Отчет по лабораторной работе №8

Лабораторная работа №8: Элементы криптографии. Шифрование (кодирование) различных исходных текстов одним ключом.

Голова Варвара Алексеевна, НФИбд-03-18

2021, 18 December

Содержание

List of Figures

1 Цель работы

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

2 Выполнение работы

2.1 Алфавит

Задала алфавит из русских букв и алфавит из соответствующих им шестнадцетиричных чисел.

```
"к", "л", "м", "н", "о", "п", "р", "с", "т", "у", "ф", "х", "ц", "ч", "ш", "щ", "ъ", "ы", "ь", "э", "ю", "я", " ", "!", ","]
```

```
alph_16=[]
q=hex(int('C0', 16))
for i in range(64):
    alph_16.append(q)
    q=hex(int(q, 16)+int('1',16))
alph_16.append(hex(int('20',16)))
alph_16.append(hex(int('21',16)))
alph_16.append(hex(int('22',16)))
```

Figure 2.1: Алфавиты

2.2 Сообщения

Ввела сообщения.

```
line_1 = 'С Новым Годом, друзья!'
len(line_1)
```

```
22
```

```
line_2 = 'С Новым Мячом, друзья!'
len(line_2)
```

```
22
```

```
list_1=list(line_1)
```

```
list_2=list(line_2)
```

Figure 2.2: Сообщения

2.3 Ключ

Создала рандомный ключ.

```python
from random import randint

key=[]
for i in range(len(line_1)):
    x=randint (0,255)
    x=hex(x)
    key.append(x)
    print(x.replace("0x",""))
```

```
c0
6f
df
89
58
36
16
cd
af
bb
5b
54
94
13
16
17
4a
ff
58
5e
42
b
```

Figure 2.3: Создание ключа

2.4 Перевод сообщений

Перевела заданные сообщение в шестнадцетиричные числа.

```
list_16_1=[]
def into_list_16(list_1, alphabet, alphabet_16, list_16):
    for i in range(len(list_1)):
        for j in range(len(alphabet)):
            if list_1[i]==alphabet[j]:
                for k in range (len(alphabet_16)):
                    if j==k:
                        list_16.append(alphabet_16[k])
                        print(alphabet_16[k].replace("0x",""))
into_list_16(list_1, alph, alph_16, list_16_1)
```

```
d1
20
cd
ee
e2
fb
ec
20
c3
ee
e4
ee
ec
22
20
e4
f0
f3
e7
fc
ff
21
```

Figure 2.4: Шестнадцетиричная система

```
list_16_2=[]
def into_list_16(list_1, alphabet, alphabet_16, list_16):
    for i in range(len(list_1)):
        for j in range(len(alphabet)):
            if list_1[i]==alphabet[j]:
                for k in range (len(alphabet_16)):
                    if j==k:
                        list_16.append(alphabet_16[k])
                        print(alphabet_16[k].replace("0x",""))
into_list_16(list_2, alph, alph_16, list_16_2)
```

```
d1
20
cd
ee
e2
fb
ec
20
cc
ff
f7
ee
ec
22
20
e4
f0
f3
e7
fc
ff
21
```

Figure 2.5: Шестнадцетиричная система

2.5 Шифрование

Зашифровала два сообщения с помощью одного и того же ключа.

```python
cipher_1=[]
def into_cipher(list_16, key, cipher):
    for i in range(len(list_16)):
        for j in range(len(key)):
            if i==j:
                x=hex(int(list_16[i],16) ^ int(key[j],16))
                cipher.append(x)
                print(x.replace("0x",""))
into_cipher(list_16_1, key, cipher_1)
```

```
11
4f
12
67
ba
cd
fa
ed
6c
55
bf
ba
78
31
36
f3
ba
c
bf
a2
bd
2a
```

Figure 2.6: Шифрование

```python
cipher_2=[]
def into_cipher(list_16, key, cipher):
    for i in range(len(list_16)):
        for j in range(len(key)):
            if i==j:
                x=hex(int(list_16[i],16) ^ int(key[j],16))
                cipher.append(x)
                print(x.replace("0x",""))
into_cipher(list_16_2, key, cipher_2)
```

```
11
4f
12
67
ba
cd
fa
ed
63
44
ac
ba
78
31
36
f3
ba
c
bf
a2
bd
2a
```

Figure 2.7: Шифрование

2.6 Способ, прочтения одного из открытых текстов

Способ, при котором злоумышленник может прочитать оба текста, не зная ключа, но зная один из открытых текстов и два зашифровки текстов.

```
P1=[]
def get_P(P1, P2, C1, C2):
    for i in range(len(C1)):
        for j in range(len(C2)):
            if i==j:
                for k in range(len(P2)):
                    if j==k:
                        x=hex(int(C1[i],16) ^ int(C2[j],16))
                        x.replace("0x","")
                        x=hex(int(P2[k],16) ^ int(x,16))
                        P1.append(x)
                        print(x.replace("0x",""))
get_P(P1, list_16_2, cipher_1, cipher_2)
```

```
d1
20
cd
ee
e2
fb
ec
20
c3
ee
e4
ee
ec
22
20
e4
f0
f3
e7
fc
ff
21
```

Figure 2.8: Прочтение

2.7 Проверка

Проверка

```
if list_16_1==P1:
    print('Yes')
```

```
Yes
```

Figure 2.9: Проверка

3 Выводы

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