Отчет по лабораторной работе №7

Дисциплина: Информационная безопасность

Выполнила Дяченко Злата Константиновна, НФИбд-03-18

Содержание

# Цель работы

Освоить на практике применение режима однократного гаммирования.

# Задание

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 известном 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.

# Теоретические вводные данные

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 между элементами гаммы и элементами подлежащего сокрытию текста. Если известны ключ и открытый текст, то задача нахождения шифротекста заключается в применении к каждому символу открытого текста следующего правила:

где — i-й символ получившегося зашифрованного послания, — i-й символ открытого текста, — i-й символ ключа. Размерности открытого текста и ключа должны совпадать, и полученный шифротекст будет такой же длины. Если известны шифротекст и открытый текст, то, чтобы найти ключ, обе части равенства необходимо сложить по модулю 2 с :

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.

# Выполнение лабораторной работы

## Шаг 1

Писать программу решила на языке C++. Были написаны функции, показанные на Рисунке 1 (рис. 1). Функция convert преобразует символы открытого текста в их коды в ASCCI. Функция printM выводит массив данных (в нашем случае коды символов открытого текста, зашифрованный текст, ключ). Функция keyGen генерирует случайный ключ определенной длины (я генерировала ключ той же длины, что и открытый текст). Функция gammir осуществляет однократное гаммирование - в результате получаем шифротекст. Функция getKey получает ключ, зная шифротекст и открытый текст.

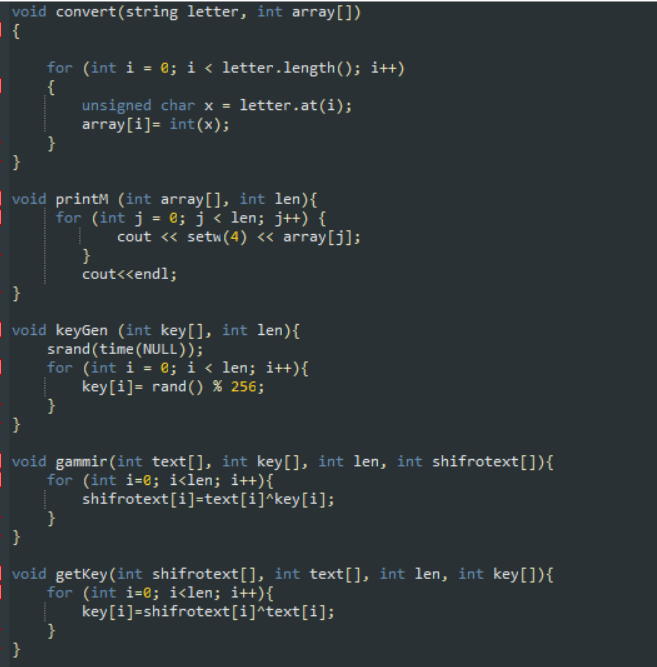


Figure 1: Функции

## Шаг 2

В функции main создаем переменную text с открытым текстом “С Новым Годом, друзья!” (рис. 2). Применяя описанные ранее функции, генерируем ключ и получаем шифротекст.



Figure 2: Генерация ключа и осуществление однократного гаммирования

## Шаг 3

Результат шифрования представлен на Рисунке 3 (рис. 3).

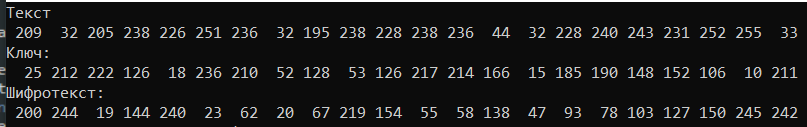


Figure 3: Открытый текст, сгенерированный ключ и шифротекст

## Шаг 4

Функцию getKey можно проверить, передав в нее полученный ранее шифротекст и тот открытый текст, который был зашифрован. Получим тот же самый ключ (рис. 5). Также можем найти ключ, который необходим для того, чтобы расшифровать полученный шифротекст в открытый текст “С Новой бедой, друзья!” (рис. 4)

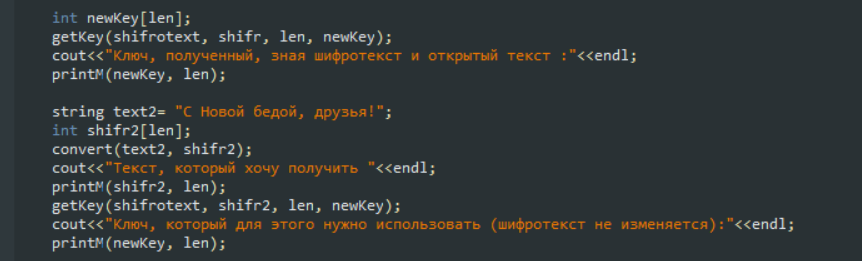


Figure 4: Поиск ключа

## Шаг 4

Результаты выполнения функций представлены на Рисунке 5 (рис. 5).

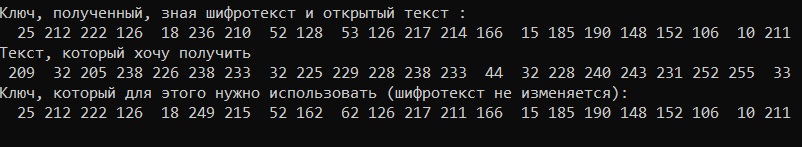


Figure 5: Полученный ключ

# Выводы

В результате работы я освоила на практике применение режима однократного гаммирования. Результаты работы находятся в [репозитории на GitHub](https://github.com/ZlataDyachenko/workD), а также есть [скринкаст выполнения лабораторной работы](https://www.youtube.com/watch?v=-JJetytTA5o).