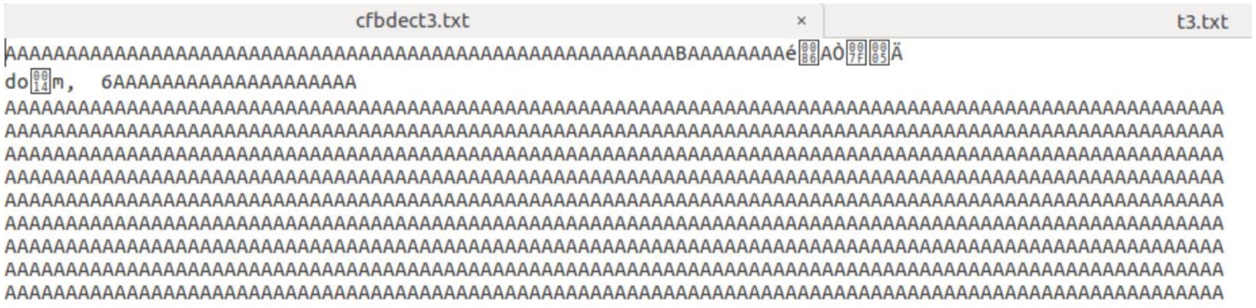


Image 20 (encrypted file)



```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABAAAAAAAAé AÒ  Ä
do m,  6AAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

Image 21 (decrypted with error)

Conclusion

From these results, it seems like ECB was the encryption mode that retained most of its data after having one byte be altered (*image 15*). In contrast, the modes that retained the least original data after modification were CBC and CFB (*image 19 and image 21*). This I believe is due to the fact that in these modes, some data relies on other data for the encryption/decryption process. CBC for example, works in blocks. If one part of the block is compromised, the entire block will be too. CFB on the other hand, as it works with data feedback, when one part becomes corrupt, the following piece of data will too. This causes entire sections to become compromised.