Лабораторная работа №3

Шифр гаммирования

Яковлев Артём Александрович

15 октября 2022

Российский университет дружбы народов, Москва, Россия

Цель работы

Цель работы

- Цель данной лабораторной работы изучение реализация алгоритма шифрования гаммированием.

Теоретическое введение

Гамми́рование, 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 [@cypher].

В этом способе шифрование выполняется путем сложения символов исходного текста и ключа по модулю, равному числу букв в алфавите. Если в исходном алфавите, например, 33 символа, то сложение производится по модулю 33. Такой процесс сложения исходного текста и ключа называется в криптографии **наложением гаммы** [@intuit].

Выполнение лабораторной работы

Выполнение лабораторной работы

Для реализации шифров мы будем использовать Python, так как его синтаксис позволяет быстро реализовать необходимые нам алгоритмы.

Реализация маршрутного шифрования

В качестве начальных значений берется гамма “гамма”. Алфавитом может быть любая строка неповторяющихся символов. Я использую кириллицу. Также задаю строку сообщение, которое будет шифроваться.

```
import numpy as np
key = 'гамма'
word = 'приказ'
alphabet = 'абвгдеёжзийклмнопрстуфчцчшщъьэюя'
```

Рис. 1: Код 1

Реализация маршрутного шифрования

Реализация маршрутного шифрования

Задам функцию *Shifr()*, в качестве параметров передаются заданные начальные данные. Внутри функции ключ-гамма, алфавит и сообщение преобразую в массив. Затем увеличу длину ключа-гаммы, чтобы число символов совпадало с сообщением, делаю это дописывая ключ пока длина не будет равной или больше сообщению, лишние символы отсекаю. Затем нахожу индексы символов сообщения и ключа в алфавите и сохраняю их в массиве. В новый массив сохраняю символы, рассчитав индексы по формуле $z = x + k \ (mod\ N)$. Полученный массив преобразую в строку и возвращаю.

Реализация маршрутного шифрования

Реализация маршрутного шифрования

```
def shifr(k, w, alp):
    alp = list(alp)
    k = list(k)
    w = list(w)
    n = len(alp)
    while len(k) < len(w):
        k+= k
    k = k[:len(w)]

    w_i = []
    for i in range(len(w)):
        for j in range(n):
            if w[i] == alp[j]:
                w_i.append(j)
    k_i = []
    for i in range(len(k)):
        for j in range(n):
            if k[i] == alp[j]:
                k_i.append(j)
    w_shifr = []
    for i in range(len(w_i)):
        w_shifr.append(alp[w_i[i]+k_i[i]%n])
    w_shifr = ''.join(w_shifr)
    return w_shifr

word_shifr = shifr(key, word, alphabet)

print(word, '-- Слово')
print(word_shifr, '-- Зашифрованное слово')
```

```
приказ -- Слово
трччак -- Зашифрованное слово
```

Выводы

Выводы

В ходе данной лабораторной работы я реализовал алгоритм шифрования гаммированием.