Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Южно-Уральский государственный университет»
(национальный исследовательский университет)
Факультет математики, механики и компьютерных наук
Кафедра дифференциальных и стохастических уравнений

РАБОТА ПРОВЕРЕНА
Рецензент, доцент кафедры комп. топологии и алгебры ЧелГУ, канд. физ.-мат. наук, доцент
_____ / О.В. Митина/
«___» _____2016 г.

ДОПУСТИТЬ К ЗАЩИТЕ
Зав. кафедрой дифференциальных и стохастических уравнений, ЮУрГУ, доктор физ-мат. наук, доцент
_____ / С.А. Загребина /
«___» _____2016 г.

КРИПТОГРАФИЧЕСКИЕ СИСТЕМЫ СВЯЗАННЫЕ С ЗАДАЧАМИ ФАКТОРИЗАЦИИ

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 09.04.04.2016.129-129.ВКР

Руководитель, доктор физ.-мат.наук, доцент
_____ /Н.Д. Зюляркина/
«07» июня 2016 г.

Автор, студент группы ММиКН-293
_____ /А.В. Чарыков/
«___» _____2016 г.

Нормоконтролер, канд. физ.-мат. наук, доцент
_____ /М.А. Сагадеева/
«___» _____2016 г.

Челябинск 2016

Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Южно-Уральский государственный университет»
(национальный исследовательский университет)
Факультет математики, механики и компьютерных наук
Кафедра дифференциальных и стохастических уравнений

З А Д А Н И Е

студенту группы ММиКН-293
Чарыкову Алексею Викторовичу
на выпускную квалификационную работу
по направлению 09.04.04 – ПРОГРАММНАЯ ИНЖЕНЕРИЯ

1. **Тема диссертации:** «Криптографические системы связанные с задачами факторизации». (Утверждена приказом по университету от «15» апреля 2016г. № 661)

2. **Перечень подлежащих исследованию вопросов**
   2.1. Литературный обзор, выбор способа модификации существующих криптосистем
   2.2. Постановка задачи
   2.3. Разработка модификации существующих криптосистем

3. **Календарный план подготовки выпускной квалификационной работы**

Наименование этапов дипломной работы	Срок выполнения этапов работы	Отметка о выполнении
1. Обзор литературы	01.02.16 – 14.02.16	Зюл
2. Анализ криптографических систем с открытым ключом и рассмотрение существующих методов факторизации	15.02.16 – 28.02.15	Зюл
3. Изучение модификаций криптосистем с открытым ключом	29.02.16 – 15.04.16	Зюл
4. Подготовка текста выпускной квалификационной работы	15.04.16 – 15.05.16	Зюл
5. Проверка и рецензирование работы руководителем, исправление замечаний	15.05.16 – 25.05.16	Зюл
6. Подготовка доклада и текста выступления	26.05.16 – 10.06.16	Зюл
7. Внешнее рецензирование	20.05.16 – 05.06.16	Зюл
8. Защита дипломной работы	10.06.16 – 20.06.16	

4. **Дата выдачи задания** «01» февраля 2016 г.

Руководитель работы
доктор. ф.-м. наук, доцент Зюл ***Зюляркина Н.Д.***
(подпись)

Задание принял к исполнению ***Чарыков А.В.***
(подпись)

ВВЕДЕНИЕ

В современных условиях глобальной информатизации компьютерные технологии получили широкое распространение и применение во всех сферах деятельности человека. При этом одной из наиболее серьезных угроз является угроза несанкционированного доступа к защищаемой информации со стороны «третьих» лиц. Для противодействия данной угрозе применяются различные методы и средства, наиболее эффективным при этом является криптографическая защита данных. Современные криптосистемы могут обеспечивать не только секретность передаваемых сообщений, но и их аутентичность (подлинность), а также подтверждение подлинности пользователя. Они делятся на симметричные криптосистемы (с секретным ключом — secret key systems) и асимметричные криптосистемы (системы открытого шифрования — с открытым ключом и т.д.- public key systems). К числу преимуществ асимметричных криптографических систем перед симметричными криптосистемами относится решение проблемы распределения ключей между пользователями, так как каждый пользователь может сгенерировать свою пару ключей сам, а открытые ключи пользователей могут свободно публиковаться и распространяться по сетевым коммуникациям. Кроме того, асимметричные криптосистемы позволяют реализовать протоколы взаимодействия сторон, которые не доверяют друг другу, поскольку при использовании асимметричных криптосистем закрытый ключ должен быть известен только его владельцу.

Большинство криптосистем с открытым ключом основаны на проблеме факторизации больших чисел. К примеру, одна из самых распространенных – RSA – использует в качестве открытого ключа n произведение двух больших чисел. Сложность взлома такого алгоритма состоит в трудности разложения числа n на множители. Но эту задачу решить реально. И с каждым годом

процесс разложения становится все быстрее. Поэтому актуальным выступает проблема формирования оптимальных с точки зрения как защищенности, так и универсальности и ресурсоемкости алгоритмов шифрования.

Одним из интересных классов данных криптосистем являются системы, основанные на задаче об укладке рюкзака. Задача об укладке рюкзака является NP-полной задачей, что является ее основным преимуществом. Кроме того, при создании алгоритма, который будет использовать NP-полноту, допускается создание системы, которая будет эффективнее, чем системы на основе целочисленной факторизации и дискретного логарифмирования.

Целью диссертационной работы является исследование и построение криптосистемы, построенной по принципу мультипликативного рюкзака с использованием линейных групп для построения открытого и секретного ключей.

В соответствии с поставленной целью были определены следующие задачи.

1. Исследование криптосистем с открытым ключом и односторонних функций, которые используются для построения таких криптосистем.

2. Исследование классической задачи об укладке рюкзака и задачи о мультипликативном рюкзаке

3. Построение математической модели и исследование криптосистемы, основанной на сплетении групп при построении ключей.

4. Разработка, реализация и оценка эффективности разрабатываемой криптосистемы. Для достижения поставленной цели в диссертационной работе используются методы теории групп, теории информации, программирования, а также компьютерной алгебры.

Научная новизна работы заключается в том, что в ней впервые предложено использование свойств сплетения групп для построения рюкзачной криптосистемы.

1. КРИПТОГРАФИЧЕСКИЕ СИСТЕМЫ С ОТКРЫТЫМ КЛЮЧОМ

1.1. Основы шифрования с открытым ключом

Существует два класса криптографических алгоритмов: симметричные (один ключевой элемент легко вычисляется при наличии второго; высокая скорость преобразования данных) и несимметричные. Их особенностью является наличие двух ключевых элементов, при этом вычисление одного из них при наличии второго является вычислительно сложной задачей. Начало асимметричным шифрам было положено в работе «Новые направления в современной криптографии» Уитфилда Диффи (Bailey Whitfield Diffie) и Мартина Хеллмана (Martin E. Heilman), опубликованной в 1976 году. Находясь под влиянием работы Ральфа Меркле (Ralph Merkle) о распространении открытого ключа, они предложили метод получения секретных ключей для симметричного шифрования, используя открытый канал.

Первой реальной криптосистемой с открытым ключом, пригодной и для шифрования, и для цифровой подписи считают алгоритм RSA(названный по имени авторов – Рон Ривест (Ronald Linn Rivest), Ади Шамир (Adi Shamir) и Леонард Адлеман (Leonard Adleman) из Массачусетского Технологического Института (MIT)).

В ассиметричных алгоритмах один из ключей используется для шифрования, а другой, – для расшифрования. Один из двух ключей является открытым (publickey) и может быть объявлен всем, а второй – закрытым (privatekey) и должендержаться строго в секрете. Криптографическая система определяет, какой из ключей, открытый или закрытый, используется для шифрования, а какой для расшифрования.

В современном мире асимметричные алгоритмы широко применяются на практике, как для обеспечения информационной безопасности телекоммуникационных сетей, в том числе сетей, имеющих сложную

топологию,так и для обеспечения информационной безопасности в глобальной сети Internet; а также в различных банковских системах.

Алгоритмы шифрования с открытым ключом используются, как минимум, для решения трех задач:

1) шифрование передаваемых и хранимых данных в целях их защиты от несанкционированного доступа третьих лиц;

2) формирование цифровой подписи под электронными документами;

3) распределение секретных ключей, используемых потом при шифровании документов симметричными методами.

1.2. Односторонние функции

Все алгоритмы шифрования с открытым ключом используют в своей основе односторонние функций. *Односторонней функцией* называется математическая функция, которую легко вычислитьдля любого входного значения, но трудно найти по значению функции соответствующее значение аргумента. Зная x легко вычислить $f(x)$,но по известному $f(x)$трудно найти подходящее значение x, то есть для вычисления понадобится не один год вычислений на современных компьютерах. Односторонние функции применяются в криптографии также в качестве хеш-функций. Использовать односторонние функции для шифрования сообщений с целью их защиты не имеет смысла, так как обратно расшифровать зашифрованное сообщение уже не получится. Для целей шифрования используются специальные односторонние функции – односторонние функции с секретом – это особый вид односторонних функций, имеющих некоторый секрет, позволяющий относительно быстро вычислить обратное значение функции.

Для односторонней функции с секретом справедливы следующие утверждения:

1) зная x, легко вычислить $f(x)$,

2) по известному значению $f(x)$ трудно найти x,

3) зная дополнительно некоторую секретную информацию, можно легко вычислить x.

Например, функция *sin*.Зная x, легко найти значение $sin(x)$(например, $x=\pi$, тогда $sin(\pi) = 0)$.Однако, если $sin(x)=0$, однозначно определить x нельзя, так как в этом случае x может быть любым числом, определяемым по формуле $i*\pi$, где i-целое число.

Не всякая односторонняя функция годится для использования в реальных криптосистемах. В их числе и функция *sin*.

Чтобы гарантировать надежную защиту информации, криптосистемы с открытым ключом должны отвечать двум важным требованиям:

1) преобразование исходного текста должно быть условно необратимым и исключать его восстановление на основе открытого ключа;

2) восстановление закрытого ключа с помощью открытого должно быть невозможным на современном технологическом уровне.

Большинство криптосистем с открытым ключом основываются на одном из следующих типов односторонних преобразований:

- факторизация (алгоритм RSA);
- дискретное логарифмирование (алгоритм Диффи-Хеллмана-Меркля, схема Эль-Гамаля);
- задача об укладке рюкзака (ранца) (авторя Р. Меркль и М.Хеллман);
- вычисление корней алгебраических уравнений;
- использование конечных автоматов (автор ТаоРснжи);
- использование кодовых конструкций;
- использование свойств эллиптических кривых.

На сегодняшний день наиболее распространенными являются криптосистемы, основанные на задаче факторизации. В математике факториза́ция — это декомпозиция объекта (например, числа, полинома или

матрицы) в произведение других объектов или факторов, которые, будучи перемноженными, дают исходный объект Рассмотрим подробнее существующие методы факторизации длинных чисел. Их можно разделить на две основные группы:

1) методы, сложность которых экспоненциально зависит от длины числа:

– метод Ферма;

– ρ-алгоритм Полларда;

– (p-1) алгоритм Полларда;

2) алгоритмы, сложность которых – субэкспоненциальная.

– алгоритм Диксона;

– метод квадратичного решета;

– метод Ленстры;

– метод решета числового поля.

Метод факторизации Ферма натурального нечетного числа n состоит в поиске таких целых чисел x и y, что $x^2 - y^2 = n$, что ведет к разложению $n = (x - y)\cdot(x + y)$. Метод быстро работает, если n является произведением двух близких к друг другу сомножителей. В частности, именно поэтому в RSA требуют, чтобы разность между секретными простыми сомножителями модуля была велика. Для разложения на множители нечетного числа n ищутся два числа x и y такие, что $x^2 - y^2 = n$, или $n = (x - y) \cdot (x + y)$. При этом числа $(x + y)$ и $(x - y)$ являются множителями n, возможно, тривиальными. Равенство $x^2 - y^2 = n$ равносильно $x^2 - n = y^2$, и $x^2 - n$ является квадратом. Поиск квадрата такого вида начинается с наименьшего числа, при котором разность $x^2 - n$ неотрицательна. Для каждого значения x вычисляют $x^2 - n$ и проверяют, не является ли это число точным квадратом. Если нет, x увеличивают на единицу, иначе получено разложение.

Алгоритм Диксона – алгоритм факторизации, использующий в своей основе идею Лежандра, заключающуюся в поиске пары целых чисел x и y таких,

что $x^2 \equiv y^2 \pmod n$ и $x \neq \pm y \pmod n$. Метод Диксона является обобщением метода Ферма.

ρ-алгоритм Джона Полларда, предложенный им в 1975 году, служит для факторизации целых чисел. Он основан на том, что вычисляется некий многочлен степени не выше второй от начального числа X – f(X). Алгоритм имеет в названии ρ потому, что эскиз итераций похож на круг с хвостом.

P-1 метод Полларда — алгоритм разложения натурального числа N на простые множители. Алгоритм предназначен для нахождения простых делителей p, у которых p-1 хорошо раскладывается на множители, то есть имеет небольшой максимальный простой делитель.

Метод квадратичного решета представляет собой разновидность метода факторных баз (обобщение метода факторизации Ферма). В качестве факторной базы B берется множество простых чисел, состоящее из p = 2 и всех таких нечетных простых чисел p, не превосходящих заданной границы P (которая выбирается из соображений оптимальности), что n — квадратичный вычет по модулю p. Множество S целых чисел, в котором ищутся B-числа (B-число — целое число, делящееся только на простые числа из B) выглядит следующим образом:

$$S = \{t^2 - n \mid [\sqrt{n}] + 1 \leq t \leq [\sqrt{n}] + A\}$$

Далее, вместо того, чтобы брать одно за другим $s \in S$, и делить его на простые числа из B, берутся одно за другим каждое $p \in B$ и проверяется делимость на p (и его степени) одновременно для всех $s \in S$ -. Для построения списка всех простых p, не превосходящих A, можно использовать решето Эратосфена.

Факторизация с помощью эллиптических кривых является третьим по скорости работы после общего метода решета числового поля и метода квадратичного решета. На практике в основном используется для выявления небольших простых делителей числа. Если полученное после работы алгоритма

число все еще является составным, то остальные сомножители — большие числа. При увеличении количества кривых шансы найти простой сомножитель возрастают.

К задачам, связанным с проблемой факторизации, можно отнести и задачи об укладке рюкзака, и их модификации.

Рюкзачные схемы удобны для иллюстрации основных идей криптографии с открытым ключом. Они являются достаточно гибкими и позволяют для преодоления криптографических недостатков вводить новые варианты.

Задача об укладке рюкзака является NP-полной задачей, то есть для неё не существует полиномиального алгоритма, решающего ее за разумное время, в этом и заключается проблема. Алгоритм выбирается либо быстрый (он далеко не всегда решает задачу оптимальным образом), либо точный (не является работоспособным для больших значений).

Задача об укладке рюкзака лежит в основе ряда криптосистем, отличающихся довольно простотой реализации. Однако при их более детальном анализе были выявлены существенные недостатки, которые делали эти системы уязвимыми для различного вида криптографических атак. В настоящее время криптосистемы, основанные на рюкзачных схемах не популярны, однако ведется работа по их корректировке и модификации, которая позволит улучшить их криптостойкость.

Одним из способов такой модификации является создание рюкзачной схемы на группах..

1.3. Описание классической задачи об укладке рюкзака

Пусть имеется упорядоченный набор чисел $(a_1,a_2,...,a_n)$ (этот набор называют *рюкзаком* или *рюкзачным вектором)* и число m. Требуется указать такой бинарный вектор $(x_1,x_2,...,x_n)$, для которого выполняется равенство

$$x_1 \cdot a_1 + x_2 \cdot a_2 + \ldots + x_n \cdot a_n = m$$

В общем случае для данной задачи нет эффективного алгоритма решения, и приходится применять полный перебор для нахождения требуемого вектора или доказательства отсутствия решения. Кроме того, в общей постановке задача о рюкзаке может иметь несколько различных решений. Но если рюкзак является сверхрастущим, то решение в случае его существования единственно и существует эффективный алгоритм его нахождения.

Рюкзак с положительными элементами $(a_1, a_2, \ldots, a_n)$ будем называть *сверхрастущим*, если $a_2 > a_1, a_3 > a_1 + a_2, \ldots, a_n > a_1 + a_2 + \ldots + a_{n-1}$.

На основе задачи о рюкзаке разработан ряд криптографических систем. Первой из них была система Меркля-Хеллмана, описание которой приведено ниже.

Генерация ключей

1. Выбирается некоторый сверхрастущий рюкзак.
2. Выбирается число $к$ ($к > a_1 + a_2 + \ldots + a_n$).
3. Выбирается число c, взаимно простое с $к$.
4. Формируется рюкзак-ловушка $(b_1, b_2, \ldots, b_n) = c(a_1, a_2, \ldots, a_n)(\mathrm{mod}\, k)$, который и является открытым ключом.
5. Числа c и $к$ являются секретными ключами.

Алгоритм шифрования

1. Открытый текст представляется в виде двоичной последовательности.
2. Последовательность разбивается на блоки длины n.
3. Каждый блок $(x_1, x_2, \ldots, x_n)$ заменяется на число m, вычисленное по следующему правилу:

$$\sum_{i=1}^{n} ai \cdot xi = m.$$

Алгоритм дешифровки

1. Находится исходный сверхрастущий рюкзак:

$$(a_1,a_2,...,a_n)=c^{-1}(b_1,...,b_n)(mod k).$$

2. Для каждого элемента mшифр текста вычисляется элемент $m'= c^{-1}\cdot m$.

3. Для вычисленного m' решается задача для рюкзака $(a_1,a_2,...,a_n)$ и находится блок открытого текста $(x, ,x_2,...,x_n)$.

Пример 1.1. Используется латинский алфавит, в котором каждая буква представлена пятиразрядной двоичной записью своего номера. Рюкзак-ловушка B=(182,128,192,175,50,100) получен из сверхрастущего рюкзака A путем умножения на $c=91$ и приведением по модулю $n=300$. Сообщение Y=(232,178,502) получено шифрованием на основе рюкзака B. Восстановить исходный рюкзак A, и, используя его, расшифровать сообщение $У$.

Решение.

1) Найдемc^{-1}(mod300): $c^{-1}=91^{-1}=211$.

2) Восстановим исходный рюкзак:

$A = 211B(\text{mod}300) = 211(182,128,192,175,50,100)(\text{mod}300) = (2,8,12, 25,50,100)$

3) Преобразуем сообщение У:

$У \rightarrow Z=211Y(\text{mod}300) = 211(232,178, 502)(\text{mod}300) = (52, 58, 22)$.

4) Решим задачу о рюкзаке для каждого элемента сообщения Z:

52=50+2→(1,0, 0, 0, 1,0),

58=50+8→(0, 1,0,0, 1,0),

22=12+10=12+8+2 → (1, 1, 1,0, 0, 0).

5) Запишем полученные двоичные векторы в единую последовательность, которую разобьем на блоки длины 5:

(1,0, 0, 0,1, 0, 0,1, 0, 0,1, 0,1,1,1, 0, 0, 0)→ (1,0, 0, 0,1)(0, 0,1, 0, 0)(1, 0,1,1,1)(0, 0, 0)

Последний блок из трех элементов исключаем из рассмотрения.

6) Сопоставим каждому полученному блоку число, для которого этот блок является двоичной записью, и найдем соответствующую букву латинского алфавита:

(1,0, 0,0, 1) →17→R,

(0, 0, 1,0, 0) →4→E,

(1, 0, 1, 1, 1) →23→X.

Задача о рюкзаке лежит в основе криптосистем Меркля-Хеллмана и Грэм-Шамира (используется сверхрастущий рюкзак, но маскировка производится не с помощью приведения по модулю, а с использование вектора случайного шума). В системе Мории-Касахара используется мультипликативный способ шифрования и формирования секретного ключа. В системе Хора-Ривеста вычисления основаны на конечных полях, а система Накаше-Штерна объединяет два метода: рюкзак Меркля-Хеллмана и алгоритм Полига-Хеллмана.

1.4. Задача о мультипликативном рюкзаке

Идея мультипликативного рюкзака была предложена в 1988 Масакацу Мории(MasakatuMorii) и Масао Касахара(MasaoKasahara). Мультипликативный рюкзак использует точно такую же функцию шифрования, как и у аддитивной рюкзачной схемы Меркля-Хеллмана.

Алгоритм построения открытого ключа

1. Выбирается *n* взаимно простых чисел $(p_1, p_2 .... p_n)$

2. Выбирается некоторое простое число *q*, такое что *q-1* раскладывается на маленькие простые множители так, что:

$$q > \prod_{i=1}^{n} p_i$$

3. Выбирается некоторый примитивный корень $b(\mathrm{mod}\ q)$.

4. Затем находятся такие $a_i (1 < a_i < q-1)$, что $p_i = b^{a_i} (\mathrm{mod}\, q)$: a_i –

дискретный логарифм от по основанию $b(\mathrm{mod}\ q)$.

Зашифрованное сообщение будет выглядеть так:

$$S=\sum_{i=1}^{n} x_i \cdot a_i \ ,$$

Где x_i – n-битный вектор исходного сообщения.

Дешифрование происходит следующим образом:

$$S'=b^S(\mathrm{mod}\ q),$$

так как $b^S=b^{\sum x_i a_i} = \prod b^{x_i a_i} = \prod b_i^{x_i}$ $(\mathrm{mod}\ q)$. Последнее равенство – результат условия.

Закрытым ключом является (b,q), открытый ключ – a_i. Затем можно легко найти соответствующий x, начинающийся с S', используя тот факт, что p_i числа взаимно простые. Это важно, так как в общем случае задача о произведении подмножеств NP-полная.

Для рюкзака Мории-Касахара в настоящее время не существует известного эффективного способа взлома.

2. РЮКЗАЧНЫЕ КРИПТОСИСТЕМЫ НА ОСНОВЕ ПРЯМОГО ПРОИЗВЕДЕНИЯ ГРУПП

2.1 Модификация рюкзачных криптосистем с использованием конечных групп

Пусть G – группа, $g_1, g_2, \ldots g_n$ – её элементы и $m_i=|g_i|$. Данную систему элементов будем называть рюкзачной системой, если для любого вектора$(x_1,x_2,\ldots,x_n)$, $x_i \in Z_{m_i}$ выполняются следующие условия:

1) элемент $g = g_1^{x_1} \cdot g_2^{x_2} \cdot \ldots \cdot g_n^{x_n}$ имеет единственное представление в указанном виде;
2) существует эффективный алгоритм, позволяющий по данному элементу g находить вектор $(x_1,x_2,\ldots,x_n)$, $x_i \in Z_{m_i}$, для которого выполняется равенство $g = g_1^{x_1} \cdot g_2^{x_2} \cdot \ldots \cdot g_n^{x_n}$.

Набор $(x_1,x_2,\ldots,x_n)$, найденный по элементу g будем называть факторизацией элемента g по системе $g_1, g_2, \ldots g_n$.

Рассмотрим рюкзачную криптосистему, основанную на вычислениях в данной группе G.

Генерация ключей

1. Выбирается рюкзак $(g_1, g_2, \ldots g_n)$, где $(g_1, g_2, \ldots g_n)$ – рюкзачная система элементов.
2. Выбирается маскирующий изоморфизм f из группы G в группу G'.
3. Формируется рюкзак-ловушка, который и является открытым ключом.
4. Отображение f является секретным ключом.

Алгоритм шифрования

1. Открытый текст представляется в виде последовательности $(x_1,x_2,\ldots,x_n)$, $x_i \in Z_{m_i}$, $m_i=|g_i|$.

2. Каждый блок $(x_1,x_2,\ldots,x_n)$ заменяется на элемент b, вычисленный по правилу: $b = b_1^{x_1} \cdot b_2^{x_2} \cdot \ldots \cdot b_n^{x_n}$

Алгоритм дешифровки

1. Находится исходный рюкзак $(g_1, g_{2,\ldots}g_n)=f^1(b_1, b_{2,\ldots}b_n)$
2. Для элемента b шифр текста вычисляется элемент $g=f^1(b)$.
3. Для вычисленного g решается задача о рюкзаке$(g_1, g_{2,\ldots}g_n)$ и находится блок открытого текста $(x_1, x_{2,\ldots}x_n)$.

Для удобства вычислений элементы $(g_1, g_{2,\ldots}g_n)$ можно выбрать так, чтобы они имели одинаковый порядок m. В этом случае исходный текст можно считать последовательностью элементов из Z_m, и при шифровании разбивать его на блоки длины n.

Пример 2.1.1 Рассмотрим общую линейную группу $G=GL_3(7)$ и выберем в ней элементы a,b и c:

$$a=\begin{pmatrix} 4 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 5 \end{pmatrix},\ b=\begin{pmatrix} 4 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 5 \end{pmatrix},\ c=\begin{pmatrix} 4 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 5 \end{pmatrix}.$$

Заметим, что $\langle a,b,c\rangle$ является прямым произведением подгрупп $\langle a\rangle$, $\langle b\rangle$ и $\langle c\rangle$, каждая из которых имеет порядок 6. Кроме того, по виду элемента $g=a^k b^l c^m$ набор(k,l,m) элементов из Z_6 легко восстанавливается. Следовательно, условия, накладываемые на элементы $g_1, g_{2,\ldots}g_n$, будут выполняться.

В качестве маскирующего изоморфизма рассмотрим сопряжение посредством элемента x:

$$x=\begin{pmatrix} 4 & 2 & 1 \\ 5 & 1 & 5 \\ 1 & 3 & 2 \end{pmatrix}.\ \text{Заметим, что } x^{-1}=\begin{pmatrix} 1 & 6 & 2 \\ 2 & 7 & 6 \\ 0 & 4 & 1 \end{pmatrix}.$$

Вычислим набор , который будет являться открытым ключом:

$$a^x=x^{-1}ax=\begin{pmatrix}3&0&4\\2&0&1\\5&3&6\end{pmatrix},\ b^x=x^{-1}bx=\begin{pmatrix}3&0&5\\6&1&3\\1&6&4\end{pmatrix},\ c^x=x^{-1}cx=\begin{pmatrix}3&1&4\\2&2&4\\3&5&0\end{pmatrix}.$$

Зашифруем в данной системе сообщение (3,4,1):

$$(3,4,1)\rightarrow g=(a^x)^2b^x(c^x)^4=\begin{pmatrix}1&2&1\\4&3&5\\4&2&4\end{pmatrix}.$$

Для дешифровки сообщения найдем элемент :

$$y=xg\,x^{-1}=\begin{pmatrix}1&0&0\\0&4&0\\0&0&1\end{pmatrix}$$

Поэтому,$g=a^2b^1c^4$ и открытый текст имеет вид (3,4,1)

2.2. Рюкзачная криптосистема на основе прямого произведения циклических групп

Определение 2.2.1. Пусть даны группа G и две ее подгруппы G_1 и G_2, причем выполнены следующие условия:

1) G_1 и G_2 являются нормальными подгруппами группы G
2) ПересечениеG_1 и G_2 состоит только из единицы e
3) Каждый элемент группы Gможет быть представлен в виде произведения a_1a_2, $a_1\in G_1$, $a_2\in G_2$.

Тогда группаG называется прямым произведением своих подгрупп и записывается как $G= G_1\times G_2$. Представление группы G в виде $G= G_1\times G_2$ можно считать факторизацией группы G.

Определение 2.2.2. Группа, порожденная одним элементом, называется *циклической.*

Пример 2.2.1. ГруппыZ, Z_n являются циклическими группами. Этими группами исчерпываются, с точностью до изоморфизма, все циклические группы.

Циклическая группа порядка mn, где m и n взаимно простые, разлагается в прямое произведение подгрупп порядков m,n.

Пусть $G=G_1\times G_2\times\ldots\times G_n$. Тогда любой элемент $g\in G$ однозначно представляется в виде $g=g_1\cdot g_2\cdot\ldots\cdot g_n$, где $g_i\in G_i$.

Заметим, что G_i – циклическая группа и $G_i=\langle g_i\rangle$,$m_i=|g_i|$, то любой элемент $g=G_1\times G_2\times\ldots\times G_n$однозначно представляется в $g=g_1^{k_1}\times g_2^{k_2}\times\ldots g_n^{k_n}$, где$k_i\in Z_{m_i}$.

Рассмотрим следующую рюкзачную криптосистему, основанную на вычислениях в прямом произведении циклических подгрупп группы G.

Пусть$G=GL_n(q)$. Рассмотрим в Gсистему элементов:

$$g_1=\begin{pmatrix}\alpha_1 & 0 & \ldots & \ldots & 0\\ 0 & 1 & 0 & \ldots & 0\\ \ldots & \ldots & \ldots & \ldots & \ldots\\ 0 & \ldots & 0 & 1 & 0\\ 0 & \ldots & \ldots & 0 & 1\end{pmatrix},\ g_2=\begin{pmatrix}1 & 0 & \ldots & \ldots & 0\\ 0 & \alpha_2 & 0 & \ldots & 0\\ \ldots & \ldots & \ldots & \ldots & \ldots\\ 0 & \ldots & 0 & 1 & 0\\ 0 & \ldots & \ldots & 0 & 1\end{pmatrix},\ldots,\ g_n=\begin{pmatrix}1 & 0 & \ldots & \ldots & 0\\ 0 & 1 & 0 & \ldots & 0\\ \ldots & \ldots & \ldots & \ldots & \ldots\\ 0 & \ldots & 0 & 1 & 0\\ 0 & \ldots & \ldots & 0 & \alpha_n\end{pmatrix}$$

Порядок элемента α_1 в поле $F_q:|\alpha_1|=k_1$, элемента α_2 в поле $F_q:|\alpha_2|=k_2$, …, α_n в поле $F_q:|\alpha_n|=k_n$.

Для удобства вычислений можно выбрать $k_1=k_2=\ldots=k_n$.

Тогда подгруппа, порожденная элементами $\langle g_1,g_2,\ldots,g_n\rangle$, будет являться прямым произведением подгрупп $\langle g_1,g_2,\ldots,g_n\rangle=\langle g_1\rangle\times\langle g_2\rangle\times\ldots\times\langle g_n\rangle$. Заметим , что для любого элемента $g\in\langle g_1,g_2,\ldots,g_n\rangle$ существует простой алгоритм нахождения таких $a_1,a_2,\ldots,a_n$, где $a_i\in Z_{k_i}$, что $g=g_1^{a_1}\times g_2^{a_2}\times\ldots g_n^{a_n}$ (при условии, что задача о нахождении дискретного логарифма в поле F_qлегко разрешима).

Таким образом, система элементов удовлетворяет условиям, описанным в модификации , рассмотренной выше, и является рюкзачной системой.

В качестве маскирующего изоморфизма можно рассмотреть сопряжение с помощью специально подобранного элемента $x \in GL_n(q)$ такого, что g_i^x – не диагональная матрица.

Пример 2.2.2. Рассмотрим циклическую группу в Z_5 и выберем в ней элементы g_1, g_2, g_3 и g_4:

$$g_1 = \begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \; g_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$g_3 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 3 \end{pmatrix}, g_4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix}.$$

Заметим, что $\langle g_1, g_2, g_3, g_4 \rangle$ является прямым произведением подгрупп $\langle g_1 \rangle \times \langle g_2 \rangle \times \langle g_3 \rangle \times \langle g_4 \rangle$, каждая из которых имеет порядок 4.

В качестве маскирующего элемента рассмотрим сопряжение посредством элемента x. Элемент x нужно выбирать так, чтобы элементы g_i^x не имели блочно-диагональный вид

$$x = \begin{pmatrix} 4 & 0 & 1 & 2 \\ 1 & 1 & 3 & 0 \\ 1 & 2 & 3 & 0 \\ 1 & 3 & 2 & 3 \end{pmatrix}. \text{ Заметим, что } x^{-1} = \begin{pmatrix} 4 & 4 & 2 & 4 \\ 0 & 4 & 1 & 0 \\ 2 & 1 & 4 & 2 \\ 4 & 4 & 4 & 2 \end{pmatrix}.$$

Вычислим набор $\langle g_1^x, g_2^x, g_3^x, g_4^x \rangle$, который будет являться открытым ключом:

$$g_1^x = \begin{pmatrix} 2 & 1 & 1 & 2 \\ 2 & 2 & 3 & 2 \\ 2 & 2 & 4 & 2 \\ 1 & 3 & 4 & 1 \end{pmatrix}, \; g_2^x = \begin{pmatrix} 0 & 4 & 2 & 0 \\ 4 & 0 & 2 & 0 \\ 1 & 1 & 4 & 0 \\ 0 & 2 & 4 & 4 \end{pmatrix},$$

$$g_3^x = \begin{pmatrix} 4 & 0 & 0 & 2 \\ 2 & 0 & 1 & 0 \\ 0 & 2 & 4 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}, g_4^x = \begin{pmatrix} 3 & 1 & 4 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 3 & 3 & 3 \\ 2 & 1 & 4 & 2 \end{pmatrix}$$

Зашифруем в данной системе сообщение (1,3,2,4):

$$(1,3,2,4) \rightarrow S = (g_1^x)^1 (g_2^x)^3 (g_3^x)^2 (g_4^x)^4 = \begin{pmatrix} 3 & 4 & 1 & 1 \\ 1 & 3 & 3 & 3 \\ 2 & 0 & 0 & 4 \\ 0 & 4 & 0 & 4 \end{pmatrix}$$

Для дешифровки сообщенияS найдем элементy:

$$y = xgx^{-1} = \begin{pmatrix} 4 & 0 & 0 & 2 \\ 2 & 0 & 1 & 0 \\ 0 & 2 & 4 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

Получаем, что открытый текст имеет вид (1,3,2,4).

2.3. Рюкзачная криптосистема на основе сплетения групп

Пусть A,B – группы. Тогда обозначим через fun (B,A) прямое произведение изоморфных копий группы A, индексированных элементами группы B. Таким образом получаем, что fun(B,A) –подгруппа функций с конечными носителями.

Расширение группы fun(B,A) посредством отображения B→ Aut (fun(B,A)) называется прямым сплетением группы A с группой B.

2.3.1. Алгоритм шифрования

1. Выбираем p – простое число;
2. Подбираем $n \in N$ таким образом, чтобы результатом частного $(p-1)/n$ являлось целое число (числа). Для повышения эффективности алгоритма нам необходимо выбрать наибольшее из подбираемых чисел n;
3. Отметим, что $\alpha \in Z_{p_i}, |\alpha| = n;$;
4. Составляем матриц размерности $n \times n$, где каждая матрица из

$g_1,...g_n$ является результатом сплетения циклических групп

$$g_1 = \begin{pmatrix} \alpha & 0 & ... & 0 & 0 \\ 0 & 1 & ... & 0 & 0 \\ ... & ... & ... & ... & ... \\ 0 & 0 & ... & 1 & 0 \\ 0 & 0 & ... & 0 & 1 \end{pmatrix}, ..., g_n = \begin{pmatrix} 1 & 0 & ... & 0 & 0 \\ 0 & 1 & ... & 0 & 0 \\ ... & ... & ... & ... & ... \\ 0 & 0 & ... & 1 & 0 \\ 0 & 0 & ... & 0 & \alpha \end{pmatrix}$$

5. Задаем и определяем перестановочную матрицу b размерности $n \times n$ (b – осуществляет перестановку элементов матриц из Z_{p-1}), а также находим b^{-1};
6. Рассчитаем матрицы $g_1^b, ..., g_n^b$ путем сопряжения матрицы b и b^{-1} с матрицами $g_1, ..., g_n$ размерности $n \times n$ поэлементно по следующему соотношению

$$g_i^b = bg_ib^{-1};$$

7. Зашифруем необходимое сообщение в полученной системе.

2.3.2. Алгоритм дешифрования

1. Находится элемент γ по соотношению

$$\gamma = bgb^{-1}.$$

Пример 2.3.2.

Зададим $p = 5$ и будем рассматривать циклическую группу в Z_5. Подбираем так, чтобы оно было наибольшим делителем $p - 1 = 4$. Соответственно $n = 4$, а следовательно $\alpha = |n| = 4$

Далее задаем 4 матрицы размерности 4×4 с $\alpha = 4$:

$$g_1 = \begin{pmatrix} 4 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, g_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, g_3 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, g_4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 4 \end{pmatrix}.$$

Определяем перестановочную матрицу размерности 4×4:

$$b=\begin{pmatrix}1&0&0&0\\0&2&0&0\\0&0&3&0\\0&0&0&4\end{pmatrix}.$$

Матрица b задает перестановку элементов (сдвиг элемента в строке). Далее рассчитываем матрицу b^{-1}:

$$b^{-1}=\begin{pmatrix}1&0&0&0\\0&3&0&0\\0&0&2&0\\0&0&0&4\end{pmatrix}.$$

Рассчитаем матрицы g_1^b, g_2^b, g_3^b и g_4^b путем сопряжения матрицы с матрицами $g_1,\ldots, g_n$ поэлементно по модулю 5:

$$g_1^b=bg_1b^{-1}=\begin{pmatrix}1&0&0&0\\0&2&0&0\\0&0&3&0\\0&0&0&4\end{pmatrix}\cdot\begin{pmatrix}4&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix}\cdot\begin{pmatrix}1&0&0&0\\0&3&0&0\\0&0&2&0\\0&0&0&4\end{pmatrix}=\begin{pmatrix}4&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix},$$

$$g_2^b=bg_2b^{-1}=\begin{pmatrix}1&0&0&0\\0&2&0&0\\0&0&3&0\\0&0&0&4\end{pmatrix}\cdot\begin{pmatrix}1&0&0&0\\0&4&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix}\cdot\begin{pmatrix}1&0&0&0\\0&3&0&0\\0&0&2&0\\0&0&0&4\end{pmatrix}=\begin{pmatrix}1&0&0&0\\0&4&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix},$$

$$g_3^b=bg_3b^{-1}=\begin{pmatrix}1&0&0&0\\0&2&0&0\\0&0&3&0\\0&0&0&4\end{pmatrix}\cdot\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&4&0\\0&0&0&1\end{pmatrix}\cdot\begin{pmatrix}1&0&0&0\\0&3&0&0\\0&0&2&0\\0&0&0&4\end{pmatrix}=\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&4&0\\0&0&0&1\end{pmatrix},$$

$$g_4^b=bg_4b^{-1}=\begin{pmatrix}1&0&0&0\\0&2&0&0\\0&0&3&0\\0&0&0&4\end{pmatrix}\cdot\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&4\end{pmatrix}\cdot\begin{pmatrix}1&0&0&0\\0&3&0&0\\0&0&2&0\\0&0&0&4\end{pmatrix}=\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&4\end{pmatrix}.$$

Таким образом

$$g_1^b=\begin{pmatrix}4&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix},\ g_2^b=\begin{pmatrix}1&0&0&0\\0&4&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix},\ g_3^b=\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&4&0\\0&0&0&1\end{pmatrix},\ g_4^b=\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&4\end{pmatrix}.$$

Зашифруем в данной системе сообщение (1,1,3,2):

$$(1,1,3,2) \to S = (g_1^b)^1 (g_2^b)^1 (g_3^b)^3 (g_4^b)^2 = \begin{pmatrix} 4 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Для дешифровки сообщения S найдем элемент γ:

$$\gamma = bgb^{-1} = \begin{pmatrix} 3 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 4 \end{pmatrix}.$$

Поэтому, и открытый текст имеет вид (1, 1, 3, 2).

2.4. Криптостойкость рюкзачной системы на основе прямого произведения циклических групп

Для оценки криптостойкости данной системы рассмотрим атаку методом полного перебора.

Пусть ключ .попытаемся подобрать , такое, что $e_i^x = t_i$, где t_i – диагональная матрица. Для расчета вероятности угадывания элемента xоценим количество таких x.Рассчитаем общий порядок $GL_n(q)$:

$$|GL_n(q)| = (q^n - 1)(q^n - q)(q^n - q^2)...(q^n - q^{n-1})$$

При произвольном выборе $x \in GL_n(q)$ вероятность того, что ключ будет диагонализован рассчитывается как:

$$P(A) = \frac{n!(q-1)^n}{(q^n - 1)(q^n - q)(q^n - q^2)...(q^n - q^{n-1})}$$

или

$$P(A) = \frac{n!(q-1)^n}{|GL_n(q)|}$$

Рассмотрим порядок $GL_4(5)$:

$|GL_4(5)| = (5^4 - 1)(5^4 - 5)(5^4 - 5^2)(5^4 - 5^3) = 1{,}2 \cdot 10^{11}$, количество способов подбора базиса равно $4!(5-1)^4 = 6144$, и тогда атака методом перебора даст

возможностьвзломать криптосистему с вероятностью

$$P(A) = \frac{4!(5-1)^4}{(5^4-1)(5^4-5)(5^4-5^2)(5^4-5^3)} = 5{,}2 \cdot 10^{-8}$$

3. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

В ходе выполнения диссертационной работы была разработана программа в учебно-демонстрационных целях, реализующая шифрование на основе задачи об укладке рюкзака, использующей сплетение групп.

Оболочка программы была выполнена на языке C#, для реализации алгоритма шифрования/дешифрования использовалась система компьютерной алгебры GAP.

3.1 Описание системы компьютерной алгебры GAP

Разработка системы компьютерной алгебры *GAP,* название которой расшифровывается как "*Groups, AlgorithmsandProgramming*", была начата в 1986 г. в городе Аахен, Германия. В 1997 г. центр координации разработки и технической поддержки пользователей переместился в Университет города Сент-Эндрюс, Шотландия. Основные центры разработки системы находятся в университетах города Сент-Эндрюс (Шотландия), городов Аахен, Брауншвейг (Германия) и Университете штата Колорадо (США).

GAPявляется свободно распространяемой, открытой и расширяемой системой. Она распространяется в соответствии с GNUPublicLicense. Система поставляется вместе с исходными текстами, которые написаны на двух языках: ядро системы написано на C, а библиотека функций – на специальном языке, также называемом GAP, который по синтаксису напоминает Pascal, однако является объектно-ориентированным языком. Пользователи могут создавать свои собственные программы на этом языке, и здесь исходные тексты являются незаменимым наглядным пособием.

Система GAPбыла задумана как инструмент комбинаторной теории групп – раздела алгебры, изучающего группы, заданные порождающими элементами

и определяющими соотношениями. В дальнейшем, с выходом каждой новой версии системы сфера ее применения охватывала все новые и новые разделы алгебры. GAPдает возможность производить вычисления с гигантскими целыми и рациональными числами, допустимые значения которых ограничены только объемом доступной памяти. Система работает с конечными полями, многочленами от многих переменных, рациональными функциями, векторами и матрицами. Пользователю доступны различные комбинаторные функции, элементарные теоретико-числовые функции, разнообразные функции для работы с множествами и списками.

Среди других областей применения системы – теория графов и их автоморфизмов, теория кодирования, теория полугрупп, кристаллография, и многое другое. Существует графический интерфейс XGAP, который работает в среде Unix/Linuxи позволяет, например, графически изобразить решетку подгрупп группы.

3.2. Руководство пользователя

При запуске программы открывается главное окно (см. рис. 4.1). В меню можно выбрать файл, который необходимо зашифровать, открыть ранее сохраненные файлы, сохранить файл при необходимости.

Меню «Файл» содержит следующие подпункты:

- «Открыть файл»;
- «Выход».

Пункт выпадающего меню «Открыть файл» предназначен для открытия файла с текстом, который необходимо зашифровать.

При шифровании сообщения/текста в окно «Исходный текст» вводится сообщение или происходит вставка текста из файла (см. рис. 4.1). Далее нажимаем кнопку «Кодирование», текст переводится в двоичный код и

записывается в файл в текстовом формате, с которым далее будет работать система GAP.

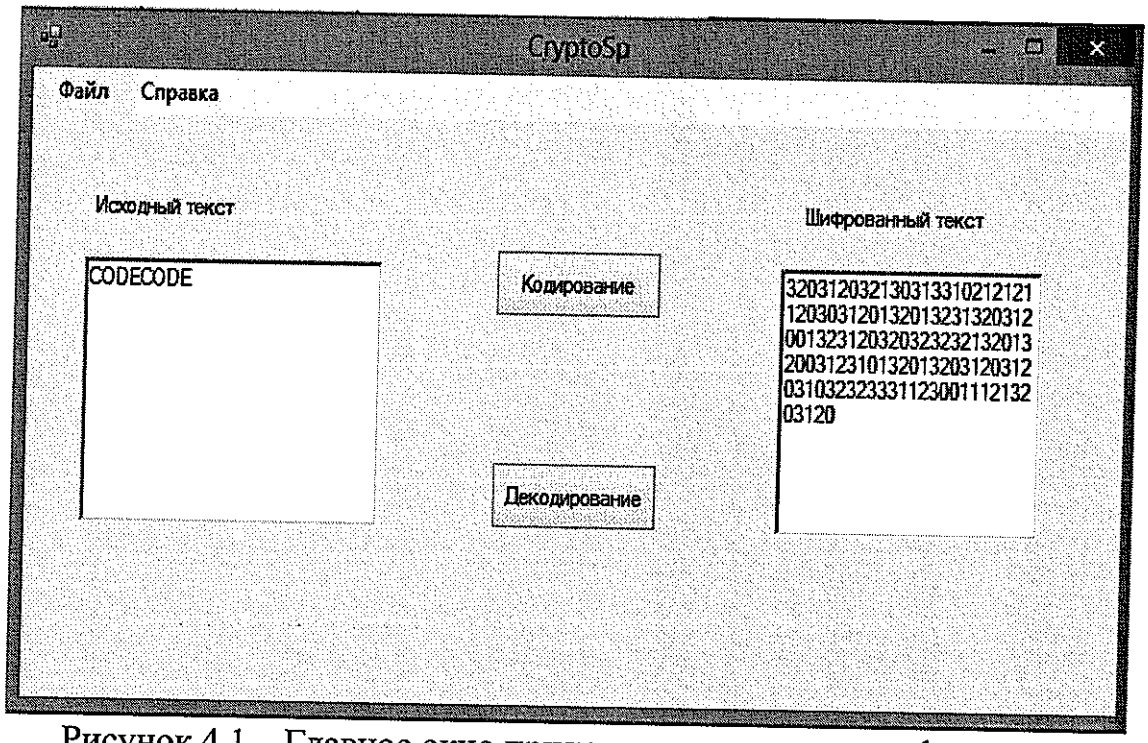

Рисунок 4.1 – Главное окно приложения в режиме шифрования

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной работы была рассмотрены криптографические системы связанные с задачами факторизации и, в частности, классическая задача об укладке рюкзака, ее применение и уже существующие модификации, описаны возможные методы модификации.

Однако в настоящее время рюкзачная схема достаточно редко используется в криптографии, в связи с тем, что существующие модификации недостаточно надежны и не используют NP-полноту задачи. Возникает необходимость модификации существующей постановки задачи, поэтому был предложен вариант применения рюкзачной схемы на группах, в частности на основе линейных групп.

Использование линейных групп, и их подгрупп специального вида, увеличивает криптостойкость шифра, поэтому был предложен алгоритм шифрования, названный рюкзачной системой, основанный на сплетении групп.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Каргаполов, М. И. Основы теории групп / М. И. Каргаполов, Ю. И. Мерзляков. – 3-е изд. – М.: Наука, 1982. – 73с.

2. Diffie, W. New Directions in Cryptography / W. Diffie, M. E. Heilman – IEEE Transactions on Information Theory ,V. Tl. 22, 1977. – P. 644–654.

3. Смарт, Н. Криптография / Н. Смарт. – М.: Техносфера, 2005. – 406 с.

4. Саломаа, А. Криптография с открытым ключом / А. Саломаа. – М.: Мир, 1995. –318 с.

5. Heilman, M.E. The Mathematics of Public-Key Cryptography / M.E. Hellman // Scientihc American – Aug 1982. – V.241, №8. – P.146-157.

6. Брассар, Ж. Современная криптология / Ж. Брассар. – М.: ПОЛИМЕД, 1999. – 281 с.

7. Харин, Ю.С. Математические основы криптологии: учебное пособие / Ю.С. Харин, В.И. Берник, Г.В. Матвеев. – Мн.: БГУ, 1999. – 319 с.

8. Henk, C. A. van Tilborg. Encyclopedia of Cryptography and Security / C.A. Henk van Tilborg. – Springer Science, 2005. – 697 p.

9. Система компьютерной алгебры GAP 4.7: методические указания / сост. А.Б. Коновалов. – Запорожье: Запорожский государственный университет, 2014. – 87 с.

10. Основы системы компьютерной алгебры GAP: методические указания / сост. Н.И. Крайнюков, Б.Ф. Мельников. – Тольятти: ТГУ, 2012. – 26 с.

ПРИЛОЖЕНИЕ

Криптостойкость рюкзачной системы на основе прямого произведения циклических групп в пакете GAP

```
Pa:=0;
Pd:=1;
#for i in [0..4] do
 # Pd:=(q^n-q^i)*Pd;
#od;
#Pa:=(Factorial(n)*(q-1 )^n)/Pd;
#Display(Pa);
for n in [1..9] do
    for q in [2.. 10] do
        fori in [O..n-l] do
        Pd:=Pd*(qAn-qAi);
        od;
Pa:=(Factorial(n)*(q-l)^n)/Pd;
DispIay(Pa);
od;
od;
```

Реализация пользовательского интерфейса в MicrosoftVisualStudioФайл

```
namespaceCryptoSp {
        using namespace System;
        using namespace System::I0;
        using namespace System::ComponentModel;
        using namespace System:Collections;
        using namespace System::Windows::Forms;
        using namespace System::Data;
        using namespace System::Drawing;
        using namespace System::Runtime::InteropServices;

        /// <summary>
        III СводкадляKnapsackCryptoIII </summary>
        public ref class KnapsackCrypto : public System::Windows::Forms::Form
        {
        public:
                KnapsackCrypto(void)
                {
                        InitializeComponent() ;
                        //
                        //TODO: добавьте код конструктора
                        //
                }
        protected:
                III <summary>
                III Освободить все используемые ресурсы.
                III </summary>
                "-KnapsackCrypto ()
                {
                        if (components)
                        {
                                delete components;
                        )
                }
        private: System: :Windows::Forms::Button'4buttonl; private: System:
        :Windows::Forms: :МепиБ^1рлmenuStripl;
        private: System::Windows::Forms::ToolStripMenuItem'4файлToolStripMenuItem;
private:
System::Windows::Forms::ToolStripMenuItem^coxpaHHTb<J>ai«iToolStripMenuItem;
        private: System: :Windows: :Forms: :ToolStripMenuItem'4 открытьФайлТоо
1StripMenuItern;

        private: System::Windows::Forms::ToolStripMenuIteir^ BUxofiToolStripMenuItem
        private : System: : Windows : : Forms : :
ToolStripMenuItem~ cnpaDKaToolStripMonuItem;
        private: System: : Windows: : Forms: :
ToolSt.ripMenuItem^noMoiubToolStripMenuItem;
        private: System::Windows::Forms::ToolStripMenuItem*
oIIporpaMMeToolStripMenuItem;
        private: System::Windows::Forms::TextBox* textBoxl;
        private: System::Windows::Forms::Label* labell; private:
```

```
System::Windows::Forms::Label* label2; private:
System::Windows::Forms::TextBox* textBox2;

protected:

private:
        /// <summary>
        III Требуется переменная конструктора.
        III </summary>
        System: :ComponentModel: :Container "components;

        #pragma region Windows Form Designer
        generated code III <summary>
        III Обязательный метод для поддержки конструктора – не изменяйте
        III содержимое данного метода при помощи редактора кода.
        /// </summary>
        voidInitializeComponent(void)
        {
                this->buttonl = (gcnew System::Windows::Forms::Button());
this->menuStr.ipl = (gcnew System: : Windows:: Forms : :MenuStrip () )
this->ct^nToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem()) ;
                ^1з->открытьФайлТоо15Гг1.рМепиТ^.ет= (gcnew
System::Windows::Forms::ToolStripMenuItem());
                this->сохранитьФайлToolStripMenuItem = (gcnew
System::Windows::Forms::ToolStripMenuItem());
                this->BbixoflToolStripMenuItem = (gcnew
System::Windows::Forms::ToolStripMenuItem());
                this->cnpaBKaToolStripMenuItem = (gcnew
System::Windows::Forms::ToolStripMenuItem());
                this->nOMOuibToolStripMenuItem = (gcnew
System::Windows::Forms::ToolStripMenuItem());
                this->onporpaMMeToolStripMenuItem = (gcnew
System::Windows::Forms::ToolStripMenuItem());
                this->textBoxl = (gcnew System::Windows::Forms::TextBox()) ;
                this->labell = (gcnew System::Windows::Forms::Label ()); this->label2
                = (gcnew System::Windows::Forms::Label()); this->textBox2 =
                (gcnew' System::Windows::Forms::TextBox());
                this->menuStripl->SuspendLayout(); this->SuspendLayout();
                //
                // buttonl
                //
                this->buttonl->Font = (gcnew System::Drawing::Font(L"Microsoft
Sans Serif", 10.2F, System::Drawing::FontStyle::Bold,
            System::Drawing::GraphicsUnit::Point,
                        static_cast<System::Byte>(204)));
                this->buttonl->Location = System::Drawing::Point (433, 86);
                this->buttonl->Name = L"buttonl";
                this->buttonl->Size = System::Drawing::Size(138, 56);
                this->buttonl->TabIndex = 0;
                this->buttonl->Text = L"Encryption/Decryption";
this->buttonl->UseVisualStyleBackColor = true; this->buttonl->Click += gcnew
System::EventHandler(this, SKnapsackCrypto::buttonl_Click) ;
                //
                // menuStripl //
```

```
this->menuStripl->Items->AddRange(gcnew cli::array<
System: : Windows :: Forms : : ToolStripItem" >(2){
        this->файлToolStripMenuItem,
            this->cnpaBKaToolStripMenuItem
}> ;
this->menuStripl->Location = System::Drawing::Point(0, 0);
this->menuStripl->Name = L"menuStripl";
this->menuStripl->Size = System::Drawing::Size (1016, 28);
this->menuStripl->TabIndex = 1; this->menuSt.ripl->Text =
L"menuStripl";
//
// <|>aiinToolStripMenuItem
//
this->файлToolStripMenuItem->DropDownItems->AddRange(gcnew
cli::array< System::Windows::Forms::ToolStripItem^>(3)          (
        ^1з->открытьФайлТоо18^1рМепи^ет,
            this->сохранитьФайлToolStripMenuItem, this-
>BbixoflToolStripMenuItem
});
this->файлToolStripMenuItem->Name =
L"файлToolStripMenuItem"; this->файлToolStripMenuItem->Size =
System::Drawing::Size(57,
24) ;
this->файлToolStripMenuItem->Text = Ь"Файл";
//
// открытьФайлToolStripMenuItem
//
this->открытьФайлToolStripMenuItem->Margin =
System::Windows::Forms::Padding(0, 0, 0, 7);
this->открытьФайлToolStripMenuItem->Name = Ь"открытьФайлТоо
15Гг1рМепи15ет";
this->откры'гьФайлToolStripMenuItem->Size =
System::Drawing::Size(191, 24);
this->ОТКрНТb0a^ToolStripMenuItem->Text = Ь"Открытьфайл";
this->открытьФайлToolStripMenuItem->ToolTipText = Ь"Открытьфайлсисходнымтекст
ом";
//
// сохранитьФайлТоо13^1рМепи^ет
//
ГЫз->сохранитьФайлТоо15^1рМепи1^ет->Ыате = L"сохранитьФа
йлToolStripMenuItem";
^1з->сохранитьФайлТоо15^1рМепи^ет->31ге =
System::Drawing::Size(191, 24);
```

```
this->coxpaHHTb®a£urToolStripMenuItem->Text = Ь"Сохранитьфа
йл";
//
// BuxoflToolStripMenuItem
11
this->BbixoflToolStripMenuItem->Margin =
System;:Windows::Forms::Padding(0, 7, 0, 0) ;
this->BbixoflToolStripMenuItem->Name =
L"BbixoflToolStripMenuItem"; this->BbixoflToolStripMenuItem->Size
= System:;Drawing::Size(191,
24) ;
this->BbixoflToolStripMenuItem->Text = Ь"Выход";
this->BbixoflToolStripMenuItem->Click += gcnew System::EventHandler(this,
&KnapsackCrypto::BbixoflToolStripMenuItem_Click);
//
// cnpaBKaToolStripMenuItem
//
this->cnpaBKaToolStripMenuItem->DropDownItems->AddRange(gc
new cli::array< System::Windows::Forms::ToolStripItem^>(2)    {
        this->noMombToolstripMenuItem,
            this->onporpaMMeToolStripMenuItem
}> ;
this->cnpaBKaToolStripMenuItem->Na
me L"cnpaBKaToolStripMenuItem";
this->cnpaBKaToolSt.ripMenuItem-->Siz    System::Drawing::Size(79,
e
24) ;
this->cnpaBKaToolStripMenuItern->Text = Ь"Справка";
//
// noMombToo.lStripMenuItem
// this->noMOiubToolStripMenuItem->Name =
I/'noMOiubToolStripMenuItem";
this->noMombToolStripMenuItem->Size =
System::Drawing::Size(173,
24);
t.his->nOMOuibToolStripMenuItem->Text = Ь"Помсщь";
this->noMombToolStripMenuItem->Click += gcnew System::EventHandler(this,
SKnapsackCrypto::noMOi«bToolStripMenuItem_Click); //
// ollporpaMMeToolStripMenuItem
//
this->oriporpaMMeToolStripMeni:Item->Name = L"
onporpaMMeToolStripMenuItem";
this->onporpaMMeToolStripMenuItem->Size =
System::Drawing::Size(173, 24);
this-->onporpaMMeToolStripMenuItem->Text = I/'спрограмме";
this->onporpaMMeToolStripMenuItem->Click += gcnew System::EventHandler(this,
SKnapsackCrypto::ollporpaMMeToolStripMenuItem_Click); //
// textBoxl
// this->textBoxl->Location = System::Drawing::Point(16, 62);
this->textBoxl->Margin = System::Windows::Forms::Padding(4);
this->textBoxl->Multiline = true; this->textBoxl->Name =
L"textBoxl"; t.his->textBoxl->ScrollBars =
System::Windows::Forms::ScrollBars::Vertical;
```

```
this->textBoxl->Size = System::Drawing::Size (382, 245);
this->textBoxl->TabIndex = 2;
11
// labell
//
this->labell->AutoSize = true;
this->labell->Font = (gcnew System::Drawing::Font(L"Microsoft Sans
Serif", 10.2F, System::Drawing::FontStyle::Regular,
System::Drawing::GraphicsUnit::Point,
        static cast<System::Byte>(204)));
this->labell->Location ;= System::Drawing::Point (16, 36);
this->labell->Name = L"labell";
this->labell->Size = System::Drawing::Size (144, 20);
this->labell->Tab!ndex = 3; this->labell->Text = Ь"Исходныйтекст";
//
// label2
//
this->label2->AutoSize = true;
this~>label2->Font = (gcnew System::Drawing::Font(L"Microsoft
Sans Serif", 10.2F, System::Drawing::FontStyle::Regular,
System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(204)));
th.is->label2->Location = System: : Drawing:: Point (617, 36);
this->label2->Name = L"label2";
this->label2->Size = System::Drawing::Size(206, 20);
this->label2->TabIndex = 4;
this->label2->Text = L"Зашифрованныйтекст";
//
// textBox2
//
this->textBox2->Location = System: :Drawing: .'Point (621, 62);
this->textBox2->Margin = System::Windows::Forms::Padding(4);
this->textBox2->Multiline = true; this->textBox2->Name =
L"textBox2"; this->textBox2->ReadOnly = true;
this->textBox2->ScrollBars =
System: :Windows: :Forms: :ScrollBars: :Vert.ical;
this->textBox2->Size = System::Drawing::Size(382, 245);
this->textBox2->TabIndex = 3;
//
// KnapsackCrypto
//
this->AutoScaleDimensions = System::Drawing::SizeF(8, 16);
this->AutoScaleMode =
System::Windows::Forms::AutoScaleMode::Font;
this->ClientSize = System::Drawing::Size(1016, 344);
this->Controls->Add(this->textBox2);
this->Controls->Add(this->label2); this->Controls->Add (th.is->labell)
; thi.s->Controls->Add (this->textBoxl) ;
this->Controls->Add(this->buttonl); this->Controls->Add
(this->menuStr.ipl) ; this->MainMenuStrip = thi.s->menuStripl;
this->Name =
L"KnapsackCrypto";
this->StartPosition =
```