Alexandria Deleon

**Lab 4: Secret-key Encryption**

**Task 1**

To decrypt the ciphertext, I ordered the letters from most to least frequent and then changed each letter to the corresponding frequency of letters in the English language. This was put in a separate file which I continuously referenced as I switched letters until the text made sense. This process is shown in the image below.

**A screen shot of a computer

AI-generated content may be incorrect.**

The result of the decrypted text is shown below.

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer screen

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

**Task 2**

The image below shows me trying at least three different ciphers.

A screenshot of a computer

AI-generated content may be incorrect.

**Task 3**

The image below shows be encrypting the picture into a new file.

A screen shot of a computer

AI-generated content may be incorrect.

This image shows the original picture.

A red oval with a white background

AI-generated content may be incorrect.

This image shows the encrypted picture.

A close-up of a computer screen

AI-generated content may be incorrect.

Observations: I can gather some information from the encrypted picture, which is that the original shows an oval and a rectangle. The colors, however, are not the same.

Afterwords, I tried to encrypt a photo I found on the internet, but the photos were not viewable.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Task 4**

A screenshot of a computer

AI-generated content may be incorrect.

This shows me encrypting a file using ECB. As shown, the file has 6 bytes while the output file has 16 bytes, meaning that padding occurs during ECB encryption.

A screenshot of a computer

AI-generated content may be incorrect.

This shows me encrypting a file using CBC. As shown, the file encrypted needed to be padded, meaning that padding occurs during CBC encryption.

A screenshot of a computer

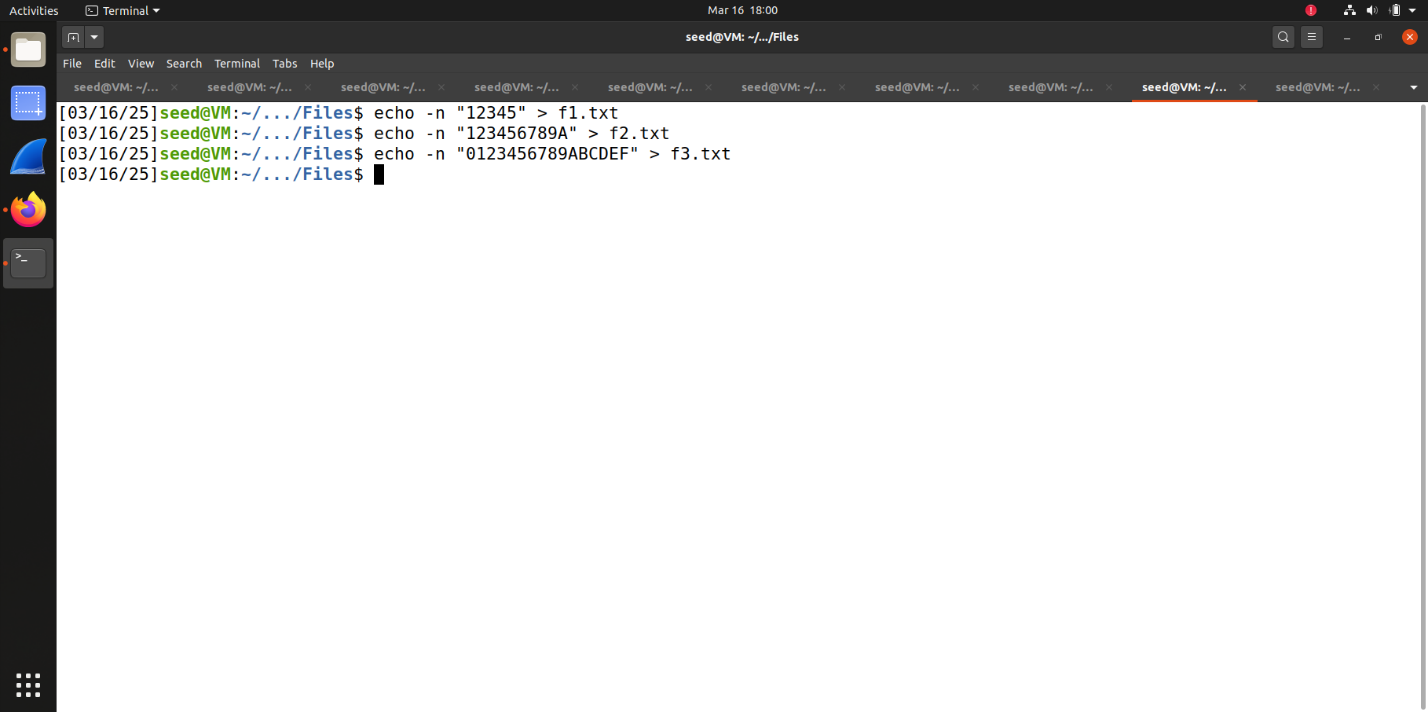
AI-generated content may be incorrect.

This shows me encrypting a file using CFB. CFB doesn’t need padding as it takes outputs of the previous block.

A screenshot of a computer

AI-generated content may be incorrect.

This shows me encrypting a file using OFB. OFB doesn’t need padding as it takes outputs of the previous block.



This shows me creating the three files for encryption.

A screenshot of a computer

AI-generated content may be incorrect.

This shows me encrypting and decrypting the first file.

A screenshot of a computer

AI-generated content may be incorrect.

This shows me encrypting and decrypting the second file.

A screenshot of a computer

AI-generated content may be incorrect.

This shows me encrypting and decrypting the third file.

**Task 5**

**A screen shot of a computer

AI-generated content may be incorrect.**

This shows me creating the file with 1000 bytes.

**A screenshot of a computer

AI-generated content may be incorrect.**

This shows me encrypting the file using ECB.

**A screenshot of a computer

AI-generated content may be incorrect.**

This shows me corrupting the file using bless.

A screenshot of a computer

AI-generated content may be incorrect.

This shows me decrypting the file.

Before I check on how much of the file was recovered, I predict that with ECB encryption, it will have the least recovered because it is the simplest encryption method. I predict that CBC and CFB will result in the same number of bytes recovered as they are similar methods of encryption with similar security. I predict that OFB will result in the most number of bytes recovered as it is more secure than CBC, CFB, and ECB.

A screenshot of a computer

AI-generated content may be incorrect.

I made a python program to find the number of different bytes in the decrypted file.

A screen shot of a computer

AI-generated content may be incorrect.

This shows the number of different bytes in the decrypted file after encrypting with ECB. As shown, the number of different bytes is 16.

I repeated this process for the other three encryption types. This process is shown below.

**A screenshot of a computer program

AI-generated content may be incorrect.**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**A screenshot of a computer program

AI-generated content may be incorrect.**

As shown, I was partly right in my prediction. Using CBC and CFB did result in the same number of different bytes, 17, but this is more than the number of bytes different from using ECB. As I predicted, OFB did result in the least number of different bytes with a value of 1.

**Task 6**

6.1

**A screenshot of a computer

AI-generated content may be incorrect.**

This is showing me encrypting a text file with the same IV and different IVs. An IV needs to be unique because using the same IV results in the same ciphertext.

6.2

If you replace OBF in the experiment with CFB, you can get p2 if you XOR p1 and c1 and then XOR that output with c2.

6.3

For this section, I first assumed that p1 was “Yes” and then assumed the IVs used. I then created a program that would add padding to match the block size.

A screen shot of a computer

AI-generated content may be incorrect.

Next, I created p2 by XOR-ing p1 (Yes) with the first IV and then XOR-ing that output with the next IV. I then got c2 by querying with p2 and using xxd -p.

A screenshot of a computer program

AI-generated content may be incorrect.

c2 was shown to have the same first block as c1. Meaning that Bob’s message was “Yes.”

A screenshot of a computer

AI-generated content may be incorrect.

This shows me verifying my conclusion.

**Task 7**

**A screen shot of a computer

AI-generated content may be incorrect.**

This is the program I made to find the encryption key utilizing the words.txt file.

A screenshot of a computer

AI-generated content may be incorrect.

I was able to find the key when executing the program.