

Лабораторная работа №8.

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

Ишанова А.И. группа НФИ-02-19

Содержание

1	Цель работы	4
2	Теоретическое введение	5
2.1	Теория к программе	5
3	Выполнение лабораторной работы	7
4	Вывод	12
5	Библиография	13

List of Figures

3.1	Вывод программы: закодированные телеграммы в виде текста . .	10
3.2	Вывод программы: раскодированные телеграммы в виде текста .	11

1 Цель работы

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

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]

2.1 Теория к программе

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

\$\$

$$C_1 = P_1 \oplus K, C_2 = P_2 \oplus K$$

\$\$

где P - исходное сообщение, K - ключ.

С учётом свойства операции XOR:

\$\$

$$1 \oplus 1 = 0, 1 \oplus 0 = 1$$

\$\$

получаем:

\$\$

$$C_1 \otimes C_2 = P_1 \otimes K \otimes P_2 \otimes K = P_1 \otimes P_2$$

\$\$

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

\$\$

$$C_1 \otimes C_2 \otimes P_1 = P_1 \otimes P_2 \otimes P_1 = P_2$$

\$\$

3 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) )

```

2. 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. 3.1)

```
# 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)

Зашифрованные тексты
['И', 'М', 'Х', 'ц', 'ц', '!', 'Ж', '"', 'z', 'ф', 'ц', 'Ч', 'К', 's', '\x11', ':', ':', 'ђ',
'@', '`'],
['З', 'Э', 'т', 'к', 'л', 's', 'б', ')', '}', 'д', 'б', 'Е', 'К', '.', '\x14', '\t', 'л', '_',
'ль', 'г']

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)

Зашифрованные тексты
ИмХццЖ'zфцЧКs::ђ@` ,
ЗЭтклсб)}}дбЕК·      л_льг
```

Figure 3.1: 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. 3.2)

```

# 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) )

```

Поиск второго текста по первому
['C2', 'D1', 'E5', 'E2', 'E5', 'F0', 'ED', 'FB', 'E9', 'F4', 'E8', 'EB', 'E8', 'E0', 'EB', 'C1', 'E0', 'ED', 'EA', 'E0'] ,
ВСеверныйфилиалБанка

Поиск первого текста по второму
['CD', 'E0', 'C2', 'E0', 'F8', 'E8', 'F1', 'F5', 'EE', 'E4', 'FF', 'F9', 'E8', 'E9', 'EE', 'F2', '31', '32', '30', '34'] ,
НаВашисходящийот1204

Figure 3.2: Вывод программы: раскодированные телеграммы в виде текста

4 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 рас-шифровывающая два текста одним ключом, и расшифровывающая их без ключа, по одному из текстов.

5 Библиография

- [illegible]