МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждения образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Практическая работа №4**

по дисциплине «Основы информационной безопасности»

на тему: Криптографическая защита информации

Выполнил студент 2 курса 7 группы специальность ПОИБМС Володькин Никифор Дмитриевич

(Ф.И.О.)

Преподаватель Ржеутская Надежда Викентьевна

(Ф.И.О.)

**Цель: Овладение основными криптографическими алгоритмами асимметричного шифрования.**

**Выполнение работы**

# **Реализация элементов криптосистемы RSA**

RSA (аббревиатура от фамилий Rivest, Shamir и Adleman) — криптографический алгоритм с открытым ключом, основывающийся на вычислительной сложности задачи факторизации больших целых чисел.

Криптосистема RSA стала первой системой, пригодной и для шифрования, и для цифровой подписи. Алгоритм используется в большом числе криптографических приложений, включая PGP, S/MIME, TLS/SSL, IPSEC/IKE и других.

Весь алгоритм расписан в таблице:

|  |  |  |
| --- | --- | --- |
| Этап | Описание операции | Результат операции |
| Генерация ключей | Выбрать два простых различных числа | p=3557,  q=2579 |
| Вычислить модуль (произведение) | n = p \cdot q = 3557 \cdot 2579 = 9173503 |
| Вычислить функцию Эйлера | \varphi(n) = (p-1) (q-1) = 9167368 |
| Выбрать открытую экспоненту | e = 3 |
| Вычислить секретную экспоненту | d = e^{-1} \mod \varphi(n)  d = 6111579 |
| Опубликовать открытый ключ | \{e, n\} = \{3,9173503 \} |
| Сохранить закрытый ключ | \{d, n\} = \{6111579, 9173503 \} |
| Шифрование | Выбрать текст для зашифровки | m = 111111 |
| Вычислить шифротекст | \begin{align} c &= E(m) \\  &= m^e \mod n \\  &= 111111^3   \mod 9173503 \\  &= 4051753 \end{align} |
| Расшифрование | Вычислить исходное сообщение | \begin{align} m &= D(c) = \\   &= c^d \mod n \\   &= 4051753^{6111579} \mod 9173503 \\   &= 111111 \end{align} |

Было написано следующее приложение на языке C#, реализующее шифрование RSA с помощью класса **RSACryptoServiceProvider**:

using System;

using System.Security.Cryptography;

using System.Text;

namespace ConsoleApp1

{

class RSACSPSample

{

static void Main()

{

try

{

UnicodeEncoding ByteConverter = new UnicodeEncoding();

Console.Write("Введите сообщение: ");

byte[] dataToEncrypt = ByteConverter.GetBytes(Console.ReadLine());

byte[] encryptedData;

byte[] decryptedData;

using (RSACryptoServiceProvider RSA = new RSACryptoServiceProvider())

{

encryptedData = RSAEncrypt(dataToEncrypt, RSA.ExportParameters(false), false);

decryptedData = RSADecrypt(encryptedData, RSA.ExportParameters(true), false);

Console.WriteLine("Расшифрованное сообщение: {0}", ByteConverter.GetString(decryptedData));

}

}

catch (ArgumentNullException)

{

Console.WriteLine("Не удалось расшифровать сообщение.");

}

}

public static byte[] RSAEncrypt(byte[] DataToEncrypt, RSAParameters RSAKeyInfo, bool DoOAEPPadding)

{

try

{

byte[] encryptedData;

using (RSACryptoServiceProvider RSA = new RSACryptoServiceProvider())

{

RSA.ImportParameters(RSAKeyInfo);

encryptedData = RSA.Encrypt(DataToEncrypt, DoOAEPPadding);

}

return encryptedData;

}

catch (CryptographicException e)

{

Console.WriteLine(e.Message);

return null;

}

}

public static byte[] RSADecrypt(byte[] DataToDecrypt, RSAParameters RSAKeyInfo, bool DoOAEPPadding)

{

try

{

byte[] decryptedData;

using (RSACryptoServiceProvider RSA = new RSACryptoServiceProvider())

{

RSA.ImportParameters(RSAKeyInfo);

decryptedData = RSA.Decrypt(DataToDecrypt, DoOAEPPadding);

}

return decryptedData;

}

catch (CryptographicException e)

{

Console.WriteLine(e.ToString());

return null;

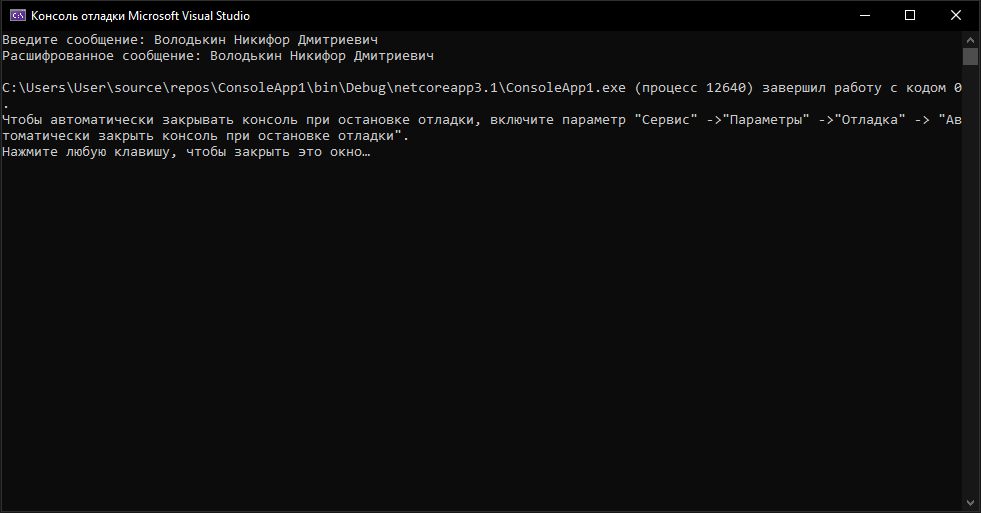
}

}

}

}

Результат выполнения программы:



**Вывод: Изучены алгоритмы асимметричного шифрования.**