Отчет по лабораторной работе №8

Дисциплина: Информационная безопасность

Выполнила Дяченко Злата Константиновна, НФИбд-03-18

Содержание

List of Figures

1 Цель работы

Освоить на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.

2 Задание

Два текста кодируются одним ключом (однократное гаммирование). Требуется не зная ключа и не стремясь его определить, прочитать оба текста. Необходимо разработать приложение, позволяющее шифровать и дешифровать тексты P1 и P2 в режиме однократного гаммирования. Приложение должно определить вид шифротекстов C1 и C2 обоих текстов P1 и P2 при известном ключе; Необходимо определить и выразить аналитически способ, при котором злоумышленник может прочитать оба текста, не зная ключа и не стремясь его определить.

3 Теоретические вводные данные

Шифротексты обеих телеграмм можно получить по формулам режима однократного гаммирования:

$$C_1 = P_1 \oplus K$$

$$C_2 = P_2 \oplus K$$

Открытый текст можно найти, зная шифротекст двух телеграмм, зашифрованных одним ключом.

$$C_1 \oplus C_2 = P_1 \oplus P_2$$

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 XOR C2 (известен вид обеих шифровок). Тогда зная P1, имеем:

$$C_1 \oplus C_2 \oplus P_1 = P_2$$

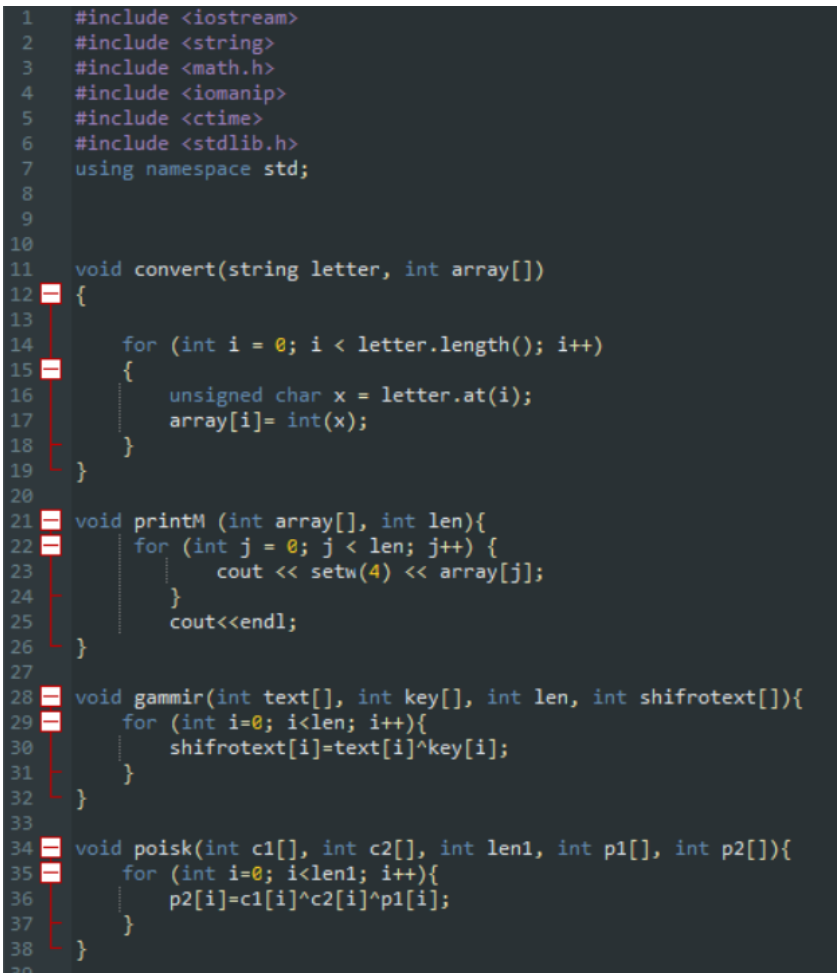
Таким образом, злоумышленник получает возможность определить те символы сообщения P2, которые находятся на позициях известного шаблона сообщения P1. В соответствии с логикой сообщения P2, злоумышленник имеет реальный шанс узнать ещё некоторое количество символов сообщения P2. Затем вновь используется формула с подстановкой вместо P1 полученных на предыдущем шаге новых символов сообщения P2. И так далее. Действуя подобным образом, зло-

умышленник даже если не прочитает оба сообщения, то значительно уменьшит пространство их поиска.

4 Выполнение лабораторной работы

4.1 Шаг 1

Писать программу решила на языке C++. Были написаны функции, показанные на Рисунке 1 (рис. 4.1). 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 gammir осуществляет однократное гаммирование. Функция poisk находит открытый текст P2, если известны шифротексты C1, C2 и открытый текст P1.



```cpp
#include <iostream>
#include <string>
#include <math.h>
#include <iomanip>
#include <ctime>
#include <stdlib.h>
using namespace std;



void convert(string letter, int array[])
{

    for (int i = 0; i < letter.length(); i++)
    {
        unsigned char x = letter.at(i);
        array[i]= int(x);
    }
}

void printM (int array[], int len){
    for (int j = 0; j < len; j++) {
        cout << setw(4) << array[j];
    }
    cout<<endl;
}

void gammir(int text[], int key[], int len, int shifrotext[]){
    for (int i=0; i<len; i++){
        shifrotext[i]=text[i]^key[i];
    }
}

void poisk(int c1[], int c2[], int len1, int p1[], int p2[]){
    for (int i=0; i<len1; i++){
        p2[i]=c1[i]^c2[i]^p1[i];
    }
}
```

Figure 4.1: Функции

4.2 Шаг 2

В функции main создаем переменные p1 и p2, содержащие открытый текст (рис. 4.2). Применяя описанную ранее функцию convert, переводим символы в их коды. С помощью массива K задаем ключ K = 05 0C 17 7F 0E 4E 37 D2 94 10 09 2E 22 57 FF C8 OB B2 70 54. С помощью однократного гаммирования шифруем сообщения.

```
int main(){
    setlocale(LC_ALL, "Russian");
    string p1,p2;
    int len1;
    p1="НаВашисходящийот1204";
    p2="ВСеверныйфилиалБанка";
    len1=p1.size();
    int s1[len1], s2[len1];
    convert(p1, s1);
    convert(p2,s2);
    cout<<"P1"<<endl;
    printM(s1, len1);
    cout<<"P2"<<endl;
    printM(s2, len1);

    int K[len1];
    K[0]=0x05;
    K[1]=0x0C;
    K[2]=0x17;
    K[3]=0x7F;
    K[4]=0x0E;
    K[5]=0x4E;
    K[6]=0x37;
    K[7]=0xD2;
    K[8]=0x94;
    K[9]=0x10;
    K[10]=0x09;
    K[11]=0x2E;
    K[12]=0x22;
    K[13]=0x57;
    K[14]=0xFF;
    K[15]=0xC8;
    K[16]=0x0B;
    K[17]=0xB2;
    K[18]=0x70;
    K[19]=0x54;

    int c1[len1], c2[len1];
    gammir(s1, K, len1, c1);
    cout<<"Шифротекст C1:"<<endl;
    printM(c1, len1);
    gammir(s2, K, len1, c2);
    cout<<"Шифротекст C2:"<<endl;
    printM(c2, len1);
```

Figure 4.2: Осуществление однократного гаммирования

4.3 Шаг 3

С помощью функции gammir осуществляем также дешифрование сообщений (рис. 4.3).

```
84
85      int d1[len1],d2[len1];
86      gammir(c1, K, len1, d1);
87      cout<<"Дешифрованный текст P1:"<<endl;
88      printM(d1, len1);
89      gammir(c2, K, len1, d2);
90      cout<<"Дешифрованный текст P12:"<<endl;
91      printM(d2, len1);
92
```

Figure 4.3: Дешифровка

4.4 Шаг 3

Результаты выполнения функций представлены на Рисунке 4 (рис. 4.4).

```
P1
 205 224 194 224 248 232 241 245 238 228 255 249 232 233 238 242  49  50  48  52
P2
 194 209 229 226 229 240 237 251 233 244 232 235 232 224 235 193 224 237 234 224
Шифротекст C1:
 200 236 213 159 246 166 198  39 122 244 246 215 202 190  17  58  58 128  64  96
Шифротекст C2:
 199 221 242 157 235 190 218  41 125 228 225 197 202 183  20   9 235  95 154 180
Дешифрованный текст P1:
 205 224 194 224 248 232 241 245 238 228 255 249 232 233 238 242  49  50  48  52
Дешифрованный текст P12:
 194 209 229 226 229 240 237 251 233 244 232 235 232 224 235 193 224 237 234 224
```

Figure 4.4: Результаты шифровки и дешифровки

4.5 Шаг 4

Для того, чтобы прочитать один из текстов, зная другой, применим функцию poisk (рис. 4.5).

```
92
93      int p22[len1];
94      poisk(c1,c2,len1,s1,p22);
95      cout<<"Прочитанный P2:"<<endl;
96      printM(p22, len1);
97
98  }
```

Figure 4.5: Чтение текста P2

4.6 Шаг 4

Мы получили верный текст P2, что видно на Рисунке 6 (рис. 4.6).

```
P2
 194 209 229 226 229 240 237 251 233 244 232 235 232 224 235 193 224 237 234 224
Шифротекст C1:
 200 236 213 159 246 166 198  39 122 244 246 215 202 190  17  58  58 128  64  96
Шифротекст C2:
 199 221 242 157 235 190 218  41 125 228 225 197 202 183  20   9 235  95 154 180
Дешифрованный текст P1:
 205 224 194 224 248 232 241 245 238 228 255 249 232 233 238 242  49  50  48  52
Дешифрованный текст P12:
 194 209 229 226 229 240 237 251 233 244 232 235 232 224 235 193 224 237 234 224
Прочитанный P2:
 194 209 229 226 229 240 237 251 233 244 232 235 232 224 235 193 224 237 234 224
```

Figure 4.6: Прочитанный P2

5 Выводы

В результате работы я освоила на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом. Результаты работы находятся в репозитории на GitHub, а также есть скринкаст выполнения лабораторной работы.