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 №9**

|  |
| --- |
| Алгоритм электронной подписи RSA |
| *тема* |

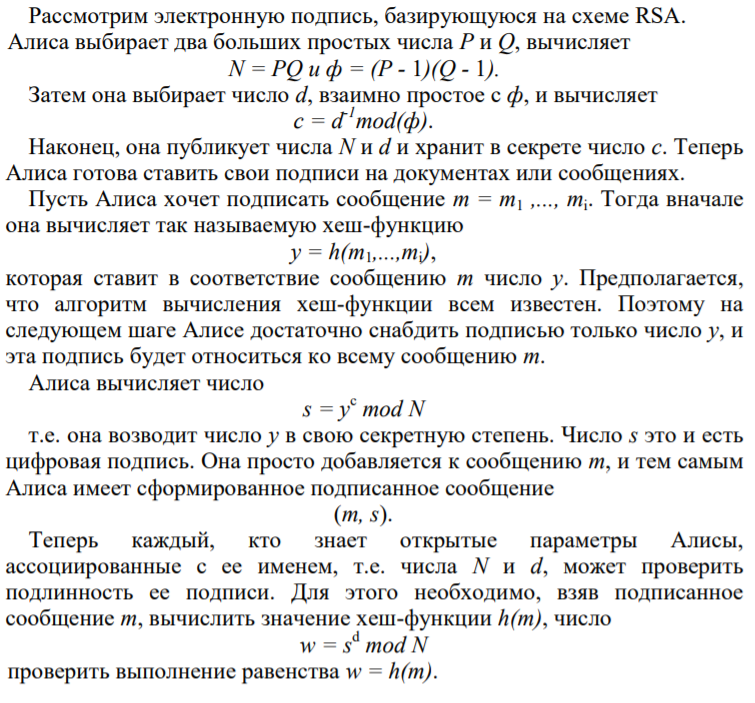
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Руководитель | | |  |  |  | В.И.Вайнштейн |
|  | |  |  | *подпись, дата* |  | *инициалы, фамилия* |
| Студент | КИ15-01 №031508683 | |  |  |  | М.С.Димаксян |
|  | *номер группы, зачетной книжки* | |  | *подпись, дата* |  | *инициалы,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

Красноярск 2019

**Задание:**

Изучить основные принципы построения и реализации алгоритма электронной подписи RSA.

**Теория:**

****

**Исходный код:**

**Функция подписи:**

private void f9\_buttonSign\_Click(object sender, EventArgs e)

{

BigInteger P = BigInteger.Parse(f9\_textP.Text);

BigInteger Q = BigInteger.Parse(f9\_textQ.Text);

BigInteger N = BigInteger.Parse(f9\_textN.Text);

BigInteger fi = P \* Q; // Функция Эйлера от N

BigInteger SecretKey = BigInteger.Parse(f9\_textSecretKey.Text);

//Открываем документ и вычисляем его хэш-функцию в переменную fileHash

string fileHash = "";

OpenFileDialog Load = new OpenFileDialog();

Load.InitialDirectory = "D:\\Учеба\\Крипта 2\\Labs\\Labs\\bin\\Debug\\TextFiles";

Load.DefaultExt = "txt";

Load.FileName = "in.txt";

Load.Filter = "txt files (\*.txt)|\*.txt|All files (\*.\*)|\*.\*";

Load.FilterIndex = 2;

Load.RestoreDirectory = true;

Load.Multiselect = false;

string sFileName;

if (Load.ShowDialog() == DialogResult.OK)

{

sFileName = Load.FileName;

byte[] file = File.ReadAllBytes(Load.FileName);

//Вычисление хэш-строки

if (f9\_chooseHashFun.Text == "MD5")

fileHash = HashFunctions.MD5.GetHashMD5(file);

if (f9\_chooseHashFun.Text == "SHA-1")

fileHash = HashFunctions.SHA1.GetHashSHA1(file);

if (f9\_chooseHashFun.Text == "ГОСТ Р 34.11-2012 256 бит")

fileHash = HashFunctions.GOST.GetHashGOST(file, 256);

if (f9\_chooseHashFun.Text == "ГОСТ Р 34.11-2012 512 бит")

fileHash = HashFunctions.GOST.GetHashGOST(file, 512);

}

BigInteger y = new BigInteger( Encoding.Default.GetBytes(fileHash));

BigInteger Sign = NumberTheory.BinaryModPow(y, SecretKey, N);

if (Sign > 0)

{

if (y > N) MessageBox.Show(("Подпись небезопасна: используйте значение N, большее h(m) = " + Convert.ToString(y)), "Предупреждение");

f9\_textBoxSign.Text = Convert.ToString(Sign);

}

else

f9\_textBoxSign.Text = ""; }

#### Функция проверки подписи:

private void f9\_buttonCheckSign\_Click(object sender, EventArgs e)

{

//Удаление недопустимых символов

f9\_textNRecieved.Text = new string(f9\_textNRecieved.Text.Where(t => char.IsDigit(t)).ToArray());

f9\_textOpenKeyRecieved.Text = new string(f9\_textOpenKeyRecieved.Text.Where(t => char.IsDigit(t)).ToArray());

f9\_textBoxSignRecieved.Text = new string(f9\_textBoxSignRecieved.Text.Where(t => char.IsDigit(t)).ToArray());

//Считывание полученных параметров подписи

BigInteger N = 0;

if (f9\_textNRecieved.TextLength > 0)

N = BigInteger.Parse(f9\_textNRecieved.Text);

BigInteger OpenKey = 0;

if (f9\_textOpenKeyRecieved.TextLength > 0)

OpenKey = BigInteger.Parse(f9\_textOpenKeyRecieved.Text);

BigInteger SignToCheck = 0;

if (f9\_textBoxSignRecieved.TextLength > 0)

SignToCheck = BigInteger.Parse(f9\_textBoxSignRecieved.Text);

//Открываем документ и вычисляем его хэш-функцию в переменную fileHash

string fileHash = "";

OpenFileDialog Load = new OpenFileDialog();

Load.InitialDirectory = "D:\\Учеба\\Крипта 2\\Labs\\Labs\\bin\\Debug\\TextFiles";

Load.DefaultExt = "txt";

Load.FileName = "in.txt";

Load.Filter = "txt files (\*.txt)|\*.txt|All files (\*.\*)|\*.\*";

Load.FilterIndex = 2;

Load.RestoreDirectory = true;

Load.Multiselect = false;

string sFileName;

if (Load.ShowDialog() == DialogResult.OK)

{

sFileName = Load.FileName;

byte[] file = File.ReadAllBytes(Load.FileName);

//Вычисление хэш-строки

if (f9\_chooseHashFunRecieved.Text == "MD5")

fileHash = HashFunctions.MD5.GetHashMD5(file);

if (f9\_chooseHashFunRecieved.Text == "SHA-1")

fileHash = HashFunctions.SHA1.GetHashSHA1(file);

if (f9\_chooseHashFunRecieved.Text == "ГОСТ Р 34.11-2012 256 бит")

fileHash = HashFunctions.GOST.GetHashGOST(file, 256);

if (f9\_chooseHashFunRecieved.Text == "ГОСТ Р 34.11-2012 512 бит")

fileHash = HashFunctions.GOST.GetHashGOST(file, 512);

}

BigInteger y = new BigInteger(Encoding.Default.GetBytes(fileHash)); //Вычисляем хэш файла

if (SignToCheck > 0 && OpenKey > 0 && N > 0)

{

BigInteger w = NumberTheory.BinaryModPow(SignToCheck, OpenKey, N);

if (w == y % N)

f9\_textVerificationResult.Text = "Подпись верна";

else

f9\_textVerificationResult.Text = "Подпись неверна";

}

else

{

f9\_textVerificationResult.Text = "Подпись неверна";

}

}

#### Функции установки параметров

private void f9\_buttonSetKeys\_Click(object sender, EventArgs e)

{

f9\_textP.Text = new string(f9\_textP.Text.Where(t => char.IsDigit(t)).ToArray());

f9\_textQ.Text = new string(f9\_textQ.Text.Where(t => char.IsDigit(t)).ToArray());

if (f9\_textP.TextLength > 0 && f9\_textQ.TextLength > 0)

{

if (RabinMiller.IsPrime(BigInteger.Parse(f9\_textP.Text), 10) && RabinMiller.IsPrime(BigInteger.Parse(f9\_textQ.Text), 10))

{

if (f9\_chooseHashFun.Text != "")

{

f9\_chooseHashFun.Enabled = false;

f9\_textP.ReadOnly = true;

f9\_textQ.ReadOnly = true;

f9\_textN.Text = Convert.ToString(BigInteger.Parse(f9\_textP.Text)\*BigInteger.Parse(f9\_textQ.Text));

f9\_buttonNextPrimeP.Enabled = false;

f9\_buttonNextPrimeQ.Enabled = false;

f9\_buttonSetKeys.Enabled = false;

f9\_buttonClearKeys.Enabled = true;

}

}

}

}

private void f9\_button\_SetSecret\_Click(object sender, EventArgs e)

{

f9\_textSecretKey.Text = new string(f9\_textSecretKey.Text.Where(t => char.IsDigit(t)).ToArray());

BigInteger SecretKey;

BigInteger P = BigInteger.Parse(f9\_textP.Text);

BigInteger Q = BigInteger.Parse(f9\_textQ.Text);

BigInteger N = BigInteger.Parse(f9\_textN.Text);

BigInteger fi = (P - 1) \* (Q - 1); // Функция Эйлера от N

if (f9\_textSecretKey.TextLength > 0)

{

SecretKey = BigInteger.Parse(f9\_textSecretKey.Text);

SecretKey = SecretKey % N;

if (SecretKey > 1 && NumberTheory.GCD(SecretKey, fi, out BigInteger x, out BigInteger y) == 1)

{

f9\_textOpenKey.Text = Convert.ToString(NumberTheory.Foo(SecretKey, fi));

f9\_button\_SetSecret.Enabled = false;

f9\_button\_GetRandomSecret.Enabled = false;

f9\_button\_ClearSecret.Enabled = true;

f9\_buttonSign.Enabled = true;

}

}

}

**Результат работы программы:**

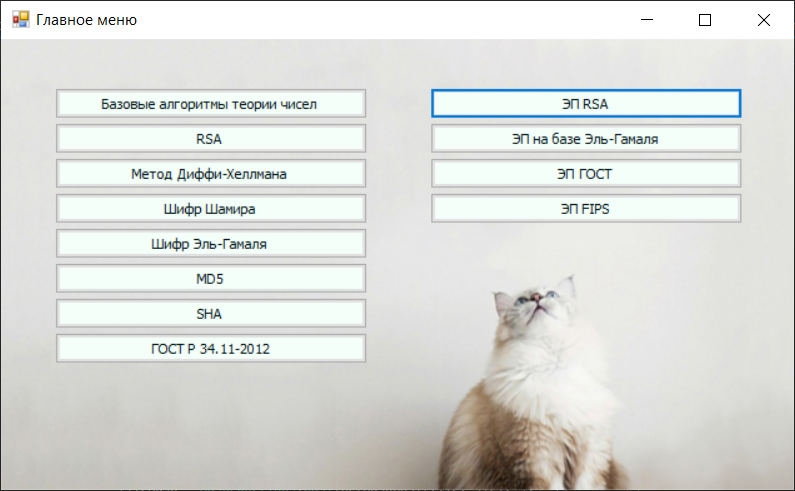


Рисунок 1. Главное меню программы.

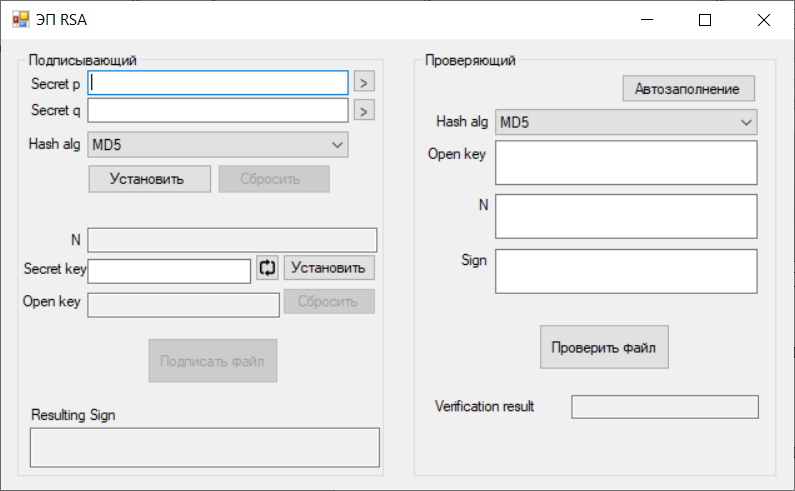


Рисунок 2. Окно электронной подписи RSA.

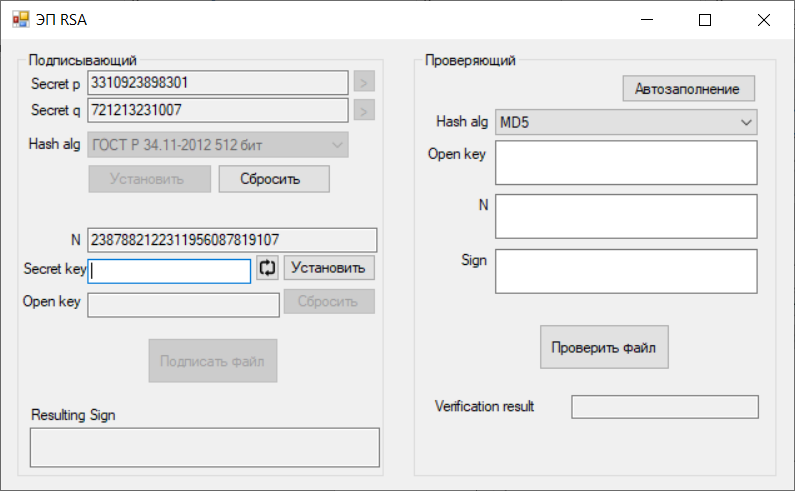


Рисунок 3. Установка параметров системы.

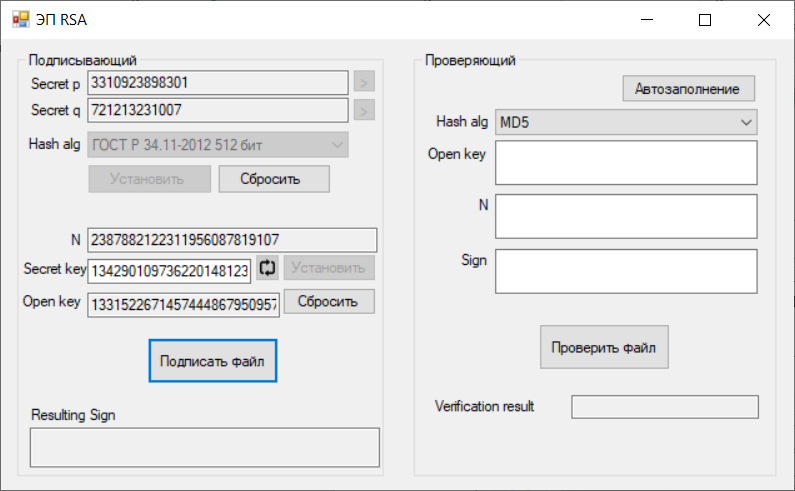


Рисунок 4. Установка ключей.

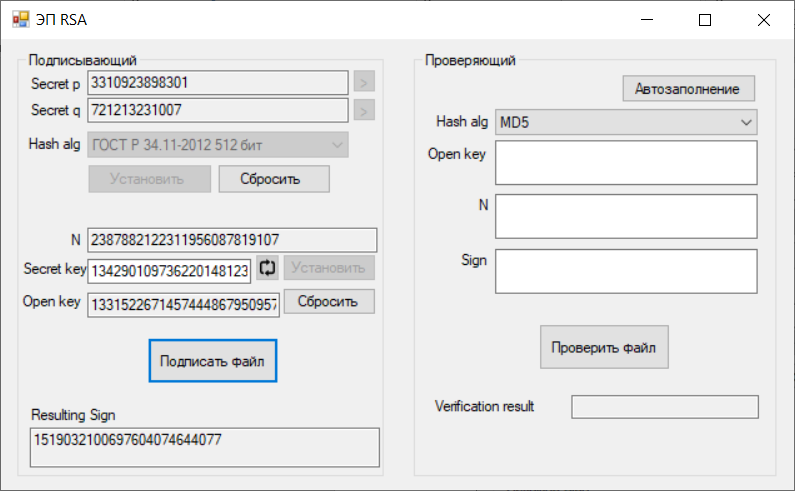


Рисунок 5. Подпись файла с текстом qwerty.

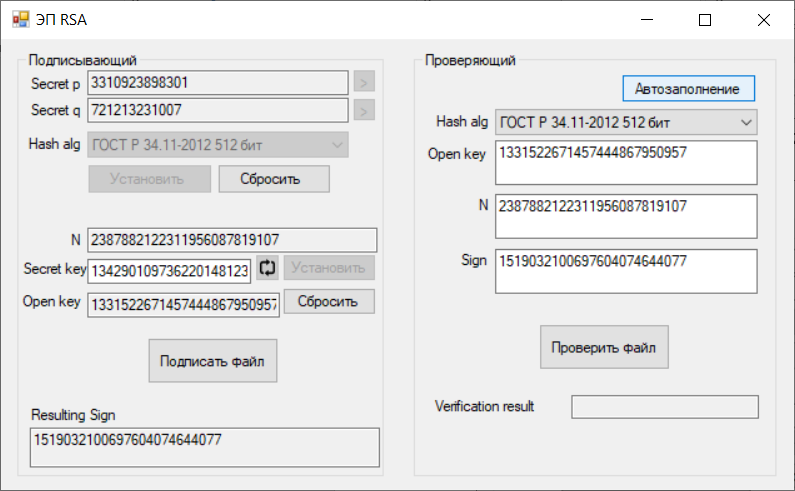


Рисунок 6. Автозаполнение полей для проверки.

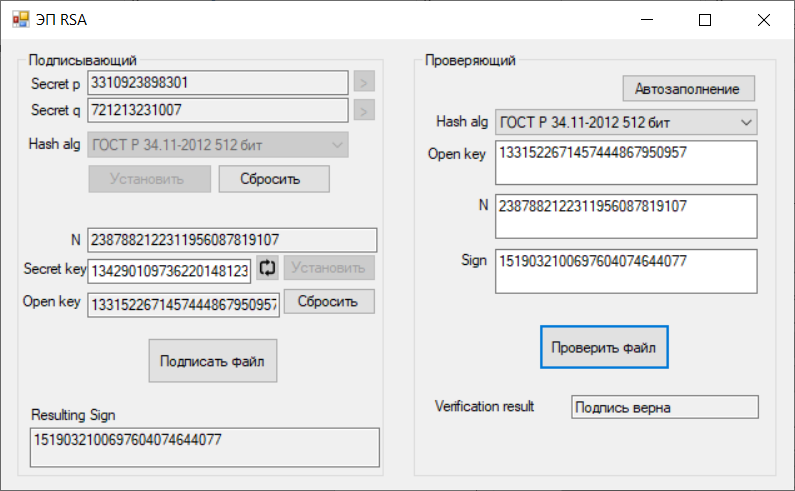


Рисунок 7. Результат проверки файла с текстом qwerty.

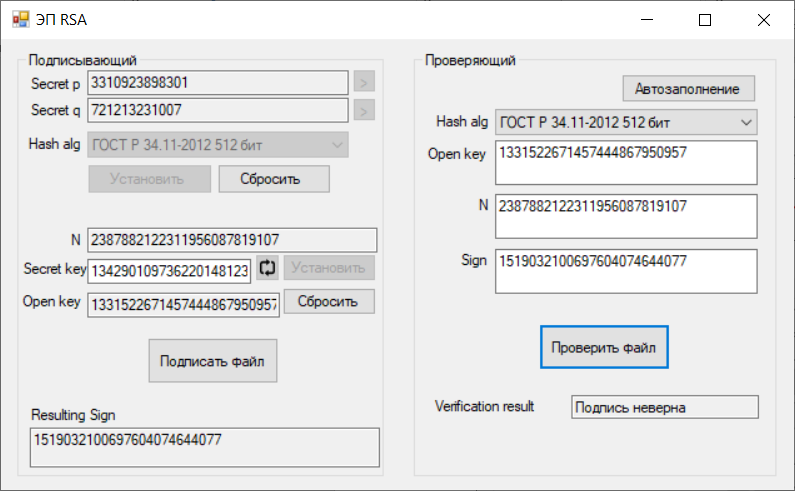


Рисунок 8. Результат проверки файла с другим текстом.

**Вывод:**

В ходе выполнения лабораторной работы я ознакомилась с алгоритмом электронной подписи RSA, реализовала на практике программу подписи и ее проверки.