Лабораторная работа 7

Попов Дмитрий Павлович, 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: Попов Дмитрий Павлович

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 функцию позволяющую зашифровывать, расшифровывать данные с помощью сообщения и ключа. А также позволяющую получить ключ.

1. Создал функцию позволяющую зашифровывать, расшифровывать данные с помощью сообщения и ключа. А также позволяющую получить ключ.

```
7   vector<uint8_t> encrypt(vector<uint8_t> message, vector<uint8_t> key)
8   {
9       if (message.size() != key.size())
10      {
11          return {};
12      }
13      vector<uint8_t> encrypted;
14      for (int i = 0; i < message.size(); i++)
15      {
16          encrypted.push_back(message[i] ^ key[i]);
17      }
18      return encrypted;
19  }
```

Figure 1: encrypt_fuction

2. Создал функцию для вывода результатов

2. Создал функцию для вывода результатов

```
49  void print_bytes(vector<uint8_t> message)
50  {
51      for (const auto &e : message)
52      {
53          cout << hex << unsigned(e) << " ";
54      }
55      cout << endl;
56  }
```

Figure 2: output_prog

3. Определил биты ключей и сообщения

3. Определил биты ключей и сообщения

```
vector<uint8_t> key{0x05, 0x0C, 0x17, 0x7F, 0x0E, 0x4E, 0x37, 0xD2, 0x94, 0x10, 0x09, 0x2E, 0x22, 0x57, 0xFF,
vector<uint8_t> key2{0x05, 0x0C, 0x17, 0x7F, 0x0E, 0x4E, 0x37, 0xD2, 0x94, 0x10, 0x09, 0x2E, 0x22, 0x55, 0xF
vector<uint8_t> message{0xD8, 0xF2, 0xE8, 0xF0, 0xEB, 0xE8, 0xF6, 0x20, 0x2D, 0x20, 0xC2, 0xFB, 0x20, 0xC3, (
```

Figure 3: bytes

4. Определил главную функцию

4. Определил главную функцию

```
58  int main()
59  {
60      vector<uint8_t> key{0x05, 0x0C, 0x17, 0x7F, 0x0
61      vector<uint8_t> key2{0x05, 0x0C, 0x17, 0x7F, 0x
62      vector<uint8_t> message{0xD8, 0xF2, 0xE8, 0xF0,
63
64      vector<uint8_t> crypt = encrypt(message, key);
65      cout << "Original Message: " << endl;
66      print_bytes(message);
67      cout << "Crypted message: " << endl;
68      print_bytes(crypt);
69      cout << "Original key: " << endl;
70      print_bytes(key);
71      cout << "Get key: " << endl;
72      print_bytes(get_key(message, crypt));
73      cout << "Decrypted with key2: " << endl;
74      print_bytes(decrypt(crypt, key2));
75      return 0;
76  }
```

Figure 4: Main

5. Запуск программы.

5. Запуск программы.

```
[dpopov@dpopov lab7]$ g++ lab7.cpp
[dpopov@dpopov lab7]$ ls
a.out  lab7.cpp
[dpopov@dpopov lab7]$ ./a.out
Original Message:
d8 f2 e8 f0 eb e8 f6 20 2d 20 c2 fb 20 c3 e5 f0 ee e9 21 21
Crypted message:
dd fe ff 8f e5 a6 c1 f2 b9 30 cb d5 2 94 1a 38 e5 5b 51 75
Original key:
5 c 17 7f e 4e 37 d2 94 10 9 2e 22 57 ff c8 b b2 70 54
Get key:
5 c 17 7f e 4e 37 d2 94 10 9 2e 22 57 ff c8 b b2 70 54
Decrypted with key2:
d8 f2 e8 f0 eb e8 f6 20 2d 20 c2 fb 20 c1 ee eb e2 e0 ed 21
[dpopov@dpopov lab7]$
```

Figure 5: output_console

Вывод

Вывод

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