T

|  |  |
| --- | --- |
| A picture of a winding road and trees  Encryption Assignment  Documentation | Abstract  This document explains the classes and functions used in each method declaration of our group’s encryption program. It gives a demonstration of the application encrypting and decrypting a text file (it can also encrypt and decrypt a JPEG image, PNG image and a RAR file). Finally, this document compares our chosen encryption algorithm being the Feistel cipher with other encryption algorithms.  Marco van der Merwe - 35314389, Markus Marais - 34906258, Denzil Richter -28885198, Arthur de Villiers - 35359099, Jacques Nel - 31986595  CMPG215 |

# Table of content:

Contents

[Table of content: 1](#_Toc72785114)

[Organisation and explanation of the encryption program created in C#: 1](#_Toc72785115)

[Using directives: 1](#_Toc72785116)

[Method declarations: 2](#_Toc72785117)

[Explanation of how to use the application: 9](#_Toc72785118)

[Our chosen encryption algorithm compared to other encryption algorithms: 13](#_Toc72785119)

[References: 20](#_Toc72785120)

# Organisation and explanation of the encryption program created in C#:

C# code is organised as methods which are contained inside classes, which are contained inside namespaces. At the very top of the code, the using directives are named which indicate which namespaces of .NET Framework the program will use.

## Using directives:

 “using System.Security.Cryptography;”

Provides cryptographic services, including secure encoding and decoding of data, as well as many other operations, such as hashing, random number generation, and message authentication. (dotnet-bot (n.d.), 2021)

“using System.IO;”

The System.IO namespace consists of IO related classes, structures, delegates and enumerations. These classes can be used to read and write data to files or data streams. It also contains classes for file and directory support. (www.javatpoint.com, 2011)

## Method declarations:

Text

Description automatically generated

Figure 1

The method “EncryptionForm\_Load” on line 26 sets the default forms state to encrypt a file first.

Text

Description automatically generated

Figure 2

In line 35 an instance of the “HashAlgorithm” class is created named algorithm.

In line 37 the SHA256 hash of the byte values of the given password is returned.

Text

Description automatically generated

Figure 3

On line 43, StringBuilder class is used to create a new variable named “sb” that can hold any text or a sequential collection of characters.

On line 44, “GetHash” created in line 32 is used to create a hashed version of the given string value (password). The hashed password is then looped through checking each byte and appending the Hexadecimal value of that byte to our StringBuilder “sb” variable on line 45.

The StringBuilder variable is then returned in string format (line 47).

Text

Description automatically generated

Figure 4

On line 50, the method “FeistelEncrypt” takes 3 parameters, int pLeft, int pRight, and int key which will be the password given by the application user. On line 52, int L1 is set to the pLeft parameter, and on line 53, int R1 is set to the pRight parameter. Encryption is then done in 2 rounds. In round 1 starting on line 56, int R2 is set to (key + 1) XORed with int L1. Int L2 on line 57 is set to int R1.

In round 2 starting on line 60, int R3 is set to (key + 2) XORed with int L2 from the previous round. Int L3 on line 61 is set to int R2.

In the final round on line 63 the two integers L3 and R3 are swapped around. R4 = L3, and L4 = R3.

On line 67 an integer array “returnArray” is created with R4 and L4. This array is then returned on line 68.

Text

Description automatically generated

Figure 5

To decrypt the encrypted file, we need to put the decrypted file through the same process but swap the keys around. Thus (key + 1) in round 1, and (key + 2) in round 2 from the encrypt method is swapped around in the decrypt method. In the decrypt method (key + 1) will be used in round 2, and (key + 2) will be used in round 1.

On line 71, the method “FeistelDecrypt” takes 3 parameters, int pLeft, int pRight, and int key which should be the password given by the application user when encrypting a file. On line 73, int L1 is set to the pLeft parameter, and on line 74, int R1 is set to the pRight parameter. Encryption is then done in 2 rounds. In round 1 starting on line 77, int R2 is set to (key + 2) XORed with int L1. Int L2 on line 78 is set to int R1.

In round 2 starting on line 81, int R3 is set to (key + 1) XORed with int L2 from the previous round. Int L3 on line 82 is set to int R2.

In the final round starting on line 85, the two integers L3 and R3 are swapped around. R4 = L3, and L4 = R3.

On line 88 an integer array “returnArray” is created with R4 and L4. This array is then returned on line 89.

Text

Description automatically generated

Figure 6

When the button “Browse” is clicked while on the “Encrypt” panel, the user will be prompted to select a file that should be encrypted. Otherwise, when the button “Browse” is clicked while on the “Decrypt” panel, the user will be prompted to select a file that should be decrypted.

Text

Description automatically generated

Figure 7

Declaration of local variables needed for the encryption starts from line 116 and ends at 122.

The supplied password is converted into a string using the method “GetHashString”.

Text

Description automatically generated

Figure 8

The password entered is received, “hashed”, and then stored in a byte array.

On line 126, an integer variable length is created and set equal to the length of the password.

The for loop on line 127 loops through the password and for each character in the byte array it gets added and is set equal to the “key” variable. It then gets subtracted by the last character of the byte array. This is done for the length of the byte array.

The variable “key” is divided by the length otherwise the maximum of bytes is reached, and an error occurs.

Text

Description automatically generated

Figure 9

First the program checks to see whether the length of the file is uneven. If it is uneven, the last value of the array is manually encrypted by increasing the ASCII value by 1.

In the for-loop, lines 147 to 153 are only executed on every 2nd iteration. The reason being that the program needs to assign a left and a right value to be used for the “FeistelEncrypt” method. On line 152 and 153 the corresponding index values in the array “convertedFile” are rewritten with the encrypted values.

Text

Description automatically generated

Figure 10

After the “convertedFile” array contains the encrypted values, a new instance of the “Stream” class is declared and is written to the file provided by the file Path variable.

Text

Description automatically generated

Figure 11

The Decrypt button works very much the same as the Encrypt button.

On line 175 a save file dialog box is populated so that the application user may choose which file type the decrypted file should be saved as.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 12

On line 204 if the length of the file is uneven, the last value of the array is manually encrypted by decreasing the ASCII value by 1.

In the for-loop, lines 211 to 217 are only executed on every 2nd iteration. The reason being that the program needs to assign a left and a right value to be used for the “FeistelDecrypt” method. On line 216 and 217 the corresponding index values in the array “convertedFile” are rewritten with the decrypted values.

Text

Description automatically generated

Figure 13

The “EncryptTabButton\_Click” method resets all critical values to default.

Text

Description automatically generated

Figure 14

The “DecryptTabButton\_Click” method resets all critical values to default.

# Explanation of how to use the application:

An executable file for the encryption program can be found in the following directory:

CMPG215\_Encryption\_Assignment/bin/Debug (The executable file is called CMPG215\_Encryption\_Assignment.exe).

A screenshot of a computer

Description automatically generated

Figure 15

1. The Encrypt button is selected by default when the application is opened.
2. The user can then select the decrypt button to switch to the decrypt panel and decrypt an encrypted file.
3. When the user clicks on the browse button, they will be able to select a file on their device to be encrypted.
4. The security key is a password given by the application user to encrypt the file. This key (password) must be used to decrypt the same file once it has been encrypted.
5. The encrypt button will call the “FeistelEncrypt” method to encrypt the selected file. As shown in the following images:

A screenshot of a computer

Description automatically generated

Figure 16

A screenshot of a computer

Description automatically generated

Figure 17

The Decrypt button will prompt the application user to select a file location to store the decrypted file. You can then choose any file location or choose to save the decrypted file over the encrypted file.

The reason the program creates a separate decrypted file is to ensure that the original file is not lost should the user enter an incorrect password.

A screenshot of a computer

Description automatically generated

Figure 18

Graphical user interface, text, application, Teams

Description automatically generated

Figure 19

# Our chosen encryption algorithm compared to other encryption algorithms:

Feistel Cipher:

The Feistel network also known as a Feistel cipher, is a cryptographic technique designed roughly around the 70s by Horst Feistel, a German American cryptographer who worked for IBM designing ciphers which led to the development of the Data Encryption Standard (DES).

A Feistel network encrypts data by splitting a data block into two equal pieces (left hand side and right-hand side) and applies encryption in multiple rounds. Each round applies permutation and combinations derived from the primary function or key such as a hash. After the right-hand side has been put through this function using a specific key, we XOR the block with the left-hand sided data block to produce the new right-hand sided block used in round two. The left-hand sided block used in round 2 will be the same data block as the original right hand sided block before it was put through a function and XORed. This method of encryption will be repeated for each round, and the number of rounds can be repeated as many times as you would like. At the very end after the last round, you flip the output and that will give you the encrypted file.

To decrypt your now encrypted file using the same Feistel cipher, you put the encrypted data block right back into the starting block and you run it again going through the same rounds it went through to encrypt it. The only difference is that you will swap the keys used in each round. For example, if you had two rounds such as in our encryption program, you would use key one in the first round, and key two in the second round. To decrypt the file, you will take this encrypted data block and run it through the same procedure only you will use key two in round one, and key one in round two.

Diagram

Description automatically generated

Figure 20 (Wikipedia Contributors, 2019)

Diagram, schematic

Description automatically generated

Figure 21 (www.tutorialspoint.com. (n.d.), 2021)

Data Encryption Standard (DES):

DES is an implementation of a Feistel Cipher that specifically uses 16 rounds to encrypt a data block. The block size is 64 bits, and the key length is 56 bits since 8 of the 64 bits are not used by the encryption algorithm. (Tutorialspoint.com, 2019) The general structure of DES is depicted in the following illustration:

Diagram, schematic

Description automatically generated

Figure 22 (Tutorialspoint.com, 2019)

The national security agency (NSA) selected a slightly modified version of DES as the Federal Information Processing Standard (FIPS) for the United States in 1977. DES was, however, declared insecure due to its relatively short 56-bit key size. The DES key was broken in 22 hours and 15 minutes by distributed.net and Electronic Frontier Foundation in January 1999. The cipher has been superseded by the Advanced Encryption Standard (AES), and DES has been withdrawn as a standard by the National Institute of Standards and Technology.

Advantages and disadvantages of the Feistel Cipher:

Advantages:

1. It is convenient for software and hardware requirements as encryption and decryption uses the same algorithm, only that in the case of DES the function needs to be reversed and the keys should be taken in opposite order.
2. A slight change in the plaintext will drastically change the cipher text.
3. Each bit of ciphertext is based upon multiple bits of the key.
4. This algorithm is relatively easy to implement.

Disadvantages:

1. If the rounds implemented uses the same keys the encryption algorithm can be used to decrypt the file and the decryption algorithms will be redundant code.
2. There can be same output from the S-Bokes used in DES on different inputs on permutation.
3. If encryption is done using a specific key, the decryption must be done using the same key otherwise the file will be further encrypted. Decrypting this file will be difficult to keep track of.

(Bajaj, P. (n.d.), 2019)

Advanced Encryption Standard (AES):

AES being a subset of the Rijndael block cipher with a block size of 128 bits and systematic keys that can be three different key lengths (128, 192, or 256 bits) is a specification for the encryption of electronic data established by the U.S National Institute of Standards and Technology (NIST) in 2001. **AES** is the **encryption** standard that is recognized and recommended by the U.S. government. In the AES algorithm, the same key is used for both encrypting and decrypting data.

Diagram, schematic

Description automatically generated

Figure 23 (tutorialspoint.com, 2019)

Advantages and disadvantages of the Advanced Encryption Standard (AES):

Advantages:

1. Because of it being implemented in both hardware and software, it is the most robust security protocol.
2. It uses higher key length sizes such as 128, 192 and 256 bits for encryption.
3. It is one of the most spread commercial and open-source solutions used all over the world.

Disadvantages:

1. It uses too simple algebraic structure.
2. Every data block is always encrypted in the same way.
3. It is hard to implement with software.

(Rfwireless-world.com, 2012)

Vigenère cipher:

This cipher implements encryption by shifting each letter along some number of places. It used a table of alphabets which is termed a tabula recta. It has the alphabet written out 26 times in different rows, each alphabet shifted cyclically to the left compared to the previous alphabet, corresponding to the 26 possible Caesar ciphers. At different points in the encryption process, the cipher uses a different alphabet from one of the rows. The alphabet used at each point depends on a repeating keyword. (Wikipedia Contributors, 2019)

A picture containing text, newspaper

Description automatically generated

Figure 24 (Wikipedia Contributors, 2019)

Advantages and disadvantages of the Vigenère cipher:

Advantages:

1. This algorithm is not susceptible to frequency analysis since the cipher rotates through different shifts.
2. It is difficult to crack using brute-force because each letter in a message could be encoded as any of the 2626 letters.

Disadvantages:

1. The primary weakness of the Vigenère cipher is the repeating nature of its key. If a cryptanalyst correctly guesses the length of the key, then the ciphertext can be treated as interwoven Caesar ciphers, which, individually, can be easily broken.
2. Repetitions in the ciphertext indicate repetitions in the plaintext, and the space between such repetitions hint at the length of the keyword.

(brilliant.org. (n.d.), 2021)

# References:

brilliant.org. (n.d.). 2021. *Vigenère Cipher | Brilliant Math & Science Wiki*. <https://brilliant.org/wiki/vigenere-cipher/#:~:text=The%20strength%20of%20the%20Vigen%C3%A8re> Date of access: 24 May. 2021.

Bajaj, P. (n.d.). 2019. *More about Data Encryption Standard (DES) and Advanced Encryption Standard (AES)*. <https://www.cryptographynotes.com/2019/07/encryption-standard-des-and-aes-differences-advantages-disadvantages.html>. Date of access: 24 May. 2021.

Rfwireless-world.com. 2012. *Advantages of AES | disadvantages of AES*. <https://www.rfwireless-world.com/Terminology/Advantages-and-disadvantages-of-AES.html>. Date of access: 24 May. 2021.

Wikipedia Contributors. 2019. *Vigenère cipher*. <https://en.wikipedia.org/wiki/Vigen%C3%A8re_cipher>. Date of access: 24 May. 2021.

Techopedia.com. (n.d.). *What is a Feistel Network? - Definition from Techopedia*. <https://www.techopedia.com/definition/27121/feistel-network#:~:text=A%20Feistel%20network%20is%20a> Date of access: 22 May. 2021.

www.tutorialspoint.com. (n.d.). 2021. *Feistel Block Cipher - Tutorialspoint*. <https://www.tutorialspoint.com/cryptography/feistel_block_cipher.htm>. Date of access: 22 May. 2021.

Wikipedia Contributors. 2019. *Feistel cipher*. <https://en.wikipedia.org/wiki/Feistel_cipher>. Date of access: 22 May. 2021.

tutorialspoint.com. 2019. *Advanced Encryption Standard*. <https://www.tutorialspoint.com/cryptography/advanced_encryption_standard.htm>. Date of access: 24 May. 2021.

‌Tutorialspoint.com. 2019. *Data Encryption Standard - Tutorialspoint*. <https://www.tutorialspoint.com/cryptography/data_encryption_standard.htm>. Date of access: 22 May. 2021.

dotnet-bot (n.d.). 2021. *System Security Cryptography Namespace*. <https://docs.microsoft.com/en-us/dotnet/api/system.security.cryptography?view=net-5.0> Date of access: 24 May. 2021.

www.javatpoint.com. 2011. *C# System.IO Namespace - javatpoint*. <https://www.javatpoint.com/c-sharp-system-io#:~:text=System.IO%20Namespace%20Enumerations&text=It%20is%20used%20to%20define%20constants%20for%20read%2C%20write%20or> Date of access: 24 May. 2021.

‌

‌