Lab 1: Secret-Key Encryption Lab

1 Overview

The learning objective of this lab is to get you familiar with symmetric encryption algorithms and the various encryption modes (ECB, CBC, CTR). [1]

2 Lab Environment

In this lab, you will use `openssl` commands and libraries. The `openssl` binaries should already be installed on the **019 lab machines** but if you are using your own Linux VM you *may* need to install the `openssl`-package.

Installing OpenSSL. In case you need to install `openssl` package, you can do so using the following command:

```
$ sudo apt-get install openssl
```

It should be noted that the above command only install the `openssl` binaries. If you want to use `openssl` libraries in your programs, you need to install several other things for the programming environment, including the header files, libraries, manuals, etc. You can use the following command to do this:

```
  $ apt-get source openssl

Untar the tar ball, and run the following commands.
You should read the INSTALL file first:

  $ ./config
  $ make
  $ make test
  $ sudo make install
```

3 Lab Tasks

3.1 Task 1: Create a text file

You can use any editor to create your `plain.txt` file and you can type any text you want as long as the first line includes **your name** as shown in the sample text below:

> *Hello my name is Angeliki Zavou.*
> *This is my plaintext file!*

If you want to use the command-line to achieve this, you can type the commands shown below:

[1] This lab is based on the lab created by Wenliang Du, at Syracuse University, which was partially funded by the National Science Foundation under Award No. 1303306 and 1318814. Permission is granted to copy, distribute and/or modify this document under the terms of the Creative Commons Attribution-NonCommercialShareAlike 4.0 International License.

```
$ touch plain.txt
$ gedit plain.txt
$ cat plain.txt
Hello my name is Angeliki Zavou.
This is my plaintext file!
$
```

The `cat` command is used to show the content of the file `plain.txt`.

3.2 Task 2: Encryption a file using different modes

In this task, you will use various encryption algorithms and modes. You can use the following `openssl enc` command to encrypt/decrypt a file.

Note: to see the manuals, you can type `man openssl` and `man enc` in the Terminal's command line or visit the link `https://www.openssl.org/docs/man1.0.2/man1/openssl-enc.html`

The `openssl` command used to encrypt a file is:

```
$ openssl enc -ciphertype -e -in infile -out outfile \
              -K key -iv initial_vector
```

In the command above:

- `-e`: indicates the encryption operation
- `ciphertype`: stands for the cipher and mode to be used (e.g., `aes-128-ecb`)
- `infile`: is the input file to be encrypted
- `outfile`: is the created file that contains the encrypted data
- `key`: is the key used for the encryption
- `initial_vector`: is the initialization vector to be used

3.2.1 Encrypt `plain.txt` file.

Your task is to encrypt your `plain.txt` file using the `AES` algorithm and the same key `K` and `IV` (shown in the table below) for modes `CBC` and `OFB`:

```
$ openssl enc -ciphertype -e -in plain.txt -out outfile \
              -K 00112233445566778889aabbccddeeff \
              -iv 0102030405060708090a0b0c0d0e0f
```

Please replace the `ciphertype` with the specified cipher types, and name the produced encrypted files as `text_aes_cbc.enc` and `text_aes_ofb.enc` respectively. These two files should be included in your submission.

3.2.2 Verify the output.

Most of the data in the encrypted files will not be printable. To observe the contents of these files, you can use the command-line hexadecimal viewing tool 'xxd' as shown in the example below:

```
$ xxd  text_aes_ofb.enc
0000000: 4af2 f028 0567 c8dd 2eb6 1873 ee53 3d54  J..(.g.....s.S=T
0000010: f3d1 466d f086 2648 6576 a8f5 907d 3307  ..Fm..&Hev...}3.
0000020: b862 c669 ac32 9e5f 89f7 3cd9 4ab9 2033  .b.i.2._..<.J. 3
0000030: e910 4cd5 dc0d
```

3.3 Task 3: Encrypting an image - ECB vs. CBC

3.3.1 Download and encrypt the image file

For this task, you need to download the picture file `pic_original.bmp` from Blackboard. The goal is to encrypt this picture, so people without the encryption key cannot know what is in the picture.

For the encryption, you will be using the same key `K` and `IV` (from the previous task) but block-modes: `ECB` and `CBC` this time. Please replace the `ciphertype` with the specified cipher types, and name the produced encrypted files as `pic_ecb.bmp` and `pic_cbc.bmp` respectively.

3.3.2 Replace the encrypted headers

Once the two encrypted files are generated the appropriate header need to be added so that the image viewing software can **recognize** the file as an image. For `bmp` files, the first 54 bytes contain the header information. Therefore, you need to extract the header from the original image `pic_original.bmp` and use it to replace the header of the encrypted files.

The commands to achieve this are shown below:

```
$ head -c 54 pic_original.bmp > header
$ tail -c +55 pic_ecb.bmp > body_ecb
$ cat header  body_ecb > new_ecb.bmp
$ tail -c +55 pic_cbc.bmp > body_cbc
$ cat header  body_cbc > new_cbc.bmp
$
```

For this part you need to include the files `new_ecb.bmp` and `new_cbc.bmp` in your submission.

3.3.3 ECB vs CBC

Use any image viewing software to open the encrypted images and compare them with the original file `pic_original.bmp`.

In your `Lab1.txt` file answer the following questions:

- **Q1**: In which of the two modes, are patterns of the original image still visible and why?
- **Q2**: In which of the two modes, the colors are still visible and why?

4 Useful Links

- `https://www.openssl.org/docs/man1.0.2/man1/openssl-enc.html`

5 Submission

You need to submit on Blackboard a `Firstname_Lastname_Lab1.tar.gz` including the **four encrypted files** as well as your `Lab1.txt`.