МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Отчет по лабораторной работе №8

Исследование ассиметричных шифров

RSA и Эль-Гамаля

Выполнил студент: Плюто Э. В.

ФИТ 3 курса, 5 группа

Проверил: Савельева М. Г.

Минск 2024

**Практическое задание:**

**1)** Разработать простое консольное приложение, которое должно вычислять зависимости времени вычисления параметра *у*, где: *а* – десятичное число от 5 до 35, *х* – простые числа из диапазона от 103 до 10100, *n* – числа, в двоичном виде состоящие из 1024 и 2048 битов.

Полученные значения и время выполнения подсчетов представлены ниже.

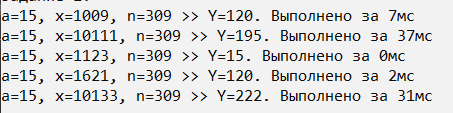


Рисунок 1 – Результаты вычисления *у*

График зависимости времени выполнения от переменных представлен на рисунке 2.

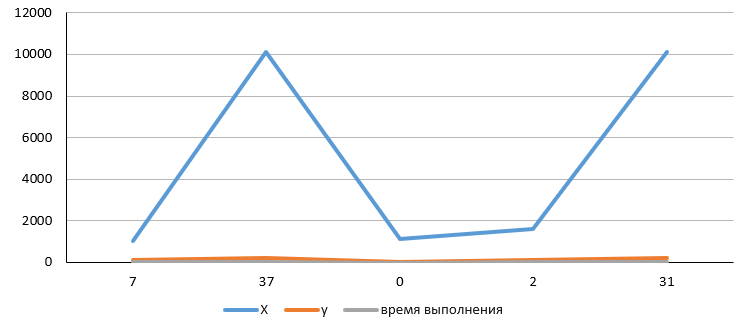


Рисунок 2 – График зависимости времени выполнения

Из графика зависимости времени выполнения можно заметить, что при увеличении значения *x* увеличивается время выполнения.

**2)** Разработать авторское приложение, которое должно реализовывать зашифрование и расшифрование текстовых документов на основе алгоритмов RSA и Эль-Гамаля, определять время выполнения операций.

Реализация алгоритма шифрования и расшифрования RSA представлена на листинге 1.

|  |
| --- |
| public static void RSA()  {  RSAParameters publicKey;  RSAParameters privateKey;  byte[] openTextBytes = Encoding.UTF8.GetBytes(text);  using (var rsa = new RSACryptoServiceProvider(4096))  {  publicKey = rsa.ExportParameters(false);  privateKey = rsa.ExportParameters(true);  }  using (var rsa = new RSACryptoServiceProvider())  { Stopwatch sw = new Stopwatch();  sw.Start();  rsa.ImportParameters(publicKey);  var encryptedTextRSA = rsa.Encrypt(openTextBytes, false); sw.Stop();  Console.WriteLine("Зашифрованный текст(RSA): " + string.Join(" ", encryptedTextRSA) + '\n' + "Время выполнения шифрования: " + sw.ElapsedMilliseconds + "мс");  sw.Restart();  rsa.ImportParameters(privateKey);  var decryptedTextRSA = rsa.Decrypt(encryptedTextRSA, false);  sw.Stop();  Console.WriteLine("Расшифрованный текст(RSA): " + Encoding.UTF8.GetString(decryptedTextRSA) + '\n' + "Время выполнения расшифрования: " + sw.ElapsedMilliseconds + "мс");  }  } |

Листинг 1 – Реализация алгоритма RSA

Результат выполнения кода представлен на рисунке 2.

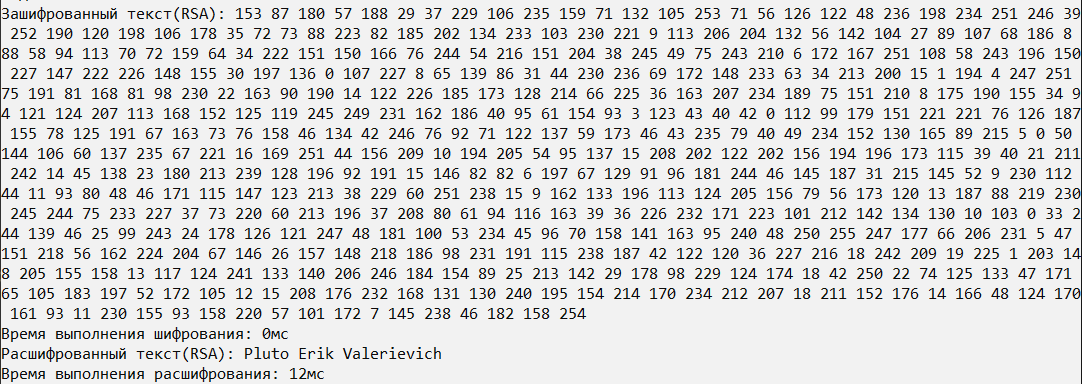


Рисунок 2 – Результат шифрования и расшифрования сообщения

Время выполнения шифрования и расшифрования сообщения представлено ниже.

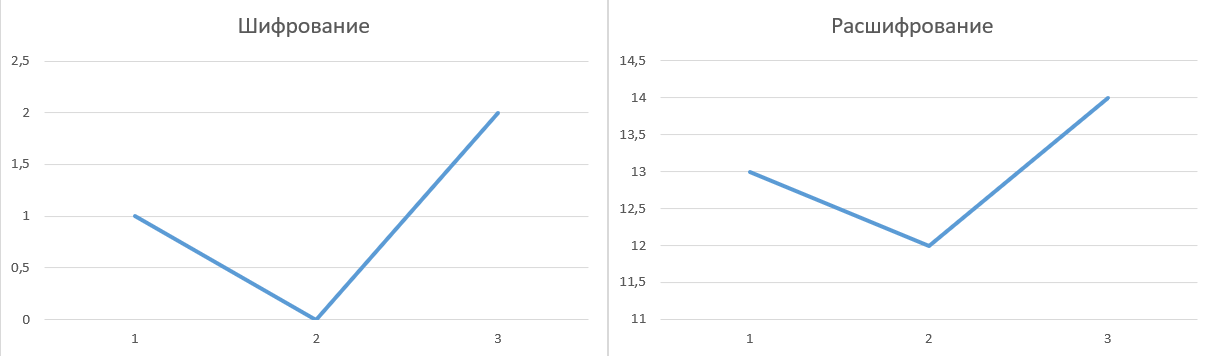


Рисунок 3 – Время выполнения шифрования и расшифрования

Реализация шифрования и расшифрования алгоритма Эль-Гамаля представлена на листинге 2.

|  |
| --- |
| public static void ElGamal(BigInteger p, BigInteger g, BigInteger x)  {  Stopwatch sw = new Stopwatch();  sw.Start();  BigInteger y = BigInteger.ModPow(g, x, p);  Random random = new Random();  BigInteger k;  while (k <= 1 || k >= p - 1);  {  byte[] bytes = new byte[p.ToByteArray().Length];  random.NextBytes(bytes);  k = new BigInteger(bytes);  }  BigInteger a = BigInteger.ModPow(g, k, p);  BigInteger b = BigInteger.ModPow(y, k, p);  byte[] ciphertext = new byte[2 \* text.Length];  for (int i = 0; i < text.Length; i++)  {  ciphertext[2 \* i] = (byte)(text[i] ^ (byte)a);  ciphertext[2 \* i + 1] = (byte)(text[i] ^ (byte)b);  }  sw.Stop();  Console.WriteLine("Зашифрованный текст(Эль-Гамаль): " + Encoding.ASCII.GetString(ciphertext) + '\n' + "Время выполнения шифрования: " + sw.Elapsed.TotalMilliseconds + "мс");  sw.Restart();  byte[] encrypted = new byte[ciphertext.Length / 2];  for (int i = 0; i < encrypted.Length; i++)  {  BigInteger n = new BigInteger(ciphertext[2 \* i]);  BigInteger m = new BigInteger(ciphertext[2 \* i + 1]);  encrypted[i] = (byte)(n ^ m ^ x);  }  sw.Stop();  Console.WriteLine("Расшифрованный текст(Эль-Гамаль): " + Encoding.ASCII.GetString(encrypted) + '\n' + "Время выполнения расшифрования: " + sw.Elapsed.TotalMilliseconds + "мс");  } |

Листинг 2 – Реализация алгоритма Эль-Гамаля

Результат выполнения кода представлен на рисунке 4.

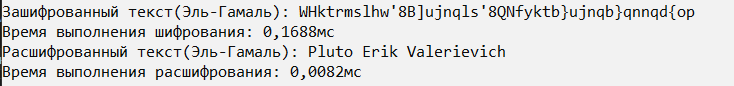


Рисунок 4 – Результат шифрования и расшифрования сообщения

Время выполнения шифрования и расшифрования сообщения представлено ниже.

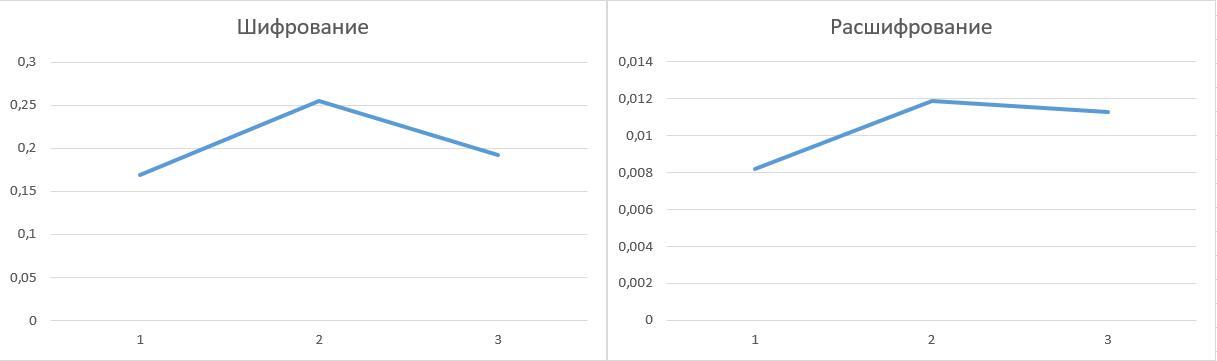


Рисунок 5 – Время выполнения шифрования и расшифрования

**3)** Оценить производительность обоих алгоритмов и относительное изменение объемов криптотекстов (по отношению к объемам открытых текстов).

Относительное изменение объемов криптотекста представлено на рисунке 6.

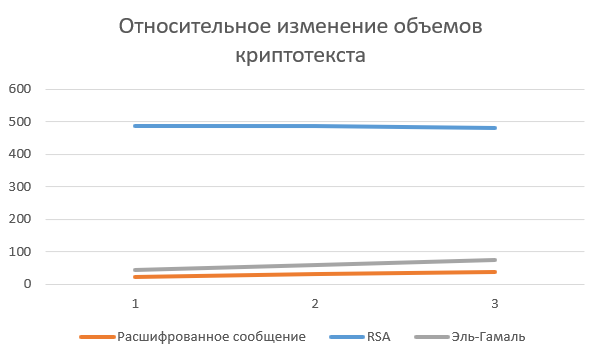


Рисунок 6 – Относительное изменение объемов криптотекста

Из рисунка представленного выше можно заметить, что при увеличении количества символов исходного текста растет и число символов зашифрованного алгоритмом Эль-Гамаля сообщения, а количество символов сообщения зашифрованного с помощью алгоритма RSA не зависит от количества символов исходного сообщения.

**Вывод:**

Были приобретены практические навыки разработки и использования приложений для реализации ассиметричных шифров RSA и Эль-Гамаля. Были реализованы алгоритмы RSA и Эль-Гамаля. Также было замерено время выполнения шифрования и расшифрования, а также построен соответствующий график.