

# **Отчёт по лабораторной работе №7**

## **Информационная безопасность**

**Элементы криптографии. Однократное гаммирование**

Выполнила: Малащенко Марина Владимировна,  
НФИбд-01-20, 1032202459

# Содержание

<b>Цель работы</b>	<b>4</b>
<b>Теоретическое введение</b>	<b>5</b>
<b>Выполнение лабораторной работы</b>	<b>8</b>
<b>Вывод</b>	<b>10</b>
<b>Список литературы. Библиография</b>	<b>11</b>

# Список иллюстраций

1	(рис. 1. Программный код приложения, реализующего режим однократного гаммирования) . . . . .	9
---	--	---

## **Цель работы**

Освоить на практике применение режима однократного гаммирования.

# Теоретическое введение

Предложенная Г. С. Вернамом так называемая «схема однократного использования (гаммирования)» является простой, но надёжной схемой шифрования данных. [0]

**Гаммирование** представляет собой наложение (снятие) на открытые (зашифрованные) данные последовательности элементов других данных, полученной с помощью некоторого криптографического алгоритма, для получения зашифрованных (открытых) данных. Иными словами, наложение гаммы — это сложение её элементов с элементами открытого (закрытого) текста по некоторому фиксированному модулю, значение которого представляет собой известную часть алгоритма шифрования.

В соответствии с теорией криптоанализа, если в методе шифрования используется однократная вероятностная гамма (однократное гаммирование) той же длины, что и подлежащий сокрытию текст, то текст нельзя раскрыть. Даже при раскрытии части последовательности гаммы нельзя получить информацию о всём скрываемом тексте.

Наложение гаммы по сути представляет собой выполнение операции сложения по модулю 2 (XOR) (обозначаемая знаком  $\oplus$ ) между элементами гаммы и элементами подлежащего сокрытию текста. Напомним, как работает операция XOR над битами:  $0 \oplus 0 = 0$ ,  $0 \oplus 1 = 1$ ,  $1 \oplus 0 = 1$ ,  $1 \oplus 1 = 0$ .

Такой метод шифрования является симметричным, так как двойное прибавление одной и той же величины по модулю 2 восстанавливает исходное значение, а шифрование и расшифрование выполняется одной и той же программой.

Если известны ключ и открытый текст, то задача нахождения шифротекста заключается в применении к каждому символу открытого текста следующего правила:

$$C_i = P_i \oplus K_i, (7.1)$$

где  $C_i$  —  $i$ -й символ получившегося зашифрованного послания,  $P_i$  —  $i$ -й символ открытого текста,  $K_i$  —  $i$ -й символ ключа,  $i = 1, m$ . Размерности открытого текста и ключа должны совпадать, и полученный шифротекст будет такой же длины.

Если известны шифротекст и открытый текст, то задача нахождения ключа решается также в соответствии с (7.1), а именно, обе части равенства необходимо сложить по модулю 2 с  $P_i$ :

$$C_i \oplus P_i = P_i \oplus K_i \oplus P_i = K_i,$$

$$K_i = C_i \oplus P_i.$$

Открытый текст имеет символьный вид, а ключ — шестнадцатеричное представление. Ключ также можно представить в символьном виде, воспользовавшись таблицей ASCII-кодов.

К. Шеннон доказал абсолютную стойкость шифра в случае, когда однократно используемый ключ, длиной, равной длине исходного сообщения, является фрагментом истинно случайной двоичной последовательности с равномерным законом распределения. Криптоалгоритм не даёт никакой информации об открытом тексте: при известном зашифрованном сообщении  $C$  все различные ключевые последовательности  $K$  возможны и равновероятны, а значит, возможны и любые сообщения  $P$ .

Необходимые и достаточные условия абсолютной стойкости шифра:

- полная случайность ключа;
- равенство длин ключа и открытого текста;
- однократное использование ключа.

Рассмотрим пример.

Ключ Центра:

05 0C 17 7F 0E 4E 37 D2 94 10 09 2E 22 57 FF C8 0B B2 70 54

Сообщение Центра:

*Штирлиц – Вы Герой!!*

D8 F2 E8 F0 EB E8 F6 20 2D 20 C2 FB 20 C3 E5 F0 EE E9 21 21

Зашифрованный текст, находящийся у Мюллера:

DD FE FF 8F E5 A6 C1 F2 B9 30 CB D5 02 94 1A 38 E5 5B 51 75

Дешифровальщики попробовали ключ:

05 0C 17 7F 0E 4E 37 D2 94 10 09 2E 22 55 F4 D3 07 BB BC 54

и получили текст:

D8 F2 E8 F0 EB E8 F6 20 2D 20 C2 FB 20 C1 EE EB E2 E0 ED 21

*Штирлиц - Вы Болван!*

Другие ключи дадут лишь новые фразы, пословицы, стихотворные строфы, словом, всевозможные тексты заданной длины.

# Выполнение лабораторной работы

Нужно подобрать ключ, чтобы получить сообщение «С Новым Годом, друзья!». Требуется разработать приложение, позволяющее шифровать и дешифровать данные в режиме однократного гаммирования. Приложение должно:

1. Определить вид шифротекста при известном ключе и известном открытом тексте.
2. Определить ключ, с помощью которого шифротекст может быть преобразован в некоторый фрагмент текста, представляющий собой один из возможных вариантов прочтения открытого текста.

Для решения задачи написан программный код:



```

In [1]: import random

In [2]: from random import seed

In [3]: import string

In [4]: # сложение двух строк по модулю
def xor_text_f(text, key):
    if len(key) != len(text): return "Ошибка: Ключ и текст разной длины"
    xor_text = ''
    for i in range(len(key)):
        xor_text_symbol = ord(text[i]) ^ ord(key[i])
        xor_text += chr(xor_text_symbol)
    return xor_text

In [5]: # ввод исходного текста
text = "С Новым Годом, друзья!"

In [6]: # создание ключа
key = ''
seed(22)
for i in range(len(text)):
    key += random.choice(string.ascii_letters + string.digits)
key

Out[6]: '96ipbNC1ShVP4wY4for9du'

In [7]: # получение шифротекста
xor_text = xor_text_f(text, key)
xor_text

Out[7]: 'И\х16VюèSŵLpібЇ[yЁЦьхvыТ'

In [8]: # открытый текст
xor_text_f(xor_text, key)

Out[8]: 'С Новым Годом, друзья!'

In [9]: # получение ключа
xor_text_f(text, xor_text)

Out[9]: '96ipbNC1ShVP4wY4for9du'

```

Рис. 1: (рис. 1. Программный код приложения, реализующего режим однократного гаммирования)

## **Вывод**

В ходе выполнения данной лабораторной работы было освоено на практике применение режима однократного гаммирования.

# **Список литературы. Библиография**

[0] Методические материалы курса