МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Отчет по лабораторной работе №10

Исследование алгоритмов генерации и верификации

электронной цифровой подписи

Выполнил студент: Плюто Э. В.

ФИТ 3 курса, 5 группа

Проверил: Савельева М. Г.

Минск 2024

**Практическое задание:**

**1)** Разработать авторское приложение, которое должно реализовывать генерацию и верификацию ЭЦП на основе алгоритмов RSA, Эль-Гамаля и Шнорра, оценку времени выполнения указанных процедур.

Реализация алгоритма генерации подписи при помощи алгоритма RSA и алгоритма хеширования MD5, созданных в предыдущих лабораторных работах, представлена на листинге 1.

|  |
| --- |
| public static byte[] CreateRSASignature(RSAParameters privateKey)  {  using (var rsa = RSA.Create())  {  rsa.ImportParameters(privateKey);  byte[] hash = GetMD5Hash();  return rsa.SignHash(hash, HashAlgorithmName.MD5, RSASignaturePadding.Pkcs1);  }  } |

Листинг 1 – Реализация алгоритма генерации подписи

Реализация алгоритма верификации подписи представлена на листинге 2.

|  |
| --- |
| public static bool VerifyRSASignature(byte[] signature, RSAParameters publicKey)  {  using (var rsa = RSA.Create())  {  rsa.ImportParameters(publicKey);  byte[] hash = GetMD5Hash();  return rsa.VerifyHash(hash, signature, HashAlgorithmName.MD5, RSASignaturePadding.Pkcs1);  }  } |

Листинг 2 – Реализация алгоритма верификации подписи

Результат выполнения генерации и верификации подписи представлены на рисунке 1.

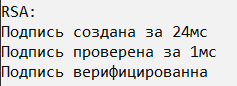


Рисунок 1 – Результат выполнения с использованием алгоритма RSA

Графики времени выполнения генерации подписи и ее верификации представлены ниже.

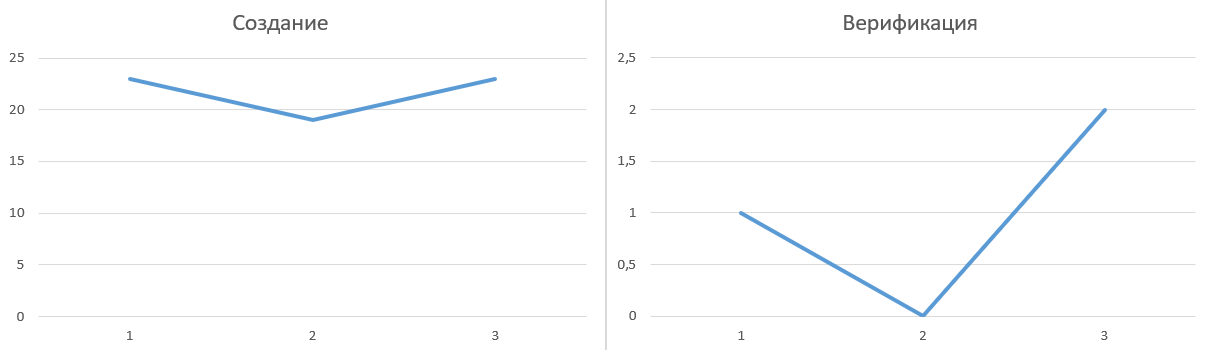


Рисунок 2 – Графики времени выполнения генерации и верификации подписи

Реализация алгоритма генерации подписи при помощи алгоритма Эль-Гамаля, созданного в предыдущих лабораторных работах, представлена на листинге 3.

|  |
| --- |
| public static byte[] GenerateElGamalSignature(int p, int g, int x, int k, int h)  {  int m = p - 1;  int k\_inverse = 1;  while (k\_inverse \* k % m != 1)  {  k\_inverse++;  }  int a = (int)BigInteger.ModPow(g, k, p);  var b = new BigInteger((k\_inverse \* (h - (x \* a) % m) % m) % m);  byte[] signature = new byte[2 \* sizeof(int)];  Buffer.BlockCopy(BitConverter.GetBytes(a), 0, signature, 0, sizeof(int));  Buffer.BlockCopy(BitConverter.GetBytes((int)b), 0, signature, sizeof(int), sizeof(int));  return signature;  } |

Листинг 3 – Реализация алгоритма генерации подписи

Реализация алгоритма верификации подписи представлена на листинге 4.

|  |
| --- |
| public static bool VerifyElGamalSignature(int p, int g, int y, int h, byte[] signature)  {  int a = BitConverter.ToInt32(signature, 0);  int b = BitConverter.ToInt32(signature, sizeof(int));  var hash1 = BigInteger.ModPow(BigInteger.Pow(y, a) \* BigInteger.Pow(a, b), 1, p);  var hash2 = BigInteger.ModPow(g, h, p);  return hash1 == hash2;  } |

Листинг 4 – Реализация алгоритма верификации подписи

Результат выполнения генерации и верификации подписи представлены на рисунке 3.

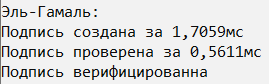


Рисунок 3 – Результат выполнения с использованием алгоритма Эль-Гамаля

Графики времени выполнения генерации подписи и ее верификации представлены ниже.

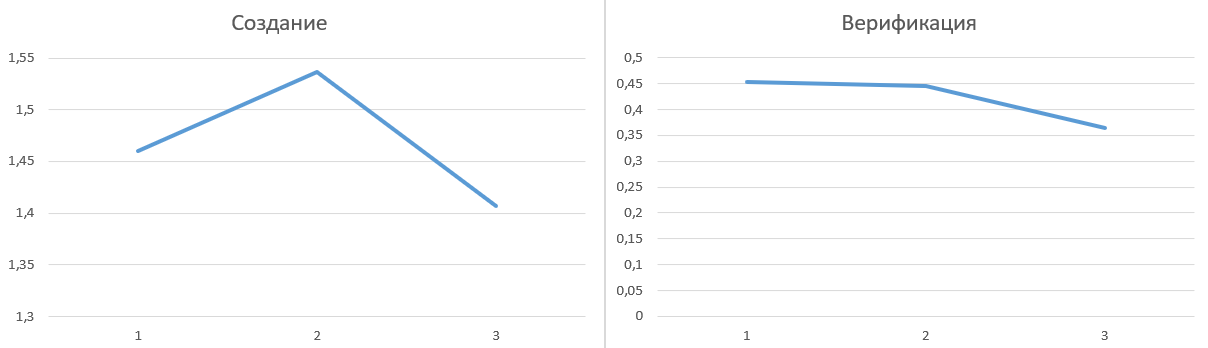


Рисунок 4 – Графики времени выполнения генерации и верификации подписи

Реализация алгоритма генерации и верификации подписи при помощи алгоритма Шнорра, представлена на листинге 5.

|  |
| --- |
| using (var schnorr = new ECDsaCng())  {  sw.Restart();  signature = schnorr.SignData(GetMD5Hash());  sw.Stop();  Console.WriteLine("Подпись создана за " + sw.ElapsedMilliseconds + "мс");  sw.Restart();  isValid = schnorr.VerifyData(GetMD5Hash(), signature);  sw.Stop();  Console.WriteLine("Подпись проверена за " + sw.ElapsedMilliseconds + "мс");  if (isValid)  Console.WriteLine("Подпись верифицированна");  else  Console.WriteLine("Подпись не верифицированна");  } |

Листинг 5 – Реализация ЭЦП Шнорра

Результат выполнения генерации и верификации подписи представлены на рисунке 5.

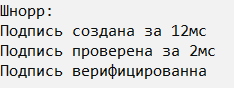


Рисунок 5 – Результат выполнения с использованием алгоритма Шнорра

Графики времени выполнения генерации подписи и ее верификации представлены ниже.

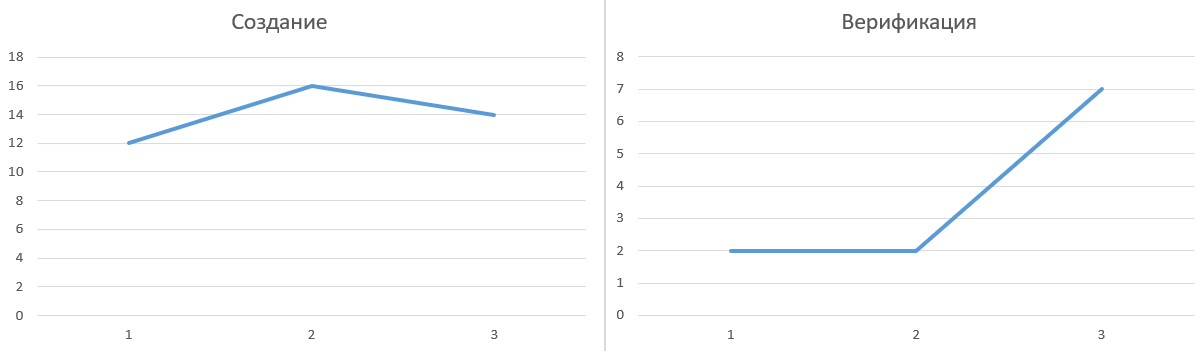


Рисунок 6 – Графики времени выполнения генерации и верификации подписи

Из описанных выше графиков можно заметить, что самым быстрым является алгоритм Эль-Гамаля.

**Вывод:**

Были приобретены практические навыки разработки и использования приложений для генерации и верификации ЭЦП. Были реализованы алгоритмы RSA, Эль-Гамаля и Шнора. Также было замерено время выполнения создания и верификации подписей, а также построены соответствующие графики.