Отчёт по лабораторной работе

Лабораторная №8

Панкратьев Александр Владимирович

Содержание

# 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 ^ K, C2 = P2 ^ 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 ^ 1 = 0, 1 ^ 0 = 1 ,

получаем:

C1 ^ C2 = P1 ^ K ^ P2 ^ K = P1 ^ P2.

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.

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

Написал 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).

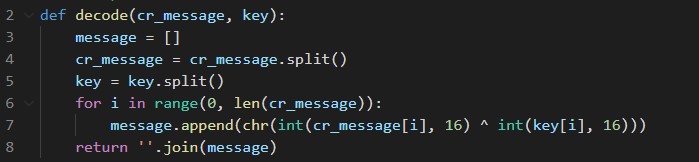


Figure 1: Функция для дешифрования сообщения

1. def 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).

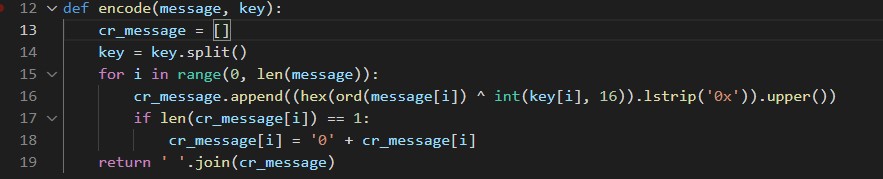


Figure 2: Функция для шифрования сообщения

1. 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, с помощью которого исходный текст был закодирован (рис. 3).

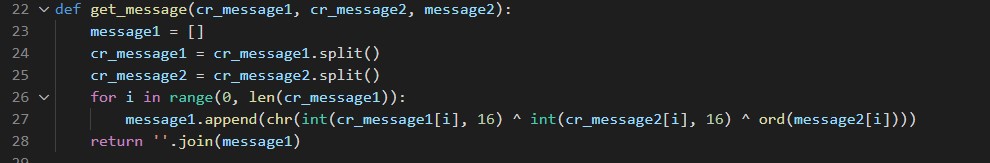


Figure 3: Функция для дешифрования сообщения без ключа

Написал код с вызовом функций для тестирования (рис. 4).

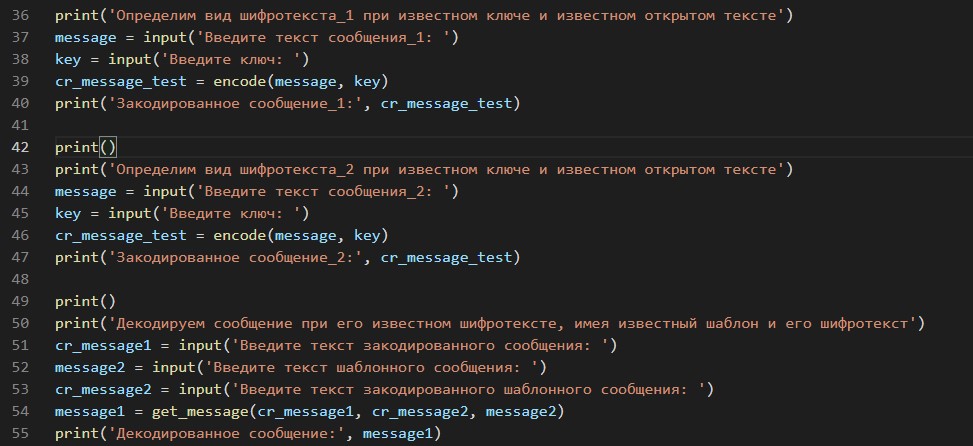


Figure 4: Вызов функций для тестирования

Протестировал программу на сообщенииях ‘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 сообшения. После обработки этих данных, функция корректно определила исходный текст первого сообщения (рис. 5).

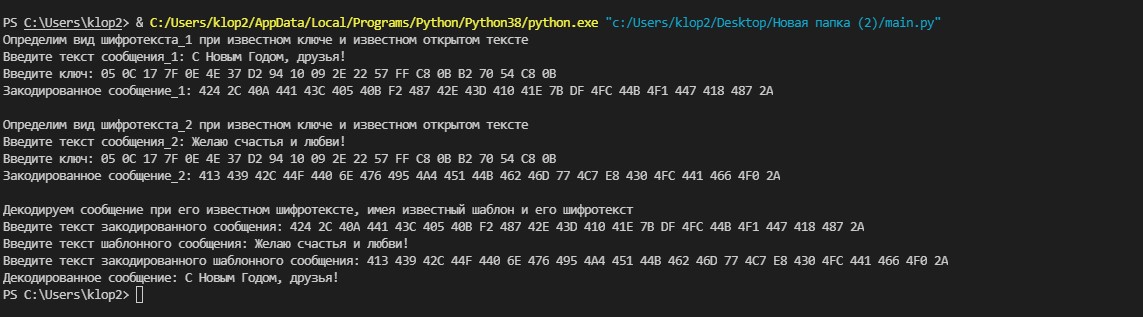


Figure 5: Тестирование программы

# Вывод

Я освоил на практике применение режима однократного гаммированияна примере кодирования различных исходных текстов одним ключом. Определила способ, при котором злоумышленник может прочитать оба текста, не зная ключа.