Отчёт по лабораторной работе №7

Шифр гаммирования

Кирилл Дроздков НПИбд-02-19

Содержание

# 1 Цель работы

Изучение алгоритма шифрования гаммированием

# 2 Теоретические сведения

## 2.1 Шифр гаммирования

Гаммирование – это наложение (снятие) на открытые (зашифрованные) данные криптографической гаммы, т.е. последовательности элементов данных, вырабатываемых с помощью некоторого криптографического алгоритма, для получения зашифрованных (открытых) данных.

Принцип шифрования гаммированием заключается в генерации гаммы шифра с помощью датчика псевдослучайных чисел и наложении полученной гаммы шифра на открытые данные обратимым образом (например, используя операцию сложения по модулю 2). Процесс дешифрования сводится к повторной генерации гаммы шифра при известном ключе и наложении такой же гаммы на зашифрованные данные. Полученный зашифрованный текст является достаточно трудным для раскрытия в том случае, если гамма шифра не содержит повторяющихся битовых последовательностей и изменяется случайным образом для каждого шифруемого слова. Если период гаммы превышает длину всего зашифрованного текста и неизвестна никакая часть исходного текста, то шифр можно раскрыть только прямым перебором (подбором ключа). В этом случае криптостойкость определяется размером ключа.

Метод гаммирования становится бессильным, если известен фрагмент исходного текста и соответствующая ему шифрограмма. В этом случае простым вычитанием по модулю 2 получается отрезок псевдослучайной последовательности и по нему восстанавливается вся эта последовательность.

Метод гаммирования с обратной связью заключается в том, что для получения сегмента гаммы используется контрольная сумма определенного участка шифруемых данных. Например, если рассматривать гамму шифра как объединение непересекающихся множеств H(j), то процесс шифрования можно пердставить следующими шагами:

1. Генерация сегмента гаммы H(1) и наложение его на соответствующий участок шифруемых данных.
2. Подсчет контрольной суммы участка, соответствующего сегменту гаммы H(1).
3. Генерация с учетом контрольной суммы уже зашифрованного участка данных следующего сегмента гамм H(2).
4. Подсчет контрольной суммы участка данных, соответствующего сегменту данных H(2) и т.д.

# 3 Выполнение работы

## 3.1 Реализация шифратора и дешифратора Python

def xor\_cipher(filename, outfilename, string):  
 '''  
 xor\_cipher: функция, которая производит операцию XOR\  
 с каждым символом из файла filename и строкой string.  
 Если символов в файле больше, чем в строке string,  
 то строка сдвигается влево на один символ  
 Результат записывается в файл outfilename  
 '''  
 with open(outfilename, "w") as filewrite:  
 with open(filename) as file:  
 if not file:  
 print("no file")  
 exit(1)  
 # Главное смещение (относительно начала строки)  
 main\_offset = 0  
 # Относительное смещение (+=1, когда строка заканчивается)  
 str\_offset = 0  
 # Чтение файла построчно  
 for line in file:  
 res\_str = ""  
 # Посимвольное гаммирование  
 for i in range(len(line)):  
 if main\_offset!=0 and main\_offset % len(string) == 0:  
 str\_offset += 1  
 res\_str += chr(ord(line[i])^ord(string[(main\_offset+str\_offset)%len(string)]))  
 main\_offset += 1  
 # Запись в файл  
 filewrite.write(res\_str)  
def main():  
 '''  
 Main function  
 '''  
 string = input("Введите строку:\n")  
 # string = "test string"  
 xor\_cipher("test.txt","output.txt", string)  
 xor\_cipher("output.txt","output#2.txt", string)  
 print("DONE!")  
 input("Нажмите Enter для завершения программы")

## 3.2 Контрольный пример

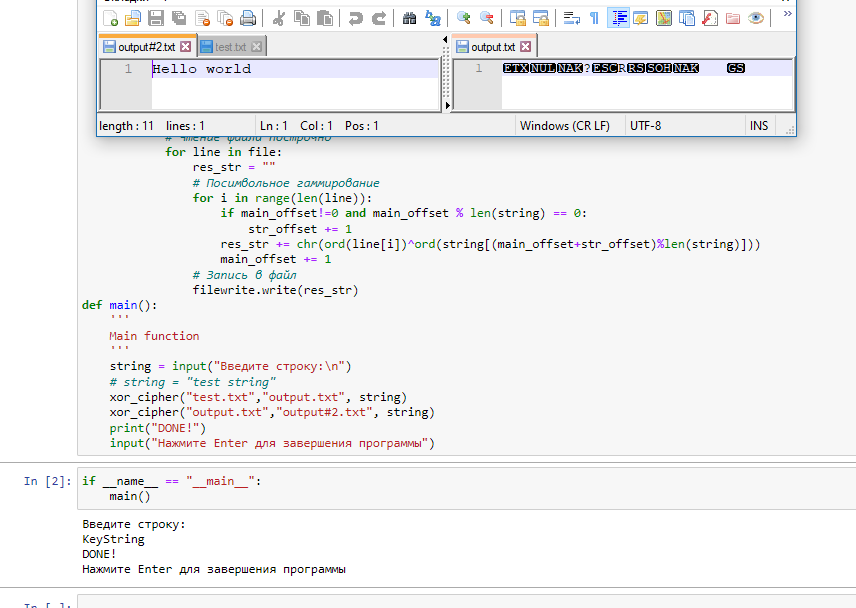


Figure 1: Работа алгоритма гаммирования

# 4 Выводы

Изучили алгоритмы шифрования на основе гаммирования

# Список литературы

1. [Шифрование методом гаммирования](http://altaev-aa.narod.ru/security/XOR.html)
2. [Режим гаммирования в блочном алгоритме шифрования](https://kabinfo.ucoz.ru/index/shifr_reshetka_kardano/0-374)