МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образование «Белорусский государственный технологический университет»

Кафедра информационных систем и технологий

**«Исследование криптографических алгоритмов на основе эллиптических кривых. Задание №3.»**

Студент:

Септилко Анастасия Антоновна

Преподаватель:

Блинова Евгения Александровна

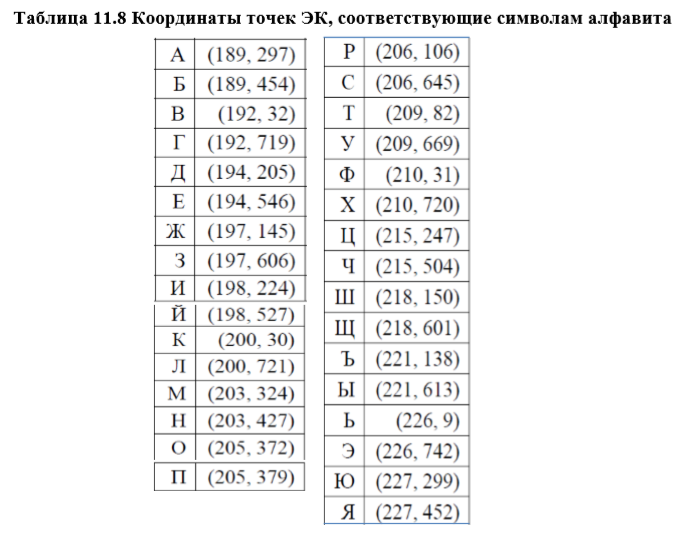
Минск 2020

**ЗАДАНИЕ №3:**

3.1. Создать оконное приложение для генерации/верификации ЭЦП на основе алгоритма ЕСDSA: ЭК Е751(–1, 1) c генерирующей точкой G = (416, 55); порядок точки q = 13. Тайный ключ – в соответствии с вариантом из табл. 11.7. Тайный ключ – в соответствии с табл. 11.9.

3.2. Вычислить самостоятельно значение открытого ключа, Q. При этом следует воспользоваться основной формулой (11.8), а также соотношениями (11.3)-(11.5) для случая P = Q; не следует также забывать, что все вычисления производятся по mod 751; см. также пример 5 (вычисление 2Р) и пример 7. Параметры k – по собственному усмотрению.

3.3. Хешем подписываемого сообщения, (Н(М)), является модуль по основанию 13 координаты х точки ЭК, соответствующей первому символу собственной фамилии, из табл. 11.8. Например, фамилия начинается на букву «Я»: х = 227, тогда 227 mod 13 = 6, значит в данном конкретном случае Н(М) = 6.





Рассмотрим генерацию и верификацию ЭЦП на основе алгоритма DSA и ЭК (EC) – ЕСDSA. Обращаем внимание на то, что используется ключевая информация отправителя (стороны А). Генерация ключей происходит так же, как и в последнем примере. Однако в анализируемом здесь случае во внимание должен приниматься еще один известный параметр ЭК: порядок точки G, т. е. число q.

Краткая характеристика алгоритма генерации и верификации ЭЦП. Полагаем, что отправитель подписывает хеш Н(М) сообщения М.

Генерация ЭЦП.

1. Выбрать число k (1 < k < q), q – порядок точки G.

2. Вычислить точку kG = (х, у), вычислить r = x mod q; при r = 0 изменить k и повторить шаг 2.

3. Вычислить t = k-1mod q (например, на основе расширенного алгоритма Евклида).

4. Вычислить s = (t (H(M) + dr)) mod q; при s = 0 изменить k и повторить алгоритм.

Стороне В отсылаются сообщение М и ЭЦП (числа r и s).

Верификация ЭЦП.

Получатель знает алгоритм хеширования, который использовался отправителем, открытый ключ отправителя, с помощью чего выполняет следующие операции над М и полученной ЭЦП (обозначения чисел оставим без изменений).

1. Проверить выполнение условия: 1 < r, s < q; если условие не выполняется, то легитимность подписи не подтверждается, в противном случае – выполняются дальнейшие шаги.

2. Вычисляются Н(М) и w = s–1 mod q.

3. Вычисляются u1 = w Н(М) (mod q), u2 = wr (mod q).

4. Вычисляются Gu1 + Qu2 = (x', y'), v = x' mod q.

5. Сравниваются v и r; если равенство выполняется, подтверждается легитимность подписи и целостность полученного сообщения.

В данной части лабораторной работы было необходимо создать приложение, реализующее генерацию и верификацию ЭЦП на основе эллиптических кривых.

Функция, реализующая генерацию и верификацию ЭЦП представлена на в листинге 1.1.

Листинг 1.1 – Функция WithString()

private void WithString(string str)

{

int hashstr = 0;

int outval;

int q = 13;

foreach (char s in str)

{

hash.TryGetValue(s, out outval);

hashstr += outval % q;

}

int d = 10;

string[] numbersQ = Multiply(d, 416, 55).Split(',');

string[] numbersG = Multiply(6, 416, 55).Split(',');

int x = int.Parse(numbersG[0]);

int r = mod(x, q) + 4;

int t = 11 % q;

int sign = mod((t \* (hashstr + d \* r)), q);

Sign.Text = r.ToString() + ',' + sign.ToString();

int w = mod(Foo(sign, q), q);

int u1 = mod((w \* hashstr), q);

int u2 = mod((w \* r), q);

string[] Expr1 = { "", "" };

string[] Expr2 = { "", "" };

Expr1 = Multiply(u1, int.Parse(numbersG[0]), int.Parse(numbersG[1])).Split(',');

Expr2 = Multiply(u2, int.Parse(numbersQ[0]), int.Parse(numbersQ[1])).Split(',');

string[] result;

result = SumTwoPoints(int.Parse(Expr1[0]), int.Parse(Expr2[0]), int.Parse(Expr1[1]), int.Parse(Expr2[1])).Split(',');

int v = mod(int.Parse(result[0]), q);

bool f = false;

if ( v != r)

{

f = true;

Verify.Text = f.ToString();

}

else

{

Verify.Text = f.ToString();

} }

Было необходимо разработать функцию, которая отвечает за сумму двух точек. Функция SumTwoPoints() отвечает за это действие, пример ее продемонстрирован в листинге 1.2.

Листинг 1.2 – Функция SumTwoPoints()

string SumTwoPoints(int xP, int xQ, int yP, int yQ)

{

BigInteger lyambda;

int raznX = xQ - xP;

int raznY = yQ - yP;

if (raznX < 0)

{

raznX += p;

}

if (raznY < 0)

{raznY += p;}

if (xP == 0 & yP == 0)

{

return xQ.ToString() + ',' + yQ;

}

if (xQ == 0 & yQ == 0)

{

return xP.ToString() + ',' + yP;

}

BigInteger xR = 0, yR = 0;

if (xP == xQ && yP != yQ || (yP == 0 && yQ == 0 && xP == xQ))

{ }

else

{

if (xP == xQ && yP == yQ)

{

lyambda = (3 \* BigInteger.Pow(xP, 2) - 1) \* (Foo(2 \* yP, p));

}

else

{

lyambda = (raznY) \* Foo(raznX, p);

}

xR = (BigInteger.Pow(lyambda, 2) - xP - xQ);

yR = yP + lyambda \* (xR - xP);

xR = xR % p < 0 ? (xR % p) + p : xR % p;

yR = -yR % p < 0 ? (-yR % p) + p : (-yR % p);

}

string Result = xR.ToString() + ',' + yR.ToString();

return Result;

}

Результат выполнения программы показан на рисунке 1.1.

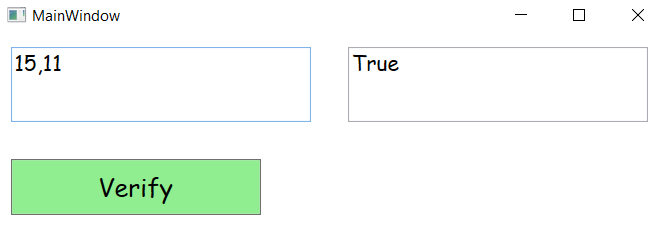


Рисунок 1.1 – Итог программы

**Вывод**

В данной части лабораторной работы было изучено создание и верификация ЭЦП на основе эллиптический кривых и создано приложение, реализующее эти функции.