МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Отчет по лабораторной работе №5

Исследование блочных шифров

Выполнил студент: Плюто Э. В.

ФИТ 3 курса, 5 группа

Проверил: Савельева М. Г.

Минск 2024

**Практическое задание:**

**1)** Разработать авторское приложение, которое 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.

Алгоритм: *DES*, ключ: первые 8 символов собственных *фамилииимени*.

Реализация разделения потока данных на блоки представлена ниже.

|  |
| --- |
| public static List<string> MakeBlocks(string text, int blockSize)  {  List<string> blocks = new List<string>();  for (int i = 0; i < text.Length; i += blockSize)  {  blockSize = Math.Min(blockSize, text.Length - i);  string block = text.Substring(i, blockSize);  block = block.PadRight(8, ' ');  blocks.Add(block);  }  return blocks;  } |

Листинг 1 – Реализация разделения потока данных на блоки

Реализация функции для шифрования информации представлена на листинге 2.

|  |
| --- |
| public static byte[] EncryptTextToMemory(string text, byte[] key, byte[] iv)  {  using (MemoryStream mStream = new MemoryStream())  {  using (DES des = DES.Create())  using (ICryptoTransform encryptor = des.CreateEncryptor(key, iv))  using (var cStream = new CryptoStream(mStream, encryptor, CryptoStreamMode.Write))  {  byte[] toEncrypt = Encoding.ASCII.GetBytes(text);  cStream.Write(toEncrypt, 0, toEncrypt.Length);  }  byte[] ret = mStream.ToArray();  return ret;  }  } |

Листинг 2 – Реализация шифрования

Реализация функции для расшифрования информации представлена на листинге 3.

|  |
| --- |
| public static string DecryptTextFromMemory(byte[] encrypted, byte[] key, byte[] iv)  {  byte[] decrypted = new byte[encrypted.Length];  int offset = 0;  using (MemoryStream mStream = new MemoryStream(encrypted))  {  using (DES des = DES.Create())  using (ICryptoTransform decryptor = des.CreateDecryptor(key, iv))  using (var cStream = new CryptoStream(mStream, decryptor, CryptoStreamMode.Read))  {  int read = 1;  while (read > 0)  {  read = cStream.Read(decrypted, offset, decrypted.Length - offset);  offset += read;  }  }  }  return Encoding.ASCII.GetString(decrypted, 0, offset);  } |

Листинг 3 – Реализация расшифрования

Результат выполнения вышеописанных функций представлен на рисунке 1.

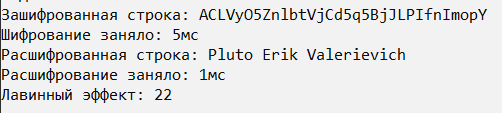


Рисунок 1 – Результат выполнения программы

Лавинный эффект, при добавлении одного символа к исходному сообщению представлен на рисунке 2.

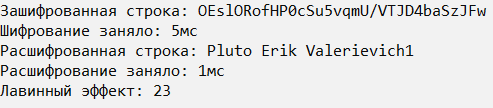


Рисунок 2 – Лавинный эффект с увеличенным сообщением

График времени шифрования и расшифрования сообщения представлен ниже.

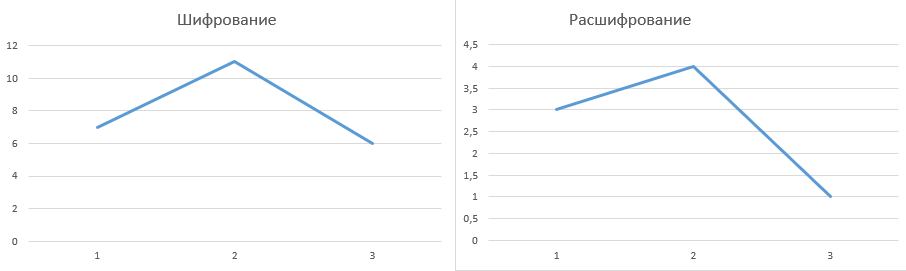


Рисунок 3 – График времени выполнения шифрования и расшифрования

**2)** Проанализировать влияние слабых ключей и полуслабых ключей на конечный результат зашифрования и на лавинный эффект.

Результат выполнения со слабым и полуслабым ключом представлен на рисунке 4.

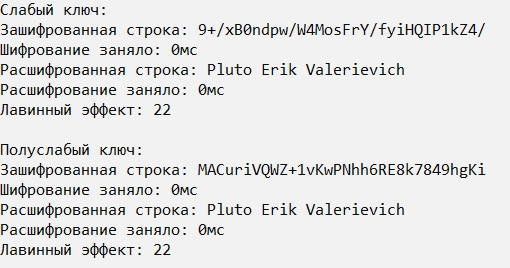


Рисунок 4 – Результат выполнения со слабым и полуслабым ключом

Лавинный эффект остался неизменным, результат шифрования слабого ключа можно подобрать с помощью другого ключа, а время выполнения уменьшилось в обоих случаях.

**3)** Оценить степень сжатия (используя любой доступный архиватор) открытого текста и соответствующего зашифрованного текста.

Результат сжатия исходного сообщения представлен на рисунке 5.

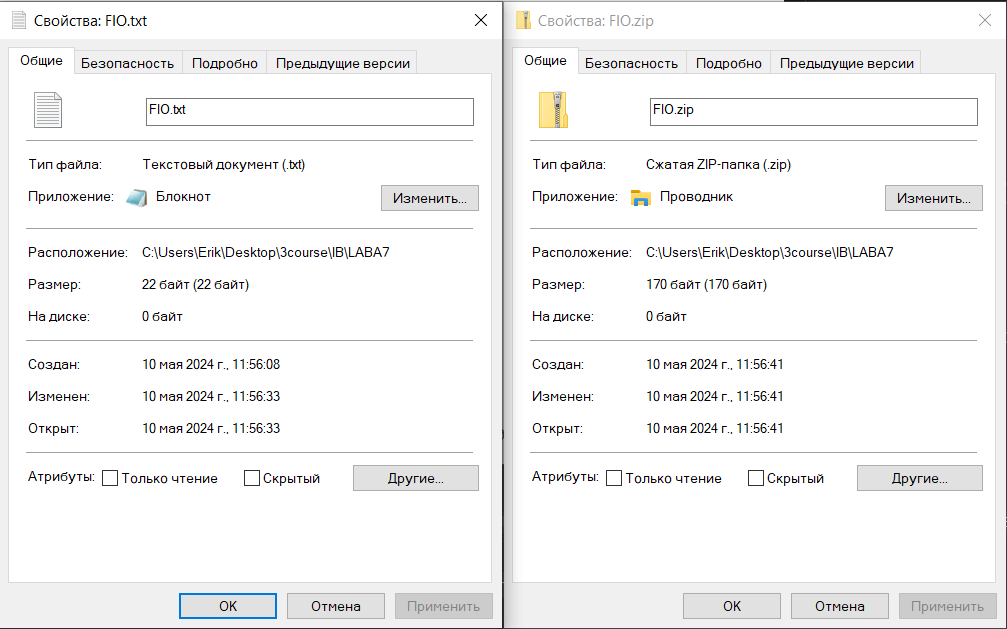


Рисунок 5 – Результат сжатия исходного сообщения

Результат сжатия зашифрованного сообщения представлен на рисунке 6.

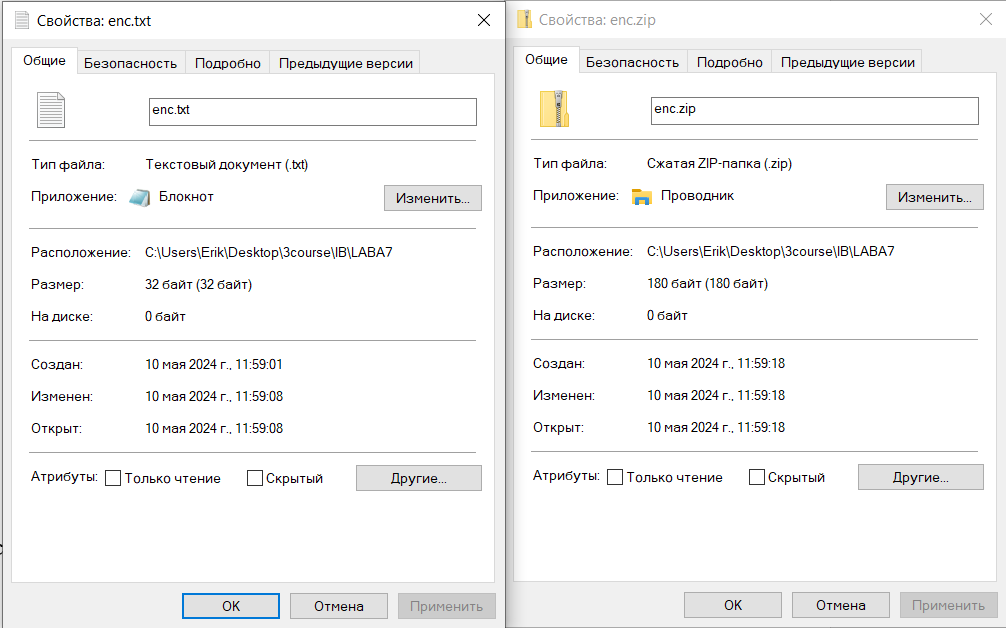


Рисунок 6 – Результат сжатия зашифрованного сообщения

Степень сжатия зашифрованного текста больше степени сжатия исходного.

**Вывод:**

Были приобретены практические навыки разработки и использования приложений для реализации блочных шифров. Был реализован алгоритм шифрования *DES*, выполнено шифрование с заданным ключом, слабым и полуслабым, построены графики времени выполнения шифрования и расшифрования.