Project Report

Project Name: "Crypto Engine"

By

Group No. 6

Sadia Ahmed Bushra (201014005)

Md. Mehedi Hasan (201014082)

Farjahan Akter Boby (201014007)

Apon Gosh (201014069)

Razaan Reza (201014045)

Structured Programming Lab

(CSE104)

Section: 01

Fall 2020

Submitted To-

Tangila Islam Tanni

Lecturer

Depart of Computer Science & Engineering

School of Science & Engineering

University of Liberal Arts Bangladesh (ULAB)

Date of Submission: January 11, 2021

Introduction:

Our group members have decided to make a cryptography program named “Crypto Engine” using Caesar cipher and Mono-alphabetic cipher. Using these methods, one can get the opportunity to encrypt and decrypt any alphabetical message. Cryptography is the process of protecting any information by transforming it into a secure format that is not understandable for everyone. To pass an encrypted message from one person to another, both parties have to have the 'key' for the cipher, so that the sender may encrypt it and the receiver may decrypt it. Both Caesar and monoalphabetic cipher are types of substitution cipher. Caesar cipher is the simplest substitution cipher whereas mono-alphabetic cipher is the improved version of it.

Process:

At first, the user will be given two options to sign up and to log in. If the user can successfully log in with the information that he has saved in the text file while doing sign up, he will be allowed to proceed to the next step. If the user is not a member of the “CRYPTO ENGINE” then he will have to sign up first. Then he will be asked in which process he wants to continue. If the user wants to use the Caesar cipher then he would have to press ‘C’ and if the user wants to use the Mono-alphabetic process then he would have to press ‘M’. The user will also be given the options of encryption and decryption. To encrypt he would have to press ‘E’ and to decrypt he would have to press ‘D’. The user will also be asked if he wants to use the default key or enter his own personal key to encrypt or decrypt his message. After entering the key, the message will be encrypted or decrypted successfully.

In the sign-up, login form the user will be asked to sign up with a username, password, and some other pieces of information. The user’s information will automatically be saved in the text file the path of which is the username. Also, the user’s username and password will be saved in the different text files respectively. For each and every time while doing sign up username must have to be different than previous information. Only then sign up is possible. While doing login the user has to login with a username and a password that was already saved in the text file while doing sign up.

In the Caesar cipher, the ASCII value of each letter of the plain text is added with the key and thereby creates the encrypted version of the plain text. That means each letter of the plain text is right-shifted by the number of positions of the secret key. For decryption, letters of the plain text are left-shifted by the key. That means the ASCII value of each letter of the encrypted text is subtracted by the key. The key has to be between 0 to 25 since there are 26 letters in the English alphabet. If the ASCII value is not between 65 to 90 and 97 to 122, they are converted and fetched between the range.

In the case of monoalphabetic ciphers, instead of shifting the alphabets by numbers, permutations of the letters in the English alphabet are used to replace the letters of the plain text. The key here is the permutation, which is a mixed-up set of alphabets. By permutation, the user can create 26! keys. Here, both English alphabets and key are valued from 0 to 25. For encryption, the sender replaces each plaintext letter by replacing the permutation letter. For decryption, the process is just the opposite.

Algorithms:

Sign-up :

The loop will be counted for infinity times.Which means I will be allowed to take as much as character I want to take for password. In {ent=getch()} part user input will read the character which will be invisible. If a user inputs an Enter key from the user keyboard (ASCII code 13) then it will store null value and break the loop. If the user input backspace(ent==8) then the prompt will delete one character from the last one and will continue doing the same process. If user input none of the characters that is mentioned above then it will print all the character as (*).

The algorithm for Sign-up password:

```
while (1) {
        ent = getch ();
        if (ent == 13) {
            pass[i] = '\0';
            break;
        }

        else if (ent == 8) {
            i--;
            printf("\b \b");
            continue;
        }

        else {
            pass[i] = ent;
```

```
            i++;
            printf("*");
        }
    }
```

Login :

While doing login the loop will be counted for infinity times.Which means the user will be allowed to take as much as the character user wants to take for password. In {ent=getch()} part user input will read the character which will be invisible.If a user inputs an Enter key from the user keyboard (ASCII code 13) then it will store null value and break the loop. If the user input backspace(ent==8) then the prompt will delete one character from the last one and will continue doing the same process. If user input none of the characters that is mentioned above then it will print all the character as (*).

The algorithm for login:

```
while (1) {
        ent = getch();
        if (ent == 13) {
            check_pass[i] = '\0';
            break;
        }

         else if (ent == 8) {
            i--;
            printf("\b \b");
            continue;
        }

         else {
            check_pass[i] = ent;
            i++;
            printf("*");
        }
    }
```

Caesar Encryption: Firstly, it is checked if the input is an alphabetic character or not. Then simply, each character is added to the key. If the ASCII value of the input is not between 65 to 90 and 97 to 122, it is converted and fetched between the range. The user has to input a key if he chooses to do the customized key process.
Also, there is also an option for using the default key in the program which has been set to 7. To get the correct result, the input should be English alphabets and the key should be between 0 to 25.

The algorithm for encryption:

```
if (ch >= 'a' && ch <= 'z') {
            ch = ch + key;

            if (ch > 'z') {
               ch = ch - 'z' + 'a' - 1;
            }
            plain_text[i] = ch;
```

Caesar Decryption: Just as the encryption, it is checked whether the input is English alphabets or not. Then, each character is subtracted by the key. If the ASCII value of the input is not between 65 to 90 and 97 to 122, it is converted and fetched between the range.

The algorithm for decryption:

```
if (ch >= 'a' && ch <= 'z') {
            ch = ch - key;

            if(ch < 'a') {
               ch = ch + 'z' - 'a' + 1;
            }
            encrypted_text[i] = ch;
         }
```

Monoalphabetic Encryption: The key has to be the permutation of English alphabet. First, it compares the position of plain text's letters with the actual sequence of English alphabet. Then, replaces each plaintext letter by the permutation letter in the key. The user has to input a key if he chooses to do the customized key process.

Also, there is also an option for using the default key in the program. To get the correct result, the letters of the key should not be repeated. It must be a unique variation of the English alphabet.

Algorithm for monoalphabetic encryption:

```
if (pt[j] == p[i]) {
        c[i] = ct[j];
    }
```

Monoalphabetic Decryption: First, it compares the position of encrypted text's letters with the actual sequence of English alphabet. Then, replaces each encrypted text letter by the permutation letter in the key.

Algorithm for monoalphabetic encryption:

```
if (ct[j] == c[i]) {
        r[i] = pt[j];
    }
```

Tools:

Crypto Engine is used to encrypt and decrypt any alphabetical message. The program consists of various functions. There are a bunch of functions that have been defined and declared by us. Other predefined functions are used from the C library. The first few lines of the program hold the declaration of the header files of the libraries.

Header files:

- stdio.h
  - printf() - used to print the integers, strings, characters, etc on the screen.
  - scanf() - takes character, string, integer, etc as input· getc() - reads the characters, string and integer from the file
  - puts() - writes a string to stdout up to but not including the null character. A newline character is appended to the output
  - gets() - reads a line from stdin and stores it into the string pointed to

by str.

- fflush() - it's used for output stream only. Its purpose is to clear the output buffer and move the buffered data to console or disk.
- fprintf() - fprintf is used to print content in file instead of stdout console.
- fgets() - For reading a string value with spaces, we can use either gets() or fgets() in C programming language.
- fscanf() - reads formatted input from a stream.
- fclose() - used to close a file
- rewind() - moves the file position indicator to the beginning of the specified stream and also clears errors along the way

· conio.h
  · getch() - holds the output screen and waits until the user gives any type of input

· ctype.h
  · toupper() - checks whether a character is alphabetic and then converts it into uppercase

· string.h
  - strcpy() - copies the string pointed by source to the destination
  - strcat() - allows one memory block to be attached with another memory block
  · strlen() - calculates the length of a string
  · strcmp() - compares between two strings and returns an integer

User Defined Functions:

. login_signup() - for signup and login purpose

. Casear_Cryptography
  . Ceaser_Encryption_DK() - for caesar default key encryption
  . Caeser_Decryption_DK() - for caesar default key decryption
  . Caeser_Encryption_CK() - for caesar customize key encryption
  . Caeser_Decryption_CK() - for caesar customize key decryption

.Monoalpha_Cryptography()
  . Monoalpha_Encryption_DK() - for monoalphabetic default key encryption
  . Monoalpha_Encryption_CK() - for monoalphabetic customize key encryption
  . Monoalpha_Decryption_DK() - for monoalphabetic default key

decryption
. Monoalpha_Decryption_CK() - for monoalphabetic customize key decryption

Uses:

- Ensures the confidentiality of any data transmitted through any network
- Provides security to sensitive information
- Increases the security of conversation between client apps and servers
- Limited people will have access to classified information
- Less probability of any information being hacked

Project contributions:

Every member of the group contributed to the making of this project. Sadia Ahmed Bushra was in charge of writing the code for Caesar Cryptography's customize key portion, making the power-point presentation, and combining everyone's tasks. Mehedi Hasan Mubin wrote the code for Caesar Cryptography's default key and also debugged and combined the whole program. Apon Ghosh handled the coding for the Mono-alphabetic Cryptography both personal and default key and also debugged some errors. Farjahan Akter Boby added the switch cases and login and signup system to the program. And lastly, Razaan Reza helped with creating the project proposal and project report. Thus, with the collaboration of all the group members, the project was successfully completed.