Block Cipher Operation

1 OVERVIEW

The learning objective of this lab is to get familiar with the block cipher operations. The lab focuses on encryption algorithms, encryption modes, paddings, initialization vector and error propagation.

You need to take screenshots of key steps and answer questions for your lab report. For the weekly lab, you can discuss with your group. However, you need to prepare your report with your own screenshots. You have one week to prepare your report. Please submit your report to Canvas.

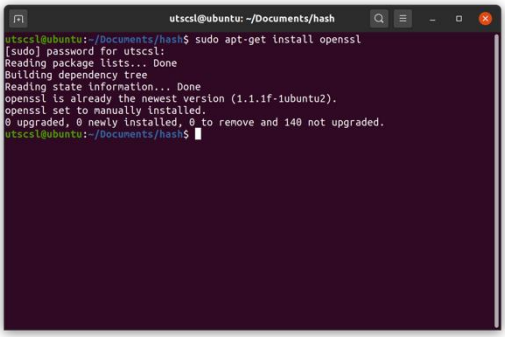
2 PRELIMINARIES

2.1 Install openssl

OpenSSL is a robust, commercial-grade, and full-featured toolkit for the Transport Layer Security (TLS) and Secure Sockets Layer (SSL) protocols. It is also a general-purpose cryptography library. More information about openssl can be found in https://www.openssl.org/

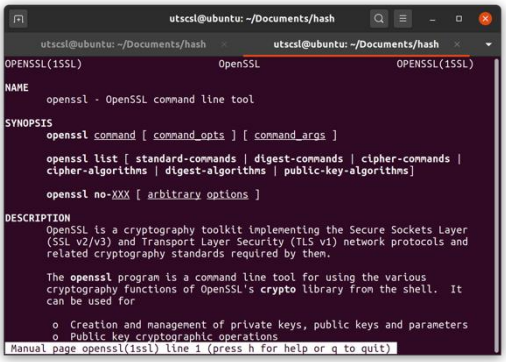
Install openssl, if openssl has not been installed, with the following command

$ sudo apt-get install openssl



Check the manual page on the openssl and supported algorithms and more options with the following command

$ man openssl



2.2 Install bless

We will be using a tool called bless to view encrypted files in hexadecimal. To install this tool, you can use:

$ sudo apt-get install bless

3 TASKS

3.1 Encryption using Different Ciphers and Modes

In this task, we will play with various encryption algorithms and modes. You can use the following *openssl enc* command to encrypt/decrypt a file. To see the manuals, you can type *man openssl* and *man enc*.

$ openssl enc -ciphertype -e -in plain.txt -out cipher.bin -K 00112233445566778889aabbccddeeff -iv 0102030405060708

Please replace the ciphertype with a specific cipher type, such as -aes-128-cbc, -bf-cbc, -aes-128-cfb. In this task, you should try at least 3 different ciphers. You can find the meaning of the command-line options and all the supported cipher types by typing "*man enc*". We include some common options for the openssl enc command in the following:

-in <file>	input file
-out <file>	output file
-e	encrypt
-d	decrypt
-K/-iv	key/iv in hex is the next argument
-[pP]	print the iv/key (then exit if -P)

3.2 Encryption Mode – ECB vs. CBC

We would like to encrypt uts.bmp, so people without the encryption keys cannot know what is in the picture. Please encrypt the file using the ECB (Electronic Code Book) and CBC (Cipher Block Chaining) modes, and then do the following:

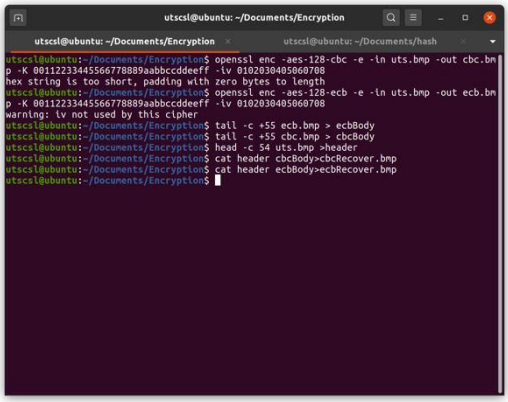
Encrypt the image using the ECB (Electronic Code Book) and CBC (Cipher Block Chaining) modes, save them separately (you can name the encrypted files ecb.bmp and cbc.bmp).

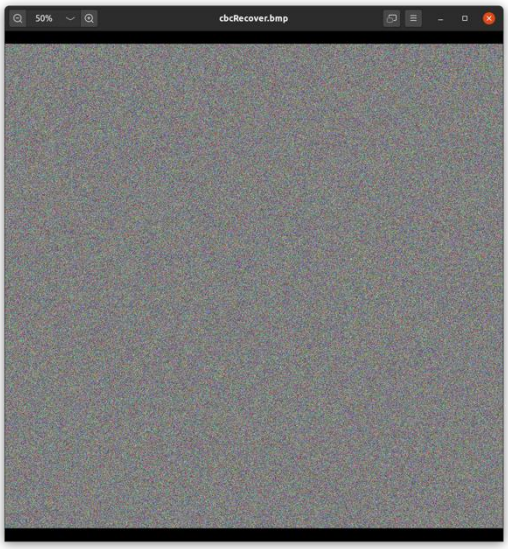
We can use the bless hex editor tool to directly modify binary files. We can also use the following commands to get the header from ecb.bmp, the data from cbc.bmp (from offset 55 to the end of the file), and then combine the header and data together into a new file.

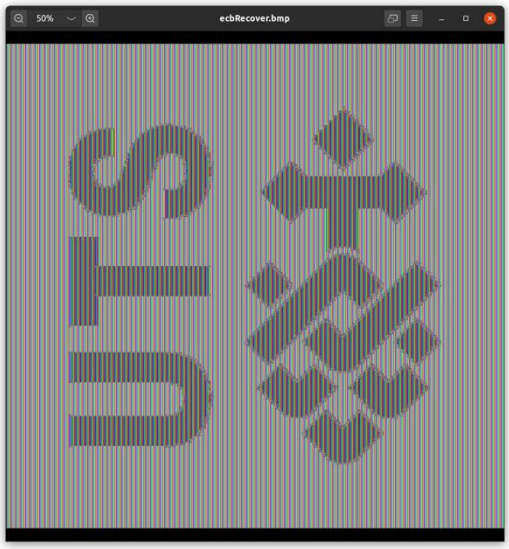
Display the encrypted picture using a picture viewing program.

Q: Can you derive any useful information about the original picture from the encrypted picture? Please explain your observations.

An example is given below







3.3 Padding

For block ciphers, when the size of a plaintext is not a multiple of the block size, padding may be required. All the block ciphers normally use PKCS#5 padding, which is known as standard block padding. We will conduct the following experiments to understand how this type of padding works:

Use ECB, CBC, CFB, and OFB modes to encrypt a file (you can pick any cipher). Please report which modes have paddings and which ones do not. For those that do not need paddings, please explain why.

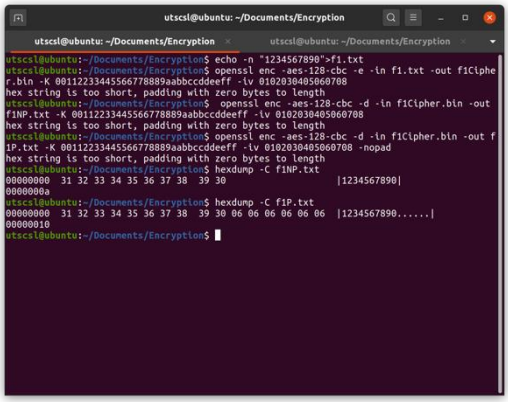
Let us create two files, which contain 10 bytes, and 16 bytes, respectively. We can use the following "echo -n" command to create such files. The following example creates a file f1.txt with length 10 (without the -n option, the length will be 11, because a newline character will be added by echo):

$ echo -n "1234567890" > f1.txt

We then use "openssl enc -aes-128-cbc -e" to encrypt these three files using 128-bit AES with CBC mode. Please describe the size of the encrypted files.

We would like to see what is added to the padding during the encryption. To achieve this goal, we will decrypt these files using "openssl enc -aes-128-cbc -d". Unfortunately, decryption by default will automatically remove the padding, making it impossible for us to see the padding. However, the command does have an option called "-nopad", which disables the padding, i.e., during the decryption, the command will not remove the padded data. Therefore, by looking at the decrypted data, we can see what data are used in the padding. Please use this technique to figure out what paddings are added to the two files.

An example is given as below



3.4 Error Propagation – Corrupted Cipher Text

To understand the error propagation property of various encryption modes, we would like to do the following exercise:

Create a text file that is 64 bytes long.

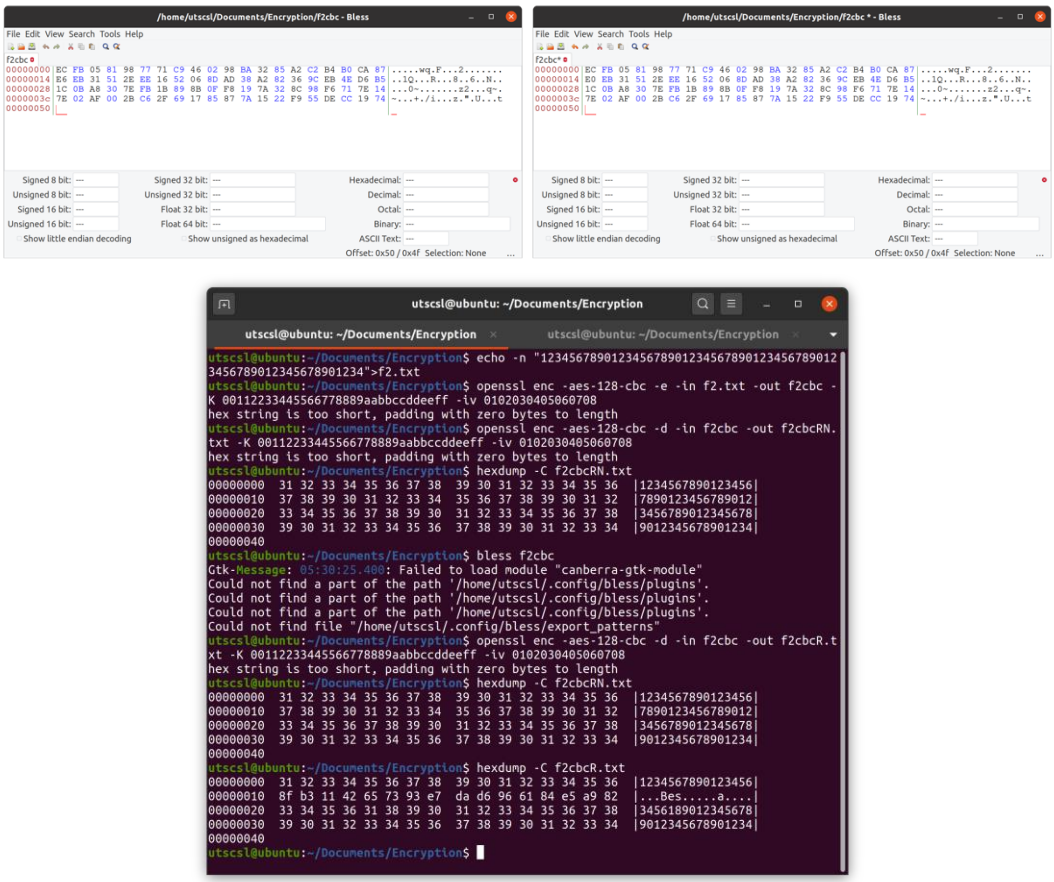
Encrypt the file using the AES-128 cipher.

Unfortunately, a single bit of the encrypted file got corrupted. You can achieve this corruption using the bless hex editor.

Decrypt the corrupted ciphertext file using the correct key and IV.

Q: How much information can you recover by decrypting the corrupted file, if the encryption mode is ECB, CBC, CFB, OFB, and CTR respectively? Please find it out after you finish this task and provide justification.

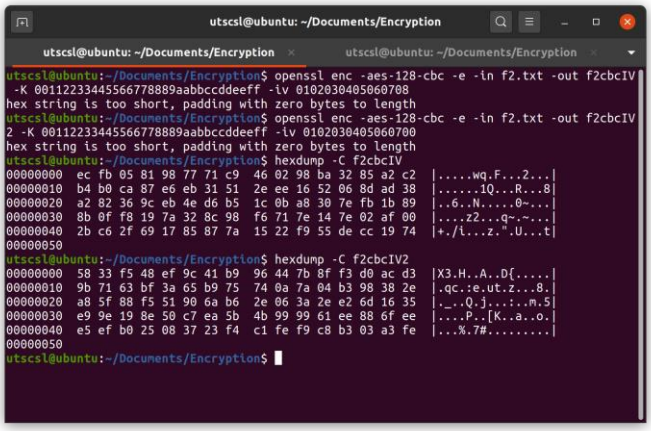
An example of CBC is given as below



3.5 Task 6: Initial Vector (IV)

Most of the encryption modes require an initial vector (IV). Properties of an IV depend on the cryptographic scheme used. If we are not careful in selecting IVs, the data encrypted by us may not be secure at all, even though we are using a secure encryption algorithm and mode. The objective of this task is to help students understand the problems if an IV is not selected properly.

A basic requirement for IV is uniqueness, which means that no IV may be reused under the same key. To understand why, please encrypt the same plaintext using the same key but different IVs. Please describe your observation, based on which, explain why IV needs to be unique. An example is given as below



Read “Should CBC Mode Initialization Vector Be Secret - Defuse Security.pdf” and “Initialization vector - Wikipedia.pdf” and answer questions.

Q: What properties the IV should have? Why?

4 Lab Summary and Discussion

Summarise and discuss the lab using your words.