15th International Research/Expert Conference
"Trends in the Development of Machinery and Associated Technology"
TMT 2011, Prague, Czech Republic, 12-18 September 2011

ELLIPTIC CURVES AND THEIR APPLICATIONS TO CRYPTOGRAPHY

Mr. sc. Hermina Alajbegović
Mr. sc. Almir Huskanović
Phd. sc. Dževad Zečić
University in Zenica
Zenica
Bosnia and Herzegovina

ABSTRACT
This paper presents elliptic curves and their application in public-key cryptography, using software Mathematica Wolphram Research. Elliptic curve over field K is the set of solutions $(x, y) \in K$ *which satisfy an equation of the form* $y^2 = x^3 + ax + b$ *where is* $\Delta(E) = -16(4a^3 + 27b^2) \neq 0$ *together with point O which is called the point of infinity. There are so many other generated definitions. We can define addition of two points in an elliptic curve, so that elliptic curve is additive group with neutral element O. In cryptography we consider elliptic curves over finite fields. These curves can be used for encrypting and decrypting messages and for digital signature. Elliptic curves are also used in several integer factorization algorithms that have applications in cryptography.*
Keywords: elliptic curve, finite fields, public – key cryptography, digital signature, elliptic curve discrete logarithm problem

1. INTRODUCTION

The digital signature is a pair of large numbers that is computed using a set of rules (i.e., algorithms for digital signature) and a set of parameters such that authentication and integrity of the data can be verified. The DSA consists of two parts: first part is for generate digital signature and second part is for verify digital signature. In algorithm for generate digital signature we use private key, and in algorithm for verify digital signature we use public key, and it is different from the private key. Each user possesses a private and public key pair. Public keys are assumed to be known to the public in general. The signatory only knows private keys. Anyone can verify the signature of a user by employing that user's public key.

The Elliptic Curve Digital Signature Algorithm (ECDSA) is analogue of the Digital Signature Algorithm (DSA). Security of DSA depends on the intractability of the discrete logarithm problem: Find x, $0 \le x \le p-2$ such that $\alpha^x \equiv \beta \pmod p$, where is given (known) a prime p, a generator α of Z_p, and an element β of Z_p. The security of modern elliptic curve cryptography depends on the intractability of determining l from $Q = lP$ given known values of Q and P. It is known as the elliptic curve discrete logarithm problem.

A *finite field F* consists of a finite set of elements together with two binary operations on F, that satisfy certain arithmetic properties. The *order* of a finite field is the number of elements in the field. If p is a prime number, then set $F_p = \{1, 2, \cdots, p-1\}$ is field with operation addition and multiplication defined by remainder divided by p of sum and product these elements in Z (set of integers).

Let $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ be points on th elliptic curve $y^2 = x^3 + ax + b$ then

$$R = (x3, y3) = P + Q$$

on elliptic curve may be computed by

$$P + Q = \begin{cases} O, if\ x_1 = x_2\ and\ y_1 = -y_2 \\ (x_3, y_3), otherwise \end{cases}$$

where

$$(x_3, y_3) = \left(\lambda^2 - x_1 - x_2, \lambda(x_1 - x_3) - y_1\right)$$

and

$$\lambda = \begin{cases} \dfrac{3x_1^2 + a}{2y_1}, if\ P = Q \\ \dfrac{y_2 - y_1}{x_2 - x_1}, otherwise \end{cases}$$

There are many algorithms for solving the ECDLP. Naive exhaustive search is method where one simply computes successive multiples of $P : P, 2P, 3P, \ldots$ { until is Q obtained. This method can take up to n steps in the worst case, where n is the order of the point P. Let $|E(a,b)/p|$ denote the number of points on elliptic curve $E(a,b)$ over F_p. Then $|E(a,b)/p| = p + 1 + \sum_{x \in F_p} \left(\dfrac{x^3 + ax + b}{p}\right)$. If $|E(a,b)/F_p|$ denote the order of $E(a,b)/F_p$, then $|E(a,b)/F_p| \in (p + 1 - 2\sqrt{p}, p + 1 + 2\sqrt{p})$. In algorithm for ECDSA we need point G on elliptic curve with order n. We can choose a point P on $E(a,b)$, find all multiples nP for $n \in (p + 1 - 2\sqrt{p}, p + 1 + 2\sqrt{p})$, stopping once we find an n such that $nP = O$, where $O =$ ponit at infinity.

This method requires $O(\sqrt{p} \log p)$ field operations and it returns a multiple of the order of P.

2. ALGORITHM

In this paper we'll describe algorithm for digital signature which use elliptic curve. Suppose that person A wants to send a signed message to person B. Both sides have to know the curve parameters wiche is used in algorithm for ECDSA. There are eight parameters: (p,a,b,G,n,Q). p is the field size, a and b are two field elements that define the equation of the curve; G is a base point of prime order on the curve ; n is the order of the point G. Person who signing message must have a key pair suitable for elliptic curve cryptography, consisting of a private key x (a randomly selected integer in the interval $[1, n-1]$) and a public key Q (where $Q = xG$).

Algoritm for signing message:

Calculate $e = \text{HASH}(m)$, where HASH is a cryptographic hash function.

1. Select a random integer k from $[1, n-1]$.
2. Calculate $r = x_1 \pmod n$, where $(x_1, y_1) = kG$. If $r = 0$, go back to step 2.
3. Calculate $s = k^{-1}(e + rx) \pmod n$. If $s = 0$, go back to step 2.
4. The signature is the pair (r,s).

It is crucial to select different k for different signatures, otherwise the equation in step 4 can be solved for d, the private key.

Signature verification algorithm

If someone wants to verificate signature, he must have a copy of public key Q. If he does not trust the source of Q, he needs to validate the key (O here indicates the identity element):

1. Check that Q is not equal to O and its coordinates are otherwise valid
2. Check that Q lies on the curve
3. Check that $nQ = O$

Algoritm for verification digital signature of message:

1. Verify that r and s are integers in $[1, n-1]$. If not, the signature is invalid.
2. Calculate $e = HASH(m)$, where HASH is the same function used in the signature generation.
3. Calculate $w = s^{-1} \pmod n$.
4. Calculate $u_1 = zw \pmod n$ and $u_2 = rw \pmod n$.
5. Calculate $(x_1, y_1) = u_1 G + u_2 Q$.
6. The signature is valid if $r = x_1 \pmod n$, invalid otherwise.

3. IN MATHEMATICA 4.0

We will give simple code in Mathematica wich comutes add and multiplication points on an elliptic curve form $y^2 = x^3 + ax + b$ over finite field F_p.

```
ellipticAdd[{{x1_, y1_}, {x2_, y2_}, p_}, {a_, b_}] :=
 Module[{},
  If[x1 ≠ Infinity && x2 ≠ Infinity,
      If[Mod[x1, p] == Mod[x2, p] && Mod[y1, p] ≠ Mod[-y2, p] ,
      s = Mod[3 * x1 * x1 + a, p] * PowerMod[2 * y1, -1, p]];
    If[Mod[x1, p] ≠ Mod[x2, p] , s = Mod[(y2 - y1), p] * PowerMod[(x2 - x1), -1, p]];
    x3 = Mod[s^2 - x1 - x2, p]; y3 = Mod[-y1 + s (x1 - x3), p]];
If[x1 == Infinity && y1 == Infinity , x3 = x2 ; y3 = y2];
 If[y2 == Infinity && x2 == Infinity , x3 = x1 ; y3 = y1];
 If[Mod[x1, p] == Mod[x2, p] && Mod[y1, p] == Mod[-y2 , p], x3 = Infinity ; y3 = Infinity];
 {x3, y3}]
ellipticMultiplicationSkalar[{{x1_, y1_}, k_, p_}, {a_, b_}] :=
 Module[{}, P = {x1, y1}; bin = IntegerDigits[k, 2]; beta1 = Length[bin];
  CC = P;
  For[i = 2, i ≤ beta1, i++,
   CC = ellipticAdd[{CC, CC, p}, {a, b}];
   If[bin[[i]] == 1 , CC = ellipticAdd[{CC, P, p}, {a, b}], CC = CC]];
  Print["C", CC]]
```

We choose concrete finite field, elliptic curve and message digest (Hm) for testing. Let set elliptic curve $y^2 = x^3 - 3x + 69424$ over finite field F_{114973}, point on elliptic curve $G = (11570, 42257)$. Order of G is 11467. Let's choose private key: $x = 86109$. Public key is p=114973, $a = -3, b = 69424, G = (11570, 42257), n = 114973$, $Q = xG = (6345, 28549)$ and algorithm for calculating message digest have to be known. We'll set message digest for some message $Hm = 1789679805$.

```
ECDSA[p_, a_, b_, G_, n_, Hm_, x_] := Module[{}, s = 0; r = 0;
  While[s == 0 || r == 0, k = Random[Integer, {1, n - 1}];
   kG = ellipticMultiplicationSkalar[{G, k, p}, {a, b}];
   r = Mod[kG[[1]], n];
   s = Mod[PowerMod[k, -1, n] * (Hm + x * r), n]]; Print["Digital signature is ", {r, s}]; r; s;]
In[165]:= ECDSA[114973, -3, 69424, {11570, 42257}, 114467, 1789679805, 86109]

      Digital signature is {40719, 109982}
Q = ellipticMultiplicationSkalar[{{11570, 42257}, 86109, 114973}, {-3, 69424}]

{6345, 28549}
In[156]:= VerECDSA[p_, a_, b_, G_, n_, Hm_, Q_, r_, s_] := Module[{}, w = PowerMod[s, -1, n];
          u1 = Mod[w * Hm, n];
          u2 = Mod[r * PowerMod[s, -1, n], n];
          A = ellipticMultiplicationSkalar[{G, u1, p}, {a, b}];
          B = ellipticMultiplicationSkalar[{Q, u2, p}, {a, b}];
          ver = ellipticAdd[{A, B, p}, {a, b}];
          If[ver[[1]] == r, "Signature is valid", "Signature is not valid"]]

In[166]:= VerECDSA[114973, -3, 69424, {11570, 42257}, 114467, 1789679805, Q, r, s]

Out[166]= Signature is valid
```

4. CONCLUSION

The elliptic curve digital signature algorithm is simply obtain from the DSA by replacing the subgroup of order q of group Z_p^* generated by g with the subgroup of points on an elliptic curve that are generated by G. The elliptic curve discrete logarithm problem is harder than the discrete logarithm problem, the strength –per key-bit is substantially greater in elliptic curve systems than in conventional discrete logarithm systems. Smaller parameters can be used in elliptic curve cryptosystem than with discrete logarithm systems but with equivalent levels of security. The advantage that can be obtained from smaller parameters include speed and smaller keys and certificates**.** A 160-bit key in ECC is considered to be as secured as 1024-bit key in DSA.

5. REFERENCE

[1] Darrel Hankerson, Alfred Menezes, Scott Vanstone, Guide to Elliptic Curve Cryptography, 2004 Springer-Verlag New York, Inc.

[2] Song Y.Yan , Primality Testing and Integer Factorization in Public-Key Cryptography , Springer Science+Business Media, LLC., 2009

[3] Web page www.FIPS 186 –(DSS), Digital Signature Standard

[4] Web page www.itl.nist.gov/**fips**pubs/fip180-1.htm, Secure Hash Standard

[5] Elliptic Curve Cryptography - An Implementation Tutorial, Anoop MS, TATA ELXSI LTD, India anoopms@tataelxsi.co.in.

[6] Hung-Zih Liao, Yuan- Yuan Shen. On the Elliptic**.** Curve Digital Signature Algorithm, Tunghai Science, 2006.

[7] Cryptographic Pairings: Efficiency and DLP Security**.** Naomi Benger, PhD thesis, Dublin City University, 2010