

# Отчет по лабораторной работе №7

---

Дерябина Мария

2021

RUDN University, Moscow, Russian Federation

Освоить на практике применение режима однократного гаммирования

Гаммирование представляет собой наложение (снятие) на открытые (зашифрованные) данные последовательности элементов других данных, полученной с помощью некоторого криптографического алгоритма, для получения зашифрованных (открытых) данных. Наложение гаммы представляет собой выполнение операции сложения по модулю 2 (XOR) между элементами гаммы и элементами подлежащего сокрытию текста. 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:

$$C_i = P_i \wedge K_i,$$

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, а именно, обе части равенства необходимо сложить по модулю 2 с  $P_i$  :

$$C_i \wedge P_i = P_i \wedge K_i \wedge P_i = K_i,$$

$$K_i = C_i \wedge P_i$$

# Выполнение лабораторной работы

Написала программу на языке Python, позволяющую шифровать и дешифровать данные в режиме однократного гаммирования.

Программа имеет 3 функции:

1. `decode(cr_message, key)`. Данная функция принимает зашифрованное сообщение и ключ (в виде строк с шестнадцатиричными значениями). Для каждого значения зашифрованного сообщения выполняется сложение по модулю 2 с соответствующим значением ключа. Функция возвращает строку с расшифрованным сообщением (рис. 1).

```
1 # Декодировать сообщение
2 def decode(cr_message, key):
3     message = []
4     cr_message = cr_message.split()
5     key = key.split()
6     for i in range(0, len(cr_message)):
7         message.append(chr(int(cr_message[i], 16) ^ int(key[i], 16)))
8     return ''.join(message)
```

Figure 1: Функция для дешифрования сообщения

# Выполнение лабораторной работы

2. `def encode(message, key)`. Данная функция принимает исходное сообщение и ключ. Каждый символ сообщения преобразовывается в число, соответствующее его коду в системе Unicode. Далее выполняется сложение по модулю 2 между получившимися кодами и соответствующими значениями ключа. Функция возвращает зашифрованное сообщение в виде строки с шестнадцатиричными значениями (рис. 2).

```
11 # Закодировать сообщение
12 def encode(message, key):
13     cr_message = []
14     key = key.split()
15     for i in range(0, len(message)):
16         cr_message.append((hex(ord(message[i]) ^ int(key[i], 16)).lstrip('0x')).upper())
17         if len(cr_message[i]) == 1:
18             cr_message[i] = '0' + cr_message[i]
19     return ' '.join(cr_message)
```

**Figure 2:** Функция для шифрования сообщения

3. `get_key(message, cr_message)`. Данная функция принимает исходное сообщение и закодированное сообщение. Выполняется сложение по модулю 2 между кодами символов исходного сообщения и значениями закодированного сообщения. Функция возвращает ключ, с помощью которого исходный текст был закодирован (рис. 3).

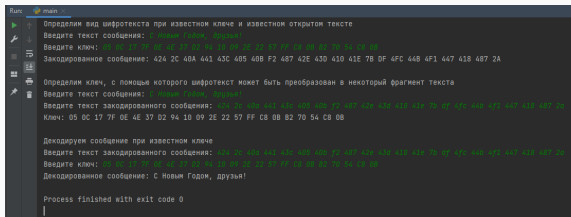
```
22 # Найти ключ
23 def get_key(message, cr_message):
24     cr_message = cr_message.split()
25     key = []
26     for i in range(0, len(message)):
27         key.append((hex(ord(message[i]) ^ int(cr_message[i], 16)).lstrip('0x')).upper())
28         if len(key[i]) == 1:
29             key[i] = '0' + key[i]
30     return ' '.join(key)
```

**Figure 3:** Функция для определения ключа шифрования



Протестировала программу на сообщении ‘С Новым Годом, друзья!’. При вызове различных функций убедилась, что программа корректно выполняет следующие задачи (рис. 4):

1. Определяет вид шифротекста при известном ключе и известном открытом тексте.
2. Определяет ключ, с помощью которого шифротекст может быть преобразован в некоторый фрагмент текста.
3. Выполняет дешифрование текста при известном ключе.



```
Run main
└─> Определим вид шифротекста при известном ключе и известном открытом тексте
Введите текст сообщения: 'С Новым Годом, друзья!'
Введите ключ: 05 0C 17 7F 0E 4E 37 D2 94 10 09 2E 22 57 FF C8 0B B2 70 54 C8 0B
Закодированное сообщение: 424 2C 4BA 441 43C 405 40B F2 487 42E 43D 41E 7B DF 4FC 44B 4F1 447 418 487 2A

Определим ключ, с помощью которого шифротекст может быть преобразован в некоторый фрагмент текста
Введите текст сообщения: 'С Новым Годом, друзья!'
Введите текст закодированного сообщения: 444 3C 44B 441 43C 40B F2 487 42E 43D 41E 7B DF 4FC 44B 4F1 447 418 487 2A
Ключ: 05 0C 17 7F 0E 4E 37 D2 94 10 09 2E 22 57 FF C8 0B B2 70 54 C8 0B

Декодируем сообщение при известном ключе
Введите текст закодированного сообщения: 444 3C 44B 441 43C 40B F2 487 42E 43D 41E 7B DF 4FC 44B 4F1 447 418 487 2A
Введите ключ: 05 0C 17 7F 0E 4E 37 D2 94 10 09 2E 22 57 FF C8 0B B2 70 54 C8 0B
Декодированное сообщение: С Новым Годом, друзья!

Process finished with exit code 0
```

Figure 4: Тестирование программы

Я освоила на практике применение режима однократного гаммирования и разработала программу, выполняющую различные функции: шифрование и дешифрование текста, определение ключа.