

Лабораторная работа №7.

Элементы криптографии. Однократное гаммирование.

Яссин Мохамад Аламин

Содержание

1	Цель работы	4
2	Теоретическое введение	5
3	Выполнение лабораторной работы	6
4	Вывод	10
5	Библиография	11

Список иллюстраций

3.1	Вывод программы: закодированное сообщение в виде текста . . .	8
3.2	Вывод программы: раскодированное сообщение в виде текста (неверный ключ)	9
3.3	Вывод программы: раскодированное сообщение в виде текста (вер- ный ключ)	9

1 Цель работы

Освоить на практике применение режима однократного гаммирования.

2 Теоретическое введение

Гаммирование, 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. Суммирование обычно выполняется в каком-либо конечном поле. Например, в поле Галуа $GF(2)$ суммирование принимает вид операции «исключающее ИЛИ (XOR)». [2]

3 Выполнение лабораторной работы

1. Была реализована программа на Python:

```
import numpy as np
from random import randrange

# 1/ задаем строку для шифрования

t = "С Новым Годом, друзья!"

# 2/ переведем строку в hex

hex_message = []
for i in t:
    hex_message.append(i.encode("cp866").hex())
print (hex_message)

# 3/ задаем ключ, такой же длины, что и строка для шифрования

def gen_key(length: int):
    key = []
    for i in range(0, length):
        key.append(hex(randrange(255))[2:])
    return key
```

```

key_1 = gen_key(len(hex_message))
print(key_1)

# 4/ кодируем строку с помощью ключа

def encode_message(hex_message, key):
    return ["%x" % (int(x,16) ^ int(y,16)) for (x, y) in zip(hex_message, key)]

encoded_message = encode_message(hex_message, key_1)
print(encoded_message)

# сообщение текстом

def code_to_lang(encoded_message):
    return bytearray.fromhex("".join(encoded_message)).decode("cp866")

encoded_text = code_to_lang(encoded_message)
print(encoded_text)

# 5/ задаем ключ для расшифровки
key_2 = gen_key(len(hex_message))
print(key_2)

# 6/ декодируем с помощью нового ключа

def decode_message(key, encoded_message):
    return ["%x" % (int(x,16) ^ int(y,16))
            for (x, y) in zip(key, encoded_message)]

```

```

decoded_message = decode_message(key_2, encoded_message)
print(decoded_message)

decoded_text = code_to_lang(decoded_message)
print(decoded_text)

# 7/ декодируем с помощью правильного ключа

decoded_message_r = decode_message(key_1, encoded_message)
decoded_text_r = code_to_lang(decoded_message_r)
print(decoded_text_r)

```

2. Запустили программу. Получили:

- сообщение в hex

```
['91', '20', '8d', 'ae', 'a2', 'eb', 'ac', '20', '83', 'ae', 'a4', 'ae', 'ac', '2c', '20', 'a4', 'e0', 'e3', 'a7', 'ec', 'ef', '21']
```

- ключ для кодировки

```
['a6', 'd6', 'e8', '35', 'f3', '1d', '41', 'e1', '88', 'd1', 'bd', '2a', '16', '80', 'a2', '20', 'ed', '6a', 'fc', '67', 'ce', '9d']
```

- закодированное сообщение

```
['37', 'f6', '65', '9b', '51', 'f6', 'ed', 'c1', 'b', '7f', '19', '84', 'ba', 'ac', '82', '84', 'd', '89', '5b', '8b', '21', 'bc']
```

- закодированное сообщение в виде текста (3.1)

Вывод программы: закодированное сообщение в виде текста

Рис. 3.1: Вывод программы: закодированное сообщение в виде текста

- ключ для расшифровки

['7a', 'f1', '5b', '3e', 'ea', 'd', '9e', '23', 'd6', '3e', '40', 'd9', 'de', '6b', 'd8', '9b', 'b', '4f', '3a', '6e', '14', 'eb']

- сообщение, раскодированное ключом для расшифровки

['4d', '7', '3e', 'a5', 'bb', 'fb', '73', 'e2', 'dd', '41', '59', '5d', '64', 'c7', '5a', '1f', '6', 'c6', '61', 'e5', '35', '57']

- раскодированное сообщение текстом (3.2)

Вывод программы: раскодированное сообщение в виде текста (неверный ключ)

Рис. 3.2: Вывод программы: раскодированное сообщение в виде текста (неверный ключ)

- текст сообщения, раскодированного ключом для кодировки (3.3)

Вывод программы: раскодированное сообщение в виде текста (верный ключ)

Рис. 3.3: Вывод программы: раскодированное сообщение в виде текста (верный ключ)

4 Вывод

В ходе выполнения лабораторной работы было изучено шифрование методом однократного гаммирования и реализована программа на python, шифрующая и расшифровывающая заданную строку этим методом.

5 Библиография

1. Методические материалы курса.
2. Wikipedia: Гаммирование (URL: <https://ru.wikipedia.org/wiki/%D0%93%D0%B0%D0%BC%D0%A8%D0%BD%D0%BB%D0%BA%D0%BF%D0%9A%D0%9B%D0%9C%D0%9D%D0%A7%D0%A6%D0%A5%D0%A4%D0%A3%D0%A2%D0%A1%D0%A0%D0%9F%D0%9E%D0%9D%D0%9C%D0%9B%D0%9A%D0%99%D0%98%D0%97%D0%96%D0%95%D0%94%D0%93%D0%92%D0%91%D0%90%D0%8F%D0%8E%D0%8D%D0%8C%D0%8B%D0%8A%D0%89%D0%88%D0%87%D0%86%D0%85%D0%84%D0%83%D0%82%D0%81%D0%80%D0%7F%D0%7E%D0%7D%D0%7C%D0%7B%D0%7A%D0%79%D0%78%D0%77%D0%76%D0%75%D0%74%D0%73%D0%72%D0%71%D0%70%D0%6F%D0%6E%D0%6D%D0%6C%D0%6B%D0%6A%D0%69%D0%68%D0%67%D0%66%D0%65%D0%64%D0%63%D0%62%D0%61%D0%60%D0%5F%D0%5E%D0%5D%D0%5C%D0%5B%D0%5A%D0%59%D0%58%D0%57%D0%56%D0%55%D0%54%D0%53%D0%52%D0%51%D0%50%D0%4F%D0%4E%D0%4D%D0%4C%D0%4B%D0%4A%D0%49%D0%48%D0%47%D0%46%D0%45%D0%44%D0%43%D0%42%D0%41%D0%40%D0%3F%D0%3E%D0%3D%D0%3C%D0%3B%D0%3A%D0%39%D0%38%D0%37%D0%36%D0%35%D0%34%D0%33%D0%32%D0%31%D0%30%D0%2F%D0%2E%D0%2D%D0%2C%D0%2B%D0%2A%D0%29%D0%28%D0%27%D0%26%D0%25%D0%24%D0%23%D0%22%D0%21%D0%20%D0%1F%D0%1E%D0%1D%D0%1C%D0%1B%D0%1A%D0%19%D0%18%D0%17%D0%16%D0%15%D0%14%D0%13%D0%12%D0%11%D0%10%D0%0F%D0%0E%D0%0D%D0%0C%D0%0B%D0%0A%D0%09%D0%08%D0%07%D0%06%D0%05%D0%04%D0%03%D0%02%D0%01%D0%00>)