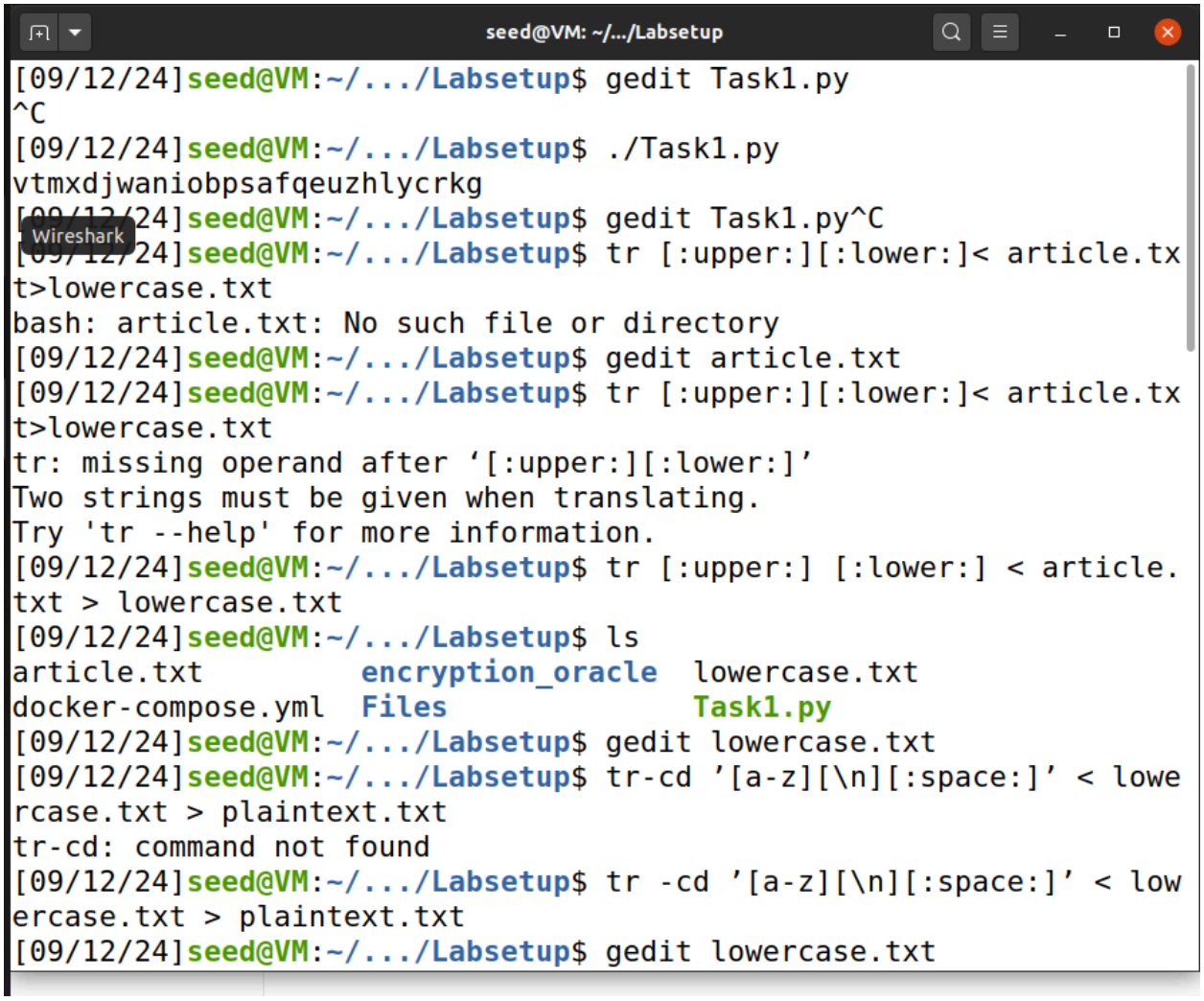
IS Assignment 1

K21-4503 Muhammad Tahir

K21-3279 Insha Javed

Task01:



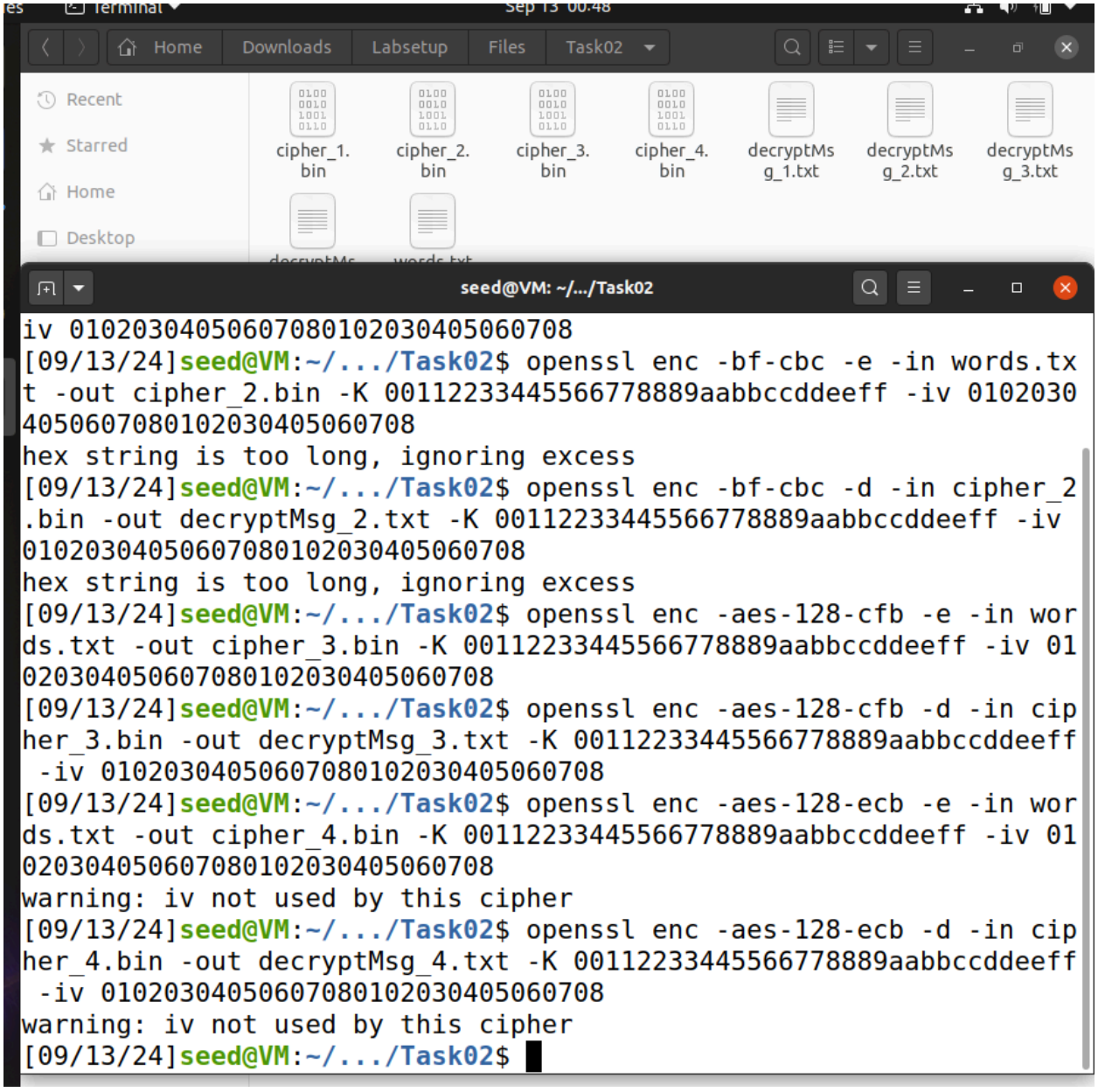

tr-cd: command not found
[09/12/24]seed@VM:~/.../Labsetup$ tr -cd '[a-z][\n][:space:]' < lowercase.txt > plaintext.txt
[09/12/24]seed@VM:~/.../Labsetup$ gedit lowercase.txt
[09/12/24]seed@VM:~/.../Labsetup$ tr 'abcdefghijklmnopqrstuvwxyz''sxtrwinqbedpvgkfmalhyuojzc'\
> < plaintext.txt > ciphertext.txt
tr: missing operand after '\'abcdefghijklmnopqrstuvwxyz\'\'sxtrwinqbedpvgkfmalhyuojzc\''
Two strings must be given when translating.
Try 'tr --help' for more information.
[09/12/24]seed@VM:~/.../Labsetup$ tr 'abcdefghijklmnopqrstuvwxyz' 'sxtrwinqbedpvgkfmalhyuojzc' \ < plaintext.txt > ciphertext.txt
tr: extra operand ' '
Try 'tr --help' for more information.
[09/12/24]seed@VM:~/.../Labsetup$ tr 'abcdefghijklmnopqrstuvwxyz' 'sxtrwinqbedpvgkfmalhyuojzc' \ < plaintext.txt > ciphertext.txt
tr: extra operand ' '
Try 'tr --help' for more information.
[09/12/24]seed@VM:~/.../Labsetup$ tr 'abcdefghijklmnopqrstuvwxyz' 'sxtrwinqbedpvgkfmalhyuojzc' < plaintext.txt > ciphertext.txt
[09/12/24]seed@VM:~/.../Labsetup$
[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 WA
6 THE EMERGENCE OF METOO TIMES UP BLACKGOWN POLITICS ARMCANDY ACTIVIS
7 A NATIONAL CONVERSATION AS BRIEF AND MAD AS A FEVER DREAM ABOUT WHE
8 OUGHT TO BE A PRESIDENT WINFREY THE SEASON DIDNT JUST SEEM EXTRA LO
9 EXTRA LONG BECAUSE THE OSCARS WERE MOVED TO THE FIRST WEEKEND IN MA
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 WHIC
15 A JUBILANT COMINGOUT PARTY FOR TIMES UP THE MOVEMENT SPEARHEADED BY
16 POWERFUL HOLLYWOOD WOMEN WHO HELPED RAISE MILLIONS OF DOLLARS TO FI
17 HARASSMENT AROUND THE COUNTRY
18
19 SIGNALING THEIR SUPPORT GOLDEN GLOBES ATTENDEES SWATHED THEMSELVES
20 SPORTED LAPEL PINS AND SOUNDED OFF ABOUT SEXIST POWER IMBALANCES FR
21 CARPET AND THE STAGE ON THE AIR E WAS CALLED OUT ABOUT PAY INEQUITY
22 ITS FORMER ANCHOR CATT SADLER QUIT ONCE SHE LEARNED THAT SHE WAS MA
23 LESS THAN A MALE COHOST AND DURING THE CEREMONY NATALIE PORTMAN TOO
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 SEAS
31 CAMPAIGN OR ONE THAT BECAME ASSOCIATED ONLY WITH REDCARPET ACTIONS
32 A SPOKESWOMAN SAID THE GROUP IS WORKING BEHIND CLOSED DOORS AND HAS
33 AMASSED MILLION FOR ITS LEGAL DEFENSE FUND WHICH AFTER THE GLOBES
seed@VM: ~/.../Files
ext.txt > mysolve.txt
[09/12/24]seed@VM:~/.../Files$ tr 'ytnvxbhmqzi' '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' 'THEAOFRISULGKDMN' < ciphertext.txt > mysolve.txt
[09/12/24]seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcud' 'THEAOFRISULGKDMNY' < ciphertext.txt > mysolve.txt
[09/12/24]seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcudgj' 'THEAOFRISULGKDMNYBQ' < ciphertext.txt > mysolve.txt
[09/12/24]seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcudgja' 'THEAOFRISULGKDMNYBQC' < ciphertext.txt > mysolve.txt
[09/12/24]seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcudgjae' 'THEAOFRISULGKDMNYBQCP' < ciphertext.txt > mysolve.txt
[09/12/24]seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcudgjael' 'THEAOFRISULGKDMNYBQCPW' < ciphertext.txt > mysolve.txt
[09/12/24]seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcudgjaelk' 'THEAOFRISULGKDMNYBQCPWX' < ciphertext.txt > mysolve.txt
[09/12/24]seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcudgjaelkf' 'THEAOFRISULGKDMNYBQCPWXV' < ciphertext.txt > mysolve.txt
[09/12/24]seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcudgjaelkfu' 'THEAOFRISULGKDMNYBQCPWXVJ' < ciphertext.txt > mysolve.txt
[09/12/24]seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcudgjaelkf' 'THEAOFRISULGKDMNYBQCPWXV' < ciphertext.txt > mysolve.txt
[09/13/24]seed@VM:~/.../Files$ tr 'ytnvxbhmqzirspcudgjaelkfo' 'THEAOFRISULGKDMNYBQCPWXVJ' < ciphertext.txt > mysolve.txt
[09/13/24]seed@VM:~/.../Files$ echo "K21-4503, K21-3279"
K21-4503, K21-3279
[09/13/24]seed@VM:~/.../Files$

This is the final result:

tr 'ytnvxbhmqzirspcudgjaelkfo' ‘THEAOFRISULGKDMNYBQCPWXVJ'
< ciphertext.txt > mysolve.txt

Task02:

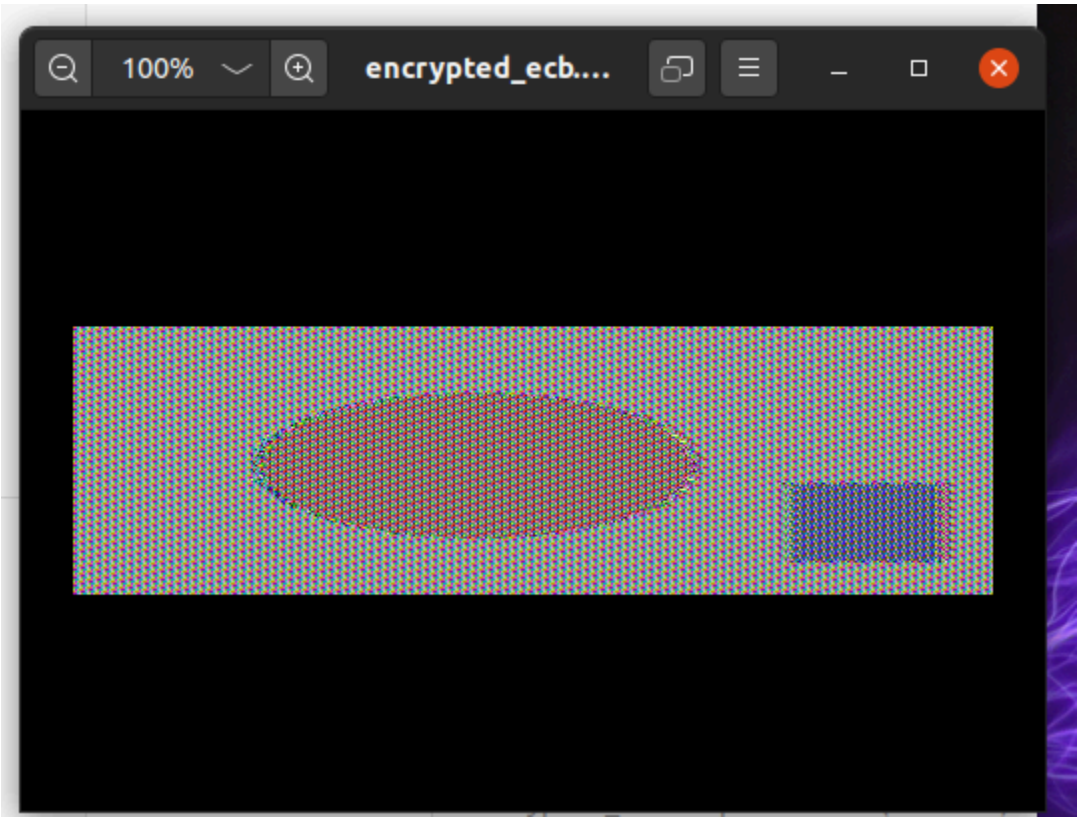


Task03:

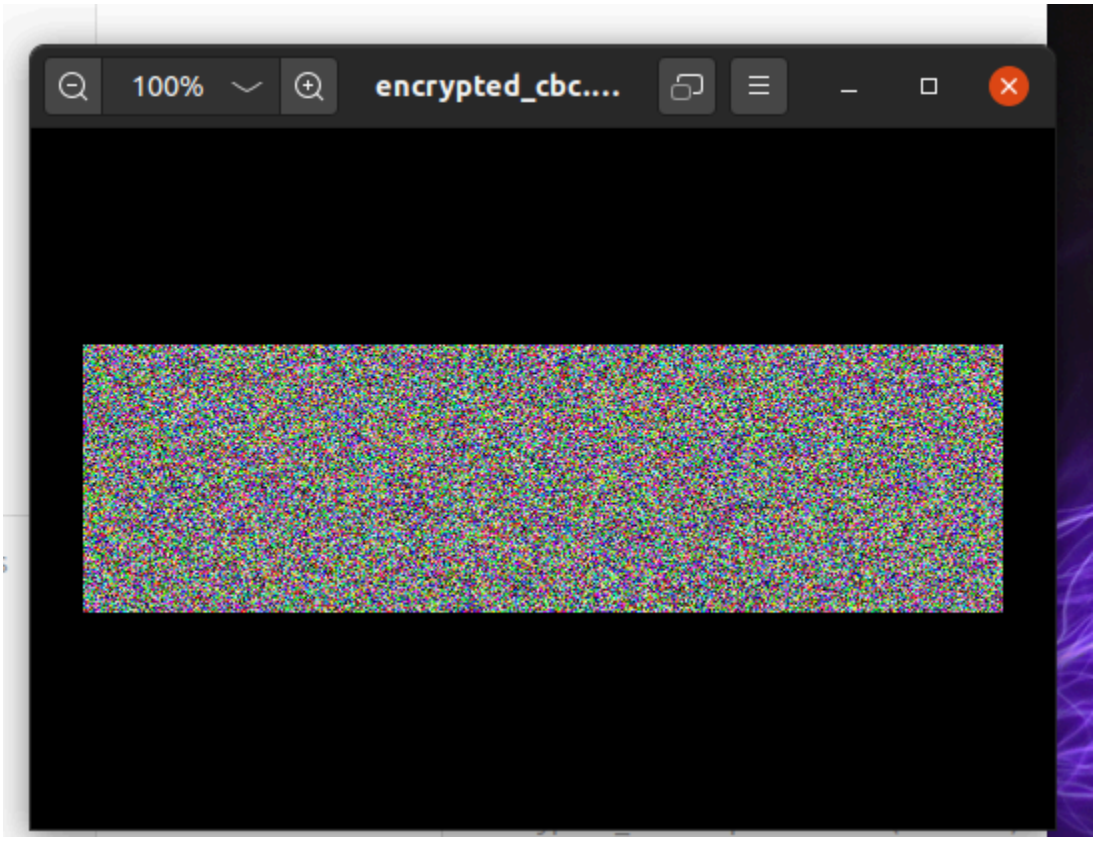
```
seed@VM: ~/.../Task03
[09/14/24]seed@VM:~/.../Task03$ head -c 54 pic_original.bmp > heade
r
[09/14/24]seed@VM:~/.../Task03$ tail -c +55 pic_original.bmp > body
[09/14/24]seed@VM:~/.../Task03$ openssl enc -aes-128-ecb -in body -
out body_ecb.enc -K 888e4d66fc63d6c7d80e92e80380bb08
[09/14/24]seed@VM:~/.../Task03$ 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 -
Terminal
dy_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.

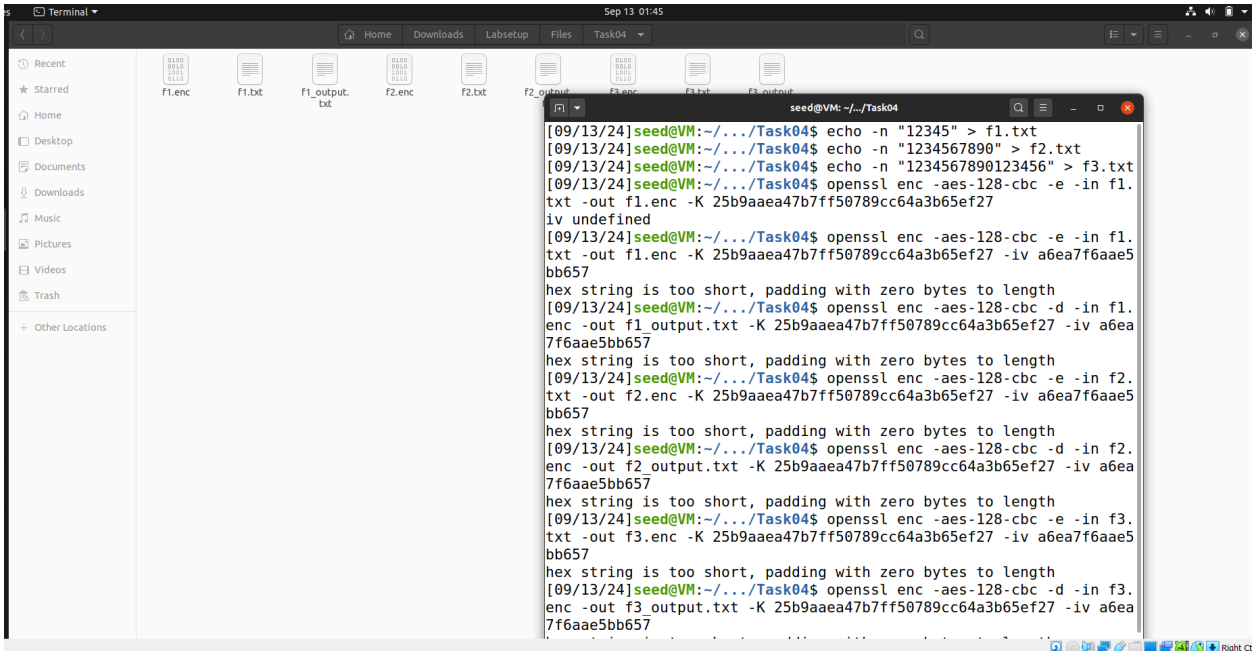
Task04:

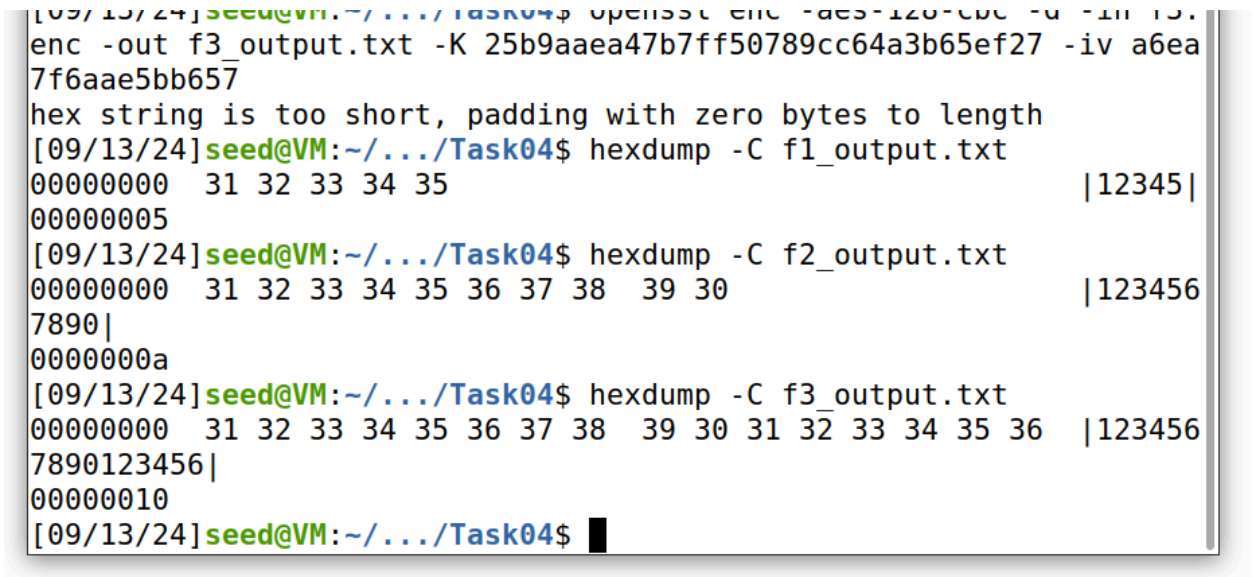
Part 1:

- **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.

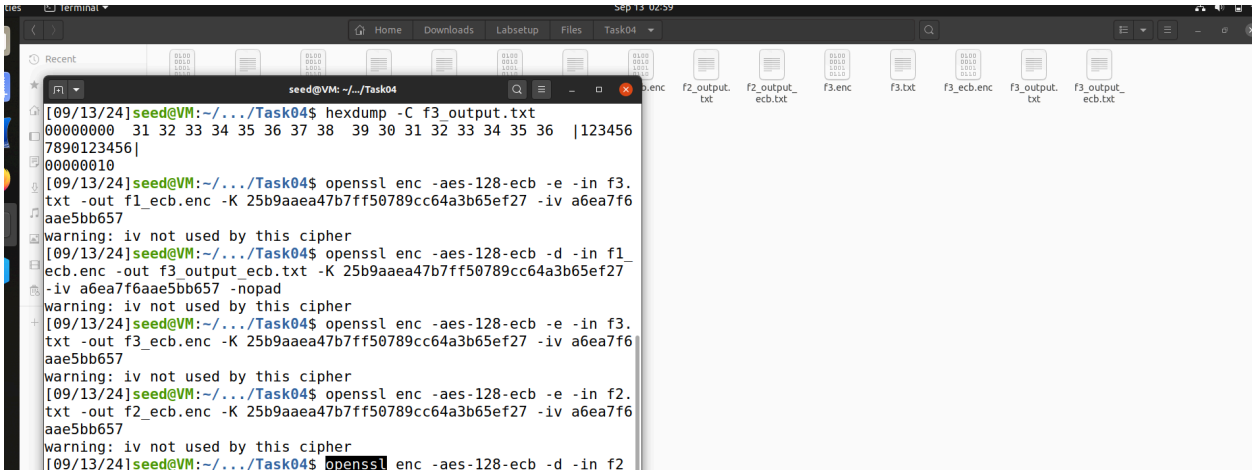
Part 2:

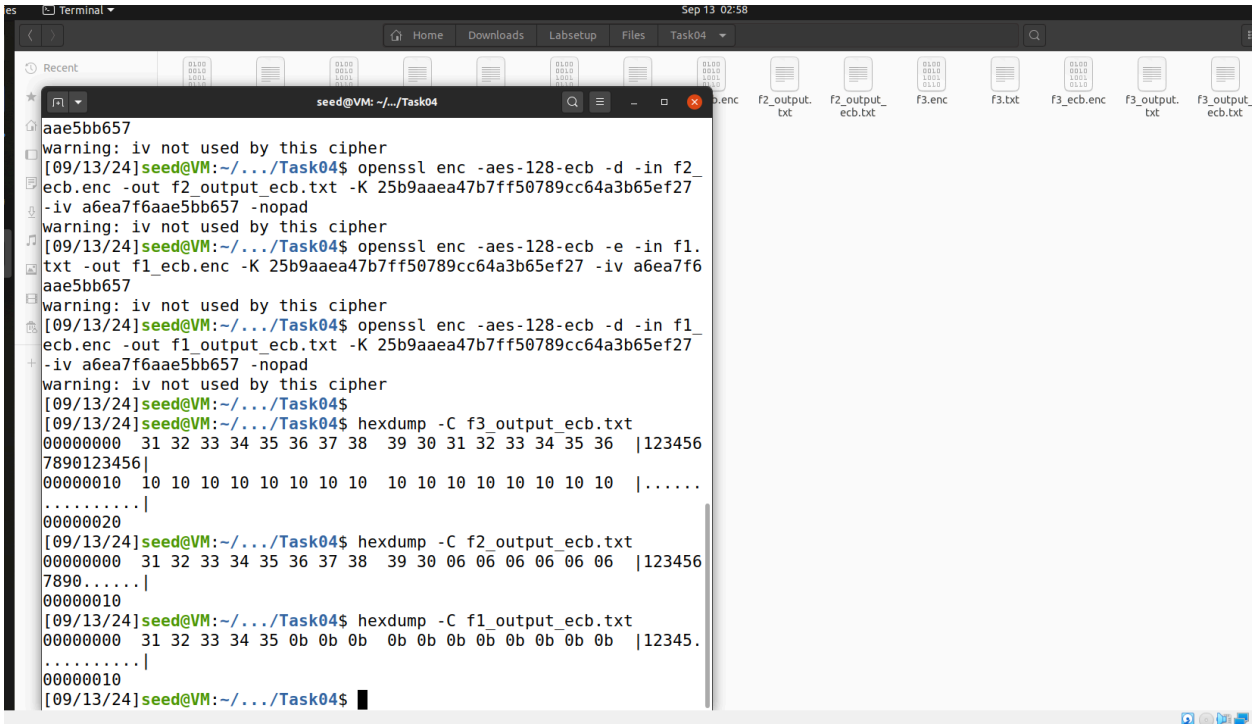
Using **“-aes-128-cbc”**





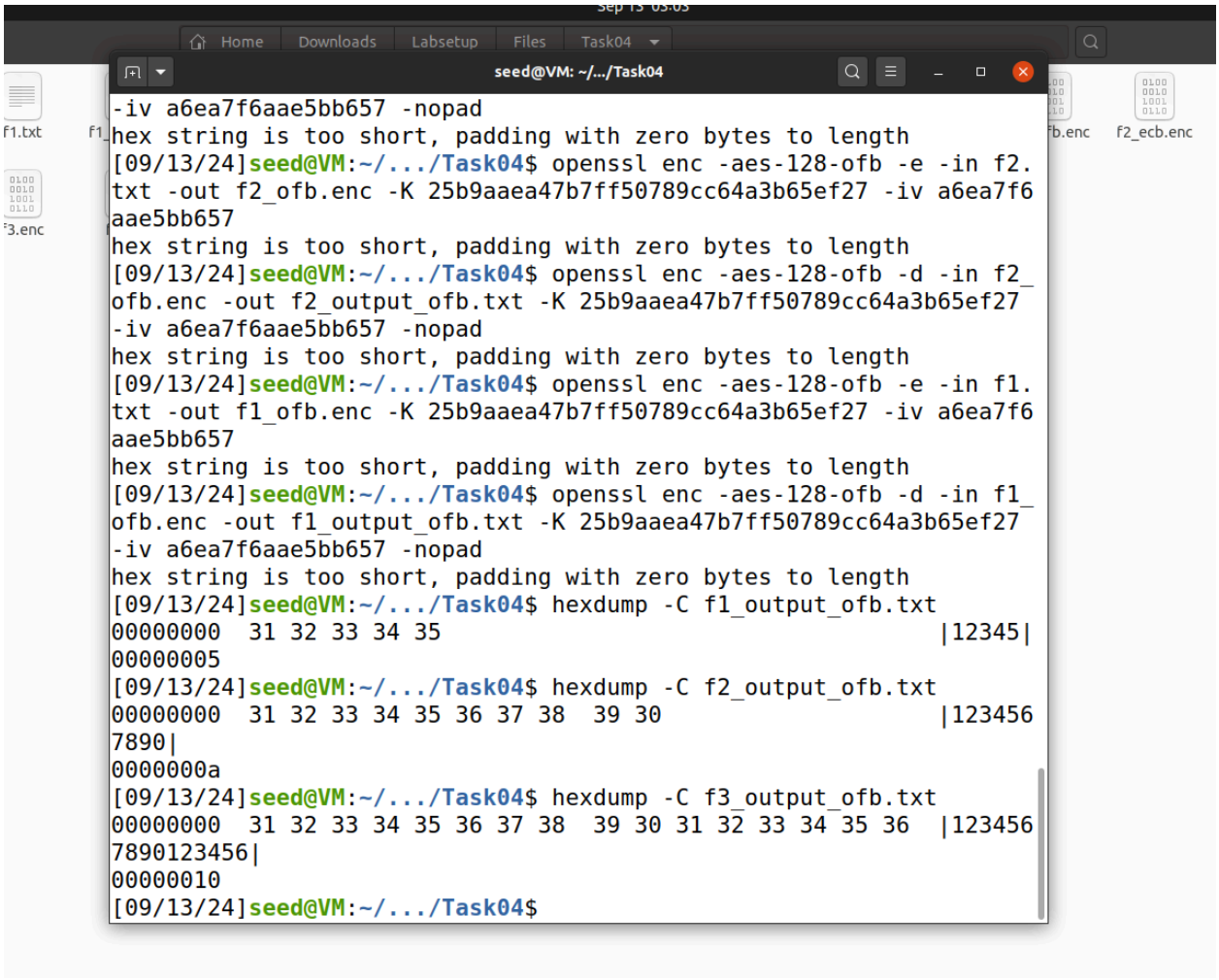
Using **“-aes-128-ecb”:**





Using **“-aes-128-cfb”**

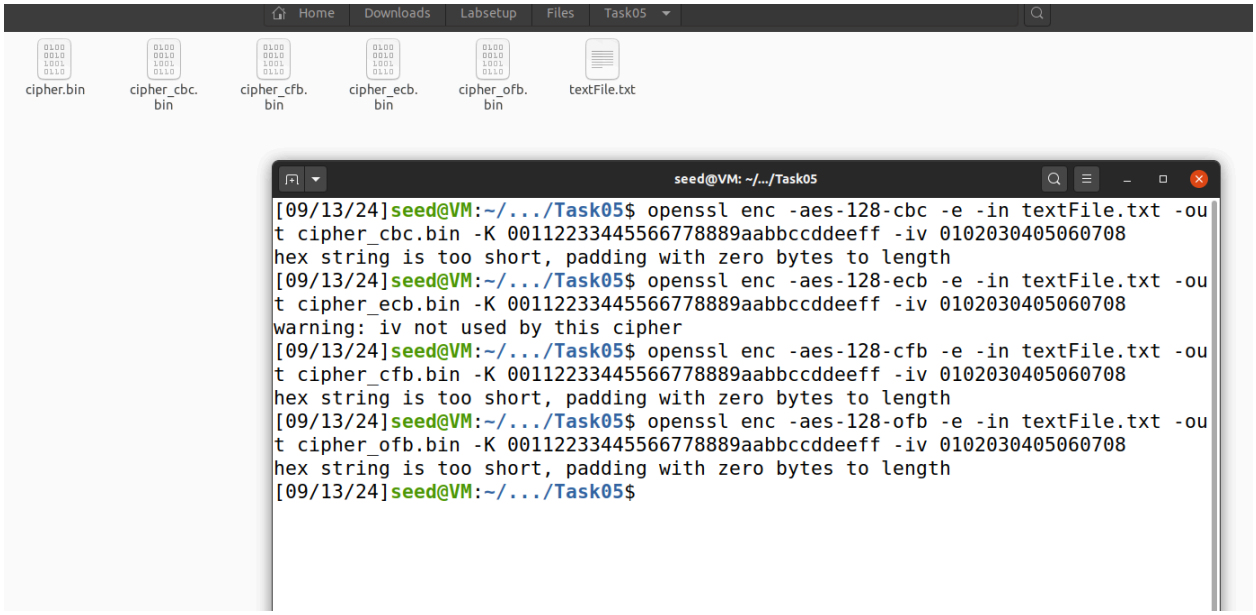
Using **“-aes-128-ofb”**



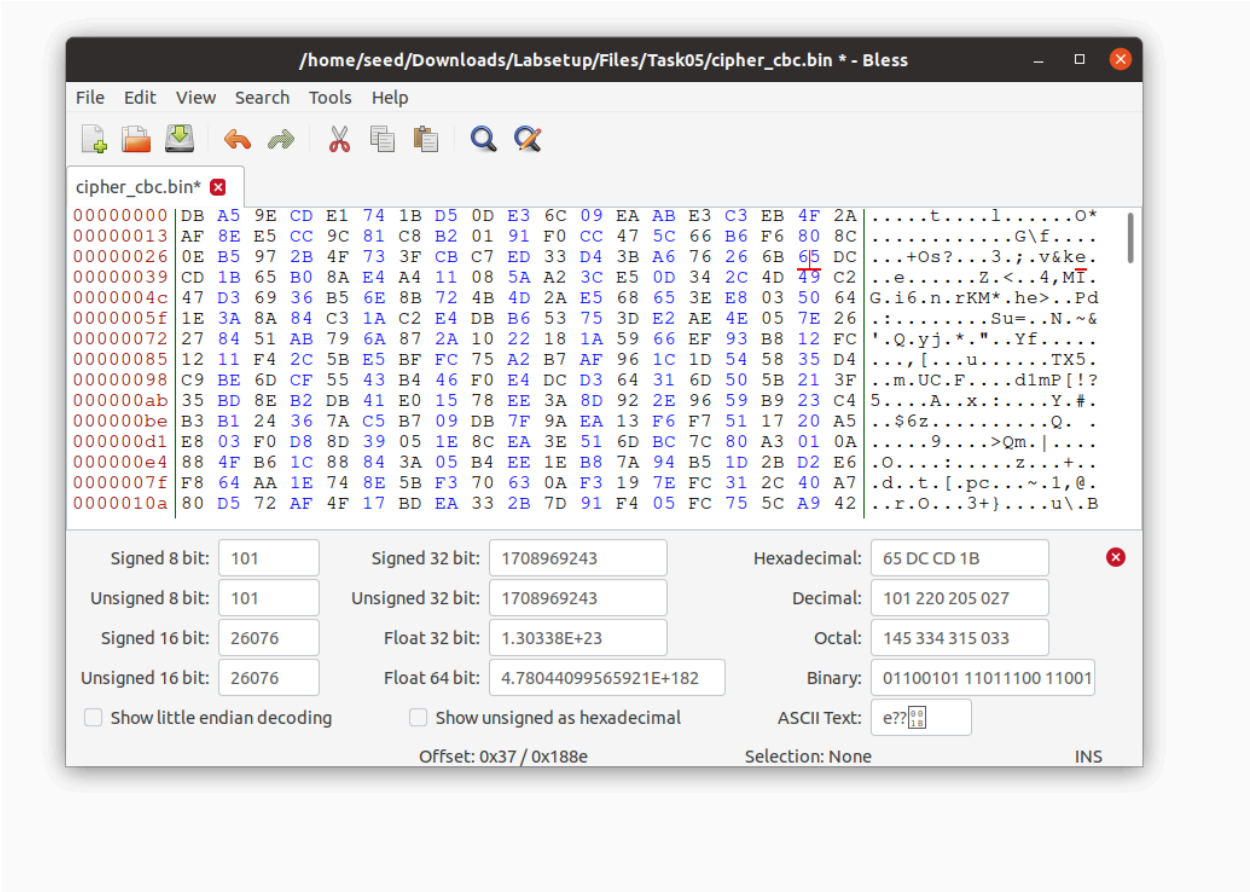
Task05:

Expected Outcomes Before Experimentation:

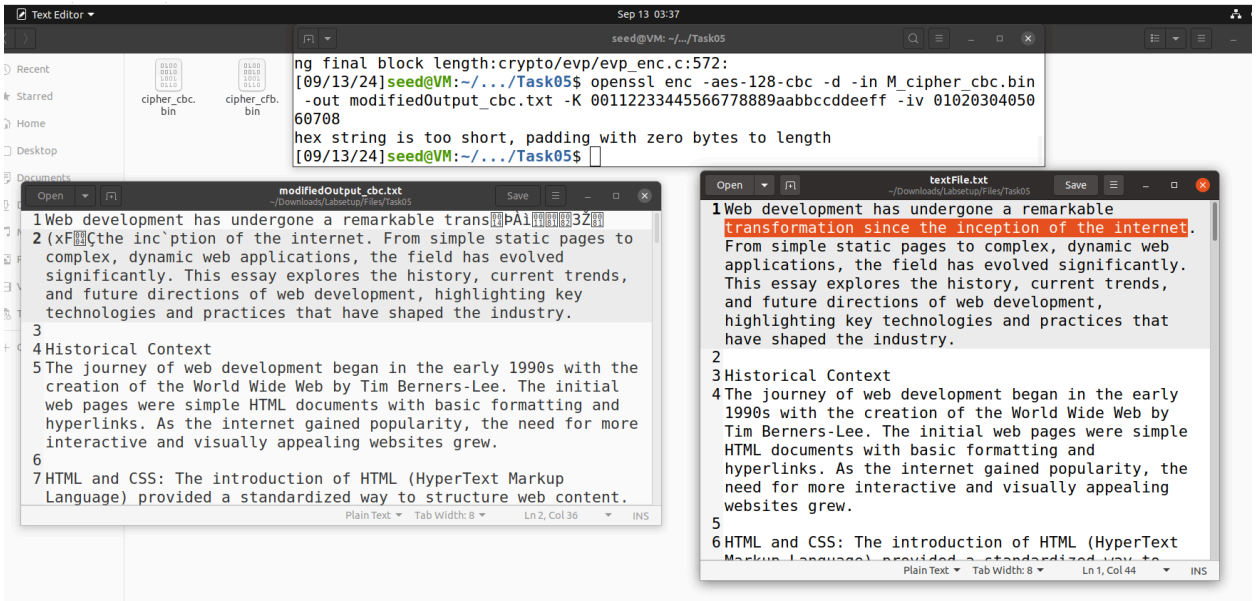
1) **Electronic Codebook Mode**:
    a) Only the corrupted block (the one containing the 55th byte) will be affected.
2) **Cipher Block Chaining Mode**:
    a) The corrupted block and the next block will be affected.
3) **Cipher Feedback Mode**:
    a) The corrupted byte will affect the corresponding byte and some subsequent bytes.
4) **Output Feedback Mode**:
    a) The corrupted byte will affect the maximum bytes.



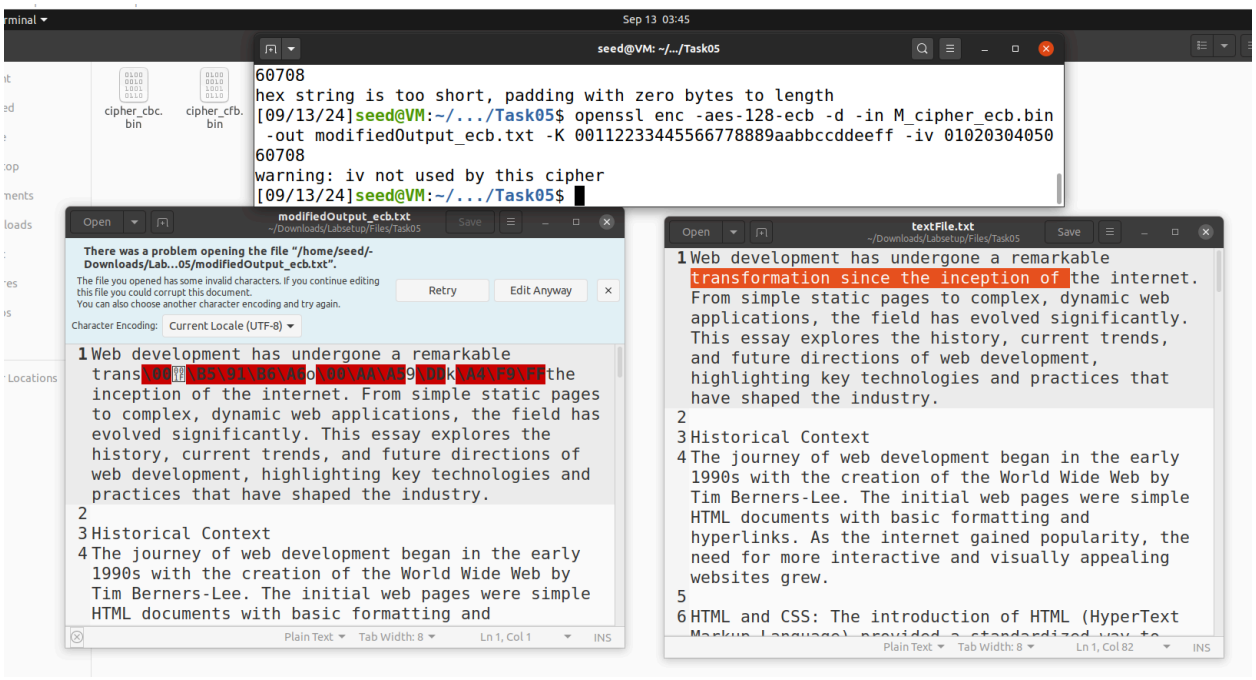
Deleting the 55th bit:



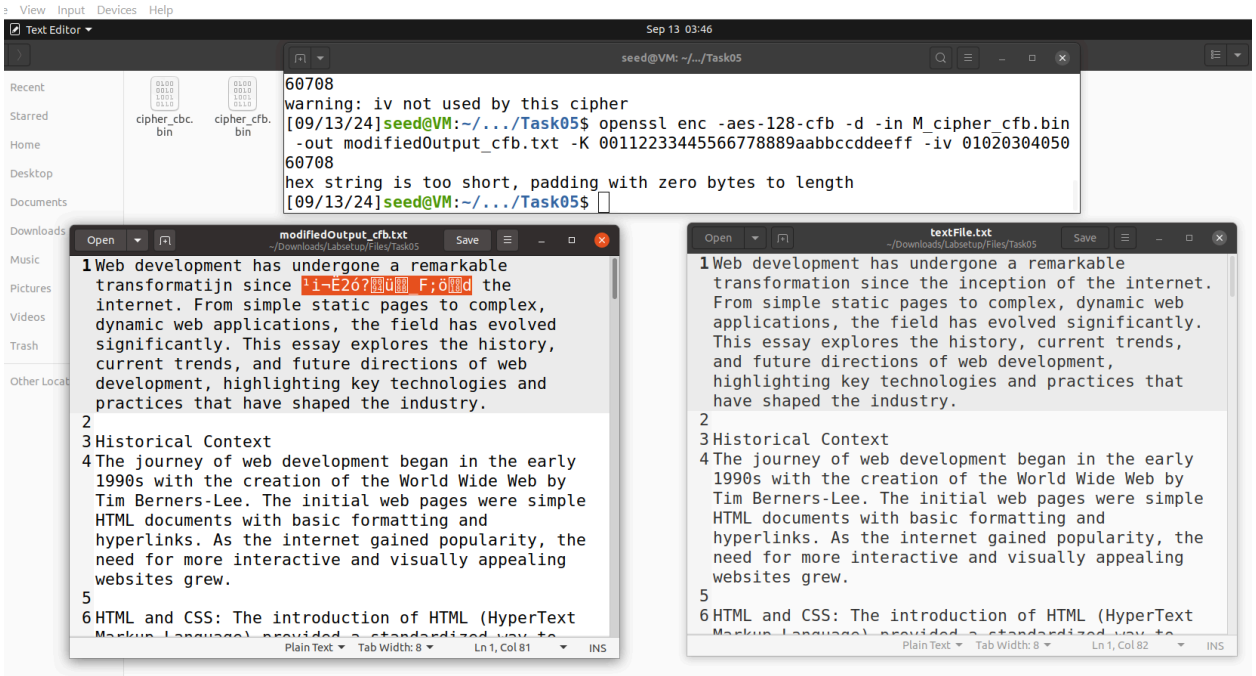
Using “-aes-128-cbc”



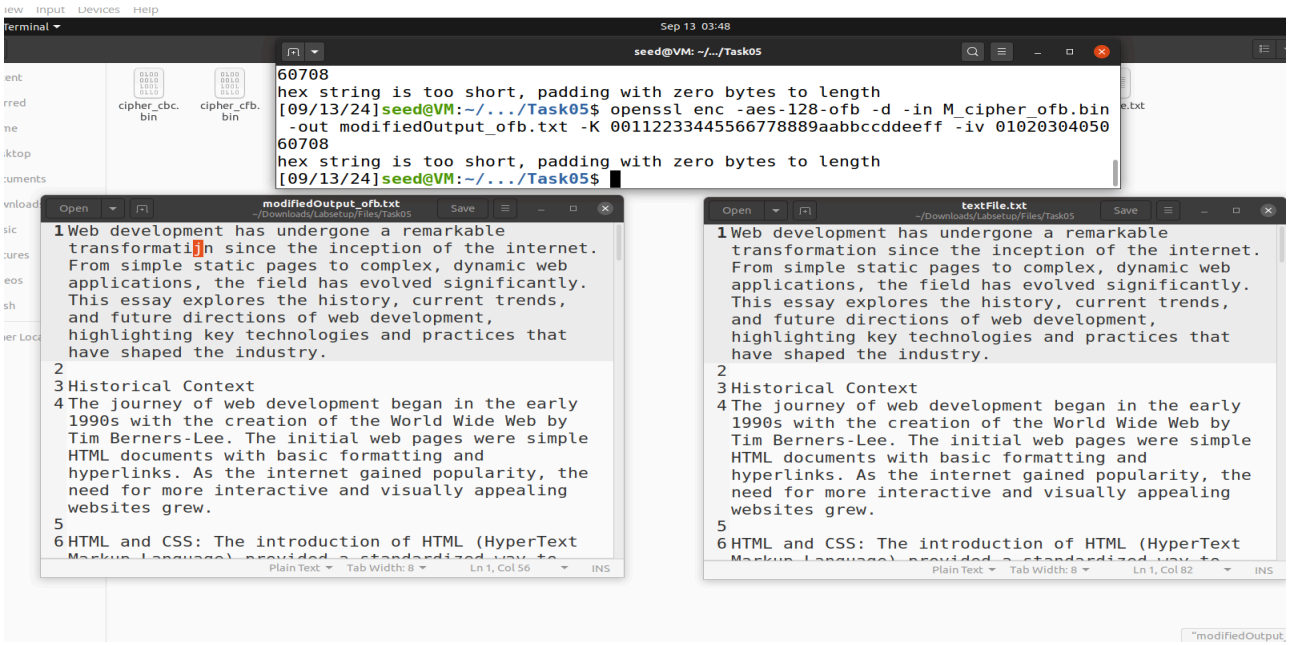
Using “-aes-128-ecb”



Using “-aes-128-cfb”



Using **“-aes-128-ofb”**



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.

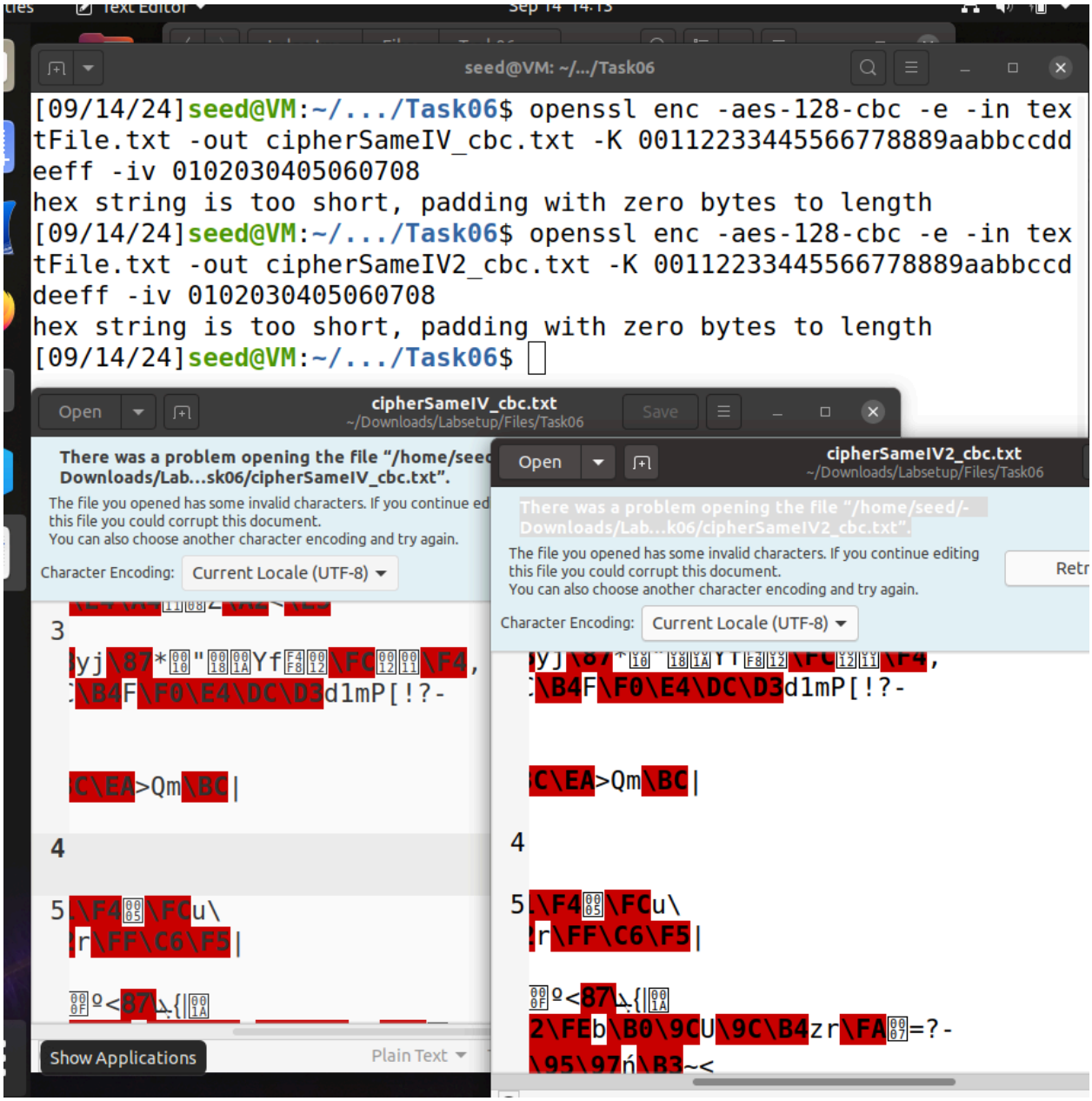
4. **Output Feedback Mode**:

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

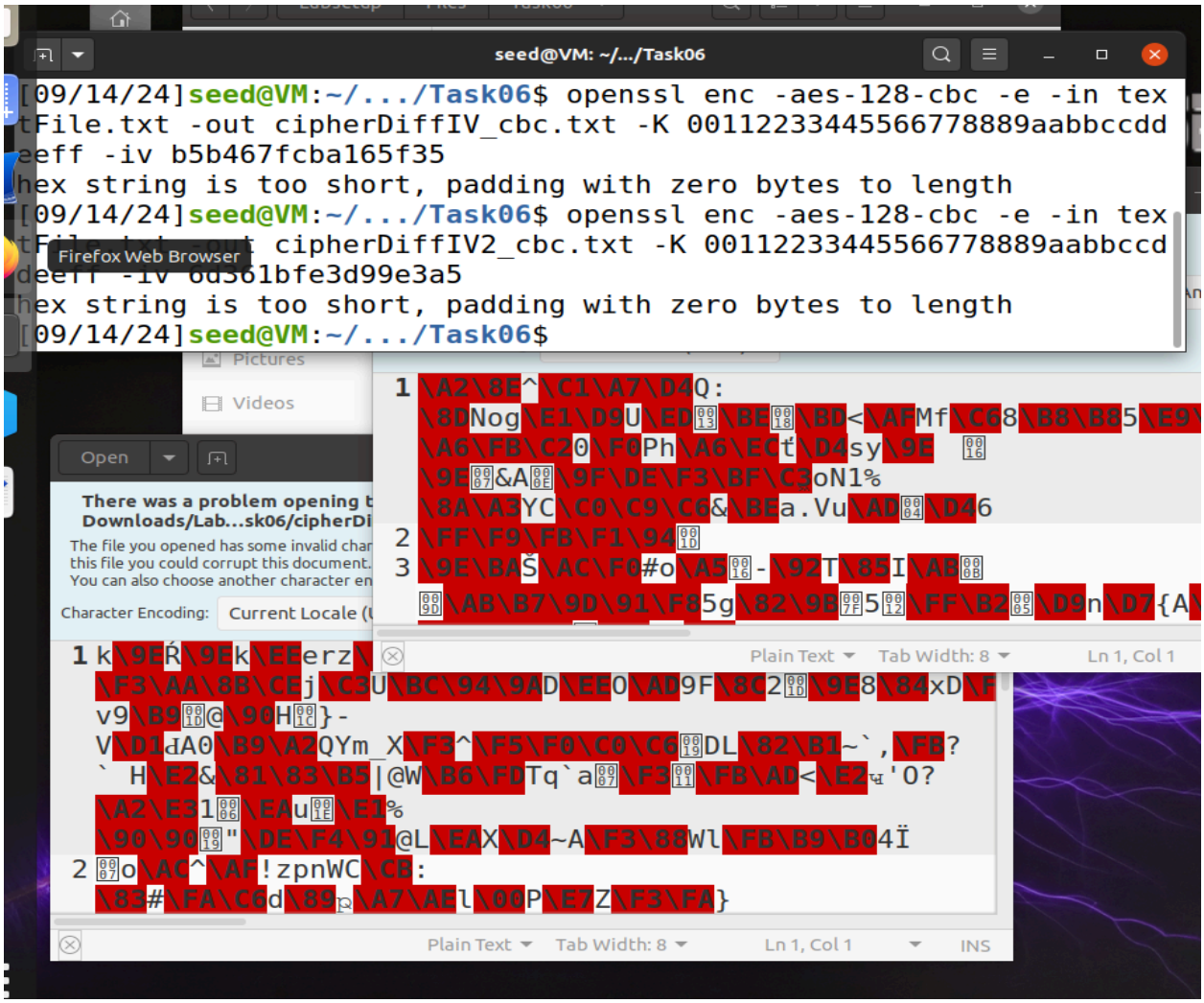
Task06:

Part01:

Using same IV:



Using Different IV:



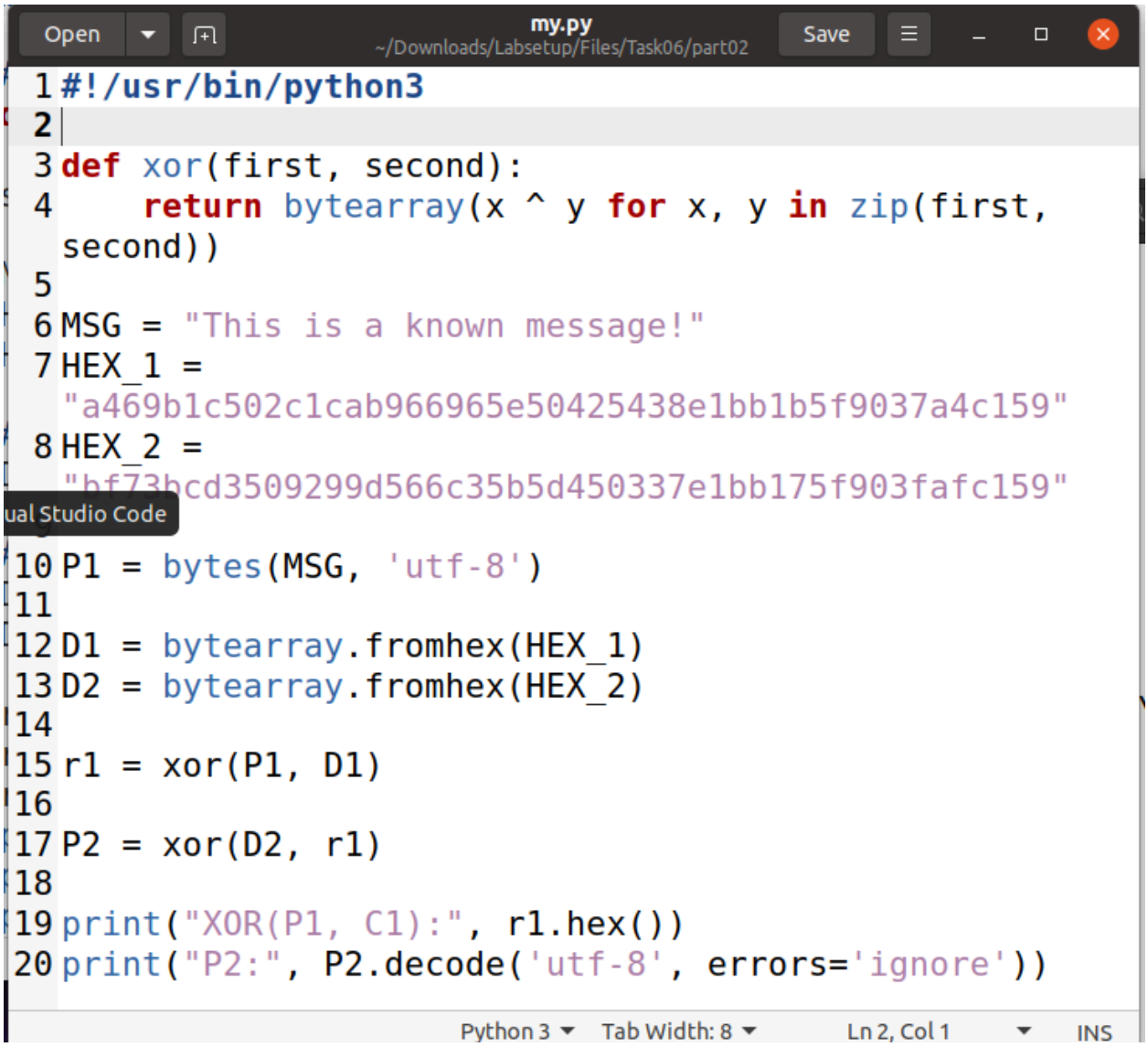
Observation:

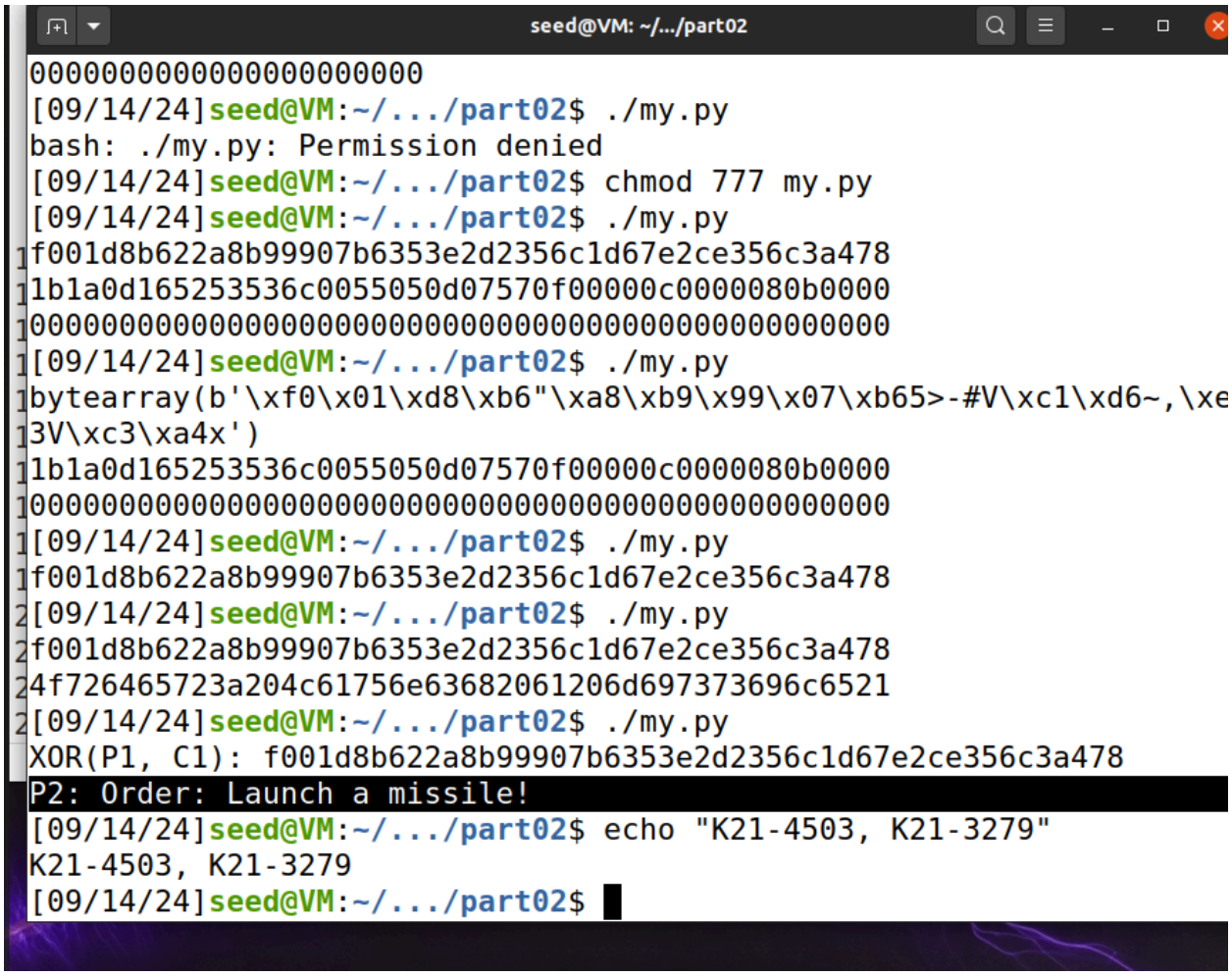
- When using different IVs, the ciphertexts will be different even if the plaintext is the same. This is because the IV ensures that each encryption operation is unique.
- When using the same IV, the ciphertexts will be the same if the plaintext is the same. This indicates that the IV does not contribute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


Open
my.py
~/Downloads/Labsetup/Files/Task06/part02
Save
#!/usr/bin/python3
def xor(first, second):
return bytearray(x ^ y for x, y in zip(first, second))
MSG = "This is a known message!"
HEX_1 = "a469b1c502c1cab966965e50425438e1bb1b5f9037a4c159"
HEX_2 = "bf73bcd3509299d566c35b5d450337e1bb175f903fafc159"
P1 = bytes(MSG, 'utf-8')
D1 = bytearray.fromhex(HEX_1)
D2 = bytearray.fromhex(HEX_2)
r1 = xor(P1, D1)
P2 = xor(D2, r1)
print("XOR(P1, C1):", r1.hex())
print("P2:", P2.decode('utf-8', errors='ignore'))
Python 3
Tab Width: 8
Ln 2, Col 1
INS



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 XOR (C1 XOR P1)

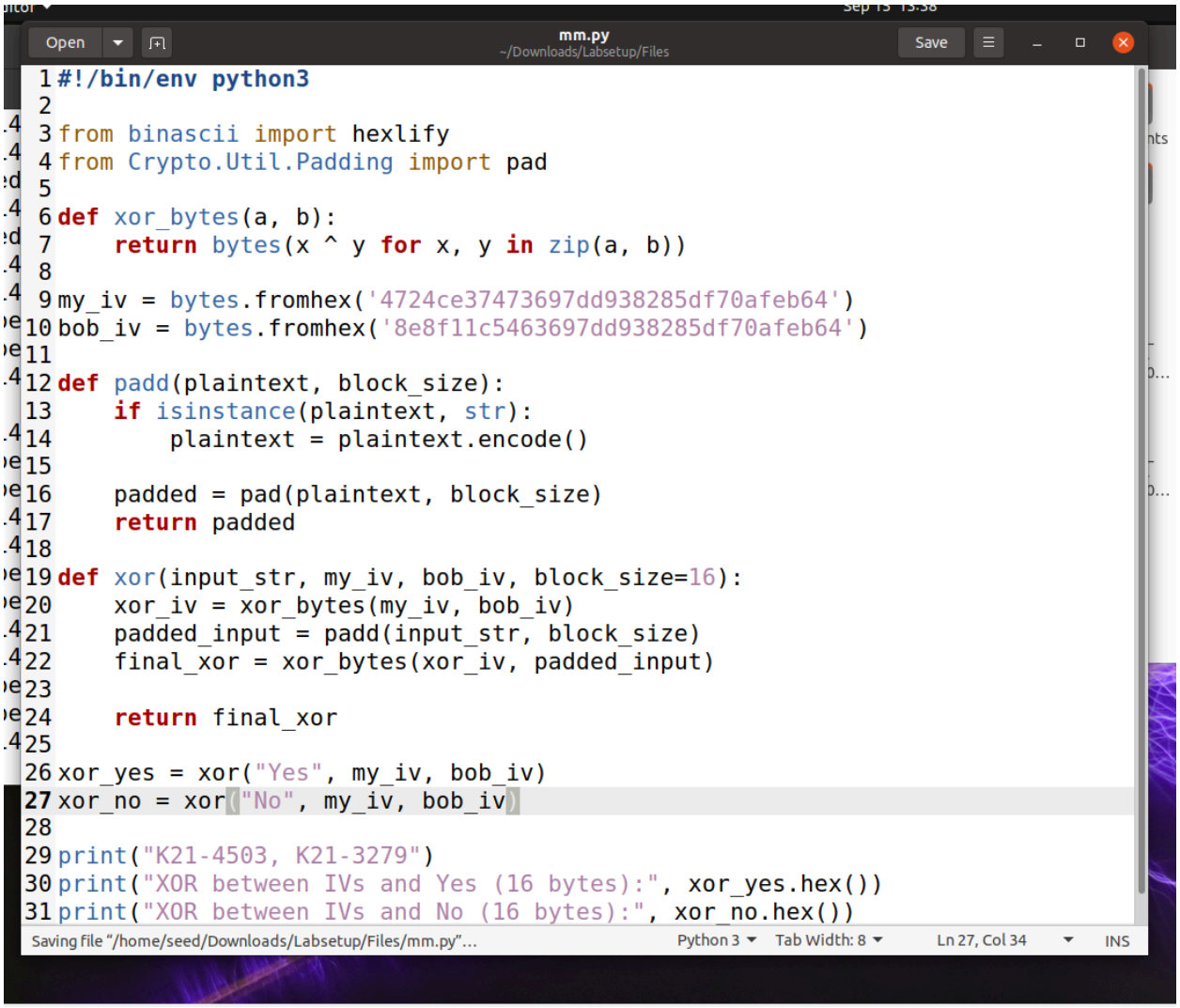
With the same IV, we can derive the content of P2 by XOR C2 with the XOR of C1 and P1. Therefore, in OFB mode, the plaintext P2 can be completely revealed if C1 and P1 are known.

Part03:

Formula:

```
eP1 = aIV XOR eIV XOR PG1

eC1 = E(eP1 XOR eIV)

eC1 = E(aIV XOR eIV XOR PG1 XOR eIV)

eC1 = E(aIV XOR PG1)
```

Code:

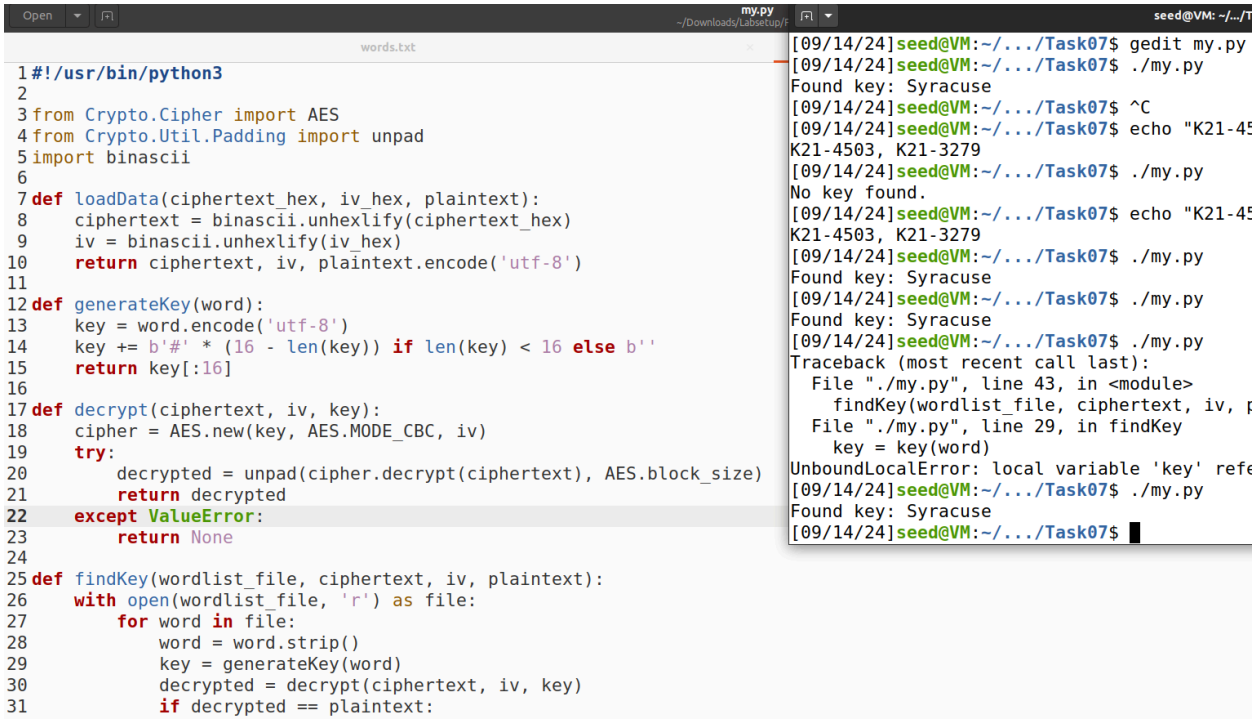

Open
mm.py
~/Downloads/Labsetup/Files
Save
#!/bin/env python3
from binascii import hexlify
from Crypto.Util.Padding import pad
def xor_bytes(a, b):
return bytes(x ^ y for x, y in zip(a, b))
my_iv = bytes.fromhex('4724ce37473697dd938285df70afeb64')
bob_iv = bytes.fromhex('8e8f11c5463697dd938285df70afeb64')
def padd(plaintext, block_size):
if isinstance(plaintext, str):
plaintext = plaintext.encode()
padded = pad(plaintext, block_size)
return padded
def xor(input_str, my_iv, bob_iv, block_size=16):
xor_iv = xor_bytes(my_iv, bob_iv)
padded_input = padd(input_str, block_size)
final_xor = xor_bytes(xor_iv, padded_input)
return final_xor
xor_yes = xor("Yes", my_iv, bob_iv)
xor_no = xor("No", my_iv, bob_iv)
print("K21-4503, K21-3279")
print("XOR between IVs and Yes (16 bytes):", xor_yes.hex())
print("XOR between IVs and No (16 bytes):", xor_no.hex())
Saving file "/home/seed/Downloads/Labsetup/Files/mm.py"...
Python 3
Tab Width: 8
Ln 27, Col 34
INS

Output:

```
Bob's ciphertex: 10ce0ad963ec386dd9a01c30e2b1cc98
The IV used    : c722a541eb93caa9f6723a0cb55f76a8

Next IV        : 34fda172eb93caa9f6723a0cb55f76a8
Your plaintext : 8d6ccec10d0d0d0d0d0d0d0d0d0d0d0d
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 : 250823c20d0d0d0d0d0d0d0d0d0d0d0d
Your ciphertext: 7dba782c47199d73986e1ac30fdf4b743
1

Next IV        : 86b684ecac341d635a269ff73cbd872a
```

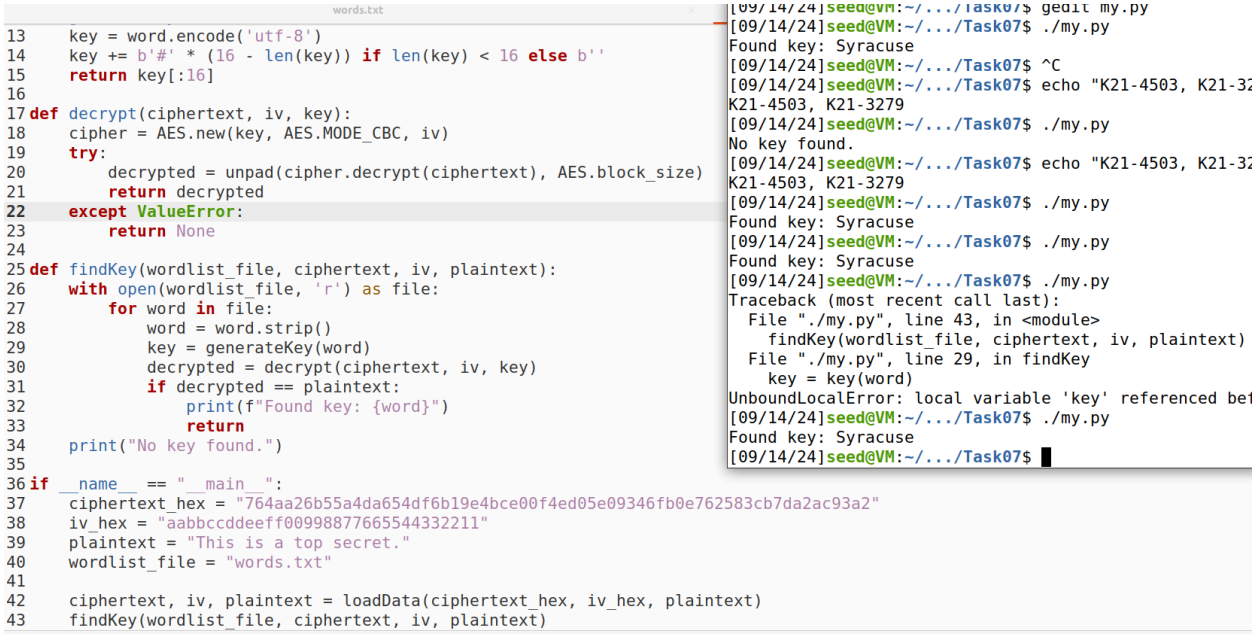
Task07:

Code:



```
#!/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)
        return decrypted
    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:
```

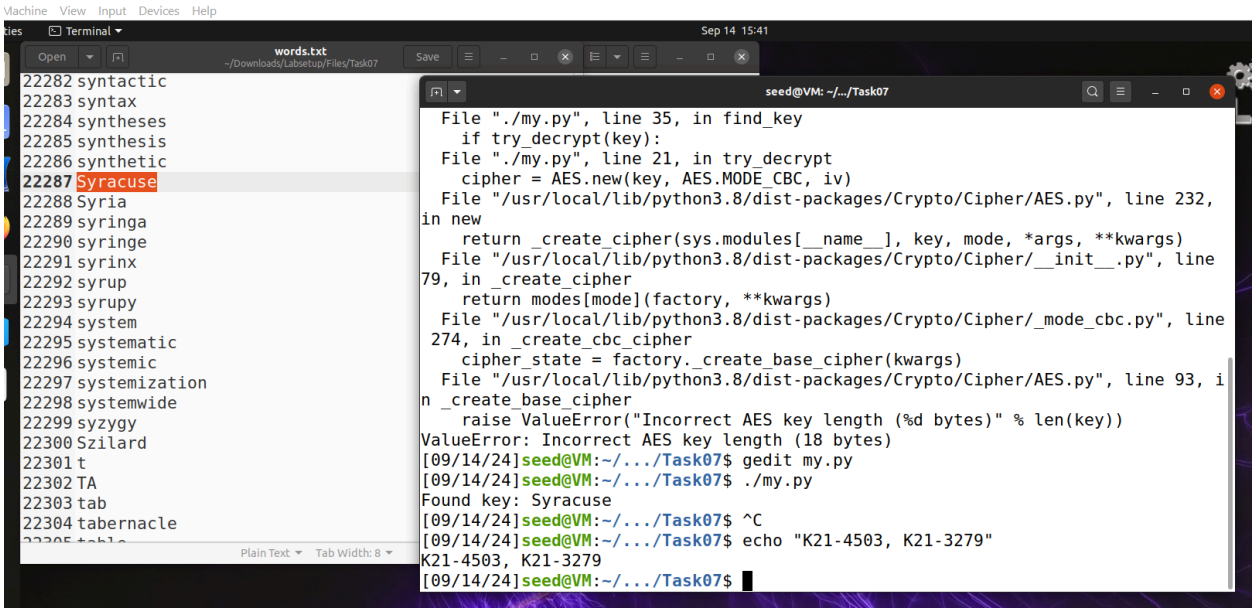
```
[09/14/24]seed@VM:~/.../Task07$ gedit my.py
[09/14/24]seed@VM:~/.../Task07$ ./my.py
Found key: Syracuse
[09/14/24]seed@VM:~/.../Task07$ ^C
[09/14/24]seed@VM:~/.../Task07$ echo "K21-45
K21-4503, K21-3279
[09/14/24]seed@VM:~/.../Task07$ ./my.py
No key found.
[09/14/24]seed@VM:~/.../Task07$ echo "K21-45
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, p
  File "./my.py", line 29, in findKey
    key = key(word)
UnboundLocalError: local variable 'key' refe
[09/14/24]seed@VM:~/.../Task07$ ./my.py
Found key: Syracuse
[09/14/24]seed@VM:~/.../Task07$
```



```
    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)
        return decrypted
    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 = "aabbccddeeff00998877665544332211"
    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, K21-32
K21-4503, K21-3279
[09/14/24]seed@VM:~/.../Task07$ ./my.py
No key found.
[09/14/24]seed@VM:~/.../Task07$ echo "K21-4503, K21-32
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 bef
[09/14/24]seed@VM:~/.../Task07$ ./my.py
Found key: Syracuse
[09/14/24]seed@VM:~/.../Task07$
```

Output:


Machine View Input Devices Help
Terminal
Sep 14 15:41
Open
words.txt
~/Downloads/Labsetup/Files/Task07
Save
22282 syntactic
22283 syntax
22284 syntheses
22285 synthesis
22286 synthetic
22287 Syracuse
22288 Syria
22289 syringa
22290 syringe
22291 syrinx
22292 syrup
22293 syrupy
22294 system
22295 systematic
22296 systemic
22297 systemization
22298 systemwide
22299 syzygy
22300 Szilard
22301 t
22302 TA
22303 tab
22304 tabernacle
Plain Text
Tab Width: 8
seed@VM: ~/.../Task07
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, in _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$