Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

|  |
| --- |
| Институт космических и информационных технологий |
| *институт* |
| Кафедра Прикладной математики и компьютерной безопасности |
| *кафедра* |

**ЛАБОРАТОРНАЯ РАБОТА №4**

|  |
| --- |
| Шифр Шамира |
| *тема* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Руководитель | | |  |  |  | В.И.Вайнштейн |
|  | |  |  | *подпись, дата* |  | *инициалы, фамилия* |
| Студент | КИ15-01 №031508683 | |  |  |  | М.С.Димаксян |
|  | *номер группы, зачетной книжки* | |  | *подпись, дата* |  | *инициалы, фамилия* |

Красноярск 2019

**Задание:**

Изучить основные принципы и процедурные аспекты алгоритма Шамира для передачи зашифрованного сообщения.

**Теория:**

Теоретические сведения Этот шифр, предложенный Шамиром (Adi Shamir), был первым, позволяющим организовать обмен секретными сообщениями по открытой линии связи для лиц, которые не имеют никаких защищенных каналов и секретных ключей и, возможно, никогда не видели друг друга.

Пусть есть два абонента А и В, соединенные линией связи. А хочет передать сообщение т абоненту В так, чтобы никто не узнал его содержание. А выбирает случайное большое простое число р и открыто передает его В. Затем А выбирает два числа и , такие, что . Эти числа А держит в секрете и передавать не будет. В тоже выбирает два числа и , такие, что , и держит их в секрете.

Шаг 1. А вычисляет , где — исходное сообщение, и пересылает к В.

Шаг 2. В, получив , вычисляет и передает к А.

Шаг 3. А вычисляет и передает его В.

Шаг 4. В, получив , вычисляет .

**Исходный код:**

**Функции теории чисел:**

public static class NumberTheory

{

public static BigInteger BinaryModPow(BigInteger Number, BigInteger Deg, BigInteger Mod)

{

BigInteger Result = 1;

BigInteger Bit = Number;

if (Bit < 0) Bit = (Number + Mod) % Mod;

while (Deg > 0)

{

if ((Deg & 1) == 1)

{

Result \*= Bit;

Result %= Mod;

}

Deg >>= 1;

if (Deg > 0)

{

Bit \*= Bit;

Bit %= Mod;

}

}

return Result;

}

public static BigInteger GCD(BigInteger a, BigInteger b, out BigInteger x, out BigInteger y)

{

if (a == 0)

{

x = 0;

y = 1;

return b;

}

BigInteger x1, y1;

BigInteger d = GCD(b % a, a, out x1, out y1);

x = y1 - (b / a) \* x1;

y = x1;

if (d < 0)

{

d = -d;

x = -x;

y = -y;

}

return d;

}

public static BigInteger Foo(BigInteger element, BigInteger Mod)

{

if (element < 0) element = (element + Mod) % Mod;

BigInteger x, y;

BigInteger g = GCD(element, Mod, out x, out y);

//Обратный элемент не существует, если кольцо не является полем (модуль не является степенью простого числа) и элемент не взаимно прост с модулем

//Возвращает 0 в случае, если обратный элемент не существует

if (g != 1) x = 0;

//Приведем результат в положительный вид по модулю

x = x % Mod;

if (x < 0) x = (x + Mod) % Mod;

return x;

}

}

**Тест Миллера-Рабина:**

public static class RabinMiller

{

// Random generator (thread safe)

private static ThreadLocal<Random> s\_Gen = new ThreadLocal<Random>(

() =>

{

return new Random();

}

);

// Random generator (thread safe)

private static Random Gen

{

get

{

return s\_Gen.Value;

}

}

public static Boolean IsPrime(BigInteger value, int witnesses)

{

if (value < 5)

return (value == 2 || value == 3);

if (witnesses <= 0)

witnesses = 1;

BigInteger d = value - 1;

int s = 0;

while (d % 2 == 0)

{

d /= 2;

s += 1;

}

Byte[] bytes = new Byte[value.ToByteArray().LongLength];

BigInteger a;

for (int i = 0; i < witnesses; i++)

{

do

{

Gen.NextBytes(bytes);

a = new BigInteger(bytes);

}

while (a < 2 || a >= value - 2);

BigInteger x = BigInteger.ModPow(a, d, value);

if (x == 1 || x == value - 1)

continue;

for (int r = 1; r < s; r++)

{

x = BigInteger.ModPow(x, 2, value);

if (x == 1)

return false;

if (x == value - 1)

break;

}

if (x != value - 1)

return false;

}

return true;

}

}

**Функции интерфейса:**

private void f5\_textParam\_KeyPress(object sender, KeyPressEventArgs e)

{

if (MyAlphabets.Numeric.IndexOf(e.KeyChar) == -1 && e.KeyChar != (char)8) e.Handled = true;

}

private void f4\_textP\_Validated(object sender, EventArgs e)

{

f4\_textP.Text = new string(f4\_textP.Text.Where(t => char.IsDigit(t)).ToArray());

if (f4\_textP.TextLength > 0)

{

BigInteger P = BigInteger.Parse(f4\_textP.Text);

if (P % 2 == 0)

{

f4\_textP.Text = Convert.ToString(++P);

}

}

}

private void f4\_button\_ChangeCa\_Click(object sender, EventArgs e)

{

f4\_textDa.Text = "";

f4\_textCa.ReadOnly = false;

f4\_button\_GetRandomCa.Enabled = true;

f4\_button\_SetCa.Enabled = true;

f4\_button\_ChangeCa.Enabled = false;

}

private void f4\_button\_ChangeCb\_Click(object sender, EventArgs e)

{

f4\_textDb.Text = "";

f4\_textCb.ReadOnly = false;

f4\_button\_GetRandomCb.Enabled = true;

f4\_button\_SetCb.Enabled = true;

f4\_button\_ChangeCb.Enabled = false;

}

**Функция поиска следующего простого p:**

private void f5\_buttonNextPrime\_Click(object sender, EventArgs e)

{

f4\_textP.Text = new string(f4\_textP.Text.Where(t => char.IsDigit(t)).ToArray());

BigInteger ParamP;

if (f4\_textP.TextLength > 0)

ParamP = BigInteger.Parse(f4\_textP.Text);

else

ParamP = 0;

if (++ParamP % 2 == 0) ParamP++;

while (!RabinMiller.IsPrime(ParamP, 3))

ParamP += 2;

f4\_textP.Text = Convert.ToString(ParamP);

}

**Установка параметров системы:**

private void f4\_buttonSetKeys\_Click(object sender, EventArgs e)

{

f4\_textP.Text = new string(f4\_textP.Text.Where(t => char.IsDigit(t)).ToArray());

if (f4\_textP.TextLength > 0)

{

if (RabinMiller.IsPrime(BigInteger.Parse(f4\_textP.Text), 10))

{

f4\_textP.ReadOnly = true;

f4\_buttonNextPrime.Enabled = false;

f4\_buttonSetKeys.Enabled = false;

f4\_buttonClearKeys.Enabled = true;

f4\_textCa.ReadOnly = false;

f4\_textCb.ReadOnly = false;

f4\_textBox\_MessageAlice.ReadOnly = false;

f4\_button\_GetRandomCa.Enabled = true;

f4\_button\_GetRandomCb.Enabled = true;

f4\_button\_SetCa.Enabled = true;

f4\_button\_SetCb.Enabled = true;

}

}

}

**Сброс параметров системы:**

private void f4\_buttonClearKeys\_Click(object sender, EventArgs e)

{

f4\_textP.ReadOnly = false;

f4\_buttonNextPrime.Enabled = true;

f4\_buttonSetKeys.Enabled = true;

f4\_buttonClearKeys.Enabled = false;

f4\_textCa.ReadOnly = true;

f4\_textCb.ReadOnly = true;

f4\_textCa.Text = "";

f4\_textCb.Text = "";

f4\_textDa.Text = "";

f4\_textDb.Text = "";

f4\_textBox\_MessageAlice.Text = "";

f4\_textBox\_MessageBob.Text = "";

f4\_textBox\_MessageAlice.ReadOnly = true;

f4\_button\_GetRandomCa.Enabled = false;

f4\_button\_GetRandomCb.Enabled = false;

f4\_button\_SetCa.Enabled = false;

f4\_button\_SetCb.Enabled = false;

f5\_buttonSendMes.Enabled = false;

f4\_button\_ChangeCa.Enabled = false;

f4\_button\_ChangeCa.Enabled = false;

}

**Генерация :**

private void f4\_button\_GetRandomCa\_Click(object sender, EventArgs e)

{

BigInteger P = BigInteger.Parse(f4\_textP.Text);

RandomNumberGenerator rng = RandomNumberGenerator.Create();

BigInteger Da, Ca = BigIntegerRandomUtils.RandomInRange(rng, 2, P - 1);

while (!(NumberTheory.GCD(Ca, P - 1, out BigInteger x, out BigInteger y) == 1))

Ca = BigIntegerRandomUtils.RandomInRange(rng, 2, P - 1); ;

f4\_textCa.Text = Convert.ToString(Ca);

}

**Установка с расчетом :**

private void f4\_button\_SetCa\_Click(object sender, EventArgs e)

{

f4\_textCa.Text = new string(f4\_textCa.Text.Where(t => char.IsDigit(t)).ToArray());

if (f4\_textCa.TextLength > 0)

{

BigInteger P = BigInteger.Parse(f4\_textP.Text);

BigInteger Ca = BigInteger.Parse(f4\_textCa.Text);

if ((NumberTheory.GCD(Ca, P - 1, out BigInteger x, out BigInteger y) == 1))

{

f4\_textDa.Text = Convert.ToString(NumberTheory.Foo(Ca, P - 1));

f4\_textCa.ReadOnly = true;

f4\_button\_GetRandomCa.Enabled = false;

f4\_button\_SetCa.Enabled = false;

f4\_button\_ChangeCa.Enabled = true;

}

}

}

**Блокировка отправки сообщений в случае сброса ключей:**

private void f4\_textDa\_OR\_Db\_TextChanged(object sender, EventArgs e)

{

if (f4\_textDa.TextLength > 0 && f4\_textDb.TextLength > 0)

{

f5\_buttonSendMes.Enabled = true;

}

else

{

f5\_buttonSendMes.Enabled = false;

}

}

**Старст отправки сообщения:**

private void f5\_buttonSendMes\_Click(object sender, EventArgs e)

{

//Очистка строки от мусора

string tmp = "";

for (int i = 0; i < f4\_textBox\_MessageAlice.TextLength; i++) { if (MyAlphabets.AllTypesTogether.IndexOf(f4\_textBox\_MessageAlice.Text[i]) != -1) tmp += f4\_textBox\_MessageAlice.Text[i]; }

f4\_textBox\_MessageAlice.Text = "";

f4\_textBox\_MessageAlice.Text = tmp;

f4\_textStepA0.Text = "";

if (f4\_textBox\_MessageAlice.TextLength > 0)

{

//Шифрование

BigInteger m = MyAlphabets.AllTypesTogether.IndexOf(f4\_textBox\_MessageAlice.Text[0]); //Сообщение

f4\_textStepA0.Text = Convert.ToString(m);

f4\_textBox\_MessageAlice.Text = f4\_textBox\_MessageAlice.Text.Substring(1);//Удаление зашифрованного символа из строки

}

}

**Шаг 1:**

private void f4\_textStepA0\_TextChanged(object sender, EventArgs e)

{

f4\_textStepA1.Text = "";

f4\_textStepB1.Text = "";

if (f4\_textStepA0.TextLength > 0)

{

BigInteger m = BigInteger.Parse(f4\_textStepA0.Text);

BigInteger p = BigInteger.Parse(f4\_textP.Text);

BigInteger Ca = BigInteger.Parse(f4\_textCa.Text);

//Шаг 1 шифрования

BigInteger x1 = NumberTheory.BinaryModPow(m, Ca, p);

f4\_textStepA1.Text = Convert.ToString(x1);

f4\_textStepB1.Text = Convert.ToString(x1);

}

}

**Шаг 2:**

private void f4\_textStepB1\_TextChanged(object sender, EventArgs e)

{

f4\_textStepA2.Text = "";

f4\_textStepB2.Text = "";

if (f4\_textStepB1.TextLength > 0)

{

BigInteger x1 = BigInteger.Parse(f4\_textStepB1.Text);

BigInteger p = BigInteger.Parse(f4\_textP.Text);

BigInteger Cb = BigInteger.Parse(f4\_textCb.Text);

//Шаг 2 шифрования

BigInteger x2 = NumberTheory.BinaryModPow(x1, Cb, p);

f4\_textStepA2.Text = Convert.ToString(x2);

f4\_textStepB2.Text = Convert.ToString(x2);

}

}

**Шаг 3:**

private void f4\_textStepA2\_TextChanged(object sender, EventArgs e)

{

f4\_textStepA3.Text = "";

f4\_textStepB3.Text = "";

if (f4\_textStepA2.TextLength > 0)

{

BigInteger x2 = BigInteger.Parse(f4\_textStepA2.Text);

BigInteger p = BigInteger.Parse(f4\_textP.Text);

BigInteger Da = BigInteger.Parse(f4\_textDa.Text);

//Шаг 3 шифрования

BigInteger x3 = NumberTheory.BinaryModPow(x2, Da, p);

f4\_textStepA3.Text = Convert.ToString(x3);

f4\_textStepB3.Text = Convert.ToString(x3);

}

}

**Шаг 4:**

private void f4\_textStepB3\_TextChanged(object sender, EventArgs e)

{

f4\_textStepB4.Text = "";

if (f4\_textStepB3.TextLength > 0)

{

BigInteger x3 = BigInteger.Parse(f4\_textStepB3.Text);

BigInteger p = BigInteger.Parse(f4\_textP.Text);

BigInteger Db = BigInteger.Parse(f4\_textDb.Text);

//Шаг 4 шифрования

BigInteger x4 = NumberTheory.BinaryModPow(x3, Db, p);

f4\_textStepB4.Text = Convert.ToString(x4);

}

}

**Вывод полученного сообщения:**

private void f4\_textStepB4\_TextChanged(object sender, EventArgs e)

{

if (f4\_textStepB4.TextLength > 0)

{

BigInteger x4 = BigInteger.Parse(f4\_textStepB4.Text);

int letterNum = (int)x4;

f4\_textBox\_MessageBob.Text += MyAlphabets.AllTypesTogether[letterNum];

}

}

**Результат работы программы:**

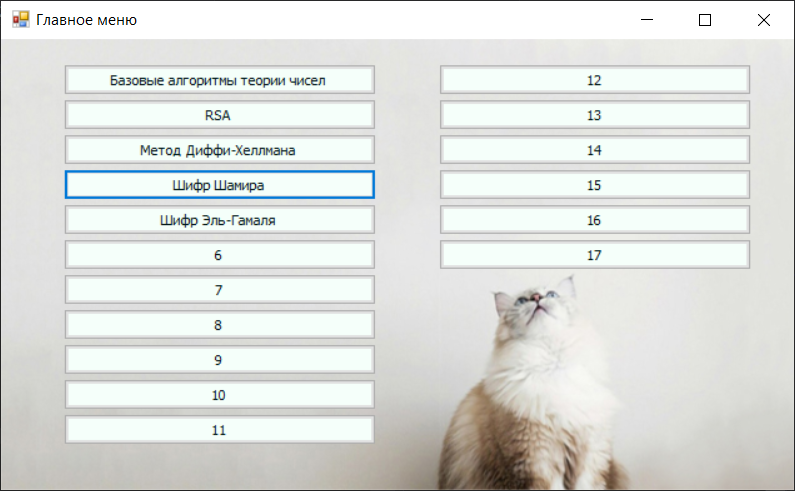


Рисунок 1. Главное меню программы.

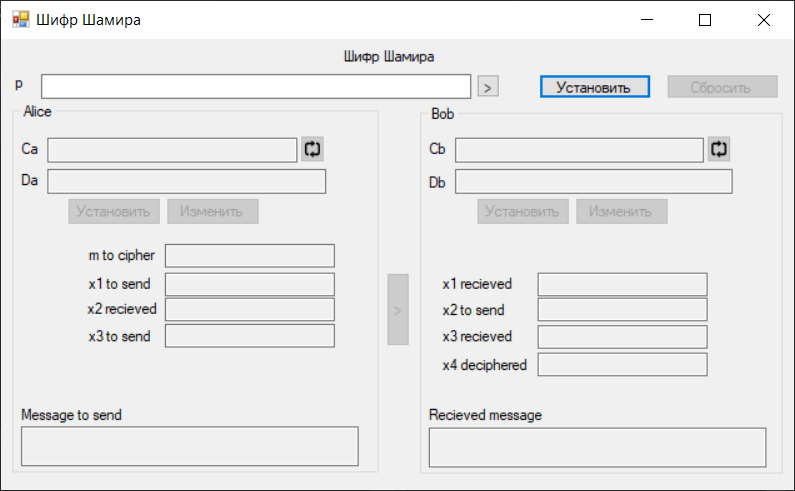


Рисунок 2. Окно шифра Шамира.

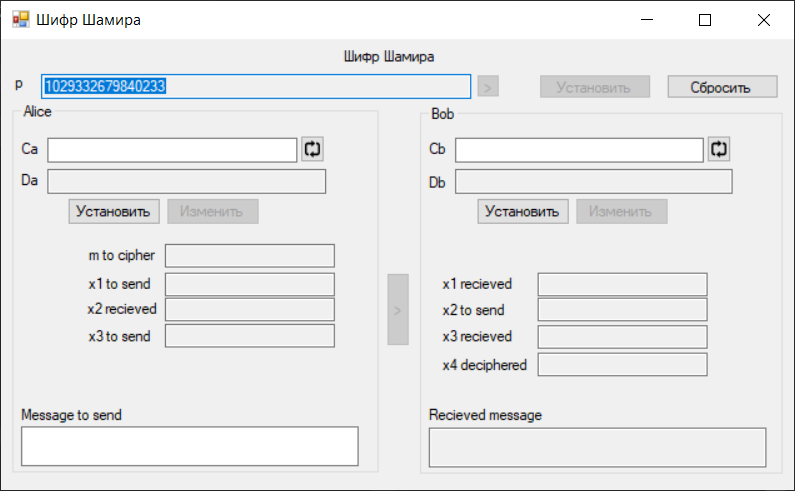


Рисунок 3. Установка параметров.

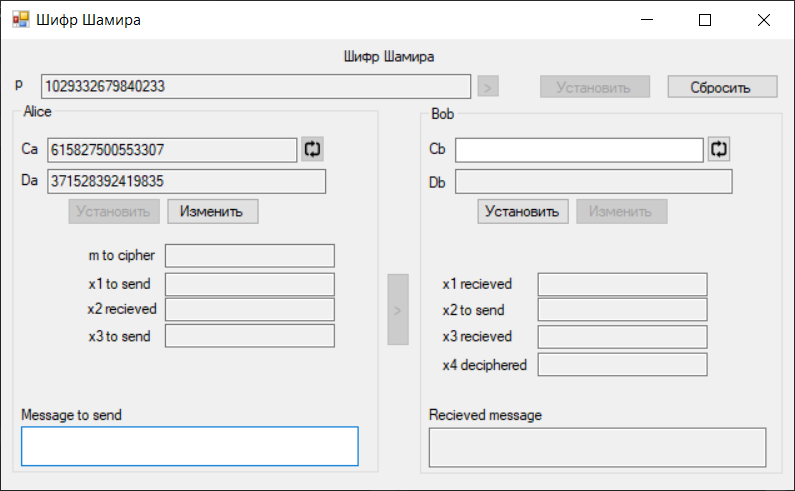


Рисунок 4. Установка ключей Алисы.

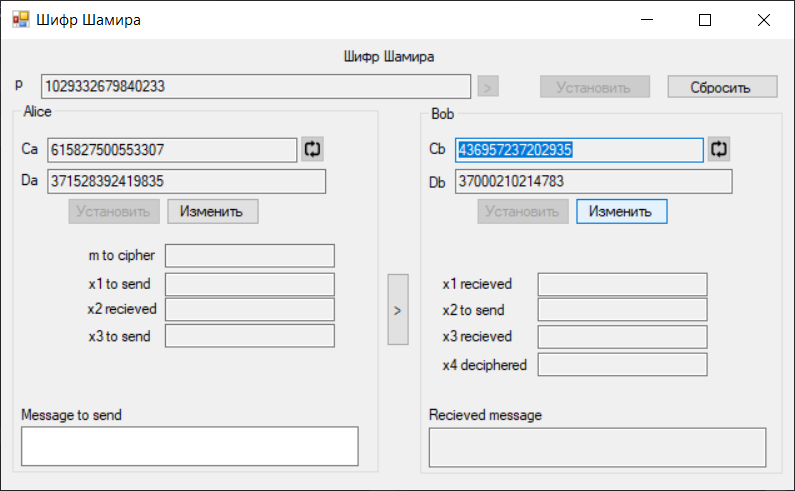


Рисунок 5. Установка ключей Боба.

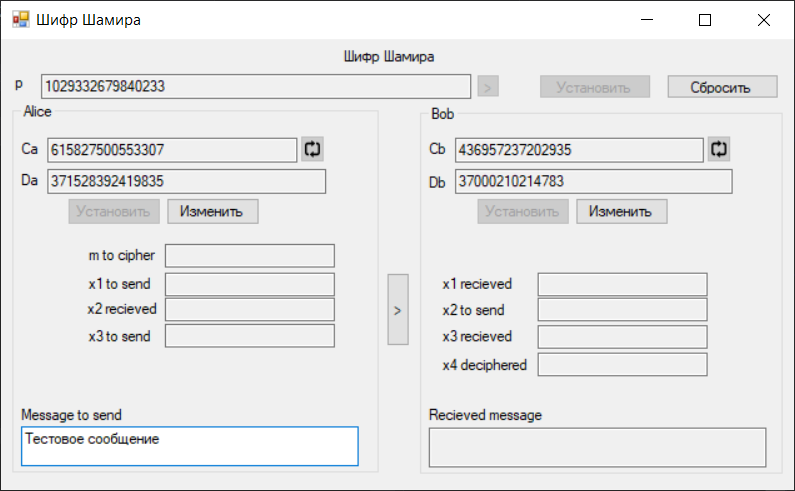


Рисунок 6. Подготовка сообщения.

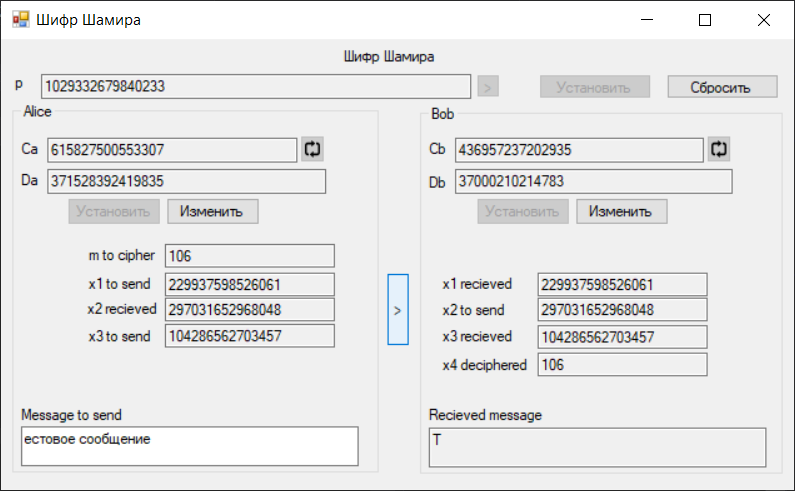


Рисунок 7. Отправка.

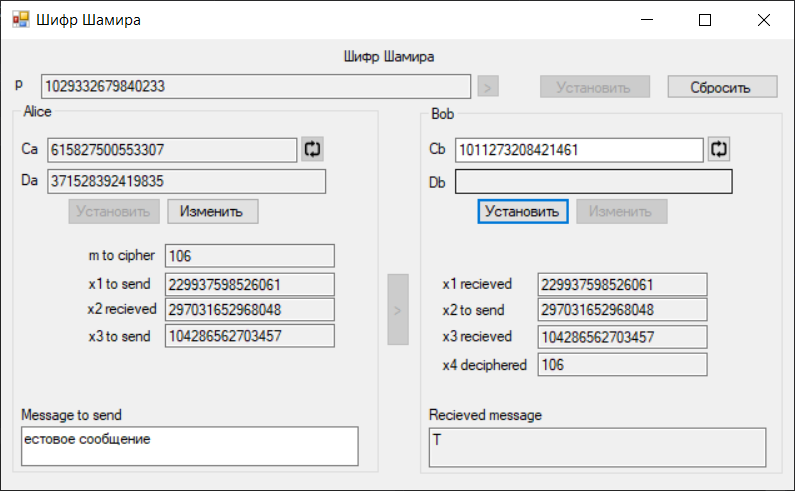


Рисунок 8. Замена ключей.

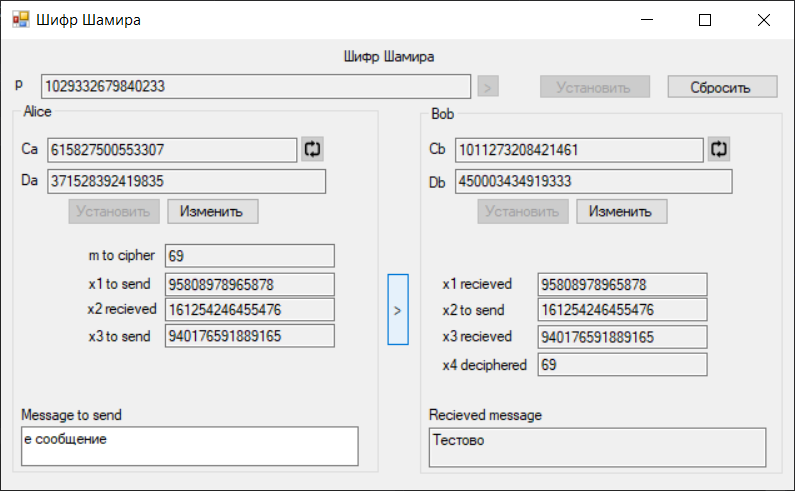


Рисунок 9. Продолжение отправки с новыми ключами.

**Вывод:**

В ходе выполнения лабораторной работы я ознакомилась с алгоритмом работы шифра Шамира, реализовала на практике программу отправке сообщений на языке C#.