Отчет по лабораторной работе №8

Дерябина Мария

2021

RUDN University, Moscow, Russian Federation

Цель работы

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

Теоретическая часть

Если даны две телеграммы Центра, то шифротексты обеих телеграмм можно получить по формулам режима однократного гаммирования:

$C1 = P1 \oplus K, C2 = P2 \oplus K$

Чтобы найти открытый текст, зная шифротекст двух телеграмм, зашифрованных одним ключом, надо сложить по модулю 2 эти два равенства. Тогда с учётом свойства операции XOR

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

получаем:

$C1 \oplus C2 = P1 \oplus K \oplus P2 \oplus K = P1 \oplus P2$

Теоретическая часть

Предположим, что одна из телеграмм является шаблоном — т.е. имеет текст фиксированный формат, в который вписываются значения полей. Допустим, что злоумышленнику этот формат известен. Тогда он получает достаточно много пар C1 ^ C2 (известен вид обеих шифровок). Тогда зная P1, имеем:

C1 ^ C2 ^ P1 = P1 ^ P2 ^ P1 = P2

Теоретическая часть

Таким образом, 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. Действуя подобным образом, злоумышленник даже если не прочитает оба сообщения, то значительно уменьшит пространство их поиска.

Выполнение лабораторной работы

Написала программу на языке Python, позволяющую шифровать и дешифровать данные в режиме однократного гаммирования. Программа имеет 3 функции:

1. decode(cr_message, key). Данная функция принимает зашифрованное сообщение и ключ (в виде строк с шестнадцатиричными значениями). Для каждого значения зашифрованного сообщения выполняется сложение по модулю 2 с сответствующим значением ключа. Функция возвращает строку с расшифрованным сообщением (рис. 1).

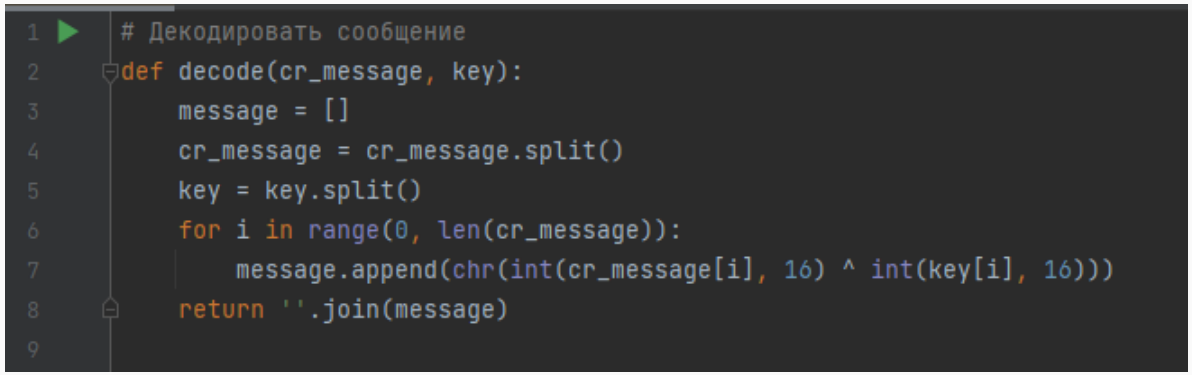

Figure 1: Функция для дешифрования сообщения

Выполнение лабораторной работы

2. encode(message, key). Данная функция принимает исходное сообщение и ключ. Каждый символ сообщения преобразовывется в число, соответствующее его коду в системе Unicode. Далее выполняется сложение по модулю 2 между получившимися кодами и соответствующими значениями ключа. Функция возвращает зашифрованное сообщение в виде строки с шестнадцатиричными значениями (рис. 2).

```
11  # Закодировать сообщение
12  def encode(message, key):
13      cr_message = []
14      key = key.split()
15      for i in range(0, len(message)):
16          cr_message.append((hex(ord(message[i]) ^ int(key[i], 16)).lstrip('0x')).upper())
17          if len(cr_message[i]) == 1:
18              cr_message[i] = '0' + cr_message[i]
19      return ' '.join(cr_message)
```

Figure 2: Функция для шифрования сообщения

Выполнение лабораторной работы

3. get_message(cr_message1, cr_message2, message2). Данная функция принимает зашифрованное сообщение, шаблон исходного сообщения и зашифрованное шаблонное сообщение. Выполняется сложение по модулю 2 между значениями закодированного сообщения, кодами символов шаблонного сообщения и значениями закодированного шаблонного сообщения. Функция возвращает строку с расшифрованным сообщением (рис. 3).

```
22  def get_message(cr_message1, cr_message2, message2):
23      message1 = []
24      cr_message1 = cr_message1.split()
25      cr_message2 = cr_message2.split()
26      for i in range(0, len(cr_message1)):
27          message1.append(chr(int(cr_message1[i], 16) ^ int(cr_message2[i], 16) ^ ord(message2[i])))
28      return ''.join(message1)
```

Figure 3: Функция для дешифрования сообщения без ключа

Результаты

Протестировала программу на сообщениях ‘С Новым Годом, друзья!’ и ‘Желаю счастья и любви!’. Вначале программа определила вид шифротекста сообщений (при одинаковом ключе). Далее была вызвана функция get_message(), в которую были переданы шифротекст первого сообщения, а также исходный текст и шифротекст второго сообщения. После обработки этих данных, функция корректно определила исходный текст первого сообщения (рис. 4).

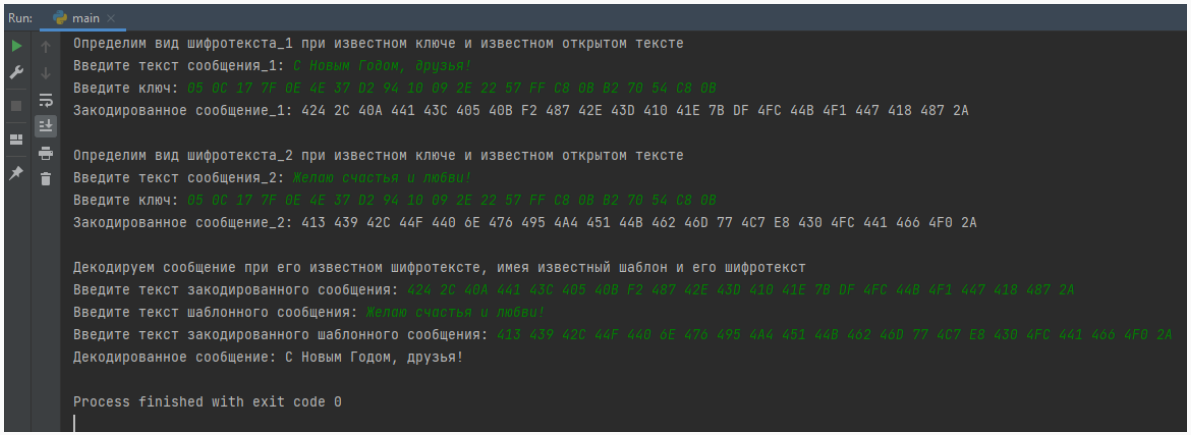

Figure 4: Тестирование программы

Выводы

Я освоила на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом. Определила способ, при котором злоумышленник может прочитать оба текста, не зная ключа и не стремясь его определить