

CSE565 Lab 1

Name: Sri Charan Reddy Teegala

Email: teegala@buffalo.edu

UBID: teegala

UB Number: 50681752

Before You Start:

Please write a detailed lab report, with **screenshots**, to describe what you have **done** and what you have **observed**. You also need to provide **explanation** to the observations that you noticed. Please also show the important **code snippets** followed by explanation. Simply attaching code without any explanation will **NOT** receive credits.

After you finish, export this report as a **PDF** file and submit it on UBLearn.

Academic Integrity Statement:

I, Sri Charan Reddy Teegala, have read and understood the course academic integrity policy.

Task 1: Frequency Analysis

Steps Performed:

1. Firstly, I ran the python script in the Files folder *freq.py* to fetch the single-letter frequencies, bigram frequencies and trigram frequencies of the letters in the file *ciphertext.txt*
2. Then I compared the relative frequencies of English alphabets (single, bigram and trigram) from sources mentioned in the Lab file with the output of *freq.py*, assumed few mappings, replaced them, and created a new file *out.txt*
3. After every iteration, I checked the words formed in the *out.txt* file and compared them with common words in English to get new mappings that we can add in the next iteration.
4. For the actual words, I used capital letters to mark the substitutions, I repeated these steps until entire *out.txt* file was capitalized, to get cipher key for decrypting the monoalphabetic cipher.

Screenshots:

The screenshot shows a Linux desktop environment with a dark theme. In the top panel, there's an 'Activities' button and a 'Text Editor' window icon. The terminal window (seed@VM: ~.../Files\$) displays three commands: tr 'ytn' 'THE' <ciphertext.txt> out.txt, tr 'ytnvup' 'THEAND' <ciphertext.txt> out.txt, and tr 'ytnvupmhrqz' 'THEANDIRGSUO' <in.txt> out.txt. The text editor window (out.txt) shows the output of the third command, which is a long, garbled string of characters.

```
[09/15/25]seed@VM:~/.../Files$ tr 'ytn' 'THE' <ciphertext.txt> out.txt
[09/15/25]seed@VM:~/.../Files$ tr 'ytnvup' 'THEAND' <ciphertext.txt> out.txt
[09/15/25]seed@VM:~/.../Files$ tr 'ytnvupmhrqz' 'THEANDIRGSUO' <in.txt> out.txt
[09/15/25]seed@VM:~/.../Files$ 
```

out.txt content (approximate):

```
1 THE xqaAhq TzhN xN qzNDAd lHmaH qEEca AgxzT hmrHT AbTEh THmq ixNr qThANrE
2 AlAhDq Thme THE gArrEh bEEiq imsE A NxNArENAhmAN Txx
3 |
4 THE AlAhDq hAaE lAq gxxsENDEd gd THE DEcmEq xb HAhfEd lEmNqTEMn AT mTq xzTqET
5 AND THE AeeAhENT mceixqmxN xb Hmq bmic axceAND AT THE END AND mT lAq qHAeED gd
6 THE EcEhrENaE xb cETxx TmcEq ze giAasrxLN eximTmaq AhcaANDd AaTmfmc AND
7 A NATmxNAi axNfEhqATmxN Aq gHEb AND cAD Aq A bEfEh DHEAc AgxzT lHETHEh THEHE
8 xzrHT Tx gE A ehEqmDENT lnNbhd Ed THE qEAqxN DmDNT ozqT qEEc EkThA ixNr mT lAq
9 EkThA ixNr gEaAzqE THE xqaAhq lEhE cxfED Tx THE bmhqt LEEsEND mN cAahH Tx
10 Afxd axNbimaTmNr lmTH THE aixqmNr aEhExcNd xb THE lmNTeh xidcemaq THANsQ
11 edExNraHAnR
12
13 xNE gmr jzEqTmxN qzhhxzNDmNr THmq dEAhq AaADEcd AlAhDq mq Hxl xh mb THE
14 aEhExcNd ADDhEqq cETxx EqeAmAiId AbTEh THE rxidEN rixgEq lHmaH gEaAcE
15 A ozgmiANT axcmNxzT eAhTd bxh TmcEq ze THE cxfEcENT qEahHEADED gd
16 exlEhbzi HxiidlxxD lxcEN lHx HEieED hAmqE cmiimxNq xb DxiiAhq Tx bmrHT qEkzAi
17 HAhAqqcENT AhxzND THE axzNThd
18
19 qmrNAimNr THEmh qzeexhT rxiDEN rixgEq ATTENDEEg qLATHEd THEEcqEifEq mN giAas
20 qexhTED iAeEi emNq AND qxzNDED xbb AgxzT qEkmqT exlEh mcgAiAnaEq bhxc THE hED
21 aAheET AND THE qTAre xN THE Anh E lAq aAiiED xzT AgxzT eAd mNEjzmTd AbTEh
22 mTq bxhcEh ANaHxh aATT qAdiEh jzmT xNaE qHE iEhNED THAT qHE lAq cAsmNr bAh
23 iEqq THAN A cAIE axHxqT AND DzhmNr THE aEhExcNd NATAiME exhTCAN Txxs A gizNT
24 AND qATmqbdmNr Dmr AT THE AiicAiE hxqTEh xb NxcmNATED DmhEaTxhq Hxl axziD
25 THAT gE TxeeED
26
27 AS IT TzRNs xzT AT iEAST IN TERcS xb THE xSaARS IT eRgAgid lxNT gE
```

Figure (1.1)

The screenshot shows a Linux desktop environment with a dark theme. In the top panel, there's an 'Activities' button and a 'Text Editor' window icon. The terminal window (seed@VM: ~.../Files\$) displays five commands related to file processing. The text editor window (out.txt) shows the output of the fifth command, which contains a large amount of garbled text.

```
[09/15/25]seed@VM:~/.../Files$ tr 'ytn' 'THE' <ciphertext.txt> out.txt
[09/15/25]seed@VM:~/.../Files$ tr 'ytnvup' 'THEAND' <ciphertext.txt> out.txt
[09/15/25]seed@VM:~/.../Files$ tr 'ytnvupmhrqz' 'THEANDIRGSUO' <in.txt> out.txt
[09/15/25]seed@VM:~/.../Files$ tr 'ytnvupmhrq' 'THEANDIRGS' <in.txt> out.txt
[09/15/25]seed@VM:~/.../Files$ 
```

out.txt content (approximate):

```
4 THE ALARDS RAde LAS gxxSENDED QU THE DECISE XD MARTED LEINSTEIN AT ITS XZISET
5 AND THE AeeARENT IceixSiXN xb HIS bIic axceAND AT THE END AND IT LAS SHAeED gd
6 THE EcERGENaE xb cETxx TICES ze giAasGxLN exiTiaS ArcaANDd AaTiFISc AND
7 A NATIxNAi axNFERSATIxN AS gRIEb AND cAD AS A bEFER DREAc AgxzT lHETHER THERE
8 xzGHT Tx gE A eRESIDENT LINBRED THE SEASxN DIDNT ozST SEEc EKTRA ixNG IT LAS
9 EKTRA ixNG gEaAzSE THE xSaARS LERE cxfED Tx THE bIRST LEESEND IN cARAH Tx
10 AfxID axNbilaTING lITH THE aixSING aEREExcNd xb THE LINTER xidceIaS THANsS
11 edExNGaHANG
12
13 xNE gIG jzESTIxN SzRRxzNDING THIS dEARS AaADEcd ALARDS IS Hxl xR Ib THE
14 aEREcNd lIi ADDRESS cETxx ESesEaIAiid AbTER THE GxiDEN GixgES lHIaH gEaAcE
15 A ozgIiANT axINGxZt eARTd bxR TiCes ze THE cxfEcENT SeEARHEADED gd
16 exlERbzi HxiidlxxD lxcEN lHx HEieED RAISE cIiiIxNS xb DxiiARS Tx bIGHT SEkzAi
17 HARASScENT ARxzND THE axzNTRd
18
19 SIGNAIING THEIR SzeexRT GxiDEN GixgES ATTENDEES SLATHED THEEcSEifEs IN giAas
20 SexRTED iAeEi eINS AND SxzdNDED xbb AgxzT SEkIST exlER IcgAiAnaEs bRxc THE RED
21 aReET AND THE STAGE xN THE AIR E lAS aAiiED xzT AgxzT eAd INEjzITd AbTER
22 ITS bxRcer ANaHxR aATT SADiER jzIT xNaE SHE iEARNED THAT SHE lAS cAsING bAR
23 iESS THAN A cAIE axHxST AND DzRING THE aEREExcNd NATAiIE exRTcAN Txxs A gizNT
24 AND SATISbDING DIG AT THE AiicAiE RxSTER xb NxcINATED DIREaTxRS Hxl axziD
25 THAT gE TxeeED
26
27 AS IT TzRNs xzT AT iEAST IN TERcS xb THE xSaARS IT eRgAgid lxNT gE
28
29 lxcEN INfxifED IN TiCes ze SAID THAT AiTHxzGH THE GixgES SIGNIBIED THE
30 INITIATIfEs iAzNaH THed NEFER INTENDED IT Tx gE ozST AN ALARDS SEASxN
```

Figure (1.2)

The screenshot shows a terminal window with a command-line interface and a text editor window. The terminal window has a dark background with light-colored text. It displays several commands using the `tr` command to process files, specifically mapping trigrams and single letters. The text editor window, titled "out.txt", also has a dark background and contains a large amount of text. The text is numbered from 1 to 25, suggesting it's a transcript or a series of numbered points. The text discusses the Academy Awards, mentioning Harvey Weinstein, the MeToo movement, and the BlackGown Politics. It also refers to the Winter Olympics in Pyeongchang and the Golden Globes. The text editor window has a "Save" button at the top right.

```

[09/15/25]seed@VM:~/.../Files$ tr 'ytn' 'THE' <ciphertext.txt> out.txt
[09/15/25]seed@VM:~/.../Files$ tr 'ytnvup' 'THEAND' <ciphertext.txt> out.txt
[09/15/25]seed@VM:~/.../Files$ tr 'ytnvupmhrqzx' 'THEANDIRGSUO' <in.txt> out.txt
[09/15/25]seed@VM:~/.../Files$ tr 'ytnvupmhrq' 'THEANDIRGS' <in.txt> out.txt
[09/15/25]seed@VM:~/.../Files$ tr 'ytnvupmhrqzxbgiecdsalfkjow' 'THEANDIRGSUOFBLPMYKCWVXQJZ' <in.txt>
> out.txt
[09/15/25]seed@VM:~/.../Files$
```

out.txt
-/Desktop/project01/abssetup/Files

1 THE OSCARS TURN ON SUNDAY WHICH SEEMS ABOUT RIGHT AFTER THIS LONG STRANGE
 2 AWARDS TRIP THE BAGGER FEELS LIKE A NONAGENARIAN TOO
 3
 4 THE AWARDS RACE WAS BOOKENDED BY THE DEMISE OF HARVEY WEINSTEIN AT ITS OUTSET
 5 AND THE APPARENT IMPLOSION OF HIS FILM COMPANY AT THE END AND IT WAS SHAPED BY
 6 THE EMERGENCE OF METOO TIMES UP BLACKGOWN POLITICS ARMCANDY ACTIVISM AND
 7 A NATIONAL CONVERSATION AS BRIEF AND MAD AS A FEVER DREAM ABOUT WHETHER THERE
 8 OUGHT TO BE A PRESIDENT WINFREY THE SEASON DIDNT JUST SEEM EXTRA LONG IT WAS
 9 EXTRA LONG BECAUSE THE OSCARS WERE MOVED TO THE FIRST WEEKEND IN MARCH TO
 10 AVOID CONFLICTING WITH THE CLOSING CEREMONY OF THE WINTER OLYMPICS THANKS
 11 PYEONGCHANG
 12
 13 ONE BIG QUESTION SURROUNDING THIS YEARS ACADEMY AWARDS IS HOW OR IF THE
 14 CEREMONY WILL ADDRESS METOO ESPECIALLY AFTER THE GOLDEN GLOBES WHICH BECAME
 15 A JUBILANT COMINGOUT PARTY FOR TIMES UP THE MOVEMENT SPEARHEADED BY
 16 POWERFUL HOLLYWOOD WOMEN WHO HELPED RAISE MILLIONS OF DOLLARS TO FIGHT SEXUAL
 17 HARASSMENT AROUND THE COUNTRY
 18
 19 SIGNALING THEIR SUPPORT GOLDEN GLOBES ATTENDEES SWATHED THEMSELVES IN BLACK
 20 SPORDED LAPEL PINS AND SOUNDED OFF ABOUT SEXIST POWER IMBALANCES FROM THE RED
 21 CARPET AND THE STAGE ON THE AIR E WAS CALLED OUT ABOUT PAY INEQUITY AFTER
 22 ITS FORMER ANCHOR CATT SADLER QUIT ONCE SHE LEARNED THAT SHE WAS MAKING FAR
 23 LESS THAN A MALE COHOST AND DURING THE CEREMONY NATALIE PORTMAN TOOK A BLUNT
 24 AND SATISFYING DIG AT THE ALLMALE ROSTER OF NOMINATED DIRECTORS HOW COULD
 25 THAT BE TOPPED

Figure (1.3)

Observations:

1. The most frequent trigram was *ytn* and the most frequent single letter was *n* from which we can conclude that *n* -> *e* and *ytn* -> *the*. Using *tr* command as shown in *Figure (1.1)* we marked the substitution.
2. After few iterations we could see some known words like '*SzRRxzNDING*' which could probably be *SURROUNDING* after decryption forming in *out.txt* in *Figure (1.2)* which we can use to find new mappings.
3. By substituting all the encrypted letters and adding the mappings to the *tr* command we could form cipher key '*ytnvupmhrqzxbgiecdsalfkjow*'

Code Snippets:

```
# Command for generating decrypted out.txt file from ciphertext.txt

tr 'ytnvupmhrqzxbgiecdsalfkjow' 'THEANDIRGSUOFBLPMYKCWVXQJZ' <ciphertext.txt> out.txt
```

Task 2: Encryption using Different Ciphers and Modes

Created a plain.txt file with some information and experimented with ***openssl enc*** command using different ciphers and modes.

1. Encryption with AES-128-CBC

Observations:

- AES-128-CBC requires both a 16-byte key and a 16-byte IV (padded IV with zeroes)
 - Decryption restored the original file correctly into *plain_cbc.txt*.

2. Encryption with AES-128-ECB

Observations:

- AES-128-ECB uses a 16-byte key and does not require an IV.
- Decryption recovered the original content in plain.txt into plain_ecb.txt

3. Encryption with Blowfish (BF_CBC)

```
seed@VM: ~/.../Files$ cat plain.txt
Computer Security Course CSE565
Sri Charan Reddy Teegala
teegala 50681752
[09/17/25]seed@VM:~/.../Files$ openssl enc -bf-cbc -e -in plain.txt -out bf_cbc.bin -K 00112233445566778889aabbccddeeff -iv 0102030405060708 -p
hex string is too long, ignoring excess
[09/17/25]seed@VM:~/.../Files$ cat bf_cbc.bin
^5*00S000r00, kbJ4D0000SH#0_00, 0R00090!00[0000eqS000:300|000`000000[09/17/25]seed@VM:~/.../Files$ cat bf_cbc.bin
^5*00S000r00, kbJ4D0000SH#0_00, 0R00090!00[0000eqS000:300|000`000000[09/17/25]seed@VM:~/.../Files$ 
[09/17/25]seed@VM:~/.../Files$ openssl enc -bf-cbc -d -in bf_cbc.bin -out plain_bf_cbc.txt -K 00112233445566778889aabbccddeeff -iv 0102030405060708-p
hex string is too long, ignoring excess
[09/17/25]seed@VM:~/.../Files$ cat plain_bf_cbc.txt
Computer Security Course CSE565
Sri Charan Reddy Teegala
teegala 50681752
[09/17/25]seed@VM:~/.../Files$ S
```

Observations:

- Blowfish uses an 8-byte IV that is the reason we see that error hex string is too long.
- Decryption restored the original plain.txt content into plain_bf_cbc.txt successfully.

Task 3: Encryption Mode – ECB vs. CBC

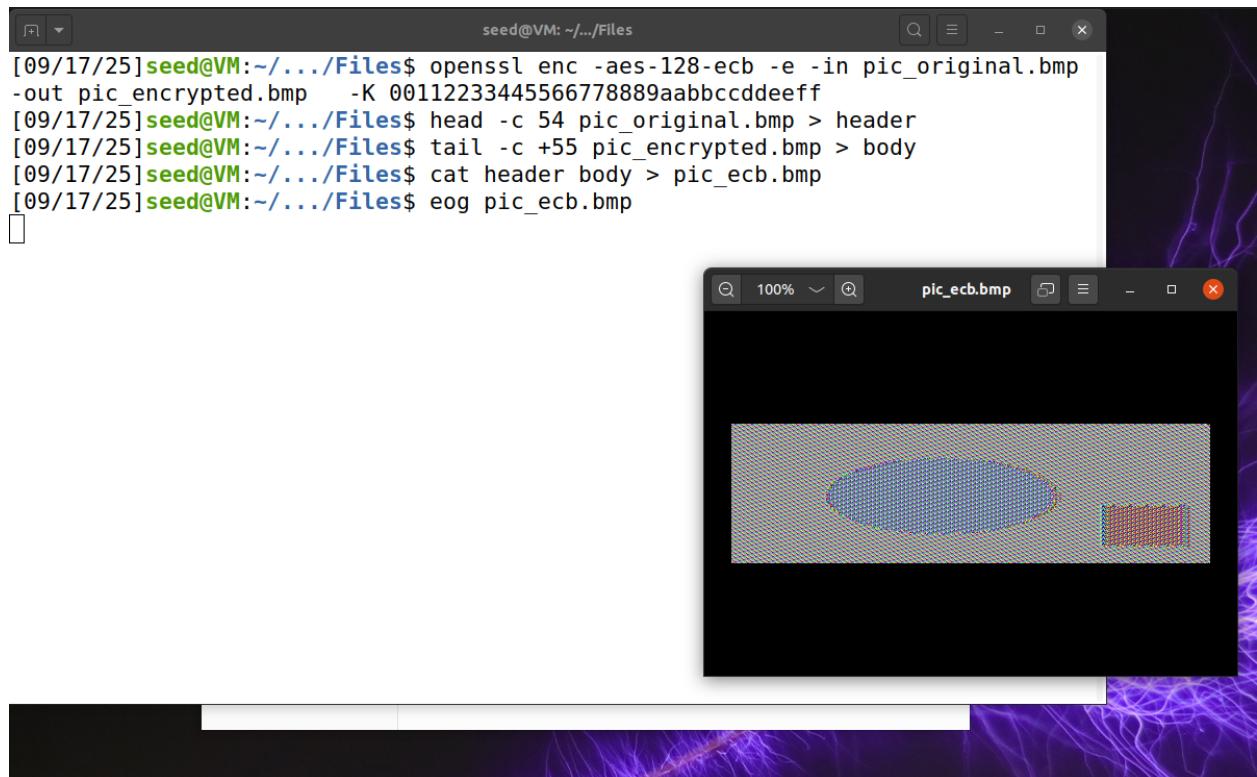
Steps Performed:

1. Encrypted the pic_original.bmp file using both ECB and CBC modes.
2. For the generated *pic_encrypted.bmp* file as mentioned we need to replace the header value with that of the original picture for retaining its properties.
3. Stored header from *pic_original.bmp* and the value of *pic_encrypted.bmp* without the header and combined them to get the *pic_ecb.bmp* and *pic_cbc.bmp* files for the modes ECB and CBC respectively.

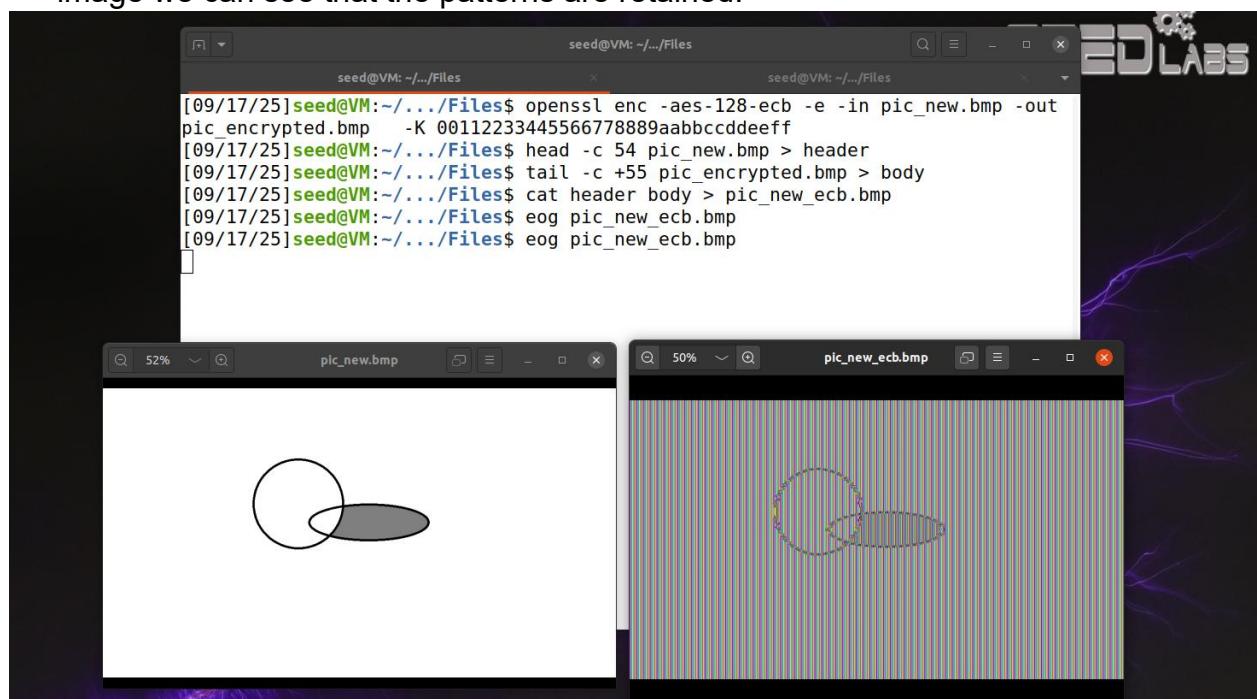
ECB Observations:

- The outline and the parameters of the original picture are still visible in the encrypted picture.

- This is because each 16-byte block is encrypted independently so identical blocks are encrypted into similar ciphertext blocks therefore the patterns remain.

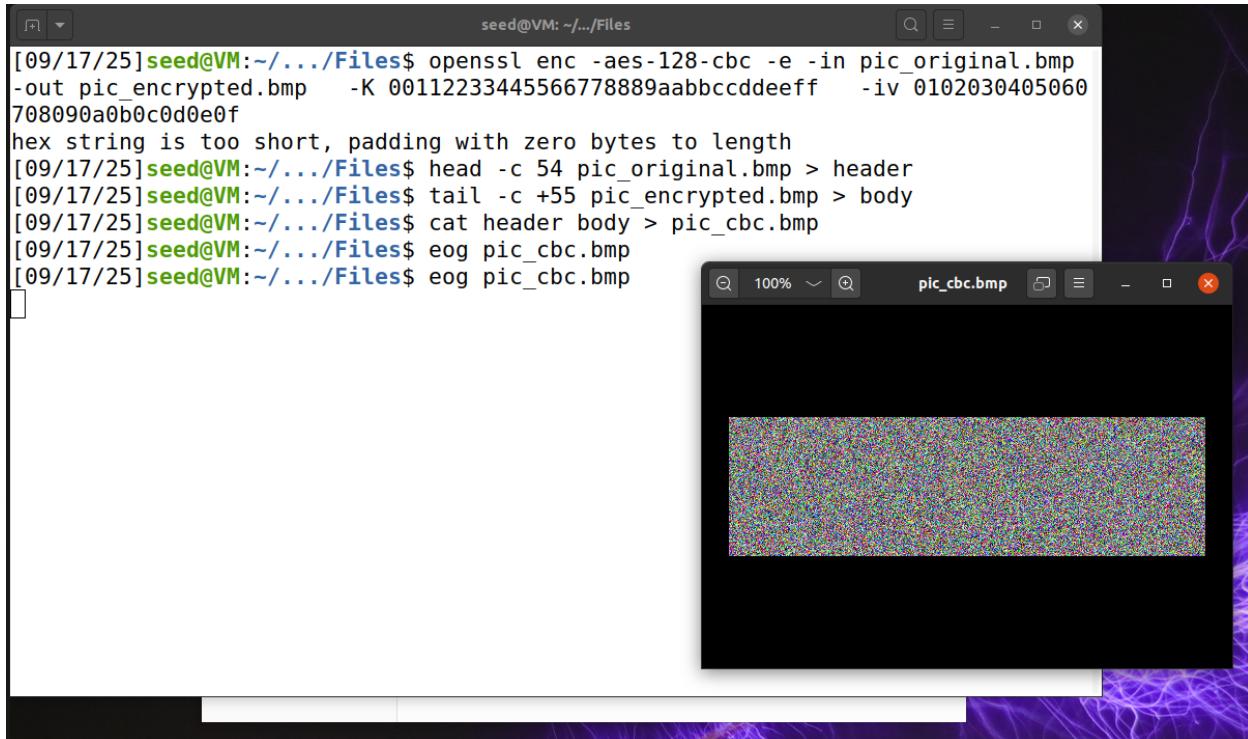


- Similarly, by taking a new picture and running the same steps to get encrypted image we can see that the patterns are retained.



CBC Observations:

- The encrypted image looks completely random and noisy.
- No structures or patterns from the original picture are still visible.
- This is because CBC introduces sequential XOR therefore each ciphertext block will depend on the previous ciphertext block, so it will not repeat patterns.



The screenshot shows a terminal window titled "seed@VM: ~/.../Files" with the following command history:

```
[09/17/25] seed@VM:~/.../Files$ openssl enc -aes-128-cbc -e -in pic_original.bmp -out pic_encrypted.bmp -K 00112233445566778889aabbcdddeeff -iv 0102030405060708090a0b0c0d0e0f
hex string is too short, padding with zero bytes to length
[09/17/25] seed@VM:~/.../Files$ head -c 54 pic_original.bmp > header
[09/17/25] seed@VM:~/.../Files$ tail -c +55 pic_encrypted.bmp > body
[09/17/25] seed@VM:~/.../Files$ cat header body > pic_cbc.bmp
[09/17/25] seed@VM:~/.../Files$ eog pic_cbc.bmp
[09/17/25] seed@VM:~/.../Files$ eog pic_cbc.bmp
```

Below the terminal is a small image viewer window titled "pic_cbc.bmp" showing a highly noisy, multi-colored rectangular image.

- Similarly, by taking a new picture and running the same steps to get encrypted image using cbc we obtain an encrypted image with no repeated patterns.

The screenshot shows a terminal window titled 'seed@VM: ~/.../Files' with the following command history:

```
[09/17/25]seed@VM:~/.../Files$ openssl enc -aes-128-cbc -e -in pic_new.bmp -out pic_encrypted.bmp -K 00112233445566778889aabcccddeff -iv 010203040506070809  
0a0b0c0d0e0f  
hex string is too short, padding with zero bytes to length  
[09/17/25]seed@VM:~/.../Files$ head -c 54 pic_new.bmp > header  
[09/17/25]seed@VM:~/.../Files$ tail -c +55 pic_encrypted.bmp > body  
[09/17/25]seed@VM:~/.../Files$ cat header body > pic_new_cbc.bmp  
[09/17/25]seed@VM:~/.../Files$ eog pic_new_cbc.bmp
```

Below the terminal are two windows from the 'eog' application. The left window is titled 'pic_new.bmp' and shows a white background with two overlapping circles: one white and one grey. The right window is titled 'pic_new_cbc.bmp' and shows a dark grey background with significant noise.

Task 4: Padding

Steps Performed:

1. I have created a file *name.txt* and verified its size using `ls -l` command and encrypted the file using all four modes ECB, CBC, CFB, and OFB.
2. Then, decrypted the file and checked the length of the new decrypted file to verify if there is any padding added
3. Created three files of sizes 5,10, and 16 bytes.
4. Encrypted all these files with AES-128-CBC cipher.
5. Decrypted files with `-nopad` to view padding
6. Used `xxd` command to inspect padded values.

Screenshots:

```
seed@VM: ~/.../Files
[09/17/25]seed@VM:~/.../Files$ ls -ld name.txt
-rw-rw-r-- 1 seed seed 11 Sep 17 21:27 name.txt
[09/17/25]seed@VM:~/.../Files$ openssl enc -aes-128-ecb -e -in name.txt -out name_ecb.txt -K 00
112233445566778889aabbccddeeff
[09/17/25]seed@VM:~/.../Files$ ls -ld name_ecb.txt
-rw-rw-r-- 1 seed seed 16 Sep 17 21:41 name_ecb.txt
[09/17/25]seed@VM:~/.../Files$ 
[09/17/25]seed@VM:~/.../Files$ 
[09/17/25]seed@VM:~/.../Files$ 
[09/17/25]seed@VM:~/.../Files$ openssl enc -aes-128-cbc -e -in name.txt -out name_cbc.txt -K 00
112233445566778889aabbccddeeff -iv 0102030405060708090a0b0c0d0e0f
hex string is too short, padding with zero bytes to length
[09/17/25]seed@VM:~/.../Files$ ls -ld name_cbc.txt
-rw-rw-r-- 1 seed seed 16 Sep 17 21:42 name_cbc.txt
[09/17/25]seed@VM:~/.../Files$ 
```

```
seed@VM: ~/.../Files
[09/17/25]seed@VM:~/.../Files$ ls -ld name.txt
-rw-rw-r-- 1 seed seed 11 Sep 17 21:27 name.txt
[09/17/25]seed@VM:~/.../Files$ openssl enc -aes-128-cfb -e -in name.txt -out name_cbc.txt -K 00
112233445566778889aabbccddeeff -iv 0102030405060708090a0b0c0d0e0f
hex string is too short, padding with zero bytes to length
[09/17/25]seed@VM:~/.../Files$ ls -ld name_cfb.txt
-rw-rw-r-- 1 seed seed 11 Sep 17 21:39 name_cfb.txt
[09/17/25]seed@VM:~/.../Files$ 
[09/17/25]seed@VM:~/.../Files$ 
[09/17/25]seed@VM:~/.../Files$ 
[09/17/25]seed@VM:~/.../Files$ openssl enc -aes-128-ofb -e -in name.txt -out name_ofb.txt -K 00
112233445566778889aabbccddeeff -iv 0102030405060708090a0b0c0d0e0f
hex string is too short, padding with zero bytes to length
[09/17/25]seed@VM:~/.../Files$ ls -ld name_ofb.txt
-rw-rw-r-- 1 seed seed 11 Sep 17 21:44 name_ofb.txt
[09/17/25]seed@VM:~/.../Files$ 
```

```
seed@VM: ~/.../Files
[09/17/25]seed@VM:~/.../Files$ echo -n "12345" > f1.txt
[09/17/25]seed@VM:~/.../Files$ ls -lh f1.txt
-rw-rw-r-- 1 seed seed 5 Sep 17 21:54 f1.txt
[09/17/25]seed@VM:~/.../Files$ echo -n "1234567890" > f2.txt
[09/17/25]seed@VM:~/.../Files$ ls -lh f2.txt
-rw-rw-r-- 1 seed seed 10 Sep 17 21:54 f2.txt
[09/17/25]seed@VM:~/.../Files$ echo -n "1234567890123456" > f3.txt
[09/17/25]seed@VM:~/.../Files$ ls -lh f3.txt
-rw-rw-r-- 1 seed seed 16 Sep 17 21:55 f3.txt
[09/17/25]seed@VM:~/.../Files$ 
```

```
seed@VM: ~/.../Files
[09/17/25]seed@VM:~/.../Files$ xxd f1.txt
00000000: 3132 3334 35 12345
[09/17/25]seed@VM:~/.../Files$ openssl enc -aes-128-cbc -e -in f1.txt -out f1_cbc.txt -K 00112233445566778889aabbccddeeff -iv 0102030405060708090a0b0c0d0e0f
hex string is too short, padding with zero bytes to length
[09/17/25]seed@VM:~/.../Files$ xxd f1_cbc.txt
00000000: 1489 50da afe4 33d8 aae2 c970 d011 37c4 ..P....3....p..7.
[09/17/25]seed@VM:~/.../Files$ openssl enc -aes-128-cbc -d -in f1_cbc.txt -out f1_decrypted.txt -K 00112233445566778889aabbccddeeff -iv 0102030405060708090a0b0c0d0e0f -nopad
hex string is too short, padding with zero bytes to length
[09/17/25]seed@VM:~/.../Files$

[09/17/25]seed@VM:~/.../Files$
[09/17/25]seed@VM:~/.../Files$ xxd f1_decrypted.txt
00000000: 3132 3334 350b 0b0b 0b0b 0b0b 0b0b 12345.....
```

```
[09/18/25]seed@VM:~/.../Files$ xxd f2.txt
00000000: 3132 3334 3536 3738 3930 1234567890
[09/18/25]seed@VM:~/.../Files$ openssl enc -aes-128-cbc -e -in f2.txt -out f2_cbc.txt -K 00112233445566778889aabbccddeeff -iv 0102030405060708090a0b0c0d0e0f
hex string is too short, padding with zero bytes to length
[09/18/25]seed@VM:~/.../Files$ xxd f2_cbc.txt
00000000: db69 c84a bfa6 0400 3ca8 0094 2778 2003 .i.J....<...'x .
[09/18/25]seed@VM:~/.../Files$ openssl enc -aes-128-cbc -d -in f2_cbc.txt -out f2_decrypted.txt -K 00112233445566778889aabbccddeeff -iv 0102030405060708090a0b0c0d0e0f -nopad
hex string is too short, padding with zero bytes to length
[09/18/25]seed@VM:~/.../Files$
[09/18/25]seed@VM:~/.../Files$
[09/18/25]seed@VM:~/.../Files$ xxd f2_decrypted.txt
00000000: 3132 3334 3536 3738 3930 0606 0606 0606 1234567890.....
[09/18/25]seed@VM:~/.../Files$
```

Observations:

- Modes and Padding Requirement.
ECB and CBC need padding (block-based)
CFB and OFB need no padding (stream-based)
- File 1 (5 bytes), Encrypted size: 16 bytes, Padding: 11 bytes of 0x0B
- File 2 (10 bytes), Encrypted size: 16 bytes, Padding: 6 bytes of 0x06
- File 1 (16 bytes), Encrypted size: 32 bytes (extra block added), Padding: 16 bytes of 0x10

Explanations:

- ECB and CBC encrypt using 16-byte blocks which require padding when plain text length is not a multiple block size.
- CFB and OFB are stream-based processes byte-by-byte therefore no padding is required.
- So, the difference between plaintext length and next block boundary is added as padding. If the plaintext length is exactly a block size, then a full block of padding is added.

Task 5: Error Propagation – Corrupted Cipher Text

How much information can you recover by decrypting the corrupted file, if the encryption mode is ECB, CBC, CFB, or OFB, respectively?

- In ECB, all plaintext except the 4th block (55th bit is in 4th block) can be recovered.
- In CBC, all plaintext except the 4th block and 5th block can be recovered.
- In CFB, ciphertext feed backs to generate future keystream i.e., single ciphertext pollutes future keystreams
- In OFB, only the bit won't be recovered everything else can be recovered.

Steps Performed:

1. Created a plaintext file *task5.txt* and encrypted it using AES-128 in ECB, CBC, CFB, OFB modes.
2. Flipped one bit in the 55th byte of ciphertext using dd (as bless editor was not working for me)

```
printf '\x01' | dd of=file.enc bs=1 seek=54 count=1 conv=notrunc
```

3. Decrypted the corrupted ciphertext with the same key/IV.
4. Printed the decrypted text and compared it with the original text.

Screenshots:

```
[09/18/25]seed@VM:~/.../Files$ cat task5.txt
Virat Kohli is one of the greatest cricketers of the modern era. Born on November 5, 1988, in Delhi, he grew from a young boy passionate about cricket into a global sporting icon. Known as the "Chase Master," Kohli has redefined batting with his incredible consistency, fitness, and hunger for runs. His aggressive style, quick running between the wickets, and ability to dominate bowlers in all formats of the game make him a nightmare for opponents.

Kohli has led India to many memorable victories, including historic wins overseas. As captain, he instilled discipline and self-belief in the team, emphasizing fitness as a key part of success. Under his leadership, India became the top-ranked Test side for several years. Off the field, his commitment to fitness has inspired millions of fans, and his intensity has set new standards for professionalism in cricket.

Beyond cricket, Kohli is admired for his philanthropic work through the Virat Kohli Foundation, which supports underprivileged children. His journey shows how determination and resilience can turn talent into greatness. Today, he remains not just a cricketer, but a role model for aspiring athletes around the world.

[09/18/25]seed@VM:~/.../Files$ openssl enc -aes-128-ecb -e -in task5.txt -out task5.bin -K 0011223344556677889aabcccddeeff
[09/18/25]seed@VM:~/.../Files$ printf "\x01" | dd of=task5.bin bs=1 seek=54 count=1 conv=notrunc,fdatasync
1+0 records in
1+0 records out
1 byte copied, 0.00131215 s, 0.8 kB/s
[09/18/25]seed@VM:~/.../Files$ openssl enc -aes-128-ecb -d -in task5.bin -out task5_decrypted.txt -K 0011223344556677889aabcccddeeff
[09/18/25]seed@VM:~/.../Files$ cat task5_decrypted.txt
Virat Kohli is one of the greatest cricketers of 0011223344556677889aabcccddeeff. Born on November 5, 1988, in Delhi, he grew from a young boy passionate about cricket into a global sporting icon. Known as the "Chase Master," Kohli has redefined batting with his incredible consistency, fitness, and hunger for runs. His aggressive style, quick running between the wickets, and ability to dominate bowlers in all formats of the game make him a nightmare for opponents.

Kohli has led India to many memorable victories, including historic wins overseas. As captain, he instilled discipline and self-belief in the team, emphasizing fitness as a key part of success. Under his leadership, India became the top-ranked Test side for several years. Off the field, his commitment to fitness has inspired millions of fans, and his intensity has set new standards for professionalism in cricket.

Beyond cricket, Kohli is admired for his philanthropic work through the Virat Kohli Foundation, which supports underprivileged children. His journey shows how determination and resilience can turn talent into greatness. Today, he remains not just a cricketer, but a role model for aspiring athletes around the world.

[09/18/25]seed@VM:~/.../Files$
```

```
[09/18/25]seed@VM:~/.../Files$ cat task5.txt
Virat Kohli is one of the greatest cricketers of the modern era. Born on November 5, 1988, in Delhi, he grew from a young boy passionate about cricket into a global sporting icon. Known as the "Chase Master," Kohli has redefined batting with his incredible consistency, fitness, and hunger for runs. His aggressive style, quick running between the wickets, and ability to dominate bowlers in all formats of the game make him a nightmare for opponents.

Kohli has led India to many memorable victories, including historic wins overseas. As captain, he instilled discipline and self-belief in the team, emphasizing fitness as a key part of success. Under his leadership, India became the top-ranked Test side for several years. Off the field, his commitment to fitness has inspired millions of fans, and his intensity has set new standards for professionalism in cricket.

Beyond cricket, Kohli is admired for his philanthropic work through the Virat Kohli Foundation, which supports underprivileged children. His journey shows how determination and resilience can turn talent into greatness. Today, he remains not just a cricketer, but a role model for aspiring athletes around the world.

[09/18/25]seed@VM:~/.../Files$ openssl enc -aes-128-cbc -e -in task5.txt -out task5.bin -K 0011223344556677889aabcccddeeff -iv 0102030405060708090a0b0c0d0e0f
hex string is too short, padding with zero bytes to length
[09/18/25]seed@VM:~/.../Files$ printf "\x01" | dd of=task5.bin bs=1 seek=54 count=1 conv=notrunc,fdatasync
1+0 records in
1+0 records out
1 byte copied, 0.0110026 s, 0.1 kB/s
[09/18/25]seed@VM:~/.../Files$ openssl enc -aes-128-cbc -d -in task5.bin -out task5_decrypted_cbc.txt -K 0011223344556677889aabcccddeeff -iv 0102030405060708090a0b0c0d0e0f
hex string is too short, padding with zero bytes to length
[09/18/25]seed@VM:~/.../Files$ cat task5_decrypted_cbc.txt
Virat Kohli is one of the greatest cricketers of 0011223344556677889aabcccddeeff. Born on November 5, 1988, in Delhi, he grew from a young boy passionate about cricket into a global sporting icon. Known as the "Chase Master," Kohli has redefined batting with his incredible consistency, fitness, and hunger for runs. His aggressive style, quick running between the wickets, and ability to dominate bowlers in all formats of the game make him a nightmare for opponents.

Kohli has led India to many memorable victories, including historic wins overseas. As captain, he instilled discipline and self-belief in the team, emphasizing fitness as a key part of success. Under his leadership, India became the top-ranked Test side for several years. Off the field, his commitment to fitness has inspired millions of fans, and his intensity has set new standards for professionalism in cricket.

Beyond cricket, Kohli is admired for his philanthropic work through the Virat Kohli Foundation, which supports underprivileged children. His journey shows how determination and resilience can turn talent into greatness. Today, he remains not just a cricketer, but a role model for aspiring athletes around the world.

[09/18/25]seed@VM:~/.../Files$
```

```

seed@VM: ~/.../Files$ cat task5.txt
Virat Kohli is one of the greatest cricketers of the modern era. Born on November 5, 1988, in Delhi, he grew from a young boy passionate about cricket into a global sporting icon. Known as the "Chase Master," Kohli has redefined batting with his incredible consistency, fitness, and hunger for runs. His aggressive style, quick running between the wickets, and ability to dominate bowlers in all formats of the game make him a nightmare for opponents.

Kohli has led India to many memorable victories, including historic wins overseas. As captain, he instilled discipline and self-belief in the team, emphasizing fitness as a key part of success. Under his leadership, India became the top-ranked Test side for several years. Off the field, his commitment to fitness has inspired millions of fans, and his intensity has set new standards for professionalism in cricket.

Beyond cricket, Kohli is admired for his philanthropic work through the Virat Kohli Foundation, which supports underprivileged children. His journey shows how determination and resilience can turn talent into greatness. Today, he remains not just a cricketer, but a role model for aspiring athletes around the world.

[09/18/25]seed@VM:~/.../Files$ openssl enc -aes-128-cfb -e -in task5.txt -out task5.bin -K 0011223344556677889aabbccddeff -iv 0102030405060708090a0b0c0d0e0f
hex string is too short, padding with zero bytes to length
[09/18/25]seed@VM:~/.../Files$ printf "\x01" | dd of=task5.bin bs=1 seek=54 count=1 conv=notrunc,fdatasync
1+0 records in
1+0 records out
1 byte copied, 0.00199313 s, 0.5 kB/s
[09/18/25]seed@VM:~/.../Files$ openssl enc -aes-128-cfb -d -in task5.bin -out task5_decrypted_cfb.txt -K 0011223344556677889aabbccddeff -iv 0102030405060708090a0b0c0d0e0f
hex string is too short, padding with zero bytes to length
[09/18/25]seed@VM:~/.../Files$ cat task5_decrypted_cfb.txt
Virat Kohli is one of the greatest cricketers of the modern era. Born on November 5, 1988, in Delhi, he grew from a young boy passionate about cricket into a global sporting icon. Known as the "Chase Master," Kohli has redefined batting with his incredible consistency, fitness, and hunger for runs. His aggressive style, quick running between the wickets, and ability to dominate bowlers in all formats of the game make him a nightmare for opponents.

Kohli has led India to many memorable victories, including historic wins overseas. As captain, he instilled discipline and self-belief in the team, emphasizing fitness as a key part of success. Under his leadership, India became the top-ranked Test side for several years. Off the field, his commitment to fitness has inspired millions of fans, and his intensity has set new standards for professionalism in cricket.

Beyond cricket, Kohli is admired for his philanthropic work through the Virat Kohli Foundation, which supports underprivileged children. His journey shows how determination and resilience can turn talent into greatness. Today, he remains not just a cricketer, but a role model for aspiring athletes around the world.

[09/18/25]seed@VM:~/.../Files$
```

```

seed@VM: ~/.../Files$ cat task5.txt
Virat Kohli is one of the greatest cricketers of the modern era. Born on November 5, 1988, in Delhi, he grew from a young boy passionate about cricket into a global sporting icon. Known as the "Chase Master," Kohli has redefined batting with his incredible consistency, fitness, and hunger for runs. His aggressive style, quick running between the wickets, and ability to dominate bowlers in all formats of the game make him a nightmare for opponents.

Kohli has led India to many memorable victories, including historic wins overseas. As captain, he instilled discipline and self-belief in the team, emphasizing fitness as a key part of success. Under his leadership, India became the top-ranked Test side for several years. Off the field, his commitment to fitness has inspired millions of fans, and his intensity has set new standards for professionalism in cricket.

Beyond cricket, Kohli is admired for his philanthropic work through the Virat Kohli Foundation, which supports underprivileged children. His journey shows how determination and resilience can turn talent into greatness. Today, he remains not just a cricketer, but a role model for aspiring athletes around the world.

[09/18/25]seed@VM:~/.../Files$ openssl enc -aes-128-ofb -e -in task5.txt -out task5.bin -K 0011223344556677889aabbccddeff -iv 0102030405060708090a0b0c0d0e0f
hex string is too short, padding with zero bytes to length
[09/18/25]seed@VM:~/.../Files$ printf "\x01" | dd of=task5.bin bs=1 seek=54 count=1 conv=notrunc,fdatasync
1+0 records in
1+0 records out
1 byte copied, 0.0123154 s, 0.1 kB/s
[09/18/25]seed@VM:~/.../Files$ openssl enc -aes-128-ofb -d -in task5.bin -out task5_decrypted_ofb.txt -K 0011223344556677889aabbccddeff -iv 0102030405060708090a0b0c0d0e0f
hex string is too short, padding with zero bytes to length
[09/18/25]seed@VM:~/.../Files$ cat task5_decrypted_ofb.txt
Virat Kohli is one of the greatest cricketers of the modern era. Born on November 5, 1988, in Delhi, he grew from a young boy passionate about cricket into a global sporting icon. Known as the "Chase Master," Kohli has redefined batting with his incredible consistency, fitness, and hunger for runs. His aggressive style, quick running between the wickets, and ability to dominate bowlers in all formats of the game make him a nightmare for opponents.

Kohli has led India to many memorable victories, including historic wins overseas. As captain, he instilled discipline and self-belief in the team, emphasizing fitness as a key part of success. Under his leadership, India became the top-ranked Test side for several years. Off the field, his commitment to fitness has inspired millions of fans, and his intensity has set new standards for professionalism in cricket.

Beyond cricket, Kohli is admired for his philanthropic work through the Virat Kohli Foundation, which supports underprivileged children. His journey shows how determination and resilience can turn talent into greatness. Today, he remains not just a cricketer, but a role model for aspiring athletes around the world.

[09/18/25]seed@VM:~/.../Files$
```

Observations:

- ECB: Entire 4th block of plaintext corrupted; all other blocks intact.
- CBC: 4th block completely corrupted, next block has some flipped bytes, rest unchanged.
- CFB: 4th block has bit flips, all subsequent blocks unreadable.
- OFB: Only flipped bit in 4th block changed, rest entirely intact.

Explanations:

- ECB: Every block independently → corruption confined to a single block.
- CBC: $P_i = D(C_i) \oplus C_{i-1}$, so corruption of C_i destroys block i . P_{i+1} depends on C_i , so same bit positions flipped. Beyond that, ciphertext fine.
- CFB: Keystream based on ciphertext feedback. Corruption of C_i changes all subsequent keystreams → all subsequent blocks destroyed.
- OFB: Keystream independent of ciphertext. Flipping a ciphertext bit flips only the corresponding plaintext bit.

Task 6: Initial Vector (IV) and Common Mistakes

Task 6.1. IV Experimented

Steps Performed:

1. Created a file task6.txt to store a small plain text.
2. Encrypted task6.txt with aes-128-cbc mode that uses an IV, once with IV1 and once with IV2.
3. I repeated the same by encrypting the same plain text with same key but reused same IV value for both encryptions and observed results using xxd.

Screenshots:

```
[09/20/25] seed@VM:~/.../Files$ cat task6.txt
Computer Security CSE565
[09/20/25] seed@VM:~/.../Files$ openssl enc -aes-128-cbc -e -in task6.txt -out task6_iv1.bin -K 00112233445566778889aabcccddeeff -iv 0102030405060708090a0b0c0d0e0f
hex string is too short, padding with zero bytes to length
[09/20/25] seed@VM:~/.../Files$ xxd task6_iv1.bin
00000000: cbf5 4d95 d52b f2ab 7af9 9ba1 34c3 d21e ..M..+.z....4...
00000010: 17b2 ea23 9a2b 6c11 de92 9b39 ad31 f47b ...#.+l....9.1.{}
[09/20/25] seed@VM:~/.../Files$ openssl enc -aes-128-cbc -e -in task6.txt -out task6_iv2.bin -K 00112233445566778889aabcccddeeff -iv 1112131415161718191a1b1c1d1e1f
hex string is too short, padding with zero bytes to length
[09/20/25] seed@VM:~/.../Files$ xxd task6_iv2.bin
00000000: 26e0 8ced 62f5 e8e0 fa55 c587 51fe cdde &....b....U..Q...
00000010: 066b d00b 941a 569e b709 e755 5912 1816 .k....V....UY...
```

```

seed@VM: ~/.../Files$ cat task6.txt
Computer Security CSE565
[09/20/25]seed@VM:~/.../Files$ openssl enc -aes-128-cbc -e -in task6.txt -out task6_1
.bin -K 00112233445566778889aabcccddeeff -iv 0102030405060708090a0b0c0d0e0fhex string
is too short, padding with zero bytes to length
[09/20/25]seed@VM:~/.../Files$ xxd task6_1.bin
00000000: cbf5 4d95 d52b f2ab 7af9 9ba1 34c3 d21e ..M..+..z...4...
00000010: 17b2 ea23 9a2b 6c11 de92 9b39 ad31 f47b ...#.+l....9.1.{

[09/20/25]seed@VM:~/.../Files$ openssl enc -aes-128-cbc -e -in task6.txt -out task6_2
.bin -K 00112233445566778889aabcccddeeff -iv 0102030405060708090a0b0c0d0e0fhex string
is too short, padding with zero bytes to length
[09/20/25]seed@VM:~/.../Files$ xxd task6_2.bin
00000000: cbf5 4d95 d52b f2ab 7af9 9ba1 34c3 d21e ..M..+..z...4...
00000010: 17b2 ea23 9a2b 6c11 de92 9b39 ad31 f47b ...#.+l....9.1.{

[09/20/25]seed@VM:~/.../Files$ █

```

Observations & Explanation:

- Using different IVs ($IV_1 \neq IV_2$) it produces different ciphertexts. IV's purpose is to randomize the first block so identical plaintexts do not result in identical ciphertexts.
- With the same IV and with the same key and same plaintext, we obtain the same ciphertexts (for deterministic modes like CBC) or you will reuse the same keystream again (for stream/stream-like modes like OFB/CTR) i.e. the ciphertexts are identical or reveal relationships between plaintexts.

Task 6.2. Common Mistake: Use the Same IV

Steps Performed:

1. Converted the given Plaintext P1 and Ciphertexts (C1 & C2) into byte arrays
2. XOR C1 and P1 to get the key stream.
3. Since C2 was encrypted with same IV(key), we can XOR the same key stream with C2 to get P2
4. All the implementation is updated in the sample_code.py file and executed it for the output

Screenshots:

```
seed@VM: ~/.../Files$ cat sample_code.py
#!/usr/bin/python3

# XOR two bytearrays
def xor(first, second):
    return bytearray(x^y for x,y in zip(first, second))

P1 = "This is a known message!"
C1 = "a469b1c502c1cab966965e50425438e1bb1b5f9037a4c159"
C2 = "bf73bcd3509299d566c35b5d450337e1bb175f903fafc159"

# Convert ascii string to bytearray
D1 = bytes(P1, 'utf-8')

# Convert hex string to bytearray
D2 = bytearray.fromhex(C1)
D3 = bytearray.fromhex(C2)

K = xor(D1, D2)
P2 = xor(D3, K)

print("Recovered P2 (hex):", P2.hex())
print("Recovered P2 (ascii):", P2.decode('utf-8'))
[09/20/25] seed@VM:~/.../Files$ ./sample_code.py
Recovered P2 (hex): 4f726465723a204c61756e63682061206d697373696c6521
Recovered P2 (ascii): Order: Launch a missile!
[09/20/25] seed@VM:~/.../Files$
```

Observations:

- If the same IV were used in OFB, the entire plaintext P2 could be constructed by P1 and C1.
- In CFB, only part of P2 would be revealed since the errors carry over through the feedback.

Explanation:

- Using a common IV in OFB is insecure as it produces the same keystream; a single plain text reveals all other messages encrypted with it.
- IVs should be different to maintain confidentiality.
- In CFB, the feedback mechanism makes the information about one plaintext only reveal partial information in subsequent ciphertexts limiting the damage.

Task 6.3. Common Mistake: Use a Predictable IV

Steps performed:

- ## 1. Connected to the oracle with:

nc 10.9.0.80 3000

2. Got the Bob's ciphertext, IV used, and next IV printed out by the oracle.
 3. Constructed a plaintext block P_{send} with:

$$P_send = IV_next \oplus IV_secret \oplus pad(guess)$$

4. where guess was "Yes" or "No" which were converted into bytes and padding used PKCS#7.
 5. Passed the constructed plaintext to the oracle and received the ciphertext.
 6. Compared the ciphertext of the oracle with the target ciphertext of Bob to ensure the secret.

Screenshots:

Observations:

- Since the IVs were predictable, a compromised plaintext could be encrypted to reveal whether Bob's secret message was "Yes" or "No."
- By comparing the generated hex code to the given, I have found that the Bob's secret message was "Yes".
- The oracle returned ciphertext that was the same as the secret upon correct guess.

Explanation:

- Exploitable IVs make AES-CBC vulnerable to a chosen-plaintext attack.
- While AES is secure, knowing the next IV allows an attacker to dictate encryption and obtain secret information.
- IVs must be random and unpredictable in CBC mode.

Task 7: Programming using the Crypto Library

Steps Performed:

1. Saved the configuration plaintext "This is a top secret." in a file to ensure exact length.
2. Read an English word dictionary from words.txt.
3. For each word:
 - Padded it with # to 16 bytes to form the AES-128 key.
 - Used the OpenSSL crypto library (EVP API) to encrypt the plaintext with the key and given IV.
 - Verified the resulting ciphertext against the provided target ciphertext.
4. Shutdown when a match was found, displaying the correct key.

Screenshots:

The screenshot shows a terminal window with a dark theme. The title bar says "seed@VM: ~/.../Files". The command history at the top of the terminal shows:

```
[09/20/25] seed@VM:~/.../Files$ nano myenc.c
[09/20/25] seed@VM:~/.../Files$ gcc -o myenc myenc.c -lcrypto
[09/20/25] seed@VM:~/.../Files$ ./myenc
Key found! Word = Syracuse
[09/20/25] seed@VM:~/.../Files$
```

Code Snippet:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <openssl/evp.h>
#include <openssl/aes.h>

int encrypt(unsigned char *plaintext, int plaintext_len, unsigned char *key,
           unsigned char *iv, unsigned char *ciphertext) {
    EVP_CIPHER_CTX *ctx;
    int len, ciphertext_len;

    if(!(ctx = EVP_CIPHER_CTX_new())) return -1;

    EVP_EncryptInit_ex(ctx, EVP_aes_128_cbc(), NULL, key, iv);

    EVP_EncryptUpdate(ctx, ciphertext, &len, plaintext, plaintext_len);
    ciphertext_len = len;

    EVP_EncryptFinal_ex(ctx, ciphertext + len, &len);
    ciphertext_len += len;

    EVP_CIPHER_CTX_free(ctx);

    return ciphertext_len;
}

int main() {
    unsigned char *iv = (unsigned char
*)"\xaa\xbb\xcc\xdd\xee\xff\x00\x99\x88\x77\x66\x55\x44\x33\x22\x11";

    FILE *fp = fopen("task7.txt", "rb");
    if (!fp) { perror("task7.txt"); exit(1); }

    fseek(fp, 0, SEEK_END);
```

```
long filesize = ftell(fp);
rewind(fp);

unsigned char *plaintext = malloc(filesize);
fread(plaintext, 1, filesize, fp);
fclose(fp);

unsigned char target_cipher[] = {
    0x76,0x4a,0xa2,0x6b,0x55,0xa4,0xda,0x65,
    0x4d,0xf6,0xb1,0x9e,0x4b,0xce,0x00,0xf4,
    0xed,0x05,0xe0,0x93,0x46,0xfb,0x0e,0x76,
    0x25,0x83,0xcb,0x7d,0xa2,0xac,0x93,0xa2
};

FILE *f = fopen("words.txt", "r");
if (!f) { perror("words.txt"); exit(1); }

char word[64];
while (fgets(word, sizeof(word), f)) {
    word[strcspn(word, "\n")] = 0;
    int wlen = strlen(word);
    if (wlen == 0 || wlen > 16) continue;

    unsigned char key[16];
    memset(key, 0, 16);
    memcpy(key, word, wlen);
    for (int i = wlen; i < 16; i++) key[i] = '#';

    unsigned char ciphertext[128];
    int clen = encrypt(plaintext, filesize, key, iv, ciphertext);

    if (clen == sizeof(target_cipher) &&
        memcmp(ciphertext, target_cipher, clen) == 0) {
        printf("Key found! Word = %s\n", word);
        break;
    }
}

fclose(f);
return 0;
}
```

Code Explanation:

- The program uses `fopen()` to open the dictionary file (`words.txt`) in read mode and `fgets()` to read a word by a time line by line. The newline characters are stripped off using `strcspn()` to correctly pad the key.
- Candidate words with a size of under 16 bytes are padded with `#` to form a valid 16-byte AES-128 key, saved in an array `unsigned char key[16]`.
- The `encrypt()` function encapsulates AES-128-CBC encryption using the OpenSSL EVP API.
- `EVP_CIPHER_CTX_new()` sets up a new encryption context.
- `EVP_EncryptInit_ex()` sets the context to use the AES-128-CBC cipher, the key created, and the provided IV.
- `EVP_EncryptUpdate()` encrypts the plaintext bytes and stores the intermediate ciphertext.
- `EVP_EncryptFinal_ex()` finishes encryption, performing padding and appending any remaining bytes to the ciphertext.
- `EVP_CIPHER_CTX_free()` releases memory held by the context.
- `memcmp()` is used to byte-by-byte compare the produced ciphertext with the destination ciphertext. If they match, the correct key has been found.
- `strlen()` is used to determine the plaintext length, and `memcpy()/memset()` are used to copy and initialize the array of the key.
- The application goes through all dictionary words, encrypting and comparing, using a dictionary attack successfully without attempting all 128-bit keys by brute force.

Observations:

- A single dictionary word, padded and used as the key, produced ciphertext indistinguishable from the target.
- The program could recover the encryption key without trying all 128-bit possibilities using brute-force.

Explanation:

- AES-128-CBC encryption is key and IV-deterministic.
- By trying all likely English words (padded to 16 bytes), the correct key can be decided efficiently.
- Keys derived from low-entropy or predictable sources (like words in a dictionary) are vulnerable to dictionary attacks.

Resources:

- [Week-3 Class-1-2 Cryptography.pptx](#)
- [EVP_EncryptInit - OpenSSL Documentation](#)
- [https://medium.com/@amit.kulkarni/encrypting-decrypting-a-file-using-openssl-evp-b26e0e4d28d4](#)
- [https://www.tutorialspoint.com/cryptography/cipher_feedback_mode.htm](#)
- [https://www.geeksforgeeks.org/c/basics-file-handling-c/](#)
- [https://linuxcommand.org/lc3_man_page_index.php](#)