1172/DCP1323 Introduction to Cryptography, Spring 2019

Homework 2: DES Programming

Due: 2019/3/18 (Monday)

1. This homework is about to implement the DES core function, which encrypts a block of plaintext to a block of ciphertext with a key of 64 bits.
    a. Input format: the input is an ordered pair of 64-bit key and 64-bit plaintext in hexadecimal (Hex), such as 5B5A57676A56676E 675A69675E5A6B5A.
    b. Output format: 16 hex characters, such 974AFFBF86022D1F, which is the ciphertext of the above key and plaintext.
    c. You can use the following key-plaintext-ciphertext tuple as a test sample for correctness: 5B5A57676A56676E 675A69675E5A6B5A 974AFFBF86022D1F
    d. Use C or C++ to write your code.
2. Submission to E3 with two files.
    a. The source code file with name: DES.c or DES.cpp.
    b. The output file “out.txt” that contains:
        i. 10 lines of ciphertexts for the ordered pairs of key and plaintext (one pair per line) from the file “DES-Key-Plaintext.txt”.
        ii. 10 lines of plaintexts for the ordered pairs of key and ciphertext (one pair per line) from the file “DES-Key-Ciphertext.txt”.
        iii. One line of time (in milliseconds) for the running time of each DES encryption.
3. On-site test
    a. Test site: to be announced. You need to go to the computer room for the on-site test at specified time.
    b. TA will ask you to modify your DES program for a modified specification MDES.
    c. You need to show the MDES ciphertext for the ordered pair of key and plaintext, which will be given on site.
    d. You need to show the running time for the above encryption.
4. Grade evaluation
    a. If you fail the on-site test, you fail this homework.
    b. Correctness of out.txt.
    c. Performance of your program.
5. TA will run a plagiarism checker on your programs to check plagiarism. So, write your own code, do not copy from others or anywhere.
6. You can use the following code to compute the running time of a function

```
#include <time.h>

  clock_t start, end;

  double cpu_time_used;

  start = clock();
```

```
    ... /* Do the work. */
    end = clock();
    cpu_time_used = ((double) (end - start)) / CLOCKS_PER_SEC;
```