

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 7

Дисциплина: Информационная безопасность

Тема: Элементы криптографии. Однократное гаммирование

Студент: Ломакина София Васильевна

Группа: НФИбд-02-19

МОСКВА

2022 г.

| | |
|---------------------------------------|----------|
| Цель работы | 3 |
| Теоретическая справка | 3 |
| Выполнение лабораторной работы | 4 |
| Контрольные вопросы | 6 |
| Вывод | 7 |

Цель работы

Освоить на практике применение режима однократного гаммирования.

Теоретическая справка

Предложенная Г. С. Вернамом так называемая «схема однократного использования (гаммирования)» является простой, но надёжной схемой шифрования данных. 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 (XOR) (обозначаемая знаком \oplus) между элементами гаммы и элементами подлежащего сокрытию текста. Напомним, как работает операция XOR над битами: $0 \oplus 0 = 0$, $0 \oplus 1 = 1$, $1 \oplus 0 = 1$, $1 \oplus 1 = 0$. Такой метод шифрования является симметричным, так как двойное прибавление одной и той же величины по модулю 2 восстанавливает исходное значение, а шифрование и расшифрование выполняется одной и той же программой.

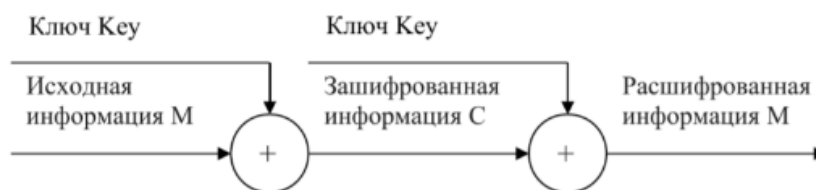


Рис. 0.1: Схема однократного использования Вернама

Если известны ключ и открытый текст, то задача нахождения шифротекста заключается в применении к каждому символу открытого текста следующего правила:

$$C_i = P_i \oplus K_i$$

, где C_i — i -й символ получившегося зашифрованного послания, P_i — i -й символ открытого текста, K_i — i -й символ ключа, $i = 1, m$. Размерности открытого текста и ключа должны совпадать, и полученный шифротекст будет такой же длины. Если известны шифротекст и открытый текст, то задача нахождения ключа решается также в соответствии с (7.1), а именно, обе части равенства необходимо сложить по модулю 2 с P_i :

$$C_i \oplus P_i = P_i \oplus K_i \oplus P_i = K_i,$$

$$K_i = C_i \oplus P_i$$

Открытый текст имеет символьный вид, а ключ — шестнадцатеричное представление. Ключ также можно представить в символьном виде, воспользовавшись таблицей ASCII-кодов. 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 дает 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

Написала следующую программу на C++, шифрующую введенное сообщение при

помощи случайно сгенерированного ключа.

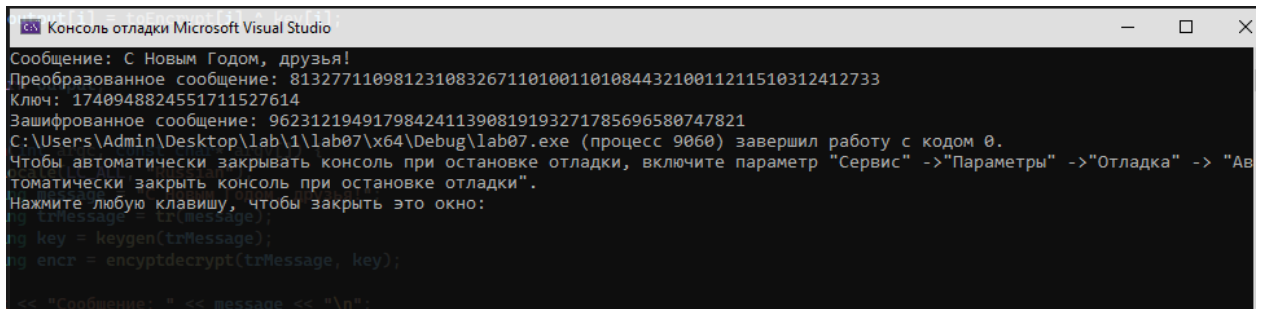
```
lab07 (Глобальная область)
1 // lab07.cpp : Этот файл содержит функцию "main". Здесь начинается и заканчивается выполнение программы.
2 //
3
4 #include <iostream>
5 #include <vector>
6 #include <string>
7
8 using namespace std;
9
10 string tr(string toEncrypt) {
11     string trMessage;
12     for (int i = 0; i < toEncrypt.size(); i++) {
13         int j = toascii(toEncrypt.at(i));
14         trMessage.push_back(j);
15     }
16     return trMessage;
17 }
18
19 string keygen(string toEncrypt) {
20     string key;
21     static const char code[] = "0123456789";
22
23     for (int i = 0; i < toEncrypt.size(); i++) {
24         key += code[rand() % (sizeof(code) - 1)];
25     }
26     return key;
27 }
28
29 string encryptdecrypt(string toEncrypt, string key) {
30     string output = toEncrypt;
31     for (int i = 0; i < toEncrypt.size(); i++) {
32         output[i] = toEncrypt[i] ^ key[i];
33     }
34
35     return output;
36 }
```

Рис. 0.1: Функции программы-шифратора

```
38 int main(int argc, const char* argv[]) {
39     setlocale(LC_ALL, "Russian");
40     string message = "С Новым Годом, друзья!";
41     string trMessage = tr(message);
42     string key = keygen(trMessage);
43     string encr = encryptdecrypt(trMessage, key);
44
45     cout << "Сообщение: " << message << "\n";
46     cout << "Преобразованное сообщение: ";
47     for (int i = 0; i < trMessage.size(); i++) {
48         cout << int(trMessage[i]);
49     }
50     cout << "\n" << "Ключ: " << key << "\n";
51     cout << "Зашифрованное сообщение: ";
52     for (int i = 0; i < encr.size(); i++) {
53         cout << int(encr[i]);
54     }
55     return 0;
56 }
```

Рис. 0.2: main функция программы-шифратора

Результат работы программы можно увидеть ниже:



```
Консоль отладки Microsoft Visual Studio
Сообщение: С Новым Годом, друзья!
Преобразованное сообщение: 813277110981231083267110100110108443210011211510312412733
Ключ: 1740948824551711527614
Зашифрованное сообщение: 9623121949179842411390819193271785696580747821
C:\Users\Admin\Desktop\lab\1\lab07\Debug\lab07.exe (процесс 9060) завершил работу с кодом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "Автоматически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно:
Press any key to close this window.
Press any key to continue.
<< "Сообщение: " << message << "\n":
```

Контрольные вопросы

1. Поясните смысл однократного гаммирования

Гаммирование - выполнение операции XOR между элементами гаммы и элементами подлежащего сокрытию текста. 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 всем скрываемом тексте.

2. Перечислите недостатки однократного гаммирования

Абсолютная стойкость шифра доказана только для случая, когда однократно используемый ключ, длиной, равной длине исходного сообщения, является фрагментом истинно случайной двоичной последовательности с равномерным законом распределения.

3. Перечислите преимущества однократного гаммирования

Во-первых, такой способ симметричен, т.е. двойное прибавление одной и той же величины по модулю 2 восстанавливает исходное значение. Во-вторых, шифрование и расшифрование может быть выполнено одной и той же программой. Наконец, криптоалгоритм не дает 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 .

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

Если ключ короче текста, то операция XOR будет применена не ко всем элементам и конец сообщения будет не закодирован. Если ключ будет длиннее, то появится

неоднозначность декодирования.

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

Наложение гаммы по сути представляет собой выполнения побитовой операции сложения по модулю 2, т.е. мы должны сложить каждый элемент гаммы с соответствующим элементом ключа. Данная операция является симметричной, так как прибавление одной и той же величины по модулю 2 восстанавливает исходное значение.

6. Как по открытому тексту и ключу получить шифротекст?

В таком случае задача сводится к правилу:

$$C_i = P_i (+) K_i$$

т.е. мы поэлементно получаем символы зашифрованного сообщения, применяя операцию исключающего или к соответствующим элементам ключа и открытого текста.

7. Как по открытому тексту и шифротексту получить ключ?

Подобная задача решается путем применения операции исключающего или к последовательности символов зашифрованного и открытого сообщений:

$$K_i = P_i (+) C_i$$

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;

Вывод

В ходе выполнения лабораторной работы было освоено на практике применение режима однократного гаммирования.