Математические основы защиты информации и информационной безопасности. Лабораторная работа №3

Шифрование гаммированием

Студент: Масолова Анна Олеговна НФИмд-02-21
Преподаватель: Кулябов Дмитрий Сергеевич

Содержание

List of Figures

List of Tables

1 Цель работы

Ознакомиться с шифрованием гаммированием на примере гаммирования конечной гаммой.

2 Задание

Реализовать алгоритм шифрования гаммированием конечной гаммой.

3 Теоретическое введение

Гаммирование, или Шифр XOR, — метод симметричного шифрования, заключающийся в «наложении» последовательности, состоящей из случайных чисел, на открытый текст. Последовательность случайных чисел называется гамма-последовательностью и используется для зашифровывания и расшифровывания данных. Суммирование обычно выполняется в каком-либо конечном поле.

3.1 Шифрование гаммированием

При шифровании гаммированием формируется `m` - разрядная случайная последовательность. Пусть `k` - передаваемое сообщение

$$k = k_1 k_2 ... k_i ... k_m,$$

а `p` - последовательность, которая является ключом:

$$p = p_1 p_2 ... p_i ... p_m,$$

тогда `i`-ый символ криптограммы будет равен:

$$c_i = p_i \oplus k_i,$$

где $\oplus$ - операция побитового сложения XOR [1]. В результате криптограмму можно записать следующим образом:

$$c = c_1 c_2 ... c_i ... c_m$$

Более подробно о шифровании гаммированием: [2]

4 Выполнение лабораторной работы

В рамках данной лабораторной работы был описан алгоритм шифрования гаммированием с конечной гаммой.

4.1 Описание реализации метода шифрования

В данной работе применяется схема однократного использования (рис. 4.1). К элементам ключа и исходного сообщения применяется побитовое сложение XOR, в результате чего формируется зашифрованное сообщение:

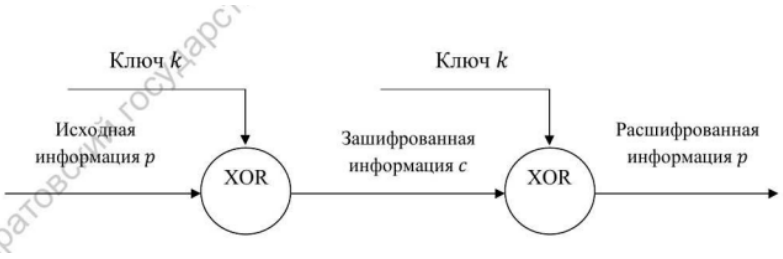

Figure 4.1: Схема однократного использования

Для того, чтобы применить операцию побитового сложения, необходимо, чтобы ключ и исходное сообщение были одной длины. Для достижения данной цели, ключ растягивается до тех пор, пока не сравняется длиной с исходным сообщением следующим образом: пусть сообщение будет

$$SECURITY,$$

длина ‘m’ которого равна 12, тогда ключ растягивается следующим образом:

$$KEY \rightarrow KEYKEYKE$$

Таким образом, к сообщению и ключу одинаковой длины можно применить операцию побитового сложения XOR.

4.2 Листинг

Код приведенной ниже программы реализован на языке python.

```
alphabet = "абвгдеёжзийклмнопрстуфхцчшщъыьэюя"
ignore_symbols = '!.,@#$%^&*()-+={}[]<>/ '

def format_gamma(gamma, message_length):
    new_gamma = ""
    for i in range(message_length):
        new_gamma += gamma[i % len(gamma)]
    return new_gamma

def get_indexes(text):
    indexes = []
    for letter in text:
        try:
            indexes.append(alphabet.index(letter) + 1)
        except ValueError:
            indexes.append(ignore_symbols.index(letter) + 10000)
    return indexes

def encrypt(message_indexes, gamma_indexes):
```

```
    indexes = []
    for ix, item in enumerate(message_indexes):
        if message_indexes[ix] > 10000 or gamma_indexes[ix] > 10000:
            result = item
            indexes.append(result)
            continue
     result = (message_indexes[ix] + gamma_indexes[ix]) % len(alphabet)
        indexes.append(result)
    return indexes

def to_text(indexes):
    text = ""
    for index in indexes:
        if index > 10000:
            text += ignore_symbols[index - 10000]
            continue
        text += alphabet[index - 1]
    return text

message = input("Введите сообщение: ")
gamma = input("Введите ключ (гамма): ")
new_gamma = format_gamma(gamma, len(message))
print("\nПреобразование {} -> {}".format(gamma.upper(), new_gamma.upper()))
message_indexes = get_indexes(message)
print("\nВаше сообщение:\n{} ({})".format(message.upper(), message_indexes))
gamma_indexes = get_indexes(new_gamma)
print("\nВаша гамма:\n{} ({})".format(gamma.upper(), gamma_indexes))
encrypted_message_indexes = encrypt(message_indexes, gamma_indexes)
encrypted_message = to_text(encrypted_message_indexes)
```

```
print("\nЗашифрованное сообщение:\n{} ({})"
    .format(encrypted_message.upper(), encrypted_message_indexes))
```

4.3 Полученные результаты

При запуске программы пользователю предлагается ввести сообщение, которое необходимо зашифровать, и ключ. После этого, если ключ и сообщения разной длины, из ключа исключаются лишние символы или добавляются в конец новые. Знаки препинания и пробелы, в свою очередь, игнорируются и остаются в неизменном виде. Для удобства, они нумеруются начиная с индекса 10000. В результате выполнения пользователь получает зашифрованное сообщение (рис. 4.2).

```
Введите сообщение: информационная безопасность
Введите ключ (гамма): криптография

Преобразование КРИПТОГРАФИЯ -> КРИПТОГРАФИЯКРИПТОГРАФИЯКРИ

Ваше сообщение:
ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ ([10, 15, 22, 16, 18, 14, 1, 24, 10, 16, 15, 15, 1, 33, 10022, 2, 6, 9, 16, 17, 1, 19, 15, 16, 19,
20, 30])

Ваша гамма:
КРИПТОГРАФИЯ ([12, 18, 10, 17, 20, 16, 4, 18, 1, 22, 10, 33, 12, 18, 10, 17, 20, 16, 4, 18, 1, 22, 10, 33, 12, 18, 10])

Зашифрованное сообщение:
ФЯЮЯДЬДЗЙДЧНЛР СШЧТББЖЧОЭДЁ ([22, 0, 32, 0, 5, 30, 5, 9, 11, 5, 25, 15, 13, 18, 10022, 19, 26, 25, 20, 2, 2, 8, 25, 16, 31, 5,
7])
```

Figure 4.2: Результаты шифрования гаммированием

5 Выводы

В ходе выполнения данной лабораторной работы было выполнено ознакомление с шифрованием гаммированием на шифрования гаммированием с конечной гаммой.
В результате проделанной работы был программно реализован этот метод шифрования.
В итоге поставленные цели и задачи были успешно достигнуты.

Список литературы

1. Исключающее «или» [Электронный ресурс]. Википедия, 2021. URL: https://ru.wikipedia.org/wiki/Исключающее_«или».
2. Шифр табличной маршрутной перестановки [Электронный ресурс]. Википедия, 2020. URL: https://ru.wikipedia.org/wiki/Гаммирование.