

ECC Encryption and Decryption Using Iterated Knapsack Algorithm

A.Uma Maheswari¹ and Prabha Durairaj²

Department of Mathematics

Quaid-E-Millath Government College for Women (Autonomous),
Chennai-600002.

¹umashiva2000@yahoo.com, ²prabhadurairaj59@gmail.com

Abstract

This paper proposes a secure method for message encryption and decryption by combining the elliptic curve analog of the El-Gamal Public key cryptosystem with the Iterated Knapsack Algorithm. A comparison of the proposed method with the existing methods shows that this method is more efficient in terms of computational complexity.

AMS Subject Classification: 11Z05, 11T71

Key Words and Phrases: Elliptic curve, El-Gamal Cryptosystem, Iterated Knapsack Cryptosystem, Discrete logarithm.

1 INTRODUCTION

Major public key cryptosystems that have withstood intruder attacks over the years are the RSA and the Diffie Hellman systems. The key sizes necessary to make these systems secure are growing substantially. Lightweight devices like mobile phones, palmtops, smartcards (where memory, processing power and bandwidths are limited) are constrained in using cryptosystems based on large key-sizes. A need for a secure cryptosystem with a small key size was felt and this was met by Elliptic curve Cryptography (ECC). In 1985, Neal koblitz and Victor Miller independently proposed using elliptic curves to design public-key Cryptographic schemes [1,2]. Elliptic curves provide greater security for a given key size and have

a hard exponential time challenge for an intruder to break into the system.

A public key Cryptosystem called the Merkle Hellman Knapsack Cryptosystem [3] was proposed in 1978 was broken by Adi Shamir [4] in 1982. One way to make the Knapsack system secure is to use a successive sequence of transformations in the Merkle-Hellman cryptosystem. This is the Iterated Knapsack Cryptosystem [3,5].

In this paper, an efficient method for ECC encryption and decryption is presented by combining the El-Gamal Algorithm [6] and the Iterated Knapsack Algorithm. To ensure non-linearity in the transformation of characters, the sender's secret key ' k ' in the El-Gamal algorithm is chosen different each time. A comparison of the proposed method with other Knapsack based algorithms [7,8,9] shows that the proposed method is more efficient. The paper is structured as follows. Section 2 reviews the elliptic curve arithmetic. Section 3 presents the proposed method. Section 4 illustrates the proposed method with numerical examples. The performance and security analysis of the proposed method is discussed in section 5 followed by conclusion.

2 PRELIMINARY CONCEPTS

This section reviews the basic elliptic curve arithmetic.

2.1 Elliptic curves modulo a prime p [10]

Elliptic curves defined over the prime field F_p are of central importance in public key cryptography.

Elliptic curve over F_p : Let $p > 3$ be prime. The elliptic curve $y^2 = x^3 + ax + b$ over F_p is the set E of solutions $(x, y) \in (F_p \times F_p)$ to the congruence $y^2 \equiv x^3 + ax + b \pmod{p}$ where $a, b \in F_p$ are constants such that $4a^3 + 27b^2 \not\equiv 0 \pmod{p}$ together with a special point O called the point at infinity. (The condition $4a^3 + 27b^2 \not\equiv 0 \pmod{p}$ ensures that the cubic polynomial $x^3 + ax + b$ has no multiple roots).

Let P and Q be two points on E . We define the negative of P and $sum(P+Q)$ as follows: (i) If $P = (x, y)$ then $-P = (x, -y)$ (ii) Suppose $P, Q \in E$ where $P = (x_1, y_1)$ & $Q = (x_2, y_2)$. We define $P + Q = (x_3, y_3)$

Where $x_3 = \lambda^2 - x_1 - x_2 \pmod{p}$, $y_3 = \lambda(x_1 - x_3) - y_1 \pmod{p}$,

$$\lambda = \begin{cases} (y_2 - y_1)(x_2 - x_1)^{-1} \pmod{p} & \text{if } P \neq Q \\ (3x_1^2 + a)(2y_1)^{-1} \pmod{p} & \text{if } P = Q \end{cases}$$

$(E, +)$ forms an abelian group.

3 PROPOSED METHOD

In this section, a new method for ECC encryption and decryption based on the El-Gamal and Iterated Knapsack Cryptosystem is described. The authors lay emphasis on the following points for secure encryption: (i) In the El-Gamal system, the random secret integer ' k ' chosen by the sender to encrypt a character, should be chosen different each time. This ensures that the same alphanumeric characters from different words are mapped onto different points on the elliptic curve. (ii) The enciphering vectors $\{u_i\}$ are not made public but are secretly shared between the communicating parties. **Alice(Source)** and **Bob(destination)** agree upon the following parameters: An elliptic curve E defined over a finite field F_q , a base point $B \in E$. These are made public.

Suppose Alice wants to send a character to Bob. Alice chooses a random integer k , where $k < q$ and keeps it secret.

Bob: (i) Chooses a random integer a , $0 < a < (q - 1)$ and keeps it secret. (ii) Computes aB and makes it public. (iii) Let n be the length of the binary string in the binary representations of x and y where $P = (x, y)$ are the encrypted texts which are points of E . Bob chooses a superincreasing sequence $\{v_i\}$ of integers, $i = 0, 1, \dots, (n - 1)$, chooses two pairs of integers (a_1, m_1) and (a_2, m_2) subject to the conditions $\gcd(a_1, m_1) = 1$, $\gcd(a_2, m_2) = 1$,

$$m_1 > \sum_{i=0}^{n-1} v_i, m_2 > nm_1$$

Bob computes the n -tuple $\{w_i\}_{i=0}^{n-1}$ where $w_i = a_1 v_i \bmod m_1$, n -tuple $\{u_i\}_{i=0}^{n-1}$ where $u_i = a_2 w_i \bmod m_2$, $b_1 = a_1^{-1} \bmod m_1$ and $b_2 = a_2^{-1} \bmod m_2$. Bob keeps the numbers $v_i, w_i, m_1, m_2, a_1, a_2, b_1, b_2$ secret but shares the enciphering key $\{u_1, u_2, \dots, u_n\}$ with only Alice. The deciphering key $k_D = \{b_1, b_2, m_1, m_2\}$ is kept secret.

When Alice wants to transmit a character to Bob, she does the following:

She imbeds the plaintext message unit with numerical equivalent m as some point $P_m = (x, y)$ on the elliptic curve E , uses the secret integer k and the publicly known point aB and computes $kB = (x_1, y_1)$ and $P_m + k(aB) = (x_2, y_2)$. The encrypted version of the plaintext character is the pair of points $(kB, P_m + k(aB)) = ((x_1, y_1), (x_2, y_2))$

Instead of directly sending the pair of points $((x_1, y_1), (x_2, y_2))$ to Bob, Alice masks these points further by using the Iterated

Knapsack Algorithm. Each co-ordinate $x_i, y_i, i = 1, 2$ is subjected to the Iterated Knapsack process. Suppose x_1 has binary form $x_1 = (\varepsilon_{k-1}\varepsilon_{k-2}\dots\varepsilon_1\varepsilon_0)_2$, Alice computes the sum $S[x_i] = \sum_{i=0}^{n-1} \varepsilon_i u_i$. Similar sums are computed for y_1, x_2, y_2 so that the encrypted message now takes the form, $C_m = ((S[x_1], S[y_1]), (S[x_2], S[y_2]))$. C_m is the final encrypted form of one plaintext character sent to Bob.

Bob takes each one of the values $S[x_i], S[y_i], i = 1, 2$ and applies the reverse Knapsack algorithm. For example, consider $S[x_1] = \sum_{i=0}^{n-1} \varepsilon_i u_i$. Bob uses the deciphering key $k_D = \{b_1, b_2, m_1, m_2\}$ to compute the product $b_2 S[x_1]$, finds the least positive residue of this product modulo m_2 . Let this value be $R[x_1]$. Bob then computes the product $b_1 R[x_1]$ and finds the least positive residue of this value modulo m_1 . The resulting value is nothing but $\sum_{i=0}^{n-1} \varepsilon_i u_i$. Applying the Knapsack Algorithm for a superincreasing sequence, Bob determines the ε_i and recovers the value x_1 . Similar procedure is used to recover the values y_1, x_2, y_2 .

Thus Bob removes the outer mask of the ciphertext and obtains the points $((x_1, y_1), (x_2, y_2)) = (kB, P_m + k(aB))$.

To remove the inner mask and retrieve P_m from the above pair, Bob uses his secret key a , computes $a(kB)$ and subtracts this from $P_m + k(aB)$ to obtain $P_m = P_m + k(aB - a(kB))$. From P_m , one can retrace the numerical equivalent m of the plaintext character.

4 IMPLEMENTATION OF THE PROPOSED ALGORITHM

The following example [5] illustrates the proposed algorithm.

Example 4.1: Let E be the elliptic curve $y^2 = x^3 - x + 188$ defined over the field of $p = 751$ elements. Suppose that plaintext message units are the decimal digits 0-9 and the letters A-Z with numerical equivalents 10-35 respectively. We shall show how to encrypt and decrypt the letter "S" using the above algorithm. **Let Alice (source) and Bob (destination) be two users of the system.** The elliptic curve is $y^2 = x^3 - x + 188 \pmod{751}$. The base point B is selected as $B = (0, 376)$. These are made public.

Bob: (i) Chooses a secret key $a = 257$ (ii) Computes $aB = 257(0, 376) = (370, 740)$ and makes it public. (iii) Since $p = 752$ is a 10-

bit string, Bob chooses a superincreasing sequence of 10 vectors $\{v_i\}, i = 0, \dots, 9$. Here $\{v_i\} = \{1, 2, 4, 9, 17, 34, 69, 137, 280, 575\}$,
 (iv) Chooses $(a_1, m_1) = (283, 1129)$, $(a_2, m_2) = (728, 11311)$.
 (v) Computes the 10-tuple $\{w_i\}$ and then the 10-tuple $\{u_i\}$,
 Here $\{u_i\} = \{2426, 4852, 2184, 6794, 11162, 11013, 5621, 8816, 5837, 6673\}$. (vi) Computes $b_1 = 283^{-1} \bmod 1129 = 754$, $b_2 = 728^{-1} \bmod 11311 = 5236$. Bob shares the enciphering key $\{u_i\}$ with Alice but keeps $\{b_1, b_2, m_1, m_2\}$ secret.

To transmit the character "S" to Bob, Alice does the following:

"S" has numerical equivalent $m = 28$. Alice imbeds m as a point $P_m = (562, 201)$ on the elliptic curve using Koblitz method. Alice computes (i) $kB = 129(0, 376) = (373, 742)$ (ii) $k(aB) = 129(370, 740) = (684, 632)$ (iii) $P_m + k(aB) = (562, 201) + (684, 632) = (740, 389)$ After this first stage of encryption, the encrypted version of the message (P_m) is the pair

$$(kB, P_m + k(aB)) = ((x_1, y_1), (x_2, y_2)) = ((373, 742), (740, 389))$$

The second stage of encryption is done using the Iterated Knapsack Algorithm. Alice computes $S[x_1]$ as follows: $x_1 = (373) = (101110101)_2 = (\varepsilon_9, \dots, \varepsilon_1, \varepsilon_0)_2$; $S[x_1] = \sum_{i=0}^9 \varepsilon_i u_i = 38243$. Similarly, $S[y_1] = 39, 159$, $S[x_2] = 34, 307$, $S[y_2] = 19, 263$ After this second stage of encryption, the encrypted version of the message P_m is $((S[x_1], S[y_1]), (S[x_2], S[y_2])) = ((38243, 39159), (34307, 19263))$. This is transmitted to Bob.

To decrypt the message and obtain P_m , Bob applies the Reverse Iterated Knapsack Algorithm, removes the outer mask and obtains the co-ordinates, $((x_1, y_1), (x_2, y_2)) = ((373, 742), (740, 389))$. The recovery of bit pattern for x_1 from $S[x_1]$ is illustrated: Computes $b_2 S[x_1] \bmod m_2 = (5263)(38243) \bmod 11311 = 1715$. Next Bob computes $b_1 R[x_1] \bmod m_1 = (754)(1715) \bmod 1129 = 405$, this value is nothing but $\sum_{i=0}^9 \varepsilon_i v_i = V = 405$. Applying the Knapsack Algorithm for a superincreasing sequence of vectors, the bit pattern for x_1 is recovered as $x_1 = (0101110101)_2 = (373)_{10}$. Similarly, obtaining the values y_1, x_2, y_2 , after the first stage of decryption, the outer mask of P_m has been removed and the values $((x_1, y_1), (x_2, y_2)) = ((373, 742), (740, 389))$ have been recovered. Next, to remove the inner mask and obtain the point P_m , Bob does the following: Uses his private key "a" = 257 and computes

(i) $a(kB) = 257(373, 742) = (684, 632)$; (ii) $P_m + k(aB) - a(kB) = (740, 389) - (684, 632) = (740, 389) + (684, 119) = (562, 201) = P_m$.

Knowing P_m , the value m is recovered as $m = \left\lfloor \frac{562 - 1}{20} \right\rfloor = 28$; $m = 28$ is the numerical equivalent of "S".

Table 1

Character	Encryption		Decryption	
	I Stage $(kB, P_m + k(aB))$	II Stage Iterated Knapsack	I Stage Reverse Knapsack	II Stage $P_m + k(aB) - a(kB)$
S=28	$((373, 742), (740, 389))$	$((38243, 39159), (34307, 19263))$	$((373, 742), (740, 389))$	$(562, 201) = P_m$

5 SECURITY ANALYSIS AND PERFORMANCE

In this section, a security analysis of the proposed method is made. It is not possible for an eavesdropper to find the value k (sender's private key) either from public information kB or using the knowledge of two plaintexts because (i) the elliptic curve discrete logarithm problem is very difficult to solve (when the elliptic curve is non-supersingular) (ii) the random secret integer k is chosen different each time (even when the same character is to be encrypted in different words). It is not possible for an eavesdropper to decrypt the message using the Iterated Knapsack decryption Algorithm because the enciphering keys $\{u_i\}$ are not made public but are secretly shared between the communicating parties. Next, the proposed method is compared with other Knapsack based methods [7,10]. From Table 2 it is seen that the proposed method uses the minimum number of elliptic curve operations.

6 CONCLUSION

In this paper a reliable encryption and decryption scheme using the Iterated Knapsack Algorithm has been proposed to provide an additional level of security for the Elliptic Curve El-Gamal Encryption. In the proposed method, same characters in the plaintext will be mapped to different points on the curve by using a different value of k (Sender's key) each time in the El-Gamal encryption. A comparison with the existing methods shows that this method uses the minimum number of elliptic curve operations to achieve the same level of security.

Table 2
In the following table, the elliptic curve operations (EC) are denoted as follows:
SM-Scalar multiplication, A-addition, S-subtraction

Algorithm	ENCRYPTION				DECRYPTION			
	Computation	No. of EC operation	Total no. of EC operation	No. of arithmetic operations	Computation	No. of EC Operations	Total no. of EC operations	No. of arithmetic operations
OUR PROPOSED ALGORITHM	Stage 1: $(kB, P_m + k(aB)) = ((x_1, y_1), (x_2, y_2))$ Stage 2: $S[x_i], S[y_i], i = 1, 2$ (Iterated Knapsack)	2 SM 1 A	-	4(n-1) additions	Stage 1: Reverse Iterated Knapsack Stage 2: $a(kB)$ $P_m + k(aB) - a(kB)$	- 1 SM 1 Subtraction	-	2 multiplication 2 divisions 4(n-1) subtraction
		Totally	2 SM + 1 A	4(n-1) additions		Totally	1 SM + 1 Subtraction	2 multiplication 2 divisions 4(n-1) subtraction
ALGORITHM [9] MATRIX APPROACH	Stage 1: $Q = AM$ where A is a 3x3 matrix Stage 2: $D_i = Q_i + R_i$ Modified Knapsack	3 SM 2 A 1 A	-	-	Stage 1: Reverse Knapsack $Q_i = D_i + R_i$ Stage 2: $M = A^{-1}Q$	1 A 3 SM 2 A	-	-
		Totally	3SM+3A			Totally	3SM+3A	-
ALGORITHM [7]	Stage 1: $P_m = AP_m$ $(kG, P_m + kP_B)$ Stage 2: Knapsack	1 SM 2SM & 1A Totally	4(n-1) additions	4(n-1) additions	Stage 1: Reverse Knapsack Stage 2: $n_B(kG)$ $P_m + kP_B - n_BkG$ Finding P_m from P_m using discrete log table	- 1 SM & 1S Totally	- 4(n-1) subtraction Preparation of discrete log table +1SM+1S	4(n-1) subtraction

References

- [1] Neal Koblitz, Elliptic Curve Cryptosystems, *Mathematics of Computation*, vol. 48, No 177, 1987, pp 203-209.
- [2] Miller V, Use of elliptic curves in Cryptography, *Advances in Cryptography-Crypto '85*, LNCS 218, Springer Verlag.
- [3] M.E. Hellman and R.C. Merkle, Hiding information and signature in trapdoor knapsacks, *IEEE Transactions on Information Theory*, 24, 1978, pp 525-530.
- [4] A. Shamir, A polynomial time algorithm for breaking the basic Merkle-Hellman Cryptosystem, *Proc. 23rd Annual Symposium on the Foundations of Computer Science*, 1982, pp 145-152.
- [5] Neal Koblitz, A Course in Number Theory and Cryptography, *Second Edition*, Springer, 1994.
- [6] T. El-Gamal, A Public key Cryptosystem and a signature scheme based on discrete logarithms, *IEEE Transactions on Information Theory*, IT-31(1985), pp 473-481.
- [7] R. Rajaram Ramasamy, M. Amutha Prabakar, M. Indra Devi and M. Suguna, Knapsack Based ECC Encryption and Decryption, *International Journal of Network Security*, vol.9, No.3, 2009, pp 218-226.
- [8] Ashish Agarwal, Encrypting Messages using the Merkle-Hellman Knapsack Cryptosystem, *International Journal of Computer Science and Network Security*, vol.11, No.5, 2011, pp 12-14.
- [9] Fatima Amounas, Efficient methodology for Encrypting Amazing alphabet using Modified Knapsack Algorithm based ECC, *International Journal on Recent and Innovation Trends in Computing and Communication*, Vol. 4, 2016, 502-506.
- [10] Douglas R. Stinson, Cryptography Theory and Practice, *Third Edition, Discrete Mathematics and Its Applications, Series Editor Kenneth H. Rosen*, pp 258.

