International Conference on Instrumentation, Communication and Information Technology (ICICI) 2005
Proc., August 3rd -5th , 2005, Bandung, Indonesia

IMPLEMENTATION OF ElGamal ELLIPTIC CURVE CRYPTOGRAPHY USING MATLAB

Wan Khudri [1)], Sutanto [2)]

1) Jurusan Matematika, Universitas Sebelas Maret Surakarta
Jl. Ir. Sutami 36A, Kentingan, Surakarta 57126
e-mail: chudry81@yahoo.com , one@oiry.net
2) Jurusan Matematika, Universitas Sebelas Maret Surakarta
Jl. Ir. Sutami 36A, Kentingan, Surakarta 57126
e-mail: sutanto@uns.ac.id

ABSTRACT

ElGamal Elliptic Curve Cryptography(ECC) is a public key cryptography analogue of the ElGamal encryption schemes which is used Elliptic Curve Discrete Logarithm Problem (ECDLP). The software which is used to implement ElGamal ECC is MATLAB. This implementation consist of 3 main programme, they are Key Generation, Encryiption and Decryption ElGamal ECC.To reach the goal of the implementation, some *functions* which are able to construct the 3 main programme are needed. Some *functions* are available in MATLAB and 31 *functions* are made by the writer himself. Those *functions* are classified into 3 categories, they are modular arithmetic *function* (7 *functions*), elliptic curve arithmetic *function* (5 *functions*) and ElGamal ECC *function* (19 *functions*).

The modular artihmetic *function* is used in addition operation, representation of NAF (Non Adjacent Form), multiplication operation, invers, division, power, and square root in modular arithmetic opereration.

The elliptic curve arithmetic *function* is used in addition operation, elliptic curve equation, invers under addition, subtraction, and elliptic curve scalar multiplication.

The ElGamal *function* is used in biner-decimal conversion, decimal-biner conversion in '*n*' bit format, to find lower and upper bound of key length, to generate prime number, to generate coefficient of elliptic curve equation, to find the order of the elliptic group, to generate one elliptic curve point, to calculate the order of point, to generate base point and its order, elliptic curve domain parameters, to generate private key and public key, to represent plaintext into number, number into point, point into number, number into plaintext, encryption of ElGamal ECC for one point, and decryption of ElGamal ECC for one chipertext of point.

Key Words: ElGamal, elliptic curve, cryptography, field, public key

I. INTRODUCTION

The idea of using elliptic curve in cryptography was introduced by Victor Miller [10] and Neal Koblitz as an alternative to established public key systems such as DSA and RSA. In 1985, they proposed a public key cryptosystems analogue of the ElGamal encryption schemes which used Elliptic Curve Discrete Logarithm Problem (ECDLP) [5,6,11]. According to Purbo and Wahyudi [11], Elliptic Curve Cryptography (ECC) can be used in some necessities such as encryption scheme (ElGamal ECC/EC ElGamal), digital signature (ECDSA) and key exchange protocol (Diffie Hellman ECC/EC Diffie Hellman).

II. Implementation of ElGamal ECC

In this paper, we just implement elliptic curve over Prime Finite Field for our application programme on Matlab.

2.1. *Finite Field*

There are two types of finite fields usually used in the cryptographic applications, they are Prime Finite Field (F_p) and Binary (Characteristic 2) Finite Filed (F_{2^m}). It is necessary to describe these fields concretely in order to precisely specify cryptographic schemes based on ECC (see [1,p.3-6], [6, p.13-15], [5,p.2-8]).

2.2. *Elliptic Curve over F_p*

In this paper, we only present the elliptic curve over F_p. An elliptic curve over F_p is defined by parameters $A,B \in F_p$ consists of the set of points (x,y) for $x,y \in F_p$ to the equation:

$$y^2 = x^3 + Ax + B \pmod{p} \qquad (1)$$

and also include with an extra point ***O*** (*point at infinity* / identity element under addition), so that $A,B \in F_p$ satisfy $4A^3+27B^2 \neq 0$ and p is odd prime number. The equation (1) is called the defining equation of elliptic curve over F_p [1,6,12,13]. An elliptic group over F_p is obtained by computing the equation (1) for each value of x, $0 \le x \le p\text{-}1$. If y^2 is *quadratic residue* modulo p (see [Stinson,p.313]) then the point is in the elliptic group over F_p [4], denoted by $E_p(A,B)$. It can be shown that $E_p(A,B)$ forms an abelian group under addition(see [6,p.18]). The addition rule in $E_p(A,B)$, see [1,4,6,12]. Three methods (geometric, number theoritic and algebric) for deriving elliptic curve was presented by Jason Hill [7].

```
T=eccparameter(nkunci)
1.  [bb ba] ← ecckunci(nkunci);
2.  pi ← 1;
3.  p ← eccprima(bb,ba);
4.  ABi ← 1;
5.  [A B]← ecckurv(p);
6.  N_E ← eccordgrup(p,A,B);
7.  NGi ← 1;
8.  [N_G G_E] ← eccbasic(p,A,B,N_E);
9.  IF (N_G<N_E)
        NGi ← NGi+1;
        GOTO STEP 8;
    ENDIF
10. IF (NGi>N_E)
        ABi←ABi+1;
        GOTOSTEP 5;
    ENDIF
11. IF ( ABi > (p-1)*(p-1) )
        pi ← pi+1; GOTO STEP 3;
    ENDIF
12. IF (pi > (ba-bb) )
        nkunci ← input key length;
        GOTO STEP 1;
    ENDIF
13. h ← N_E / N_G ;
14. T ← [p,A,B, G,N_G,h];
15. RETURN T ;
```

Alg 1. *Function* to generate elliptic curve domain parameters

2.3. *Elliptic Curve Domain Parameters over F_p*

The elliptic curve domain parameters

over F_p are defined by the sextuple T.

$T = (p,A,B,G_E , N_G , h)$

(see [2,p.3], [3,p.11], [5,p.12], [6,p.22])

The *function* algorithm to generate elliptic curve domain parameters, see Alg 1.

2.4. ElGamal ECC Algorithm

ElGamal ECC is a public key cryptography which used ECDLP and analogue of the generalized ElGamal encryption schemes. For more information about the generalized ElGamal Encryption, see Menezes *et.al* [9,p.294-298].

Generally, we have 2 entities in the process of encryption and decryption. The one at the encryption side (Let us assume: Alice) and the other at the decryption side (Let us assume: Bob). So that the ElGamal ECC algorithm with elliptic curve domain parameters $T= (p,A,B,G_E , N_G , h)$ can be defined as follows:

1. *Key generation*
   a. Select a random number V, $V\in[1, N_G -1]$.
   b. Compute $\beta = V.\ G_E$.
   c. V is *private key* and β is *public key*.
2. *Representation of message into elliptic curve point*
   The method proposed by Koblitz (see [3] and [14]). Let us assume that s_j is an integer in F_p and $s_j = x^3 +Ax +B \pmod p$. The probability is ½ that $s_j = x^3 +Ax +B \pmod p$ is a square mod p. Let k is an integer so that a failure rate $1/2^r$ is acceptable when trying to encode a message as a point. The method can be defined as follows:
   a. Express the message as number m, $m \geq 0$, such that $m.r < p$.
   b. Assume $x_j = m.r + j$, *for* $j\in[0,\varepsilon-1]$ and compute $s_j = x_j^3 + Ax_j + B$ until we get $s_j^{(p-1)/2} = 1 \pmod p$.
   c. Compute the square root modulo p (see [8,p.128-129]) of s_j as y_j.
   d. The point $P_M = (x_j , y_j)$ is representing of the message.
3. *Encryption*
   a. Alice get Bob's public key (β).
   b. Select a random number k, $k\in[1, N_G -1]$.
   c. Compute $P_1 = k.G_E$ dan $P_2 = P_M + k\ \beta$.
   d. $P_C = (P_1, P_2)$ is the chipertext pair of points.
4. *Decryption*
   a. Compute $M_1 = V.P_1$, which V is Bob's private key.
   b. Compute $P_M = P_2 – M_1$.
   c. The point P_M is the representation of message.
5. *Representation of elliptic curve point into message*
   a. Compute $m = \left\lfloor \frac{x_j}{r} \right\rfloor$.
   b. Convert m into message.

2.5. Implementation on Matlab

The most time consuming part of the computation is elliptic scalar multiplication. So we need to represent the multiplication algorithm (see [5] and [6]) to make the computation more efficient, such as binary algorithm, addition-subtraction algorithm and repeated-dobling algorithm. We use the addition-subtraction algorithm in our implementation and Non Adjacent Form (NAF) to represent the scalar. For computing the elliptic scalar multiplication k.P, k is represented by NAF form.

$k = \sum_{j=0}^{l-1} u_j 2^j$, where $u_j \in \{-1,0,1\}$.

And the result of NAF will be used in the addition-subtractiion algorithm.

There are 3 main programmes which will be described in our implementation, they are:
1. Key Generation Programme.
2. Encryption Programme
3. Decryption Programme.

There are many *functions* must be made to build the main programmes which are classified into 3 categories, they are
1) *Functon* of Modular Arithmetics
2) *Function* of Elliptic Curve Arithmatics
3) *Function* of ElGamal ECC

2.5.1. Key Generation Programme

This programme need key length as input. If the operator need to generate new elliptic curve domain parameters, then programme will call *function* 'eccparameter' (return the elliptic curve domain parameters) . If not, then this programme will ask to the operator to input that paramaters .Hereinafter, it will call *function* 'eccprivkey' (return private key) and 'eccpubkey' (return public key). There are 2 final output of key generation programme. They are private key and public key.

```
Genkunci.m
1.  nkunci ← input key length;
2.  GenT ← Generate parameters ?
3.  IF ((GenT='Y') | (GenT='y'))
          T ← eccparameter(nkunci)
    END
4.  IF((GenT='N') | (GenT='n'))
          T ← input the elliptic curve
                domain parameters
                ( $p,A,B,G_E , N_G , h$ );
    END
5.  V ← eccprivkey(nkunci,T{5});
6.  PB←eccpubkey(T{1},T{2},V, T{4});
7.  RETURN T,V,PB;
```

Alg 2. Key Generation Programme

```
Enkripsi.m
1.  T ← input domain parameters;
2.  r ← input the number of
          producing points;
3.  PB ← input public key;
4.  pesan←input message(plaintext);
5.  IF (isnumeric(pesan) = 1)
          pesan ← num2str(pesan);
    END
6.  lpesan ← length(pesan);
7.  nkunci ← length(des2bin(p));
8.  bpesan ← $\left\lceil \frac{\text{nkunci}}{8} - 1 \right\rceil$;
9.  ipesan ← $\left\lceil \frac{\text{lpesan}}{\text{bpesan}} \right\rceil$;
10. akhir ← 0;
11. IF (lpesan>bpesan)
      FOR ips=1:1:ipesan
        IF (ips<ipesan)
          awal ← akhir+1;
          akhir ← ips * bpesan;
          plain ← pesan(awal:akhir);
        ELSE
          plain←pesan(akhir+1:lpesan);
        END
        p2n ← eccplain2num(plain);
        PM← eccnum2titik(T,p2n,r);
          n2t{ips} ← PM;
      END
      FOR (nti=1:1:length(n2t)
       PC{nti}←ccenk(T,PB,n2t{nti});
      END
    END
12. IF (lpesan<=bpesan)
          p2n ← eccplain2num(pesan);
          n2t ← eccnum2titik(T,p2n,r);
          PC ← eccenk(T,PB,n2t);
    END
13. RETURN PC;
```

Alg 3. Encryption of ElGamal ECC

2.5.2. Encryption Programme

This programme will need the elliptic curve domain prameters and public key as input and also the number of producing point. Then the programme will ask to the operator to input the message that will be encrypted. The message will be divided for every $\left\lceil \frac{nkunci}{8} - 1 \right\rceil$ characters and represented into points by the *function* 'eccplain2num' and 'eccnum2titik'. All pairs of message will be encrypted by *function* 'eccenk'. So we get the chipertext pair of points from this programme, see Alg 3.

2.5.3. Decryption Programme

Input of this programme are elliptic curve domain parameters, private key and number of producing points. Chipertext pair of points will be decrypted by *function* 'eccdek' and then the result will be encoded into original message (plaintext) by *function* 'ecctitik2num' and 'eccnum2plain'. So the message can be understood by the receiver, see Alg 4.

```
Dekripsi.m
1.  T ← input domain parameters;
2.  V ← input private key;
3.  r ← input the number of producing
        points;
4.  PC ← input chipertext pair of
        points;
5.  PCs ← size(PC);
6.  lps ← 1;
7.  FOR dek:1:1:PCs(1)
     PM ← eccdek(T,V,PC(dek , : );
     t2n ← ecctitik2num(PM,e);
     n2p ← eccnum2plain(t2n);
     pesan(lps:lps+length(char(n2p)-1))
           ← char (n2p);
      lps ← length(pesan);
    END
8.  RETURN pesan;
```

Alg 4. Decryption of ElGamal ECC

For example the result of running programmes with 32 bit key length are given in Table 1 and Table 2

III. Conclusion

Based on the result of the implementation programme of ElGamal ECC, if 'nkunci' is the key length, then the message (plaintext) will be divided for every $\left\lceil \frac{nkunci}{8} - 1 \right\rceil$ characters. For each pairs of the message will be represented into elliptic curve point (P_M) and then will be encrypted using the public key (β) receiver with the computation as follows

$P_1 = k.G_E$

$P_2 = P_M + k.\ \beta$

$P_C = (P_1\ , P_2) = (k.G_E\ , P_M + k.\ \beta\ ).$

where the elliptic curve domain parameters are $(p,A,B,G_E\ , N_G\ , h)$ and k is integer. The result is chipertext pair of points (P_C). For decrypting the chipertext pair of point will be needed the private key (V). The computation of decrypting chipertext in the ElGamal ECC can be described as follows

$$M_1 = V.P_1$$
$$P_M = P_2 - M_1.$$

So we get the representation of point (P_M). the point P_M must be converted into original message. In this implementation we used the combination of binary-decimal. For more information about source code,algorithm, *functions* or the others, contact us.

INPUT	
Elliptic Curve Domain Parameters (*T*)	
p	3946183951
A	537680305
B	1059676324
G_E	[1152222263 , 3133703258]
N_G	3946206427
r	100
β	[3539395206 , 1802765602]
Plaintext	Intel Pentium 2.4 GHz
OUTPUT	
2721233492 2106701877 2390822029 1608866709 721005031 972268584 3482646965 3008581623 1496398035 2796975696 1599552902 1715898382 1750258559 1031174283 2529634365 1612132131 2155963162 32053424 2125957278 3328929268 2939262549 801116234 3527493980 2490239375 980655110 3741116511 3149813578 775953281	

Table 1. Output of Encryption Programme

INPUT	
Elliptic Curve Domain Parameters (*T*)	
p	3946183951
A	537680305
B	1059676324
G_E	[1152222263 , 3133703258]
N_G	3946206427
r	100
V	2759936539
Chipertext	
2721233492 2106701877 2390822029	
1608866709 721005031 972268584	
3482646965 3008581623 1496398035	
2796975696 1599552902 1715898382	
1750258559 1031174283 2529634365	
1612132131 2155963162 32053424	
2125957278 3328929268 2939262549	
801116234 3527493980 2490239375	
980655110 3741116511 3149813578	
775953281	
OUTPUT	
Intel Pentium 2.4 GHz	

Table 2. Output of Decryption Programme

REFERENCES

[1] Certicom, *SEC 1:Elliptic Curve Cryptography*, http://www.secg.org/collateral/sec1_final.pdf, 2000.

[2] Certicom , *S EC 2:Recommended Elliptic Curve Domain Parameter*,http://www.secg.org/collateral/sec2_final.pdf, 2000.

[3] Cheng, Z., *Simple Tutorial on Elliptic Curve Cryptosystem*, ver. 0.1, m.z.cheng@mdx.ac.uk, School of Computing Science, June 2003.

[4] Chouinard, J.Y.,*Design of Secure Computer Systems CSI4138/CEG 4394, Notes on Elliptic Curve Cryptography* (ECC), http://www.csi.uottawa.ca/~chouinar/Handout_CSI4138_ECC_2002.pdf, 2002.

[5] Dahab,R., and J.Lopez, *An Overview of Elliptic Curve Cryptography*, Institute of Computing State University of Campinas Brazil, Brazil, May 2000.

[6] Doraiswamy,S.,Z.Jainullabudeen and V.V. Kumar,*CSE450/598, Design and Analysis of Algorithms, Project ID : P113, Elliptic Curve Cryptography*, Final Report**,** http://www.eas.asu.edu/~cse450sp/projects/final_P113.doc.

[7] Hill, J., *Introduction to Elliptic Curve Cryptography: A Mathematical Overview*. http://www.geocities.com/khadimir/Introduction_to_Elliptic_Curve_Cryptography.doc. 29 Oct 1001.

[8] Koblitz, N., *Algebric Aspects of Cryptography : Algorithms and Computation in Mathematics.* **3**, Springer-Verlag, New York,1999.

[9] Menezes,A.J., P.C.V. Oorschot and S.A. Vanstone, *Handbook of Applied Cryptography*,CRCPress LLC, 1997.

[10] Miller, V., *Use of Elliptic Curve in Cryptography*, Advanced in Cryptography-Crypto ’85, LNCS. **218**, New York, Springer-Verlag, 1986, p.417-426.

[11] Purbo, O.W. and A.A.Wahyudi, *Mengenal e-Commerce*, P.T.Elex Media Komputindo,Jakarta, 2001.

[12] Stallings, W., *Cryptography and Network Security:Principles and Practice*,3 ed., Prentice Hall,New Jersey, 2003.

[13] Stinson, D.R., *Cryptography: Theory and Practice*, CRC Press LLC, USA, 1995.

[14] Trappe,W. and L.C. Washington, *Introduction to Cryptography with Coding Theory*,Prentice Hall, New Jersey, 2002.