КАЗАНСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ

А.А. КУРБАНГАЛЕЕВ, Ш.Т. ИШМУХАМЕТОВ, Р.Г. РУБЦОВА

ЛАБОРАТОРНЫЕ РАБОТЫ ПО ОСНОВАМ КРИПТОГРАФИИ

Учебно-методическое пособие

КАЗАНЬ
2025

УДК 512.624
ББК 22.144

Рекомендовано к изданию
Учебно-методической комиссией ИВМиИТ
(протокол № 7 от 18.04.2025 года)

Рецензент:
кандидат физико-математических наук, доцент **Ахтямов Р.Б.**

Курбангалеев А.А.
Лабораторные работы по основам криптографии: учебно-методическое пособие / А.А. Курбангалеев, Ш.Т. Ишмухаметов, Р.Г. Рубцова. – Казань: Казан. ун-т. 2025. – 76 с.

Учебное пособие представляет собой введение в раздел «Основные криптографические алгоритмы» курса «Теоретические основы информационной безопасности» и содержит базовые сведения из этого раздела дискретной математики. Пособие снабжено описанием ряда алгоритмов и учебными примерами и заданиями для самостоятельной подготовки по курсу.

УДК 512.624
ББК 22.144

СОДЕРЖАНИЕ

ЛАБОРАТОРНАЯ РАБОТА № 1. ИСТОРИЧЕСКИЕ ШИФРЫ

Цель работы: изучение исторических шифров

1.1. Теоретический материал

В данной работе мы осуществим подробный анализ основных исторических методов криптографии, которые были разработаны и применялись с целью обеспечения защиты информации от несанкционированного доступа. Особое внимание будет уделено рассмотрению тех техник и подходов, которые использовались в различные исторические периоды для того, чтобы предотвратить получение доступа к конфиденциальной информации лицами, не имеющими на это разрешения. Эта задача особенно актуальна в контексте повышения угроз информационной безопасности и необходимости защиты персональных данных от неавторизированных вторжений.

1.2. Шифр Цезаря

Рассмотрим историю шифров. Первые наскальные рисунки уже можно отнести к шифрам. Ключом к ним были знания об окружающем мире. Далее с развитием общества появлялось всё больше потребности в шифровании. Например, если рассмотреть время, когда жил Юлий Цезарь и была ожесточённая борьба с галлами, то римляне того времени пользовались шифром, названным в честь Юлия Цезаря. Его алгоритм был прост: существовал диск (каменный) с нанесенным на него алфавитом, он входил в другой диск также с алфавитом, так что диски могли друг относительно друга смещаться. То есть, например букве A сопоставлялась буква со смещением на сколько-то позиций, это и был секретный ключ. Например, если ключ был равен трём, то A→D, B→E, C→F и т.д. (табл. 1, табл. 2):

Таблица 1

Шифр Цезаря (английский алфавит)

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Таблица 2

Шифр Цезаря (русский алфавит)

А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В

Шифр Цезаря относится к категории шифрования, известных как подстановочные или шифры простой замены. В этом методе каждый символ, цифра или их сочетание замещает определенную букву из алфавита. Приведем примеры использования шифра Цезаря.

Пример 1.1

ЫЛЧУ	УГДСХГЗЧ	ФС	ФЖЕЛЁСП	ХУЛ
Шифр	работает	со	сдвигом	три

1.3. Тарабарская грамота

Использование секретной письменности в России впервые зафиксировано в XIII столетии и получило название «тарабарская грамота». В рамках данной системы происходит замена согласных букв согласно определенной схеме (табл. 3):

Таблица 3

Тарабарская грамота

Б	В	Г	Д	Ж	З	К	Л	М	Н
Щ	Ш	Ч	Ц	Х	Ф	Т	С	Р	П

При шифровании буквы, расположенные на одной вертикали, переходят одна в другую. Остальные буквы остаются без изменения.

Пример 1.2.

ВФЛ	ЙКХК	ВФЛ	ФДКТК
Век	живи	век	учись

1.4. Таблица Виженера

Для того, чтобы эффективно зашифровать конкретное сообщение, используется определенная методика (табл. 4).

Таблица 4

Таблица Виженера

а	**б**	**в**	**г**	**д**	**е**	**ё**	**ж**	**з**	**и**	**й**	**к**	**л**	**м**	**н**	**о**	**п**	**р**	**с**	**т**	**у**	**ф**	**х**	**ц**	**ч**	**ш**	**щ**	**ъ**	**ы**	**ь**	**э**	**ю**	**я**
б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а
в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б
г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в
д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г
е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д
ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е
ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё
з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж
и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з
й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и
к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й
л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к
м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л
н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м
о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н
п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о

Продолжение табл. 4

р	с	т	у	ф	х	ц	ч	ш	ш	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п
с	т	у	ф	х	ц	ч	ш	ш	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р
т	у	ф	х	ц	ч	ш	ш	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с
у	ф	х	ц	ч	ш	ш	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т
ф	х	ц	ч	ш	ш	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у
х	ц	ч	ш	ш	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф
ц	ч	ш	ш	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х
ч	ш	ш	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц
ш	ш	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч
ш	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш
ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ
ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	ш	ъ
ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	ш	ъ	ы
э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	ш	ъ	ы	ь
ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	ш	ъ	ы	ь	э
я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	ш	ъ	ы	ь	э	ю

Сначала выбирается ключевое слово, которое размещается в виде повторяющейся последовательности над каждым символом исходного сообщения, создавая тем самым основу для дальнейших шифровальных операций. Далее, для создания зашифрованного текста применяется следующий подход: начиная с первой буквы ключевого слова, идентифицируют соответствующий символ в верхней строке (вертикальной) таблицы алфавита. Затем находят символ исходного сообщения, который располагается на

горизонтальной линии (слева) указанной таблицы. Зашифрованная буква определяется на основе точки пересечения соответствующего столбца и строки в этой алфавитной таблице.

Этот метод позволяет превратить исходное сообщение в зашифрованный текст путем систематического применения описанной схемы ко всем символам сообщения.

Пример 1.3 (табл. 5)

Ключевое слово – солнце.

Шифруемый текст – гори ярко.

Таблица 5

Методика шифрования шифром Виженера

С	О	Л	Н	Ц	Е	С	О
Г	О	Р	И	Я	Р	К	О
Ф	Э	Ь	Ц	Х	Х	Ь	Э

1.5. Постолбцовая транспозиция

Шифрование, известное под названием «маршрутная транспозиция» и его вариант, которую можно назвать «транспозицией по столбцам», относится к обширной категории криптографических методов, принимающих основу в принципах перестановки символов. В этом методе, сообщение сначала записывается в форме построчного заполнения прямоугольника, имеющего размеры n x m. Далее, для получения зашифрованного текста, необходимо аккуратно переписать символы, но в данном случае следует осуществлять это согласно новому порядку, определяемому последовательностью столбцов прямоугольника. Таким образом, шифрование достигается за счёт изменения исходного порядка букв в сообщении на основе заданной схемы распределения по столбцам.

Пример 1.4 (табл. 6).

Шифруемый текст – «Друзья познаются в беде».

Получаем следующую таблицу 5 х 4:

Таблица 6

Постолбцовая транспозиция

1	**2**	**3**	**4**	**5**
Д	р	у	з	ь
Я	п	о	з	н
А	ю	т	с	я
В	б	е	д	е

Для шифрования выпишем буквы по столбцам:
дяав рпюб уоте ззсд ьняе.

1.6. Шифр Вернама

Шифр Вернама, также широко известный под наименованием «одноразовый блокнот», является одним из методов симметричного криптографического шифрования, который обеспечивает, согласно теоретическим расчетам, абсолютную невозможность взлома защищенных сообщений. Ключевой аспект данного шифра заключается в необходимости использования ключа, длина которого строго соответствует длине шифруемого текста, что является одним из его основных требований. Этот ключ представляет собой уникальную и полностью случайную последовательность символов, которая применяется строго однажды и никогда не повторяется для последующей передачи информации. Такая система шифрования подразумевает высокий уровень конфиденциальности и исключает возможность повторного использования ключа, что, в свою очередь, максимально усиливает безопасность от передачи к передаче.

1.6.1. Принцип действия шифра Вернама

1) Инициация ключа. На данном этапе формируется ключ, который генерируется полностью случайным образом, обеспечивая, что каждый символ в нем уникален и непредсказуем. Длина этого ключа строго соответствует длине сообщения, которое предстоит зашифровать. При этом крайне важно, чтобы ключ был абсолютно случаен, и использовался строго один раз, исключительно для одного, конкретного сообщения. После использования ключ должен быть или немедленно уничтожен, или храниться в условиях, исключающих любую возможность его раскрытия или несанкционированного доступа, с тем, чтобы обеспечить максимальную безопасность передаваемой информации.

2) Процесс шифрования. В данном процессе используется техника шифрования XOR (исключающее ИЛИ). Эта операция применяется к каждому биту или символу исходного сообщения в сочетании с соответствующим битом или символом из ключа. В результате этого слияния возникает зашифрованная версия исходного текста, которая практически неразборчива без соответствующего ключа.

3) Метод дешифрования. Дешифрование сообщения осуществляется методом, аналогичным шифрованию. Зашифрованный текст повторно подвергается операции XOR, но на этот раз ключ используется тот же, который был применен при шифровании. Благодаря тому, что операция XOR является обратимой, в результате обработки зашифрованный текст преобразуется обратно в его исходное, нешифрованное состояние, позволяя получателю восстановить исходное сообщение в точном виде, в котором оно было отправлено.

1.6.2. Основные преимущества шифра Вернама

1) Абсолютная криптографическая стойкость. Если ключ подходит по всем параметрам – он сгенерирован совершенно

случайным образом, используется строго однократно и его длина не меньше, чем длина шифруемого сообщения, то становится совершенно невозможным дешифрование данного сообщения без точного знания ключа. Это гарантирует, что никакие методы подбора или взлома не позволят раскрыть содержание зашифрованного текста без соответствующего ключа.

2) Невосприимчивость к крипто анализу: в силу отсутствия какой-либо статистической связи между текстом сообщения и его зашифрованным аналогом, шифр Вернама не поддаётся традиционным крипто аналитическим атакам, таким как анализ частотности символов или прочие методы, основанные на статистическом анализе. Это делает его превосходным решением для задач, где необходим высочайший уровень конфиденциальности.

Однако, несмотря на описанные преимущества, существует ряд серьезных ограничений, которые снижают практическую применимость шифра Вернама. Основная проблема заключается в сложностях, связанных с генерацией абсолютно случайных ключей, их сохранением в секрете, реализацией безопасного обмена этими ключами между отправителем и получателем, а также необходимостью уничтожения ключей после каждого использования, чтобы предотвратить возможность их повторного применения или перехвата. Эти аспекты требуют сложной инфраструктуры и строгих процедур обращения с ключами, что делает шифр менее удобным для повседневного использования в условиях, когда время и доступные ресурсы ограничены.

Пример 1.5.

Шифруемый текст – Солнцестояние.

Ключ – Авиадиспетчер.

Нумерация букв для слова «солнцестояние»:	Для ключевого слова «авиадиспетчер»:
с = 19	а = 1
о = 16	в = 3

л = 13	и = 10
н = 15	а = 1
ц = 24	д = 5
е = 6	и = 10
с = 19	с = 19
т = 20	п = 17
о = 16	е = 6
я = 33	т = 20
н = 15	ч = 23
и = 10	е = 6
е = 6	р = 18

Процесс шифрования:

1) с (19) + а (1) = 20 (т)
2) о (16) + в (3) = 19 (с)
3) л (13) + и (10) = 23 (ч)
4) н (15) + а (1) = 16 (о)
5) ц (24) + д (5) = 29 (ы)
6) е (6) + и (10) = 16 (о)
7) с (19) + с (19) = 5 (е) 38 - 33
8) т (20) + п (17) = 4 (д) 37 - 33
9) о (16) + е (6) = 22 (х)
10) я (33) + т (20) = 20 (т) 53 - 33
11) н (15) + ч (23) = 5 (е) 38 - 33
12) и (10) + е (6) = 16 (о)
13) е (6) + р (18) = 24 (ц)

Таким образом, зашифрованное сообщение слово «солнцестояние» с использованием ключа «авиадиспетчер» будет выглядеть как «тсчоыоедхтеоц».

1.7. Омофоническая замена

Омофоническая замена представляет собой один из методов шифрования, используемый в классической криптографии, который предназначен для повышения степени защиты данных, устраняя

эффективность частотного анализа исходного текста. Отличаясь от классических моно алфавитных подстановочных шифров, где каждому символу открытого текста соответствует только один символ в зашифрованном тексте, омофонический шифр вводит механизм, позволяющий одной букве открытого текста соответствовать множеству различных символов или даже целым группам символов в зашифрованном тексте.

Ключевая идея данного метода заключается в создании равенства частот появления символов в шифрованном тексте, что достигается за счет присвоения более часто встречающимся буквам исходного текста большего количества альтернативных кодов или омофонов. Это действие существенно усложняет использование частотного анализа как метода криптоанализа, потому что частота появления символов в зашифрованном тексте перестает напрямую коррелировать с их частотой в обычном, нешифрованном языке. Итак, когда аналитик пытается применить частотный анализ к омофонически зашифрованному тексту, он сталкивается с тем, что более частые символы являются амбигвальными и не указывают однозначно на конкретные буквы исходного текста. Все это в совокупности делает метод омофонической замены весьма эффективным инструментом в области обеспечения криптографической безопасности.

1.7.1. Реализация шифра

1) Создание таблицы омофонов.

Этот критически важный первый шаг включает в себя разработку специализированной таблицы, где каждой букве открытого текста присваиваются один или несколько уникальных кодов. Например, в русском языке буквы, которые появляются наиболее часто, такие как «о», «е», и «а», получают значительно больше кодов по сравнению с буквами, встречающимися реже. Это распределение кодов позволяет уравнять вероятность появления

различных зашифрованных символов, затрудняя применение частотного анализа к зашифрованному тексту.

2) Шифрование сообщения.

На втором этапе, когда требуется зашифровать конкретное сообщение, происходит выбор кодов для каждой буквы исходного текста. Выбор осуществляется случайным образом из числа кодов, соответствующих каждой букве, указанных в таблице омофонов. Это означает, что даже одна и та же буква может быть зашифрована различными символами или группами символов в разных местах текста, что делает шифрование не только надежным, но и динамичным.

3) Дешифрование сообщений.

Последний этап процесса включает в себя обратное преобразование каждого зашифрованного кода в соответствующую ему букву открытого текста с помощью той же таблицы омофонов. Несмотря на множество возможных кодов для одной буквы, использование статической таблицы омофонов позволяет дешифрование оставаться точным и однозначным, гарантируя, что каждый код корректно интерпретируется в соответствие с первоначальным значением буквы.

Таким образом, омофонический шифр и его реализация обеспечивают один из самых надежных методов шифрования информации в классической криптографии, обходя различные методики криптоанализа благодаря уникальному распределению и использованию множественных кодов для одних и тех же букв.

1.7.2. Преимущества и недостатки

Одним из явных преимуществ омофонической замены является значительное усложнение частотного анализа, достигаемое за счёт обеспечения равномерного распределения частот выходных символов. Это равномерное распределение значительно повышает криптографическую надёжность и стойкость зашифрованных

сообщений, делая их менее уязвимыми к различным методам дешифровки. Применение омофонической замены способствует созданию условий, при которых становится крайне трудно определить, какие символы в тексте встречаются чаще, тем самым затрудняя попытки взлома кода.

В то же время, метод омофонической замены имеет свои недостатки. Ключевым из них является сложная задача разработки эффективной и надёжной таблицы омофонов. Создание такой таблицы требует тщательной работы и значительных усилий, так как необходимо учесть и равномерно распределить большое количество кодовых символов по различным буквам. Другой существенный недостаток заключается в необходимости безопасного обмена таблицей омофонов между отправителем и получателем. Это представляет собой серьёзный вызов, поскольку любое утечка или несанкционированное раскрытие этой таблицы может полностью скомпрометировать безопасность передаваемой информации. Эффективность и надёжность метода напрямую зависят от способности обеспечить конфиденциальность данной таблицы в процессе её использования, что требует продуманных мер безопасности и доверия между участниками коммуникации.

Пример 1.6 (табл. 7).

Предположим, для каждой буквы русского алфавита мы определили следующие омофоны:

Таблица 7

Реализация шифра

С	01	02	03
О	04	05	06
К	07	08	09

Шаги шифрования.

1) Выбор омофонов для буквы С. Мы можем выбрать любой

из трех вариантов (01, 02, 03). Предположим, что это будет 02.

2) Выбор омофонов для буквы О. Из вариантов (04, 05, 06) выберем 05.

3) Выбор омофонов для буквы К. Из представленных вариантов (07, 08, 09) допустим, мы выбрали 09.

Таким образом, слово «СОК» будет зашифровано как «020509».

1.8. Общая аудиторная работа

Реализуйте программу шифрования/дешифрования по алгоритму «Цезаря» с визуализацией проекта для заданного преподавателем текста. Номер вашего варианта из списка фамилий будет являться ключом для сдвига.

1.8.1. Индивидуальные задания к лабораторной работе № 1

Реализуйте программу шифрования/дешифрования для текстов из вашего варианта по таблице 8 тремя разными историческими шифрами. В качестве сдвига или ключевого используйте номер вашего варианта или свою фамилию, соответственно.

Таблица 8

Задание к лабораторной работе № 1

№ задания	Выражение
1	«А меж тем сердце трепетало, ждало, Алексей, ждало тебя...» «И жалость к ней уж в сердце не кипит, ровно и чисто в сердце отзывается» «И вот на тихой улице остановилась карета, дом передо мной»
2	«Буря мглою небо кроет, вихри снежные крутя» «На вечерней заре алеют облака» «Сквозь горницу идут мерцающие сумерки»
3	«Онегин был, по мненью многих (судей решительных и строгих)» «Как ни странно, как ни странно, мне смеяться охота» «Как часто летнею порою, когда прозрачно и светлою ночь»

Продолжение табл. 8

4	«Седой Кавказ со всех сторон мордами гор встречает нас» «Здесь море, горы и долины, так чудно сочетаются» «Не нагнетает ли со всех сторон таинственная темнота?»
5	«Отгорели кремнистые дали — и сонное небо зажглось» «Печально я гляжу на нашу молодежь». «Оставалось лишь трое свидетелей, ничего не понимавших из увиденного»
6	«Читатель, помнишь? Была зима ненастья» «С тех пор подует ветер северный, зима крепчает, пруды льдом замерзнут. «"Стук колес, треск плетей и крики возничих»
7	«В самом деле, солнце встает и заходит» «Так закат пламенный мрак ночной поглотил» «Призрак печали ты, ночная тень»
8	«Что значит русская степь, да солнышко в летний день!» «Белеет парус одинокий в тумане моря голубом!». «В жизни мирной нежданно негодяи, сие больших бедствий предвестник»
9	«Зашумел, завеселился народ на улицах» «Басня светит сердцу из далека» «Век живи – век учись, дураком помрешь»
10	«На душе стало как-то несвойственно тихо, как в небе без ветра». «Вдоль по улице метелица метет, вдоль по улице фонари зажгли» «В те дни слово 'свобода' звучало чаще обычного»
11	«Веселый шум идет из дальнего угла» «Жизнь – борьба и бег: стоит остановиться – подавят» «Быть может, поздно, а быть может, рано»
12	«Осенью деревья сбрасывают разноцветные листья» «Мрак ночной как будто глубже стал» «Дерево старое ветер одолел»
13	«Лес шумит, им мороз владеет» «Мороз уже задел воду; на брегах лежит тонкий ледок» «Солнышко мерцает на снегу, блестит, как золото»
14	«Скатертью белою укрыта земля, к вечеру встал морозец» «Ветер ищет вход в куртку, завывает, как собака» «И тяжело в лесу! И блаженство смотреть на него!»

Продолжение табл. 8

15	«Мы не можем предугадать, как слова наши отзовутся» «В жизни чудесное рядом с нами всегда» «Человек подобен луке: сколько в нем, столько и выйдет»
16	«Стоял зимний вечер так ясно, и звезды горели так блистательно» «Ехали на тройке с бубенцами, и колокольчик звенел». «Вдали виднеются светлые окна домов, манящих людей теплом и уютом»
17	«Мы были юны, полны надежд, и стремились к знаниям» «Вода канула в песок, как и наши годы» «За окном шумел май, мир полон был весны и света»
18	«На дальней станции зашумели сосны под утренним ветром» «Слышны были только крики перелетных птиц» «Дорога петляла серед молодых осин и поднималась к холму»
19	«За облаками ждет нежданный мрак, или светлая прохлада?» «Земной орбите мало не покажется» «Через тучи молнии сверкали и быстро исчезли в ночи»
20	«В комнату входит старый знакомец, такой же, как и всегда». «Ночь тиха. Пустырь слушает бога, и звезда говорит звезде» «Мы встречались случайно, но вся жизнь стала казаться мне до того нецелесообразной»
21	«Золото осеннее деревьев прощалось с наступающими холодами» «Ветер унес последние листья, оставив деревья голыми» «Тучи надвигались, принося с собой зимнюю прохладу»
22	«Плыли облака, тяжелые и серые, полные дождя» «Зарево заката обливало все вокруг последними лучами солнца» «На берегу старушка сидела и смотрела на реку, что текла вечно»
23	«Луна вышла из-за туч, освещая снежную дорогу» «Снег скрипел под ногами путника, одиноко шедшего в ночи» «Темные ели возвышались, как тихие стражи заснеженного леса»
24	«Мороз создавал на окнах узоры, напоминающие далекие страны» «Тепло камина согревало комнату, где все собрались в ожидании чуда». «Запах елки и мандаринов наполнял дом, предвещая новогодние праздники»
25	«По лугам разлился туман, скрывая утренние капли росы» «Солнце медленно поднималось, предвещая начало нового дня» «Птицы начали свои ранние песни, встречая рассвет»

Продолжение табл. 8

26	«На площади зажглись фонари, один за другим, освещая вечерний город». «Люди спешили по делам, забывая оставить время на мечты». «Кафе заполнились посетителями, икающими уют в холодные осенние вечера»
27	«Небо над полем было ясным, и звезды светили ярче, чем обычно» «Ночная прохлада заставляла дрожать, но мир был так прекрасен» «На горизонте вырисовывались силуэты гор, темные и молчаливые»
28	«Старый дом стоял на окраине деревни, полный тайн прошлого» «Лес окутывал его, как старый друг, который знает все его истории» «Дверь скрипела, будто каждый раз рассказывая новую историю о пришлом»
29	«Утро встретило его морозным воздухом, который заставлял дышать полной грудью» «Снег еще не растаял, и все вокруг было белым, чистым и свежим» «Лед на реке стал крепче, и дети уже начали кататься на коньках»
30	«Городские огни мерцали вдалеке, как звезды на земле» «Старинные часы на башне издали полночный звон, отмечая конец еще одного дня». «Прохожие уже редко встречались на улицах, каждый устремился к своему укрытию»

ЛАБОРАТОРНАЯ РАБОТА № 2. КРИПТОСИСТЕМА RSA

Цель работы: изучение алгоритма шифрования RSA.

2.1. Теоретический материал

Криптосистема RSA (Rivest-Shamir-Adleman) – это одна из самых известных и широко используемых асимметрических систем шифрования. Она была разработана в 1977 году Рональдом Ривестом, Ади Шамиром и Леонардом Адлеманом и носит их фамилии. Безопасность RSA обеспечивается математической сложностью задачи факторизации больших чисел.

В текущем разделе мы подробно изучим алгоритм RSA, разбирая его математические принципы и теоретические основы, которые обеспечивают его функционирование. Мы также рассмотрим основные принципы работы данной криптосистемы и ключевые аспекты, обеспечивающие её безопасность в применении. Это даст возможность глубже понять, почему система RSA до сих пор остаётся эффективным инструментом в области криптографической защиты информации на фоне других асимметричных методов шифрования.

2.2. Математические основы RSA

Криптография в большей своей части основана на вычислениях в конечных полях. Конечные поля – это алгебраические структуры, в которых все алгебраические арифметические операции выполняются по модулю большого простого числа. При модульных операциях деления остаток обозначается от слова «module» – означающее деление по модулю. В модульной алгебре данная операция деления записывается как $a = \mathrm{b} \bmod \mathrm{n}$. Число «a» равно положительному остатку от деления числа «b» на «n». В модульных операциях вместо знака равно пишут знак «тождественно» равно: «≡», но иногда его записывают как просто равно: «=».

2.3. Примеры модульных операций

В модульной арифметике результат должен быть равен положительному остатку от деления числа, поэтому, если делится отрицательное число, то нужно данное число путем сложения с делителем по модулю, сначала довести до положительного, а потом применить операцию модульного деления:

$a \equiv (-)b \bmod n \rightarrow a \equiv (-)b + n \bmod n$ .... Выполнять сложение пока число $((-)b + n)$ не станет положительным $((-)b + n) \geq 0$.

Пример. $-3 \bmod 5 \equiv 2 \rightarrow (-3 + 5) \bmod 5 \equiv 2$; $-31 \bmod 5 \equiv 4 \rightarrow (-31 + 5 + 5 \ldots . . +5) \bmod 5 \equiv (-31 + 35) \bmod 5 \equiv 4 \bmod 5 \equiv 4$.

Если в операции сам делитель отрицательный, то меняем знак, как умножение на -1. Пример: $3 \bmod (-5) \equiv -2$.

$3 \bmod 5 \equiv 3, \quad 7 \bmod 5 \equiv 2, \quad 13 \; mod \; 7 \equiv 6, \quad 13 \bmod 6 \equiv 1, \quad -3 \bmod 5 \equiv 2$

В модульной математике операцию $a = b \bmod n$ можно заменить на выражение $b = a + m \cdot n$, для некоторых целых чисел m, и здесь число «a» называют ***вычетом*** числа «b» по модулю n. Используя набор чисел m, можно получить целый класс вычетов. Любое целое число «b» при делении на «n» мы можем представить как $b = a + m \cdot n$, где $m \in Z$ и число $a, 0 \leq a < n$, является остатком от деления. Отсюда $b - a = m \cdot n$ или $b = a \bmod n$, то есть при равенстве двух чисел от модульной операции деления на «n» числа имеют одинаковые остатки.

Понятие «сравнение числа $b = a \bmod n$» значит, что целые числа «b» находятся в одном классе эквивалентности, что и остаток от деления – «a». Всего наборов таких остатков равно $0, 1, \ldots, n-1$, тогда и наборов классов эквивалентности будет $\{0\}, \{1\}, \{2\}, \ldots, \{n-1\}$, называемых ***классами вычетов по модулю числа*** **n**, а сами числа называют просто: вычеты.

2.4. Пример классов вычетов

Наборов классов вычетов по модулю числа n = 3 будет 3: {0}, {1}, {2}.

– класс эквивалентности {0}: $\ldots \equiv -6 \bmod 3 \equiv -3 \bmod 3 \equiv 0 \bmod 3 \equiv 3 \bmod 3 \equiv 6 \bmod 3 \equiv 9 \bmod 3 \equiv \cdots$

– класс эквивалентности {1}: $\ldots \equiv -5 \bmod 3 \equiv -2 \bmod 3 \equiv 1 \bmod 3 \equiv 4 \bmod 3 \equiv 7 \bmod 3 \equiv 10 \bmod 3 \equiv \ldots$

– класс эквивалентности {2}: $\ldots -4 \bmod 3 \equiv 8 \bmod 3 \equiv 11 \bmod 3 \equiv 14 \bmod 3 \equiv \cdots$

Если брать по одному числу из каждого класса по одному и тому же модулю, то получится так называемая полная система вычетов. Например, если $n = 3$, то полная система вычетов состоит из 3 чисел, по одному из каждого класса эквивалентности. Например, $F = \{-2, 3, 8\}$.

Теорема о вычетах:

В полной системе вычетов $a_1, a_2, \ldots a_n \, (mod\, n)$ *все числа попарно несравнимы, а именно,* $a_k \neq a_l \, mod\, n \;\; (\forall\, k \neq l)$*, и обратно, если есть набор чисел* $a_1, a_2, \ldots a_n \, (mod\, n)$ *таких, что они все попарно несравнимы между собой, то эта система представляет полный набор вычетов.*

Свойства модульной арифметики:

1) $(a \pm b) \bmod n \equiv \big((a \bmod n) \pm (b \bmod n)\big) \bmod n$,
2) $(a * b) \bmod n \equiv \big((a \bmod n) * (b \bmod n)\big) \bmod n$,
3) дистрибутивность

$$(a * (b + c)) \bmod n \equiv \Big(\big((a * b) \bmod n\big) + \big((a * c) \bmod n\big)\Big) \bmod n,$$

Одним из важных моментов для модульной арифметики являются ***обратные числа***. По определению, обратным числом является число a^{-1} такое, что если его умножить на исходное, то $\mathbf{\left(a^{-1} \cdot a\right) mod\ n \equiv 1\ mod\ n}$, но обратное число будет существовать только при условии, что числа «a» и «n» будут взаимно простыми: НОД(a, n) = 1, то есть у этих чисел нет общих делителей.

2.5. Пример нахождения обратного числа по модулю

Назовем два числа a и b взаимно-простыми, если НОД$(a, b) = 1$. Найдём обратное число для числа 3 mod 11. Можно воспользоваться простым перебором:

$$3 * 0 \equiv 0 \bmod 11,$$
$$3 * 1 \equiv 3 \bmod 11,$$
$$3 * 2 \equiv 6 \bmod 11,$$
$$3 * 3 \equiv 9 \bmod 11,$$
$$3 * 4 \equiv 12 \bmod 11 = 1.$$

Найден обратный элемент $a^{-1} = 4$ для числа a= 3 по $\bmod 11$.

Очевидно, что если числа будут иметь больший порядок, то находить обратные простым перебором будет не эффективно, поэтому для быстрого нахождения обратных чисел существует расширенный алгоритм Евклида (кратко, РАЕ), описанный ниже.

По теореме Безу существуют для любых целых чисел A и B существуют целые числа x и y такие, что выполнится условие:

$$\text{НОД}\ (A, B) = x \cdot A + y \cdot B,$$

и таких решений бесконечное множество. Причем, если числа A и B, взаимно простые, то существуют целые числа x и y, что выполнится тождество: $x \cdot A + y \cdot B = 1$.

Рассмотрим ещё некоторые важные теоремы модульной арифметики. Для произвольного натурального числа n определим функцию Эйлера $\varphi(n)$, равную числу натуральных чисел $k \leq n$, взаимно-простых с n.

Теорема Эйлера.

Если a и b – взаимно простые числа, то: $a^{\varphi(n)} \equiv 1\ mod\ n$.

Свойства функции Эйлера:

1) Если n является простым числом (будем обозначать буквой p), функция Эйлера $\varphi(p) = p - 1$;

2) Если p – простое число, n – натуральное, то $\varphi(p^n) = p^{n-1}(p - 1)$;

3) Если же $n = uv$, и числа u и v взаимно-простые, тогда: $\varphi(uv) = \varphi(u) \cdot \varphi(v)$.

Примеры вычисления и использования функции Эйлера

1. $\varphi(75) = \varphi(3 \cdot 5^2) = \varphi(3)\varphi(5^2) = (3-1)(5^2 - 5^{2-1}) = 2 \cdot 20 = 40$.

2. Найдем $2^{105} mod\ 25$. Числа a = 2, n = 25 взаимно простые, значит, выполняется теорема Эйлера: $a^{\varphi(n)} \equiv 1 \bmod n$.

$\varphi(25) = \varphi(5^2) = 5^{2-1}(5-1) = 20$. Значит, $2^{20} \bmod 25 = 1$.

$2^{105} \equiv 2^5 \cdot (2^{20 \cdot 5}) \equiv 2^5 \bmod 25 \equiv 32 \bmod 25 = 7$.

Теорема Ферма (малая – частный случай теоремы Эйлера**).**

Если число n – простое и a, не делится на p, тогда, $a^{p-1} \equiv 1\ mod\ p$**.**

Следствие теоремы Ферма.

Если p – простое число и a – целое, не сравнимое с p, тогда выполняется сравнение: $a^p \equiv a \bmod p$.

Пример 1.

Проверим следствие из теоремы Ферма для чисел a=7 и p=3: $7^3 \equiv 343 \bmod 3 \equiv 7 \bmod 3 \equiv 1 \bmod 3$.

Пример 2.

Вычислить $11^{27} \bmod 7 \equiv 4^{27} \bmod 7 \equiv 4^{6 \cdot 4+3} \equiv 4^3 \equiv 64\ mod\ 7 = 1$.

Пример 3. Вычислить $17^{137} \bmod 7$. Ответ: 5.

Ознакомившись кратко с основами модульной арифметики, можно приступить к рассмотрению алгоритма шифрования RSA.

2.6. Алгоритм RSA

2.6.1. Метод создания ключей

1) Выберем два случайных простых числа p и q определенного размера;

2) вычислим n, равное произведению чисел p и q;

3) вычислим значение функции Эйлера числа n: $\varphi(n) = \varphi(p) \cdot \varphi(q)$;

4) выбирается произвольное число e из диапазона $(1;\ \varphi(n))$, при условии НОД(e, φ(n)) = 1;

5) используя расширенный алгоритм Евклида, вычислим секретное число d, удовлетворяющее условию):

$$e * d \bmod \varphi(n) = 1.$$

Таким методом получается открытый ключ K_O, являющийся парой (e, n), и секретный ключ K_C, представленный парой (d, n).

2.6.2. Процедура шифрования

Сообщение M переводится в набор чисел m_i, $0 < m_i < n,\ i \leq k$, и закодированное сообщение определяется по формуле:

$$C_i \equiv (m_i)^e \bmod n,\ i \leq k.$$

2.6.3. Процедура расшифровки

Набор k закодированных сообщений C_i расшифровывают, используя приватный ключ $K_c = (d, n)$ и формулу:

$$m_i \equiv (C_i{}^d) \bmod n,\ i \leq k.$$

Пример 1.

При использовании метода RSA мы проведем операцию шифрования и дешифрования с информацией, представленной сообщением «КФУ». Предположим, что требуется передать секретную информацию от пользователя «А» к пользователю «Б».

На начальном этапе пользователю «Б» потребуется получить как открытый, так и закрытый ключ. Для этого он производит следующие шаги:

1) генерирует простые числа p и q, с проверкой их на простоту. Например: p = 17, q = 13;

2) вычисляет их произведение $n = 17 * 13 = 221$;

3) вычисляет функцию Эйлера для числа n:

$$\varphi(n) = (p - 1) * (q - 1) = 16 * 12 = 192;$$

4) выбирает открытое число e в пределах $1 < e < \varphi(n)$ и которое будет взаимно простым с функцией Эйлера $\varphi(n)$: $1 = НОД(e, \varphi(n))$, например e = 7;

5) формирует открытый ключ K_O: (e, n) и посылает его в открытый доступ пользователю А;

6) пользователь А, получив e и φ(n), формирует зашифрованное сообщение по формуле $C_i \equiv (m_i)^e \bmod n$ и пересылает пользователю Б;

7) пользователь Б, получив сообщение от А, расшифровывает его по формуле $m_i \equiv (C_i{}^d) \bmod n$, используя свой секретный ключ K_C: (d, n).

Сложность процедуры факторизации числа n защищает алгоритм RSA от взлома злоумышленником, который знает только открытый ключ (n, e). Нахождение простых множителей p и q большого числа n – вычислительно сложная задача. Не существует известных алгоритмов, которые могли бы сделать это быстро (за полиномиальное время) для больших n, используемых в RSA.

Процедуру поиска закрытого ключа d методом расширенного алгоритма Евклида проще записывать в виде таблицы 9. Если уравнение в общем виде выглядит как: $x * A + y * B = НОД(A, B)$, то таблица заполняется по шагам: на первом шаге заполняются значения в первых четырех столбцах по первой строке (табл. 9).

Таблица 9

Алгоритм Евклида (часть 1)

A	B	A mod B	A div B	x	y
192	7	3	27		
7	3	1	2		
3	1	0	3		

Далее вторая строка заполняется со смещением влево: значение в столбце «A» становится значением столбца «B», а значение столбца «A mod B» становится столбцом «B» (показано стрелками). Значение

для столбцов «A mod B» и «A div B» вычисляются. Далее процесс повторяется, пока значение столбца «A mod B» не равно нулю. После заполнения первых четырёх столбцов, выполняется обратный ход метода Евклида. Для этого заполняются значения для столбцов «x» и «y», но снизу вверх по формулам:

$$x_i = y_{i-1}\,, \qquad y_i = x_{i-1} - y_{i-1} * (A\ div\ B)_i.$$

На первом шаге полагают $x_i = 0,\ y_i = 1$ при $i = 0$ (табл. 10–12).

Таблица 10

Алгоритм Евклида (часть 2)

A	B	A mod B	$(A\ div\ B)_i$	x_i	y_i	i
192	7	3	64			2
7	3	1	2			1
3	1	0	3	0	1	0

$$x_1 = y_{1-1=0} = 1,$$
$$y_1 = x_{1-1=0} - y_{1-1=0} * (A\ div\ B)_1 = 0 - 1 * 2 = -2.$$

Таблица 11

Алгоритм Евклида (часть 3)

A	B	A mod B	$(A\ div\ B)_i$	x_i	y_i	i
192	7	3	27			2
7	3	1	2	1	-2	1
3	1	0	3	0	1	0

$$x_2 = y_{2-1} = -2,$$
$$y_2 = x_{2-1} - y_{2-1} * (A\ div\ B)_2 = 1 - (-2) * 27 = 55$$

Таблица 12

Алгоритм Евклида (часть 4)

A	B	A mod B	$(A\ div\ B)_i$	x_i	y_i	i
192	7	3	27	-2	55	2
7	3	1	2	1	-2	1
3	1	0	3	0	1	0

Процедура шифрования.

Для начала приведем наше сообщение M к числовой форме, представив его как последовательность целых чисел в определенном диапазоне, например, от 1 до 33 (для русского алфавита), где каждой русской букве будет назначено соответствующее число в соответствии с их позицией в алфавите начиная с числа 1. Таким образом, наше сообщение «КФУ» будет представлено в виде набора {12, 22, 21}. Используем публичный ключ $K_o = (7,221)$, получаем:

$C_1 \equiv (m_1{}^e) \bmod n \equiv 12^7 \bmod 221 \equiv 12^2 * 12^2 * 12^2 * 12^1 \bmod 221 \equiv 194,$

$C_2 \equiv (m_2{}^e) \bmod n \equiv 22^7 \bmod 221 \equiv 22^2 * 22^2 * 22^2 * 22^1 \bmod 221 \equiv 61,$

$C_3 \equiv (m_3{}^e) \bmod n \equiv 21^7 \bmod 221 \equiv 21^2 * 21^2 * 21^2 * 21^1 \bmod 221 \equiv 200.$

Расшифрование.

Используем приватный ключ $K_c = (55, 221)$ и по формуле $m_k \equiv \left(C_k{}^d\right) \bmod n$, получаем ряд ответов:

$m_1 \equiv \left(C_1{}^d\right) \bmod n \equiv 194^{55} \bmod 221 \equiv 194^{2^{27}} * 194 \bmod 221 \equiv 12,$

$m_2 \equiv (C_2{}^d) \bmod n \equiv 61^{55} \bmod 221 \equiv 61^{2^{27}} * 61 \bmod 221 \equiv 22,$

$m_3 \equiv (C_3{}^d) \bmod n \equiv 200^{55} \bmod 221 \equiv 200^{2^{27}} * 200 \bmod 221 \equiv 21.$

Декодируя сообщение {12, 22, 21} получаем слово «КФУ».

2.7. Безопасность системы RSA

Безопасность криптографической системы RSA основана на сложности процесса факторизации больших чисел n, то есть разложение n на простые множители. Этот процесс является ключевым моментом в обеспечении безопасности RSA. Задача факторизации является вычислительно сложной, и на данный момент не существует алгоритмов, способных реализовать факторизацию за приемлемое время для больших чисел (длина n составляет 1024 бита или более). Такая сложность обеспечивает надежную защиту данных, закодированных с использованием данной системы.

Вместе с тем, стремительное развитие технологий в области квантовых вычислений и постоянное увеличение вычислительных

мощностей компьютеров вносит определенные риски и вызывает беспокойство среди специалистов по кибербезопасности.

2.8. Общая аудиторная работа

Задание 1. Реализуйте программу нахождения НОД(A,B) для двух чисел по расширенному алгоритму Евклида (числа «A» и «B» брать по номеру варианта (табл. 14)).

Задание 2. Реализуйте программу для быстрого возведения любого числа в степень по модулю: $A^B \bmod D$. Представить решение для варианта из таблицы 14.

Таблица 14

Задание к лабораторной работе № 2

№ задания	Выражение
1	A = 23433, B = 88397, D =17
2	A = 29413, B = 48797, D =19
3	A = 13431, B = 82597, D =21
4	A = 23473, B = 88517, D =13
5	A = 21383, B = 88557, D =11
6	A = 56433, B = 12497, D =31
7	A = 24885, B = 81097, D =33
8	A = 24431, B = 88395, D =19
9	A = 13437, B = 12397, D =17
10	A = 12333, B = 46597, D =37
11	A = 23423, B = 88567, D =13
12	A = 23123, B = 88387, D =19
13	A = 23233, B = 18997, D =7
14	A = 26633, B = 84497, D =3
15	A = 13455, B = 55391, D =29
16	A = 12433, B = 88557, D =23
17	A = 97437, B = 12393, D =27

Продолжение табл. 14

18	A = 83553, B = 18773, D =83
19	A = 24879, B = 45253, D =19
20	A = 87859, B = 45257, D =97
21	A = 34877, B = 55251, D =93
22	A = 34855, B = 25257, D =63
23	A = 34907, B = 55341, D =53
24	A = 34875, B = 54251, D =91
25	A = 33377, B = 25255, D =77
26	A = 24565, B = 55221, D =23
27	A = 74861, B = 57213, D =13
28	A = 39077, B = 19001, D =71
29	A = 30077, B = 90051, D =3

2.9. Индивидуальные задания к лабораторной работе № 2

Реализуйте программу с интерфейсом для шифратора и дешифратора по алгоритму RSA для русского алфавита с учётом пробелов. Для вычисления пары открытый/закрытый ключ использовать расширенный метод Евклида. Также добавьте проверки на простоту для чисел p, q и проверку на взаимную простоту для чисел e и $\varphi(n)$. Индивидуальные тексты для шифрования брать из первой лабораторной работы по номеру варианта. Подготовьте отчет по выполненному заданию. Отчет должен содержать в себе титульный лист, постановку задачи, аналитическое решение по заданному заданию, блок схему, код программы и скриншоты реализации.

ЛАБОРАТОРНАЯ РАБОТА № 3. ГЕНЕРАТОР ПОЛЯ

Цель работы: научиться находить генератор поля для чисел, используя алгоритм Евклида и китайскую теорему об остатках.

3.1. Теоретический материал

Как известно, GF(n) – это конечное поле (поле Галуа), содержащее n элементов. Генератором поля в модульной арифметике называют элемент, который может быть использован для получения всех остальных элементов поля с помощью операций сложения или умножения. Число элементов поля является либо простым числом, либо степенью простого числа p ($n = p^k$ для некоторого натурального k).

Можно также сказать, что генератор поля по умножению – это элемент, являющийся первообразным корнем, то есть элемент $g \in GF(n)^*$ (мультипликативная группа ненулевых элементов поля), такой, что все ненулевые элементы поля могут быть выражены как степень элемента g: $a = g^k$ для $k = 0,1, \dots, q-1$.

Для полей GF(n) все обычные правила арифметики выполняются с учетом того, что результаты всегда сводятся по модулю n. Выясним, какое число будет генератором поля GF(n)?

Определение. Мультипликативным порядком ненулевого элемента a поля GF(n) называется наименьшая степень k такая, что $a^k = 1$. Порядок обозначается через $ord_n(a)$.

Теорема (Эйлер). Порядок любого ненулевого элемента конечного поля есть делитель порядка мультипликативной группы поля $GF^*(n)$, то есть делитель $n-1$.

Теорема (Лагранж). Мультипликативная группа поля GF(n) содержит ровно $\varphi(k)$ элементов порядка k для каждого делителя k порядка группы $GF^*(n)$, то есть делителя $p-1$.

Пусть, например, n = 7. По теореме Лагранжа порядки всех элементов поля GF(7) являются делителями $p-1=6$, то есть могут

принимать значения 1, 2, 3 и 6. Элементов порядка 6 будем $\varphi(6) = 2$, то есть GF(7) содержит 2 примитивных элемента (генератора).

Чтобы найти генератор (первообразный корень) поля GF(7), нужно найти элемент, который порождает все ненулевые элементы поля GF(7) через свои степени (табл. 15). Найдем порядки каждого элемента $GF^*(7)$:

Таблица 15

Нахождение порядка элемента

a	1	2	3	4	5	6
$ord_7(a)$	1	3	6	3	6	2

Из таблицы видно, что все порядки являются делителями 6, а генераторами поля GF(7) являются элементы g = 3 и g = 5. Они порождают все ненулевые элементы поля через свои степени. Элементы же 4 и 6 не являются генераторами поля, так как их порядки (3 и 2 соответственно) меньше 6. Понятие генератора поля является важным в теории конечных полей, используемый в криптографии и теории кодирования. Он позволяет представить все ненулевые элементы поля в виде степеней одного числа.

Поиск возможных порядков элементов поля GF(p).

Чтобы найти значения порядков элементов поля, необходимо разложить $p - 1$ в произведение простых сомножителей. Если число p велико, то надо использовать методы факторизации типа алгоритма Ферма или Полларда.

3.2. Индивидуальные задания к лабораторной работе № 3

1. Написать программу для поиска всех простых делителей элемента $p - 1$ заданного поля GF(p), используя ρ −метод факторизации Полларда.

2. Найти число генераторов поля GF(p), равное $\varphi(p - 1)$.

3. Найти наименьший генератор поля GF(p), используя множество делителей $p-1$.

Пример. Пусть p=17148005297.

1) Используя $\rho-$метод факторизации Полларда, найдем все простые делители числа $p-1$:

$$17148005296 = 2^4 \cdot 397 \cdot 2699623$$

2) Найдем значение функции Эйлера:

$$\varphi(p-1) = p \cdot \prod_{t|p-1}\left(1-\frac{1}{t}\right) = 17148005296\left(1-\frac{1}{2}\right)\left(1-\frac{1}{397}\right)\left(1-\frac{1}{2699623}\right) = 8552402496.$$

3) Для поиска наименьшего генератора предлагается следующий алгоритм:

3.1) Перебираем в цикле все элементы a от 2 до $p-2$. Пусть произвольный элемент a выбран.

3.2) В цикле перебираем все простые делители t числа $p-1$, вычисляем $x=(p-1)/t$ и $z=a^x mod\, p$. Если $z=1$, то порядок a меньше $p-1$ и a не является генератором. Отбрасываем данное a и переходим к следующему.

3.3) Если для данного a все вычисленные $z \neq 1$, элемент a – генератор поля.

Выполнив предложенные вычисления, получим значение наименьшего генератора поля $GF(p),\ p=17148005297$, равное 3.

Варианты заданий:

№	1	2	3	4
p	1152140745299	1779363684401	1324450860781	1086511565081

№	5	6	7	8
p	1061921596181	1629670642307	1308187698731	1579502180561

№	9	10	11	12
p	1709417653991	1323395418581	1211849279933	1261386354577

№	13	14	15	16
p	1106709010241	1992388566661	1030516351441	1046018945681

№	17	18	19	20
p	1704203825267	1305570972067	1030516351441	1539772389733

№	21	22	23	24
p	3204256385261	8305470990063	9036516351771	2539442389731

№	25	26	27	28
p	5704343829297	2305570112897	7030016351443	9539772389993

№	29	30
p	8704266825261	2205870972057

ЛАБОРАТОРНАЯ РАБОТА № 4. ЭЛЕКТРОННАЯ ЦИФРОВАЯ ПОДПИСЬ

Цель работы: анализ и программирование методов шифрования и дешифрования криптосистемы Эль-Гамаля и разработка программного инструмента для создания электронной цифровой подписи (ЭЦП).

4.1. Хеш-функция (или функция хеширования)

Хеш-функция является неотъемлемой частью современной криптографии и вычислительной техники. Она представляет собой математическую функцию, которая преобразует входные данные произвольного размера в фиксированной длины строку символов, называемую хеш-значением или хешем. Хеш-функции играют ключевую роль в различных областях компьютерной безопасности, включая хранение паролей, проверку целостности данных и цифровые подписи.

Для того чтобы хеш-функция была полезной и безопасной в криптографическом контексте, она должна обладать несколькими важными свойствами:

1) Детерминированность.

Для одного и того же входного значения хеш-функция всегда должна возвращать одно и то же хеш-значение.

2) Быстрота вычисления.

Хеш-функция должна быстро вычислить свои значения, используя полиномиальный алгоритм.

3) Невозможность восстановления аргумента по значению (one-way property).

Исходные данные не должны быть легко восстановимы по хешу исходного сообщения. Это свойство делает функцию обратимой только с огромными вычислительными затратами.

4) Устойчивость к коллизиям.

Ввероятность того, что два разных входных значения дают одинаковые хеш-значения, должна быть крайне мала.

5) Устойчивость к атакам на основе второго прообраза.

Для заданного хеша невозможно найти другой аргумент с тем же значением за разумное время.

В процессе создания хеш-значения выполняется разбиение начального текста на символьные секции, обозначаемые как m_i, которые затем подлежат пошаговой обработке согласно следующей формуле: $H_k = f(H_{k-1}, m_k)$.

Таким образом, структура хеш-функции напоминает пирамидальное устройство, где для формирования хеш-отображения последующего символа применяется хеш-образ предшествующего ему символа. После обработки конечного символа в сообщении, полученное хеш-значение определяется как хеш-образ всего текста сообщения.

Для наглядности возьмем упрощенную версию хеш-функции, пусть она задана:

$$H_k = (H_{k-1} + m_k)^2 \bmod n,$$

здесь $n = p * q$, p и q – большие простые числа;

H_0 – начальное произвольное значение;

m_k – k-й блок сообщения $M = \{m_1, m_2, \ldots, m_k\}$.

Пример.

Подсчитаем хеш-функцию для слова «КФУ», данное сообщение для русского алфавита в цифровом эквиваленте будет выглядеть как набор чисел {12, 22, 21}. Выберем произвольно два простых числа и подсчитаем их произведение $n = p * q = 23 * 37 = 851$ (в реальном шифровании естественно нужно выбирать числа высоких порядков). Пусть $H_0 = 150$. Шифруем каждую букву:

$$H_1 = (H_{1-1} + m_1)^2 \bmod n \equiv (150 + 12)^2 \bmod 851 \equiv 754 \bmod 851,$$

$$H_2 = (H_{2-1} + m_2)^2 \bmod n \equiv (754 + 22)^2 \bmod 851 \equiv 519 \bmod 851,$$

$$H_3 = (H_{3-1} + m_3)^2 \bmod n \equiv (776 + 21)^2 \bmod 851 \equiv 363 \bmod 851,$$

В результате хеш-функция для слова «КФУ» будет $H(M) = 363$.

4.2. Электронная цифровая подпись (ЭЦП)

В 1976 году Уитфилд Диффи и Мартин Хеллман впервые представили концепцию электронной цифровой подписи (ЭЦП), заложив теоретическую основу для создания схем, гарантирующих надежную защиту данных. ЭЦП служит для обеспечения целостности информации при ее передаче и хранении в электронном виде, предотвращая возможность подделки. Электронная цифровая подпись формируется методом криптографической обработки данных с применением приватного ключа, что гарантирует подлинность ключа подписи и защищает информацию от искажений.

Фактически, электронная цифровая подпись представляет собой сложную программно-криптографическую систему, обеспечивающую определенные обязательства:

1) гарантирует защиту данных от несанкционированных изменений, сохраняя их неприкосновенность;

2) обеспечивает конфиденциальность данных, предоставляя доступ к информации исключительно уполномоченным лицам;

3) подтверждает подлинность отправителя, удостоверяя аутентичность источника информации.

Цифровая подпись играет для электронных документов ту же роль, что и ручная подпись для бумажных документов. Однако отличительная черта электронной подписи заключается в её уникальности для каждого конкретного сообщения, что делает каждую подпись уникальной и невоспроизводимой, в отличие от традиционной ручной подписи.

Разработка алгоритма цифровой подписи влечёт за собой решение нескольких важных задач:

1) создание подписи, которую невозможно подделать;

2) верификация принадлежности подписи определённому лицу;

3) невозможность отказа от авторства подписи со стороны её создателя.

Эти аспекты современной криптографии и цифровой безопасности представляют собой основу надежности и функциональности систем защиты информационных потоков в цифровую эру.

4.3. Классический метод генерации цифровой подписи

Перед началом создания цифровой подписи, инициатор данного процесса должен выполнить генерацию двух ключей, которые являются неотъемлемой частью данной операции. Эти ключи включают в себя публичный ключ $К_О$ и приватный ключ $К_С$. Приватный ключ, как следует из названия, должен быть строго конфиденциален и храниться только у лица, выполняющего подписание, или отправителя, тогда как публичный ключ должен быть доступен для всех желающих проверить аутентичность цифрового сообщения. Вместо того чтобы подписывать текст сообщения напрямую, принято выполнять подписание его хеш-значения, что увеличивает безопасность процесса.

Реализация цифровой подписи в соответствии с общепринятым алгоритмом включает выполнение следующих детально расписанных шагов:

1. Сначала отправитель задает хеш-значение m данного текста M, используя заранее выбранную хеш-функцию H, что позволяет создать уникальный и необратимый отпечаток исходного текста.

2. Затем, на основе этого хеш-значения, генерируется цифровая подпись S, с использованием приватного ключа $К_С$, что обеспечивает возможность верификации исходного сообщения при сохранении конфиденциальности ключа подписывающего.

3. Последней операцией является создание и отправка модифицированного сообщения (M, S), которое объединяет в себе оригинальный текст M и соответствующую ему цифровую подпись S.

В результате совершения процесса цифровой подписи не только обеспечивается подлинность сообщения, но также предоставляется возможность для прозрачной и надежной проверки этой подлинности всеми заинтересованными участниками благодаря применению открытого ключа.

4.4. Классический метод верификации цифровой подписи

При использовании стандартного метода верификации цифровой подписи, получатель сталкивается с задачей аутентификации подписи и проверки целостности сообщения (M', S). Это требует выполнения определенной последовательности действий:

1. Инициирование процесса генерации хеш-значения m' для данного сообщения M', применяя ту же хеш-функцию H, которая была использована при создании подписи. Это нужно для формирования уникального хеш-изображения, которое будет служить контрольной суммой для исходного текста сообщения.

2. Использование публичного ключа $К_О$ для извлечения первоначального хеш-значения m из электронной подписи S. Этот шаг позволяет расшифровать подпись и получить хеш, на основе которого подпись была сформирована.

3. Сопоставление двух хеш-значений: m', полученного из сообщения, и m, извлеченного из подписи. Если данные значения идентичны, это подтверждает не только то, что подпись аутентична, но и то, что текст сообщения не был изменен в процессе передачи.

Однако следует отметить, что сохранность сообщения и подписи может быть под угрозой в случае успешной атаки на хеш-функцию или в случае, если несанкционированные лица получат доступ к приватному ключу $К_С$. Современные хеш-функции представляют собой сложные математические алгоритмы с особыми характеристиками и механизмами защиты, которые существенно снижают вероятность успешных попыток хакерских атак. Например, хеш-функция SHA-1, которая была стандартизирована

Национальным институтом стандартов и технологий США (NIST) в 1995 году, использует сложный процесс для генерации 160-битных хеш-значений. Этот процесс включает обработку входных данных блоками по 512 бит, что обеспечивает высокую степень надежности и служит основой для безопасного создания цифровых подписей и других криптографических приложений.

Однако, несмотря на исходную надежность SHA-1, постоянное развитие методов крипто анализа и увеличение вычислительных мощностей компьютеров привели к необходимости улучшения безопасности. Поэтому начиная с 2010 года было инициировано постепенное внедрение более современных и мощных хеш-функций из семейства SHA-2. Эти новые функции разработаны для удовлетворения более строгих требований безопасности и могут генерировать хеш-значения различной длины, такие как 224, 256, 384, 512 и даже 1024 бита. Увеличение длины хеш-значений значительно усиливает защиту от потенциальных атак, делая такие системы менее уязвимыми для взлома и различных видов атак, таких как атаки с коллизиями.

Семейство SHA-2 обладает рядом значительных преимуществ по сравнению со своим предшественником. Эти преимущества включают улучшенную стойкость к атакам методом подбора, улучшенную защиту от коллизий и более высокую общую надежность. В частности, более длинные хеш-значения усложняют процесс поиска двух различных входных данных, которые дают один и тот же хеш. В результате эти функции получили широкое распространение в применениях, требующих высокой степени доверия и безопасности, таких как финансовые транзакции, аутентификация и защита данных в критических системах.

Последнее достижение в мире информационной безопасности – хеш-функция SHA-3 (Keccak – произносится как «кечак») – алгоритм хеширования переменной разрядности, разработанный группой авторов во главе с Йоаном Дайменом, соавтором Rijndael, автором

шифров MMB, SHARK, Noekeon, SQUARE и BaseKing. 2 октября 2012 года Keccak стал победителем конкурса криптографических алгоритмов, проводимого Национальным институтом стандартов и технологий США. 5 августа 2015 года алгоритм утверждён и опубликован в качестве стандарта FIPS 202. В программной реализации авторы заявляют о 12,5 циклах на байт при выполнении на ПК с процессором Intel Core 2. Однако в аппаратных реализациях Keccak оказался намного быстрее, чем все другие финалисты.

В целом, эволюция хеш-функций от SHA-1 к SHA-3 отражает общую тенденцию в области криптографии – стремление к созданию и внедрению более безопасных и надежных методов защиты информации. Эта тенденция обеспечивает постоянное укрепление систем информационной безопасности в условиях текущих и будущих угроз.

4.4.1. Процесс создания ключей в рамках RSA

Для создания цифровой подписи в рамках методологии RSA необходимо сгенерировать ключевую пару, состоящую из публичного и приватного ключей, в соответствии с криптографическими принципами RSA. Этот процесс включает несколько этапов:

1. Выбираем два случайных простых чисел p и q, при этом условие $p \approx q$.

2. Вычисление их произведения, обозначающееся как $n = p * q$.

3. Определение значения функции Эйлера для n, которая вычисляется как $\varphi(n) = (p - 1) * (q - 1)$.

4. Выбирается открытая экспонента e, удовлетворяющая условиям: $1 < e < \varphi(n)$ и $НОД(e, \varphi(n)) = 1$.

5. Вычисляется закрытая экспонента d (обратное число к e) такая, что $(e * d) \bmod \varphi(n) = 1$.

6. Публичный ключ $K_0 = (e, n)$ передается по открытым каналам для проверки цифровой подписи.

7. Автор сохраняет секретный ключ $K_C = (d, n)$ для подписания сообщений в будущем.

Этот алгоритм ключевой генерации использует математические принципы RSA и обеспечивает создание безопасных цифровых подписей для защиты информации.

4.4.2. Процесс проверки подписи

Проверка подписи пары (M', S) происходит по определенному алгоритму:

1) с помощью криптографического преобразования подписи S с публичным ключом (e, r) восстанавливают хеш m учитывая, что $m = S^e \bmod r$;

2) находят вычисленное хеш-значение m' из входного M', используя хеш-функцию H: $m' = H(M')$. В случае корреляции вычисленных значений, а именно $H(M) = S^e \bmod r$, получатель утверждает подлинность пары (M', S).

В общем виде на рисунках 4.4.1 и 4.4.2, показан процесс формирования ЭЦП.

Рис. 4.4.1. Наложение цифровой подписи на документ

Пример.

Рассмотрим сценарий подписи сообщения «КФУ» и последующей проверки подлинности подписи.

1) Сначала выполним генерацию открытого и закрытого ключей. Выберем простые числа p = 13, q = 17 и вычислим произведение: r = 13 * 17 = 221. Затем найдем значение функции Эйлера от r: $\varphi(r) = (p-1) * (q-1) = 12 * 16 = 192$. Для открытой экспоненты e = 23, взаимно простой с φ(r), находим закрытый ключ d=167 (обратное число к e) с помощью расширенного алгоритма Евклида. Получаем открытый ключ (23, 221) и закрытый ключ (167, 221).

2) Следующим шагом будет получение хеш-функции для сообщения, которое в данном случае равно $H(M) = m = 363$, согласно предыдущему вычислению, в литературе ещё её называют хеш-образом (рис. 4.2.2).

Рис. 4.4.2. Наложение цифровой подписи на документ, на который наложен хеш-образ

3) Далее подпишем сообщение: формируем ЭЦП {«КФУ», S}, $S = H(M)^d \bmod r = m^d \bmod r = 363^{167} \bmod 221 = 116$.

По открытому ключу (23,221) можно проверить сформированную подпись {«КФУ»,116} или {363,116}:

$S = m^e \bmod r = 363^{23} \bmod 221 = 116$ – подпись верна.

Для упрощения формирования сообщения с ЭЦП, можно использовать одни и те же простые числа, что для формирования хеш образа, что и для ЭЦП:

Возьмём также то же сообщение «КФУ» – {12, 22, 21}. Только теперь выберем два простых числа p = 13, q = 17, найдем $n = p * q = 13 * 17 = 221$. Пусть $H_0 = 150$. Шифруем каждую букву:

$H_1 = (H_{1-1} + m_1)^2 \bmod n \equiv (150 + 12)^2 \bmod 221 = 166,$

$H_2 = (H_{2-1} + m_2)^2 \bmod n \equiv (754 + 22)^2 \bmod 221 = 172,$

$H_3 = (H_{3-1} + m_3)^2 \bmod n \equiv (776 + 21)^2 \bmod 221 = 55,$

В результате хеш образ для слова «КФУ» теперь будет $H(M) = m = 55$.

Тогда при тех значениях простых чисел функция Эйлера $\varphi(r) = 192$. Для e = 23, находим обратное число d = 167 и формируем ключи: открытый ключ (23, 221) и закрытый ключ (167, 221).

Формируем ЭЦП {«КФУ», S}:

$S = H(M)^d \bmod r = m^d \bmod r = 55^{167} \bmod 221 = 217$

По открытому ключу (23, 221) можно проверить:

$S = m^e \bmod r = 55^{23} \bmod 221 = 217$ – подпись верна.

Хотя данный вариант с одинаковыми простыми числами для хеш-образа и формирования ЭЦП работает быстрее, но криптостойкость будет чуть страдать.

Есть один метод улучшения криптостойкости всей конструкции хеш-образа с ЭЦП, выглядит это так:

$$S = H(M) * d \bmod r \equiv 363 * 167 \bmod 221 \equiv 60621 \bmod 221 = 67,$$

$$M = S^d \bmod r = 67^{167} \bmod 221 = 33$$

Формируем ЭЦП {S, M}, как видим на хеш-образ была наложена запись.

Для проверки подписи имеем входной набор {S, M}= {67, 33} и открытый ключ (23, 221)

$$M' = S^e \bmod r = 67^{23} \bmod 221 = 33,$$

сверяем M с M' и утверждаем, что подписи верны.

Как видно, хеш-образ не передаётся в открытом виде, он спрятан в формуле для S, что может повысить уровень защиты.

Здесь также можно брать одни и те же изначально простые числа, тогда:

$$S = H(M) * d \bmod r \equiv 55 * 167 \bmod 221 = 124,$$
$$M = S^d \bmod r = 124^{167} \bmod 221 = 197,$$
$$M' = S^e \bmod r = 124^{23} \bmod 221 = 197.$$

4.5. Метод цифровой аутентификации алгоритма DSA

Алгоритм цифровой подписи, известный под аббревиатурой DSA (Digital Signature Algorithm), был разработан в 1991 году и стал важным элементом в системе защиты информации. Этот алгоритм основан на схеме Эль-Гамаля и получил официальное одобрение от Национального института стандартов и технологий США (NIST) и был включен в стандарт для электронных подписей, известный как Digital Signature Standard (DSS). DSS подробно объясняет использование DSA в сочетании с хеш-функцией SHA, что обеспечивает целостность и подлинность данных.

DSA является открытым стандартом, что позволяет экспертам в области безопасности анализировать и проверять его на устойчивость к различным атакам и надёжность. Основным принципом данного алгоритма является вычисление дискретного логарифма, которое предоставляет значительную сложность для потенциальных взломов и, следовательно, обеспечивает высокий уровень защиты.

Создание DSA в значительной мере опиралось на исследования криптографов Тахера Эль-Гамаля и Клауса Шнорра. Их работы в области криптографии и цифровых подписей оказали значительное влияние на создание DSA, подчёркивая важность научных изысканий

в разработке надёжных технологий защиты информации. Эти исследования позволили разработать алгоритм, который и сегодня играет важную роль в защите цифровых данных.

Ключевой особенностью DSA является применение пар ключей: открытого и закрытого. Генерация цифровой подписи производится с использованием закрытого ключа, который должен храниться в секрете, в то время как открытый ключ служит для проверки подписи и может быть общедоступным. Хеш-функция SHA (Secure Hash Algorithm) усиливает процесс, гарантируя неизменность данных после их подписания.

Дополнительно, благодаря своей открытости, DSA многократно подвергался тщательному исследованию и тестированию, что позволило обнаружить и устранить возможные уязвимости, улучшив его работу. В сочетании с хеш-функцией SHA, алгоритм обеспечивает не только целостность информации, но и её подлинность, что особенно важно в условиях современных киберугроз.

Сегодня DSA остаётся значимым инструментом в арсенале криптографических методов защиты, широко применяемым в различных сферах, от электронной коммерции до государственных информационных систем. Это подчёркивает его надёжность и соответствие современным стандартам информационной безопасности.

4.5.1. Процедура создания ключей

Для создания цифровой подписи в рамках методологии DSA необходимо сгенерировать ключевую пару, состоящую из публичного и приватного ключей в соответствии с криптографическими принципами DSA. Этот процесс включает несколько этапов:

1. Выбрать большое простое число p такое, что p − 1 содержит простой делитель q, имеющий размерность L. Размерность L вы-

бирается приблизительно равной длине хеш-функции (например, 128 или 256 бит).

2. Найти число g, порядок которого в поле GF(p) равен q. следующему условию. Элемент g можно вычислить по формуле $g = h^{(p-1)/q} \bmod p$, где h – генератор поля GF(p).

3. Секретный ключ отправителя, обозначенный как x, определяется случайным образом в промежутке от 0 до q.

4. Открытый ключ вычисляется из закрытого ключа по формуле:

$$y = g^x \bmod p .$$

Чтобы верифицировать цифровые подписи авторов, все участники информационного обмена применяют публичный ключ $K_0 = (p, q, y)$, который автор отправляет своим контактам. Приватный ключ $K_C = x$ автор использует для создания подписи на своих сообщениях. Параметры p и q могут быть одинаковыми для группы пользователей, однако значения x и y являются уникальными для каждого пользователя индивидуально.

4.5.2. Процедура создания подписи

ЭЦП в алгоритме DSA создаётся по шагам:

1. На первом шаге вычисляют хеш-образ H(M), причем вычисления проводят по модулю простого числа q (одно из простых чисел при вычислении $n = p * q$).

2. Подбирается уникальное случайное число $0 < j < q$, для каждой подписи оно выбирается заново.

3. Пара (r, S) создаётся на основе:

$$r = \left(g^j \bmod p\right) \bmod q \quad \text{и}$$

$$S = (j^{-1} * H(M) + j^{-1} * x * r) \bmod q,$$

здесь j^{-1} – обратное число $(j^{-1} * j) \bmod q = 1$, но и одновременно получается $j^{-1} = b * a^{x(p-2)} \bmod p$. Если при вычислении получаем, что в паре (r, S), которая и есть ЭЦП, одно из значений равно нуль, то

нужно заново подобрать число j и пересчитать. Итоговое сообщение с подписью будет представлять тройку {r, S, M}.

4.5.3. Проверка подписи

При проверке ЭЦП на подлинность, сначала из полученного сообщения {r, S, M′} вычисляется хеш-образ H(M′), затем находят значение μ и сравнивают его с r:

$$\delta = S^{-1} \bmod q, \quad S^{-1} - \text{обратное к S};$$

$$\alpha = (H(M') * \delta) \bmod q;$$

$$\beta = (r * \delta) \bmod q;$$

$$\mu = \left(\left(g^{\alpha} * y^{\beta}\right) \bmod p\right) \bmod q.$$

Пример.

Подпишем сообщение «КФУ» с использованием алгоритма DSA и проверим выполненную подпись. Тогда по шагам, представленным ранее, сгенерируем ключи: публичный и приватный. Выберем простые p = 367 и q = 61, причем так чтобы число «p − 1» делилось на «q» без остатка $\frac{p-1}{q} = \frac{366}{61} = 6$.

Вычислим: $g = 2^{(367-1)/61} \bmod 367 = 64$. Далее выберем случайное число x = 35, которое будет секретным ключом. Вычислим для него открытый ключ по формуле:

$$y = g^{x} \bmod p = 64^{35} mod\ 367 = 323.$$

Теперь подсчитаем ЭЦП:

1. Строим H(M), но теперь с учетом нового выбранного q=61:

$H_1 = (H_{1-1} + m_1)^2 \bmod n \equiv (150 + 12)^2 \bmod 61 = 14,$

$H_2 = (H_{2-1} + m_2)^2 \bmod n \equiv (14 + 22)^2 \bmod 61 = 15,$

$H_3 = (H_{3-1} + m_3)^2 \bmod n \equiv (15 + 21)^2 \bmod 61 = 15.$

В результате новый хеш-образ для слова «КФУ» теперь будет $H(M) = 15$.

2. Подберём уникальное случайное при условии $0 < j < q$, пусть $j = 17$, и вычислим для него сразу его обратное $j^{-1} = 18$, т. к. $(17 * 18) \bmod 61 = 1$;

3. Пара (r, S) создаётся на основе:

$$r = \left(g^{j} \bmod p\right) \bmod q = (64^{17} \bmod 367) \bmod 61 \equiv 211 \bmod 61 = 28$$

$$S = (j^{-1} * H(M) + j^{-1} * x * r) \bmod q \equiv (18 * 15 + 18 * 35 * 28) \bmod 61 = 37,$$

Так как оба полученных значения r и s не равны 0, то подпись будет равна паре значений (28, 37), и отправляемое сообщение будет иметь вид: {КФУ, 28, 37}. Для проверки подлинности подписи получатель выполняет следующие действия. Сначала он вычисляет хеш-образ сообщения «КФУ», которое равно 64. Далее находим значение μ и сравниваем его с r:

$$\delta = S^{-1} \bmod q, \quad S^{-1} - \text{обратное к } S;$$

Вычисляем обратное $S^{-1} = 33$, проверим

$$\delta = S^{-1} * S \bmod q \equiv 33 * 37 \bmod 61 = 1,$$

$$\delta = S^{-1} \bmod q \equiv 33 \bmod 61 = 33;$$

$$\alpha = (H(M') * \delta) \bmod q \equiv (15 * 33) \bmod 61 = 7;$$

$$\beta = (r * \delta) \bmod q \equiv (28 * 33) \bmod 61 = 9;$$

$$\mu = \left(\left(g^{\alpha} * y^{\beta}\right) \bmod p\right) \bmod q \equiv \left((64^{7} * 323^{9}) mod\ 367\right) mod\ 61 \equiv$$

$$\equiv 211\ mod\ 61 = 28.$$

Подпись верна!

4.6. Индивидуальные задания к лабораторной работе № 4

1. Реализуйте программу с визуальным интерфейсом вычисления и проверки электронной цифровой подписи для сообщений русского алфавита методом RSA.

2. Реализуйте программу с визуальным интерфейсом вычисления и проверки электронной цифровой подписи для текста на русском языке с учётом пробелов методом DSA. Составьте отчет о проделанной работе.

ЛАБОРАТОРНАЯ РАБОТА № 5. РЕАЛИЗАЦИЯ АЛГОРИТМА ФЕРМА И ТЕСТОВ ПРОСТОТЫ

Цель работы: изучение и реализация алгоритмов факторизации и проверки целых чисел на простоту.

5.1. Теоретический материал. Метод Ферма

Пусть задано нечетное составное целое число n есть произведение двух простых чисел. Поставлена задача найти разложение $n = p * q$ в произведение двух сомножителей. Такая задача называется факторизацией числа, то есть разложением числа на множители. В настоящее время многие методы факторизации используют идеи, предложенные Пьером Ферма. Алгоритм Ферма состоит в нахождении пар чисел A и B таких, чтобы имело место равенство:

$$n = A^2 - B^2;$$

Выполним для этого следующие операции:

Шаг 1: найти целую часть от квадратного корня из n: $x_0 = [\sqrt{n}]$.

Шаг 2: для $x_i = x_0 + i,\ i = 1,\ 2, \ldots$ найти значения по формуле функции $q(x_i) = x_i^2 - n$. Процесс поиска продолжается, пока последующее значение $q(x_i)$ не окажется равным полному квадрату.

Пусть функция $q(x_i)$ является полным квадратом $q(x_i) = B^2$. Определим A равным x_i, тогда из равенства $A^2 - n = B^2$ получим $n = A^2 - B^2$ или $n = (A + B)(A - B)$. Таким образом, получим требуемое разложение $n = pq$.

Пример.

Задано число n = 876. Факторизуем его методом Ферма. Следуя шагу 1, найдём целую часть $x_0 = \lceil \sqrt{n} \rceil$. Процедура поиска делителей n представлена в таблице 15:

Таблица 15

Поиск делителей

x	q(x)	$\sqrt{q(x)}$
30	24	4,9
31	85	9,22
32	148	12,17
…	…	…
75	4749	68,91
76	4900	70

Процедура остановится, когда будет найдено целое значение $\sqrt{q(x)} = 70$. Тогда $n = 76^2 - 70^2 = 876 = 73 * 12$. Всего понадобилось 47 итераций, на каждой из которых выполнялась операция возведение в квадрат, разность и извлечение корня.

5.2. Оценка эффективности метода Ферма

Когда значение параметра q почти достигает единицы, а величина p становится практически равной числу n, алгоритм начинает демонстрировать меньшую эффективность по сравнению с методом пробных делений. В подобных обстоятельствах необходимо отметить, что производительность алгоритма менее оптимальна. Если исходить из предположения, что переменная A представляет собой среднее арифметическое значений p и q и вычисляется по формуле $(p + q)/2$, то можно определить, сколько шагов потребуется для успешного выполнения метода Ферма.

Этот метод опирается на последовательные вычисления, в которых каждое новое вычисление зависит от результатов предыдущих шагов, а количество шагов определяется начальными значениями параметров p и q.

Для глубокого понимания поведения алгоритма при этих условиях важно учитывать, что эффективность метода Ферма сильно

зависит от параметров и их взаимосвязи. Среднее арифметическое значение A, вычисляемое как $(p+q)/2$, играет важную роль в определении общего количества шагов, необходимых для выполнения метода. Колебания в значениях p и q могут существенно воздействовать на эффективность алгоритма, делая его более или менее производительным по сравнению с пробными делениями.

$$\mathrm{Iter}(n) = \frac{p+q}{2} - \lceil \sqrt{n} \rceil \approx \frac{n}{2} - \lceil \sqrt{n} \rceil.$$

Следовательно, общепринятая оценка сложности алгоритма представляется в виде O(n). Для достижения эффективности по методу Ферма, необходимо чтобы $\mathrm{Iter}(n) \ll \lceil \sqrt{n} \rceil$, отсюда число p должно быть меньше, чем четырехкратный $\lceil \sqrt{n} \rceil$.

Алгоритм Ферма экспоненциально сложен, поэтому его использование не эффективно при разложении на множители больших чисел. Но есть методы, повышающие эффективность: например, можно на первом шаге произвести пробное деление факторизуемого числа в диапазоне от двух до заданного числа B.

При таком подходе исключаются все небольшие делители числа n до значения B включительно, что уменьшает размер задачи перед применением более сложного алгоритма поиска. Это сокращает количество итераций и увеличивает общую производительность и скорость работы метода Ферма, делая его более пригодным для практического использования в случаях, когда раскладываемое число не слишком велико.

5.3. Тест простоты Миллера-Рабина

Теорема.

Нечетное число $n \geq 3$ является составным, тогда и только тогда, когда это число является полным квадратом или существуют два натуральных числа x и y такие, что:

$$x^2 \equiv y^2 \bmod n \quad \text{и} \quad x \neq \pm y \bmod n.$$

Доказательство.

Если выполняются вышеописанные условия, то НОД(n, $x^2 - y^2$) $\neq 1 \neq$ n . Напротив, если $n = p * q$, где $p > q$, то зададим переменные такими: x = (p + q) / 2 и y = (p − q) / 2. Отсюда, условия: $x^2 \equiv y^2 \bmod n$ и $x \neq \pm y \bmod n$ будут выполняться.

5.3.1. Алгоритм проверки числа на простоту.

Задано нечетное число n. Требуется определить, составное оно или нет. Запишем число $n - 1 = 2^s * d$, где d – нечетно. Произвольное число $a \in Z_n^*$ называют свидетелем простоты n, если выполнено хотя бы одно из условий:

1. $x = a^d \equiv \pm 1 (\bmod n)$, или
2. $(\exists k,\ 0 < k < s)\ x^{2^k} \equiv -1 \ (\bmod n)$.

Иначе a называется свидетелем непростоты n.

Докажем, что если найдется хотя бы один свидетель непростоты числа n, то n – составное.

Предположим, что для какого-то $a \in Z_n^*$ оба вышеописанных условия не выполняются, тогда последовательность: $x_0 = a^d \ (\bmod n), x_1 = x^2{}_0 (\bmod n), \ldots . mx_{s-1} = x^2{}_{s-2} (\bmod n)$ не содержит единиц.

Вычислим x_s , равное $x_{s-1}^2 \ (\bmod n) \neq 1$. Но, если число n – простое, тогда по малой теореме Ферма $x_s = a^{d*2^s} = a^{n-1} \equiv 1 \bmod n$, следовательно n – составное.

Можно доказать, что число свидетелей простоты не простого числа не превышает $\varphi(n)/4$, где $\varphi(n)$ – функция Эйлера, поэтому вероятность ошибки в одном раунде теста меньше 0,25. При выполнении k раундов с различными базами a вероятность ошибки будет меньше или равна $1/4^k$. не более чем 4^k . Таким образом, тест Миллера-Рабина позволяет успешно проверять числа больших порядков на простоту.

5.3.2. Тест Миллера–Рабина

Зададим нечетное число $n \geq 3$, запишем число $\mathrm{n}-1$ в виде $\mathrm{n}-1=2^{\mathrm{s}} * \mathrm{d}$, где d – нечетно. Далее для каждого числа a из интервала $3 \leq a < r+1$ (r – число раундов теста) выполним проверку:

1.) Вычислим $\mathrm{x}_0 = \mathrm{a}^{\mathrm{d}} (\mathrm{mod}\ \mathrm{n})$.

2.) Проверим условие $\mathrm{x}_0 \in \{1,\ \mathrm{n}-1\}$. Если оно выполнится, тогда a является свидетелем простоты. Перейдем к следующему a. В противном случае проверим, находится ли число $\mathrm{n}-1$ в последовательности $\{\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_{\mathrm{s}-1}\}$, где каждое следующее x вычисляется по формуле:

$$\mathrm{x}_{i+1} = \mathrm{x}^2{}_{\mathrm{i}} (\mathrm{mod}\ \mathrm{n}).$$

Если 1) или 2) выполнено, число a становится свидетелем простоты числа n. Повторим эту процедуру для каждого значения a.

Если в ходе теста будет найдено хотя бы одно значение a, не являющееся свидетелем простоты, то тест останавливается с сообщением «число n составное». Если же после выполнения r раундов все значения a подтвердят простоту числа n, то число n вероятно простое.

Пример.

Проверим на простоту число n = 877.

Запишем число $\mathrm{n}-1=2^{\mathrm{s}} * \mathrm{d}$ в виде $\mathrm{n}-1=2^2 * 219$.

Выполним тест Миллера-Рабина для a = 2:

$$\mathrm{x}_0 = 2^{219} (\mathrm{mod}\ 877) = 151 \neq 1 \neq n-1,$$

$$\mathrm{x}_1 = \mathrm{x}^2{}_0 (\mathrm{mod}\ 877) = 151^2 (\mathrm{mod}\ 877) = 876 \neq 1 \neq n-1,$$

$$\mathrm{x}_2 = \mathrm{x}^2{}_1 (\mathrm{mod}\ 877) = 876^2 (\mathrm{mod}\ 877) = 1.$$

Элементы $\{\mathrm{x}_\mathrm{i}\}$ с индексами $i \geq 2$ равны 1, значит в последовательности $\{\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_{\mathrm{s}-1}\}$ отсутствует $\mathrm{n}-1$, откуда следует, что число a = 2 является свидетелем непростоты числа n. Число n – составное.

5.3.3. Оценка эффективности теста Миллера-Рабина

Лемма, доказанная Майклом Рабином, строится на предположении истинности расширенной гипотезы Римана (РГР). Эта гипотеза уже долгое время считается одной из центральных в аналитической теории чисел, которая изучает природу и закономерности распределения простых чисел вдоль числовой оси. Расширенная версия гипотезы Римана не только предполагает выполнение базовых положений гипотезы Римана, но и вводит более строгие критерии, уточняющие распределение этих фундаментальных чисел.

Лемма Рабина

Пусть задано некоторое нечетное число n *и* $n - 1 = 2^s * d$*, где* d *– нечетно, тогда в предположении истинности РГР если для каждого числа* a *в диапазоне* $0 < a < 2 * (log_2(n))^2$ *выполняется одно из условий:*

1) $a^d \equiv 1\ (mod\ n)$*, или*

2) $a^{2^k * d} \equiv -1\ (mod\ n)$ *для некоторого* $0 \leq k < s$*,*

тогда исследуемое число является простым.

Оценка, приведенная в лемме Рабина, не может быть применена для решения реальных практических задач из-за необходимости доказательства расширенной версии гипотезы Римана. Сама гипотеза Римана является одной из самых известных и сложных нерешенных проблем в области математики.

Эта гипотеза относится к числу «проблем тысячелетия» и привлекает большой интерес со стороны математиков всего мира.

Пока не будет найдено окончательное доказательство РГР, найденная Рабиным, полиномиальная оценка остается лишь теоретической конструкцией. Несмотря на это, её влияние на современные математические исследования остается значительным, и её практическое применение будет возможно лишь после разрешения одной из величайших загадок в истории математической науки.

Кроме того, хотя теоретическая полиномиальная оценка представляет интерес для исследователей, на практике она кажется чрезмерно высокой для выполнения большинства реальных вычислений. В повседневных вычислительных задачах, как правило, используется более реалистичная оценка сложности, которая обычно выражается через более консервативную асимптотическую границу порядка $O(\log_2 n)$. Этот факт подчеркивает значительное различие между теоретической значимостью некоторых математических предположений и их практической применимостью в текущих научных и инженерных задачах.

5.4. Вероятностный тест простоты Соловея-Штрассена

Тест на простоту чисел, который был разработан такими известными математиками, как Роберт Мартин Соловей и Фолькер Штрассен, активно применяет основания, вытекающие из малой теоремы Ферма. Согласно этой теореме, при определенных условиях можно утверждать, что число является простым или составным. Кроме того, алгоритм учитывает характеристики, присущие символу Якоби, который является важным инструментом в теории чисел, особенно в контексте квадратичных вычетов и невычетов.

Теорема

Если n – нечетное составное число, то количество целых чисел $a < n$, взаимно-простых с n, удовлетворяющих сравнению:

$$A\frac{(n-1)}{2} \equiv (a * n)(mod\ n)\ a^{\frac{n-1}{2}} \equiv \left(\frac{a}{n}\right)\ mod\ n \qquad (*)$$

не превосходит $\frac{n}{2}$.

Здесь $\left(\frac{a}{n}\right)$ – символ Якоби.

5.4.1. Алгоритм Соловея-Штрассена

Для успешного старта алгоритма Соловея-Штрассена необходимо вначале определить целочисленный параметр $k \geq 1$. Этот параметр определяет количество итераций, которые должны

быть выполнены в рамках теста на простоту для числа n. Каждая из этих k итераций включает в себя определённый строгий набор шагов. Основная задача этих операций – проверить, является ли заданное число n простым или составным.

Когда параметр k определён, алгоритм проводит k независимых тестов числа n. В каждой из этих итераций выполняются определённые вычисления, в рамках которых алгоритм исследует число n на основе различных вероятностных критериев и статистических методов. Следует подчеркнуть, что точность и надёжность полученных результатов напрямую связаны с количеством итераций k. Чем больше значение k, тем выше вероятность получения точных результатов теста.

Во время каждой итерации алгоритм может выполнять такие шаги, как выбор случайного базиса, вычисление символа Якоби и сравнение полученных результатов с исходными условиями простоты. Последовательное выполнение этих шагов является важнейшей частью алгоритма, что обеспечивает его способность корректно определять, простое число n или составное, на основе строго математически обоснованных процедур. Итог выполнения всех k итераций сводится к общему заключению, позволяющему с высокой точностью определить, является ли число n простым или составным.

На каждом этапе тестирования производится следующий набор действий:

1. С помощью генератора случайных чисел выбирается число a, которое удовлетворяет условию $a < n$.

2. Для выбранного числа a и числа n вычисляется их наибольший общий делитель (НОД.), обозначенный как d. Это делается для того, чтобы проверить, нет ли простого пути исключить число n из простых чисел. Если d более 1, это является доказательством, что n составное, так как у него есть общий делитель с a, который не равен единице.

3. Если НОД. равен 1, то переходят к следующему этапу проверки, который состоит в проверке выполнения равенства (∗). Это условие необходимо для подтверждения простоты числа n с использованием специфических свойств числовых операций по определенному модулю. Несоответствие равенства (∗) свидетельствует о том, что число n отнюдь не простое, а является составным.

4. Если равенство (∗) выполнится, то число a считается свидетельством простоты числа n для данного теста.

После проведения k таких раундов, если в каждом раунде число n успешно проходит проверку, собирается статистика свидетельств простоты n. Если n находит k свидетелей простоты на протяжении всех k раундов, то предполагается, что число n с большой вероятностью является простым. Этим методом Соловея-Штрассена определяется вероятностный характер простоты числа, что делает его одним из ключевых инструментов в области криптографии и теории чисел для проверки простоты больших чисел.

5.5. Общая аудиторная работа

Задание. Напишите код программы с визуальным интерфейсом, который реализует проверку заданного числа на простоту по алгоритму Ферма.

5.6. Индивидуальные задания к лабораторной работе № 5

Реализуйте программу с визуальным интерфейсом для тестирования числа на простоту по методам Ферма, Миллера-Рабина и Соловея-Штрассена. Оценку методов проводить по временному счетчику. Составьте отчет по проделанной работе.

ЛАБОРАТОРНАЯ РАБОТА № 6. ШИФРОВАНИЕ НА ЭЛЛИПТИЧЕСКИХ КРИВЫХ

Цель работы: изучение криптографических алгоритмов, основанных на эллиптических кривых.

6.1. Криптографические протоколы распределения ключей

Протокол определяется как набор норм и правил, которые описывают: последовательность действий, выполняемых двумя или более участниками для достижения общей цели, структуру данных, обмениваемых между субъектами коммуникации, и процедуры, применяемые при возникновении ошибок. Криптографический протокол – это вид протокола, в процессе выполнения которого стороны используют методы криптографии. Ведущей задачей ключевых распределительных протоколов служит генерация общего секретного ключа с использованием операций, направленных на создание защищённого канала общения через генерацию и дистрибуцию сессионных ключей, а также аутентификацию передачи данных. Протокол Диффи-Хеллмана выделяется как один из наиболее известных методов создания и раздачи ключевых данных, работающий на основе асимметричного шифрования. Надёжность данного протокола обусловливается трудностью нахождения решения проблемы дискретного логарифмирования. Участники заранее согласовывают параметры системы – простое число n и генератор g конечного поля GF(n). В ходе реализации этого протокола, каждый участник выбирает случайное число (x для первого участника, y для второго), после чего стороны обмениваются данными в форме $g^x \bmod n$ и $g^y \bmod n$. Завершает процесс обмен данными вычисление общего секретного ключа как $g^{xy} \bmod n$. Информация, циркулирующая по сети, не дает злоумышленникам возможности воспроизвести ключ, поскольку задача вычисления дискретного логарифма является вычислительно сложной и не решается в короткие сроки.

6.2. Криптография на основе эллиптических кривых

Эллиптические кривые являются фундаментом в методике асимметричного шифрования. В основе алгоритма шифрования на эллиптических кривых построение публичных и приватных крипто - ключей связанных с геометрическим положением точек точками на эллиптической кривой, которая является сложным математическим объектом с детерминированными свойствами и структурой. Особенностью эллиптических кривых в криптографии является то, что они создают условия, при которых задача вычисления дискретного логарифма становится вычислительно трудоёмкой и, на текущий момент, не поддаётся решению с помощью субэкспоненциальных методов в пределах разумного времени. Это свойство делает криптосистемы, базирующиеся на эллиптических кривых, чрезвычайно устойчивыми к атакам, направленным на взлом шифрования.

Сложность криптографических задач, связанных с такими системами, прямо пропорциональна размеру группы точек, определённой на кривой. Ключевым преимуществом эллиптических кривых является тот факт, что для достижения уровня безопасности, сопоставимого с традиционной системой RSA, требуется использовать значительно меньшие группы. Это обстоятельство напрямую влияет на повышение эффективности систем с точки зрения использования и требований к ресурсам памяти, что особенно актуально в условиях ограниченных вычислительных ресурсов и ресурсов памяти таких, как мобильные устройства или встраиваемые системы. Таким образом, использование эллиптических кривых в асимметричном шифровании представляет собой не только мощный, но и эффективный подход к обеспечению цифровой безопасности в современном мире.

В криптографических системах используют специальное уравнение эллиптической кривой $E_p\ (a, b)$ над конечным полем F_p и имеет следующий вид:

$$y^2 = x^3 + a * x + b \pmod p \qquad (**),$$

здесь числа x, y, a, b принадлежат F_p. Такая форма кривой называется форма Вейерштрасса (рис. 6.2.1). Точкой эллиптической кривой является пара чисел (x, y), удовлетворяющая уравнению (**).

В контексте эллиптической кривой (ЭК) задействованы две основные операции: аддитивное сложение точек и удвоение точки. Для удобства вычислений вводится дополнительная точка O, которая объявляется бесконечно удаленной точкой или нулем, является точкой, куда сходятся все вертикальные линии ЭК. Когда три точки, находящиеся на одной прямой, суммируются, их общая сумма оказывается равной O. Правила сложения можно записать в виде:

$$(x, y) + O = (x, y), \qquad O + O = O, \qquad (x, y) + (x, -y) = O.$$

Суммой двух точек $P(x_1, y_1)$ и $Q(x_2, y_2)$ называется точка $-R(x_3, y_3)$, обратная точке R пересечения эллиптической кривой и прямой, проходящей через точки P и Q. Координаты полученной точки определяются формулами:

$$\begin{cases} x_3 = \lambda^2 - x_1 - x_2 \pmod p \\ y_3 = \lambda * (x_1 - x_3) - y_1 \pmod p \end{cases},$$

где $\lambda = \frac{y_2 - y_1}{x_1 - x_2} \bmod p$ – угловой коэффициент прямой, проходящей через точки $P(x_1, y_1)$ и $Q(x_2, y_2)$.

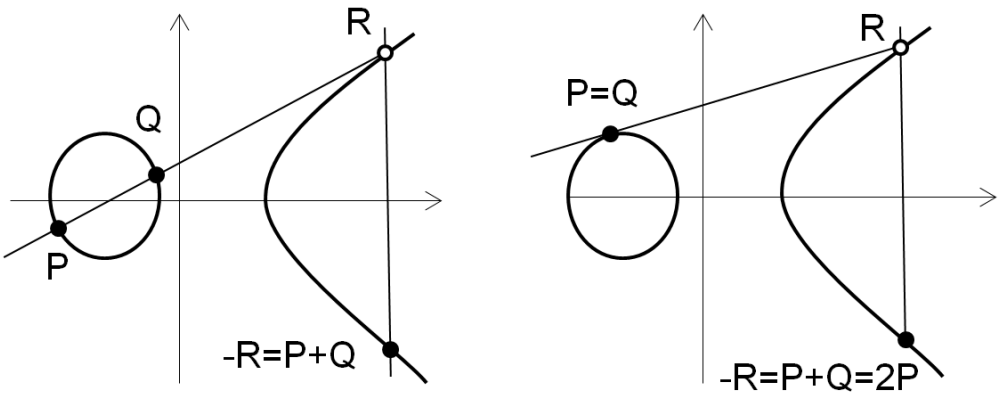

Рис. 6.2.1. Эллиптические кривые

Когда возникает задача удвоения точки $\mathrm{P}(\mathrm{x}_1, \mathrm{y}_1)$, то есть вычисления P + P, через точку P прокладывается касательная к эллиптической кривой. Результатом удвоения является точка $-\mathrm{R}(\mathrm{x}_3, \mathrm{y}_3)$, которая противоположна точке R, в которой касательная к точке P пересекает эллиптическую кривую. Расчёт координат x_3 и y_3 выполнен на основе определённых математических выражений:

$$\begin{cases} \mathrm{x}_3 = \lambda^2 - 2 * \mathrm{x}_1 (\mathrm{mod\ p}) \\ \mathrm{y}_3 = \lambda * (\mathrm{x}_1 - \mathrm{x}_3) - \mathrm{y}_1 (\mathrm{mod\ p}), \end{cases}$$

где $\lambda = \dfrac{3 * \mathrm{x}_1^2 + \mathrm{a}}{2 * \mathrm{y}_1} \bmod \mathrm{p}$

При расчете координат применяются принципы модулярной арифметики, при которых:

– все операции проводятся с использованием модуля p;

– процедура деления числителя на знаменатель заменяется операцией умножения числителя на обратное значение знаменателя в рамках модуля p;

– для перевода отрицательных значений в положительные используется добавление модуля p до достижения положительного результата.

Пример операций над точками эллиптической кривой.

Рассмотрим кривую EC: $\mathrm{y}^2 = \mathrm{x}^3 + 2 * \mathrm{x} + 6 \ (\mathrm{mod}\ 7)$ и точку $P = (2,\ 6)$. Проверим условие $P \in EC$:

$$4 * 2^3 + 27 * 6^2 \ (\mathrm{mod}\ 7) \ = \ 3.$$

Найдем сумму точек $P(5,\ 1)$ и $Q(4,\ 6)$.

$\lambda = \frac{6-1}{4-5} = \frac{5}{6} = 5 * 6^{-1} = 2 \bmod 7,$

$\mathrm{x}_3 = 2^2 - 5 - 4 = 2(\mathrm{mod}\ 7),\ \mathrm{y}_3 = 2(5 - 2) - 1 = 5(\mathrm{mod\ p})$

Координаты полученной точки (2, 5).

Для нахождения обратных значений применяют либо расширенный алгоритм Евклида, либо формулу $a^{-1} mod\ p \equiv a^{p-2} mod\ p$.

6.3. Алгоритм Диффи-Хеллмана на эллиптических кривых

Для создания надежного и безопасного канала связи между пользователями, именуемыми в данном контексте как A и B, они совместно принимают решение о выборе подходящей эллиптической кривой, обозначаемой как E, и определяют координаты базовой точки G(x, y) на этой кривой. На первоначальной стадии процесса, пользователь A генерирует случайное значение, обозначаемое как k_1. Затем, он выполняет вычисления точку k_1G, и координаты этой точки передает пользователю Б. В ответ на это Б также выбирает собственное уникальное значение k_2 , вычисляет точку k_2G и отправляет координаты полученной точки пользователю A.

Важно отметить, что данные, касающиеся выбранной эллиптической кривой, координаты точки G и результаты выполненных операций умножения, отправляются через каналы связи, которые могут быть не защищены, так как для вычисления k_1 и k_2 требуется вычислять дискретный логарифм, то есть решать вычислительно сложную задачу.

На последующей стадии, оба участника используют значения k_1 и k_2, а также координаты точек k_1G и k_2G, полученные в результате обмена через сеть, для расчета совместного секретного ключа K. Этот ключ получается путем умножения их соответствующих приватных значений k_1 и k_2 на полученные точки:

$$K(x,y) = k_1(k_2\mathrm{G}) = \mathrm{k}_2(\mathrm{k}_1G).$$

В силу ассоциативности операции умножения на эллиптической кривой, оба пользователя получают одинаковое секретное значение $K(x,\ y)$, которое может быть использовано для создания ключа симметричного шифрования.

Умножение точки на скаляр

Умножение точки на число реализуется последовательностью сложений и удвоений точки эллиптической кривой:

– вход: точка P, число m, представленное в двоичном виде $\mathrm{m} = (\mathrm{m_t}, \mathrm{m_{t-1}}, \ldots, \mathrm{m_1})$,

– выход: $Q = [m]P$.

1. $Q = \mathbf{O}$
2. Для каждого $i = t,\ t-1, \ldots, 1$ выполнить:

2.1. $Q = [2]Q$;

2.2. Если $m_i = 1$, то $Q = Q + P$;

3. Результат Q .

Данный алгоритм требует не более t сложений и t удвоений точек.

Пример.

Работа алгоритма вычисления m-кратной композиции на примере вычисления точки 29P. Здесь $29 = (11101)_2$, t = 5. На каждой итерации цикла алгоритма:

$[i = 5 m_5 = 1]$: Q ← O, Q ← Q +P = P;

$[i = 4 m_4 = 1]$: Q ← 2Q = 2P, Q ← Q +P = 3P;

$[i = 3 m_3 = 1]$: Q ← 2Q = 6P, Q ← Q +P = 7P;

$[i = 2 m_2 = 1]$: Q ← 2Q = 14P;

$[i = 1 m_1 = 1]$: Q ← 2Q = 28P;

Q ← Q+P = 29P.

Кратная точка вычислена с применением 5 умножений и 4 сложений точек.

6.4. Задание для выполнения лабораторной работы № 6

Порядок выполнения работы:

1. Выбрать коэффициенты a, b и модуль p эллиптической кривой, координаты x, y точки G, а также секретные значения k_1, k_2 абонентов в соответствии со своим вариантом (табл. 17).

2. Разработать программную реализацию метода Диффи-Хеллмана. Исходными данными являются параметры кривой, координаты точки и секретные значения каждого участника обмена. Результат работы программы – координаты произведения точки G на число, которые должны совпасть у каждого из участников.

3. Оформить отчет.

На рисунках 6.4.1 и 6.4.2 представлены примеры выполнения программ.

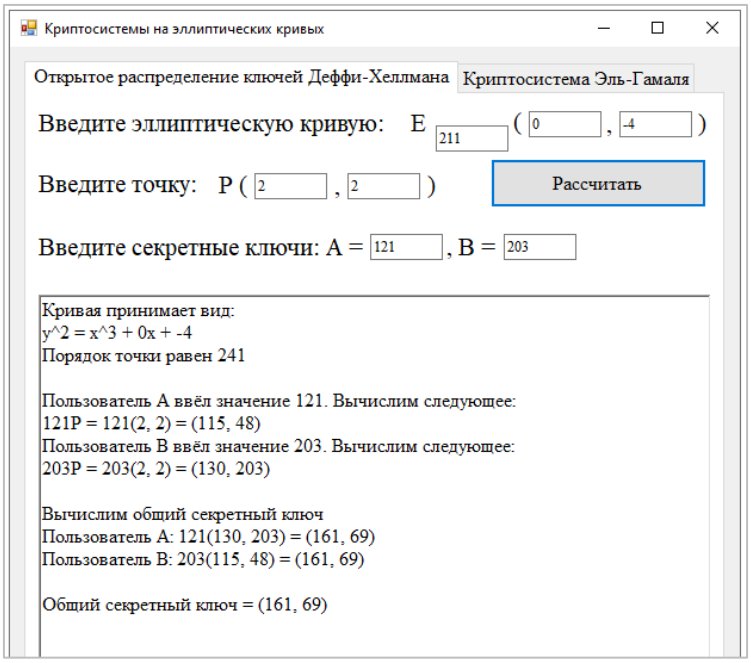

Рис. 6.4.1. Примеры выполнения программы

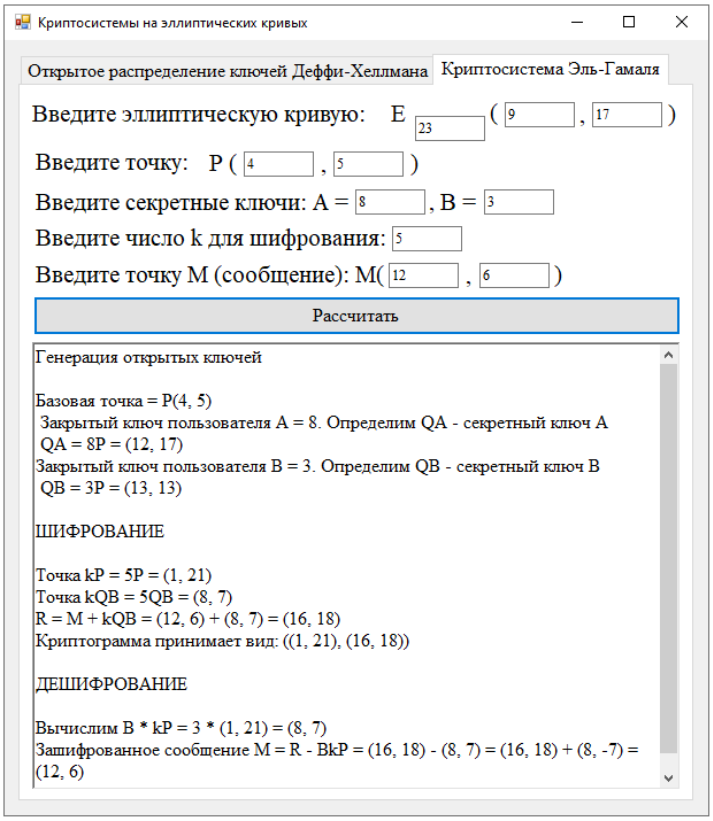

Рис. 6.4.2. Графический интерфейс программы шифрование на эллиптических кривых

Таблица 17

Задание к лабораторной работе № 6

№ варианта	a	b	p	G(x,y)	k1	k2
1	3	2	23	G(5,4)	10	18
2	4	7	47	G(6,22)	13	29
3	5	3	59	G(2,33)	21	34
4	3	7	21	G(6,22)	22	19
5	7	8	37	G(8,18)	14	23
6	2	9	67	G(22,56)	26	42
7	9	6	71	G(29,48)	31	53
8	11	5	79	G(31,37)	22	67
9	3	11	83	G(18,64)	27	72
10	1	13	97	G(5,76)	33	58
11	14	7	101	G(19,55)	36	65
12	15	3	113	G(34,67)	45	72
13	4	16	127	G(47,78)	58	85
14	9	8	131	G(52,89)	64	92
15	13	11	139	G(59,102)	75	118
16	18	14	149	G(66,115)	84	133
17	17	19	157	G(71,124)	93	144
18	21	6	163	G(80,132)	69	153
19	12	23	167	G(85,147)	98	136
20	22	9	173	G(90,156)	107	132
21	5	24	179	G(2,111)	112	157
22	25	4	191	G(3,123)	118	164
23	26	18	193	G(5,134)	127	172
24	7	28	197	G(9,143)	136	181
25	29	12	199	G(4,153)	145	188
26	11	31	211	G(7,163)	154	197
27	32	6	223	G(12,172)	165	208
28	14	35	227	G(18,181)	176	219
29	36	9	229	G(20,190)	185	228
30	17	38	233	G(22,199)	194	237

ЛАБОРАТОРНАЯ РАБОТА № 7. МЕТОД ПОЛЛАРДА. МЕТОД ВИЛЬЯМСА

Цель работы: изучение методов факторизации целого числа (p–1) Полларда, (p+1) Вильямса.

7.1. Теоретический материал

7.1.1. (p–1) - метод Полларда

Рассмотрим составное число n, которое требуется разложить на множители, и пусть p – простой делитель n, $1 < p < n$. Согласно малой теореме Ферма, для любого целого числа a, удовлетворяющего условию $1 \le a < p$, справедливо равенство $a^{p-1} \equiv 1 \pmod p$. Если же мы возьмем любое натуральное число M, которое делится на $p - 1$ (то есть $M = k(p-1)$ для некоторого k), то получится, что $a^M = (a^{p-1})^k \equiv 1^k \equiv 1 \pmod p$. Это равенство также означает, что $a^M - 1$ делится на p и, следовательно, может быть записано как $p \cdot r$ для целого числа r. Если p – делитель числа n, то очевидно, что p также делит наибольший общий делитель (НОД) чисел n и $a^M - 1$. Представим $p - 1$ в виде произведения простых чисел и их степеней: $p - 1 = p_1^{r_1} * p_2^{r_2} * \ldots * p_i^{r_i}$.

(p−1) – метод Полларда предлагает выбрать M как произведение максимально возможного количества таких простых множителей или их степеней, чтобы M оставалось кратным каждому из множителей $p_i^{r_i}$, входящих в разложение $p - 1$. Тогда НОД $(n, a^M - 1)$ будем содержать простой делитель числа n. Алгоритм разделён на два шага.

Первый этап.

1) Выберем некоторую границу B_1, например, $B = 10$.

2) Создадим набор P, включающий простые числа вместе с их степенями, которые не превышают границу B_1.

3) Вычислим произведение M всех элементов набора P. Для $B = 10$ набор $P = \{2, 3, 5, 7, 9\}$.

4) Выберем произвольное число a, например 2, и вычислим

$b = (\mathrm{a}^{\mathrm{M}} - 1) \bmod \mathrm{n}$. Для n = 10 961 значение b = 9264.

5) Рассчитаем наибольший общий делитель НОД(n, b), который при удачном выборе B может оказаться искомым делителем числа n.

В случае, когда первый этап алгоритма не приводит к нахождению нужного делителя, то есть НОД (n, a^{M}–1) рекомендуется либо увеличить границу B_1 , либо перейти ко второму этапу выполнения алгоритма.

Установим новую границу $B_2 > B$, например, $B_2 = B^2$.Выпишем простые числа из интервала (B, B_2): $q_0 < q_1 < \cdots < q\_k$.

Второй этап.

1) Определить $\mathrm{c}_0 = \mathrm{b}^{\mathrm{q_o}} \bmod \mathrm{n}$.

2) Найти НОД $(\mathrm{n}, \mathrm{c}_0 - 1)$, если НОД $(\mathrm{n}, \mathrm{c}_0 - 1) = 1$, то вычислить $\mathrm{c}_1 = \mathrm{b}^{\mathrm{q}_1} \bmod \mathrm{n}$ и снова найти НОД $(\mathrm{n}, \mathrm{c}_1 - 1)$, и т.д. Каждое значение для c_{k+1} будет вычисляться по формуле:

$\mathrm{c}_{k+1} = \mathrm{b}^{\mathrm{q}_{k+1}} \bmod \mathrm{n} \equiv \mathrm{b}^{\mathrm{q}_k + \delta_k} \bmod \mathrm{n}$, где $\delta_k = \mathrm{q}_{\mathrm{k}+1} - \mathrm{q}_{\mathrm{k}}$,

Далее можно записать:

$\mathrm{c}_{k+1} = \mathrm{b}^{\mathrm{q}_k + \delta_k} \bmod \mathrm{n} \equiv \mathrm{b}^{\mathrm{q}_k} * \mathrm{b}^{\delta_k} \bmod \mathrm{n} \equiv \mathrm{c}_k * \mathrm{b}^{\delta_k} \bmod \mathrm{n}$,

а здесь данные вычисления для $\mathrm{b}^{\delta_k} \bmod \mathrm{n}$ уже можно заранее просчитать, и для следующих значений c_{k+1} требуется только одна операция с умножением и нахождением числа по модулю n. Поэтому считается, что второй шаг в данном алгоритме самый быстрый.

Пример 1

Пусть n = 12311123123111111231

B = 50,

a = 2,

$\mathrm{B}_{\mathrm{n}} = 2500 = \mathrm{B}^2$.

Факторизация.

Шаг 1:

$\mathrm{M} = 2^5 * 3^5 * 5^2 * 9 * 11 * 13 \ldots * 49 = 774761126061499176600000$;

1. $\mathrm{a}^{\mathrm{M}} \bmod \mathrm{n} =$

$2^{774761126061499176600000} \bmod 12311123123111111231 =$

$= 3284866341935212 00;$

2. $НОД(n, a^M - 1) =$ $НОД(12311123123111111231,3284866341935211 99) = 1.$

Следовательно, переходим ко второму шагу;

Шаг 2:

3. Простые числа от b до b_n: $|53, 59 \ldots . 2473, 2477|$, $q = \{53, 59 \ldots . 2473, 2477\}$;

4. $c_0 = b^{q_0} \bmod n = 3284 \ldots 225^{53} \bmod(123 \ldots 231) =$ $12066249443640105 00;$

5. $НОД(n, c_0 - 1) = 1.$

Следовательно, для множества q: $c_k^{q_{k+1}-q_k} \bmod n$. И так далее, пока не найдем НОД $\neq 1$. На $k = 4$ находим 293.

Ответ: $12311123123111111231 = 293 * 420174850618126 7.$

7.1.2. Метод Вильямса (p+1)

Метод Вильямса, который часто обозначается как (p+1)-метод, представляет собой вычислительную стратегию, тесно связанную (p−1)-методом Полларда. Основная концепция этого метода заключается в использовании предположения о том, что число p+1, где p − простой делитель числа n, подлежащего разложению на множители, обладает свойством гладкости. Гладким числом в математике принято считать число, которое разлагается на простые множители таким образом, что каждый множитель не превышает определённой заранее заданной границы. В контексте метода Вильямса (p+1) предполагается, что для числа p уже существует гладкое разложение числа p+1. Это означает, что если мы можем предположить или установить, что p+1 гладко, то можно эффективно использовать этот факт для факторизации числа n, анализируя свойства и структуру числа p+1.

Этот метод представляет собой инструмент, который в некоторых случаях может быть эффективен, особенно когда прямое

применение других методов, таких как (p−1)-метод Полларда, не приводит к быстрой факторизации. В силу своей специфики и ориентации на определенные числовые свойства метод Вильямса (p+1) занимает важное место среди алгоритмов факторизации, предоставляя дополнительный инструмент для исследования чисел в сфере криптографии и теории чисел.

$$p + 1 = \prod_{i=1}^{k} q_i^{a_i}$$

Алгоритм Вильямса можно описать следующим образом:

1. Определить число B в качестве верхней границы для анализируемого числа n.

2. Сформировать последовательность простых чисел $2 < 3 < 5 < \ldots < p_m$, меньших B, а также последовательность степеней a_i таких, что для $p_i^{a_i}$ выполняется условие $p_i^{a_i} < B$.

3. Вычислить число R, являющееся произведением найденных простых чисел и их степеней, меньших B. В случае если p+1 является B – степенно-гладким, тогда R кратен p+1.

4. Случайным образом выбрать числа P и Q и построить последовательность Люка до значения u_R.

5. Затем вычислить $d = НОД(n, u_R) \, mod \, n$. Если d больше 1, то задача решена.

Показано, что при условиях взаимной простоты Q и p и значении $(\frac{P^2-4Q}{p}) = -1$, характеристики последовательности Люка позволяют найти нетривиальный делитель числа n.

Пример 2. Пусть n = 36863, B = 10, A = –10.

Факторизация:

1. b = 17217 – случайное число от 0 до n.
2. $d = НОД\,(b^2 - 4,\ n) = 1$. Следовательно, продолжаем.
3. Решето Эратосфена: $p : |2,3,5,7|$.
4. $2^3 * 3^2 * 5^1 * 7^1 = 2520 = R$.

5. Последовательность Люка

$$P = 1, \quad Q = -10$$

$$u_0 = 0\,,\ u_1 = 1$$

$$u_{i+1} = u_{i-1} * P + u_i * Q$$

$$u_R = u_{2520} = -5957 \ldots\ldots 2600.$$

6. $НОД\,(u_R\,, n\,) = 191.$

Ответ: найден делитель 191.

7.2. Задание для выполнения лабораторной работы № 7

1. Разработайте код программы, который реализует (p+1)-алгоритм Вильямса.

2. Разработайте код программы, который реализует (p–1)-метод Полларда.

Пример реализации программы для лабораторной работы 7 $(p-1)$-метод Полларда и (p + 1)-метод Вильямса (рис. 7.3).

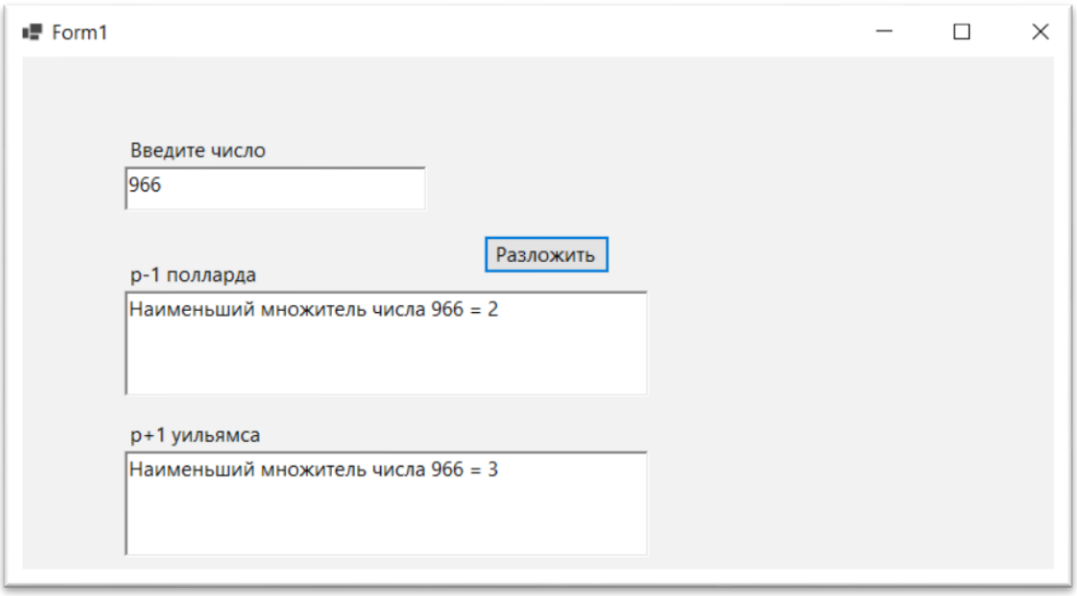

Рис. 7.3. Графический интерфейс программы (p–1)-метод Полларда

ЛИТЕРАТУРА

1. *Сингх С.* История шифров: от древнего искусства к квантовой криптографии / С. Сингх – М.: Альпина нон-фикшн, 2016. – 432 с.

2. *Янсон С.А.* История криптографии / С.А. Янсон, Ю.В. Цой. – М.: БИНОМ. Лаборатория знаний, 2015. – 290 с.

3. *Шнайер Б.* Прикладная криптография / Б. Шнайер. – 2-е изд. – М.: Триумф, 2002. – 784 с.

4. *Менезес А.* Введение в криптографию / А. Менезес, П. ван Ооршот, С. Ванстоун. – М.: Техносфера, 2005. – 832 с.

5. *Смарт Н.* Криптография / Н. Смарт. – М.: Техносфера, 2005. – 528 с.

6. *Ривест Р.* Метод и компьютерная программа для обеспечения информационной безопасности / Р. Ривест, А. Шамир, Л. Адлеман. – Берлин: Springer, 1978. – 12 с.

7. *Катц Дж.* Введение в современную криптографию / Дж. Катц, Й. Линдель. – М.: Вильямс, 2007. – 536 с.

8. *Вильямсон К.* Электронные подписи / К. Вильямсон. – М.: Вильямс, 2002. – 320 с.

9. *Граверсмайдт Г.* Методы проверки на простоту / Г. Граверсмайдт. – М.: Мир, 1994. – 214 с.

10. *Ишмухаметов Ш.Т.* Методы факторизации натуральных чисел: учебное пособие / Ш.Т. Ишмухаметов. – Казань: Казан. ун-т, 2011. – 190 с.

11. *Розен К.* Элементарная теория чисел и методы проверки на простоту / К. Розен. – М.: Мир, 2001. – 428 с.

12. *Сильверман Дж.* Арифметика эллиптических кривых / Дж. Сильверман. – М.: Мир, 1987. – 512 с.

13. *Коблиц Н.* Курс теории чисел и криптографии / Н. Коблиц.– 2-е изд. – М.: Издательский дом «МИФ», 2017. – 352 с.

14. *Сталингс У.* Защита информации и надежная криптография / У. Сталингс. – М.: Вильямс, 2003. – 944 с.

15. *Рожков М.П.* История криптографических методов / М.П. Рожков, В.Ю. Романов. – СПб.: Наука и Техника, 2011. – 264 с.

16. *Ингледью Дж.* Эллиптические кривые в криптографии / Дж. Ингледью, К. Качалова. – М.: Физматлит, 2014 – 240 с.

17. *Элдегжаусен Г.* Современные криптографические методы / Г. Элдегжаусен, Ж. Робинсон, Г. Цайц. – М.: Наука и Техника, 2010. – 670 с.

18. *Фельдман В.* Практическая криптография / В. Фельдман. – М.: Наука и Техника, 2012. – 400 с.

19. *Барнет А.* Современная криптография: Теория и практика / А. Барнет. – СПб.: Питер, 2011. – 576 с.

Учебное издание

Курбангалеев Артур Аскарович,
Ишмухаметов Шамиль Талгатович,
Рубцова Рамиля Гакилевна

ЛАБОРАТОРНЫЕ РАБОТЫ ПО ОСНОВАМ КРИПТОГРАФИИ

Учебно-методическое пособие