CS6500 - Network Security
Assignment 1: OpenSSL interface and performance analysis

Instructor: Krishna M. Sivalingam
Assigned on: *Feb. 2, 2022*
Due on: *Feb. 11, 2022, 11PM on Moodle*
Updated on: January 30, 2022

This assignment has two parts: (i) OpenSSL based performance benchmarking and (ii) Brute-force attack on a simple polyalphabet substitution-based encryption algorithmic.

1 OpenSSL

The objective of this assignment is to understand and use the OpenSSL (`http://openssl.org`) API, for invoking existing cryptographic algorithmic implementations. Any OpenSSL interface language/libraries such as C/C++ (default), Java and Python (pyOpenSSL), can be used.

1.1 Symmetric Encryption

The following inputs are obtained from the command line:

```
./myenc -p <oper> -a <alg> -m <mode> -k <keysize> -i <inpfile> -o <outfile>
```

The possible choices for each command line argument are:

- Operation: Enc, Dec
- Algorithm: 3DES, AES
- Mode: ECB (DES) and CBC (AES)
- Key size: depending on the algorithm one of 168 (3DES); 128 and 256 (AES)

The objective is to obtain the relevant key from the user input passphrase and call the associated **OpenSSL API routines** for encryption and decryption. The data from the input file is either encrypted or decrypted (based on the command line operation specified) and stored in the output file. Base64 encoding can also be used.

1.2 Performance evaluation

The performance metrics to be measured are: (i) Mean Block encryption time (in microseconds or nanoseconds), (ii) Mean Block Decryption time (in microseconds or nanoseconds), for each algorithm. The clock accuracy will be dependent on the underlying system call/library functions.

For each filesize given below, generate 10 different files of the same size but with different random text. Report the average per-block time across all blocks of the 10 files.

For 3DES, the parameters varied are: (i) filesize (10KB, 1MB); (ii) all modes specified.

For AES, the parameters varied are: (i) filesize (10KB, 1MB); (ii) all modes specified; (iii) two choices for the keysize (128 and 256).

The performance values are to be presented in a tabular form, one for each encryption algorithm. The table should also include the time required for a successful brute-force attack for each of the algorithms/keysize combinations, based on extrapolation of the average times obtained. Decide on a suitable table format. The report should specify the system configuration used (CPU, Memory, etc.).

2 Brute-Force Attack of Simple Encryption Algorithm

Consider the following encryption algorithm. The character set is the uppercase English alphabets, i.e. $\{a,b,c,...,z\}$. The secret key contains 5 integers (in the range of $1-10$), e.g. $\{1,4,9,8,2\}$. The key is repeated as many times depending on the plaintext length.

The substitution is as discussed in class, $C_i =$ 'a' $+(P_i-$ 'a' $+2k_i+1) \pmod{26}$, with the output character set also being $\{a,b,c,...,z\}$. Spaces in the plaintext are NOT encrypted, they are retained as is. Thus, the key will apply ONLY to the alphabets in the range 'a' – 'z', repeated for each set of 5 characters.

You will be given as input a set of files. Each file will contain the ciphertext of a 35-byte plaintext phrase. The ciphertext can be stored using a C-language string of length 36 characters (or equivalent, in other languages). The plaintext will consist of 6 common dictionary-based English words (each of length 5 characters), with exactly ONE blank space between two consecutive words. The paragraph need not have any valid meaning.

For an example plaintext phrase of:

```
amend birth evade climb champ sheep
```

The corresponding ciphertext with the example key, $\{1,4,9,8,2\}$, given above is:

dvxei erkkm hetuj fubdg fqtdu vqxvu

Your program will run the decryption algorithm using all possible key combinations. The program will then print the corresponding plaintext, the secret key (as a set of numbers) and the total time taken to break the encryption, for each input file.

3 What to Submit on IITM Moodle

A single tar.gz file with name ROLLNO-Lab1.tgz containing:

- Source files and Makefile for OpenSSL and Brute-force Decryption Algorithm
- PDF version of the OpenSSL performance study report
- Output from the Decryption program for each given input file
- A README File that explains how to compile and run the programs; whether your programs works correctly or whether there are any known bugs/errors in your program.

4 Grading

- OpenSSL Symmetric Encryption and Decryption: 35 points (20 each for AES and 3DES)
- OpenSSL Performance Study and Report: 30 points (10 per algorithm)
- Brute-force Decryption Algorithm: 25 points
- Viva Voce Session (to answer questions about the program, demo. of running code, etc.): 10 points

5 Policies

- This is an INDIVIDUAL assignment. Please refer to first day handout (on Moodle) regarding penalties for any form of academic dishonesty, plagiarism, etc. There should be no downloaded code.
- Plagiarism Checking Software will be used.
- The Brute-force decryption algorithm can be written in your favorite language.
- You can refer:
  - `https://wiki.openssl.org/index.php/Libcrypto_API`
  - `https://pypi.org/project/pyOpenSSL/`
  - `https://wiki.openssl.org/index.php/EVP_Symmetric_Encryption_and_Decryption`
  - `https://www.feistyduck.com/library/openssl-cookbook/`
  - `http://shop.oreilly.com/product/9780596002701.do`
  - `https://wordfinder.yourdictionary.com/letter-words/5/`
    for creating a dictionary of valid 5-letter English words. You may use other sources for the dictionary too.