

Лабораторная работа №3

Шифрование гаммированием

Баулин Егор Александрович, НФИМд-02-22

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
4.1	Описание реализации метода шифрования	9
4.2	Листинг	10
4.3	Полученные результаты	12
5	Выводы	13
	Список литературы	14

Список иллюстраций

4.1	Схема однократного использования	9
4.2	Результаты шифрования гаммированием	12

Список таблиц

1 Цель работы

Знакомство 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.

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

$$k = k_1 k_2 \dots k_i \dots k_m,$$

а p - последовательность, которая является ключом:

$$p = p_1 p_2 \dots p_i \dots p_m,$$

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:

$$c = c_1 c_2 \dots c_i \dots c_m$$

Более подробно о шифровании гаммированием: [2]

4 Выполнение лабораторной работы

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:

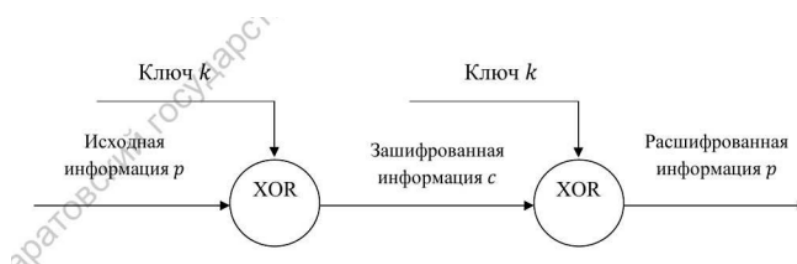


Рис. 4.1: Схема однократного использования

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 → KEYKEYKE

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

4.2 Листинг

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

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

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_shifr(gamma, len(message))
print("\nПреобразование {} -> {}".format(gamma.upper(), new.upper()))
message_indexes = get_indexes(message)
print("\nВаше сообщение:\n{} ({})" .format(message.upper(), message_indexes))
gamma_indexes = get_indexes(new)
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:  шифр
Введите ключ (гамма):  шифр

Преобразование ШИФР -> ШИФР

Ваше сообщение:
ШИФР ([26, 10, 22, 18])

Ваша гамма:
ШИФР ([26, 10, 22, 18])

Зашифрованное сообщение:
СТЙВ ([19, 20, 11, 3])
```

Рис. 4.2: Результаты шифрования гаммированием

5 Выводы

Таким образом в процессе лабораторной работы изучено и реализовано шифрование гаммирования конечной гаммой.

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>.