Лабораторная работа №8.

Элементы криптографии. Шифрование (кодирование) различных исходных текстов одним ключом.

Ишанова А.И. группа НФИ-02-19

Содержание

# Цель работы

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

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

## Теория к программе

Шифротексты двух телеграмм можно получить по формулам режима однократного гаммирования[1]:

$$

C\_1 = P\_1 + K , C\_2 = P\_2 + K

$$

где - исходное сообщение, - ключ, а оператор + подразумевает прямую сумму.

С учётом свойства операции XOR:

$$

1 + 1 = 0, 1 +0 = 1

$$

где оператор + подразумевает прямую сумму.

получаем:

$$

C\_1 + C\_2 = P\_1 + K + P\_2 + K = P\_1 + P\_2

$$

где оператор + подразумевает прямую сумму.

Из этого следует, что можно найти один текст по двум шифрам, зная другой:

$$

C\_1 + C\_2 + P\_1 = P\_1 + P\_2 + P\_1 = P\_2

$$

где оператор + подразумевает прямую сумму.

# Выполнение лабораторной работы

1. Была реализована программа на Python:

# исходные данные  
P1 = 'НаВашисходящийот1204'  
P2 = 'ВСеверныйфилиалБанка'  
K = ['{:02X}'.format(0x05), '{:02X}'.format(0x0C),   
 '{:02X}'.format(0x17), '{:02X}'.format(0x7F),  
 '{:02X}'.format(0x0E), '{:02X}'.format(0x4E),  
 '{:02X}'.format(0x37), '{:02X}'.format(0xD2),  
 '{:02X}'.format(0x94), '{:02X}'.format(0x10),  
 '{:02X}'.format(0x09), '{:02X}'.format(0x2E),   
 '{:02X}'.format(0x22), '{:02X}'.format(0x57),   
 '{:02X}'.format(0xFF), '{:02X}'.format(0xC8),   
 '{:02X}'.format(0x0B), '{:02X}'.format(0xB2),   
 '{:02X}'.format(0x70), '{:02X}'.format(0x54)]  
print("Тексты:", P1, ", ", P2)  
print("Ключ Центра:", K)  
  
# перевод текста в hex  
def to\_hex(text):  
 return [(i.encode('cp1251')).hex().upper() for i in text]  
  
t1 = to\_hex(P1)  
t2 = to\_hex(P2)  
print("Тексты в hex \n", t1, ", \n", t2)  
  
# кодируем строку с помощью ключа  
  
def encode\_message(hex\_message, key):  
 return (["%02X" % (int(x,16) ^ int(y,16)) for (x, y) in zip(hex\_message, key)])  
  
C1 = encode\_message(t1, K)  
C2 = encode\_message(t2, K)  
  
print("Зашифрованные тексты в hex \n", C1, ", \n", C2)  
  
# переводим шифр в текст  
  
  
def cipher\_text(C):  
 return [(bytes.fromhex(i)).decode('cp1251') for i in C]  
  
T1 = cipher\_text(C1)  
T2 = cipher\_text(C2)  
  
print("Зашифрованные тексты \n", T1, ", \n", T2)  
  
def code\_to\_lang(encoded\_message):  
 return bytearray.fromhex("".join(encoded\_message)).decode("cp1251")  
  
T\_1 = code\_to\_lang(C1)  
T\_2 = code\_to\_lang(C2)  
print("Зашифрованные тексты \n", T\_1, ", \n", T\_2)  
  
# разгадывание второго текста по первому  
def guess\_text(c1, c2, p1):  
 return (["%02X" % (int(x,16) ^ int(y,16) ^ int(z,16)) for (x, y,z)   
 in zip(c1, c2, p1)])  
  
g1 = guess\_text(C1, C2, t1)  
g2 = guess\_text(C2, C1, t2)  
  
print("Поиск второго текста по первому \n", g1, " , \n", code\_to\_lang(g1) )  
print("Поиск первого текста по второму \n", g2, " , \n", code\_to\_lang(g2) )

1. Запустили программу. Получили:

* телеграммы в hex

[‘CD’, ‘E0’, ‘C2’, ‘E0’, ‘F8’, ‘E8’, ‘F1’, ‘F5’, ‘EE’, ‘E4’, ‘FF’, ‘F9’, ‘E8’, ‘E9’, ‘EE’, ‘F2’, ‘31’, ‘32’, ‘30’, ‘34’]

[‘C2’, ‘D1’, ‘E5’, ‘E2’, ‘E5’, ‘F0’, ‘ED’, ‘FB’, ‘E9’, ‘F4’, ‘E8’, ‘EB’, ‘E8’, ‘E0’, ‘EB’, ‘C1’, ‘E0’, ‘ED’, ‘EA’, ‘E0’]

* закодированные телеграммы

[‘C8’, ‘EC’, ‘D5’, ‘9F’, ‘F6’, ‘A6’, ‘C6’, ‘27’, ‘7A’, ‘F4’, ‘F6’, ‘D7’, ‘CA’, ‘BE’, ‘11’, ‘3A’, ‘3A’, ‘80’, ‘40’, ‘60’]

[‘C7’, ‘DD’, ‘F2’, ‘9D’, ‘EB’, ‘BE’, ‘DA’, ‘29’, ‘7D’, ‘E4’, ‘E1’, ‘C5’, ‘CA’, ‘B7’, ‘14’, ‘09’, ‘EB’, ‘5F’, ‘9A’, ‘B4’]

* закодированные телеграммы в виде текста (fig. 1)



Figure 1: Вывод программы: закодированные телеграммы 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’]

* раскодированные телеграмм (fig. 2)

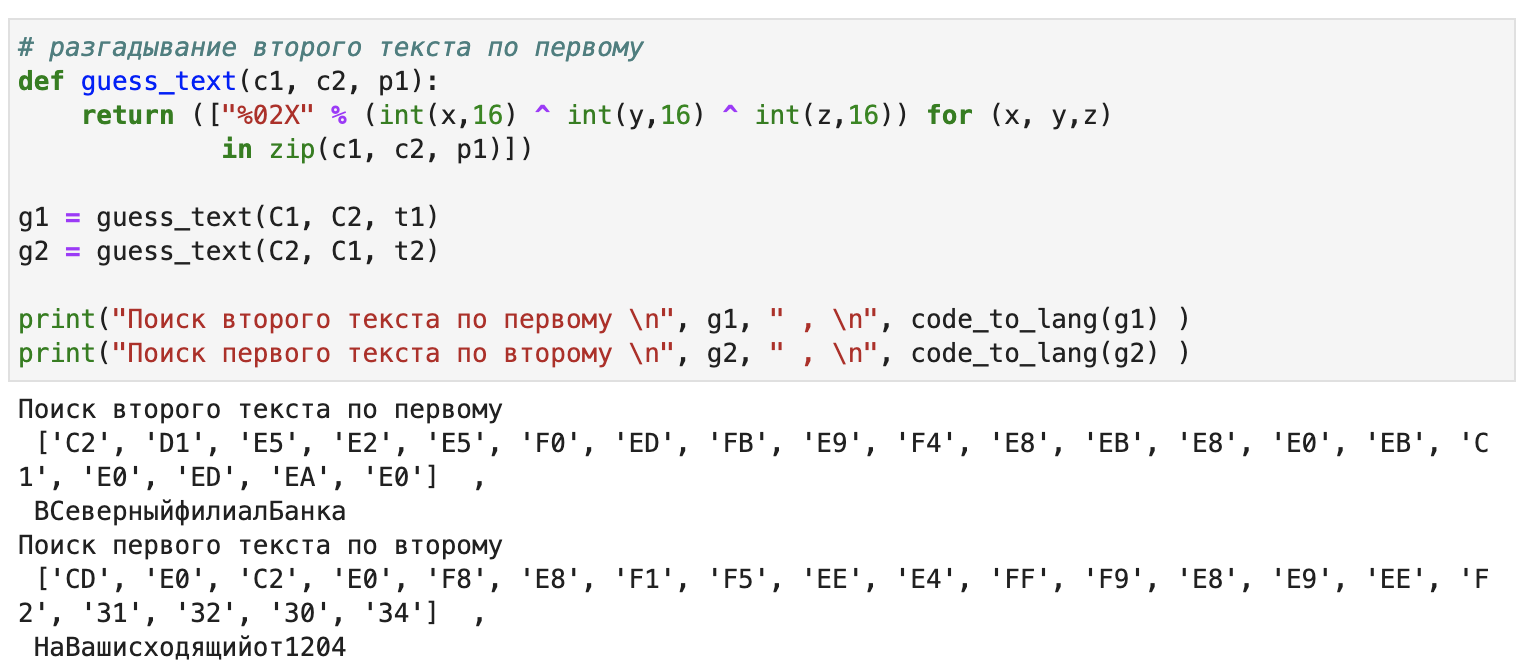


Figure 2: Вывод программы: раскодированные телеграммы в виде текста

# Вывод

В ходе выполнения лабораторной работы было изучено шифрование методом однократного гаммирования на примере кодирования различных исходных текстов одним ключом и реализована программа на python, шифрующая и расшифровавующая два текста одним ключом, и расшифровывающая их без ключа, по одному из текстов.

# Библиография

1. Методические материалы курса.
2. Wikipedia: Гаммирование (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)