Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОССУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОННИКИ

КАФЕДРА ИНФОРМАТИКИ

Отчёт по лабораторной работе №1

По дисциплине «Методы защиты информации»

По теме «Симметричная криптография. Тройной DES»

Выполнил:

студент гр. 653502

Максименко П.В.

Проверил:

Артемьев В. С.

Минск 2019

1. Постановка зада4и

Написать программу, реализующую алгоритм шифрования информации при помощи тройного DES шифрования.

2. Теорети4еская 4асть

**DES** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Data Encryption Standard*) — алгоритм для [симметричного](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BC%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%87%D0%BD%D1%8B%D0%B9_%D1%88%D0%B8%D1%84%D1%80) [шифрования](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), разработанный фирмой [IBM](https://ru.wikipedia.org/wiki/IBM) и утверждённый правительством [США](https://ru.wikipedia.org/wiki/%D0%A1%D0%A8%D0%90) в 1977 году как официальный стандарт ([FIPS](https://ru.wikipedia.org/wiki/FIPS) 46-3). Размер блока для DES равен 64 [битам](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D1%82). В основе алгоритма лежит [сеть Фейстеля](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D1%8C_%D0%A4%D0%B5%D0%B9%D1%81%D1%82%D0%B5%D0%BB%D1%8F) с 16 циклами ([раундами](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%83%D0%BD%D0%B4_(%D0%B2_%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D0%B8))) и [ключом](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D1%8E%D1%87_(%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F)), имеющим длину 56 [бит](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D1%82). Алгоритм использует комбинацию нелинейных (S-блоки) и линейных (перестановки E, IP, IP-1) преобразований.

Прямым развитием DES в настоящее время является алгоритм [Triple DES](https://ru.wikipedia.org/wiki/3DES) (3DES). В 3DES шифрование/расшифровка выполняются путём троекратного выполнения алгоритма DES.

3. Блок-схема

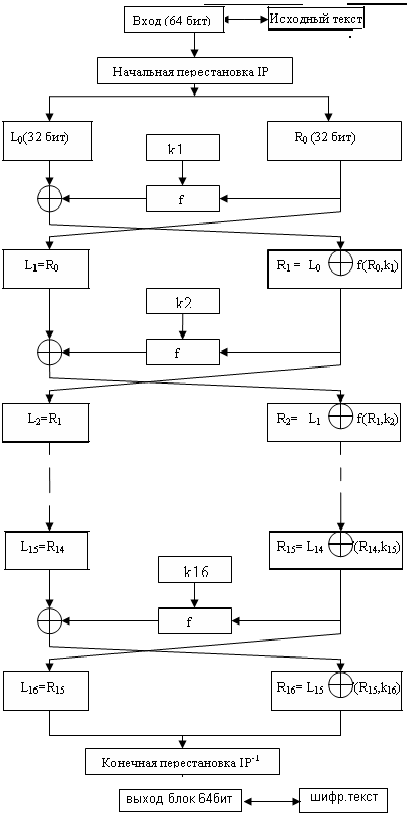


Рис 3.1. Блок-схема работы программы.

4. Ход программы



Рис 4.1. Данные, поступающие на вход



Рис 4.2. Данные после IP перестановки



Рис 4.3. Полу4ение полуклю4ей

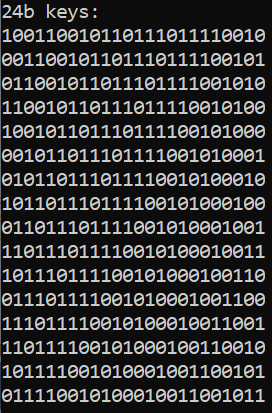


Рис 4.4. Генерирование 24-битных клю4ей



Рис 4.5. Данные после обработки

После этого алгоритм повторяется ещё 2 раза с новыми клю4ами:

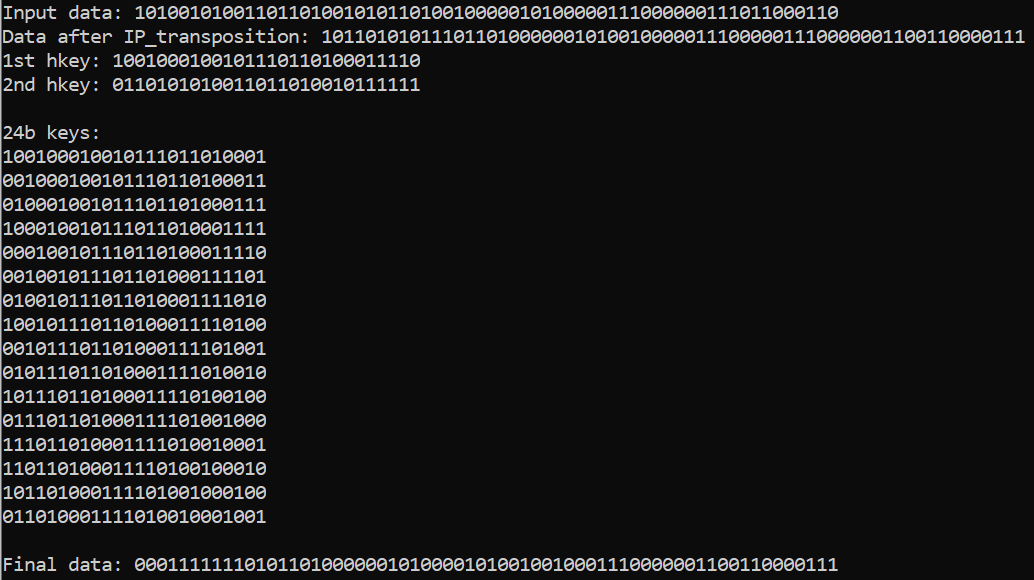


Рис 4.6. Второй цикл обработки

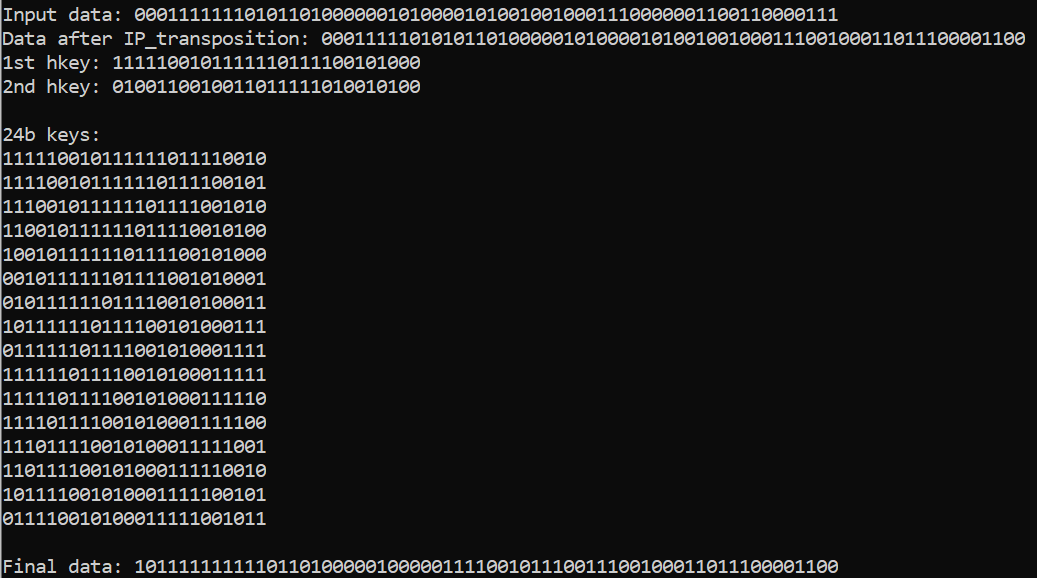


Рис 4.7. Третий цикл обработки

Результат работы последнего круга алгоритма является результатом работы всего 3DES алгоритма.

5. Код программы

|  |  |
| --- | --- |
|  | #include <iostream> |
|  | #include <fstream> |
|  | #include <string> |
|  |  |
|  | using namespace std; |
|  |  |
|  | string read\_input\_file() |
|  | { |
|  | std::string line; |
|  |  |
|  | std::ifstream in("C:\\Users\\Picasso\\Desktop\\Старая папка\\sem7labs\\msi\\input.txt"); |
|  | if (in.is\_open()) |
|  | { |
|  | while (getline(in, line)) |
|  | { |
|  | std::cout << line << std::endl; |
|  | } |
|  | } |
|  | in.close(); |
|  |  |
|  | return line; |
|  | } |
|  |  |
|  |  |
|  | void write\_into\_file(string content, int type\_of\_write) |
|  | { |
|  | std::ofstream out; |
|  | // type 0 - encrypt |
|  | if (type\_of\_write == 0) { |
|  | out.open("C:\\Users\\Picasso\\Desktop\\Старая папка\\sem7labs\\msi\\encrypt.txt"); |
|  | } |
|  | // type 1 - decrypt |
|  | else if (type\_of\_write == 1) { |
|  | out.open("C:\\Users\\Picasso\\Desktop\\Старая папка\\sem7labs\\msi\\decrypt.txt"); |
|  | } |
|  |  |
|  | /\* |
|  | if (out.is\_open()) |
|  | { |
|  | out << content << std::endl; |
|  | } |
|  | \*/ |
|  | } |
|  |  |
|  |  |
|  | string key\_xor(string str, string key, int mod) |
|  | { |
|  | string final\_str = str; |
|  |  |
|  | for (int i = 0; i < key.length(); i++) { |
|  | if (final\_str[i + mod] == '0' && key[i] == '0') { |
|  | final\_str[i + mod] = '0'; |
|  | } |
|  | else if (final\_str[i + mod] == '0' && key[i] == '1') { |
|  | final\_str[i + mod] = '1'; |
|  | } |
|  | else if (final\_str[i + mod] == '1' && key[i] == '0') { |
|  | final\_str[i + mod] = '1'; |
|  | } |
|  | else if (final\_str[i + mod] == '1' && key[i] == '1') { |
|  | final\_str[i + mod] = '0'; |
|  | } |
|  | } |
|  |  |
|  | return final\_str; |
|  | } |
|  |  |
|  |  |
|  | std::string DES(string inp\_data, string inp\_key) |
|  | { |
|  | string data\_help = inp\_data; |
|  | std::cout << '\n'; |
|  | std::cout << data\_help; |
|  |  |
|  | // IP transposition |
|  | int IP\_tr[64] = {0,1,2,23,4,5,6,7,47,61,10,11,12,13,14,15,16,17,18,20,36,22,3,24,25,26,27,28,29,30,31,32,33,34,35,21,37,38,39,40,41,42,43,44,45,46,8,48,49,50,51,52,53,63,55,56,57,58,59,60,9,62,54}; |
|  | string data = data\_help; |
|  | for (int i = 0; i < 64; i++) { |
|  | data[i] = data\_help[IP\_tr[i]]; |
|  | } |
|  | std::cout << '\n'; |
|  | std::cout << data; |
|  |  |
|  | // 56bit key |
|  | string key = inp\_key; |
|  |  |
|  | // halve key |
|  | string hkey[2]; |
|  | int key\_length = key.length(); |
|  | for (int i = 0; i < key\_length/2; i++) { |
|  | hkey[0] = hkey[0] + key[i]; |
|  | } |
|  | for (int i = key\_length/2; i < key\_length; i++) { |
|  | hkey[1] = hkey[1] + key[i]; |
|  | } |
|  | std::cout << '\n'; |
|  | std::cout << hkey[0]; |
|  | std::cout << '\n'; |
|  | std::cout << hkey[1]; |
|  | std::cout << '\n'; |
|  |  |
|  | // shifts and forming 16 24b keys |
|  | string t1key[16], t2key[16]; |
|  | for (int i = 0; i < 16; i++) { |
|  | for (int j = 0; j < 28; j++) { |
|  | if (t1key[i].length() < 24) { |
|  | if (i + j < 28) { |
|  | t1key[i] = t1key[i] + hkey[0][i + j]; |
|  | t2key[i] = t2key[i] + hkey[1][i + j]; |
|  | } |
|  | else { |
|  | t1key[i] = t1key[i] + hkey[0][i + j - 28]; |
|  | t2key[i] = t2key[i] + hkey[1][i + j - 28]; |
|  | } |
|  | } |
|  | } |
|  | } |
|  | std::cout << '\n'; |
|  | for (int i = 0; i < 16; i++) { |
|  | std::cout << t1key[i]; |
|  | std::cout << '\n'; |
|  | } |
|  |  |
|  | // xor 1st part of keys and data |
|  | for (int i = 0; i < 16; i++) { |
|  | data = key\_xor(data, t1key[i], i); |
|  | } |
|  |  |
|  | // xor 2nd part of keys and data |
|  | for (int i = 0; i < 16; i++) { |
|  | data = key\_xor(data, t2key[i], i); |
|  | } |
|  |  |
|  | std::cout << '\n'; |
|  | std::cout << data; |
|  |  |
|  | return data; |
|  | } |
|  |  |
|  |  |
|  | int main() |
|  | { |
|  | SetConsoleCP(1251); |
|  | SetConsoleOutputCP(1251); |
|  |  |
|  | string wdata; |
|  | string rdata; |
|  | rdata = read\_input\_file(); |
|  |  |
|  | string key1 = "10011001011011101111001010000111111010011011111010000110"; |
|  | string key2 = "10010001001011101101000111100110101010011011010010111111"; |
|  | string key3 = "11111001011111101111001010000100110010011011111010010100"; |
|  |  |
|  | wdata = DES(rdata, key1); |
|  | wdata = DES(wdata, key2); |
|  | wdata = DES(wdata, key3); |
|  |  |
|  | std::cout << '\n'; |
|  |  |
|  | return 0; |
|  | } |

6. Вывод

В ходе выполнения лабораторной работы был успешно реализован алгоритм тройного DES шифрования при мощи языка программирования С++. Для улу4шения криптозащиты клю4и для каждого прохода алгоритма независимы друг от друга.