# Crypto and Symmetric Key Encryption Applications

## Lab Environment

This lab has been adapted from SEED Project. https://seedsecuritylabs.org/labsetup.html

Install VirtualBox and Ubuntu 20.04 on your computer according to the instructions given the link above.

## Task 1: Frequency Analysis

It is well-known that monoalphabetic substitution cipher (also known as monoalphabetic cipher) is not secure, because it can be subjected to frequency analysis.

In this lab, you are given a cipher-text (Bil420ciphertext.txt) that is encrypted using a monoalphabetic cipher; each letter in the original text is replaced by another letter, where the replacement does not vary (i.e., a letter is always replaced by the same letter during the encryption).

Your job is to find out the original text using frequency analysis. It is known that the original text is an English article.

**Guidelines**. Using the frequency analysis, you can find out the plaintext for some of the characters quite easily. For those characters, you may want to change them back to its plaintext, as you may be able to get more clues. It is better to use capital letters for plaintext, so for the same letter, we know which is plaintext and which is ciphertext. You can use the tr command to do this. For example, in the following, we replace letters a, e, and t in in.txt with letters X, G, E, respectively; the results are saved in out.txt.

```
$ tr 'aet' 'XGE' < in.txt > out.txt
```

There are many online resources that you can use. Four useful links in the following:

http://www.richkni.co.uk/php/crypta/freq.php: This website can produce the statistics for a ciphertext, including the single-letter frequencies, bigram frequencies (2-letter sequence), and trigram frequencies (3-letter sequence), etc.

https://en.wikipedia.org/wiki/Frequency_analysis: This Wikipedia page provides frequencies for a typical English plaintext.

https://en.wikipedia.org/wiki/Bigram: Bigram frequency.

https://en.wikipedia.org/wiki/Trigram: Trigram frequency.

## Task 2: Encryption Mode - ECB vs. CBC

The file pic_original.bmp is bundled with this lab and it contains our university's logo. We would like to encrypt this picture, so people without the encryption keys cannot know what is in the picture.

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
```

Replace the ciphertype with a specific cipher type, such as -aes-128-cbc, -bf-cbc, -aes-128-cfb, etc. 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". 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)

Encrypt the file using the ECB (Electronic Code Book) and CBC (Cipher Block Chaining) modes, and then do the following:

1. Let us treat the encrypted picture as a picture and use a picture viewing software to display it. However, For the .bmp file, the first 54 bytes contain the header information about the picture, we have to set it correctly, so the encrypted file can be treated as a legitimate .bmp file. We will replace the header of the encrypted picture with that of the original picture. We can use the bless hex editor tool (already installed on our VM) to directly modify binary files. We can also use the following commands to get the header from p1.bmp, the data from p2.bmp (from offset 55 to the end of the file), and then combine the header and data together into a new file.

$ head -c 54 p1.bmp > header

$ tail -c +55 p2.bmp > body

$ cat header body > new.bmp

2. Display the encrypted picture using a picture viewing program (SEED labs have installed an image viewer program called eog on VM). Can you derive any useful information about the original picture from the encrypted picture? Please explain your observations.

Select a picture of your choice, repeat the experiment above, and report your observations.

### Task 3: Error Propagation - Corrupted Cipher Text

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

1. Create a text file that is at least 1000 bytes long.

2. Encrypt the file using the AES-128 cipher.

3. Unfortunately, a single bit of the 55th byte in the encrypted file got corrupted. You can achieve this corruption using the bless hex editor. (If this editor does not work in your VM, you can download Hex Editor Neo for Windows OS and you can manipulate file with this app.)

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

Answer the following question: How much information can you recover by decrypting the corrupted file, if the encryption mode is ECB, CBC, CFB, or OFB, respectively? (You can take advantage of our lecture slides to see the details of these modes) Please answer this question before you conduct this task, and then find out whether your answer is correct or wrong after you finish this task. Provide justification.