Programming Assignment 1 - Part 1

Middle East Technical University, Computer Engineering 29/10/2018

Overview

Here is your first assignment-part1. We are also planning to give a second part in a couple of days, which is independent from this part1. So, you can start this homework without waiting for the second part.
In this assignment, you will exercise with different encryption/decryption algorithms and modes with openssl. To learn more details, you can write “man openssl” and “man enc” in your terminal. In this assignment, there are several tasks that you should follow. You should read through the rest of this document carefully to find out the exact block cipher implementation details, the expected operation of your program (including input-output format) and submission guidelines. Your program should follow the exact format specifications, as your assignment will most likely be graded using automated scripts.

1.Task

Task 0: Warmup in different encryption algorithms and modes

Do not submit this part!
This part is only for warm-up. In this task, you will play with different ciphers and modes.

Listing 1: Sample Terminal work – Format is explained.

```
1 openssl enc ciphertype -e -in input.txt -out cipher \
2 -K 00112233445566778889aabbccddeeff \
3 -iv 0102030405060708
```

Replace ciphertype with a real cipher type like -aes-128-cbc. Try at least 3 different cipher and mode combinations. You can get more details about the format using "man enc".

Task 1: Image Encryption with ECB and CBC

There is a given sample image named "original.bmp". We would like to encrypt this image, so people without the encryption keys cannot see the image properly. Please use ECB(Electronic Code Book) mode and CBC(Cipher Block Chaining) mode, then do these steps:

1. Use image viewing software to view the original image, then encrypt it using ECB and CBC modes of AES-128 with any key and IV of your choice.

2. Display the encrypted image using any picture viewing software. First, you need to fix the header part. Bmp files contain 54 bytes of header information about the image. We need to set it correctly so that the encrypted file can be treated as a normal .bmp extended file. Replace the header of the encrypted image with the one of the original image. You can use a hex editor tool like "ghex" or another one you choose to directly change the binary files.

Can you observe any useful information about the original picture from the encrypted picture? Please explain your observations for each encryption mode in "report.pdf" giving the task number.

Task 2: Programming with Openssl crypto library

In this task, you will use an OpenSSL API called EVP which offers a common interface for various encryption algorithms and C++ as programming language. To ask EVP to use a specific encryption algorithm, we only need to pass our choice to the EVP interface. A sample code is given in here. Please get yourself familiar with this program, and then do the following exercise. Here is the story: You are given a plaintext and a ciphertext. Your ultimate goal is to find the secret key. But you are not alone in this adventure. There are some clues that will help you. First, you know that "aes-128-cbc" is used to generate the ciphertext from the plaintext, and you know that IV are all zeros (not the ASCII character '0'). The last thing you learned about the key is that it is an English word shorter than 16 characters, the word that can be found from a typical English dictionary (we provide a sample dictionary, "words.txt"). Since the word has less than 16 characters (i.e. 128 bits), space characters (hexadecimal value 0x20) are appended to the end of the word to form a key of 128 bits. Sample plaintext and ciphertext are provided in the compressed file. In the program, you will read the plaintext from "plaintext.txt". The provided ciphertext to be read will be "ciphertext". Your dictionary file name to be used is "words.txt". In the evaluation, we plan to give different ciphertexts with different keys and expect to see the key we used as output from your program. Plaintext and words will stay the same.

Note-1:The input file length should be 21 characters. Some editors may add specific characters to the end of the file, if you open and modify it. If that happens, you can use a hex editor like ghex to remove the special character.

Note-2:To compile your program, you need to include header files in openssl, and link to openssl libraries. A sample compilation can be:

Listing 2: Sample compilation command.

```
g++ -o out crypto.cpp -lcrypto -ldl
```

Submission

Part1 and Part2 (to be announced later) will be submitted together. Do not submit separately. You need to submit your program file named "crypto.cpp" and a report named "report.pdf". Your code should only print the secret key when we compile and run your code. We do not want to see any output file as an intermediate step. If you create any, remove it in every step.
Submission will be done via COW in zip format. Your submission also should have a README file. In your README, include your name, student ID, the names of external libraries that you used in your code, and describe how to run your program. You can also write anything you want us to know that we should take into consideration when grading. Zip all files of your program and the README file. Use your student id to name the zip file. Example naming "e1234567.zip".

Regulations

- Due date: 14 November 2018, Wednesday, 23:55.
- Submission: Electronically. You will be submitting a compressed file containing your source code and report through COW.
- Cheating: We have a zero tolerance policy for cheating. All submissions must be your own work and you must work alone.
- Newsgroup: Please follow Piazza for discussions and possible updates on a daily basis.