МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Отчет по лабораторной работе №10

Исследование асимметричных шифров RSA и Эль-Гамаля

Выполнил студентка: Авсюкевич П.В.

ФИТ 3 курса, 5 группа

Проверила: Савельева М. Г.

Минск 2024

Задание 1. Составить табличную или графическую форму зависимости времени вычисления параметра *у*, функционально заданного выражением вида: *y = ax* mod *n.* Реализация зашифрования представлена на листинге 1.

|  |
| --- |
| static BigInteger ParamY(BigInteger a, BigInteger x, BigInteger n)  {  var pow = a;  for (BigInteger i = 1; i < x; i++)  {  pow \*= a;  }  return pow % n;  } |

Листинг 1 – Реализация функции

Результат выполнения программы представлен на рисунке 1.

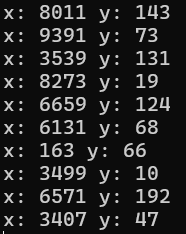


Рисунок 1 – Результат выполнения программы

График времени выполнения представлен ниже.

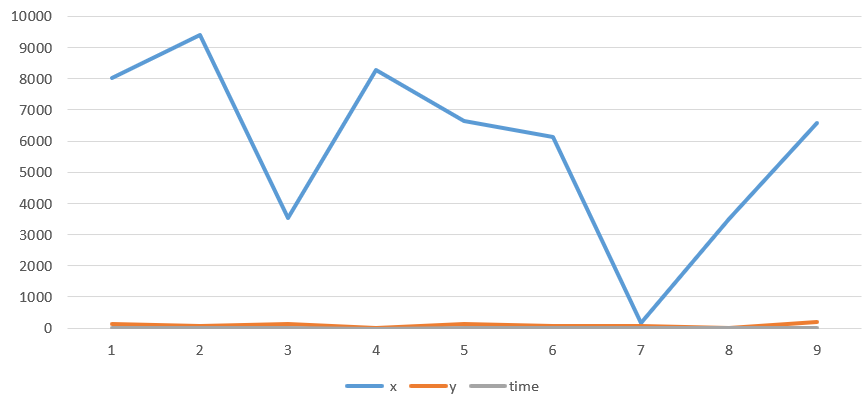


Рисунок 2 – График выполнения программы

Задание 2. Зашифровать и расшифровать текст на основе алгоритма RSA.

Алгоритм зашифрования представлен на листинге 2.

|  |
| --- |
| public static byte[] Encrypt(byte[] plaintext, RSAParameters publicKey)  {  var rsa = new RSACryptoServiceProvider();  rsa.ImportParameters(publicKey);  return rsa.Encrypt(plaintext, true);  } |

Листинг 2 – Шифрование на основе алгоритма RSA

Для расшифрования применяется алгоритм, представленный на листинге 3.

|  |
| --- |
| public static byte[] Decrypt(byte[] ciphertext, RSAParameters privateKey)  {  var rsa = new RSACryptoServiceProvider();  rsa.ImportParameters(privateKey);  return rsa.Decrypt(ciphertext, true);  } |

Листинг 3 – Расшифрование на основе алгоритма RSA

Результат выполнения программы представлен на рисунке 3.

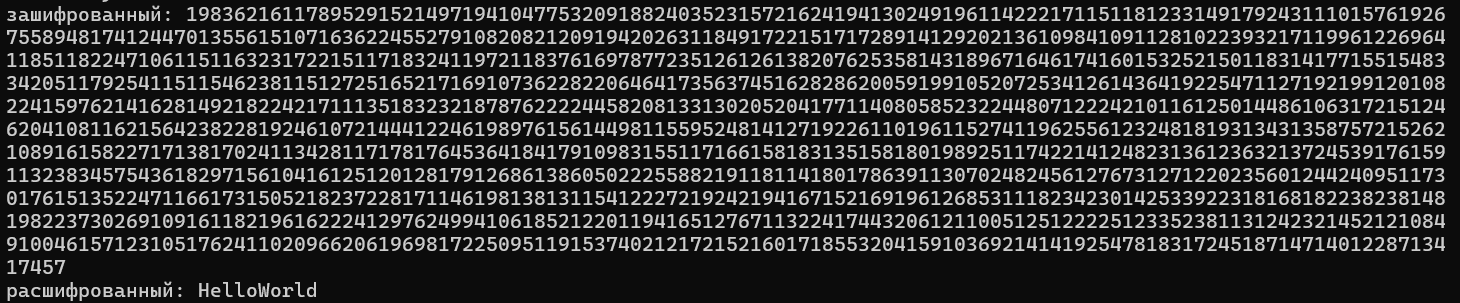


Рисунок 3 – Результат работы алгоритма RSA

График времени представлен ниже.

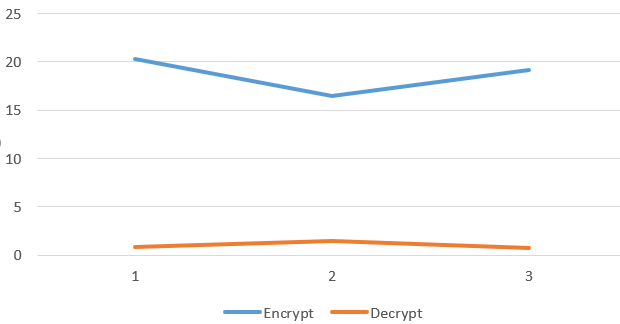


Рисунок 4 – График времени шифрования/расшифрования алгоритма RSA

Задание 3. Шифрование/расшифрование текста на основе алгоритма Эль-Гамаля. Для реализации алгоритма написана функция на листинге 4.

|  |
| --- |
| public byte[] Encrypt(byte[] plaintext)  {  Random random = new Random();  BigInteger k;  do  {  byte[] bytes = new byte[p.ToByteArray().Length];  random.NextBytes(bytes);  k = new BigInteger(bytes);  } while (k <= 1 || k >= p - 1);  BigInteger a = BigInteger.ModPow(g, k, p);  BigInteger b = BigInteger.ModPow(y, k, p);  byte[] ciphertext = new byte[2 \* plaintext.Length];  for (int i = 0; i < plaintext.Length; i++)  {  ciphertext[2 \* i] = (byte)(plaintext[i] ^ (byte)a);  ciphertext[2 \* i + 1] = (byte)(plaintext[i] ^ (byte)b);  }  return ciphertext;  } |

Листинг 4 – Шифрование по алгоритму Эль-Гамаля

Шифрование происходит по формуле: *m = ((bi (ai)x)-1) mod p*. На листинге 5 представлен код реализации.

|  |
| --- |
| public byte[] Decrypt(byte[] ciphertext)  {  byte[] plaintext = new byte[ciphertext.Length / 2];  for (int i = 0; i < plaintext.Length; i++)  {  BigInteger a = new BigInteger(ciphertext[2 \* i]);  BigInteger b = new BigInteger(ciphertext[2 \* i + 1]);  plaintext[i] = (byte)(a ^ b ^ x);  }  return plaintext;  } |

Листинг 5 – Расшифрование по алгоритму Эль-Гамаля

Результат выполнения программы на рисунке 5.



Рисунок 5 – Результат выполнения программы

График врмени выполнения программы представлен ниже.

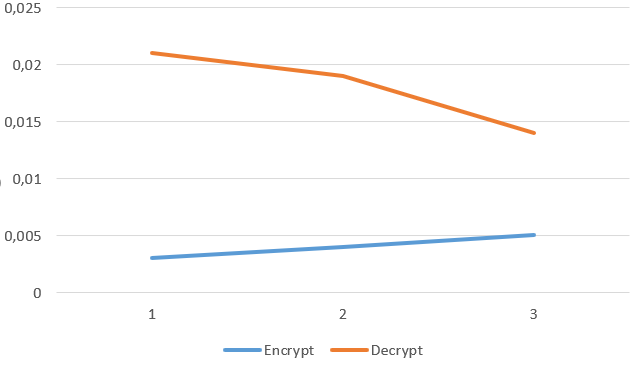


Рисунок 6 – График времени выполнение алгоритма Эль-Гамаля

Для оценки относительного изменения объема криптотекста была подсчитана длина исходного текста, которая составила 10 символов и длина зашифрованного текста в RSA: 51 символов, Эль-Гамаль: 20. Относительное изменение объемов криптотекста представлено на рисунке 7.

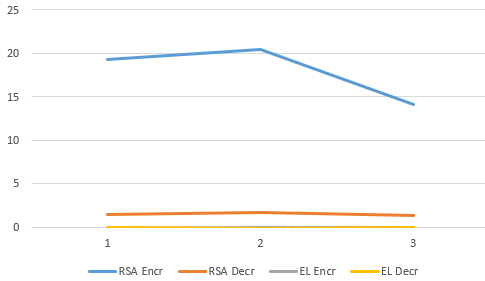


Рисунок 7 – Относительное изменение объемов криптотекста

В ходе изучения и практической работы были освоены и усвоены основы разработки программ для реализации двух основных ассиметричных шифров: RSA и Эль-Гамаля.