Информационная безопасность. Отчет по лабораторной работе №7

Элементы криптографии. Однократное гаммирование

Поляков Арсений Андреевич 1032192874

Содержание

Цель работы 1
Выполнение лабораторной работы 1
Выводы 4
Список литературы 4

Цель работы

Освоить на практике применение режима однократного гаммирования.

Выполнение лабораторной работы

Требуется разработать приложение позволяющие 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.

Для этого у меня есть функция позволяющая зашифровывать, расшифровывать данные с помощью сообщения и ключа. А также позволяющая получить ключ.

```cpp
vector<uint8_t> operate(vector<uint8_t> message, vector<uint8_t> key)
{
    if (message.size() != key.size()) {
        return {};
    }

    vector<uint8_t> temp;

    for (int i = 0; i < message.size(); i++) {
        temp.push_back(message[i] ^ key[i]);
    }
    return temp;
}
```

encrypt_fuction

Функция для вывода результатов

```cpp
void print_bytes(vector<uint8_t> message)
{
    for (const auto &e : message) {
        cout << hex << unsigned(e) << " ";
    }
    cout << endl;
}
```

output_prog

Биты сообщения и ключей.

```cpp
vector<uint8_t> key{0x05, 0x0C, 0x17, 0x7F, 0x0E, 0x4E, 0x37, 0xD2, 0x94, 0x10, 0x09, 0x2E, 0x22, 0x57, 0xFF, 0xC8, 0x0B, 0xB2, 0x70, 0x54};
vector<uint8_t> key2{0x05, 0x0C, 0x17, 0x7F, 0x0E, 0x4E, 0x37, 0xD2, 0x94, 0x10, 0x09, 0x2E, 0x22, 0x55, 0xF4, 0xD3, 0x07, 0xBB, 0xBC, 0x54};
vector<uint8_t> message{0xD8, 0xF2, 0xE8, 0xF0, 0xEB, 0xE8, 0xF6, 0x20, 0x2D, 0x20, 0xC2, 0xFB, 0x20, 0xC3, 0xE5, 0xF0, 0xEE, 0xE9, 0x21, 0x2
```

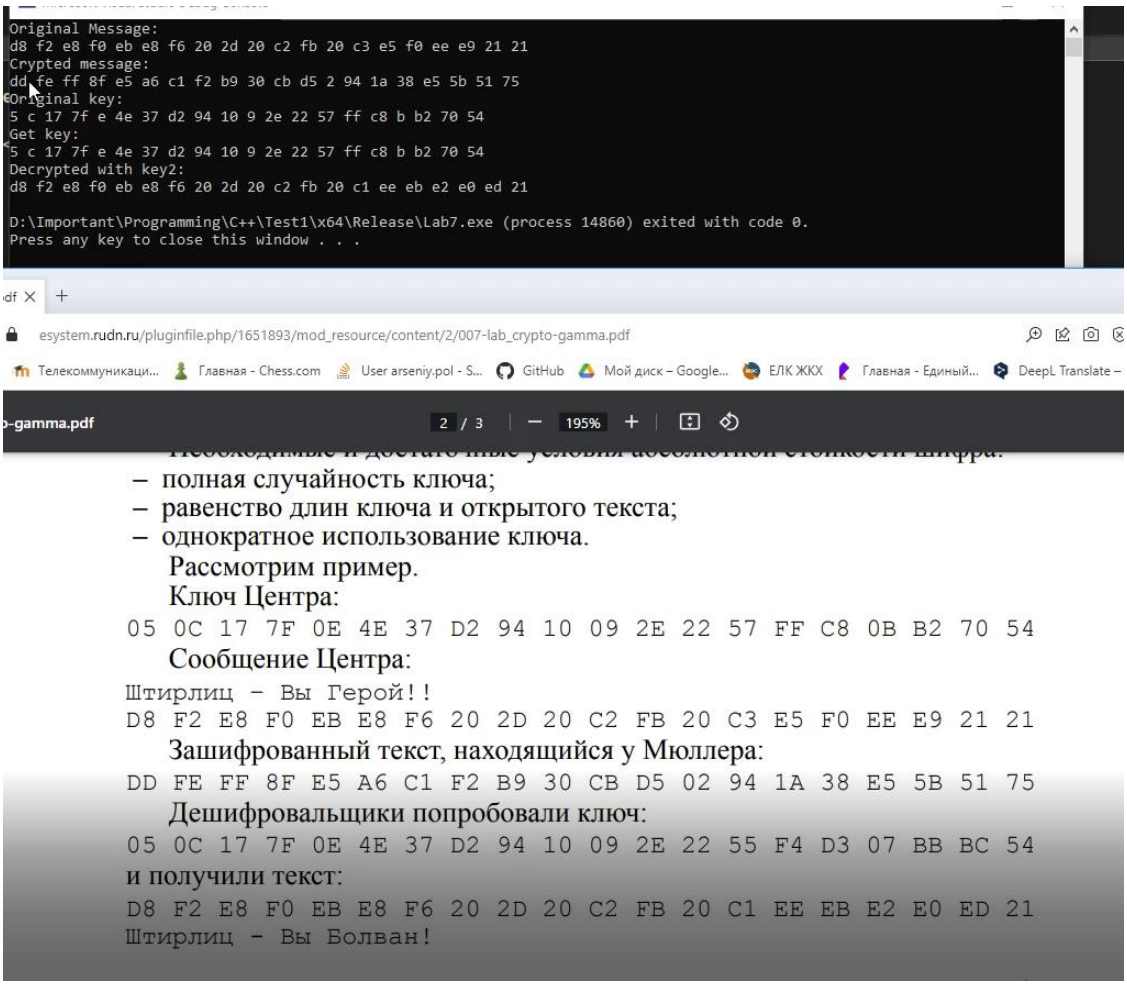
bytes

Главная функция

```
vector<uint8_t> crypt = operate(message, key);

cout << "Original Message: " << endl;
print_bytes(message);

cout << "Crypted message: " << endl;
print_bytes(crypt);

cout << "Original key: " << endl;
print_bytes(key);

cout << "Get key: " << endl;
print_bytes(operate(message, crypt));

cout << "Decrypted with key2: " << endl;
print_bytes(operate(crypt, key2));
```

Main

Затем я запускаю программу и сравниваю полученные результаты с тем, что должен был получить в методичке. Видно, что все ключи и закодированные и раскодированные сообщения сошлись



console_output

Выводы

В результате выполнения работы я освоил на практике применение режима однократного гаммирования.

Список литературы

1. Методические материалы курса