APPLIED CRYPTOGRAPHY

LAB-4

Date: 07/10/2022

Submitted by: Mahika Gupta

SRN: PES1UG20CS243

Task 1: Encryption using Different Ciphers and Modes

1. Using AES-128-cbc:

```
PES1UG20CS243:Mahika~/.../Labsetup$>cat>plain.txt
Hi this is a message from Mahika!
^C
PES1UG20CS243:Mahika~/.../Labsetup$>cat plain.txt
Hi this is a message from Mahika!
PES1UG20CS243:Mahika~/.../Labsetup$>
```

```
PES1UG20CS243:Mahika~/.../Labsetup$>openssl enc -aes-128-cbc -e  -in plain.txt -
out cipher.bin \-K  00112233445566778889aabbccddeeff \-iv 0102030405060708
hex string is too short, padding with zero bytes to length
PES1UG20CS243:Mahika~/.../Labsetup$>cat cipher.bin
5��`U!��.m��P0����6�    u��f����R*ғ >a���2-}���TPES1UG20CS243:Mahika~/.../Labset
up$>
```

```
PES1UG20CS243:Mahika~/.../Labsetup$>openssl enc -aes-128-cbc -d  -in cipher.bin
-out decrypted.txt \-K  00112233445566778889aabbccddeeff \-iv 0102030405060708
hex string is too short, padding with zero bytes to length
PES1UG20CS243:Mahika~/.../Labsetup$>cat decrypted.txt
Hi this is a message from Mahika!
PES1UG20CS243:Mahika~/.../Labsetup$>
```

2. Using AES-128-ecb:

```
PES1UG20CS243:Mahika~/.../Labsetup$>rm decrypted.txt
PES1UG20CS243:Mahika~/.../Labsetup$>openssl enc -aes-128-ecb -e  -in plain.txt -
out cipher.bin \-K  00112233445566778889aabbccddeeff
PES1UG20CS243:Mahika~/.../Labsetup$>cat cipher.bin
*[illegible]{[illegible]e*ö[illegible]{[illegible]vd[illegible]t]lL[illegible]&[illegible]Ca[illegible]$jl[illegible]PES1UG20CS243:Mahika~/.../Labsetup$>
```

```
PES1UG20CS243:Mahika~/.../Labsetup$>openssl enc -aes-128-ecb -d  -in cipher.bin
-out decrypted.txt \-K  00112233445566778889aabbccddeeff
PES1UG20CS243:Mahika~/.../Labsetup$>cat decrypted.txt
Hi this is a message from Mahika!
PES1UG20CS243:Mahika~/.../Labsetup$>
```

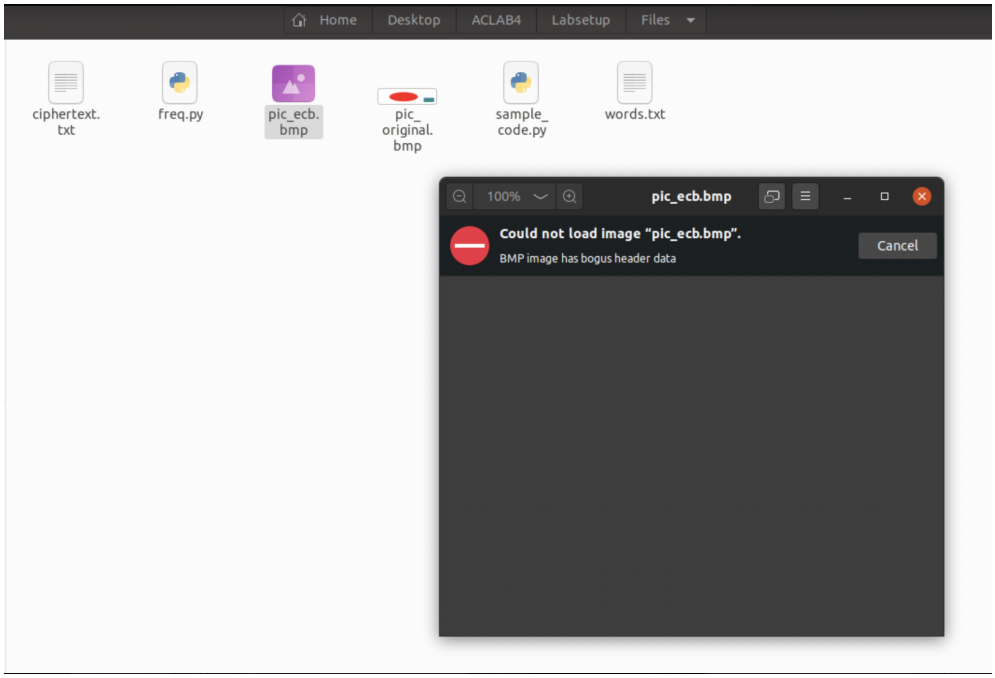
3. Using DES:

```
PES1UG20CS243:Mahika~/.../Labsetup$>openssl enc -des -e  -in plain.txt -out ciph
er.bin \-K  00112233445566778889aabbccddeeff \-iv 0102030405060708
hex string is too long, ignoring excess
PES1UG20CS243:Mahika~/.../Labsetup$>cat cipher.bin
[illegible]$C+[illegible]c[illegible]T/j[illegible]m[illegible]IxF[illegible]8g[illegible]dXh[illegible]PES1UG20CS243:Mahika~/.../Labsetup$>
```

```
PES1UG20CS243:Mahika~/.../Labsetup$>openssl enc -des -d  -in cipher.bin -out dec
rypted.txt \-K  00112233445566778889aabbccddeeff \-iv 0102030405060708
hex string is too long, ignoring excess
PES1UG20CS243:Mahika~/.../Labsetup$>cat decrypted.txt
Hi this is a message from Mahika!
PES1UG20CS243:Mahika~/.../Labsetup$>
```

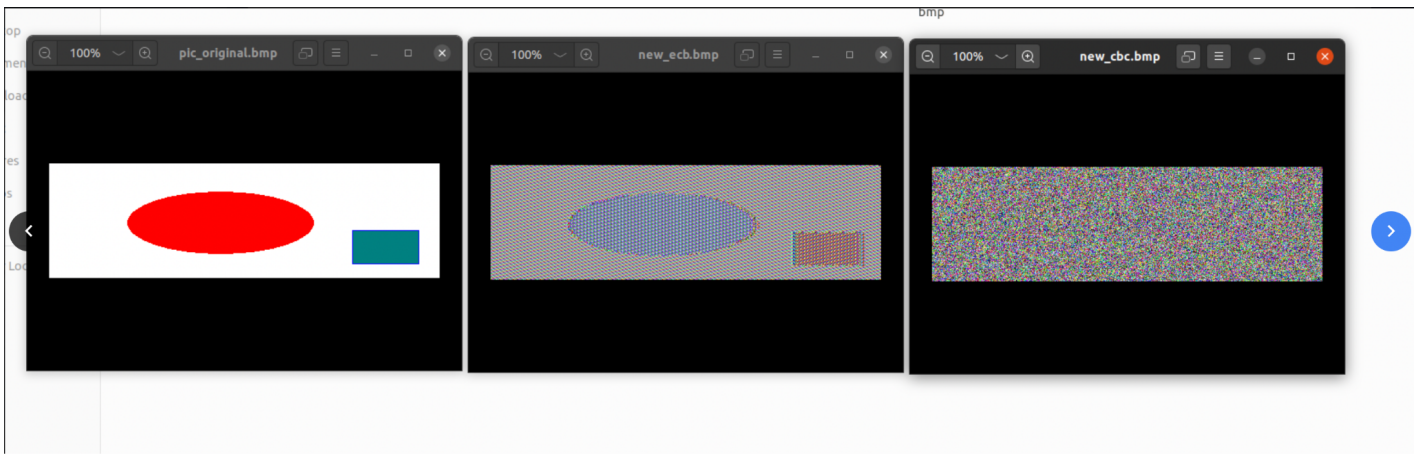
Task 2: Encryption Mode – ECB vs. CBC

```
PES1UG20CS243:Mahika~/.../Files$>openssl enc -aes-128-ecb -e -in pic_original.bm
p -out pic_ecb.bmp -K 00112233445566778889aabbccddeeff -iv 0102030405060708
warning: iv not used by this cipher
PES1UG20CS243:Mahika~/.../Files$>
```




```
PES1UG20CS243:Mahika~/.../Files$>openssl enc -aes-128-cbc -e -in pic_original.bm
p -out pic_cbc.bmp -K 00112233445566778889aabbccddeeff -iv 0102030405060708
hex string is too short, padding with zero bytes to length
PES1UG20CS243:Mahika~/.../Files$>ls
ciphertext.txt  pic_cbc.bmp  pic_original.bmp  words.txt
freq.py         pic_ecb.bmp  sample_code.py
PES1UG20CS243:Mahika~/.../Files$>
```

After adding headers to the image files:



Here we can observe that after using ecb for encryption, the image is still slightly distinguishable. However, on encrypting with cbc, the image is completely imperceptible.

Task 3: Padding:

```
PES1UG20CS243:Mahika~/.../Labsetup$>echo -n "12345" > f1.txt
PES1UG20CS243:Mahika~/.../Labsetup$>echo -n "1234567890" > f2.txt
PES1UG20CS243:Mahika~/.../Labsetup$>echo -n "1234567890abcdef" > f3.txt
PES1UG20CS243:Mahika~/.../Labsetup$>ls -l f*.txt
-rw-rw-r-- 1 seed seed  5 Oct  7 11:49 f1.txt
-rw-rw-r-- 1 seed seed 10 Oct  7 11:49 f2.txt
-rw-rw-r-- 1 seed seed 16 Oct  7 11:49 f3.txt
PES1UG20CS243:Mahika~/.../Labsetup$>
```

```
PES1UG20CS243:Mahika~/.../Labsetup$>openssl enc -aes-128-cbc -e -in f1.txt -out
f1.bin -K 00112233445566778889aabbccddeeff -iv 0102030405060708
hex string is too short, padding with zero bytes to length
PES1UG20CS243:Mahika~/.../Labsetup$>cat f1.bin
▯|▯f▯▯m▯Fr0▯dPES1UG20CS243:Mahika~/.../Labsetup$>^C
PES1UG20CS243:Mahika~/.../Labsetup$>openssl enc -aes-128-cbc -e -in f2.txt -out
f2.bin -K 00112233445566778889aabbccddeeff -iv 0102030405060708
hex string is too short, padding with zero bytes to length
PES1UG20CS243:Mahika~/.../Labsetup$>cat f2.bin
▯g▯Y▯▯A5t2FA▯▯PES1UG20CS243:Mahika~/^C
PES1UG20CS243:Mahika~/.../Labsetup$>openssl enc -aes-128-cbc -e -in f3.txt -out
f3.bin -K 00112233445566778889aabbccddeeff -iv 0102030405060708
hex string is too short, padding with zero bytes to length
PES1UG20CS243:Mahika~/.../Labsetup$>cat f3.bin
▯.%▯;▯$▯f▯s▯▯▯▯▯▯Gdo▯|F▯▯P▯PES1UG20CS243:Mahika~/.../Labsetup$>
```

```
1234567890abcdefPES1UG20CS243:Mahika~/.../Labsetup$>xxd p1.txt
00000000: 3132 3334 350b 0b0b 0b0b 0b0b 0b0b 0b0b  12345...........
PES1UG20CS243:Mahika~/.../Labsetup$>xxd p2.txt
00000000: 3132 3334 3536 3738 3930 0606 0606 0606  1234567890......
PES1UG20CS243:Mahika~/.../Labsetup$>xxd p3.txt
00000000: 3132 3334 3536 3738 3930 6162 6364 6566  1234567890abcdef
00000010: 1010 1010 1010 1010 1010 1010 1010 1010  ................
PES1UG20CS243:Mahika~/.../Labsetup$>
```

In p1.txt padding is done by 0b.

In p2.txt it is done by 06.

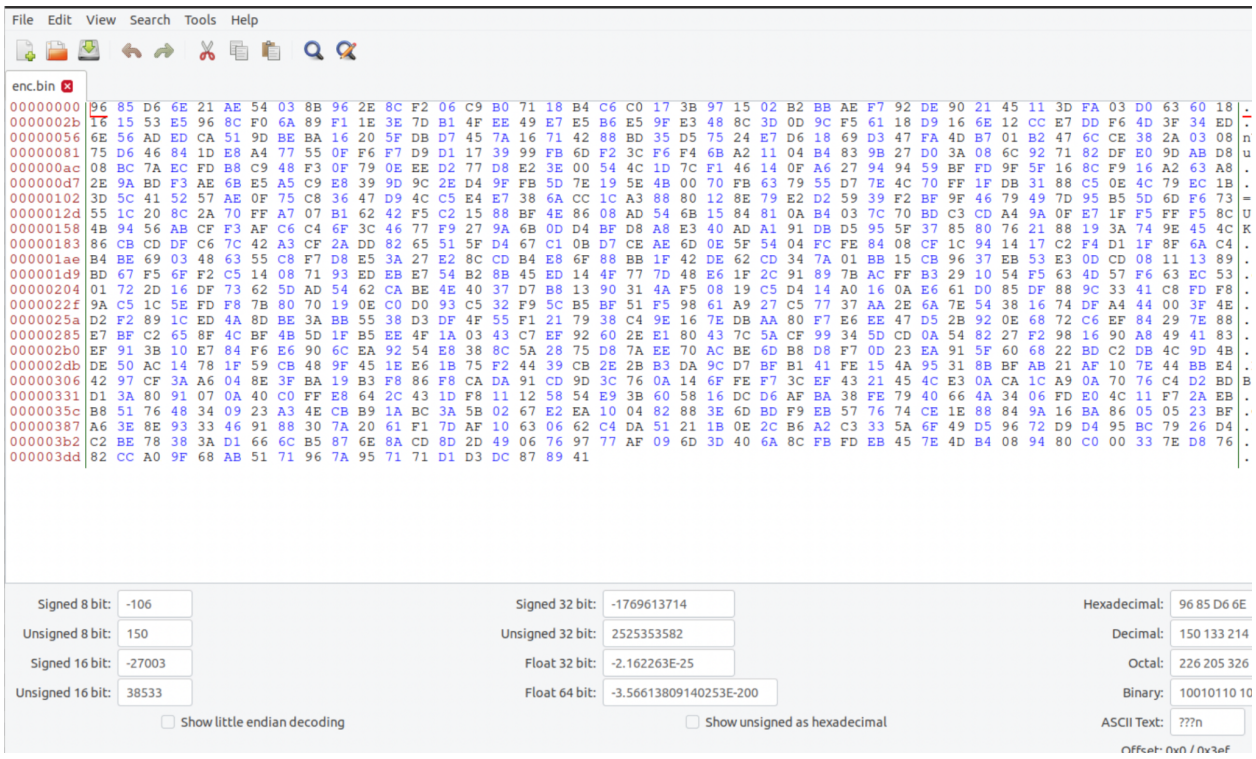
In p3.txt it is done by 10.

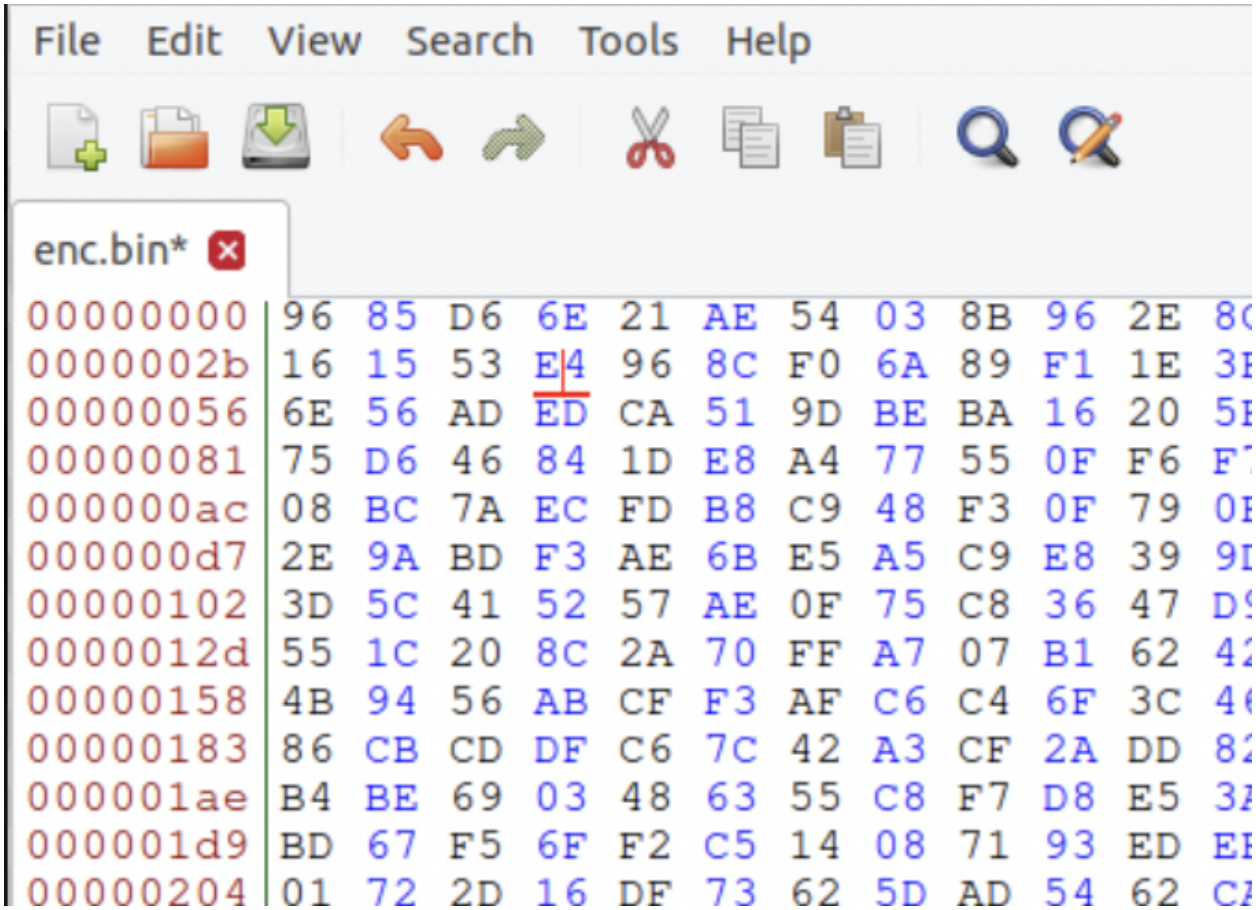
Task 4: Error Propagation – Corrupted Cipher Text

```
PES1UG20CS243:Mahika~/.../Labsetup$>nano error.txt
PES1UG20CS243:Mahika~/.../Labsetup$>cat error.txt
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Suspendisse in dui
et velit dictum dignissim. Suspendisse placerat erat ac ipsum tempor, id ves
tibulum urna bibendum. Cras placerat vehicula sem, ut ultrices felis. Etiam
congue posuere mi, sed tincidunt lorem euismod sit amet. Phasellus congue ma
ssa ac metus tempor sollicitudin. Aenean cursus massa at quam suscipit vulpu
tate. Nunc eget bibendum lorem. Sed eros libero, pulvinar ut dolor eu, cursu
s volutpat tellus. Vestibulum et neque condimentum, pharetra velit sodales,
rhoncus enim. Sed rhoncus sapien sed facilisis interdum. Nunc suscipit liber
o a ipsum gravida sollicitudin. Nulla turpis risus, ornare sed nunc nec, fin
ibus venenatis ipsum. Donec congue malesuada congue. Praesent mollis consequ
at nisi sed pretium. Donec a lacus ut neque convallis tincidunt.

Sed interdum, nunc eu hendrerit semper, lectus neque gravida dui, a blandit
lorem dolor et magna. Suspendisse luctus consequat urna ut venenatis. Nunc q
uis rutrum eros, in cras.
PES1UG20CS243:Mahika~/.../Labsetup$>wc -c error.txt
1004 error.txt
PES1UG20CS243:Mahika~/.../Labsetup$>
```

```
PES1UG20CS243:Mahika~/.../Labsetup$>openssl enc -aes-128-cbc -e -in error.tx
t -out enc.bin -K 00112233445566778889aabbccddeeff -iv 0102030405060708
hex string is too short, padding with zero bytes to length
PES1UG20CS243:Mahika~/.../Labsetup$>bless enc.bin
```





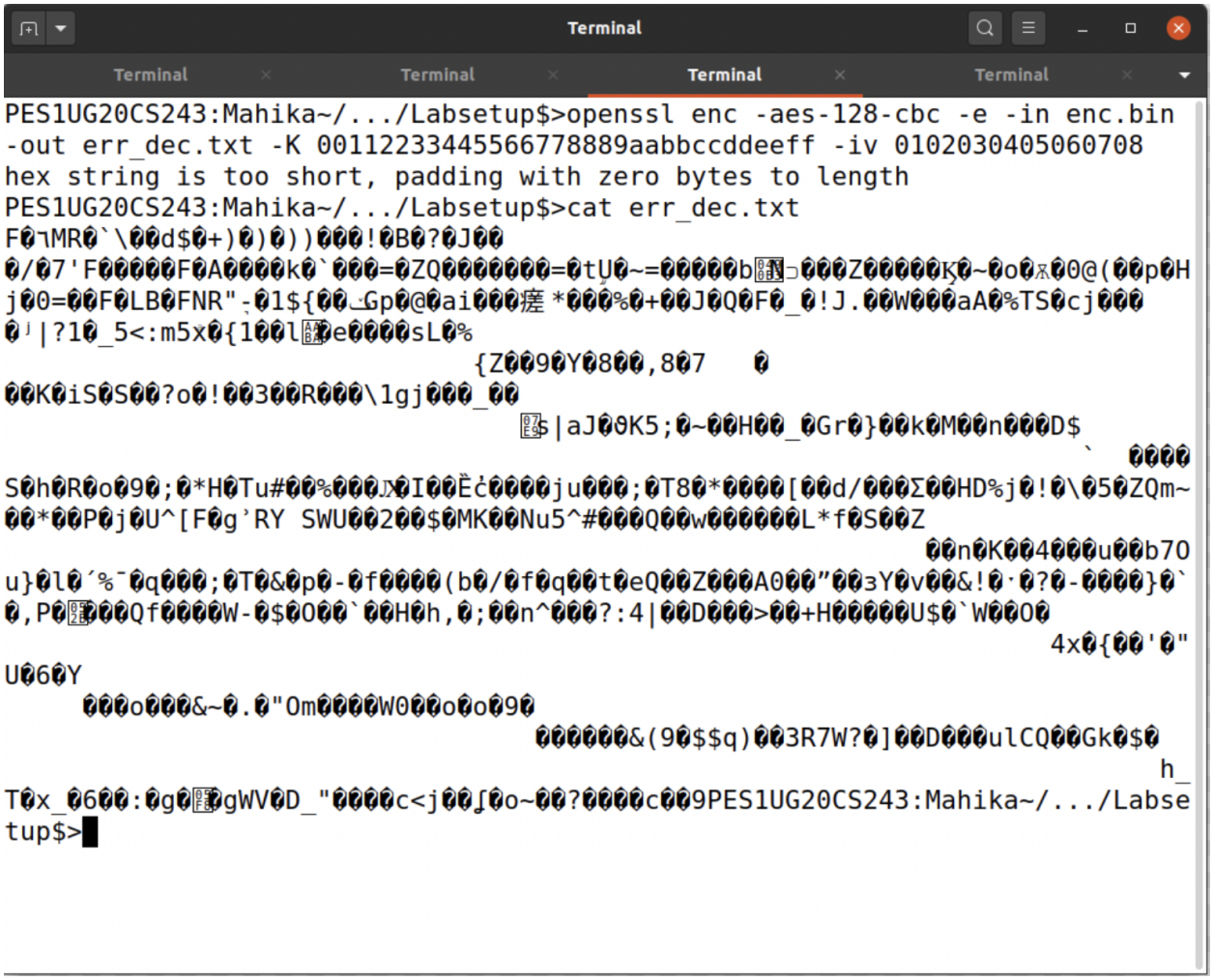
In the bless file, one byte is changed.

1. **Using AES-128-ecb**

```
PES1UG20CS243:Mahika~/.../Labsetup$>openssl enc -aes-128-ecb -e -in enc.bin
-out err_dec.txt -K 00112233445566778889aabbccddeeff
PES1UG20CS243:Mahika~/.../Labsetup$>cat err_dec.txt
????t#?mI?S?g???M????k\?
o3gN???_]?o???Mn        ?M??t???PV??t!e?+?\??s??Cy???r:?;L?b?f1H'?
               ???y??u?h????C?????È)?,#|???<???>??)?j???A]fD??? c36b,?h3|?
??J?Z?O??S?m??n?J?????????PI?H]$?藪O???7`
?x????i??N??o???^?I?(???Z?9?ebT??D?z??<?I+?j?????????q??2??????簪/??` ??9$?:
???{;?X?76???[R"????C?X??s?x??S?Ga??`?S?&3?y??<???      ?O??/??A?3?^??e??%?
                                                                        L
?v[????5?????w??F+D?K?V9Vy}>!???fF\?6?????_]?????Q?????N?Å???????d?????;n6??
?'?T_?"E?a+?I??/P???h-?q???GÉ?7C- d?1[???5? N???r?x?E??
?       W?N
           oa*e?
                ?u
W|?5??
i?????l?d?>r?X?xQ?>?{??0?/^??Ud?"R?N+´?[M8?]@Pal?'?????+iaą?_
                                                                8????????Z???y??
?V?0???S$^?I-??6?u?????a?%xsM
                                ?#?'??lh
                                        J?%1c?X?&???y`?R?^N?7V?
                                                                ?p>?X???????????O
n?x?e???7??55[?[??e^??#J???X?y{P??H??K??)??6??R???
                                                        _s???4?s??
t?D
鑭)LJ?#iw?$??A???&??4)e?St?G??E?9O??OS??       ?/???-u??w??yfj?PES1UG20CS24
3:Mahika~/.../Labsetup$>
```

We can see that the text is not recoverable on corrupting the encrypted file.

2. **Using AES-128-cbc:**



Similar to ecb, the text is not recoverable.

3. Using AES-128-cfb:

```
PES1UG20CS243:Mahika~/.../Labsetup$>openssl enc -aes-128-cfb -e -in enc.bin
-out err_dec.txt -K 00112233445566778889aabbccddeeff -iv 0102030405060708
hex string is too short, padding with zero bytes to length
PES1UG20CS243:Mahika~/.../Labsetup$>cat err_dec.txt
Lnrem ipsum dolo2Ψ@ỹ@@@^@@IC@3?@@@.0"@ @^@IqL{@I@@@2u]@@@媽@e@@@>@@ş@^@@@@@@
u@o@@T
@m@@h@@($ỹ@[,@#@@<n@Vmn@@@@1cT'}@@L@=@h@@@c.@@&@@)@@h@!@@@quп@C@@`@IL@
@@;-@@@@@@&@b@@@S@[zms@@@@@meFu@@@@@7=V@+T@E/@N@ob
                                                    @@@GN@@;@@@2@@@K@@OHù @@@ж
@@.@@p-@ca@cL2@\@@"@ɾɛr@j@^AE@@(~@@@C=@@
b@@Gq@x@
  l)_.ç@@@      @@@@ak@@
@[qTO@@ @@(@@@@@ʅ&@@@@OB@1@,@"@@@ɑ_@,u@@@@@&m@'V@@°@@@C>@@@tE@K@~@@xw:I]@6@@
@r@@@@I@p@@@V+@@@6j@@2@@L@@p@@{K@@@D@@w-@@@6@@^ @G@3@J9@bʙ@f]yF
}xCA@E@@772@@B@0@@
c?:p:w@yT@@5
@@@@@@,em@@@@`@@@@P@k@j@诺"@2@@F@@@@@@@k@@@0yc@š;@q/@@m@@@@ռ@@<@%B@7ɛ@SĎl3ᒍ@,@~
@@@@@@@`@@H@@@@0gm@q@W(-@ @d@@@5@u@CP@@'@@m.@@@R@@@@X@A@@@@Ĭ|yI?@@/0$@@7F@+@
@@@Fg@Z@@}@1@n@@a'@xp5@;@Y@e@d@X@,@E@@u@Js_K@@+d@=@fG@@@@@P@5@VQ@xg@66@6df@@
@@2@@)@ʅ@(@v=d
              @~X@@$@@F@2@@y@@J|ES@^@@@?;@a!@@h@@@i@@@d@4@c@@@C @{@.@@H@om|
@
 @@A)@@6@@@/@I@@@@O!@~@N@]@K@cD@ @$'@@m;Z ?@PES1UG20CS243:Mahika~/.../Labset
up$>
```

Only a few characters are discernible and recoverable.

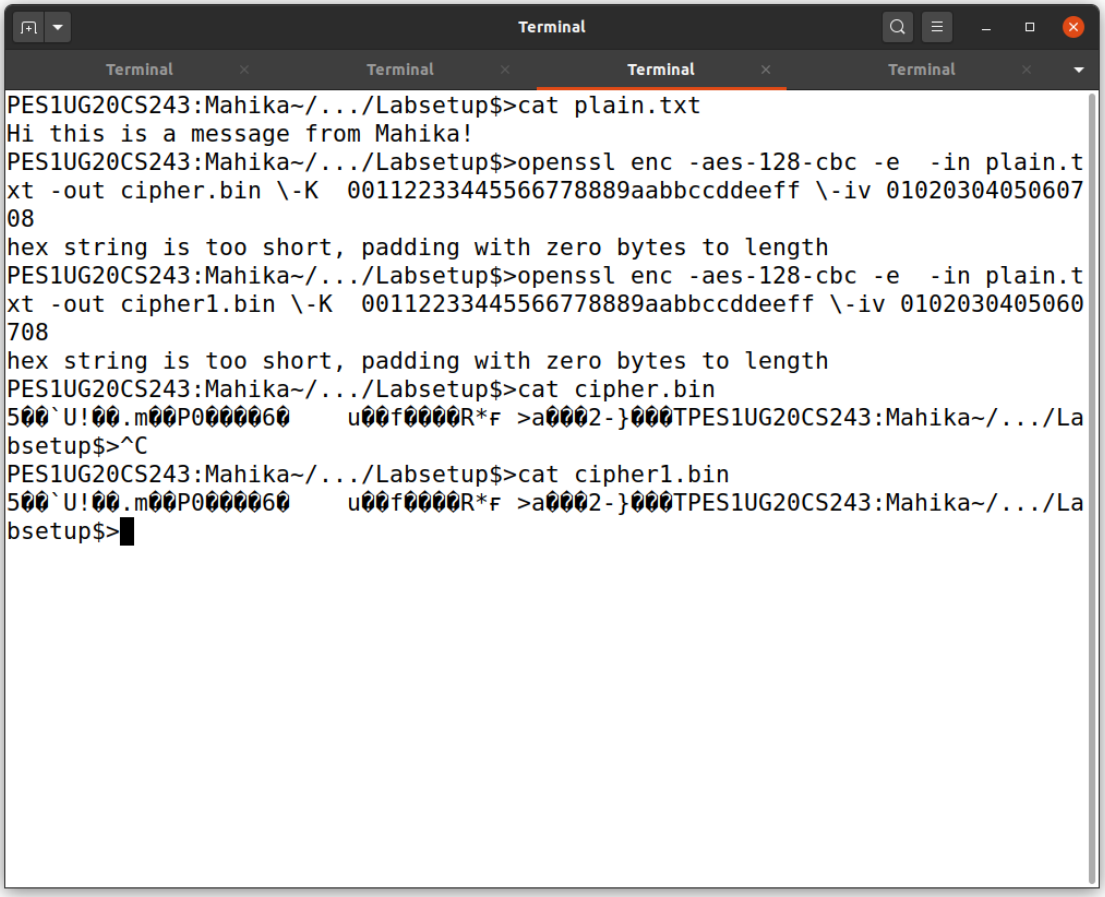
4. Using AES-128-ofb:

```
PES1UG20CS243:Mahika~/.../Labsetup$>openssl enc -aes-128-ofb -e -in enc.bin
-out err_dec.txt -K 00112233445566778889aabbccddeeff -iv 0102030405060708
hex string is too short, padding with zero bytes to length
PES1UG20CS243:Mahika~/.../Labsetup$>cat err_dec.txt
Lnrem ipsum dolor sit amet, consectetur adipiscing elit. Suspendisse in dui
et velit dictum dignissim. Suspendisse placerat erat ac ipsum tempor, id ves
tibulum urna bibendum. Cras placerat vehicula sem, ut ultrices felis. Etiam
congue posuere mi, sed tincidunt lorem euismod sit amet. Phasellus congue ma
ssa ac metus tempor sollicitudin. Aenean cursus massa at quam suscipit vulpu
tate. Nunc eget bibendum lorem. Sed eros libero, pulvinar ut dolor eu, cursu
s volutpat tellus. Vestibulum et neque condimentum, pharetra velit sodales,
rhoncus enim. Sed rhoncus sapien sed facilisis interdum. Nunc suscipit liber
o a ipsum gravida sollicitudin. Nulla turpis risus, ornare sed nunc nec, fin
ibus venenatis ipsum. Donec congue malesuada congue. Praesent mollis consequ
at nisi sed pretium. Donec a lacus ut neque convallis tincidunt.

Sed interdum, nunc eu hendrerit semper, lectus neque gravida dui, a blandit
lorem dolor et magna. Suspendisse luctus consequat urna ut venenatis. Nunc q
uis rutrum eros, in cras.
PES1UG20CS243:Mahika~/.../Labsetup$>
```

Using ofb, it is almost completely recoverable, there are only minor changes.

Task 5: Initial Vector (IV) and Common Mistakes:




```
PES1UG20CS243:Mahika~/.../Labsetup$>xxd cipher.bin
00000000: 35d8 de60 5521 a798 2e6d 9e95 5030 93db  5..`U!...m..P0..
00000010: 14d6 e436 a309 75c4 f066 f783 efdd 0252  ...6..u..f.....R
00000020: 2ad2 9320 3e61 a9ff ba32 2d7d f8d6 ec54  *.. >a...2-}...T
PES1UG20CS243:Mahika~/.../Labsetup$>xxd cipher1.bin
00000000: 35d8 de60 5521 a798 2e6d 9e95 5030 93db  5..`U!...m..P0..
00000010: 14d6 e436 a309 75c4 f066 f783 efdd 0252  ...6..u..f.....R
00000020: 2ad2 9320 3e61 a9ff ba32 2d7d f8d6 ec54  *.. >a...2-}...T
PES1UG20CS243:Mahika~/.../Labsetup$>
```

Here we can see that on using the same iv value, we produce the same cipher text.

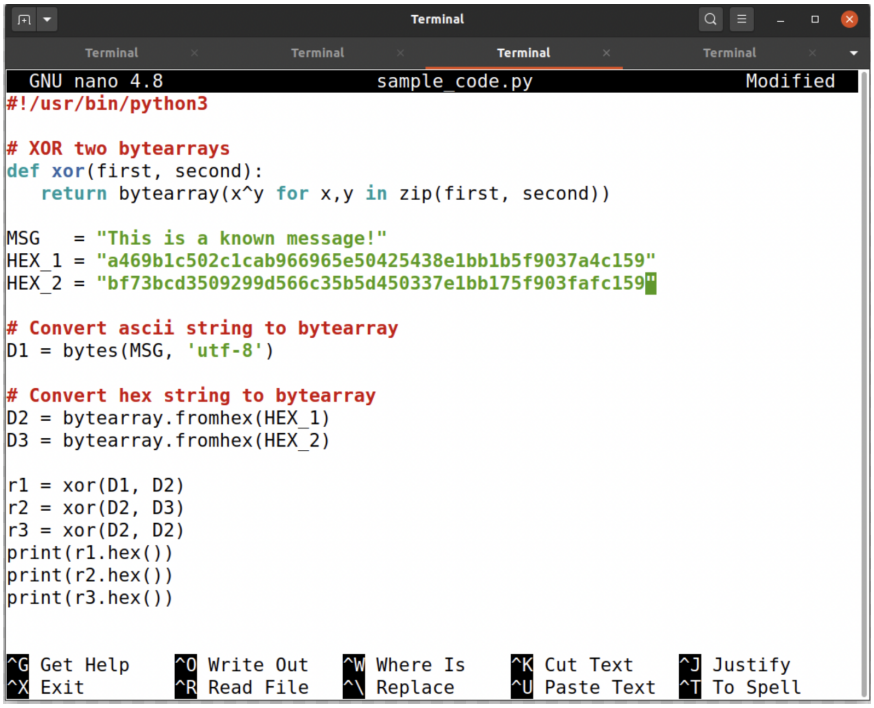
This can reveal some information about the cipher text to

the attacker.

Hence changing the iv value is important.

```
PES1UG20CS243:Mahika~/.../Labsetup$>openssl enc -aes-128-cbc -e  -in plain.t
xt -out cipher2.bin \-K  00112233445566778889aabbccddeeff \-iv 0102030405060
709
hex string is too short, padding with zero bytes to length
PES1UG20CS243:Mahika~/.../Labsetup$>xxd cipher.bin
00000000: 35d8 de60 5521 a798 2e6d 9e95 5030 93db  5..`U!...m..P0..
00000010: 14d6 e436 a309 75c4 f066 f783 efdd 0252  ...6..u..f.....R
00000020: 2ad2 9320 3e61 a9ff ba32 2d7d f8d6 ec54  *.. >a...2-}...T
PES1UG20CS243:Mahika~/.../Labsetup$>xxd cipher2.bin
00000000: 82f4 575b 6f1f fd4a 0dc1 aab3 5ded 05a7  ..W[o..J....]...
00000010: 28be 9787 f4f1 c359 561e 9700 25f8 aead  (......YV...%...
00000020: a1fa a6b9 f4f1 407c 6573 a39d 5b4c 417f  ......@|es..[LA.
PES1UG20CS243:Mahika~/.../Labsetup$>
```

Here we can see that on changing the iv value, we produce 2 different cipher texts.




```
PES1UG20CS243:Mahika~/.../Labsetup$>cd Files
PES1UG20CS243:Mahika~/.../Files$>ls
body_cbc         freq.py       new_ecb.bmp  pic_original.bmp
body_ecb         header        pic_cbc.bmp  sample_code.py
ciphertext.txt   new_cbc.bmp   pic_ecb.bmp  words.txt
PES1UG20CS243:Mahika~/.../Files$>nano sample_code.py
PES1UG20CS243:Mahika~/.../Files$>python3 sample_code.py
f001d8b622a8b99907b6353e2d2356c1d67e2ce356c3a478
1b1a0d165253536c0055050d07570f00000c0000080b0000
00000000000000000000000000000000000000000000000000
PES1UG20CS243:Mahika~/.../Files$>
```

The 2 cipher texts are XOR-ed. Since the XOR produces 0's, we can conclude that the same plaintext produces the same cipher texts.

Task 7: Programming using the Crypto Library :

```
PES1UG20CS243:Mahika~/.../Labsetup$>gcc code.c -o findkey -lcrypto
PES1UG20CS243:Mahika~/.../Labsetup$>./findkey
Got key: median
PES1UG20CS243:Mahika~/.../Labsetup$>
```

Here the words.txt file is searched by brute force to find the key which is used for this encryption.