Назарбаев Интеллектуальная школа химико-биологического направления города Атырау

Шифр Вернама

Направление: Математическое моделирование экономических и социальных проектов

Секция: Прикладная математика

Выполнила: София Аманжан

Класс: 11D

Научный руководитель: Айгараев Б. К

Атырау, 2023

Оглавление

Абстракт

Целью данной исследовательской работы является улучшение общеизвестного симметричного криптоалгоритма – «Шифра Вернама». Был проведен анализ сильных и слабых сторон. При этом были выявлены две основные минусы данного алгоритма и пути их решения. Первая идея была направлена на модификацию шифрованного сообщения путем псевдо-увеличения размера. Такой подход повышает криптостойкость более чем в 100 000 раз. Далее в работе был рассмотрен способ безопасной передачи ключа через ассиметричное шифрование. Главная идея заключалась в сравнении остатков по модулю 257. Модуль 257 позволяет зашифровать до 256 различных символов. Сильная сторона данного способа в том, что невозможно восстановить остатки зная только степень числа и модуль.

Все идеи были реализованы на языке программирования «Python» и могут быть использованы после тестирования. Целевая аудитория это – банки, центры обслуживания населения, коммерческие организации и так далее.

На основе этого способа можно составить задачи олимпиадного характера и использовать в школьных кружках в качестве упражнений. Улучшенная версия «Шифра Вернама» отлично подойдет для всех организаций, где требуется шифрование данных.

Abstract

The purpose of this research work is to improve the well-known symmetric cryptoalgorithm - "Vernam Cipher". An analysis of strengths and weaknesses was carried out. At the same time, two main disadvantages of this algorithm and ways to solve them were identified. The first idea was to modify the encrypted message by pseudo-increasing the size. This approach increases cryptographic strength by more than 100,000 times. Further in the work, a method for secure key transmission through asymmetric encryption was considered. The main idea was to compare the remainders modulo 257. Modulo 257 allows you to encrypt up to 256 different characters. The strength of this method is that it is impossible to restore the remainders knowing only the degree of the number and the modulus.

All ideas were implemented in the programming language "Python" and can be used after testing. The target audience is banks, public service centers, commercial organizations and so on.

Based on this method, it is possible to compose tasks of an Olympiad nature and use them in school clubs as exercises. An improved version of the “Vernam Cipher” is perfect for all organizations that require data encryption.

Введение

В современном мире, где информационная безопасность играет важнейшую роль, шифрование данных становится неотъемлемой частью обеспечения конфиденциальности и целостности информации. Одним из наиболее надежных методов шифрования является шифр Вернама, который базируется на концепции одноразовых ключей и обладает высокой степенью криптографической стойкости.

Исследование шифра Вернама привлекает внимание ученых и специалистов в области криптографии, так как его принципы и подходы лежат в основе многих современных криптографических систем. Шифр Вернама был предложен в 1917 году инженером Гильбертом Вернамом и основан на использовании случайных ключей, длина которых равна длине шифруемого сообщения. Эта концепция обеспечивает абсолютную криптографическую стойкость, так как даже при наличии всех возможных вычислительных ресурсов невозможно вычислить исходный ключ и расшифровать сообщение без знания этого ключа.

Важной особенностью шифра Вернама является его невозможность криптоанализа методами классической криптографии, так как шифрование основано на случайности и не подвержено шаблонам, которые можно было бы использовать для расшифровки. Кроме того, использование одноразовых ключей делает данный метод особенно устойчивым к атакам типа "подмены" и "воспроизведения".

В данной исследовательской работе будут рассмотрены принципы функционирования шифра Вернама, его математическая модель, а также практические аспекты его применения. Будут проанализированы достоинства и ограничения данного метода, а также рассмотрены возможности его применения в современных информационных системах и областях, где требуется высокий уровень безопасности и конфиденциальности данных.

Исследование шифра Вернама имеет важное значение для развития криптографии и информационной безопасности. Понимание его основополагающих принципов и потенциала позволяет создавать более надежные системы шифрования и защиты данных в современном цифровом мире, где сохранение конфиденциальности становится все более актуальным и критически важным аспектом.

Помимо теоретических аспектов, исследование также уделяет внимание практическим реализациям шифра Вернама, включая алгоритмы генерации случайных ключей, методы обмена ключами между сторонами, а также анализ уязвимостей и сценариев атак, которые могут подвергнуть данный метод риску. Это позволяет разработчикам и аналитикам применять шифр Вернама с учетом его сильных сторон и ограничений.

Исследовательская часть

Глава 1. Обзор результатов известных исследований

Шифр Вернама представляет собой систему симметричного шифрования, которая была изобретена в 1917 году. Этот вид шифра относится к категории одноразовых блокнотов и оперирует с использованием булевой функции "исключающее или". Шифр Вернама служит примером криптосистемы с абсолютной криптографической надежностью, и несмотря на свою простоту, он является одной из базовых методов шифрования.

В современных условиях метод шифрования Вернама применяется редко, главным образом из-за размера ключа, который должен быть такой же длины, как и сообщение. Это означает, что производство, хранение и уничтожение ключей требует значительных ресурсов. Однако даже с учетом этих ограничений, шифры Вернама нашли свое применение в областях, где требуется высокий уровень безопасности для ограниченных объемов данных. Например, во времена Второй мировой войны англичане и американцы использовали шифры Вернама. Даже на правительственной "горячей линии" между Вашингтоном и Москвой использовались шифры Вернама по модулю 2, где ключевые материалы представляли собой перфорированные бумажные ленты с знаками ключевой последовательности.

Практически, данная система позволяет передать физически носитель информации с длинным случайным ключом, а затем обмениваться зашифрованными сообщениями. Отсюда и происходит концепция шифроблокнотов: отправитель и получатель оборудованы блокнотами, где каждая страница содержит ключи. Использованные страницы уничтожаются для поддержания безопасности.

Пример применения шифра Вернама:

Исходное сообщение представляется в виде последовательности битовых данных. Предположим, у нас есть текст, который компьютер интерпретирует в числовых кодах (где каждая буква имеет свой числовой эквивалент). Эти числа затем представлены в двоичном виде, то есть в виде битов данных. Этот этап описывает, как компьютер обрабатывает текстовую информацию.

Для шифрования используются случайно сгенерированные биты в качестве ключа. Если у нас есть, например, три числа, каждое из которых состоит из 8 бит, то для шифрования нам потребуется 24 случайных бита. Эти биты выбираются случайным образом и не несут никакого смысла:

10101101 01111010 10101011

Налагаем одну последовательность битов поверх другой и применяем алгоритм шифрования. Принцип работы шифра Вернама основан на операции

"исключающего ИЛИ" (XOR). Этот алгоритм анализирует каждую пару битов и определяет, равны они друг другу или нет. Если биты одинаковы, результат XOR будет 0, а если они различны — результат будет 1.

Вы можете проверить это самостоятельно: операция XOR задает вопрос "Эти биты разные?" Если ответ утвердительный, то результат XOR равен 1; если нет, то результат будет 0.

Буква K	**0**	**1**	**0**	**0**	**1**	**0**	**1**	**1**
Ключ	1	0	1	0	1	1	0	1
XOR (Они разные?)	1	1	1	0	0	1	1	0

Если мы воспользуемся этим методом для кодирования трех букв, то в результате получим три новых набора битов:

KOD (сообщение)	**01001011**	**01001111**	**01000100**
Ключ	10101101	01111010	10101011
Результат шифрования с помощью XOR	11100110	00110101	11101111

Таким образом, на входе у нас было 24 бита данных, и также на выходе получается 24 бита данных. Но важно отметить, что эти данные совершенно отличаются от исходных. Если провести обратное преобразование чисел в текст, то результат будет следующим:

KOD → æ5ï

Если у нас есть зашифрованное сообщение и соответствующий ключ, то дешифрование становится относительно простой задачей. Мы можем разработать алгоритм, который выполнит обратную операцию XOR для каждого бита сообщения. Вот как это может быть сделано:

Если бит ключа равен 1, то мы применяем операцию XOR к соответствующему биту зашифрованного сообщения. Если в зашифрованном сообщении был бит 1, то мы изменяем его на 0, и наоборот, если был бит 0, то меняем его на 1.

Если бит ключа равен 0, то мы оставляем соответствующий бит зашифрованного сообщения неизменным.

Повторяем этот процесс для каждого бита сообщения, чтобы получить исходный текст.

Благодаря высокой скорости обработки данных компьютер может выполнять миллионы таких операций в секунду. Основным условием для успешной

расшифровки остается наличие правильного ключа, который использовался при шифровании.

Шифр Вернама считается невзламываемым в контексте секретного чтения сообщения без наличия ключа. Это обусловлено рядом причин:

Исключающее ИЛИ (XOR) с случайными ключами: Каждый бит исходного сообщения XOR'ится с соответствующим битом случайного ключа. Ключ состоит из случайных битов, и следующий бит может быть любым. Это создает шумоподобный характер шифра, делая его непредсказуемым и трудным для анализа.

Сильная ключевая пространственная структура: Ключи генерируются случайным образом и имеют огромное количество вариантов, что делает перебор всех возможных ключей неосуществимым из-за огромного пространства ключей.

Криптографическая стойкость: Принципиальная случайность ключей и применение XOR создают высокий уровень криптографической стойкости. Поскольку XOR является обратимой операцией, даже небольшие изменения в ключе или входных данных полностью меняют вывод.

Отсутствие шаблонов и зависимостей: Так как ключи генерируются случайным образом, нет никаких шаблонов, по которым можно было бы анализировать шифр. Каждый бит зависит только от своего бита ключа и соответствующего бита сообщения.

Высокая длина ключа: Для каждого бита входных данных требуется свой бит ключа. Это делает перебор ключей практически невозможным, особенно при использовании достаточно длинных ключей.

Однако стоит отметить, что для сохранения абсолютной криптографической стойкости необходимо, чтобы ключи были действительно случайными и не использовались повторно. В практике шифрования также могут возникнуть проблемы, связанные с генерацией и передачей ключей, а также с физическими атаками на хранилище ключей.

Шифр Вернама, несмотря на свою высокую стойкость, обладает некоторыми минусами, которые ограничивают его практичное использование:

1. **Обмен секретными ключами:** Для безопасного использования шифра Вернама обе стороны должны обладать секретным ключом. Однако передача ключа между сторонами может быть небезопасной задачей, особенно в сетях или средах, где возможен перехват ключа злоумышленниками.

2. **Физическая безопасность ключей:** Чтобы обеспечить надежность шифрования, ключи должны храниться в абсолютной физической безопасности.

Это может быть сложно обеспечить, так как ключи могут быть украдены, подсмотрены, потеряны или даже перепутаны, что подрывает всю систему шифрования.

3. **Неудобство использования:** Процесс генерации, передачи и хранения ключей требует дополнительных усилий и ресурсов. Это может быть неудобным и затратным процессом, особенно при шифровании больших объемов данных.

4. **Ограниченная практичность:** В современных информационных системах часто требуется шифрование больших объемов данных в реальном времени. Применение шифра Вернама в таких случаях может оказаться неэффективным из-за необходимости обеспечения высокой безопасности ключей.

5. **Ограниченная масштабируемость:** Генерация и управление ключами для больших объемов данных или множества участников может стать сложной задачей, особенно если каждая пара участников должна иметь уникальный ключ.

В результате этих ограничений, хотя шифр Вернама обладает высокой криптографической стойкостью, он используется редко в современных сценариях и системах, где существуют более удобные и эффективные методы шифрования.

Глава 2. Методы решения поставленной цели

В данной главе будут представлены авторские идеи для усиления шифра Вернама.

2.1 Модификация «Шифра Вернама»

Для примера возьмем букву «К». В бинарной системе оно эквивалентно 01001011. Пусть произвольный ключ будет 11001110. Согласно шифру Вернама, наше зашифрованное сообщение будет 10000101.

1-я проблема.
Через метод «Брут Форс» вероятность расшифровки с первого раза 1/256 или нужно 256 попыток. Если у нас много компьютеров которые работают параллельно, то расшифровать будет не сложно.

Идея удвоения размера сообщения
Для решения этой проблемы будем добавлять по 2 произвольных бита через каждые 2 бита к шифрованному сообщению. Теперь злоумышленник не знает ни ключа, не истинного размера шифрованного сообщения. Пробуя различные варианты ключа, будет получать бессмыслицу. Так как даже с правильным ключом не будет выходит исходное сообщение. Размер сообщения будет увеличен в два раза. Перед злоумышленником теперь 2 задачи: найти ключ и истинный размер сообщения. Для этого нужно будет перебрать все варианты от 16 до 8 битов, если конечно он будет искать в направлении уменьшения. Это

$$\overline{A_2^8} + \overline{A_2^9} + \cdots + \overline{A_2^{16}} = 130816$$

вариантов. Теперь злоумышленнику нужно рассмотреть

$$130816 \cdot 256 = 33\,488\,896 \text{ вариантов.}$$

Это усложняет работу злоумышленника, но никак не влияет на процесс расшифровки. Мы просто удаляем через каждые два бита по два бита и будем использовать то, что осталось для расшифровки.

2-я проблема.
Главный минус шифра Вернама в том, что ключ сложно передать безопасным путем. Для безопасной передачи рекомендую асимметрично зашифровать ключ.

2.2 Асимметричное шифрование

Идея остатков по модулю 257
Возьмём приведенную систему вычетов по модулю 257. То есть остатки от 1 до 256. Сопоставим к ним 256 различных вариантов бинарного кода.
Например,

Ключ из 8 бит	Остаток
00000000	1
00000001	2

00011101	3
01111011	4
01101100	5
11101101	6
11111111	7
11011111	8
11100110	9

…

Возьмём два простых числа a, p *и степень* n, *где* $a < p, 1 \leq n \leq 256, p \geq 257$. *Если* $p > 257$, то остатки 1 и 257, 2 и 258 и так далее будут в одной коробке по модулю 256.

Числа $a, a^2, a^3, a^4, \dots, a^n, \dots a^{256}$ должны дать всевозможные остатки (кроме 0) по модулю 257, потому что 256 — это первообразный корень.

Определение. Число называется первообразным корнем по модулю p, если его показатель равен в точности $\varphi(p)$.

Определение. Значение функции Эйлера $\varphi(p)$ равно количеству натуральных чисел, не превосходящих p и взаимно простых с p.

Теорема. По любому простому модулю p существует первообразный корень.

Пара чисел a, p – ключ шифрования, а пара чисел b, p – ключ расшифровки.

$$ab \equiv 1(modp)$$
$$a^n \equiv x_1(modp)$$
$$b^n \equiv y_1(modp)$$
$$(ab)^n \equiv 1^n(modp)$$
$$x_1 y_1 \equiv 1(modp)$$
$$x_1 = \frac{(pk+1)}{y_1}$$

Рассмотрим пример:

$$a = 3, p = 257$$
$$3^1 \equiv 3(\text{mod } 257)$$
$$3^2 \equiv 9(\text{mod } 257)$$
$$3^3 \equiv 27(\text{mod } 257)$$
$$3^4 \equiv 81(\text{mod } 257)$$
$$3^5 \equiv 243(\text{mod } 257)$$
$$3^6 \equiv 215(\text{mod } 257)$$

…

Пусть ключ, состоящий из 8 битов (11100110) эквивалентен остатку «9», и я хочу передать остаток «9». Анализируя ПрСВ по модулю 257, я понимаю, что степень должна быть 2. Тогда по каналу передаются только степень n и число p.

Получатель имея на руках $b = 86, n = 2, p = 257$ начинает расшифровку.

$$86^2 \equiv 200(mod257)$$
$$200x \equiv 1(mod257)$$
$$x = 9$$

2.3 Реализация «Шифра Вернама» на языке Python

Чтобы провести криптоанализ модифицированной модели «Шифра Вернама», реализуем авторские идеи на языке программирования «Python». Сначала напишем код, в котором шифрованное сообщение будет в два раза длиннее исходного сообщения.

```
import secrets

def generate_random_key(length):
    return "".join(secrets.choice("ABCDEFGHIJKLMNOPQRSTUVWXYZ") for _ in
range(length))

def vernam_encrypt(message, key):
    if len(message) != len(key):
        raise ValueError("Message and key must be of the same length.")

    encrypted_message = ""
    for i in range(len(message)):
        encrypted_char = chr(ord(message[i]) ^ ord(key[i]))
        encrypted_message += encrypted_char + key[i]

    return encrypted_message

def vernam_decrypt(encrypted_message, key):
    if len(encrypted_message) % 2 != 0 or len(encrypted_message) // 2 != len(key):
        raise ValueError("Encrypted message length must be twice the length of the
key.")

    decrypted_message = ""
    for i in range(0, len(encrypted_message), 2):
        encrypted_char = encrypted_message[i]
        decrypted_char = chr(ord(encrypted_char) ^ ord(encrypted_message[i + 1]))
        decrypted_message += decrypted_char

    return decrypted_message

if __name__ == "__main__":
    try:
        message = input("Enter the message you want to encrypt: ")
        if not message:
            raise ValueError("Message cannot be empty.")

        key_length = len(message)
        key = generate_random_key(key_length)

        print("Generated key:", key)

        encrypted_message = vernam_encrypt(message, key)
        print("Encrypted message:", encrypted_message)
```

```
        decrypted_message = vernam_decrypt(encrypted_message, key)
        print("Decrypted message:", decrypted_message)

    except ValueError as e:
        print("Error:", e)
```

Скриншот примера на Python

```
PS C:\Users\User> & C:/Users/User/AppData/Local/Programs/Python/Python311/python.exe
Enter the message you want to encrypt: hot
Generated key: GKB
Encrypted message: /G$K6B
Decrypted message: hot
```

Теперь напишем код для вычисления остатков при введении следующих данных:

$$a, n, p$$

```
def extended_gcd(a, b):
    if a == 0:
        return b, 0, 1
    else:
        gcd, x, y = extended_gcd(b % a, a)
        return gcd, y - (b // a) * x, x

def modular_inverse(a, m):
    gcd, x, y = extended_gcd(a, m)
    if gcd != 1:
        raise ValueError("The modular inverse does not exist.")
    else:
        return x % m

def mod_exp(base, exponent, modulus):
    """
    Modular exponentiation function to compute (base^exponent) % modulus efficiently.
    """
    result = 1
    base = base % modulus

    while exponent > 0:
        if exponent % 2 == 1:
            result = (result * base) % modulus
        exponent = exponent // 2
        base = (base * base) % modulus

    return result

def calculate_values(a, n, p):
    """
    Function to calculate b, x_1, and y_1 based on the input values a, n, and p.
```

```
    """
    if a >= p:
        raise ValueError("a should be less than p.")

    # Verify that a and p are prime numbers.
    is_a_prime = all(a % i != 0 for i in range(2, a))
    is_p_prime = all(p % i != 0 for i in range(2, p))

    if not is_a_prime or not is_p_prime:
        raise ValueError("a and p should be prime numbers.")

    # Calculate b such that a * b ≡ 1 (mod p).
    b = modular_inverse(a, p)

    # Calculate x_1 ≡ a^n (mod p).
    x_1 = mod_exp(a, n, p)

    # Calculate y_1 ≡ b^n (mod p).
    y_1 = mod_exp(b, n, p)

    return b, x_1, y_1

# Example usage:
try:
    a = int(input("Enter the value of a (less than p and prime): "))
    n = int(input("Enter the value of n (a natural number): "))
    p = int(input("Enter the value of p (prime number): "))

    b, x_1, y_1 = calculate_values(a, n, p)
    print("b =", b)
    print("x_1 =", x_1)
    print("y_1 =", y_1)
except ValueError as e:
    print("Error:", e)
```

Скриншот примера на Python

```
PS C:\Users\User> & C:/Users/User/AppData/Local/Programs/Python/Python311/python.exe
Enter the value of a (less than p and prime): 3
Enter the value of n (a natural number): 2
Enter the value of p (prime number): 257
b = 86
x_1 = 9
y_1 = 200
```

Глава 3. Результаты работы и их обсуждение

С каждым годом вычислительные способности компьютеров растут с огромной скоростью. Это позволяет злоумышленникам более эффективно использовать метод «brute force». При попытке решить данную проблему было предложено удвоение размера исходного сообщения, что в свою очередь увеличивает надежность кода. Данная идея сильно увеличивает время расшифровки с помощью метода «brute force», но, к сожалению, будет занимать большое количество памяти на сервере из-за значительного размера исходного сообщения. С развитием скорости чтения данных на сервере и размеров памяти этим минусом можно будет пренебречь. Данную идею не сложно реализовать и внедрить в различных организациях. Написанный код на языке «Python» показал, что для этого не требуется большой объем написанного кода. Теперь нужно данный код протестировать на различных сайтах перед использованием в коммерческих или других целях.

Вторая проблема была решена через ассиметричное шифрование. При этом использовалось 256 остатков, что в свою очередь покрывает весь латинский алфавит, числа и различные символы. Иными словами, при шифровании можно будет использовать широкий арсенал различных символов для увеличения надежности шифра. Использование обратного остатка позволило легко написать код, а при передаче только степени и модуля невозможно будет взломать шифр. Зная только степень и модуль невозможно однозначно восстановить основание и остаток. Для увеличения криптостойкости нужно взять очень большие простые числа. При этом взлом может длиться более 100 лет, что свою очередь достаточно для хранения данных с не высокой степенью важности. Аналогичные алгоритмы широко используются в наше время. Например, криптоалгоритм RSA. При работе с ЭЦП-ключом для получения услуг ЦОНА вы могли увидеть два файла AUTH и RSA. Данный алгоритм широко известен в мире, поэтому различные специалисты мира в криптоалгоритмах постоянно ищут методы эффективного взлома. Создание иного алгоритма, который заставит злоумышленников искать новые способы взлома, является вынужденным шагом.

Полученные свойства в сравнении по модулю простого числа можно использовать для составления задач или в качестве упражнений в олимпиадных кружках.

Заключение

Со временем, вычислительные ресурсы компьютеров постоянно расширяют свои возможности с поразительной интенсивностью. Это, в свою очередь, создает среду, в которой киберпреступники могут наиболее эффективно применять метод "brute force". Попытка решить данную проблему включала в себя стратегию увеличения размера исходного сообщения вдвое, что, в конечном итоге, поднимает уровень надежности кодирования. Однако стоит учитывать, что данная идея значительно увеличивает время, необходимое для расшифровки методом "brute force". Единственным недостатком является потребность в большем объеме серверной памяти из-за увеличенного размера исходного сообщения. Но с развитием скорости чтения данных на серверах и увеличением объема памяти, этот недостаток постепенно уходит на второй план. Реализация данной идеи в различных организациях не представляет собой сложной задачи, как подтверждено написанным на языке "Python" кодом, который, несмотря на все, не требует обширных объемов написания кода. Важным этапом становится теперь тестирование этого кода на разнообразных веб-сайтах перед его применением в коммерческих или других целях.

Вторая проблема была решена через ассиметричное шифрование, при котором использовались 256 остатков, охватывающих весь латинский алфавит, числа и разнообразные символы. Иными словами, при шифровании появляется широкий арсенал символов, доступных для обеспечения надежности шифра. Использование обратного остатка упростило написание кода, и при передаче только степени и модуля, взлом шифра становится невозможным. Знание только степени и модуля не предоставляет однозначной возможности восстановить основание и остаток. Для максимальной криптостойкости необходимо выбирать очень большие простые числа, что делает взлом практически невозможным и может занять более столетия, что вполне достаточно для хранения данных с невысокой степенью важности. Подобные алгоритмы широко используются в современных приложениях, например, криптоалгоритм RSA. При работе с ЭЦП-ключом для получения услуг Центра обработки национальных ассоциаций (ЦОНА) пользователи могут заметить два файла: AUTH и RSA. Этот алгоритм широко известен во всем мире, поэтому различные специалисты в области криптографии постоянно ищут эффективные методы взлома. Создание нового алгоритма, который вынудит злоумышленников искать новые методы взлома, становится необходимым шагом в постоянно меняющемся мире кибербезопасности. Собранные свойства, сравниваемые по модулю с простыми числами, могут быть использованы для формулирования сложных задач и олимпиадных упражнений, способствуя развитию навыков криптографии.

Отзыв

Поздравляю автора исследовательской работы с проделанной работой по улучшению симметричного криптоалгоритма "Шифр Вернама". Работа представляет собой значительный вклад в область криптографии и показывает глубокое понимание принципов криптографической стойкости.

В абстракте четко очерчены цель и задачи исследования. Автор успешно проанализировал сильные и слабые стороны "Шифра Вернама" и выявил два ключевых минуса. Особенно интересен подход к псевдо-увеличению размера сообщения для повышения криптостойкости. Это действительно является инновационным и перспективным подходом, способным значительно усложнить задачу "brute force" атаки. Анализ сильных и слабых сторон является важным шагом для разработки новых методов криптографии.

Применение ассиметричного шифрования для безопасной передачи ключей также заслуживает внимания. Идея сравнения остатков по модулю 257 демонстрирует хорошее понимание математических основ криптографии. Этот подход позволяет обеспечить безопасную передачу ключа, трудно подверженную атакам.

Реализация данных методов на языке программирования "Python" говорит о том, что автор обладает не только теоретическими знаниями, но и практическими навыками. Тестирование на различных платформах и сайтах является следующим шагом для оценки эффективности и надежности предложенных методов.

Автор также правильно обратил внимание на перспективы применения улучшенного "Шифра Вернама" в различных организациях, включая сферы финансов, обслуживания населения и коммерческие организации. Подходы, представленные в работе, могут также найти свое применение в образовательных целях, таких как олимпиадные задачи.

С учетом вышеизложенного, я уверен, что данная исследовательская работа имеет значимость для области криптографии и может служить основой для дальнейших исследований в этой области. Желаю автору дальнейших успехов в его научных исследованиях.

Руководитель: Айгараев Б.К
учитель математики

Список использованной литературы

1. Фомичёв В. М. Дискретная математика и криптология: Курс лекций / под ред. Н. Д. Подуфалов — М.: Диалог-МИФИ, 2013. — С. 239—246. — 397 с. — ISBN 978-5-86404-185-7

2. Габидулин Э. М., Кшевецкий А. С., Колыбельников А. И. Защита информации: учебное пособие — М.: МФТИ, 2011. — 225 с. — ISBN 978-5-7417-0377-9

3. Мао В. Современная криптография: Теория и практика / пер. Д. А. Клюшина — М.: Вильямс, 2005. — С. 255. — 768 с. — ISBN 978-5-8459-0847-6

4. Robert L. Benson. The Venona Story (англ.) // Center for Cryptologic History National Security Agency. Архивировано 27 мая 2010 года.

5. Бабаш А. В., Гольев Ю. И., Ларин Д. А., Шанкин Г. П. Криптографические идеи XIX века // Защита информации. Инсайд — СПб.: 2004. — вып. 3.

6. Stallings, W. (2017). "Cryptography and Network Security: Principles and Practice." Эта книга предоставляет обширное введение в основы криптографии и криптоалгоритмов.

7. Paar, C., & Pelzl, J. (2010). "Understanding Cryptography: A Textbook for Students and Practitioners." Данное учебное пособие обращает внимание на основы криптографии и принципы криптоалгоритмов.

8. Schneier, B. (1996). "Applied Cryptography: Protocols, Algorithms, and Source Code in C." Это классическое руководство о криптографических алгоритмах и их применении.

9. Menezes, A., van Oorschot, P., & Vanstone, S. (1996). "Handbook of Applied Cryptography." Этот справочник предоставляет обширную информацию о различных криптоалгоритмах и протоколах.

10. Katz, J., & Lindell, Y. (2014). "Introduction to Modern Cryptography: Principles and Protocols." Эта книга охватывает современные методы криптографии и протоколы.

11. Boneh, D., & Shoup, V. (2004). "A Graduate Course in Applied Cryptography." Этот учебник предоставляет более глубокий анализ криптографических алгоритмов и их математических основ.

12. Ferguson, N., Schneier, B., & Kohno, T. (2010). "Cryptography Engineering: Design Principles and Practical Applications." Эта книга охватывает принципы проектирования криптоалгоритмов и их практическое применение.