# Лабораторная работа №7

Лукьянова Ирина Владимировна, НФИбд-02-19

# Содержание

# List of Figures

# List of Tables

# 1 Цель работы

Освоить на практике применение режима однократного гаммирования.[1]

[1] Элементы криптографии. Однократное гаммирование.

# 2 Теоретические сведения

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 преоб- разован в некоторый фрагмент текста, представляющий собой один из возможных вариантов прочтения открытого текста.

# 3 Выполнение лабораторной работы

Добавляем необходимые библиотеки и создаем функцию генерации ключа.(рис. 3.1)

```
[15]: import random
      import string
      import numpy as np

[16]: sms = 'С Новым Годом, друзья!'

[17]: def f1(n): #генерация ключа
          k = []
          for i in range(n):
              k.append(random.choice(string.ascii_letters + string.digits))
          return k

[18]: key = f1(len(sms))
      np.array(key)

[18]: array(['8', 'c', 'T', 'A', 'w', 'C', 'j', 'l', '8', 'z', 'a', 'H', 'E',
             'J', 'V', '2', '0', 'V', 'r', 'e', 'm', 'A'], dtype='<U1')
```

Figure 3.1: Код

Создаем функцию перевода в 16 строку и переводим ключ, с помощью этой функции.(рис. 3.2)

```
[22]: def f2(t): #перевод в 16 строку
          a = []
          for i in t:
              a.append(hex(ord(i))[2:])
          return a

[23]: key16 = f2(key)
      np.array(key16)

[23]: array(['38', '63', '54', '41', '77', '43', '6a', '6c', '38', '7a', '61',
             '48', '45', '4a', '56', '32', '30', '56', '72', '65', '6d', '41'],
            dtype='<U2')
```

Figure 3.2: Код 2

Создаем функцию сложения по модулю 2 (XOR) для нее нам нужна еще одна функция, которая переводит наши символы в числа. (рис. 3.3)

```
[25]: def f3(t,k): #XOR
          a = []
          for (i,j) in zip(t,k):
              a.append(chr(i^j))
          return a

[27]: def for16(a): #превращаем символы в числа
          f = []
          for i in a:
              f.append(ord(i))
          return f

[28]: cipher = f3(for16(sms), for16(key))
      np.array(cipher)

[28]: array(['Й', 'C', 'щ', 'Ѿ', 'x', 'J', 'i', 'L', 'Ы', 'ф', 's', 'Ѵ', 'оу',
             'f', 'v', 'I', 'Ѱ', 'E', 'x', 'Щ', 'T', '`'], dtype='<U1')

[29]: cipher16 = f2(cipher)
      np.array(cipher16)

[29]: array(['419', '43', '449', '47f', '445', '408', '456', '4c', '42b', '444',
             '455', '476', '479', '66', '76', '406', '470', '415', '445', '429',
             '422', '60'], dtype='<U3')

[32]: decoding = f3(for16(cipher), for16(key))
      np.array(decoding)

[32]: array(['С', ' ', 'Н', 'о', 'в', 'ы', 'м', ' ', 'Г', 'о', 'д', 'о', 'м',
             ',', ' ', 'д', 'р', 'у', 'з', 'ь', 'я', '!'], dtype='<U1')
```

Figure 3.3: Функции

В итоге получаем следующие данные: (рис. 3.4)

```
print("Открытый текст: ", sms)
print("Ключ: ", ''.join(key))
print("Ключ в 16: ", ' '.join(key16))
print("Шифр: ", ''.join(cipher))
print("Шифр в 16: ", ' '.join(cipher16))
print("Расшифрованный текст: ", ''.join(decoding))
```

```
Открытый текст:  С Новым Годом, друзья!
Ключ:  8cTAwCjl8zaHEJV20VremA
Ключ в 16:  38 63 54 41 77 43 6a 6c 38 7a 61 48 45 4a 56 32 30 56 72 65 6d 41
Шифр:  ЙСщѾxJiLЫфsѴoyfvIѰExЩT`
Шифр в 16:  419 43 449 47f 445 408 456 4c 42b 444 455 476 479 66 76 406 470 415 445 429 422 60
Расшифрованный текст:  С Новым Годом, друзья!
```

Figure 3.4: Вывод данных

После мы создаем новый ключ, используя открытый текст и шифр и расшифровываем сообщение с новым ключом: (рис. 3.5)

```
new_key = f3(for16(sms), for16(cipher))
np.array(new_key)
```

```
array(['8', 'c', 'T', 'A', 'w', 'C', 'j', 'l', '8', 'z', 'a', 'H', 'E',
       'J', 'V', '2', '0', 'V', 'r', 'e', 'm', 'A'], dtype='<U1')
```

```
new_decoding = f3(for16(new_key), for16(cipher))
np.array(new_decoding)
```

```
array(['С', ' ', 'Н', 'о', 'в', 'ы', 'м', ' ', 'Г', 'о', 'д', 'о', 'м',
       ',', ' ', 'д', 'р', 'у', 'з', 'ь', 'я', '!'], dtype='<U1')
```

```
print("Открытый текст: ", sms)
print("Новый ключ: ", ''.join(new_key))
print("Шифр: ", ''.join(cipher))
print("Расшифрованный текст: ", ''.join(new_decoding))
```

```
Открытый текст:  С Новым Годом, друзья!
Новый ключ:  8cTAwCjl8zaHEJV20VremA
Шифр:  ЙСщЩxJiLЫфsЎoyfvIЎExЩT`
Расшифрованный текст:  С Новым Годом, друзья!
```

```
if key == new_key:
    print("Одинаковый ключ")
else:
    print("Другой ключ")
```

```
Одинаковый ключ
```

Figure 3.5: Проверяем работу сервиса

В завершении сравниваем ключи и убеждаемся в правильности найденного нового ключа.

# 4 Контрольные вопросы

1. Поясните смысл однократного гаммирования. Гаммирование — это наложение (снятие) на открытые (зашифрованные) данные последовательности элементов других данных, полученной с помощью некоторого криптографического алгоритма, для получения зашифрованных (открытых) данных.

2. Перечислите недостатки однократного гаммирования. Ключ одного размера с сообщением, что является уязвимостью.

3. Перечислите преимущества однократного гаммирования. Стойкость и легкость в использоавании.

4. Почему длина открытого текста должна совпадать с длиной ключа? Каждый символ текста попарно складывается с символом ключа. Следовательно шифротекст получится той же длины.

5. Какая операция используется в режиме однократного гаммирования,назовите её особенности? Сложение по модулю 2 (XOR). Каждая пара двоичных знаков заменяется одним двоичным знаком шифрованного текста в соответствии с принятым алгоритмом;

6. Как по открытому тексту и ключу получить шифротекст? Сложить по модулю 2 символы открытого текста и ключа.

7. Как по открытому тексту и шифротексту получить ключ? Сложить по модулю 2 символы открытого текста и шифротекста.

8. В чем заключаются необходимые и достаточные условия абсолютной стойкости шифра? полная случайность ключа; равенство длин ключа и открытого текста; однократное использование ключа.

# 5 Выводы

В ходе выполнения данной лабораторной работы я освоила на практике применение режима однократного гаммирования.

# 6 Список литературы

1. Элементы криптографии. Однократное гаммирование. / Кулябов Д. С., Королькова А. В., Геворкян М. Н. - Москва: - 7 с.
2. Руководство по оформлению Markdown.