Green University of Bangladesh

Department of Computer Science and Engineering (CSE)
Semester: (Spring, Year: 2024), B.Sc. in CSE (Day)

Message Encryption And Decryption Tool

Course Title: Data Communication Lab
Course Code: CSE 308
Section: 221 D20

Students Details

Name	ID
Tanvir Ahmed	221002461

Submission Date: 07/06/2024
Course Teacher's Name: Md. Romzan Alom

[For teachers use only: Don't write anything inside this box]

Lab Project Status	
Marks:	**Signature:**
Comments:	**Date:**

Contents

Chapter 1

Introduction

1.1 Overview

In the 21st century, the need for secure communication has become one of the most important things. With the huge amount of data transmitted over networks and stored in various systems, ensuring the security of the information is imperative. With this demand, my project proposes the development of a secret message encryption and decryption tool using c. We aim to provide an efficient solution for encrypting messages to secure them from unauthorized access and decryption to retrieve the information securely. By using cryptographic algorithm technics, our tool will offer a secure means of protecting the transmission and storage of information. With this project we aspire to continue to contribute to the development of data security and privacy, helping the user with a secure safeguard to protect their sensitive information in this interconnected world.

1.2 Problem Definition

In today's world of information, the security of data is paramount, most of the existing encryption tools are more challenging in terms of complexity and customization. Our project overcame this by developing a simple Secret Message Encryption and Decryption Tool using C. Despite the standard encryption, data breaches and unauthorized access happen more often nowadays. Highlighting the need for more accessible and reliable encryption solutions, our tool aims to provide users with a user-friendly simple interface to ensure the security and integrity of transmitted and stored information. By prioritizing flexibility, users can select the option to either encrypt plaintext messages or decrypt encrypted messages, empowering them to measure their specific needs. With this project, we aim to offer a simple and intuitive solution to enhance data security and reduce the risk of unauthorized access in the digital world.

1.3 Design Goals/Objectives

- Develop a lightweight and straightforward user interface: Create a simple and intuitive interface that enables users to input encryption keys and perform encryption and decryption operations with ease,
- Implement basic encryption and decryption techniques with user-provided keys using learning techniques that I learned from the Data Communication Lab Course.
- Provide flexibility for users to choose between encryption and decryption: Design the tool to offer users the option to either encrypt plaintext messages or decrypt encrypted messages, giving users control over the functionality based on their specific needs and preferences.
- Ensure ease of use: Prioritize user experience by designing the tool to guide users through the process of entering encryption keys and performing encryption and decryption tasks, even for individuals with minimal technical knowledge.
- Allow for future enhancements: Design the tool with flexibility in mind to allow for potential future updates or additions of features, while keeping the core functionality simple and user-friendly.

1.4 Application

Our Secret Message Encryption and Decryption Tool using C has a wide range of applications across various fields. Users can use the tool to encrypt personal messages, and sensitive information shared over email, messaging apps, and cloud storage platforms, enhancing their privacy and security in the digital world. In the world of business and finance, this tool can be utilized to secure sensitive communications, financial transactions, and confidential data exchange between shareholders. In the healthcare sector, it can safeguard patient records and medical information, ensuring compliance with data protection. Additionally, in government and military settings, this tool can ensure secure communication and information sharing among personnel while protecting classified data from unauthorized access. Overall, our encryption tool offers a simple and reliable solution for safeguarding data across a multitude of applications and industries.

Chapter 2

Design/Development/Implementation of the Project

2.1 Introduction

The Secret Message Encryption and Decryption Tool was designed, developed, and implemented to provide an efficient secure communication solution. This chapter describes the systematic approach used to conceptualize, develop, and implement the tool, emphasizing the key design principles, development processes, and technical details that resulted in a robust and functional encryption tool. By emphasizing simplicity and security, we hoped to create a useful application that addresses the growing need for data protection in a variety of fields.

2.2 Project Details

This project is all about encryption messages and after reviewing the message in the receiver site it also decrypts the message. In this project, I am taking advantage of ASCII code to encrypt and decrypt messages. I am taking the message from the user and also taking the key. This project converts every character of the message into the corresponding ASCII code and passes it to the equation with the key then the equation gives us an encrypted message. After receiving the message this project uses the decryption equation to get the original message.

2.3 Implementation

Source Code:

```
#include <stdio.h>
#include <string.h>
void addCharToBeginning(char str[], char ch) {
    int length = strlen(str);
```

```
    for (int i = length; i >= 0; i--) {
        str[i + 1] = str[i];
    }

    str[0] = ch;
}

void removeFirstChar(char str[]) {
    int length = strlen(str);
    for (int i = 0; i < length; i++) {
        str[i] = str[i + 1];
    }
}


int gcd(int a, int b) {
    while (b != 0) {
        int t = b;
        b = a % b;
        a = t;
    }
    return a;
}

int modInverse(int a, int m) {
    int m0 = m, t, q;
    int x0 = 0, x1 = 1;

    if (m == 1)
        return 0;

    while (a > 1) {
        q = a / m;
        t = m;
        m = a % m;
        a = t;
        t = x0;
        x0 = x1 - q * x0;
        x1 = t;
    }

    if (x1 < 0)
        x1 += m0;

    return x1;
}
```

```
int encode(int k, int x, int b, int m) {
    return (k * x + b) % m;
}

int decode(int k, int y, int b, int m) {
    int a_inv = modInverse(k, m);
    return (a_inv * (y - b + m)) % m;  // +m to handle negative y-b
}

int main()
{
   int i, x, key,kr;
   char str[100],rk;

   printf("\nPlease enter your message: ");
   gets(str);

   printf("\nPlease choose following options:\n");
   printf("1 = Encrypt the message\n");
   printf("2 = Decrypt the message.\n");
   scanf("%d", &x);

   switch(x)
   {
   case 1:
       printf("Please enter the key:  \n");
       scanf("%d", &key);

      for(i = 0; (i < 100 && str[i] != '\0'); i++){
        int as = str[i];
        str[i] = encode(key, as, 3, 116);
      }

        rk = (char)(key+3);
       addCharToBeginning(str, rk);
      printf("\nEncrypted message: %s\n", str);
      break;

   case 2:
      kr = (int)(str[0])-3;
      for(i = 0; (i < 100 && str[i] != '\0'); i++){
        int as = str[i];
        str[i] = decode(kr, as, 3, 116);
      }
      removeFirstChar(str);
      printf("\nDecrypted message: %s\n", str);
      break;
```

```
    default:
        printf("\nWrong option\n");
    }
    return 0;
}
```

2.4 Pseudocode

```
FUNCTION addCharToBeginning(string str, char ch):
    length = LENGTH of str
    FOR i from length DOWN TO 0:
        str[i + 1] = str[i]
    END FOR
    str[0] = ch
END FUNCTION

FUNCTION removeFirstChar(string str):
    length = LENGTH of str
    FOR i from 0 TO length - 1:
        str[i] = str[i + 1]
    END FOR
END FUNCTION

FUNCTION gcd(int a, int b):
    WHILE b is NOT 0:
        temp = b
        b = a MOD b
        a = temp
    END WHILE
    RETURN a
END FUNCTION

FUNCTION modInverse(int a, int m):
    m0 = m
    x0 = 0
    x1 = 1
    IF m == 1:
        RETURN 0
    WHILE a > 1:
        q = a DIV m
        temp = m
        m = a MOD m
        a = temp
        temp = x0
        x0 = x1 - q * x0
```

```
        x1 = temp
    END WHILE
    IF x1 < 0:
        x1 = x1 + m0
    RETURN x1
END FUNCTION

FUNCTION encode(int k, int x, int b, int m):
    RETURN (k * x + b) MOD m
END FUNCTION

FUNCTION decode(int k, int y, int b, int m):
    a_inv = modInverse(k, m)
    RETURN (a_inv * (y - b + m)) MOD m
END FUNCTION

MAIN:
    DECLARE str as STRING with size 100
    DECLARE x, key, kr as INTEGER
    DECLARE rk as CHAR

    PRINT "Please enter your message: "
    GET str FROM user

    PRINT "Please choose following options:"
    PRINT "1 = Encrypt the message"
    PRINT "2 = Decrypt the message"
    GET x FROM user

    IF x == 1:
        PRINT "Please enter the key: "
        GET key FROM user

        FOR i from 0 TO LENGTH of str:
            IF str[i] == END OF STRING:
                BREAK
            as = ASCII value of str[i]
            str[i] = encode(key, as, 3, 116)
        END FOR

        rk = CHAR value of (key + 3)
        addCharToBeginning(str, rk)
        PRINT "Encrypted message: ", str

    ELSE IF x == 2:
        kr = ASCII value of str[0] - 3
        FOR i from 0 TO LENGTH of str:
            IF str[i] == END OF STRING:
```

```
                BREAK
            as = ASCII value of str[i]
            str[i] = decode(kr, as, 3, 116)
        END FOR
        removeFirstChar(str)
        PRINT "Decrypted message: ", str

    ELSE:
        PRINT "Wrong option"
    END IF
END MAIN
```

Chapter 3

Performance Evaluation

3.1 Results Analysis/Testing

3.1.1 Result_portion_1

Figure 3.1: Encrypting the message.

3.1.2 Result_portion_2

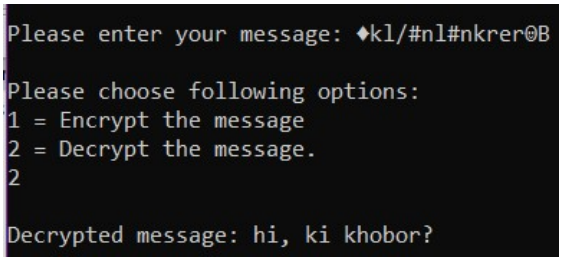

Figure 3.2: Decrypting the message

3.2 Results Overall Discussion

In the output section, we can see the project taking a message and taking what the user wants to encrypt the message or decrypt. After taking the want, it will give the encrypted data if the user wants to encrypt. In the second portion, we can see after giving the encrypted data we can decrypt back to the original data. Of course, it will happen in the receiver part. We can see, I complete the objectives here. The project encrypts the data properly and decrypts it again back to the original message.

Chapter 4

Conclusion

4.1 Discussion

The Secret Message Encryption and Decryption Tool successfully meets its primary objectives by providing a straightforward and effective means for secure communication. The tool's implementation of basic cryptographic techniques demonstrates a clear understanding of encryption and decryption processes using ASCII values and an equation. The simple interface allows individuals with minimal technical knowledge to easily encrypt and decrypt messages, enhancing data security across various applications.

4.2 Limitations

Despite its functionality, the Secret Message Encryption and Decryption Tool has several limitations that need to be addressed. Firstly, the tool uses a relatively simple encryption scheme that may not provide sufficient security against this AI era. Second, using a fixed key for encryption and decryption can be a security risk. Furthermore, the tool currently only supports messages with ASCII characters, which limits its use for encrypting other data types or character sets. The lack of advanced error handling and input validation could also cause problems if the user enters invalid data or keys. Finally, the implementation has not been optimized for performance, which may be an issue when encrypting or decrypting large amounts of data. These limitations highlight the need for further enhancements to the tool's security, flexibility, and efficiency.

4.3 Scope of Future Work

In this project, in the equation part, I take constant values for the variables b and m. It may not be the optimal approach. It can be better. I can pick the value from the random value picker. But that will give us a problem. If the number is random then how it will decrypt? I have a solution here. I can pass those values with the encrypted message, as I pass the key, Then in the receiver part I can extract those variables and can use them to decrypt.

References

- GeeksForGeeks website
- Foruzan, Data communication and Networking
- Blanchard, Introduction to Networking and Data Communications,