РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ
Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

Отчёт по лабораторной работе №3
Шифрование гаммированием

Дисциплина: *Математические основы защиты информации и информационной безопасности*

Студент: Леонова Алина Дмитриевна, 1032212306
Группа: НФИмд-01-21
Преподаватель: Кулябов Дмитрий Сергеевич,
д-р.ф.-м.н., проф.

Москва 2021

Содержание

List of Figures

1 Цель работы

Целью данной работы является ознакомление с шифрованием гаммированием и реализация алгоритма на выбранном языке программирования.

2 Задание

Реализовать алгоритм шифрования гаммированием конечной гаммой программно.

3 Теоретическое введение

Гаммирование, наложение гаммы или Шифр XOR ($\oplus$) – 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.

Суммирование обычно выполняется в каком-либо конечном поле. Например, суммирование может принимать вид операции исключающее ИЛИ / XOR / $\oplus$ [1].

3.1 Схема однократного использования

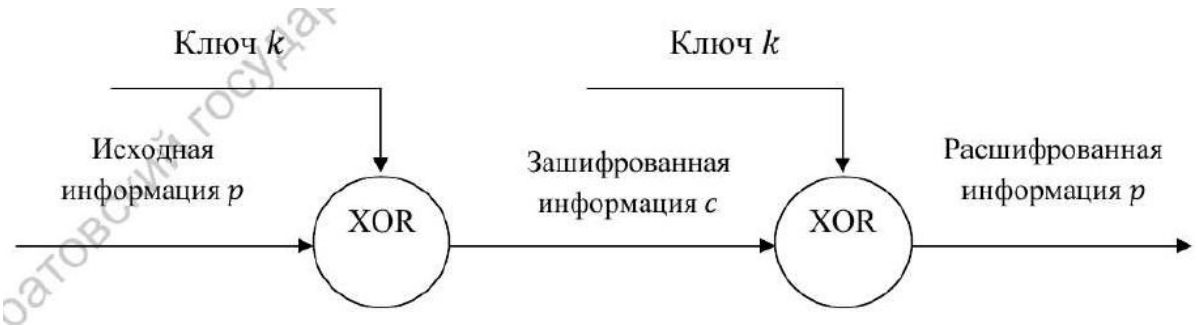

Figure 3.1: Схема однократного использования

Классический одноразовый шифровальный блокнот – большой неповторяющийся случайный набор символов ключа, написанный на листах бумаги, склеенных в блокнот. Шифровальщик при личной встрече снабжался блокнотом, каждая страница которого содержала ключ. Такой же блокнот имелся

и у принимающей стороны. Использованные страницы после однократного использования уничтожались [2] (см. рис. 3.1).

Недостаток метода заключается в равенстве объёма ключевой информации объёму передаваемой информации.

3.2 Гаммы

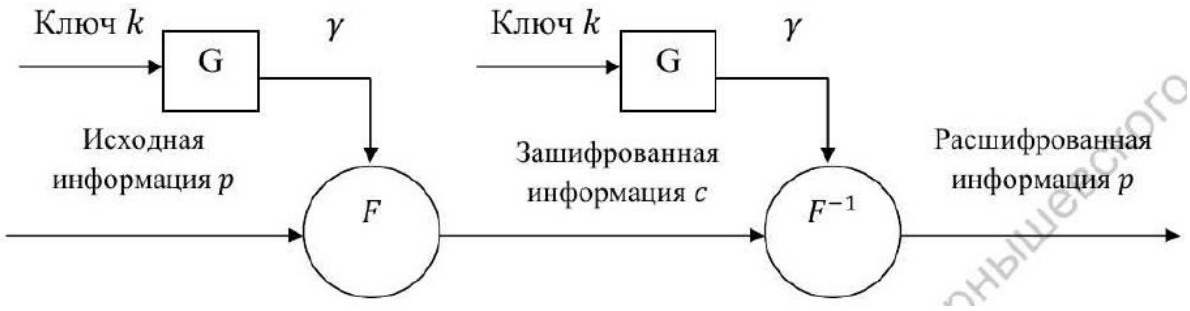

Figure 3.2: Схема гаммирования с использованием генератора псевдослучайных чисел

Стойкость этих шифров определяется качеством гаммы, которое зависит от длины периода (минимального количества символов, после которого последовательность начинает повторяться) и случайности распределения по периоду [3]. Можно использовать случайные (оцифрованные данные случайных процессов) или псевдослучайные гаммы (вычисленные по определённому алгоритму) (см. рис. 3.2).

В лабораторной работе будет реализовываться вариант, когда гамма при необходимости увеличивается путём повторения ключа до тех пор, пока его длина не станет равна длине сообщения.

3.3 Сложение по модулю 2

При равенстве объёма ключевой информации и объёма пеоедаваемого текста символы текста и гаммы представляются в двоичном виде, а затем каждая па-

ра двоичных разрядов складывается по модулю 2, это значит, что процедуры шифрования и дешифрования выполняются по следующим формулам:

$C_i = P_i \oplus K_i$

$P_i = C_i \oplus K_i$

где P_i, C_i – i-ый символ открытого и шифрованного сообщения;

K_i – i-ый символ гаммы (ключа).

3.4 Сложение по модулю N

При замене букв исходного сообщения и ключа на числа в рамках определённого алфавита процедуры шифрования и дешифрования выполняются по следующим формулам:

$C_i = (P_i + K_i)\ mod\ N$

$P_i = (C_i + N - K_i)\ mod\ N$

где P_i, C_i – i-ый символ открытого и шифрованного сообщения;

N – количество символов в алфавите;

K_i – i-ый символ гаммы (ключа).

4 Выполнение лабораторной работы

4.1 1. Шифрование гаммированием

Шифрование гаммированием будет реализовано для конкретных алфавитов (русского и английского). Для сопоставления букв с номерами используется словарь, он начинается с 0 (буква а), а в примере с 1, также Юникод-символ ‘ё’ находится после буквы ‘я’, и размер алфавита равняется не 33, а 32, потому пример из задания не сходится с полученным результатом ровно на одну букву в каждой позиции.

Заданиие алфавитов и словарей для них:

```
# Шифрование гаммированием


# Алфавиты
ru = [chr(i) for i in range(ord('а'), ord('я')+1)]
en = [chr(i) for i in range(ord('a'), ord('z')+1)]
#print('ru: ',ru)
#print('en: ',en)


# Словари букв и номеров
dict_ru = {ru[i]:i for i in range(len(ru))}
dict_en = {en[i]:i for i in range(len(en))}
print('\nru: ',dict_ru)
print('en: ',dict_en)
```

Реализация функции шифрования гаммированием, которой на вход подаются: текст для шифрования, ключ иди гамма и используемый алфавит:

```
def gamma(text, key, abc):
    if abc == ru:
        dict_abc = dict_ru
    else:
        dict_abc = dict_en
    abc_len = len(abc)

    print('\nТекст: ', text)
    text = text.replace(' ','').lower()
    t_len = len(text)

    print('Ключ:', key)
    key = key.lower()
    k_len = len(key)
    gamma = key

    while len(gamma) < t_len:
        gamma += gamma[len(gamma) - k_len]

    print('-----')
    print(text)
    print(gamma)
    print('-----')

    res = ''
    for i in range(t_len):
        x = dict_abc[gamma[i]]   # номера букв ключа
```

```
        y = dict_abc[text[i]]    # номера букв текста

        res += abc[(x + y) % abc_len]
    print('Криптограмма: ', res)
```

Задание входных параметров и вызов функции гаммирования с конечной гаммой:

```
text = 'приказ'
key = 'гамма'
gamma(text, key, ru)

print('-----------------------------------------------------')

text = 'Live long and prosper'
key = 'Spock'
gamma(text, key, en)
```

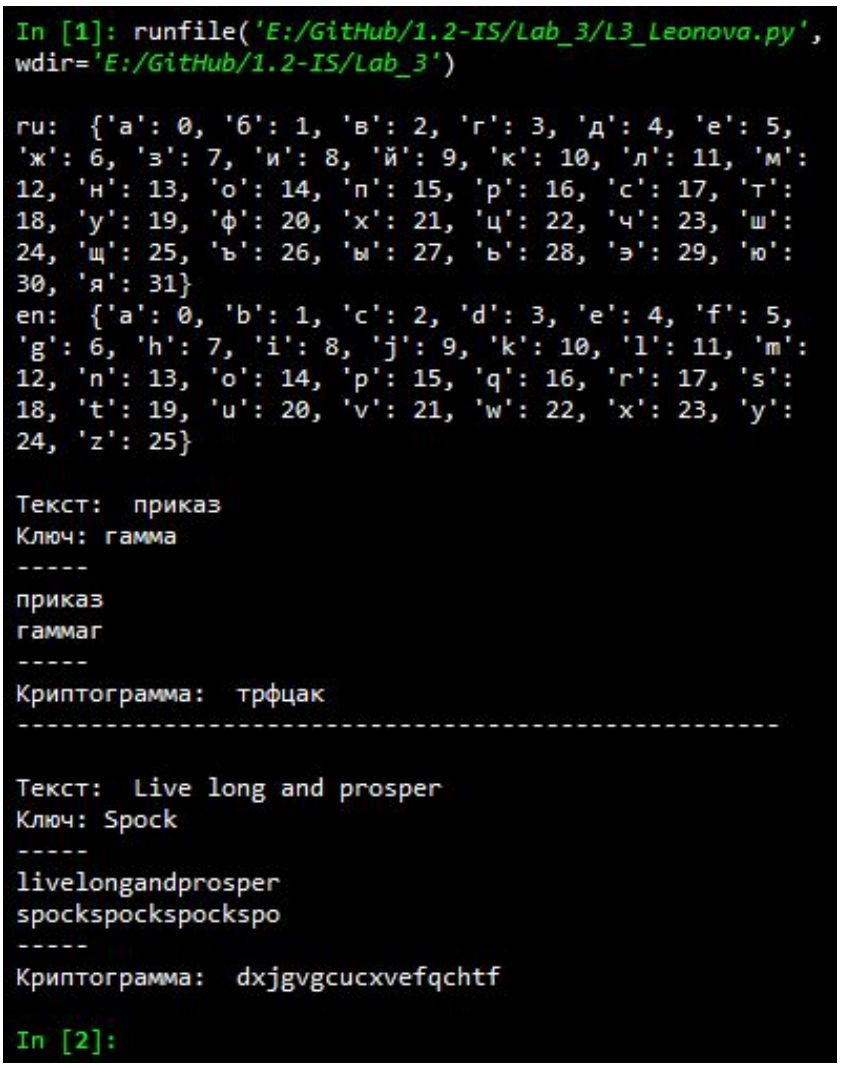

Figure 4.1: Результат выполнения L3_Leonova.py

Результат выполнения программы, реализации шифрования гаммированием с конечной гаммой, проверка на примере из задания и произвольном (см. рис. 4.1).

5 Выводы

Цель лабораторной работы была достигнута, алгоритм шифрования гаммированием с конечной гаммой был реализован на языке программирования Python.

Список литературы

1. Гаммирование [Электронный ресурс]. Википедия, 2020. URL: https://ru.wikipedia.org/wiki/%D0%93%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5.
2. anisimovkhv. Криптографические методы защиты информации. 6. ШИФРЫ ГАММИРОВАНИЯ [Электронный ресурс]. Учебная и научная деятельность Анисимова..., 2021. URL: https://sites.google.com/site/anisimovkhv/learning/kripto/lecture/tema6.
3. Шифры гаммирования [Электронный ресурс]. kriptografia21, 2021. URL: https://sites.google.com/site/kriptografia21/sifry-gammirovania?tmpl=%2Fsystem%2Fapp%2Ftemplates%2Fprint%2F&showPrintDialog=1.