Учреждение образования

«Белорусский государственный технологический университет»

**Кафедра информационных систем и технологий**

**«Отчёт по лабораторной работе 7»**

“ИССЛЕДОВАНИЕ БЛОЧНЫХ ШИФРОВ”

**Выполнила:** студентка 3 курса

4 группы специальности ПОИТ

Сапегина Екатерина Игоревна,

Вариант 11

Минск 2023

***Цель:*** изучение и приобретение практических навыков разработки и использования приложений для реализации блочных шифров (рассчитана на 4 часа аудиторных занятий).

***Задачи:***

1. Закрепить теоретические знания по алгебраическому описанию, алгоритмам реализации операций зашифрования/расшифрования и оценке криптостойкости блочных шифров.

2. Разработать приложение для реализации указанных преподавателем методов блочного зашифрования/расшифрования.

3. Выполнить анализ криптостойкости блочных шифров.

4. Оценить скорость зашифрования/расшифрования реализованных шифров.

5. Результаты выполнения лабораторной работы оформить в виде

описания разработанного приложения, методики выполнения экспериментов с использованием приложения и результатов эксперимента

***Практическая часть:***

1. Разработать авторское приложение в соответствии с целью

лабораторной работы. При этом можно воспользоваться готовыми

библиотеками либо программными кодами, реализующими некоторые блочные алгоритмы, из приложения в [5].

Приложение должно реализовывать следующие операции:

• разделение входного потока данных на блоки требуемой

длины с необходимым дополнением последнего блока;

• выполнение требуемых преобразований ключевой информации;

• выполнение операций зашифрования/расшифрования;

• оценка скорости выполнения операций зашифрования/расшифрования;

• пошаговый анализ лавинного эффекта с подсчетом количества изменяющихся символов по отношению к исходному слову.

Исследуемый метод шифрования и ключевая информация – в соответствии с вариантом.

|  |  |  |
| --- | --- | --- |
| Вариант | Алгоритм | Ключ |
| 1 | DES | Первые 8 символов собственных фамилии имени |

По желанию студент может разработать приложение и выполнить связанные исследования для любого другого блочного алгоритма, не указанного в табл. 5.5.

***Код реализации:***

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Security.Cryptography;  using System.IO;  using System.Diagnostics;  namespace Lab7  {  internal class Program  {  private static void Main()  {  while (true)  {  Stopwatch time = new Stopwatch();    Console.WriteLine("Enter text:");  var str = Console.ReadLine();  if (int.TryParse(str, out int res))  {  if (res == 0) break;  }  time.Start();  var crypted = Des.Crypt(str);  time.Stop();  Console.WriteLine($"Сrypted: {crypted} | {(float)time.ElapsedMilliseconds / 1000} sec");  time.Reset();  time.Start();  var decrypted = Des.Decrypt(crypted);  time.Stop();  Console.WriteLine($"Decrypted: {decrypted} | {(float)time.ElapsedMilliseconds / 1000} sec");  Console.ReadKey();  }  }  }  public class Des  {  public static string Crypt(string text)  {  var key = new byte[] { 0xea, 0xee, 0xe2, 0xe0, 0xeb, 0xe5, 0xe2, 0xe0 };  var iv = new byte[] { 0xa1, 0xa2, 0xa3, 0xa4, 0xa5, 0xa6, 0xa7, 0xa8 };  var bytes = Encoding.Default.GetBytes(text);  var data = Encrypt(bytes, key, iv);  return Encoding.Default.GetString(data);  }  public static string Decrypt(string text)  {  var key = new byte[] { 0xea, 0xee, 0xe2, 0xe0, 0xeb, 0xe5, 0xe2, 0xe0 };  var iv = new byte[] { 0xa1, 0xa2, 0xa3, 0xa4, 0xa5, 0xa6, 0xa7, 0xa8 };  var bytes = Encoding.Default.GetBytes(text);  var data = Decrypt(bytes, key, iv);  return Encoding.Default.GetString(data);  }  public static byte[] Encrypt(byte[] inputBytes, byte[] key, byte[] iv)  {  using (var des = new DESCryptoServiceProvider())  {  des.Mode = CipherMode.CFB;  des.Padding = PaddingMode.Zeros;  var encryptor = des.CreateEncryptor(key, iv);  var stream = new MemoryStream();  using (var cryptoStream = new CryptoStream(stream, encryptor, CryptoStreamMode.Write))  {  cryptoStream.Write(inputBytes, 0, inputBytes.Length);  }  return stream.ToArray().Take(inputBytes.Length).ToArray();  }  }  public static byte[] Decrypt(byte[] inputBytes, byte[] key, byte[] iv)  {  using (var des = new DESCryptoServiceProvider())  {  des.Mode = CipherMode.CFB;  des.Padding = PaddingMode.Zeros;  var decryptor = des.CreateDecryptor(key, iv);  var input = new List<byte>(inputBytes);  if (inputBytes.Length % 8 != 0)  {  input.AddRange(new byte[8 - inputBytes.Length % 8]);  }  using (var result = new MemoryStream())  {  using (var cryptoStream = new CryptoStream(new MemoryStream(input.ToArray()), decryptor,  CryptoStreamMode.Read))  {  cryptoStream.CopyTo(result);  }  return result.ToArray().Take(inputBytes.Length).ToArray();  }  }  }  }  } |

Это программа на языке C#, которая реализует шифрование и дешифрование текста с использованием алгоритма DES (Data Encryption Standard) с режимом шифрования CFB (Cipher Feedback mode).

Переменная "key" и "iv" представляют собой ключ и вектор инициализации соответственно, используемые при шифровании и дешифровании. В данном случае они являются захардкоженными в программу.

Главная функция "Main" бесконечно запрашивает ввод текста от пользователя, затем шифрует его с помощью функции "Crypt" и выводит результат на экран вместе с временем, затраченным на шифрование. Затем происходит обратный процесс - текст дешифруется с помощью функции "Decrypt" и выводится на экран вместе с временем, затраченным на дешифрование.

Функция "Encrypt" принимает входные байтовые данные и использует объект "DESCryptoServiceProvider" для создания шифратора с заданным ключом и вектором инициализации. Затем она записывает зашифрованные данные в поток "stream" и возвращает только те байты, которые соответствуют исходным данным.

Функция "Decrypt" принимает зашифрованные байты и использует объект "DESCryptoServiceProvider" для создания дешифратора с заданным ключом и вектором инициализации. Затем она дополняет зашифрованные данные, если они не кратны 8 байтам, и расшифровывает их. Наконец, функция возвращает только те байты, которые соответствуют исходным данным.

Обе функции "Crypt" и "Decrypt" используют кодировку по умолчанию для преобразования текста в байты и обратно. В данном случае используется "Encoding.Default", который зависит от системной локали и может отличаться в разных странах.

***Результат:***

