Лабораторная работа 7

Терентьев Егор Дмитриевич, НФИбд-01-19

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ПРЕЗЕНТАЦИЯ ПО ЛАБОРАТОРНОЙ РАБОТЕ №7

дисциплина: Информационная безопасность

Преподователь: Кулябов Дмитрий Сергеевич

Студент: Терентьев Егор Дмитриевич

Группа: НФИбд-01-19

МОСКВА

2022 г.

Прагматика выполнения лабораторной работы

Прагматика выполнения 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.

Цель работы

Цель работы

Освоить на практике применение режима однократного гаммирования.

Выполнение лабораторной работы

1. Создал функция позволяющая зашифровывать данные с помощью сообщения и ключа

1. Создал функция позволяющая зашифровывать данные с помощью сообщения и ключа

```
vector<uint8_t> encrypt(vector<uint8_t> message, vector<uint8_t> key)
{
    if (message.size() != key.size())
    {
        return {};
    }
    vector<uint8_t> encrypted;
    for (int i = 0; i < message.size(); i++)
    {
        encrypted.push_back(message[i] ^ key[i]);
    }
    return encrypted;
}
```

Figure 1: encrypt

2. Создал функцию для того, чтобы расщифровывать сообщения с помощью сообщения и ключа

2. Создал функцию для того, чтобы расшифровывать сообщения с помощью сообщения и ключа

```cpp
vector<uint8_t> decrypt(vector<uint8_t> message, vector<uint8_t> key)
{
    if (message.size() != key.size())
    {
        return {};
    }
    vector<uint8_t> decrypted;
    for (int i = 0; i < message.size(); i++)
    {
        decrypted.push_back(message[i] ^ key[i]);
    }
    return decrypted;
}
```

Figure 2: decrypt

3. Создал функцию получения ключа

3. Создал функцию получения ключа

```
vector<uint8_t> get_key(vector<uint8_t> message, vector<uint8_t> crypt)
{
    if (message.size() != crypt.size())
    {
        return {};
    }
    vector<uint8_t> key;
    for (int i = 0; i < message.size(); i++)
    {
        key.push_back(message[i] ^ crypt[i]);
    }
    return key;
}
```

Figure 3: get_key

4. Остальное в программе отвечает за вывод полученных результатов

4. Остальное в программе отвечает за вывод полученных результатов

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

int main()
{
    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, 0x21 };

    vector<uint8_t> crypt = encrypt(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(get_key(message, crypt));
    cout << "Decrypted with key2: " << endl;
    print_bytes(decrypt(crypt, key2));
    return 0;
}
```

Figure 4: output_in_prog

5. Запуск программы.

5. Запуск программы.

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

```
Original Message:
d8f2e8f0ebe8f6202d20c2fb20c3e5f0eee92121
Crypted message:
ddfeff8fe5a6c1f2b930cbd52941a38e55b5175
Original key:
5c177fe4e37d2941092e2257ffc8bb27054
Get key:
5c177fe4e37d2941092e2257ffc8bb27054
Decrypted with key2:
d8f2e8f0ebe8f6202d20c2fb20c1eeebe2e0ed21
```

Figure 5: output_console

Выводы

Выводы

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