**АННОТАЦИЯ**

Данный документ содержит пояснительную записку для «Программы отрицаемого шифрования на основе блочного шифра», которая может применяться для безопасного хранения информации. В разделе «Введение» указано название программы и документы, которые были использованы как основа при разработке. Раздел «Назначение и область применения» содержит функциональное и эксплуатационное назначение ПО, а также краткую характеристику его области применения. Раздел «Технические характеристики» описывает постановку задачи на разработку программы, алгоритм ее функционирования, обоснование выбора схемы алгоритма решения задачи, состав входных и выходных данных, состав технических и программных средств. В разделе «Ожидаемые технико-экономические показатели» отражены планируемая потребность и экономические превосходства разработки по сравнению с аналогами.

Настоящий документ разработан в соответствии с требованиями:

1) ГОСТ 19.101-77 Виды программ и программных документов [1];

2) ГОСТ 19.102-77 Стадии разработки [2];

3) ГОСТ 19.103-77 Обозначения программ и программных документов [3];

4) ГОСТ 19.104-78 Основные надписи [4];

5) ГОСТ 19.105-78 Общие требования к программным документам [5];

6) ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом [6];

7) ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению [7].

**СОДЕРЖАНИЕ**

[**1. Введение 3**](#_Toc418514176)

[**1.1. Наименование программы 3**](#_Toc418514177)

[**1.2 Документы, на основании которых ведется разработка 3**](#_Toc418514178)

[**2. Назначение и область применения 4**](#_Toc418514179)

[**2.1. Назначение программы 4**](#_Toc418514180)

[**2.1.1. Функциональное назначение 4**](#_Toc418514181)

[**2.1.2. Эксплуатационное назначение 4**](#_Toc418514182)

[**3. Технические характеристики 5**](#_Toc418514183)

[**3.1. Постановка задачи на разработку программы 5**](#_Toc418514184)

[**3.2. Описание алгоритма и функционирования программы 5**](#_Toc418514185)

[**3.2.1. Описание алгоритма программы 5**](#_Toc418514186)

[**3.2.2. Обоснование выбора алгоритма решения задачи 8**](#_Toc418514187)

[**3.2.3. Возможные взаимодействия программы с другими программами 8**](#_Toc418514188)

[**3.3. Описание и обоснование выбора метода организации входных и выходных данных 9**](#_Toc418514189)

[**3.3.1. Описание метода организации входных и выходных данных 9**](#_Toc418514190)

[**3.3.2. Обоснование выбора метода организации входных и выходных данных 9**](#_Toc418514191)

[**3.4. Описание и обоснование выбора состава технических и программных средств 9**](#_Toc418514192)

[**3.4.1. Состав технических и программных средств 9**](#_Toc418514193)

[**4. Технико-экономические показатели 11**](#_Toc418514194)

[**4.1. Предполагаемая потребность 11**](#_Toc418514195)

[**4.2. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами 11**](#_Toc418514196)

[**5. Источники, использованные при разработке 12**](#_Toc418514197)

**Приложение 1.** [**Описание и функциональное назначение классов 13**](#_Toc418514198)

**Приложение 2.** [**Описание и функциональное назначение методов, полей и свойств 1**](#_Toc418514199)**4**

# ВВЕДЕНИЕ

* 1. **Наименование программы**

1. “Программа отрицаемого шифрования на основе блочного шифра”. (“Deniable Encryption Program based on Block Cipher”)

## Документы, на основании которых ведется разработка

Основанием для разработки является приказ от 10.12.2018 г. № 2.3-02/1012-0 2 декана факультета компьютерных наук НИУ ВШЭ.

# 2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ

* 1. **Назначение программы**
     1. **Функциональное назначение**

Программа представляет из себя платформу для зашифровки и расшифровки информации, целью которой является сохранить зашифрованную информацию даже если злоумышленник насильственным образом заставляет выдать пользователя ключ расшифровки. Работа программы основана на алгоритме отрицаемого шифрования.

* + 1. **Эксплуатационное назначение**

Приложение предназначено для пользователей персональных компьютеров, которые заинтересованы в безопасном хранении или передачи своих данных, даже при непосредственном давлении на субъект, имеющий информацию для расшифровки сообщения.

# ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

* 1. **Постановка задачи на разработку программы**

Разработать две программы, одна из которых будет реализовывать алгоритм отрицаемого шифрования и результатом работы которой будет запись зашифрованного текста в файл, а вторая - расшифровки щифротекста из файла.

* 1. **Описание алгоритма и функционирования программы**
     1. **Описание алгоритма программы**

Схему работы программы можно разделить на несколько шагов:

1. Чтение секретного и фиктивного (которое не содержит важной информации) сообщений из текстовых файлов, обработка введенного пользователем ключа и генерация необходимых параметров для зашифровки
2. Получение шифротекста алгоритмом отрицаемого шифрования на основе блочного шифра[8]
3. Расшифровка шифротекста по параметрам, введенным пользователем в приложении для расшифровки

* + - 1. **Чтение секретного и фиктивного сообщения и обработка параметров шифрования**

1. После чтение данных из файлов (обозначим секретное сообщение TrueM, а фиктивное FalseM), количество символов в сообщениях нужно сделать одинаковым, для этого можно просто дополнить кратчайшее из сообщений пробелами. К тому же длинна сообщений должна делиться на размер блока (в алгоритме, который использован в программе n = 128), для этого можно дополнить сообщения символом “звездочка”.
2. Размер ключа для алгоритма шифрования в данной программе равен 256 бит. Поэтому чтобы не заставлять пользователя придумывать ключ, состоящий из такой большой длины, программа работает по следующему алгоритму: пользователь вводит 4 числа в диапазоне от 0 до 255, а остальные 28 байт автоматически заполняются этими 4 числами. Таким же образом обрабатывается и ключ для фиктивного сообщения
3. Векторы инициализации (IV1 для TrueM, IV2 для FalseM) генерируются автоматически при создании нового объекта класса Aes и предоставляется пользователю в специальном окне.
4. Для осуществления алгоритма отрицаемого необходимы два параметра: P1 - для секретного сообщения и P2 – для фиктивного. P1 и P2 должны иметь длину на один бит больше размера блока, т.е. 129 бит, а также быть взаимно простыми, поэтому для их генерации был использован следующий алгоритм. Генерируется случайное число P1, также генерируется случайное число P2, затем увеличиваем P на единицу пока НОД (P1, P)!=1.

Генерирование 129-битного числа на языке C#

public static BigInteger Gnr1()

{

Random rnd = new Random();

byte[] y = new byte[17];

y[16] = 1;

for (int i = 0; i < 16; i++)

{

y[i] = (byte)rnd.Next(0, 256);

}

BigInteger a = new BigInteger(y);

return a;

}

1. Далее выбирается пользователь (именуется как «текущий.
   * + 1. **Получение шифротекста алгоритмом отрицаемого шифрования на основе блочного шифра**

Данный алгоритм получает на вход два независимых друг от друга сообщения и шифрует в общий шифротекст, по двум независимым ключам. Цель алгоритма обеспечить возможность восстановления из полученного шифротекста каждого из входных сообщений, с использованием одинакового алгоритма.

Пусть M – секретное сообщение, а N – фиктивное, полный ключ отрицаемого шифрования будет состоять из секретного ключа (K1, P1), и фиктивного ключа (K2, P2). Где K1, K2 - ключи для бличного шифрования сообщений с помощью блочного шифра F (в данной программе использован встроенный в Framework алгоритм AES[9]), а P1 и P2, как уже было сказано, взаимно простые числа длиной n+1 бит, тогда алгоритм действий следующий.

1. Зашифруем секретное сообщение M с помощью ключа K1 алгоритмом блочного шифрования F, на выходе получим массив блоков EnM
2. Зашифруем фиктивное сообщение N с помощью ключа K2 алгоритмом блочного шифрования F, на выходе получим массив блоков EnN
3. Переведем очередные блоки EnM[i] и EnM[i] в числа
4. блоком шифротекста будет являться решение следующего уравнения

По китайской теореме об остатках[10]

.

– обратный элемент к P2 в кольце по модулю P1[11].

Поиск обратного элемента в кольце по модулю

public static BigInteger Invert(BigInteger modn, BigInteger a)

{

BigInteger z;

BigInteger b = modn, x = BigInteger.Zero, d = BigInteger.One;

while (a.CompareTo(BigInteger.Zero) == 1)

{

BigInteger q = BigInteger.DivRem(b, a, out z);

BigInteger k = a;

a = z;

b = k;

k = d;

d = BigInteger.Subtract(x, BigInteger.Multiply(q, d));

x = k;

}

x = BigInteger.Remainder(x, modn);

if (x.CompareTo(BigInteger.Zero) == -1)

{ x = BigInteger.Remainder(BigInteger.Add(x, modn), modn); }

return x;

}

Получения блока шифротекста С[i]

koef1 = BigInteger.Multiply(p2, BigArifm.Inverse(p1, p2));

koef2 = BigInteger.Multiply(p1, BigArifm.Inverse(p2, p1));

koef3 = BigInteger.Multiply(p1, p2);

byte[][] C = BlockCrypt.CryptC(koef1, koef2, koef3, EnM, EnN);

public static byte[][] CryptC(BigInteger koef1, BigInteger koef2, BigInteger koef3, byte[][] EnM, byte[][] EnN)

{

byte[][] res = new byte[EnM.Length][];

for (int i = 0; i < EnM.Length; i++)

{

res[i] = new byte[34];

BigInteger CM = new BigInteger(EnM[i]);

BigInteger CN = new BigInteger(EnN[i]);

BigInteger C = BigInteger.Remainder(BigInteger.Add(BigInteger.Multiply(CM, koef1), BigInteger.Multiply(CN, koef2)), koef3);

byte[] m = C.ToByteArray();

for (int j = 0; j < m.Length; j++)

{

res[i][j] = m[j];

}

res[i][33] = (byte)m.Length;

}

return res; }

1. Перевод полученых блоков шифротекста в текст

public static string BytetoSymb(byte[][] bt)

{

string ss="";

for (int i = 0; i < bt.Length; i++)

{

for (int j = 0; j < bt[i].Length; j++)

{

ss += (char)bt[i][j];

}

}

return ss;

}

* + - 1. **Расшифровка шифротекста по параметрам, введенным пользователем**

1. Прочитаем из файла шифротекст, а также обработаем данные, введенные пользователем (K, P, IV)
2. Разобьем считанный текст на блоки и поместим в массив С
3. Представим блоки, состоящие из байтов, как числа
4. Перейдем к новым блокам по следующему алгоритму:
5. Расшифруем полученные блоки алгоритмом шифрование и если при расшифровке был использован секретный ключ K1, секретный вектор инициализации IV1 и секретный параметр P1, то получим секретное сообщение, а если фиктивные параметры K2, IV2 и P2, то фиктивное

.

* + 1. **Обоснование выбора алгоритма решения задачи**

Основным требованием к алгоритму отрицаемого шифрование является его стойкость и возможность обеспечения безопасного хранения информации. Можно выделить несколько критериев по которым можно дать оценку алгоритму.

1) Криптограмма, полученная в результате отрицаемого шифрования,

не должна быть отличима от криптограммы, получаемой в ходе вероятностного шифрования[11].

Проверим данный критерий. Для этого рассмотрим алгоритм вероятностного шифрования, построенный на блочном шифре, который можно описать следующим образом:

1. Разбить сообщение N на блоки по n бит
2. Зашифровать каждый блок N[i] алгоритмом блочного шифрования F по ключу K2, результатом которого будет криптограмма EnN[i]
3. Сгенерировать случайным образом число *r,* взаимно простое с P2, и такое что
4. Сгенерировать случайное число R
5. Получить C[i] блок шифротекста EnN[i] как решение системы

Таким образом, легко показать, что любой блок C[i] шифротекста C мог быть получен в процессе вероятностного шифрования фиктивного сообщения N по фиктивному ключу при использовании данного алгоритма вероятностного шифрования, основанного на блочном шифре.

2) Восстановление фиктивного и секретного сообщения должно происходить независимо друг от друга.

Выполнение данного критерия очевидно. Входными данными для расшифровки является криптограмма и ключ, фиктивный или секретный, но не вместе

3) Алгоритмы расшифровывания фиктивного и секретного сообщений должны совпадать.

В данной программе для расшифровки и зашифровки используется блочный шифр AES

* + 1. **Возможные взаимодействия программы с другими программами**

Взаимодействия с другими программами не предусмотрено.

* 1. **Описание и обоснование выбора метода организации входных и выходных данных**
     1. **Описание метода организации входных и выходных данных**

Секретное и фиктивное сообщение должно храниться в разных текстовых файлах txt. Ввести ключи пользователю предлагается вручную. Вектор инициализации и параметр P отображается в специально выделенных в приложении окнах. Криптограмма загружается в текстовый файл.

* + 1. **Обоснование выбора метода организации входных и выходных данных**

Текстовый файл формата txt удобен и прост для работы в С#. А размер сообщения может быть настолько большим, что ввод его с помощью клавиатуры нецелесообразен. В свою очередь ручной ввод ключа, а также вектора инициализации и параметра P в приложении для расшифровки не составит труда, потому что их длина относительно небольшая.

* 1. **Описание и обоснование выбора состава технических и программных средств**
     1. **Состав технических и программных средств**

Для корректной работы программы рекомендуется компьютер, со следующими   
техническими характеристиками:

- процессор с частотой 1,2 ГГц или более;  
 - 1024 МБ ОЗУ и выше;

- 4 ГБ свободного места на жестком диске  
 - монитор с разрешением 800х600 или более высоким;

Состав программных средств, необходимых для работы системы:

- операционная система Windows 7, Windows 8, Windows 8.1 и выше;  
 - установленный Microsoft .NET Framework 4.0 или выше.

1. **ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ**
   1. **Предполагаемая потребность**

Безопасность данных в дни, когда информация один из главных ресурсов, очень важна. Данная программа позволяет не только зашитить важную информацию от кибератак, при которых злоумышленник пытается искать бреши в алгоритме шифрования, но и при непосредственном насильственном воздействии на владельца данных.

* 1. **Экономические преимущества разработки по сравнению с аналогами**

Для использования программы не требуется компьютер с дорогой комплектацией и особенными характеристиками. приложение бесплатно для скачивания.

1. **ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ**
2. ГОСТ 19.101-77 Виды программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.102-77 Стадии разработки. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.103-77 Обозначения программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001
5. ГОСТ 19.104-78 Основные надписи. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.105-78 Общие требования к программным документам. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001
8. ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. Способы и алгоритмы псевдовероятностного шифрования с разделяющим ключом/ Молдовян А.А., Молдовян Н.А.: журнал: Информационная безопастность, 2018. – с 123-127
10. Алгоритм блочного шифрования Advanced\_Encryption\_Standard [Электронный ресурс] // URL: <https://ru.wikipedia.org/wiki/Advanced_Encryption_Standard> (Дата обращения: 19.05.2019, режим доступа: свободный).
11. Китайская теорема об остатках [Электронный ресурс] //URL: <http://e-maxx.ru/algo/export_chinese_theorem> (Дата обращения: 18.05.2019, режим доступа: свободный).
12. Обратный элемент в кольце по модулю [Электронный ресурс] //URL: <http://e-maxx.ru/algo/export_reverse_element> (Дата обращения: 10.05.2019, режим доступа: свободный).

|  |  |
| --- | --- |
| **ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ** | **ПРИЛОЖЕНИЕ 1** |

Таблица 1 – Классы решения Slock

|  |  |
| --- | --- |
| **Класс** | **Назначение** |
| *Form1* | Работа с интерфейсом программы |
| *BlockCrypt* | Реализация методов, требуемых для шифрования |
| *BigArifm* | Реализация методов для работы с большими числами |

Таблица 2 – Классы решения DeCr

|  |  |
| --- | --- |
| **Класс** | **Назначение** |
| *Form1* | Работа с интерфейсом программы |
| *Decrypt* | Реализация методов расшифровки |

|  |  |
| --- | --- |
| **ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ МЕТОДОВ, ПОЛЕЙ И СВОЙСТВ** | **ПРИЛОЖЕНИЕ 2** |

Таблица 3 – Описание членов класса *Form1 проекта Slock*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | *Назначение* |
| открытьToolStripMenuItem\_Click | private | void | Object, eventArgs | Чтения данных из файла |
| AddStr | public | void |  | Дополнение строки до нужного размера |
| Readkey | public | byte[] | string | Чтение ключа из textbox’а |
| сохранитьКакToolStripMenuItem\_Click | private | void | Object, eventArgs | Получения криптограммы и запись в файл |

Таблица 4 – Описание членов класса BigArifm

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | *Назначение* |
| Gnr1 | public | BigInteger |  | Создание 129-битного случайного числа |
| Gnr2 | public | BigInteger | BigInteger | Генерация случайного числа, взаимно простого с данным |
| Inverse | public | BigInteger | BigInteger,BigInteger | Поиск обратного элемента в кольце по модулю |

Таблица 5 – Описание членов класса BlockCrypt

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | *Назначение* |
| CorrectKey | public | Byte[] | Byte[] | Увеличение размера ключа до требуемого |
| EncryptStringToBytes\_Aes | public | Byte[] | string, byte[], byte[] | Зашифровка текста по ключу и вектору инициализации алгоритмом Aes |
| CryptC | public | byte[][] | BigInteger, BigInteger,  BigInteger,  Byte[][], byte[] | Формирование одно блока криптограммы из двух, полученных шифрованием фиктивного и секретного сообщений |

Таблица 6 – Описание членов класса Form1 проекта DeCr

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | *Назначение* |
| открытьToolStripMenuItem\_Click | private | void | Object, eventArgs | Чтение криптограммы для расшифровки из файла |
| MestoByte | public | Byte[] | string | Перевод текста в блоки |
| сохранитьToolStripMenuItem\_Click | Private | void | Object, eventArgs | Выполнение основного кода программы и запись расшифрованного текста в файл |
| ReadKey | public | Byte[] | string | Чтение ключа из textbox’а |
| ProvP | public | bool | BigInteger | Корректен ли веденный параметр P |
| CorrectKey | public | Byte[] | byte[] | Увеличение размера ключа до требуемого |
| toBlocks | public | Byte[][] | Byte[] | Разбивает сообщение на блоки |
| toNewBlocks | public | Byte[][] | Byte[][] | Изменение блоков шифротекста по заданному алгоритму |

Таблица 7 – Описание членов класса Decrypt

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | *Назначение* |
| DecryptStringFromBytes\_Aes | public | string | byte, byte[], byte[] | Расшифровка данных по блочному алгоритму AES |
| BytetoSymb | public | string | Byte[][] | Перевод из массива байтов в строку |
| BytetoOneArray | public | Byte[] | Byte[] | Получение массива символов |

Таблица 8 – Описание членов класса Form1 решения Slock

|  |  |  |  |
| --- | --- | --- | --- |
| **Поля** | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Назначение* |
| TrueM | private | string | Секретное сообщение |
| FalseM | private | String | Фиктивное сообщение |
| EnM | private | Byte[][] | Секретное сообщение разделенное на блоки |
| EnN | private | Byte[][] | Фиктивное сообщение разделенное на блоки |
| C | private | Byte[][] | Криптограмма разделенная на блоки |
| P1 | private | BigInteger | Параметр для шифровки секретного сообщения |
| P2 | private | BigInteger | Параметр для шифровки фиктивного сообщения |
| IV\_1 | private | Byte[] | Вектор инициализации для шифровки секретного сообщения |
| IV\_2 | private | Byte[] | Вектор инициализации для шифровки фиктивного сообщения |
| Koef1 | private | BigInteger | Коэфициент для реализации алгоритма |
| Koef2 | private | BigInteger | Коэфициент для реализации алгоритма |
| Koef3 | private | BigInteger | Коэфициент для реализации алгоритма |
| StrC | private | string | Полученная криптограмма |

Таблица 9 – Описание членов класса Form1 проекта DeCr

|  |  |  |  |
| --- | --- | --- | --- |
| **Поля** | | | |
| Имя | *Мод. Доступа* | *Тип* | *Назначение* |
| Mes | private | string | Считанная из файла криптограмма |
| key | private | Byte[] | Ключ для расшифровки |
| С1 | private | Byte[][] | Считанное сообщение, разделенное на блоки |
| С2 | private | Byte[][] | Преобразованное сообщение, разделенное на блоки |
| forCr | private | Byte[] | Расшифрованный текст, разделенный на блоки |
| P | private | BigInteger | Параметр для расшифровки сообщения |
| IV | private | BigInteger | Вектор инициализации для расшифровки сообщения |
| StrC | private | string | Полученный расшифрованный текст |

**ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопрово-дительно-  го докум. и дата | Подл. | Да-  та |
| Изм. | Изменен-  ных | Заменен-  ных | новых | аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |