Отчет по лабораторной работе №8

Информационная безопасность

Чекалова Лилия Руслановна

Содержание

1 Цель работы **5**

2 Задание **6**

3 Теоретическое введение **7**

4 Выполнение лабораторной работы **8**

5 Выводы **10**

Список литературы **11**

Список таблиц

Список иллюстраций

4.1 Программа, 1 8
4.2 Программа, 2 9
4.3 Результат запуска программы 9

1 Цель работы

- Освоить на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.

2 Задание

- Написание программы
- Зашифровка текстов по открытым текстам и известному ключу
- Расшифровка текстов без использования ключа

3 Теоретическое введение

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.

Наложение гаммы представляет собой выполнение операции сложения по модулю 2 (XOR) между элементами гаммы и элементами подлежащего сокрытию текста.

Более подробно о см. в [1].

4 Выполнение лабораторной работы

Для выполнения лабораторной работы я написала программу для зашифрования и расшифровки текста. Импортировав необходимые библиотеки, я задала две функции — для генерации ключа по размеру сообщения (выбор случайных букв в кодировке ASCII) и для шифрования (поэлементный XOR) (рис. 4.1).

```
import string
import random
import sys
```

```
def gen_key(size):
    res = ''
    for i in range(size):
        res += random.choice(string.ascii_letters)
    return res


def encrypt(text, key):
    res = ''
    for (t, k) in zip(text, key):
        res += chr(t^k)
    return res
```

Рис. 4.1: Программа, 1

Далее я задала два открытых сообщения одинаковой длины, сгенерировала ключ и закодировала сообщения с помощью этого ключа. После этого я ввела промежуточную переменную temp, в которой сохранила результат поэлементного XOR между двумя зашифрованными сообщениями. Чтобы расшифровать первый текст, я произвела по-

элементный XOR temp и второго открытого сообщения, а для расшифровки второго сообщения — поэлементный XOR temp и первого сообщения (рис. 4.2).

```
message1 = 'С Новым Годом, друзья!'
message2 = 'Счастливого рождества!'
key = gen_key(len(message1))
encr_m1 = encrypt([ord(i) for i in message1], [ord(i) for i in key])
encr_m2 = encrypt([ord(i) for i in message2], [ord(i) for i in key])
temp = encrypt([ord(i) for i in encr_m1], [ord(i) for i in encr_m2])
open1 = encrypt([ord(i) for i in temp], [ord(i) for i in message2])
open2 = encrypt([ord(i) for i in temp], [ord(i) for i in open1])

print('open text1:', message1, '\nopen text2:', message2, '\nkey:', key)
print('encrypted text1:', encr_m1, '\nencrypted text2:', encr_m2)
print('decrypted text1:', open1, '\ndecrypted text2:', open2)
```

Рис. 4.2: Программа, 2

Полученные сообщения я вывела на экран (рис. 4.3). Сообщения были успешно закодированы с помощью заданного ключа и раскодированы без использования этого ключа.

```
open text1: С Новым Годом, друзья!
open text2: Счастливого рождества!
key: FiVePMeBSsTEGunnuETpsA
encrypted text1: ѧIыЋЪIљbpэЩQoYNњeIѣмм`
encrypted text2: ѧЮѦФВЎЇЊрЖeЇыjњрЄЖту`
decrypted text1: С Новым Годом, друзья!
decrypted text2: Счастливого рождества!
```

Рис. 4.3: Результат запуска программы

5 Выводы

В результате лабораторной работы я закрепила знания о базовых элементах криптографии и освоила на примере шифрования двух текстов одним ключом применение режима однократного гаммирования, написав программу, позволяющую зашифровывать тексты и расшифровывать их, даже не зная ключа.

Список литературы

1. Элементы криптографии. Шифрование (кодирование) различных исходных текстов одним ключом [Электронный ресурс]. URL: https://esystem.rudn.ru/pluginfile.php/2090286/mod_resource/content/2/008-lab_crypto-key.pdf.