Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Отчет по лабораторной работе №12

«Исследование алгоритмов генерации и верификации электронной цифровой подписи»

Студентка: Пунько А.А,

ФИТ 3 курс 5 группа

Преподаватель: Берников В. О.

Минск 2020

**Цель:** изучение алгоритмов генерации и верификации электронной цифровой подписи и приобретение практических навыков их реализации

**Задачи:**

1.Закрепить теоретические знания по алгебраическому описанию и алгоритмам реализации операций генерации и верификации электронной цифровой подписи (ЭЦП).

2. Получить навыки практической реализации методов генерации и верификации ЭЦП на основе хеширования подписываемых сообщений и алгоритмов RSA, Эль-Гамаля и Шнорра, а также DSA.

3. Разработать приложение для реализации заданных алгоритмов генерации и верификации ЭЦП. 4. Оценить скорость генерации и верификации ЭЦП.

5. Результаты выполнения лабораторной работы оформить в виде описания разработанного приложения, методики выполнения экспериментов с использованием приложения и результатов эксперимента.

# Теоретические сведения

Электронная цифровая подпись (ЭЦП) является важным элементом современных информационных систем, использующих методы и технологии криптографического преобразования информации.

Электронная цифровая подпись – контрольная характеристика сообщения, которая вырабатывается с использованием личного ключа, проверяется с использованием открытого ключа, служит для контроля целостности и подлинности сообщения и обеспечивает невозможность отказа от авторства.

Таким образом, ЭЦП выполняет те же функции, что и собственноручная (поставленная «от руки») подпись:

* аутентифицировать лицо, подписавшее сообщение;
* контролировать целостность подписанного сообщения;
* защищать сообщение от подделок;
* доказать авторство лица, подписавшего сообщение, если это лицо отрицает свое авторство.

Важнейшими отличительными особенностям ЭЦП являются:

* ЭЦП представляет собой бинарную последовательность (в отличие от графического образа, каковым является подпись от руки);
* указанная бинарная последовательность зависит от содержания подписываемого сообщения.

Основным компонентом в технологии ЭЦП является ключ. Принадлежность ключа, в предположении, что он известен только законным пользователям, позволяет решать все «возложенные на ЭЦП», сформированную на основе этого ключа, задачи. В соответствии с этим обстоятельством перечисленные выше функции ЭЦП могут быть реализованы на основе классических методов зашифрования/расшифрования:

* на основе симметричных систем (с тайным ключом),
* на основе симметричных систем и посредника,
* на основе асимметричных систем (с открытым ключом).

Первый из перечисленных методом ничем не отличается, например, от DES.

Во втором случае создаются по существу две симметричные системы: между отправителем и посредником и между посредником и получателем. Причем посредник выдает двум сторонам различный тайный (для иных субъектов системы) ключ.

Во последнем случае сообщение, отправляемое получателю, шифруется тайным ключом отправителя. Отправитель же верифицирует подпись (в данном случае – устанавливает авторство, используя для расшифрования публичный ключ отправителя, и получает гарантию в защищенности переданного сообщения от подделок, если после расшифрования формат и содержание документа имеют логическую стройность) с помощью открытого ключа отправителя. Таким образом, в этом случае, как и в первых двух случаях, ЭЦП, как отдельный, самостоятельный, присоединенный к исходному документу элемент получаемого сообщения, отсутствует. Кроме, того, в отличие от классической асимметричной криптографии, где используется ключевая информация получателя, в нашем случае используется ключевая информация отправителя: открытый ключ – для зашифрования, тайный – для расшифрования.

При использовании ЭЦП на основе RSA можно рассматривать две ситуации:

* сообщение Мo подписывается и передается в открытом (незашифрованном) виде;
* сообщение Мo подписывается и передается в зашифрованном.

При первом случае подпись S вычисляется на основе известного соотношения: S = (H(Mo))dо mod no при указанном выше реверсе в отношении ключевой информации; в (10.1) dо и no – элементы тайного ключа отправителя. Передаваемое сообщение М' = Мo||S. Соответственно, операция расшифрования на приемной стороне (получатель анализирует Мп||S) будет производиться в соответствии с известной модификацией ключей: H(Mo) = (S)ео mod no.

Далее вычисляется Н(Mп). Если H(Mo) = H(Mп), подпись верифицирована. Если подписываемое сообщение М (М') также должно передаваться в зашифрованном виде, то обычно М' шифруется на стороне отправителя стандартным образом: с помощью открытого ключа получателя (еп и nп), который перед основным процессом верификации подписи расшифровывает послание своим тайным ключом: dп и nп . Далее осуществляются вычисления и анализ, как и в первом случае.

Ключевая информация отправителя для ЭЦП по алгоритму Эль-Гамаля создается точно так же, как это описано в материалах к ЛР №8. Она состоит из тех же элементов, что и ключи в DSA. Основное отличие в применении расчетов состоит в том, что результатом зашифрования является только одна пара чисел, а не пара – для каждого блока исходного сообщения. Причем в рассматриваемом случае таким сообщением является хеш подписываемого документа: Н(Mо). Итак, ключевая информация отправителя: открытый ключ: y, g и р; тайный ключ: х. Чтобы подписать сообщение Мо, обладатель используемых для ЭЦП ключей должен выбрать, как и в предыдущей схеме, случайное число k, взаимно простое с р-1. Затем вычисляется числа а и b, являющиеся цифровой подписью (S = {a, b}): a = gk mod p, для вычисления b с помощью расширенного алгоритма Евклида решается уравнение Н(Mо) = (xa + kb) mod (p-1).

Получателю отправляется сообщение М' = Мо||S. Для верификации подписи вычисляется хеш полученного сообщения, Н(Мп) = h. Далее нужно убедиться, что выполняется равенство ya \* ab = gh (mod p). Если равенство выполняется, подпись верифицируется.

Алгоритм ЭЦП К. Шнорра (К. Schnorr ) является вариантом алгоритма ЭЦП Эль-Гамаля. Одной из особенностей ЭЦП Эль-Гамаля является то, что число p должно быть очень большим, чтобы сделать действительно трудной проблему дискретного логарифма. Рекомендуется длина p, по крайней мере, должна составлять 1024 бита. Чтобы уменьшить размер подписи, Шнорр предложил новую схему, но с уменьшенным размером подписи. Ключевая информация: p – простое число в диапазоне от 512 до 1024 бит; q –160-битное простое число, делитель p-1; любое число g (g≠1) такое, что gq=1 mod p. Числа p, g, q являются открытыми и могут применяться группой пользователей. Выбирается число х < q (х является тайным ключом) и вычисляется последний элемент открытого ключа: y = g-х mod p. Секретный ключ имеет длину не менее 160 бит. Для подписи сообщения Мо выбирается случайное число k (1<k<q) и вычисляет параметр а: а = gk mod p. Далее вычисляется хеш от конкатенации сообщения Мо и числа а: h = H(Mo||a). Обратим внимание, что хэш-функция непосредственно не применяется к сообщению. Создается хеш-образ подписываемого сообщения, спереди присоединенного к числу а. Далее вычисляется значение b: b = (k + xh ) mod q. Получателю отправляются М' = Мо||S; S = {h, b}. Для проверка подписи получатель вычисляет Х = gb yh (mod p).

Затем он проверяет выполнение равенства: h = Н(Mп||Х). Подпись достоверна, если равенство выполняется. Основные вычисления для генерации подписи могут производиться предварительно. Порядок величин х и h – около 140 двоичных разрядов, порядок числа k – около 70-72 разрядов. С учетом этого сложность операций умножения можно считать ничтожно малой по сравнению с модульным умножением в схеме RSA.

# Практическая часть

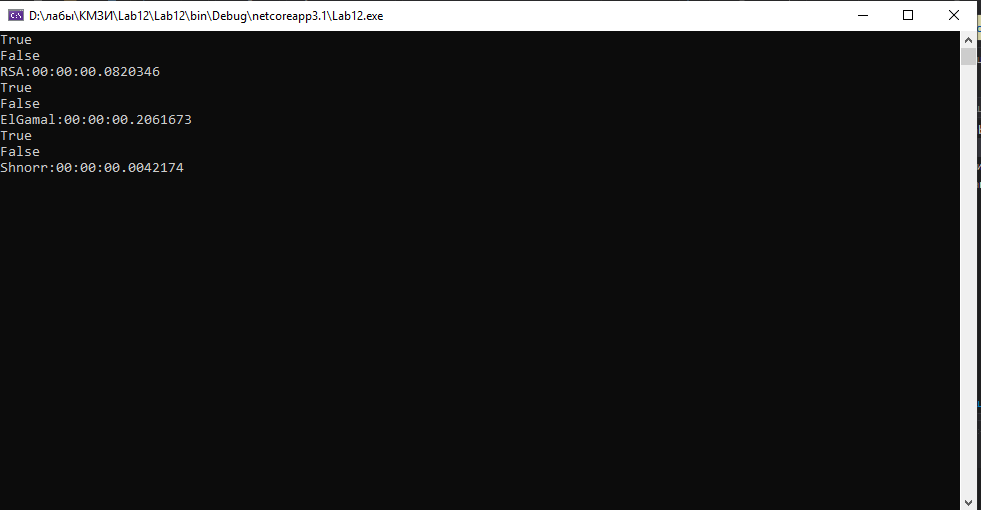
В результате выполнения данной лабораторной работы было создано приложение, выполняющее создание и верификацию ЭЦП с использованием алгоритмов RSA, Эль-Гамаль, и Шнорра. Результат работы данного приложения представлен на рисунке 1.

Рисунок 1 – Результат работы приложения

Функции, реализующие создание и верификацию ЭЦП на основе алгоритма RSA, представлены в листинге 1.

private static (long d, long n) Create(long p, long q, string sourceFilePathTextBox, string signFilePathTextBox)

{

var hash = File.ReadAllText(sourceFilePathTextBox).GetHashCode().ToString();

var n = p \* q;

var m = (p - 1) \* (q - 1);

var d = Calculate\_d(m);

var e = Calculate\_e(d, m);

var result = RSA\_Encode(hash, e, n);

var sw = new StreamWriter(signFilePathTextBox);

foreach (var item in result)

{

sw.WriteLine(item);

}

sw.Close();

return (d, n);

}

private static bool Verify(long d, long n, string signFilePathTextBox, string sourceFilePathTextBox)

{

var input = new List<string>();

var sr = new StreamReader(signFilePathTextBox);

while (!sr.EndOfStream)

{

input.Add(sr.ReadLine());

}

sr.Close();

var result = RSA\_Decode(input, d, n);

var hash = File.ReadAllText(sourceFilePathTextBox).GetHashCode().ToString();

if (result.Equals(hash))

{

return true;

}

else

{

return false;

}

}

Листинг 1 – Создание и верификация ЭЦП алгоритмом RSA

Функция, реализующая создание и верификацию ЭЦП на основе алгоримат Шнорра, представлена в листинге 2.

public static void Do()

{

BigInteger p = 29;

BigInteger q = 7;

string text = File.ReadAllText(".\\Test.txt");

BigInteger g = 16;

BigInteger obg = 20;

int x = 6;

BigInteger y = BigInteger.ModPow(obg, x, p);

BigInteger a = BigInteger.Pow(g, 3) % p;

BigInteger hash = (text + a).GetHashCode();

File.WriteAllText(".\\shnorr.txt", hash.ToString());

BigInteger b = (3 + 6 \* hash) % q;

BigInteger dov = BigInteger.ModPow(g, b, p);

BigInteger X = dov \* BigInteger.ModPow(y, hash, p);

BigInteger hash2 = (text + X).GetHashCode();

var f = hash == hash2;

Console.WriteLine(true);

string text2 = File.ReadAllText(".\\FakeTest.txt");

BigInteger hash3 = (text2 + X).GetHashCode();

var f2 = hash == hash3;

Console.WriteLine(f2);

}

Листинг 2 – Создание и верификация ЭЦП алгоритмом Шнорра

Функции, реализующие генерацию и верификацию ЭЦП с использованием алгоритма Эль-Гамаля, приведены в листинге 3.

public static void CheckCorrectly()

{

var str = "Hello world";

var hash = CalculateMd5Hash(str).ToString();

var sign = ElGamalClass.EnCrypt(hash);

var verify = ElGamalClass.DeCrypt(sign) == CalculateMd5Hash(str).ToString();

Console.WriteLine(verify);

}

public static void CheckInCorrectly()

{

var str = "Hello world";

var fakeStr = "Helloworld";

var hash = CalculateMd5Hash(str).ToString();

var sign = ElGamalClass.EnCrypt(hash);

var verify = ElGamalClass.DeCrypt(sign) == CalculateMd5Hash(fakeStr).ToString();

Console.WriteLine(verify);

}

public static BigInteger CalculateMd5Hash(string input)

{

var md5 = MD5.Create();

var inputBytes = Encoding.ASCII.GetBytes(input);

var hash = md5.ComputeHash(inputBytes);

return new BigInteger(hash.Concat(new byte[] { 0 }).ToArray());

}

Листинг 3 – Создание и верификация ЭЦП алгоритмом Эль-Гамаля

# Вывод

В результате выполнения данной лабораторной работы было изучено понятие электронно-цифровой подписи, изучена её классификация и назначение, также были рассмотрены различные способы генерации ЭЦП, и создано приложение, реализующее их.