Документация

Курсова работа

Криптографски методи за защита на информацията в бази данни

Окан Сеид 121219017

3 курс 32 ИТ КСИ ФКСТ

при ас. П. Стойнов

Криптографски алгоритъм **DES**

Съдържание:

Описание

DES (Data Encryption Standard) е симетричен алгоритъм за криптиране, в който един и същи ключ се използва както за шифриране, така и за дешифриране на текста. Стандартно алгоритъмът работи с 64-битови ключове – 56, от които се избират случайно. Останалите 8 бита са за четност – по един на всеки 7-битов блок.

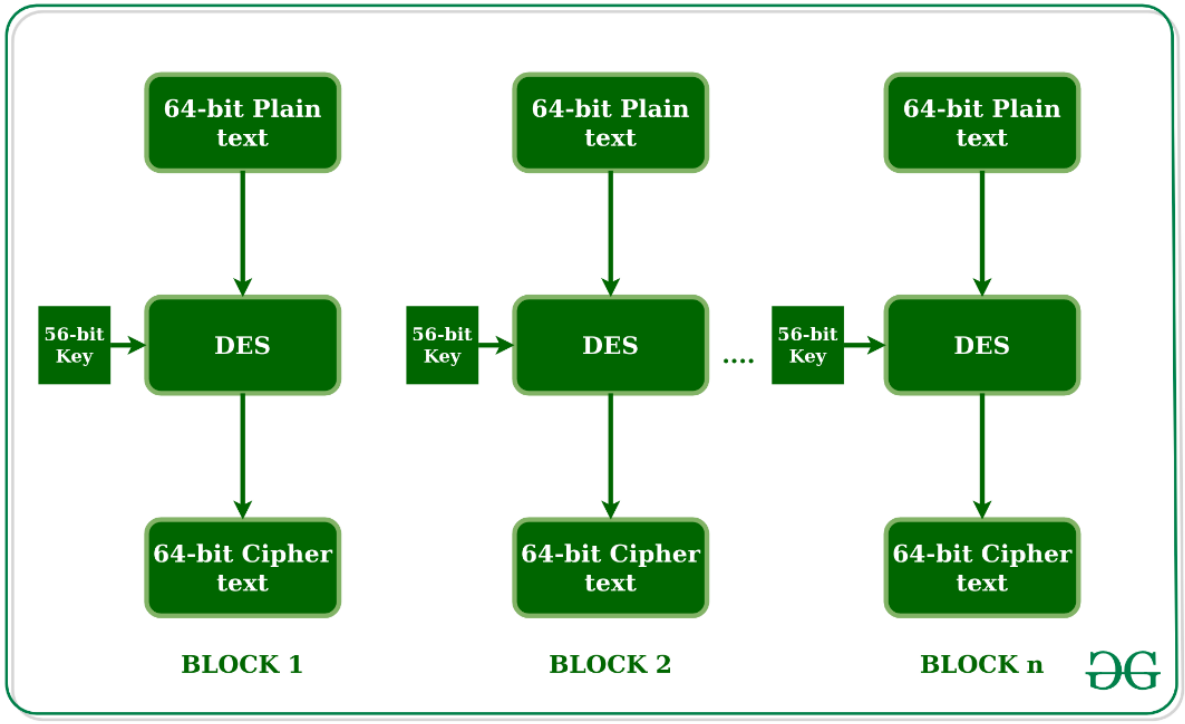
Цел

DES е остарял алгоритъм, възникнал в началото на 70-те години в САЩ от разработчиците на IBM. Скоро след това той бива одобрен от правителството като федерален стандарт за обработка и съхранение на данни. Няколко години по-късно алгоритъмът е оповестен за обществено

ползване в редица индустрии – банковата, финансовата, комуникационната индустрия и др. Алгоритъмът обаче от своя страна не е от най-сигурните, защото се преодолява лесно с метода на грубата атака и поради тази причина по-късно възникват негови наследници, които го заменят.

Имплементация

Реализирал съм конзолно приложение на примерна имплементация на алгоритъма за шифриране и дешифриране на програмния език C# с Framework .NET 5.0 под средата за разработка на Microsoft – Visual Studio 2019. То съставлява 3 функции – една главна, която се изпълнява, и 2 – една за шифриране и друга за дешифриране.



Програмен код

```
using System;
using System.IO;
using System.Security.Cryptography;
using System.Text;

namespace KR_Cryptography
{
    public class Program
    {
        static string Encryption(string plainText)
```

```
        {
            try
            {
                string encryptedText = "";
                string publicKey = "12345678";
                string secretKey = "87654321";
                byte[] publicKeyByte = Array.Empty<byte>();
                publicKeyByte = Encoding.UTF8.GetBytes(publicKey);
                byte[] secretKeyByte = Array.Empty<byte>();
                secretKeyByte = Encoding.UTF8.GetBytes(secretKey);
                MemoryStream ms = null;
                CryptoStream cs = null;
                byte[] inputByteArray = Encoding.UTF8.GetBytes(plainText);
                using (DESCryptoServiceProvider des = new())
                {
                    ms = new MemoryStream();
                    cs = new CryptoStream(ms, des.CreateEncryptor(publicKeyByte,
secretKeyByte), CryptoStreamMode.Write);
                    cs.Write(inputByteArray, 0, inputByteArray.Length);
                    cs.FlushFinalBlock();
                    encryptedText = Convert.ToBase64String(ms.ToArray());
                }
                return encryptedText;
            }
            catch (Exception ex)
            {
                throw new Exception("Error!", ex.InnerException);
            }
        }

         static string Decryption(string cipherText)
        {
            try
            {
                string plainText = "";
                string publicKey = "12345678";
                string secretKey = "87654321";
                byte[] publicKeyByte = Array.Empty<byte>();
                publicKeyByte = Encoding.UTF8.GetBytes(publicKey);
                byte[] secretKeyByte = Array.Empty<byte>();
                secretKeyByte = Encoding.UTF8.GetBytes(secretKey);
                MemoryStream ms = null;
                CryptoStream cs = null;
                byte[] inputByteArray = new byte[cipherText.Replace(" ",
"+").Length];
                inputByteArray = Convert.FromBase64String(cipherText.Replace(" ",
"+"));
                using (DESCryptoServiceProvider des = new())
                {
                    ms = new MemoryStream();
                    cs = new CryptoStream(ms, des.CreateDecryptor(publicKeyByte,
secretKeyByte), CryptoStreamMode.Write);
                    cs.Write(inputByteArray, 0, inputByteArray.Length);
                    cs.FlushFinalBlock();
                    Encoding encoding = Encoding.UTF8;
                    plainText = encoding.GetString(ms.ToArray());
                }
                return plainText;
            }
```

```
            catch (Exception ex)
            {
                throw new Exception("Error!", ex.InnerException);
            }
        }
        public static void Main(string[] args)
        {
                    string plainText = "";

                    while(true) {
                    Console.WriteLine("Make a choice:");
                    Console.WriteLine("1 - Type a plain text to encrypt");
                    Console.WriteLine("2 - Decrypt the encrypted text");
                    Console.WriteLine("3 - Exit");
                    Console.WriteLine();
                    int choice = Convert.ToInt32(Console.ReadLine());
                    Console.WriteLine();

                            switch(choice)
                            {
                                    case 1:
                                            Console.WriteLine("Type a plain text
here");
                                            plainText = Console.ReadLine();
                                            Console.WriteLine();
                                            Console.WriteLine(plainText + " -> " +
Encryption(plainText));
                                            Console.WriteLine();
                                            break;
                                    case 2:
                                            Console.WriteLine("Decrypting the cipher
text...");
                                            Console.WriteLine();
                                            Console.WriteLine(Encryption(plainText) + "
-> " + Decryption(Encryption(plainText)));
                                            Console.WriteLine();
                                            break;

                                    case 3:
                                            return;
                                    default:
                                            Console.WriteLine("Error, wrong number!");
                                            Console.WriteLine();
                                            break;
                            }
                    }
        }
    }
}
```

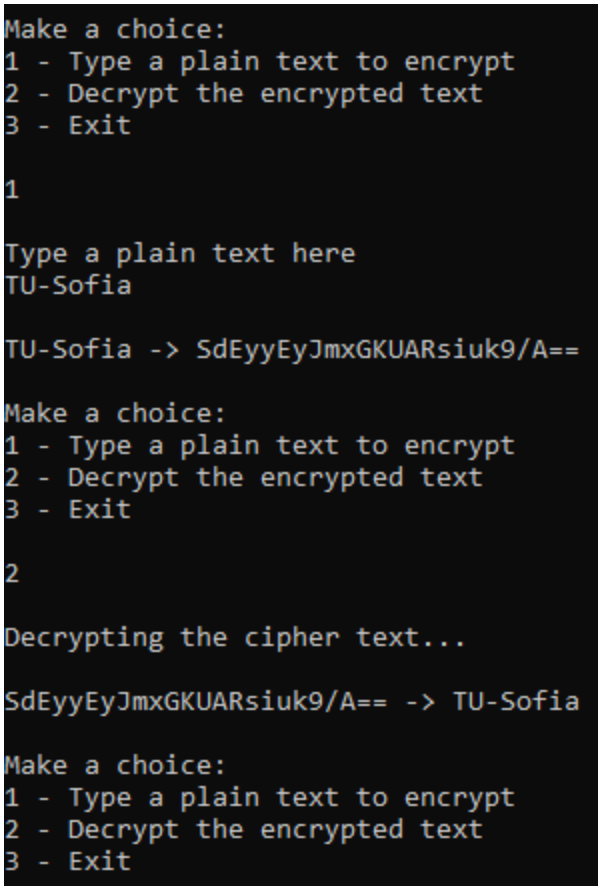
Резултати

```
Type a plain text here
Okan

Okan -> jcSbEFuQzxg=

Make a choice:
1 - Type a plain text to encrypt
2 - Decrypt the encrypted text
3 - Exit

2

Decrypting the cipher text...

jcSbEFuQzxg= -> Okan
```

Okan

```
Make a choice:
1 - Type a plain text to encrypt
2 - Decrypt the encrypted text
3 - Exit

1

Type a plain text here
Cryptography

Cryptography -> b+A9wRyVBVLNj5Fcf9mQtA==

Make a choice:
1 - Type a plain text to encrypt
2 - Decrypt the encrypted text
3 - Exit

2

Decrypting the cipher text...

b+A9wRyVBVLNj5Fcf9mQtA== -> Cryptography

Make a choice:
1 - Type a plain text to encrypt
2 - Decrypt the encrypted text
3 - Exit
```

Cryptography



TU-Sofia

Използвана литература

Източници:

www.geeksforgeeks.org

www.codeproject.com

www.bg.education-wiki.com

www.c-sharpcorner.com