Отчёт по лабораторной работе №7

Элементы криптографии. Однократное гаммирование

Тимур Дмитриевич Калинин

Содержание

List of Figures

List of Tables

1 Цель работы

Освоить на практике применение режима однократного гаммирования.

2 Выполнение лабораторной работы

1. Напишем программу encryptor.c, которая будет выводить шифротекст по известному тексту и ключу. Шифрование будет заключаться в применении операции побитовой XoR ко всем символом строки. (Рис. 2.1)

lab07 > C encryptor.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>


char* encryptor(char* original_str, char* key, int length){
    char* encrypted_str = (char *)malloc(length*sizeof(char));
    int i;
    for(i=0; i<length; i++){
        encrypted_str[i] = (char)(original_str[i] ^ key[i]);
    }
    return encrypted_str;
}

int main(){
    int length = 64;
    char initial_string[length];
    char key[length];

    printf("Введите строку: ");
    fgets(initial_string, length, stdin);
    printf("Введите ключ: ");
    fgets(key, length, stdin);

    char *encrypted_str = encryptor(initial_string, key, length);
    printf("Шифрованный текст: %s\n", encrypted_str);

    char *decrypted_str = encryptor(encrypted_str, key, length);
    printf("Расшифрованный текст: %s\n", decrypted_str);
    free(encrypted_str);

    return 0;
}
```

Figure 2.1: Код encryptor.c

2. Напишем аналогичную программу key_finder, которая будет выводить ключ по известному тексту и зашифрованному тексту (Рис. 2.2)

lab07 > C key_finder.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>


char* encryptor(char* original_str, char* key, int length){
    char* encrypted_str = (char *)malloc(length*sizeof(char));
    int i;
    for(i=0; i<length; i++){
        encrypted_str[i] = (char)(original_str[i] ^ key[i]);
    }
    return encrypted_str;
}

int main(){
    int length = 64;
    char initial_string[length];
    char key[length];

    printf("Введите строку: ");
    fgets(initial_string, length, stdin);
    printf("Введите зашифрованную строку: ");
    fgets(key, length, stdin);

    char *encrypted_str = encryptor(initial_string, key, length);
    printf("Ваш ключ: %s\n", encrypted_str);

    free(encrypted_str);

    return 0;
}
```

Figure 2.2: Код key_finder.c

3. Проверим работу программы encryptor (Рис. 2.3)

```
[tdkalinin@tdkalinin lab07]$ ./encryptor
Введите строку: Happy new year, friends!
Введите ключ: 00000000000000000000000000000000
Шифрованный текст: xQ@@I^UGIUQBVBYU^TC:0

Расшифрованный текст: Happy new year, friends!
```

Figure 2.3: Работа программы encryptor

4. Проверим работу программы key_finder (Рис. 2.4)

```
[tdkalinin@tdkalinin lab07]$ ./key_finder
Введите строку: Happy new year, friends!
Введите зашифрованную строку: vxz^fgasdfg230432vasd
Ваш ключ: >
.GFWRBT
ns!
```

Figure 2.4: Работа программы key_finder

3 Контрольные вопросы

1. Поясните смысл однократного гаммирования.

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.

2. Перечислите недостатки однократного гаммирования.

Необходимость иметь огромные объемы данных, которые можно было бы использовать в качестве гаммы.

3. Перечислите преимущества однократного гаммирования.

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.

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

Для абсолютной стойкости шифра

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

Исключающее или

6. Как по открытому тексту и ключу получить шифротекст?

Применить побитовую операцию “исключающее или” между символами текста

7. Как по открытому тексту и шифротексту получить ключ?

Применить побитовую операцию “исключающее или” между символами текста

8. 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.

4 Выводы

Мы освоили на практике применение режима однократного гаммирования.

5 Библиография

1. Лабораторная работа №7. - 5 с. URL: https://esystem.rudn.ru/pluginfile.php/1651893/mod_resource/content/2/007-lab_crypto-gamma.pdf