Лабораторная работа №3. Шифрование гаммированием.

Предмет: Математические основы защиты информации и информационной безопасности

Александр Сергеевич Баклашов

Содержание

List of Figures

1 Цель работы

Рассмотреть и реализовать алгоритм шифрования гаммированием конечной гаммой.

2 Задание

Реализовать шифрование гаммированием конечной гаммой.

3 Теоретическое введение

3.1 Шифрование гаммированием конечной гаммой

Гаммирование конечной гаммой — это один из методов симметричного шифрования, который использует последовательность случайных символов (гамму), имеющую конечную длину, для шифрования и расшифрования сообщения. В данном методе каждый символ исходного текста комбинируется с соответствующим символом из гаммы с использованием операции побитового XOR (исключающее ИЛИ). Центральной идеей является то, что использование гаммы делает шифрованный текст статистически случайным и с трудом поддается анализу.

Теоретическое введение в гаммирование конечной гаммой включает следующие ключевые понятия и аспекты:

Симметричное шифрование: Гаммирование конечной гаммой относится к симметричным методам шифрования, где один и тот же ключ (гамма) используется как для шифрования, так и для расшифрования сообщения. Это отличается от асимметричных методов, где используются разные ключи для шифрования и расшифрования.

Последовательность гаммы: Гамма представляет собой последовательность символов (обычно битов или байтов), имеющую конечную длину. Для успешного шифрования и расшифрования длина гаммы должна быть такой же, как длина исходного текста.

Операция XOR: Для каждого символа исходного текста и символа гаммы выполняется операция XOR. Это операция, которая возвращает 1 (или true), если биты операндов разные, и 0 (или false), если биты одинаковые.

Ключевое пространство: Ключевое пространство метода гаммирования конечной гаммой зависит от длины гаммы. Чем больше длина гаммы, тем больше возможных ключей и, следовательно, тем выше стойкость шифрования.

Стойкость к криптоанализу: Стойкость шифра гаммирования конечной гаммой зависит от случайности и длины гаммы. Чем более случайной и длинной

является гамма, тем сложнее проводить атаки на шифр, такие как атаки методом частотного анализа.

Гаммирование конечной гаммой остается одним из важных методов симметричного шифрования, и его применение может быть найдено в различных областях, включая информационную безопасность, сетевые коммуникации, телекоммуникации и другие сферы, где требуется конфиденциальность и защита данных от несанкционированного доступа.

4 Выполнение лабораторной работы

4.1 Шифрование гаммированием конечной гаммой

4.1.1 Задача

Реализовать шифрование гаммированием конечной гаммой

4.1.1.1 Решение

Напишем функцию для шифрования текста (рис. 4.1)

In [1]:
```python
def encrypt(text, key):
    # Создаем словарь, который соотносит буквы с их номерами в алфавите
    alphabet = "абвгдежзийклмнопрстуфхцчшщъыьэюя"
    letter_to_number = {letter: index for index, letter in enumerate(alphabet, start=1)}

    encrypted_text = ""

    for i in range(len(text)):
        # Получаем номер буквы в тексте и ключе
        text_letter = text[i]
        key_letter = key[i % len(key)]

        text_number = letter_to_number.get(text_letter, 0)
        key_number = letter_to_number.get(key_letter, 0)

        # Вычисляем зашифрованную букву
        encrypted_number = (text_number + key_number) % 33

        # Получаем зашифрованную букву из словаря
        encrypted_letter = list(letter_to_number.keys())[list(letter_to_number.values()).index(encrypted_number)]

        encrypted_text += encrypted_letter

    return encrypted_text
```

Figure 4.1: Шифрование текста

Напишем функцию для дешифрования текста (рис. 4.2)

In [2]:
```python
def decrypt(encrypted_text, key):
    # Также создаем словарь, который соотносит буквы с их номерами в алфавите
    alphabet = "абвгдежзийклмнопрстуфхцчшщъыьэюя"
    letter_to_number = {letter: index for index, letter in enumerate(alphabet, start=1)}

    decrypted_text = ""

    for i in range(len(encrypted_text)):
        # Получаем номер зашифрованной буквы и ключа
        encrypted_letter = encrypted_text[i]
        key_letter = key[i % len(key)]

        encrypted_number = letter_to_number.get(encrypted_letter, 0)
        key_number = letter_to_number.get(key_letter, 0)

        # Вычисляем исходную букву
        text_number = (encrypted_number - key_number) % 33

        # Получаем исходную букву из словаря
        text_letter = list(letter_to_number.keys())[list(letter_to_number.values()).index(text_number)]

        decrypted_text += text_letter

    return decrypted_text
```

Figure 4.2: Дешифрование текста

Напишем реализацию шифрования гаммированием конечной гаммой с помощью функций (рис. 4.3)

In [3]:
```python
text = "приказ"
key = "гамма"

# Шифрование
encrypted_text = encrypt(text, key)
print("Зашифрованный текст:", encrypted_text)

# Дешифрование
decrypted_text = decrypt(encrypted_text, key)
print("Расшифрованный текст:", decrypted_text)
```

```
Зашифрованный текст: усхчбл
Расшифрованный текст: приказ
```

Figure 4.3: Реализация

5 Выводы

В ходе данной лабораторной работы я рассмотрел и реализовал алгоритм шифрования гаммированием конечной гаммой.

6 Библиография

1. Python documentation. [Электронный ресурс]. М. URL: Python documentation (Дата обращения: 28.09.2023).

2. Лабораторная работа №3. Шифрование гаммированием. - 3 с. [Электронный ресурс]. М. URL: Лабораторная работа №3. Шифрование гаммированием. (Дата обращения: 07.10.2023).