Лабораторная работа №8

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

Ли Тимофей Александрович

Содержание

Список таблиц

Список иллюстраций

Цель работы

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

Выполнение лабораторной работы

Импортировал нужные библиотеки и задал два текста одной длины. (рис. @fig:001):

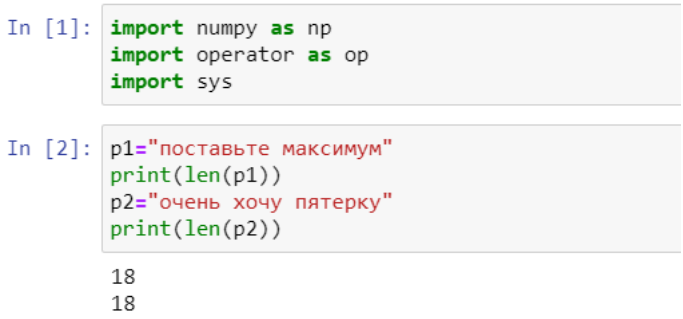

Рис. 0.1: вводные

Написал функцию, определяющую вид шифротекстов C1 и C2 обеих строк при известном ключе. Функция получает на вход две символьные строки, которые затем переводятся в 16-ую систему. Далее генерируется рандомный ключ, при помощи которого определяются соответствующие шифротексты в 16-й системе. Затем шифротекст переводится в строковый формат. Функция возвращает ключ, оба шифротекста в 16-ой системе и строковом формате. (рис. @fig:002)

```
In [3]: def encrypt(text1,text2):
            print("text1: ", text1)
            newtext1=[]
            for i in text1:
                newtext1.append(i.encode("cp1251").hex())
            print("text1 in 16: ", newtext1)
            print("text2: ", text2)
            newtext2=[]
            for i in text2:
                newtext2.append(i.encode("cp1251").hex())
            print("text2 in 16: ", newtext2)
            r=np.random.randint(0,255, len(text1))
            key=[hex(i)[2:] for i in r]
            newkey=[]
            for i in key:
                newkey.append(i.encode("cp1251").hex().upper())
            print("key in 16: ", key)
            xortext1=[]
            for i in range(len(newtext1)):
                xortext1.append("{:02x}".format(int(key[i], 16) ^ int(newtext1[i],16)))
            print("cypher text1 in 16: ", xortext1)
            en_text1=bytearray.fromhex("".join(xortext1)).decode("cp1251")
            print("cypher text1: ", en_text1)
            xortext2=[]
            for i in range(len(newtext2)):
                xortext2.append("{:02x}".format(int(key[i], 16) ^ int(newtext2[i],16)))
            print("cypher text2 in 16: ", xortext2)
            en_text2=bytearray.fromhex("".join(xortext2)).decode("cp1251")
            print("cypher text2: ", en_text2)
            return key, xortext1, en_text1, xortext2, en_text2
```

Рис. 0.2: функция шифрования

Вывод функции: (рис. @fig:003)

```
In [4]: k, t1, et1, t2, et2=encrypt(p1,p2)

text1:  поставьте максимум
text1 in 16:  ['ef', 'ee', 'f1', 'f2', 'e0', 'e2', 'fc', 'f2', 'e5', '20', 'ec', 'e0', 'ea', 'f1', 'e8', 'ec', 'f3', 'ec']
text2:  очень хочу пятерку
text2 in 16:  ['ee', 'f7', 'e5', 'ed', 'fc', '20', 'f5', 'ee', 'f7', 'f3', '20', 'ef', 'ff', 'f2', 'e5', 'f0', 'ea', 'f3']
key in 16:  ['7a', '89', '8a', 'f3', 'ce', 'f8', 'd9', '50', 'a6', '7f', '60', '7d', '64', '4', 'f2', 'ec', '10', '9b']
cypher text1 in 16:  ['95', '67', '7b', '01', '2e', '1a', '25', 'a2', '43', '5f', '8c', '9d', '8e', 'f5', '1a', '00', 'e3', '7
7']
cypher text1:  •g{□.□%ÿC_ћќЋx□rw
cypher text2 in 16:  ['94', '7e', '6f', '1e', '32', 'd8', '2c', 'be', '51', '8c', '40', '92', '9b', 'f6', '17', '1c', 'fa', '6
8']
cypher text2:  ”~o□2Ш,sQЋ@’›ц□□ьh
```

Рис. 0.3: результат работы функции1

Написал функцию, которая при известных двух шифротекстах и одном открытом тексте находит вид второго открытого текста без ключа. Функция получает на вход два шифротекста и один открытый в строковом формате, затем переводит их в 16-ю систему. Затем применяя принцип однократного гаммирования, находит вид второго открытого сообщения без использования ключа шифрования. Возвращает

функция второе расшифрованное сообщение в строковом формате и 16-ой системе. (рис. @fig:004)

```
In [5]: def decrypt(c1, c2, p1):
            print("cypher text1: ", c1)
            newc1=[]
            for i in c1:
                newc1.append(i.encode("cp1251").hex())
            print("cypher text1 in 16: ", newc1)
            print("cypher text2: ", c2)
            newc2=[]
            for i in c2:
                newc2.append(i.encode("cp1251").hex())
            print("cypher text2 in 16: ", newc2)
            print("open text1: ", p1)
            newp1=[]
            for i in p1:
                newp1.append(i.encode("cp1251").hex())
            print("open text1 in 16: ", newp1)
            xortmp=[]
            sp2=[]
            for i in range(len(p1)):
                xortmp.append("{:02x}".format(int(newc1[i],16) ^ int(newc2[i], 16)))
                sp2.append("{:02x}".format(int(xortmp[i],16) ^ int(newp1[i], 16)))
            print("open text2 in 16: ", sp2)
            p2=bytearray.fromhex("".join(sp2)).decode("cp1251")
            print("open text2: ", p2)
            return p1, p2
```

Рис. 0.4: функция расшифровки

Вывод функции: (рис. @fig:005)

```
In [6]: decrypt(et1, et2, p1)

        cypher text1:  •g{□.□%ўC_ъќЋx□rw
        cypher text1 in 16:  ['95', '67', '7b', '01', '2e', '1a', '25', 'a2', '43', '5f', '8c', '9d', '8e', 'f5', '1a', '00', 'e3', '7
        7']
        cypher text2:  ”~o□2Ш,sQЋ@’›ц□□ъh
        cypher text2 in 16:  ['94', '7e', '6f', '1e', '32', 'd8', '2c', 'be', '51', '8c', '40', '92', '9b', 'f6', '17', '1c', 'fa', '6
        8']
        open text1:  поставьте максимум
        open text1 in 16:  ['ef', 'ee', 'f1', 'f2', 'e0', 'e2', 'fc', 'f2', 'e5', '20', 'ec', 'e0', 'ea', 'f1', 'e8', 'ec', 'f3', 'ec']
        open text2 in 16:  ['ee', 'f7', 'e5', 'ed', 'fc', '20', 'f5', 'ee', 'f7', 'f3', '20', 'ef', 'ff', 'f2', 'e5', 'f0', 'ea', 'f3']
        open text2:  очень хочу пятерку

Out[6]: ('поставьте максимум', 'очень хочу пятерку')

In [8]: decrypt(et2, et1, p2)

        cypher text1:  ”~o□2Ш,sQЋ@’›ц□□ъh
        cypher text1 in 16:  ['94', '7e', '6f', '1e', '32', 'd8', '2c', 'be', '51', '8c', '40', '92', '9b', 'f6', '17', '1c', 'fa', '6
        8']
        cypher text2:  •g{□.□%ўC_ъќЋx□rw
        cypher text2 in 16:  ['95', '67', '7b', '01', '2e', '1a', '25', 'a2', '43', '5f', '8c', '9d', '8e', 'f5', '1a', '00', 'e3', '7
        7']
        open text1:  очень хочу пятерку
        open text1 in 16:  ['ee', 'f7', 'e5', 'ed', 'fc', '20', 'f5', 'ee', 'f7', 'f3', '20', 'ef', 'ff', 'f2', 'e5', 'f0', 'ea', 'f3']
        open text2 in 16:  ['ef', 'ee', 'f1', 'f2', 'e0', 'e2', 'fc', 'f2', 'e5', '20', 'ec', 'e0', 'ea', 'f1', 'e8', 'ec', 'f3', 'ec']
        open text2:  поставьте максимум

Out[8]: ('очень хочу пятерку', 'поставьте максимум')
```

Рис. 0.5: результат работы функции2

Так же проверил нахождение первого текста при известном втором, и, как видно, функция работает исправно.

Ответы на контрольные вопросы

1. Как, зная один из текстов (P1 или P2), определить другой, не зная при этом ключа?

Для этого надо воспользоваться формулой:
C1(+)C2(+)P1 = P1(+)P2(+)P1 = P2,
где C1 и C2 – шифротексты. Как видно, ключ в данной формуле не используется.

2. Что будет при повторном использовании ключа при шифровании текста?

В таком случае мы получим исходное сообщение.

3. Как реализуется режим шифрования однократного гаммирования одним ключом двух открытых текстов?

Он реализуется по следующей формуле:
C1=P1(+)K
C2=P2(+)K,
где Ci – шифротексты, Pi – открытые тексты, K – единый ключ шифрования.

4. Перечислите недостатки шифрования одним ключом двух открытых текстов.

Во-первых, имея на руках одно из сообщений в открытом виде и оба шифротекста, злоумышленник способен расшифровать каждое сообщение, не зная ключа.

Во-вторых, зная шаблон сообщений, злоумышленник получает возможность определить те символы сообщения P2, которые находятся на позициях известного шаблона сообщения P1.

В соответствии с логикой сообщения P2, злоумышленник имеет реальный шанс узнать ещё некоторое количество символов сообщения P2. Таким образом, применяя формулу из п. 1, с подстановкой вместо P1 полученных на предыдущем шаге новых символов сообщения P2 злоумышленник если не прочитает оба сообщения, то значительно уменьшит пространство их поиска. Наконец, зная ключ, злоумышленник смоет расшифровать все сообщения, которые были закодированы при его помощи.

5. Перечислите преимущества шифрования одним ключом двух открытых текстов.

Такой подход помогает упростить процесс шифрования и дешифровки. Также, при отправке сообщений между 2-я компьютерами, удобнее пользоваться одним общим ключом для передаваемых данных

Выводы

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