

IS Assignment 1

K21-4503 Muhammad Tahir
K21-3279 Insha Javed

Task01:

```
[09/12/24] seed@VM:~/.../Labsetup$ gedit Task1.py
^C
[09/12/24] seed@VM:~/.../Labsetup$ ./Task1.py
vtxmxdjwaniobpsafqeuzhlycrkg
[09/12/24] seed@VM:~/.../Labsetup$ gedit Task1.py^C
[09/12/24] seed@VM:~/.../Labsetup$ tr [:upper:][:lower:] < article.txt > lowercase.txt
bash: article.txt: No such file or directory
[09/12/24] seed@VM:~/.../Labsetup$ gedit article.txt
[09/12/24] seed@VM:~/.../Labsetup$ tr [:upper:][:lower:] < article.txt > lowercase.txt
tr: missing operand after '[:upper:][:lower:]'
Two strings must be given when translating.
Try 'tr --help' for more information.
[09/12/24] seed@VM:~/.../Labsetup$ tr [:upper:] [:lower:] < article.txt > lowercase.txt
[09/12/24] seed@VM:~/.../Labsetup$ ls
article.txt      encryption_oracle  lowercase.txt
docker-compose.yml  Files           Task1.py
[09/12/24] seed@VM:~/.../Labsetup$ gedit lowercase.txt
[09/12/24] seed@VM:~/.../Labsetup$ tr -cd '[a-z][\n][:space:]' < lowercase.txt > plaintext.txt
tr-cd: command not found
[09/12/24] seed@VM:~/.../Labsetup$ tr -cd '[a-z][\n][:space:]' < lowercase.txt > plaintext.txt
[09/12/24] seed@VM:~/.../Labsetup$ gedit lowercase.txt
```

```

tr-cd: command not found
[09/12/24] seed@VM:~/.../Labsetup$ tr -cd '[a-z][\n][:space:]' < lowercase.txt > plaintext.txt
[09/12/24] seed@VM:~/.../Labsetup$ gedit lowercase.txt
[09/12/24] seed@VM:~/.../Labsetup$ tr 'abcdefghijklmnopqrstuvwxyz' 'sxtrwinqbedpvgkfmalyuojzc' \
> < plaintext.txt > ciphertext.txt
tr: missing operand after '\abcdefghijklmnopqrstuvwxyz\' `sxtrwinqbedpvgkfmalyuojzc\''
Two strings must be given when translating.
Try 'tr --help' for more information.
[09/12/24] seed@VM:~/.../Labsetup$ tr 'abcdefghijklmnopqrstuvwxyz' 'sxtrwinqbedpvgkfmalyuojzc' \ < plaintext.txt > ciphertext.txt
tr: extra operand ''
Try 'tr --help' for more information.
[09/12/24] seed@VM:~/.../Labsetup$ tr 'abcdefghijklmnopqrstuvwxyz' 'sxtrwinqbedpvgkfmalyuojzc' \ < plaintext.txt > ciphertext.txt
tr: extra operand ''
Try 'tr --help' for more information.
[09/12/24] seed@VM:~/.../Labsetup$ tr 'abcdefghijklmnopqrstuvwxyz' 'sxtrwinqbedpvgkfmalyuojzc' < plaintext.txt > ciphertext.txt
[09/12/24] seed@VM:~/.../Labsetup$ ls
article.txt docker-compose.yml Files plaintext.txt
ciphertext.txt encryption_oracle lowercase.txt Task1.py
[09/12/24] seed@VM:~/.../Labsetup$ gedit ciphertext.txt

```

mysolve.txt

```

1 THE OSCARS TURN ON SUNDAY WHICH SEEMS ABOUT RIGHT AFTER THIS LONG
2 AWARDS TRIP THE BAGGER FEELS LIKE A NONAGENARIAN TOO
3
4 THE AWARDS RACE WAS BOOKENDED BY THE DEMISE OF HARVEY WEINSTEIN AT
5 AND THE APPARENT IMPLOSION OF HIS FILM COMPANY AT THE END AND IT WAS
6 THE EMERGENCE OF METOO TIMES UP BLACKGOWN POLITICS ARMCANDY ACTIVISM
7 A NATIONAL CONVERSATION AS BRIEF AND MAD AS A FEVER DREAM ABOUT WHO
8 OUGHT TO BE A PRESIDENT WINFREY THE SEASON DIDNT JUST SEEM EXTRA LONG
9 EXTRA LONG BECAUSE THE OSCARS WERE MOVED TO THE FIRST WEEKEND IN MARCH
10 AVOID CONFLICTING WITH THE CLOSING CEREMONY OF THE WINTER OLYMPICS
11 PYEONGCHANG
12
13 ONE BIG QUESTION SURROUNDING THIS YEARS ACADEMY AWARDS IS HOW OR IF
14 CEREMONY WILL ADDRESS METOO ESPECIALLY AFTER THE GOLDEN GLOBES WHICH
15 A JUBILANT COMINGOUT PARTY FOR TIMES UP THE MOVEMENT SPEARHEADED BY
16 POWERFUL HOLLYWOOD WOMEN WHO HELPED RAISE MILLIONS OF DOLLARS TO FIGHT
17 HARASSMENT AROUND THE COUNTRY
18
19 SIGNALING THEIR SUPPORT GOLDEN GLOBES ATTENDEES SWATHED THEMSELVES
20 SPOTTED LAPEL PINS AND SOUNDED OFF ABOUT SEXIST POWER IMBALANCES FROM
21 CARPET AND THE STAGE ON THE AIR E WAS CALLED OUT ABOUT PAY INEQUALITY
22 ITS FORMER ANCHOR CATT SADLER QUIT ONCE SHE LEARNED THAT SHE WAS MORE
23 LESS THAN A MALE COHOST AND DURING THE CEREMONY NATALIE PORTMAN TOOK
24 AND SATISFYING DIG AT THE ALLMALE ROSTER OF NOMINATED DIRECTORS HOW
25 THAT BE TOPPED
26
27 AS IT TURNS OUT AT LEAST IN TERMS OF THE OSCARS IT PROBABLY WONT BE
28
29 WOMEN INVOLVED IN TIMES UP SAID THAT ALTHOUGH THE GLOBES SIGNIFIED
30 INITIATIVES LAUNCH THEY NEVER INTENDED IT TO BE JUST AN AWARDS SEASON
31 CAMPAIGN OR ONE THAT BECAME ASSOCIATED ONLY WITH REDCARPET ACTIONS
32 A SPOKESWOMAN SAID THE GROUP IS WORKING BEHIND CLOSED DOORS AND HAD
33 AMASSED MILLION FOR ITS LEGAL DEFENSE FUND WHICH AFTER THE GLOWS

```

seed@VM:~/.../Files

```

[09/12/24] seed@VM:~/.../Files$ tr 'ytnvxbhmqzis' 'THEAOFRISUL' < ciphertext.txt > mysolve.txt
[09/12/24] seed@VM:~/.../Files$ tr 'ytnvxbhmqzir' 'THEAOFRISULG' < ciphertext.txt > mysolve.txt
[09/12/24] seed@VM:~/.../Files$ tr 'ytnvxbhmqzirs' 'THEAOFRISULGK' < ciphertext.txt > mysolve.txt
[09/12/24] seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcu' 'THEAOFRISULGKDNY' < ciphertext.txt > mysolve.txt
[09/12/24] seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcud' 'THEAOFRISULGKDNYC' < ciphertext.txt > mysolve.txt
[09/12/24] seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcudgj' 'THEAOFRISULGKDNYBQCPWVX' < ciphertext.txt > mysolve.txt
[09/12/24] seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcudgjael' 'THEAOFRISULGKDNYBQCPWVJ' < ciphertext.txt > mysolve.txt
[09/12/24] seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcudgjaelk' 'THEAOFRISULGKDNYBQCPWX' < ciphertext.txt > mysolve.txt
[09/12/24] seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcudgjaelkf' 'THEAOFRISULGKDNYBQCPWXV' < ciphertext.txt > mysolve.txt
[09/12/24] seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcudgjaelfu' 'THEAOFRISULGKDNYBQCPWXVJ' < ciphertext.txt > mysolve.txt
[09/12/24] seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcudgjaelkf' 'THEAOFRISULGKDNYBQCPWXV' < ciphertext.txt > mysolve.txt
[09/12/24] seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcudgjaelko' 'THEAOFRISULGKDNYBQCPWXVJ' < ciphertext.txt > mysolve.txt
[09/13/24] seed@VM:~/.../Files$ echo "K21-4503, K21-3279"
[09/13/24] seed@VM:~/.../Files$ K21-4503, K21-3279
[09/13/24] seed@VM:~/.../Files$ 

```

This is the final result:

```
tr 'ytnvxvhmqzirspcudgjaelkfo' 'THEAOFRISULGKDMNYBQCPWXVJ'  
< ciphertext.txt > mysolve.txt
```

Task02:

The screenshot shows a desktop environment with a file manager window and a terminal window.

File Manager Window: The title bar says "Terminal". The sidebar shows "Recent" and "Starred" items. The main area contains files: cipher_1.bin, cipher_2.bin, cipher_3.bin, cipher_4.bin, decryptMsg_1.txt, decryptMsg_2.txt, and decryptMsg_3.txt. Below these are two partially visible files: decryptMsg_4.txt and words.txt.

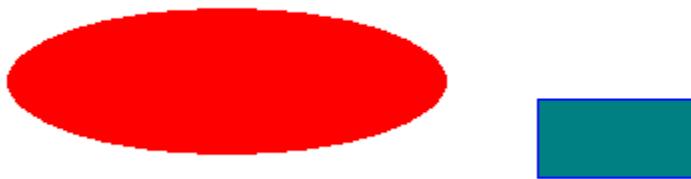
Terminal Window: The title bar says "seed@VM: ~.../Task02". The terminal output is as follows:

```
iv 01020304050607080102030405060708  
[09/13/24] seed@VM:~/.../Task02$ openssl enc -bf-cbc -e -in words.txt -out cipher_2.bin -K 00112233445566778889aabbccddeeff -iv 01020304050607080102030405060708  
hex string is too long, ignoring excess  
[09/13/24] seed@VM:~/.../Task02$ openssl enc -bf-cbc -d -in cipher_2.bin -out decryptMsg_2.txt -K 00112233445566778889aabbccddeeff -iv 01020304050607080102030405060708  
hex string is too long, ignoring excess  
[09/13/24] seed@VM:~/.../Task02$ openssl enc -aes-128-cfb -e -in words.txt -out cipher_3.bin -K 00112233445566778889aabbccddeeff -iv 01020304050607080102030405060708  
[09/13/24] seed@VM:~/.../Task02$ openssl enc -aes-128-cfb -d -in cipher_3.bin -out decryptMsg_3.txt -K 00112233445566778889aabbccddeeff -iv 01020304050607080102030405060708  
[09/13/24] seed@VM:~/.../Task02$ openssl enc -aes-128-ecb -e -in words.txt -out cipher_4.bin -K 00112233445566778889aabbccddeeff -iv 01020304050607080102030405060708  
warning: iv not used by this cipher  
[09/13/24] seed@VM:~/.../Task02$ openssl enc -aes-128-ecb -d -in cipher_4.bin -out decryptMsg_4.txt -K 00112233445566778889aabbccddeeff -iv 01020304050607080102030405060708  
warning: iv not used by this cipher  
[09/13/24] seed@VM:~/.../Task02$
```

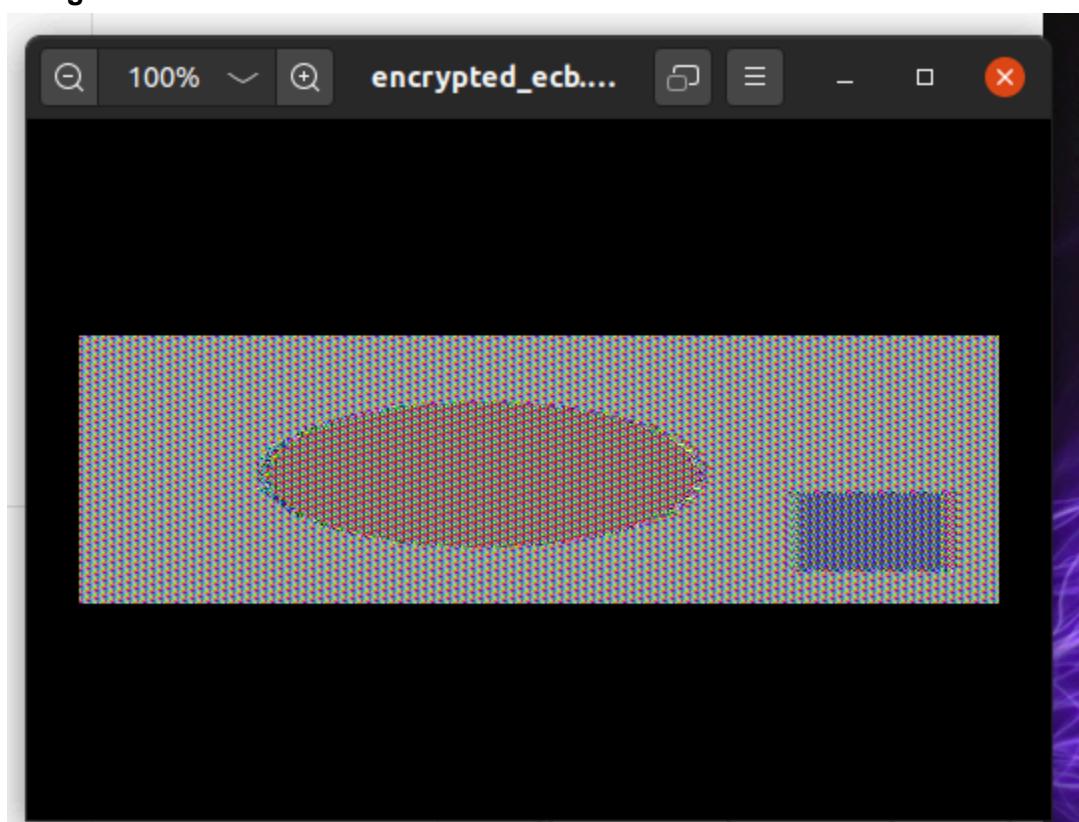
Task03:

```
seed@VM: ~/.../Task03$ head -c 54 pic_original.bmp > header  
[09/14/24] seed@VM:~/.../Task03$ tail -c +55 pic_original.bmp > body  
[09/14/24] seed@VM:~/.../Task03$ openssl enc -aes-128-ecb -in body -  
out body_ecb.enc -K 888e4d66fc63d6c7d80e92e80380bb08  
[09/14/24] seed@VM:~/.../Task03$ openssl enc -aes-128-ecb -in body -  
out body_cbc.enc -K 888e4d66fc63d6c7d80e92e80380bb08 -iv 026483c770  
31f4d8  
warning: iv not used by this cipher  
[09/14/24] seed@VM:~/.../Task03$ openssl enc -aes-128-ecb -in body -  
out body_ecb.enc -K 888e4d66fc63d6c7d80e92e80380bb08 -iv 026483c770  
31f4d8  
warning: iv not used by this cipher  
[09/14/24] seed@VM:~/.../Task03$ openssl enc -aes-128-cbc -in body -  
out body_cbc.enc -K 888e4d66fc63d6c7d80e92e80380bb08 -iv 026483c770  
31f4d8  
hex string is too short, padding with zero bytes to length  
[09/14/24] seed@VM:~/.../Task03$ cat header body_ecb.enc > encrypted  
_ecb.bmp  
[09/14/24] seed@VM:~/.../Task03$ cat header body_cbc.enc > encrypted  
_cbc.bmp  
[09/14/24] seed@VM:~/.../Task03$ echo "K21-4503, K21-3279"  
K21-4503, K21-3279  
[09/14/24] seed@VM:~/.../Task03$
```

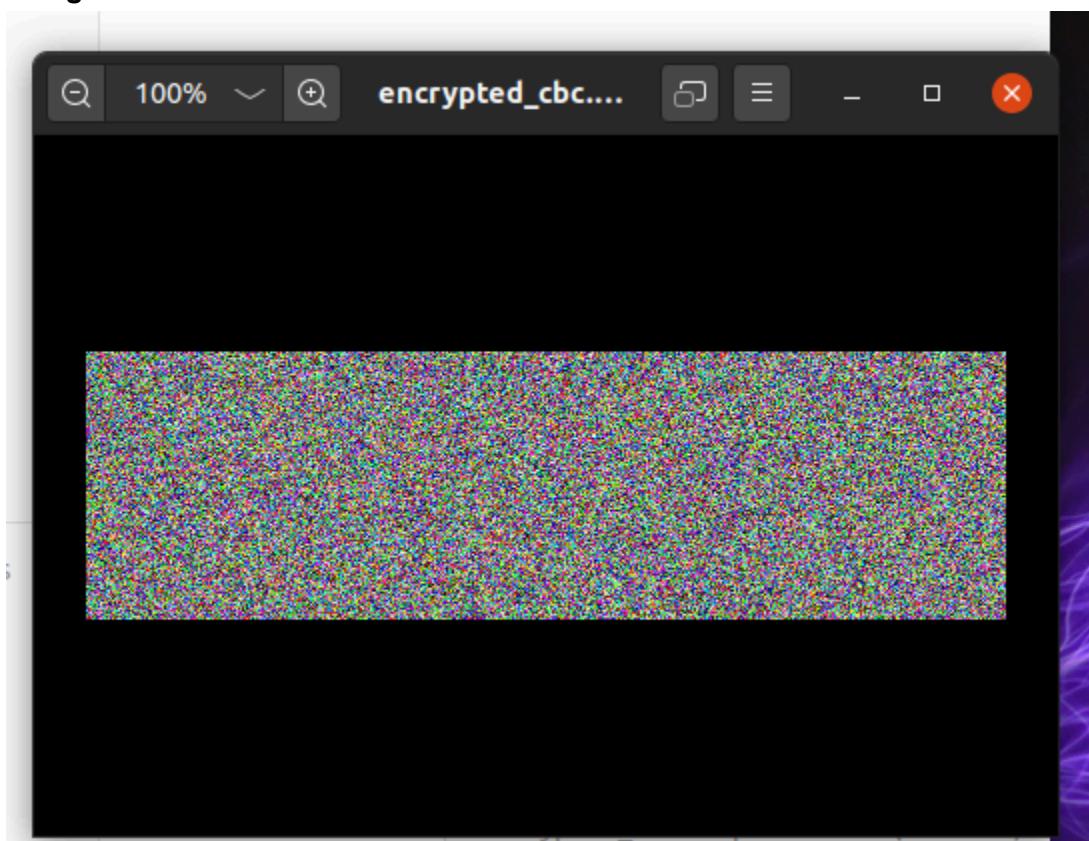
Original Pic1:



Using ECB:



Using CBC:



Observation:

I performed the following steps:

1. Extracted the Header and Body:

- i) Used head to extract the first 54 bytes (header) from Original_Pic.bmp.
- ii) Used tail to extract the image data from byte 55 to the end of the file from Original_Pic.bmp.

2. Combined Header and Encrypted Body:

I combined the header from Original_Pic.bmp with the encrypted body from Original_Pic.bmp using the Cat command to create new.bmp.

3. Verified File Integrity:

I used the Bless hex editor to confirm that new.bmp had the header from Original_Pic.bmp and the encrypted body from Original_Pic.bmp.

Key Observation:

The image displayed in new.bmp shows the intact header from Original_Pic.bmp, which preserves the structural information. However, the visual content is scrambled and unrecognizable due to the encryption. This confirms that encryption modifies the image data, while the header remains unchanged. The presence of the original header does not affect the scrambled appearance of the encrypted data, demonstrating that the encryption, independent of the header, directly impacts the image's visual content.

Task04:

Part 1:

- **Electronic Codebook:** Padding is required because plaintext must be divided into fixed-size blocks. If the plaintext size is not a multiple of the block size, padding is added.
- **Cipher Block Chaining:** Padding is also necessary for the same reason as ECB. Plaintext must fit the block size, and any partial block requires padding.
- **Cipher Feedback:** Padding is **not** required. CFB is a stream cipher mode, meaning encryption happens bit by bit or byte by byte, so there's no need to pad the plaintext to fit block size.
- **Output Feedback:** Padding is **not** required. Like CFB, OFB is a stream cipher mode, which doesn't work in fixed-size blocks.

Part 2:

Using “-aes-128-cbc”

The screenshot shows a desktop environment with a file manager window on the left and a terminal window on the right. The terminal window displays a series of openssl commands used for encryption and decryption.

```
[09/13/24] seed@VM:~/.../Task04$ echo -n "12345" > f1.txt
[09/13/24] seed@VM:~/.../Task04$ echo -n "1234567890" > f2.txt
[09/13/24] seed@VM:~/.../Task04$ echo -n "1234567890123456" > f3.txt
[09/13/24] seed@VM:~/.../Task04$ openssl enc -aes-128-cbc -e -in f1.txt -out f1.enc -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
iv undefined
[09/13/24] seed@VM:~/.../Task04$ openssl enc -aes-128-cbc -e -in f1.txt -out f1.enc -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
hex string is too short, padding with zero bytes to length
[09/13/24] seed@VM:~/.../Task04$ openssl enc -aes-128-cbc -d -in f1.enc -out f1_output.txt -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
hex string is too short, padding with zero bytes to length
[09/13/24] seed@VM:~/.../Task04$ openssl enc -aes-128-cbc -d -in f2.txt -out f2.enc -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
hex string is too short, padding with zero bytes to length
[09/13/24] seed@VM:~/.../Task04$ openssl enc -aes-128-cbc -d -in f2.enc -out f2_output.txt -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
hex string is too short, padding with zero bytes to length
[09/13/24] seed@VM:~/.../Task04$ openssl enc -aes-128-cbc -d -in f3.txt -out f3.enc -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
hex string is too short, padding with zero bytes to length
[09/13/24] seed@VM:~/.../Task04$ openssl enc -aes-128-cbc -d -in f3.enc -out f3_output.txt -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
[09/13/24] seed@VM:~/.../Task04$ hexdump -C f1_output.txt
00000000  31 32 33 34 35          | 12345 |
00000005
[09/13/24] seed@VM:~/.../Task04$ hexdump -C f2_output.txt
00000000  31 32 33 34 35 36 37 38 39 30          | 123456
7890|
0000000a
[09/13/24] seed@VM:~/.../Task04$ hexdump -C f3_output.txt
00000000  31 32 33 34 35 36 37 38 39 30 31 32 33 34 35 36 7890123456|
00000010
[09/13/24] seed@VM:~/.../Task04$
```

Using “-aes-128-ecb”:

```
[09/13/24]seed@VM:~/.../Task04$ hexdump -C f3_output.txt
00000000 31 32 33 34 35 36 37 38 39 30 31 32 33 34 35 36 |123456
7890123456|
00000010
[09/13/24]seed@VM:~/.../Task04$ openssl enc -aes-128-ecb -e -in f3.txt -out f1_ecb.enc -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
warning: iv not used by this cipher
[09/13/24]seed@VM:~/.../Task04$ openssl enc -aes-128-ecb -d -in f1_ecb.enc -out f3_output_ecb.txt -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657 -nopad
warning: iv not used by this cipher
[09/13/24]seed@VM:~/.../Task04$ openssl enc -aes-128-ecb -e -in f3.txt -out f3_ecb.enc -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
warning: iv not used by this cipher
[09/13/24]seed@VM:~/.../Task04$ openssl enc -aes-128-ecb -d -in f2.txt -out f2_ecb.enc -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
warning: iv not used by this cipher
[09/13/24]seed@VM:~/.../Task04$ openssl enc -aes-128-ecb -d -in f1_ecb.enc -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
warning: iv not used by this cipher
[09/13/24]seed@VM:~/.../Task04$ openssl enc -aes-128-ecb -d -in f1_ecb.enc -out f2_output_ecb.txt -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657 -nopad
warning: iv not used by this cipher
[09/13/24]seed@VM:~/.../Task04$ hexdump -C f3_output_ecb.txt
00000000 31 32 33 34 35 36 37 38 39 30 31 32 33 34 35 36 |123456
7890123456|
00000010 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 |.....
.....|
00000020
[09/13/24]seed@VM:~/.../Task04$ hexdump -C f2_output_ecb.txt
00000000 31 32 33 34 35 36 37 38 39 30 06 06 06 06 06 06 |123456
7890.....|
00000010
[09/13/24]seed@VM:~/.../Task04$ hexdump -C f1_output_ecb.txt
00000000 31 32 33 34 35 0b |12345.
```

```
aae5bb657
warning: iv not used by this cipher
[09/13/24]seed@VM:~/.../Task04$ openssl enc -aes-128-ecb -d -in f2_ecb.enc -out f2_output_ecb.txt -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657 -nopad
warning: iv not used by this cipher
[09/13/24]seed@VM:~/.../Task04$ openssl enc -aes-128-ecb -e -in f1.txt -out f1_ecb.enc -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
warning: iv not used by this cipher
[09/13/24]seed@VM:~/.../Task04$ openssl enc -aes-128-ecb -d -in f1_ecb.enc -out f2_output_ecb.txt -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657 -nopad
warning: iv not used by this cipher
[09/13/24]seed@VM:~/.../Task04$ hexdump -C f3_output_ecb.txt
00000000 31 32 33 34 35 36 37 38 39 30 31 32 33 34 35 36 |123456
7890123456|
00000010 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 |.....
.....|
00000020
[09/13/24]seed@VM:~/.../Task04$ hexdump -C f2_output_ecb.txt
00000000 31 32 33 34 35 36 37 38 39 30 06 06 06 06 06 06 |123456
7890.....|
00000010
[09/13/24]seed@VM:~/.../Task04$ hexdump -C f1_output_ecb.txt
00000000 31 32 33 34 35 0b |12345.
```

Using “-aes-128-cfb”

The terminal window shows the following command sequence:

```
seed@VM:~/.../Task04$ openssl enc -aes-128-cfb -e -in f2.txt -out f2_cfb.enc -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
hex string is too short, padding with zero bytes to length
[09/13/24]seed@VM:~/.../Task04$ openssl enc -aes-128-cfb -d -in f2_cfb.enc -out f2_output_cfb.txt -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
hex string is too short, padding with zero bytes to length
[09/13/24]seed@VM:~/.../Task04$ openssl enc -aes-128-cfb -e -in f3.txt -out f3_cfb.enc -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
hex string is too short, padding with zero bytes to length
[09/13/24]seed@VM:~/.../Task04$ openssl enc -aes-128-cfb -d -in f3_cfb.enc -out f3_output_cfb.txt -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
hex string is too short, padding with zero bytes to length
[09/13/24]seed@VM:~/.../Task04$ hexdump -C f1_output_cfb.txt
00000000  31 32 33 34 35                                |12345|
00000005
[09/13/24]seed@VM:~/.../Task04$ hexdump -C f2_output_cfb.txt
00000000  31 32 33 34 35 36 37 38 39 30                |123456
7890|
0000000a
[09/13/24]seed@VM:~/.../Task04$ hexdump -C f3_output_cfb.txt
00000000  31 32 33 34 35 36 37 38 39 30 31 32 33 34 35 36 |123456
7890123456|
00000010
[09/13/24]seed@VM:~/.../Task04$
```

The terminal also displays the contents of the hexdump output for each file.

Using “-aes-128-ofb”

The terminal window shows the following command sequence:

```
-iv a6ea7f6aae5bb657 -nopad
hex string is too short, padding with zero bytes to length
[09/13/24]seed@VM:~/.../Task04$ openssl enc -aes-128-ofb -e -in f2.txt -out f2_ofb.enc -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
hex string is too short, padding with zero bytes to length
[09/13/24]seed@VM:~/.../Task04$ openssl enc -aes-128-ofb -d -in f2_ofb.enc -out f2_output_ofb.txt -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657 -nopad
hex string is too short, padding with zero bytes to length
[09/13/24]seed@VM:~/.../Task04$ openssl enc -aes-128-ofb -e -in f1.txt -out f1_ofb.enc -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657
hex string is too short, padding with zero bytes to length
[09/13/24]seed@VM:~/.../Task04$ openssl enc -aes-128-ofb -d -in f1_ofb.enc -out f1_output_ofb.txt -K 25b9aaea47b7ff50789cc64a3b65ef27 -iv a6ea7f6aae5bb657 -nopad
hex string is too short, padding with zero bytes to length
[09/13/24]seed@VM:~/.../Task04$ hexdump -C f1_output_ofb.txt
00000000  31 32 33 34 35                                |12345|
00000005
[09/13/24]seed@VM:~/.../Task04$ hexdump -C f2_output_ofb.txt
00000000  31 32 33 34 35 36 37 38 39 30                |123456
7890|
0000000a
[09/13/24]seed@VM:~/.../Task04$ hexdump -C f3_output_ofb.txt
00000000  31 32 33 34 35 36 37 38 39 30 31 32 33 34 35 36 |123456
7890123456|
00000010
[09/13/24]seed@VM:~/.../Task04$
```

The terminal window title is "seed@VM: ~/.../Task04". The current directory is indicated by the prompt "[09/13/24]seed@VM:~/.../Task04\$". The terminal window also shows file icons for f1.txt, f2.txt, and f3.txt on the left, and f2_ofb.enc, f2_ecb.enc, and f3_ofb.enc on the right.

Task05:

Expected Outcomes Before Experimentation:

1) Electronic Codebook Mode:

- a) Only the corrupted block (the one containing the 55th byte) will be affected.

2) Cipher Block Chaining Mode:

- a) The corrupted block and the next block will be affected.

3) Cipher Feedback Mode:

- a) The corrupted byte will affect the corresponding byte and some subsequent bytes.

4) Output Feedback Mode:

- a) The corrupted byte will affect the maximum bytes.

The screenshot shows a desktop environment with a file manager window and a terminal window.

File Manager Window: The title bar says "Task05". The contents show several files: "cipher.bin", "cipher_cbc.bin", "cipher_cfb.bin", "cipher_ecb.bin", "cipher_ofb.bin", and "textFile.txt".

Terminal Window: The title bar says "seed@VM: ~/.../Task05". The terminal output is as follows:

```
[09/13/24] seed@VM:~/.../Task05$ openssl enc -aes-128-cbc -e -in textFile.txt -out cipher_cbc.bin -K 00112233445566778889aabbccddeeff -iv 0102030405060708
hex string is too short, padding with zero bytes to length
[09/13/24] seed@VM:~/.../Task05$ openssl enc -aes-128-ecb -e -in textFile.txt -out cipher_ecb.bin -K 00112233445566778889aabbccddeeff -iv 0102030405060708
warning: iv not used by this cipher
[09/13/24] seed@VM:~/.../Task05$ openssl enc -aes-128-cfb -e -in textFile.txt -out cipher_cfb.bin -K 00112233445566778889aabbccddeeff -iv 0102030405060708
hex string is too short, padding with zero bytes to length
[09/13/24] seed@VM:~/.../Task05$ openssl enc -aes-128-ofb -e -in textFile.txt -out cipher_ofb.bin -K 00112233445566778889aabbccddeeff -iv 0102030405060708
hex string is too short, padding with zero bytes to length
[09/13/24] seed@VM:~/.../Task05$
```

Deleting the 55th bit:

The screenshot shows the Bless debugger interface. The main window displays a memory dump of the file `cipher_cbc.bin`. The dump shows various bytes in hex, ASCII, and decimal formats. Below the dump, there are conversion tools for different data types:

- Signed 8 bit: 101
- Unsigned 8 bit: 101
- Signed 16 bit: 26076
- Unsigned 16 bit: 26076
- Signed 32 bit: 1708969243
- Unsigned 32 bit: 1708969243
- Float 32 bit: 1.30338E+23
- Float 64 bit: 4.78044099569521E+182
- Hexadecimal: 65 DC CD 1B
- Decimal: 101 220 205 027
- Octal: 145 334 315 033
- Binary: 01100101 11011100 11001
- ASCII Text: e??_{1B}

Checkboxes for "Show little endian decoding" and "Show unsigned as hexadecimal" are present. The bottom status bar shows an offset of 0x37 / 0x188e and a selection of None.

Using “-aes-128-cbc”

The screenshot shows a terminal window, a file editor, and a text file viewer.

Terminal:

```
Sep 13 03:37
seed@VM:~/.../Task05$ openssl enc -aes-128-cbc -d -in M_cipher_cbc.bin
[09/13/24]seed@VM:~/.../Task05$ modifiedOutput_cbc.txt -K 00112233445566778889aabccddeff -iv 01020304050
60708
hex string is too short, padding with zero bytes to length
[09/13/24]seed@VM:~/.../Task05$
```

File Editor:

Shows two files: `cipher_cbc.bin` and `cipher_cfb.bin`. The `cipher_cbc.bin` file contains the decrypted content of the modified output.

Text File Viewer:

Shows the contents of the `modifiedOutput_cbc.txt` file, which is a plain text document containing an essay about web development.

Using “-aes-128-ecb”

The terminal window shows the command:

```
openssl enc -aes-128-ecb -d -in M_cipher_ecb.bin -out modifiedOutput_ecb.txt -K 0011223344556677889aabbcdddeeff -iv 01020304050
```

Output:

```
60708 hex string is too short, padding with zero bytes to length
[09/13/24]seed@VM:~/.../Task05$
```

Warning:

```
warning: iv not used by this cipher
[09/13/24]seed@VM:~/.../Task05$
```

Below the terminal, two text files are shown:

- modifiedOutput_ecb.txt**:
There was a problem opening the file "/home/seed/Downloads/Lab_05/modifiedOutput_ecb.txt". 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.
- textFile.txt**:
1 Web development has undergone a remarkable transformation since the inception of the internet. From simple static pages to complex, dynamic web applications, the field has evolved significantly. This essay explores the history, current trends, and future directions of web development, highlighting key technologies and practices that have shaped the industry.
2
3 Historical Context
4 The journey of web development began in the early 1990s with the creation of the World Wide Web by Tim Berners-Lee. The initial web pages were simple HTML documents with basic formatting and hyperlinks. As the internet gained popularity, the need for more interactive and visually appealing websites grew.
5
6 HTML and CSS: The introduction of HTML (HyperText Markup Language) provided a standardized way to

Using “-aes-128-cfb”

The terminal window shows the command:

```
openssl enc -aes-128-cfb -d -in M_cipher_cfb.bin -out modifiedOutput_cfb.txt -K 0011223344556677889aabbcdddeeff -iv 01020304050
```

Output:

```
60708 warning: iv not used by this cipher
[09/13/24]seed@VM:~/.../Task05$
```

Warning:

```
hex string is too short, padding with zero bytes to length
[09/13/24]seed@VM:~/.../Task05$
```

Below the terminal, two text files are shown:

- modifiedOutput_cfb.txt**:
1 Web development has undergone a remarkable transformation since the inception of the internet. From simple static pages to complex, dynamic web applications, the field has evolved significantly. This essay explores the history, current trends, and future directions of web development, highlighting key technologies and practices that have shaped the industry.
2
3 Historical Context
4 The journey of web development began in the early 1990s with the creation of the World Wide Web by Tim Berners-Lee. The initial web pages were simple HTML documents with basic formatting and hyperlinks. As the internet gained popularity, the need for more interactive and visually appealing websites grew.
5
6 HTML and CSS: The introduction of HTML (HyperText Markup Language) provided a standardized way to
- textFile.txt**:
1 Web development has undergone a remarkable transformation since the inception of the internet. From simple static pages to complex, dynamic web applications, the field has evolved significantly. This essay explores the history, current trends, and future directions of web development, highlighting key technologies and practices that have shaped the industry.
2
3 Historical Context
4 The journey of web development began in the early 1990s with the creation of the World Wide Web by Tim Berners-Lee. The initial web pages were simple HTML documents with basic formatting and hyperlinks. As the internet gained popularity, the need for more interactive and visually appealing websites grew.
5
6 HTML and CSS: The introduction of HTML (HyperText Markup Language) provided a standardized way to

Using “-aes-128-ofb”

The screenshot shows a Linux desktop environment with several windows open:

- Terminal:** Shows the command `seed@VM: ~.../Task05\$ openssl enc -aes-128-ofb -d -in M_cipher_ofb.bin -out modifiedOutput_ofb.txt` and its output: "hex string is too short, padding with zero bytes to length".
- File Browser:** Shows files like `ent`, `rred`, `ne`, `ktop`, `uments`, `vndload`, `sic`, `tues`, `eos`, `sh`, and `ier Loc`.
- Text Editor (modifiedOutput_ofb.txt):** Displays the original content of the file, which is a long block of text about web development.
- Text Editor (textFile.txt):** Displays the decrypted content of the file, showing the same text as the original file.

Expected Outcomes After Experimentation:

1. Electronic Codebook Mode:

My assumption was true. ECB works on independent blocks, so corruption in one block doesn't affect others.

2. Cipher Block Chaining Mode:

My assumption was true. In CBC, each block depends on the previous one, so corruption in one block will propagate to the next during decryption.

3. Cipher Feedback Mode:

My assumption was true. Since CFB is a stream mode, the corruption propagates forward, but only one byte per block is directly affected.

4. Output Feedback Mode:

My assumption was false. The reason is that OFB operates like a stream cipher, so errors don't propagate beyond the corrupted byte.

Task06:

Part01:

Using same IV:

The screenshot shows a Linux desktop environment with several windows open. At the top, there are two terminal windows. The left terminal window shows the command:

```
[09/14/24] seed@VM:~/.../Task06$ openssl enc -aes-128-cbc -e -in tex  
tFile.txt -out cipherSameIV_cbc.txt -K 00112233445566778889aabbccdd  
eeff -iv 0102030405060708
```

The right terminal window shows a similar command with a different IV:

```
[09/14/24] seed@VM:~/.../Task06$ openssl enc -aes-128-cbc -e -in tex  
tFile.txt -out cipherSameIV2_cbc.txt -K 00112233445566778889aabbccdd  
deeff -iv 0102030405060708
```

Both terminals output messages indicating that the hex string is too short, so padding with zero bytes is used to reach the required length.

Below the terminals are two text editor windows. Both are titled "cipherSameIV_cbc.txt" and "cipherSameIV2_cbc.txt". Both files are located at "/Downloads/Labsetup/Files/Task06". Both files contain the same content, which is a hex dump of the encrypted file. The hex dump includes characters like FF, C0, and various letters and numbers. The text editors also display a warning message about opening files with invalid characters.

At the bottom of the screen, there is a dock with icons for various applications, and a status bar at the very bottom.

Using Different IV:

The terminal window shows two separate command executions:

```
[09/14/24] seed@VM:~/.../Task06$ openssl enc -aes-128-cbc -e -in testFile.txt -out cipherDiffIV_cbc.txt -K 00112233445566778889aabbccdd  
[09/14/24] seed@VM:~/.../Task06$ openssl enc -aes-128-cbc -e -in testFile.txt -out cipherDiffIV2_cbc.txt -K 00112233445566778889aabbccdd
```

Both commands result in hex strings that are too short, so they are padded with zero bytes to length 16. The resulting ciphertexts are different due to the different IVs.

Below the terminal, a Firefox browser window is open, displaying a corrupted PDF file. The file contains several redacted sections of text, indicating that the file was opened with an invalid encoding.

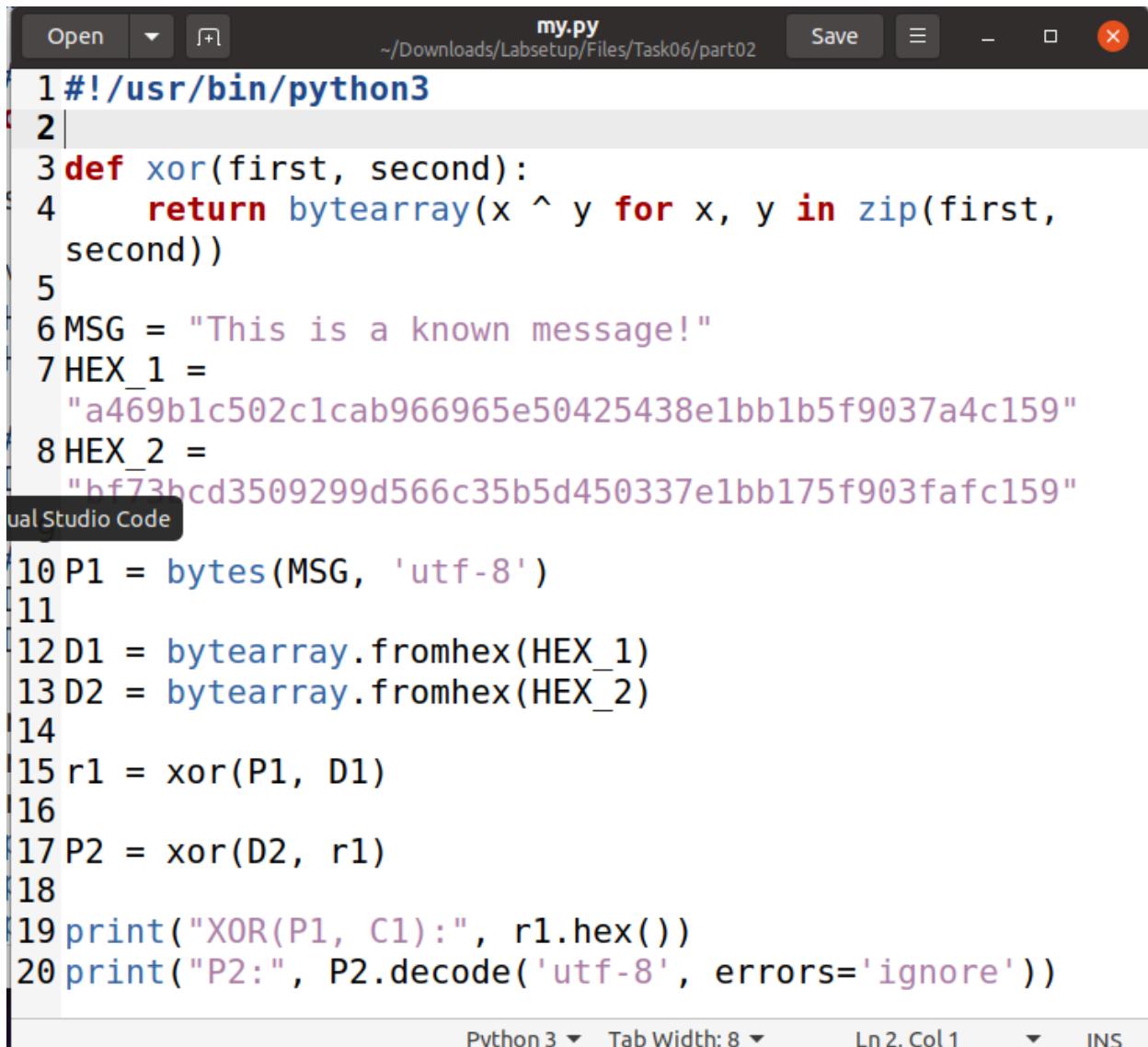
Observation:

- When using different IVs, the ciphertexts will be different even if the plaintext is the same. This is because the IV ensures that each encryption operation is unique.
- When using the same IV, the ciphertexts will be the same if the plaintext is the same. This indicates that the IV does not contribute to the randomness in this case.

Explanation:

- An IV should be unique for each encryption to ensure that the ciphertexts are different even if the plaintexts are the same. Reusing an IV with the same key can result in the same ciphertext for identical plaintexts, which can expose patterns and potentially leak information.

Part02:



The screenshot shows a code editor window titled "my.py" with the file path "~/Downloads/Labsetup/Files/Task06/part02". The code is written in Python 3 and performs XOR operations on two hex strings.

```
1 #!/usr/bin/python3
2
3 def xor(first, second):
4     return bytearray(x ^ y for x, y in zip(first,
5     second))
6
7 MSG = "This is a known message!"
8 HEX_1 =
9     "a469b1c502c1cab966965e50425438e1bb1b5f9037a4c159"
10 HEX_2 =
11     "bf73bcd3509299d566c35b5d450337e1bb175f903fafc159"
12 P1 = bytes(MSG, 'utf-8')
13 D1 = bytearray.fromhex(HEX_1)
14 D2 = bytearray.fromhex(HEX_2)
15 r1 = xor(P1, D1)
16
17 P2 = xor(D2, r1)
18
19 print("XOR(P1, C1):", r1.hex())
20 print("P2:", P2.decode('utf-8', errors='ignore'))
```

The status bar at the bottom indicates "Python 3" and "Tab Width: 8".

```
seed@VM: ~/.../part02
00000000000000000000000000000000
[09/14/24] seed@VM:~/.../part02$ ./my.py
bash: ./my.py: Permission denied
[09/14/24] seed@VM:~/.../part02$ chmod 777 my.py
[09/14/24] seed@VM:~/.../part02$ ./my.py
1f001d8b622a8b99907b6353e2d2356c1d67e2ce356c3a478
11b1a0d165253536c0055050d07570f00000c0000080b0000
100000000000000000000000000000000000000000000000000000000
1[09/14/24] seed@VM:~/.../part02$ ./my.py
1bytearray(b'\xf0\x01\xd8\xb6"\xa8\xb9\x99\x07\xb5>-\#V\xc1\xd6~, \xe
13V\xc3\x4x')
11b1a0d165253536c0055050d07570f00000c0000080b0000
100000000000000000000000000000000000000000000000000000000
1[09/14/24] seed@VM:~/.../part02$ ./my.py
1f001d8b622a8b99907b6353e2d2356c1d67e2ce356c3a478
2[09/14/24] seed@VM:~/.../part02$ ./my.py
2f001d8b622a8b99907b6353e2d2356c1d67e2ce356c3a478
24f726465723a204c61756e63682061206d697373696c6521
2[09/14/24] seed@VM:~/.../part02$ ./my.py
XOR(P1, C1): f001d8b622a8b99907b6353e2d2356c1d67e2ce356c3a478
P2: Order: Launch a missile!
[09/14/24] seed@VM:~/.../part02$ echo "K21-4503, K21-3279"
K21-4503, K21-3279
[09/14/24] seed@VM:~/.../part02$
```

Since the IV is the same for both encryptions, and we have P1 and C1, so we can compute the XOR between P1 and C1 to derive the XOR of P2 and C2.

- Compute:

$$P2 = C2 \text{ XOR } (C1 \text{ XOR } P1)$$

With the same IV, we can derive the content of P2 by XOR C2 with the XOR of C1 and P1. Therefore, in OFB mode, the plaintext P2 can be completely revealed if C1 and P1 are known.

Part03:

Formula:

$eP1 = aIV \text{ XOR } eIV \text{ XOR } PG1$

$eC1 = E(eP1 \text{ XOR } eIV)$

$eC1 = E(aIV \text{ XOR } eIV \text{ XOR } PG1 \text{ XOR } eIV)$

$eC1 = E(aIV \text{ XOR } PG1)$

Code:

A screenshot of a terminal window titled "mm.py" showing Python code. The code defines functions for XORing bytes, padding plaintext, and performing XOR operations between IVs and strings. It also prints the results of these operations.

```
1#!/bin/env python3
2
3 from binascii import hexlify
4 from Crypto.Util.Padding import pad
5
6 def xor_bytes(a, b):
7     return bytes(x ^ y for x, y in zip(a, b))
8
9 my_iv = bytes.fromhex('4724ce37473697dd938285df70afeb64')
10 bob_iv = bytes.fromhex('8e8f11c5463697dd938285df70afeb64')
11
12 def padd(plaintext, block_size):
13     if isinstance(plaintext, str):
14         plaintext = plaintext.encode()
15
16     padded = pad(plaintext, block_size)
17     return padded
18
19 def xor(input_str, my_iv, bob_iv, block_size=16):
20     xor_iv = xor_bytes(my_iv, bob_iv)
21     padded_input = padd(input_str, block_size)
22     final_xor = xor_bytes(xor_iv, padded_input)
23
24     return final_xor
25
26 xor_yes = xor("Yes", my_iv, bob_iv)
27 xor_no = xor("No", my_iv, bob_iv)
28
29 print("K21-4503, K21-3279")
30 print("XOR between IVs and Yes (16 bytes):", xor_yes.hex())
31 print("XOR between IVs and No (16 bytes):", xor_no.hex())
```

Output:

```
Bob's ciphertex: 10ce0ad963ec386dd9a01c30e2b1cc98  
The IV used     : c722a541eb93caa9f6723a0cb55f76a8  
  
Next IV          : 34fda172eb93caa9f6723a0cb55f76a8  
Your plaintext  : 8d6ccecc10d0d0d0d0d0d0d0d0d0d0d0d0d0d0d  
Your ciphertext: 565229833a776b648cd7f4e95e0beb47c  
4  
  
Next IV          : 02eef7e0eb93caa9f6723a0cb55f76a8  
Your plaintext  : ^C  
[09/14/24]seed@VM:~/.../Files$ nc 10.9.0.80 3000  
Bob's secret message is either "Yes" or "No", with  
Bob's ciphertex: 7dba782c47199d73986e1ac30fdf4b74  
The IV used     : 3ed1e96bac341d635a269ff73cbd872a  
  
Next IV          : 42bcb9a4ac341d635a269ff73cbd872a  
Your plaintext  : 250823c20d0d0d0d0d0d0d0d0d0d0d0d0d0d  
Your ciphertext: 7dba782c47199d73986e1ac30fdf4b74  
1  
  
Next IV          : 86b684ecac341d635a269ff73cbd872a
```

Task07:

Code:

The screenshot shows a terminal window titled "my.py" with the command "gedit my.py" run. The code in "my.py" is a Python script for cracking an AES-CBC ciphertext. It includes functions for generating keys from words, decrypting ciphertext, and finding a key from a wordlist. The terminal output shows the script being run and it finds the key "Syracuse".

```
#!/usr/bin/python3
from Crypto.Cipher import AES
from Crypto.Util.Padding import unpad
import binascii
def loadData(ciphertext_hex, iv_hex, plaintext):
    ciphertext = binascii.unhexlify(ciphertext_hex)
    iv = binascii.unhexlify(iv_hex)
    return ciphertext, iv, plaintext.encode('utf-8')
def generateKey(word):
    key = word.encode('utf-8')
    key += b'#' * (16 - len(key)) if len(key) < 16 else b''
    return key[:16]
def decrypt(ciphertext, iv, key):
    cipher = AES.new(key, AES.MODE_CBC, iv)
    try:
        decrypted = unpad(cipher.decrypt(ciphertext), AES.block_size)
    except ValueError:
        return None
def findKey(wordlist_file, ciphertext, iv, plaintext):
    with open(wordlist_file, 'r') as file:
        for word in file:
            word = word.strip()
            key = generateKey(word)
            decrypted = decrypt(ciphertext, iv, key)
            if decrypted == plaintext:
                return key[:16]
key = word.encode('utf-8')
key += b'#' * (16 - len(key)) if len(key) < 16 else b''
return key[:16]
def decrypt(ciphertext, iv, key):
    cipher = AES.new(key, AES.MODE_CBC, iv)
    try:
        decrypted = unpad(cipher.decrypt(ciphertext), AES.block_size)
    except ValueError:
        return None
def findKey(wordlist_file, ciphertext, iv, plaintext):
    with open(wordlist_file, 'r') as file:
        for word in file:
            word = word.strip()
            key = generateKey(word)
            decrypted = decrypt(ciphertext, iv, key)
            if decrypted == plaintext:
                print(f"Found key: {word}")
                return
print("No key found.")
if __name__ == "__main__":
    ciphertext_hex = "764aa26b55a4da654df6b19e4bce00f4ed05e09346fb0e762583cb7da2ac93a2"
    iv_hex = "aabbcdddeeff00998877665544332211"
    plaintext = "This is a top secret."
    wordlist_file = "words.txt"
    ciphertext, iv, plaintext = loadData(ciphertext_hex, iv_hex, plaintext)
    findKey(wordlist_file, ciphertext, iv, plaintext)
```

[09/14/24]seed@VM:~/.../Task07\$ gedit my.py
[09/14/24]seed@VM:~/.../Task07\$./my.py
Found key: Syracuse
[09/14/24]seed@VM:~/.../Task07\$ ^C
[09/14/24]seed@VM:~/.../Task07\$ echo "K21-4503, K21-3279"
[09/14/24]seed@VM:~/.../Task07\$./my.py
No key found.
[09/14/24]seed@VM:~/.../Task07\$ echo "K21-4503, K21-3279"
[09/14/24]seed@VM:~/.../Task07\$./my.py
Found key: Syracuse
[09/14/24]seed@VM:~/.../Task07\$./my.py
Found key: Syracuse
[09/14/24]seed@VM:~/.../Task07\$./my.py
Traceback (most recent call last):
 File "./my.py", line 43, in <module>
 findKey(wordlist_file, ciphertext, iv, plaintext)
 File "./my.py", line 29, in findKey
 key = key(word)
UnboundLocalError: local variable 'key' referenced before assignment
[09/14/24]seed@VM:~/.../Task07\$./my.py
Found key: Syracuse
[09/14/24]seed@VM:~/.../Task07\$

[09/14/24]seed@VM:~/.../Task07\$ gedit my.py
[09/14/24]seed@VM:~/.../Task07\$./my.py
Found key: Syracuse
[09/14/24]seed@VM:~/.../Task07\$ ^C
[09/14/24]seed@VM:~/.../Task07\$ echo "K21-4503, K21-3279"
[09/14/24]seed@VM:~/.../Task07\$./my.py
No key found.
[09/14/24]seed@VM:~/.../Task07\$ echo "K21-4503, K21-3279"
[09/14/24]seed@VM:~/.../Task07\$./my.py
Found key: Syracuse
[09/14/24]seed@VM:~/.../Task07\$./my.py
Found key: Syracuse
[09/14/24]seed@VM:~/.../Task07\$./my.py
Traceback (most recent call last):
 File "./my.py", line 43, in <module>
 findKey(wordlist_file, ciphertext, iv, plaintext)
 File "./my.py", line 29, in findKey
 key = key(word)
UnboundLocalError: local variable 'key' referenced before assignment
[09/14/24]seed@VM:~/.../Task07\$./my.py
Found key: Syracuse
[09/14/24]seed@VM:~/.../Task07\$

Output:

The screenshot shows a terminal window titled "seed@VM: ~.../Task07" running on a Linux system. The terminal displays the following log output:

```
File "./my.py", line 35, in find_key
    if try_decrypt(key):
      File "./my.py", line 21, in try_decrypt
        cipher = AES.new(key, AES.MODE_CBC, iv)
        File "/usr/local/lib/python3.8/dist-packages/Crypto/Cipher/AES.py", line 232,
in new
    return _create_cipher(sys.modules[__name__], key, mode, *args, **kwargs)
    File "/usr/local/lib/python3.8/dist-packages/Crypto/Cipher/_init_.py", line
79, in _create_cipher
    return modes[mode](factory, **kwargs)
    File "/usr/local/lib/python3.8/dist-packages/Crypto/Cipher/_mode_cbc.py", line
274, in _create_cbc_cipher
    cipher_state = factory._create_base_cipher(kwargs)
    File "/usr/local/lib/python3.8/dist-packages/Crypto/Cipher/AES.py", line 93, i
n _create_base_cipher
    raise ValueError("Incorrect AES key length (%d bytes)" % len(key))
ValueError: Incorrect AES key length (18 bytes)
[09/14/24]seed@VM:~/.../Task07$ gedit my.py
[09/14/24]seed@VM:~/.../Task07$ ./my.py
Found key: Syracuse
[09/14/24]seed@VM:~/.../Task07$ ^C
[09/14/24]seed@VM:~/.../Task07$ echo "K21-4503, K21-3279"
K21-4503, K21-3279
[09/14/24]seed@VM:~/.../Task07$
```

The terminal window is part of a desktop environment, with a file browser window titled "words.txt" visible in the background. The file browser lists various words, with "Syracuse" highlighted in red.