Отчет по лабораторной работе №7

Информационная безопасноть

Астафьева Анна Андреевна НПИбд-01-18

Содержание

Список иллюстраций

1 Цель работы

Освоить на практике применение режима однократного гаммирования.

2 Теоретическое описание

Предложенная Г. С. Вернамом так называемая «схема однократного использования (гаммирования)» является простой, но надёжной схемой шифрования данных. *Гаммирование* представляет собой наложение (снятие) на открытые (зашифрованные) данные последовательности элементов других данных, полученной с помощью некоторого криптографического алгоритма, для получения зашифрованных (открытых) данных. Иными словами, наложение гаммы — это сложение её элементов с элементами открытого (закрытого) текста по некоторому фиксированному модулю, значение которого представляет собой известную часть алгоритма шифрования. В соответствии с теорией криптоанализа, если в методе шифрования используется однократная вероятностная гамма (однократное гаммирование) той же длины, что и подлежащий сокрытию текст, то текст нельзя раскрыть. Даже при раскрытии части последовательности гаммы нельзя получить информацию о всём скрываемом тексте. Наложение гаммы по сути представляет собой выполнение операции сложения по модулю 2 (XOR) (обозначаемая знаком $\oplus$) между элементами гаммы и элементами подлежащего сокрытию текста. Напомним, как работает операция XOR над битами: $0 \oplus 0 = 0, 0 \oplus 1 = 1, 1 \oplus 0 = 1, 1 \oplus 1 = 0$. Такой метод шифрования является симметричным, так как двойное прибавление одной и той же величины по модулю 2 восстанавливает исходноезначение, а шифрование и расшифрование выполняется одной и той же программой. Если известны ключ и открытый текст, то задача нахождения шифротекста заключается в применении к каждому

символу открытого текста следующего правила:

$$C_i = P_i \oplus K_i$$

где C_i — i-й символ получившегося зашифрованного послания, P_i — i-й символ открытого текста, K_i — i-й символ ключа, i = 1, m. Размерности открытого текста и ключа должны совпадать, и полученный шифротекст будет такой же длины. Если известны шифротекст и открытый текст, то задача нахождения ключа решается также, а именно, обе части равенства необходимо сложить по модулю 2 с P_i:

$$C_i \oplus P_i = P_i \oplus K_i \oplus P_i = K_i, K_i = C_i \oplus P_i.$$

Открытый текст имеет символьный вид, а ключ — шестнадцатеричное представление. Ключ также можно представить в символьном виде, воспользовавшись таблицей ASCII-кодов. К. Шеннон доказал абсолютную стойкость шифра в случае, когда однократно используемый ключ, длиной, равной длине исходного сообщения, является фрагментом истинно случайной двоичной последовательности с равномерным законом распределения. Криптоалгоритм не даёт никакой информации об открытом тексте: при известном зашифрованном сообщении C все различные ключевые последовательности K возможны и равновероятны, а значит, возможны и любые сообщения P. Необходимые и достаточные условия абсолютной стойкости шифра: – полная случайность ключа; – равенство длин ключа и открытого текста; – однократное использование ключа.

3 Выполнение лабораторной работы

Требуется разработать приложение, позволяющее шифровать и дешифровать данные в режиме однократного гаммирования.

1. Написана функция *encryption*, которая с помощью однократного гаммирования из сообщения и ключа получает шифротекст (рис. 3.1).

```
def encryption (message, key):
    cypher=[]
    cypher_1=[]
    if len(message)>len(key):
        for _ in range(len(message)-len(key)):
            key.append('00')
    for i, j in zip(message, key):
        c = hex(int(i,16)^int(j,16))[2:]
        c = (c, '0'+c)[len(c)==1]
        cypher.append(c)
        cypher_1.append(chr(int(i,16)^int(j,16)))
    return cypher, cypher_1
```

Рис. 3.1: Код функции *encryption*

2. Написана функция *gen_key*, генерирующая случайный ключ (рис. 3.2).

```
from random import randrange

def gen_key (lenght):
    key=[]
    for _ in range(lenght):
        temp=randrange(256)
        temp=hex(temp)[2:]
        key.append((temp, '0'+temp)[len(temp)==1])
    return ' '.join(key)

#print(gen_key(20))
```

Рис. 3.2: Код функции *gen_key*

3. Написана функция *to_hex*, трансформирующая текст в шестнадцатиричное представление (рис. 3.3).

```
def to_hex (text):
    hexa=[]
    for i in text:
        hexa.append(hex(ord(i))[2:])
    return hexa
```

Рис. 3.3: Код функции *to_hex*

4. Определяю вид шифротекста при известном ключе и известном открытом тексте. Применяю к шифротексту ключ снова, чтобы получить исходное сообщение (рис. 3.4).

```
message='Лабораторная работа №7, Астафьева Анна Андреевна'
key=gen_key(len(message))
#key='01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01

key_m=key.split()
message_hex=to_hex(message)

print('Применение ключа к исходному сообщению. \nСообщение:\t\t\t %s \nКлюч:\t\t\t\t %s' %(message, key))
cypher_hex, cypher = encryption(message_hex, key_m)
cypher_hex=' '.join(cypher_hex)
cypher=''.join(cypher)
print('Зашифрованное сообщение:\t %s' %cypher_hex)
#print('Зашифрованное сообщение:\t %s' %cypher)

print('\nПрименение ключа к зашифрованному сообщению. \nЗашифрованное сообщение:\t %s \nКлюч:\t\t\t\t %s' %(cypher_hex, key))
mess_hex, mess = encryption(cypher_hex.split(), key_m)
mess=''.join(mess)
print('Расшифрованное сообщение:\t %s' %mess)
```

```
Применение ключа к исходному сообщению.
Сообщение:                 Лабораторная работа №7, Астафьева Анна Андреевна
Ключ:                      ea 3b 26 88 86 cc 76 e9 33 32 64 03 56 d5 53 09 3f 83 51 4a 8b b4 d3 fe fe 46 30 6e 8a af 03 9
8 3a 66 5e 49 cc 2b c4 1d 0d 39 83 f6 05 e8 ed d5
Зашифрованное сообщение:   4f1 40b 417 4b6 4c6 4fc 434 4d7 473 40f 454 44c 76 495 463 438 401 4c1 461 6a 219d 83 ff de 4e
e 407 472 45e 4ce 4e3 436 4aa 40a 46 44e 474 4f1 41b e4 40d 430 40d 4c3 4c3 430 4da 4d0 4e5

Применение ключа к зашифрованному сообщению.
Зашифрованное сообщение:   4f1 40b 417 4b6 4c6 4fc 434 4d7 473 40f 454 44c 76 495 463 438 401 4c1 461 6a 219d 83 ff de 4e
e 407 472 45e 4ce 4e3 436 4aa 40a 46 44e 474 4f1 41b e4 40d 430 40d 4c3 4c3 430 4da 4d0 4e5
Ключ:                      ea 3b 26 88 86 cc 76 e9 33 32 64 03 56 d5 53 09 3f 83 51 4a 8b b4 d3 fe fe 46 30 6e 8a af 03 9
8 3a 66 5e 49 cc 2b c4 1d 0d 39 83 f6 05 e8 ed d5
Расшифрованное сообщение:  Лабораторная работа №7, Астафьева Анна Андреевна
```

Рис. 3.4: Получение шифротекста

5. Пробую расшифровать шифротекст с помощью неправильного ключа(рис. 3.5).

```
wrong_key=gen_key(len(cypher_hex.split()))
print('\nПрименение неправильного ключа к зашифрованному сообщению. \nЗашифрованное сообщение:\t %s \nКлюч:\t\t\t\t %s' %(cypher_
mess_hex, mess = encryption(cypher_hex.split(), wrong_key.split())
mess=''.join(mess)
print('Расшифрованное сообщение:\t %s' %mess)
```

```
Применение неправильного ключа к зашифрованному сообщению.
Зашифрованное сообщение:   457 453 4c4 4ef 40f 471 435 4f3 428 457 47f 4d3 e8 478 473 439 494 490 44e 24 2108 81 98 d2 42
0 49b 49a 4e7 4bb 446 4c5 414 46f d3 4d1 4d5 45f 45e 9c 469 470 4d3 438 447 4a8 4b4 42f 4bd
Ключ:                      84 43 af 42 c6 ee 5e ff 1e ec 61 85 6f 5e 55 6f 20 39 e4 ea c9 2c ab fe 7b 09 2e b0 df 65 bf b
b 24 72 3f 9f bd df e7 60 96 eb 22 93 ea b0 ef cb
Расшифрованное сообщение:  äAѫҭҤҟѫЌжњOiЦЦiЦѻÇÎ→3,ћғЦïЄҮѺүы¦ЎьЙç{ЉÖиKÆтЄIѶ
```

Рис. 3.5: Применение неправильного ключа

6. Определяю ключ, с помощью которого шифротекст может быть преобразован в некоторый фрагмент текста, представляющий собой один из возможных вариантов прочтения открытого текста (Нужно подобрать ключ, чтобы получить сообщение «С Новым Годом, друзья!»)(рис. 3.6).

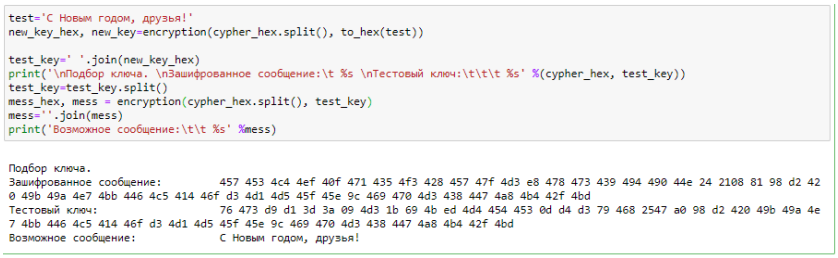

Рис. 3.6: Один из вариантов прочтения шифротекста

4 Контрольные вопросы

1. Поясните смысл однократного гаммирования. Смысл однократного гаммирования состоит в том, что каждый символ попарно с символом ключа складываются по модулю.

2. Перечислите недостатки однократного гаммирования. Недостатками является то, что ключ нельзя переиспользовать, а также размер ключа должен быть равен размеру текста.

3. Перечислите преимущества однократного гаммирования. Основными преимуществами являются симметричность и криптостойкость.

4. Почему длина открытого текста должна совпадать с длиной ключа? Каждый символ открытого текста должен попарно складываться с символом ключа.

5. Какая операция используется в режиме однократного гаммирования, назовите её особенности? В режиме однократного гаммирования используется сложение по модулю 2: при сложении чисел с другим получается исходное. Например, 0+0 = 0, 0+1=1, 1+0=1, 1+1=0. Если в методе шифрования используется однократная вероятностная гамма той же длины, что и подлежащий сокрытию текст, то текст нельзя раскрыть.

6. Как по открытому тексту и ключу получить шифротекст? Для этого необходимо сложить попарно символы текста с ключом по модулю 2.

7. Как по открытому тексту и шифротексту получить ключ? Для этого необходимо сложить попарно по модулю 2 символы открытого текста с символами

шифротекста.

8. В чём заключаются необходимые и достаточные условия абсолютной стойкости шифра? Необходимые и достаточные условия абсолютной стойкости шифра заключаются в полной случайности ключа; равенстве длин ключа и открытого текста; использовании ключа однократно.

5 Выводы

На основе проделанной работы освоила на практике применение режима однократного гаммирования.