# Практическая работа №6

# Электронная подпись

**Цели:**

1. Рассмотреть алгоритм цифровой подписи DSA.
2. Закрепить навыки использования криптографических классов платформы Microsoft .Net.

***Задание:*** Разработать приложение, которое позволит осуществлять вычисление и проверку электронной цифровой подписи с использованием алгоритма DSA для данных, хранящихся в файле. Для осуществления проверки, необходимо реализовать хранение ЭЦП и открытого ключа в отдельных файлах в папке, хранящей исходный файл.

***Константы:*** отсутствуют.

## Исходные данные: Открытое сообщение «Открытое сообщение Пузанова Владимира Евгеньевича»

**Код MainForm:**

***Листинг 1 – Код MainForm:***

namespace DSAApplication

{

public partial class MainForm : Form

{

// объект класса для криптоалгоритма DSA

DSACryptoServiceProvider dsa;

// буфер для открытого ключа ЭЦП

byte[] key;

// объекты для потоков защищенного файла, ЭЦП и открытого ключа

FileStream finStream, fsigStream, fpubStream;

// буфер для ЭЦП

byte[] sigbytes;

// буфер для ввода данных из защищенного файла

byte[] bytes;

// длина буфера ввода

int numBytesToRead;

// объект класса исключения

ArgumentException ex;

private void Browse\_Click(object sender, EventArgs e)

{

openFileDialog = new OpenFileDialog();

openFileDialog.Filter = "Все файлы (\*.\*)|\*.\*";

openFileDialog.CheckFileExists = true;

openFileDialog.InitialDirectory = Directory.GetCurrentDirectory();

openFileDialog.RestoreDirectory = true;

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

InputFName.Text = openFileDialog.FileName;

Sign.Enabled = InputFName.Text.Length > 0 &&

File.Exists(InputFName.Text);

Verify.Enabled = InputFName.Text.Length > 0 &&

File.Exists(InputFName.Text) &&

File.Exists(InputFName.Text + ".sig") &&

File.Exists(InputFName.Text + ".pub");

}

private void Sign\_Click(object sender, EventArgs e)

{

try

{

// отображение имени файла с ЭЦП (к имени исходного файла добавляетсярасширение.sig)

SigFName.Text = InputFName.Text + ".sig";

// отображение имени файла с открытым ключом (к имени исходного файладобавляется расширение .pub)

PubFName.Text = InputFName.Text + ".pub";

// создание объектов для файловых потоков

finStream = new FileStream(InputFName.Text, FileMode.Open);

fsigStream = new FileStream(SigFName.Text, FileMode.Create);

fpubStream = new FileStream(PubFName.Text, FileMode.Create);

// хеширование и вычисление ЭЦП для данных из входного потока

sigbytes = dsa.SignData(finStream);

// сохранение ЭЦП в файле

fsigStream.Write(sigbytes, 0, sigbytes.Length);

// экспорт открытого ключа

key = dsa.ExportCspBlob(false);

// сохранение открытого ключа в файле

fpubStream.Write(key, 0, key.Length);

// разблокировка кнопки "Проверить ЭЦП"

Verify.Enabled = true;

// закрытие файлов

finStream.Close();

fsigStream.Close();

fpubStream.Close();

}

// обработка ошибки криптографической операции

catch (CryptographicException ex)

{

// вывод сообщения об ошибке

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,

MessageBoxIcon.Error);

// закрытие файлов

finStream.Close();

fsigStream.Close();

fpubStream.Close();

}

// обработка остальных ошибок

catch (Exception ex)

{

// вывод сообщения об ошибке

MessageBox.Show(ex.Message, "Ошибка",

MessageBoxButtons.OK,

MessageBoxIcon.Error);

}

}

private void Verify\_Click(object sender, EventArgs e)

{

try

{

// отображение имени файла с ЭЦП (к имени исходного файла добавляетсярасширение.sig

SigFName.Text = InputFName.Text + ".sig";

// отображение имени файла с открытым ключом(к имени исходного файладобавляется расширение .pub)

PubFName.Text = InputFName.Text + ".pub";

// создание объектов для файловых потоков

finStream = new FileStream(InputFName.Text, FileMode.Open);

fsigStream = new FileStream(SigFName.Text, FileMode.Open);

fpubStream = new FileStream(PubFName.Text, FileMode.Open);

// получение длины ЭЦП

numBytesToRead = (int)fsigStream.Length;

// выделение памяти для буфера ЭЦП

sigbytes = new byte[fsigStream.Length];

// чтение ЭЦП из файла

fsigStream.Read(sigbytes, 0, numBytesToRead);

// получение длины открытого ключа

numBytesToRead = (int)fpubStream.Length;

// выделение памяти для блоба с открытым ключом

key = new byte[fpubStream.Length];

// чтение открытого ключа из файла

fpubStream.Read(key, 0, numBytesToRead);

// получение длины защищенного файла

numBytesToRead = (int)finStream.Length;

// выделение памяти для буфера ввода

bytes = new byte[finStream.Length];

// чтение данных из защищенного файла

finStream.Read(bytes, 0, numBytesToRead);

// импорт открытого ключа

dsa.ImportCspBlob(key);

// проверка ЭЦП из файла для данных из файлового потока

if (dsa.VerifyData(bytes, sigbytes))

MessageBox.Show("Подпись верна!",

"Результат проверки ЭЦП");

else MessageBox.Show("Подпись не верна!",

"Результат проверки ЭЦП");

// закрытие файлов

finStream.Close();

fsigStream.Close();

fpubStream.Close();

}

// обработка ошибки криптографической операции

catch (CryptographicException ex)

{

// вывод сообщения об ошибке

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,

MessageBoxIcon.Error);

// закрытие файлов

finStream.Close();

fsigStream.Close();

fpubStream.Close();

}

// обработка остальных ошибок

catch (Exception ex)

{

// вывод сообщения об ошибке

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,

MessageBoxIcon.Error);

}

}

// объект для диалога открытия файла

OpenFileDialog openFileDialog;

public MainForm()

{

InitializeComponent();

dsa = new DSACryptoServiceProvider(1024);

}

}

}

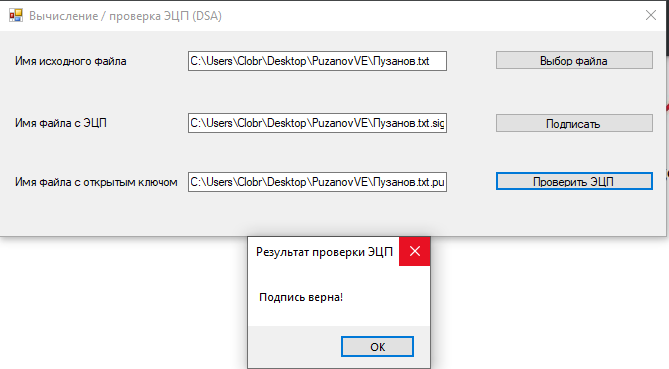


Рис.1 – проверка ЭЦП

Выполнил студент Пузанов В. Е., ФИТУ 010304-КМСб-о22 Проверил ст. преподаватель каф. ПМ Черноиван Д.Н.