CMPUT 333, Assignment 1, Winter 2021

(non-sliding part - 75%)

University of Alberta / Department of Computing Science

Instructor: Ioanis Nikolaidis (nikolaidis@ualberta.ca)

(weak encryption and cipher modes)

Part 1 (25%)

You are given an encrypted file `ciphertext1`. The file was encrypted using a variation of the Vigenère cipher. The plaintext is ASCII text. The key is a combination of printable ASCII characters. The code is not a "textbook" Vigenère code, but rather a slightly more complicated version with respect to how a single plaintext byte and a single key byte are combined to produce a ciphertext byte. That is, the way the encryption takes place between a single plaintext byte, `p`, and a single key byte, `k`, to produce a single ciphertext byte, `c`, can be summarized as follows: First we split the key byte into the lower 4 bits (`kl`) and the higher 4 bits (`kh`). We split the plaintext byte to its lower 4 bits (`pl`) and the higher 4 bits (`ph`). Likewise, we will generate the resulting ciphertext from two parts, the lower 4 bits (`cl`) and the higher 4 bits (`ch`).

```
ch <- map[ph][kl]
cl <- map[pl][kh]
```

where:

```
map[16][16] = {{0x8, 0x9, 0xb, 0xa, 0xe, 0xf, 0xd, 0xc, 0x4, 0x5, 0x7, 0x6, 0x2,
0x3, 0x1, 0x0 },
               {0x9, 0xb, 0xa, 0xe, 0xf, 0xd, 0xc, 0x4, 0x5, 0x7, 0x6, 0x2, 0x3,
0x1, 0x0, 0x8 },
               {0xb, 0xa, 0xe, 0xf, 0xd, 0xc, 0x4, 0x5, 0x7, 0x6, 0x2, 0x3, 0x1,
0x0, 0x8, 0x9 },
               {0xa, 0xe, 0xf, 0xd, 0xc, 0x4, 0x5, 0x7, 0x6, 0x2, 0x3, 0x1, 0x0,
0x8, 0x9, 0xb },
               {0xe, 0xf, 0xd, 0xc, 0x4, 0x5, 0x7, 0x6, 0x2, 0x3, 0x1, 0x0, 0x8,
0x9, 0xb, 0xa },
               {0xf, 0xd, 0xc, 0x4, 0x5, 0x7, 0x6, 0x2, 0x3, 0x1, 0x0, 0x8, 0x9,
0xb, 0xa, 0xe },
               {0xd, 0xc, 0x4, 0x5, 0x7, 0x6, 0x2, 0x3, 0x1, 0x0, 0x8, 0x9, 0xb,
0xa, 0xe, 0xf },
               {0xc, 0x4, 0x5, 0x7, 0x6, 0x2, 0x3, 0x1, 0x0, 0x8, 0x9, 0xb, 0xa,
0xe, 0xf, 0xd },
               {0x4, 0x5, 0x7, 0x6, 0x2, 0x3, 0x1, 0x0, 0x8, 0x9, 0xb, 0xa, 0xe,
```

```
0xf, 0xd, 0xc },
                {0x5, 0x7, 0x6, 0x2, 0x3, 0x1, 0x0, 0x8, 0x9, 0xb, 0xa, 0xe, 0xf,
0xd, 0xc, 0x4 },
                {0x7, 0x6, 0x2, 0x3, 0x1, 0x0, 0x8, 0x9, 0xb, 0xa, 0xe, 0xf, 0xd,
0xc, 0x4, 0x5 },
                {0x6, 0x2, 0x3, 0x1, 0x0, 0x8, 0x9, 0xb, 0xa, 0xe, 0xf, 0xd, 0xc,
0x4, 0x5, 0x7 },
                {0x2, 0x3, 0x1, 0x0, 0x8, 0x9, 0xb, 0xa, 0xe, 0xf, 0xd, 0xc, 0x4,
0x5, 0x7, 0x6 },
                {0x3, 0x1, 0x0, 0x8, 0x9, 0xb, 0xa, 0xe, 0xf, 0xd, 0xc, 0x4, 0x5,
0x7, 0x6, 0x2 },
                {0x1, 0x0, 0x8, 0x9, 0xb, 0xa, 0xe, 0xf, 0xd, 0xc, 0x4, 0x5, 0x7,
0x6, 0x2, 0x3 },
                {0x0, 0x8, 0x9, 0xb, 0xa, 0xe, 0xf, 0xd, 0xc, 0x4, 0x5, 0x7, 0x6,
0x2, 0x3, 0x1 }};
```

Your tasks are to:

- Determine and provide the plaintext and the key which resulted in `ciphertext1`.
- Explain in your submission what technique(s) you used to determine the key and why.
- Explain how you automated the process of deriving the key.
- Submit any source code you write to solve this problem.

Part 2 (25%)

You are given another encrypted file `ciphertext2` encrypted using the same scheme as before but using a (much) longer key than the first one. You know that the plaintext file is of a commonly found file format. The file format is *not* necessarily ASCII text.

Your tasks are to:

- Determine and provide the plaintext and key which resulted in `ciphertext2`.
- Explain how you modified/refined the technique of Part 1 to solve Part 2.
- Explain how you automated the process of deriving the key.
- Explain how you exploited the fact that the plaintext was a commonly found file format.
- Submit any source code you write to solve this problem.

Note: To get full marks on both Part 1 and Part 2, you should *not* resort to brute force / exhaustive search methods.

Part 3 (25%)

In this part of the assignment you will experiment with the various cipher modes of DES. You should review the definition of ECB, CBC, CFB, and OFB modes of operation.

Create three files:

1. one that repeats the same 8 character long pattern continuously

2. one that repeats the same 12 character long pattern continuously, and
3. one with random characters. Ensure that each file is at least a couple hundred characters long.

Your tasks are the following:

- Now encrypt each file with DES using the following command: `openssl enc -e -des-XYZ -nosalt -in fileA -out fileAXYZ.enc` where `XYZ` is `ecb`, `cbc`, `cfb`, `ofb` and `A` stands for the file number (`1-3`). So, in total you have produced twelve files. You are free to choose any key you like as long as you submit it with your assignment. All the files will also be submitted.
- Inspect the contents of the `.enc` files. Point out what you observe (i) with respect to their sizes vs. the plaintext files, and (ii) with respect to seeing any patterns in their contents. *Explain* your observations (what is the cause of what you observe?).
- Next, create a "error" version of each file (name these versions `fileAXYZerror.enc` following the previous convention) where each comes from the error free file but where you have replaced a single byte (close to the middle of each file) by a different byte. These files will also be submitted.
- Try now to decrypt each of the `fileAXYZerror.enc` and check the decryption outcomes. Describe what was the impact of the "error" to the decryption outcome and explain any differences in the outcome you may have noticed.
- Repeat the generation of twelve files (now called `fileAXYZsalted.enc`) and their errored versions (`fileAXYZerrorsalted.enc`), but this time do *not* use the `-nosalt` flag. What is the result of the decryption of the "errored" files now? Explain (to the best of your knowledge) the reasons for the particular observed behavior. Again, all files are to be submitted.
- Finally, you are given an encypted file `ciphertext3`, which you know has been encrypted using *AES* in ECB mode with 128 bit key (`aes-128-ecb` and `-nosalt`). You also know that the plaintext is a text spreadsheet of 15 lines. Each line is 32-bytes long. The first 16 bytes of the line is a name and the last 16 are a salary. (Look at this sample file file for a similarly formated file with fewer lines.)
- a) How many different salaries are there in the plaintext corresponding to `ciphertext3`, and which lines have the same salaries? b) assuming you are group *X*, by modifying `ciphertext3` exchange the salary entry of line *X* with that of line *X+1*. Provide the modified file (name it `ciphertext3.mod`). The decryption of the modified file should not result in errors or mis-formatted entries.

Part 4 (25%)

The sliding component is posted separately.

Deliverables

Only one of the group members need to submit on behalf of the entire group (in the event of more than one submission, the last one will be considered). Your report should include answers to the questions and should cite any resources that you used to answer the questions. By default it is assumed that all group members equally contribute to the assignment. If you need to deviate from this model of cooperation, explain why and indicate who was responsible for what. There is no restriction to the language you can use for programming as long as you can provide the instructions of how to compile and run your code in CS undergraduate lab machines (and specifically the *ucXX.cs.ualberta.ca* hosts). Your report (in plaintext, markdown, or pdf format) to address the

questions raised in this assignment should be submitted as a single file accompanying your results, source code, etc.

(The sliding part (and the sliding part only) will be submitted separately by the deadline of the (non-sliding) part of Assignment 2. It is *not* due on the deadline of Assignment 1.)

Friday, January 15, 2021