# Лабораторная работа 8

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 №8

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

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.

# Цель работы

# Цель работы

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

# 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. Создал функцию получения расшифрованного сообщения без ключа

# 4. Создал функцию получения расшифрованного сообщения без ключа

```
vector<uint8_t> get_message_with_three_pieces(vector<uint8_t> cr1, vector<uint8_t> cr2, vector<uint8_t> msg1)
{
    if (cr1.size() != cr2.size() and cr1.size() != msg1.size())
    {
        return {};
    }
    vector<uint8_t> msg2;
    for (int i = 0; i < cr1.size(); i++)
    {
        msg2.push_back(cr1[i] ^ cr2[i] ^ msg1[i]);
    }
    return msg2;
}
```

Figure 4: decrypt_without_key

# 5. Остальное в программе отвечает за вывод полученных результатов

# 5. 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;
}

void print_text(vector<uint8_t> message)
{
    string str(message.begin(), message.end());
    cout << str << endl;
}


int main()
{
    string message1 = "hello this is lab 8";
    string message2 = "this lab 8 ab hello";
    vector<uint8_t> first(message1.begin(), message1.end());
    vector<uint8_t> second(message2.begin(), message2.end());

    string keystr = "thisiskeystringlab7";
    vector<uint8_t> key(keystr.begin(), keystr.end());

    vector<uint8_t> crypt1 = encrypt(first, key);
    vector<uint8_t> crypt2 = encrypt(second, key);


    cout << "Original Message number 1: " << endl;
    print_text(first);
    cout << endl << "Original Message number 2: " << endl;
    print_text(second);
    cout << endl << "Crypted message number 1: " << endl;
    print_bytes(crypt1);
    cout << endl << "Crypted message number 2: " << endl;
    print_bytes(crypt2);


    cout << endl <<"Finding message 2:" << endl;
    vector<uint8_t> msg_found = get_message_with_three_pieces(crypt1, crypt2, first);
    print_text(msg_found);
    return 0;
}
```

Figure 5: output_in_prog

# 6. Запуск программы.

# 6. Запуск программы.

Получаю вывод программы, где мы видим, что мы смогли расшифровать сообщение без знаний ключа.

```
Original Message number 1:
hello this is lab 8

Original Message number 2:
this lab 8 ab hello

Crypted message number 1:
1cd51f6531fd100541b1a4ebd342f

Crypted message number 2:
0000491fa7594b5413b4ef9de58

Finding message 2:
this lab 8 ab hello
```

Figure 6: output_console

# Выводы

# Выводы

В результате выполнения работы я освоил на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.