Лабораторная работа №7

Однократное гаммирование

Доборщук В.В.

НФИбд-01-18

11 декабря 2021

Содержание

# Цель работы

Освоить на практике применение режима однократного гаммирования.

# Выполнение лабораторной работы

Нужно подобрать ключ, чтобы получить сообщение «С Новым Годом, друзья!». Требуется разработать приложение, позволяющее шифровать и дешифровать данные в режиме однократного гаммирования. Приложение должно:

1. Определить вид шифротекста при известном ключе и открытом тексте.
2. Определить ключ, с помощью которого шифротекст может быть преобразован в некоторый фрагмент текста, представляющий собой один из возможных вариантов прочтения открытого текста

## Реализация функционала

Создали дополнительную функцию для генерации случайного ключа:

def gen\_key(text):  
 rn = np.random.randint(0, 255, len(text))  
 key = [hex(e)[2:] for e in rn]  
 return key

Реализована следующая функция для определения вида шифротекста при известных ключе и открытом тексте. На вход подается строка, после чего она переводится в 16-ричную систему, после чего в программе генерируется ключ случайным образом. С его помощью получается зашифрованное сообщение в шестнадцатеричной системе счисления.

def crypt\_message(open\_text, key):  
 print(f"Open Text: {open\_text}")  
 hex\_open\_text = []  
 for ch in open\_text:  
 hex\_open\_text.append(ch.encode("cp1251").hex())  
   
 print("Hex Open Text: ", \*hex\_open\_text)  
  
 print("Key: ", \*key)  
 hex\_crypted\_text = []  
 for i in range(len(hex\_open\_text)):  
 hex\_crypted\_text.append("{:02x}".format(int(key[i], 16)^int(hex\_open\_text[i], 16)))  
   
 print("Hex Crypted Text: ", \*hex\_crypted\_text)  
 crypted\_text = bytearray.fromhex("".join(hex\_crypted\_text)).decode("cp1251")  
 print(f"Crypted Text: {crypted\_text}")  
   
 return crypted\_text

Далее определяем ключ, с помощью которого шифротекст может быть преобразован в некоторый фрагмент текста, представляющий собой один из возможных вариантов прочтения открытого текста. Функция получает на вход открытый текст и зашифрованный, после чего преобразует строки в 16-ричный формат и выполняет операцию сложения по модулю 2 (XOR).

def find\_key(open\_text, crypted\_text):  
 print(f"Open Text: {open\_text}\nCrypted Text: {crypted\_text}")  
 hex\_open\_text = []  
 for ch in open\_text:  
 hex\_open\_text.append(ch.encode("cp1251").hex())  
   
 hex\_crypted\_text = []  
 for ch in crypted\_text:  
 hex\_crypted\_text.append(ch.encode("cp1251").hex())  
   
 print("Hex Open Text: ", \*hex\_open\_text)  
 print("Hex Crypted Text: ", \*hex\_crypted\_text)  
   
 key = [hex(int(i,16)^int(j,16))[2:] for (i,j) in zip(hex\_open\_text, hex\_crypted\_text)]  
 print("Key ", \*key)  
   
 return key



Figure : Реализованные функции

С помощью реализованного функционала проверяем работоспособность нашего приложения:

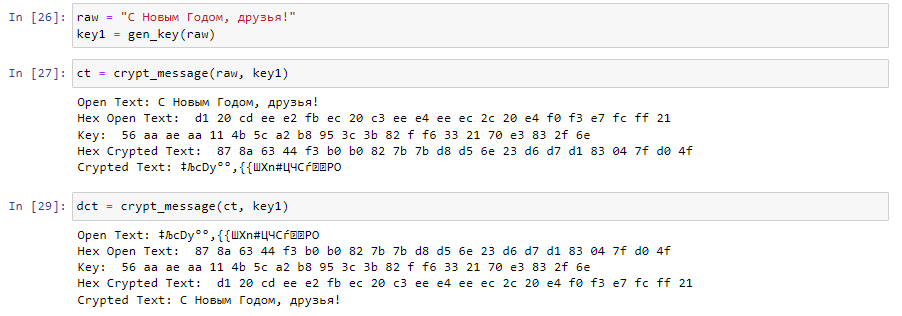


Figure : Проверка шифрования

Все корректно отрабатывает с одним и тем же ключом.

## Проверка ключа

В конце проверяем совпадают ли полученный ключ для идентичного и неидентичного текста и начальный ключ.

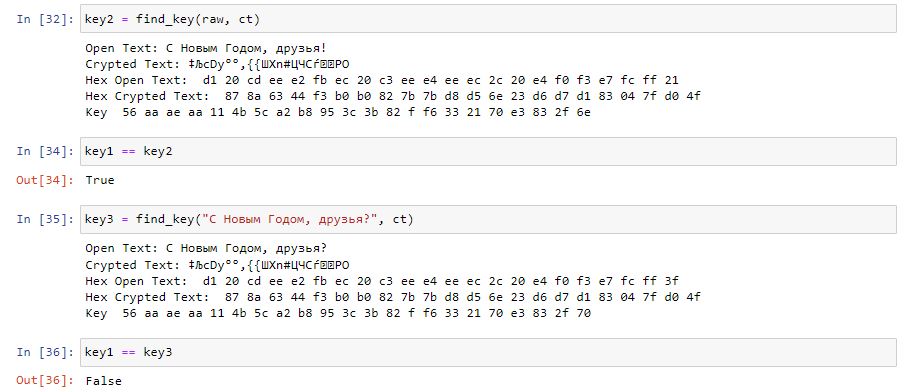


Figure : Сравнение ключей

Получили, что для идентичного текста ключи совпадают, а для неидентичного, соответственно, не совпадают.

# Контрольные вопросы

1. Поясните смысл однократного гаммирования.

**Гаммирование** – выполнение операции XOR между элементами гаммы и элементами подлежащего сокрытию текста. 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.

1. Перечислите недостатки однократного гаммирования.

Абсолютная стойкость шифра доказана только для случая, когда однократно используемый ключ, длиной, равной длине исходного сообщения, является фрагментом истинно случайной двоичной последовательности с равномерным законом распределения.

1. Перечислите преимущества однократного гаммирования.

* Во-первых, такой способ симметричен, т.е. двойное прибавление одной и той же величины по модулю 2 восстанавливает исходное значение.
* Во-вторых, шифрование и расшифрование может быть выполнено одной и той же программой. Наконец, 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.

1. Почему длина открытого текста должна совпадать с длиной ключа?

Если ключ короче текста, то операция XOR будет применена не ко всем элементам и конец сообщения будет не закодирован. Если ключ будет длиннее, то появится неоднозначность декодирования.

1. Какая операция используется в режиме однократного гаммирования, назовите её особенности?

Наложение гаммы по сути представляет собой выполнение побитовой операции сложения по модулю 2, т.е. мы должны сложить каждый элемент гаммы с соответствующим элементом ключа. Данная операция является симметричной, так как прибавление одной и той же величины по модулю 2 восстанавливает исходное значение.

1. Как по открытому тексту и ключу получить шифротекст?

В таком случае задача сводится к правилу:

т.е. мы поэлементно получаем символы зашифрованного сообщения, применяя операцию исключающего или к соответствующим элементам ключа и открытого текста.

1. Как по открытому тексту и шифротексту получить ключ?

Подобная задача решается путем применения операции исключающего или к последовательностям символов зашифрованного и открытого сообщений:

1. В чем заключаются необходимые и достаточные условия абсолютной стойкости шифра?

Необходимые и достаточные условия абсолютной стойкости шифра:

* полная случайность ключа;
* равенство длин ключа и открытого текста;
* однократное использование ключа.

# Выводы

Мы изучили на практике применение режима однократного гаммирования.