

# Отчёт по лабораторной работе №7

## Информационная безопасность

### Элементы криптографии. Однократное гаммирование

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

#### Содержание

1	Цель работы .....	1
2	Теоретическое введение.....	1
3	Выполнение лабораторной работы.....	2
4	Вывод.....	4
5	Список литературы. Библиография .....	4

#### 1 Цель работы

Освоить на практике применение режима однократного гаммирования.

#### 2 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.

### 3 Выполнение лабораторной работы

Два текста кодируются одним ключом (однократное гаммирование). Требуется не зная ключа и не стремясь его определить, прочитать оба текста. Необходимо разработать приложение, позволяющее шифровать и дешифровать тексты  $P_1$  и  $P_2$  в режиме однократного гаммирования. Приложение должно определить вид шифротекстов  $C_1$  и  $C_2$  обоих текстов  $P_1$  и  $P_2$  при известном ключе ; Необходимо определить и выразить аналитически способ, при котором злоумышленник может прочитать оба текста, не зная ключа и не стремясь его определить. Для решения задачи написан программный код:

```
# получение шифротекста два
xor_text2 = xor_text_f(text2,key)
xor_text2

'ЧVАжжVJСшЖŸЫйїЖЄJёĥ'
```

```
[ ] # открытый текст 1
xor_text_f(key,xor_text1)

'НаВашисходящийот1204'
```

```
[ ] # открытый текст 2
xor_text_f(key,xor_text2)

'ВСеверныйфилиалБанка'
```

```
[ ] # c1+c2
c1_c2 = xor_text_f(xor_text1,xor_text2)
c1_c2

'\x0f\x11'\x02}х|\x0e\x07pwr\x00\t\x05SЁЦЬЕ'
```

```
[ ] # p1+p2
p1_p2 = xor_text_f(text1,text2)
p1_p2

'\x0f\x11'\x02}х|\x0e\x07pwr\x00\t\x05SЁЦЬЕ'
```

```
[ ] # p2
px = xor_text_f(c1_c2,text1)
px

'ВСеверныйфилиалБанка'
```

```
# получение ключа
xor_text_f(text,xor_text)

'96ipbNC1ShVP4wY4for9du'
```

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

```
[ ] import random

[ ] from random import seed

[ ] import string

[ ] # 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

[ ] # ввод исходного текста
text1 = "НаВашиходящийот1204"
text2 = "ВСеверныйфилиалБанка"

[ ] # ввод ключа
key = ''
seed(20)
for i in range(len(text1)):
    key += random.choice(string.ascii_letters + string.digits)
key

↔ '5URYX45jqR025g3uK5kb'

[ ] # получение шифротекста один
xor_text1 = xor_text_f(text1, key)
xor_text1

↔ 'ШєrмAKVЯяAЁoЙўЙзz\х07[V'
```

(рис. 2. Программный код приложения, реализующего режим однократного гаммирования)

## 4 Вывод

Таким образом, без знания ключа `K` мы можем получить результат `T1 XOR T2`. Если один из открытых текстов (например, `T1`) известен частично или полностью, то можно получить соответствующую часть или весь второй текст (`T2`).

## 5 Список литературы. Библиография

[0] Методические материалы курса