AUTHENTICATED PUBLIC KEY ENCRYPTION SCHEME USING ELLIPTIC CURVE CRYPTOGRAPHY

P. L. Sharma[1], Kritika Gupta[2], Nikhlesh Kumar Badoga[3],
Ashima[4] and Himanshu Monga[4]

[1,2,4]Department of Mathematics & Statistics, Himachal Pradesh
University, Shimla, India
[3]Department of Computer Science and Engineering,
Thapar Institute of Engineering and Technology, Punjab, India
[4]Department of Electronics & Communication Engineering
Jwaharlal Nehru Government Engineering College, Sundernagar, Mandi (H.P), India

ABSTRACT

Secure transformation of data is of prime importance in today's world. In the present paper, we propose a double fold authenticated public key encryption scheme which helps us in securely sending the confidential data between sender and receiver. This scheme makes the encrypted data more secure against various cryptographic attacks.

1. INTRODUCTION

The theory of elliptic curves in cryptography was first introduced by Miller [9] and Koblitz [4] in the year 1985. Security of Elliptic Curve Cryptography mainly depends on the problem of solving Elliptic Curve Discrete Logarithm Problem. The main advantage of using ECC over RSA is that it provides same level of security but with lesser key size. Elliptic Curve Cryptography finds many applications in wireless networks and mobile computing. Elliptic Curve Cryptography is an active area of research. ECC can be used in several tasks like digital signature, decryption, encryption, key agreement and authentication, see [2, 3, 5, 7, 8, 13]. The book "Guide to Elliptic Curve Cryptography" provides many details of arithmetic of elliptic curves and implementation issues, see [1]. Kumar et al. [6] proposed a secure and authentic method to encrypt a message. Through Elliptic curve cryptography, we can securely transport keys between sender and receiver and it also helps in authentic session key establishment protocols, see [10, 11, 12, 14, 15]. In the present paper, we propose an encryption scheme that makes use of the specific public key and XOR operation to encrypt the message securely. This scheme uses double fold encryption to enhance the security of encrypted message. To securely send the data between sender and real receiver is of prime importance in Bank transactions, e-commerce and tenders etc. Proposed scheme provides confidentiality along with authentication and provides higher level of security. We use elliptic curves over finite fields for this encryption algorithm.

2. PRELIMINARIES

2.1. Group Law on Elliptic Curves

Suppose that $E(F_p)$ is an elliptic curve defined over the finite field F_p, then the set of points of elliptic curve along with addition operation form an abelian group and point of infinity act as an identity element of the group.

DOI : 10.5121/iju.2022.13202

2.2. Point Addition

Let $F_p$be any finite field with prime p. Equation of elliptic curve over F_p is
$y^2 = (x^3 + ax + b)$ *mod p.*

Let $S(p_1,q_1)$ and $T(p_2,q_2)$ be two points on $E(F_p)$ such that $S \neq T$ and $S + T = U(p_3,q_3)$, then

$$p_3 = \{\lambda^2 - p_1 - p_2\} \; mod \; p$$

and

$$q_3 = \{\lambda(p_1 - p_3) - q_1\} \; mod \; p,$$

where

$$\lambda = \frac{q_2 - q_1}{p_2 - p_1} \; mod \; p.$$

2.3. Point Doubling

Let S and T be two overlapping points on elliptic curve over F_p and $S(p_1,q_1) + T(p_1, q_1) = U(p_2,q_2)$, then

$$p_2 = \{\lambda^2 - 2p_1\} \; mod \; p$$

and

$$q_2 = \{\lambda(p_1 - p_2) - q_1\} \; mod \; p,$$

where

$$\lambda = \frac{3{p_1}^2 + a}{2q_1} \; mod \; p.$$

2.4. Elliptic Curve Discrete Logarithm Problem

Security of ECC depends on Elliptic Curve Discrete Logarithm. Given two points S and T on elliptic curve over finite field. For given x and S, it is easy to compute $T = xS$, but very difficult to find x if S and T are given. The problem to find x is known as Elliptic Curve Discrete Logarithm Problem.

3. PROPOSED SCHEME

If Alice wants to send some message to Bob, then they both will agree on an elliptic curve over finite field and some common code table. Let G be a generator point of order n. Alice selects his private keys α and n_A in such a manner that αn_A lies in the interval $[1, n - 1]$ and α and n_A should be large random numbers. He computes his public key as $P_A = n_A G$.

In the similar manner, Bob selects his private keys as β and n_B and public key as $P_B = n_B G$. After publishing their public keys, they calculate specific public keys for each other.

Specific public key generated by Alice only for Bob is $A_B = \alpha \, n_A P_B$ and specific public key generated by Bob only for Alice is$B_A = \beta n_B P_A$.

3.1. Encryption Stage 1

Alice converts the character of the message to point on elliptic curve using Code table. He chooses a random integer k and generates cipher points for each message point as:

$$C_1 = kG$$

and

$$C_2 = P_m + kP_B + \alpha \mathrm{B_A}.$$

3.2. Encryption Stage 2

Step 1: Convert each coordinate of the points obtained in the above step to binary form.
Step 2: Perform XOR operation between binary form of x-coordinate of C_1, binary form of x-coordinate of A_B, binary form of x-coordinate of $\mathrm{B_A}$and binary form of a randomly chosen integer t. Randomly chosen integer must be kept confidential between sender and receiver.
Step 3: Result obtained in the above step will be changed to decimal value.
Step 4: Perform XOR operation between binary form of x-coordinate of C_2, binary form of x-coordinate of A_B, binary form of x-coordinate of $\mathrm{B_A}$and binary form of a randomly chosen integer t.
Step 5: Result obtained in the above step will be changed to decimal value.
Step 6: Performs XOR operation between binary form of y-coordinate of C_1, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B_A}$and binary form of a randomly chosen integer t and change the resultant value to decimal value.
Step 7: Performs XOR operation between binary form of y-coordinate of C_2, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B_A}$and binary form of a randomly chosen integer t and change the resultant value to decimal value. These four decimal values act as cipher text corresponding to first character of the message.

Following the same procedure, he creates the cipher text corresponding to the remaining characters of the message. But to encrypt the remaining characters of the message, he will use binary form of preceding plain text in place of randomly chosen integer and send the cipher text to receiver.

3.3. Decryption Stage 1

Step 1: After obtaining cipher text, he converts all the decimal values to binary form.
Step 2: Performs XOR operation between binary form of first decimal value, binary form of x-coordinate of A_B, binary form of x-coordinate of $\mathrm{B_A}$and binary form of a randomly chosen integer t and the resultant will correspond to x-coordinate of C_1.
Step 3: Performs XOR operation between binary form of second decimal value, binary form of x-coordinate of A_B, binary form of x-coordinate of $\mathrm{B_A}$and binary form of a randomly chosen integer t and the resultant will correspond to x-coordinate of
C_2.

Step 4: Performs XOR operation between binary form of third decimal value, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B_A}$and binary form of a randomly chosen integer t and the resultant will correspond to y-coordinate of C_1. Step 5: Performs XOR operation between binary form of fourth decimal value, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B_A}$and binary form of a randomly chosen integer t and the resultant will correspond to y-coordinate ofC_2.

In the similar manner, he obtains all the cipher points (C_1,C_2) corresponding to each character of the message.

3.4. Decryption Stage 2

After obtaining all the values for (C_1,C_2), Bob will decrypt the message P_m as:

$$P_{\mathrm{m}}= C_2- C_1 n_B- \beta \mathrm{A}_B.$$

and gets the original message.

4. ILLUSTRATION

Suppose Alice and Bob agree on elliptic curve $y^2 = (x^3 + 2x + 2)$ *mod* 17 over the finite field F_{17}. Elliptic curve points are as given below:

Table 1

∞	(16,13)	(0,11)	(5,16)
(5,1)	(0,6)	(16,4)	
(6,3)	(13,7)	(9,1)	
(10,6)	(7,6)	(3,16)	
(3,1)	(7,11)	(10,11)	
(9,16)	(13,10)	(6,14)	

Here, ∞ is the point of infinity.
Generator of the elliptic curve is G = (5,1).
Private keys of Alice are α = 2, n_A= 3.
Private keys of Bob are β = 3, n_B= 4.
Public key of Alice is $P_A= n_A G$ = (10,6).
Public key of Bob is $P_B= n_B G$ = (3,1).
Specific public key generated by Alice for Bob is $\mathrm{A}_B= \alpha n_A P_B$= (9,16) and Specific public key generated by Bob for Alice is $\mathrm{B}_{\mathrm{A}}= \beta n_B P_A$= (6,14).

4.1. Encryption

If Alice wants to send the message 'daring' to Bob, he encodes the characters of the message 'daring' to points on elliptic curve using Code Table given below:

Code Table

*	a	b	c	d	e	f	g
∞	(5,1)	(6,3)	(10,6)	(3,1)	(9,16)	(16,13)	(0,6)
h	i	j	k	l	m	n	o
(13,7)	(7,6)	(7,11)	(13,10)	(0,11)	(16,4)	(9,1)	(3,16)
p	q	r					
(10,11)	(6,14)	(5,16)					

4.2. Encryption Stage 1

- Alice represents the letter 'd' of daring as (3,1). He selects a random number k = 6 and encrypts the point (3,1). Cipher Points corresponding to point (3,1) are $C_1= kG$= (16,13) and $C_2= P_m+ kP_B+ \alpha \mathrm{B}_{\mathrm{A}}$= (9,16).

- Alice represents the letter ‘a’ of daring as (5,1). He selects a random number k = 4 and encrypts the point (5,1). Cipher Points corresponding to point (5,1) are C_1= kG= (3,1) and C_2= P_m+ kP_B+ $\alpha\mathrm{B}_\mathrm{A}$= (16,4).
- Alice represents the letter ‘r’ of daring as (5,16). He selects a random number k = 5 and encrypts the point (5,16). Cipher Points corresponding to point (5,16) are C_1= kG= (9,16) and C_2= P_m+ kP_B+ $\alpha\mathrm{B}_\mathrm{A}$= (3,16).
- Alice represents the letter ‘i’ of daring as (7,6). He selects a random number k = 8 and encrypts the point (7,6). Cipher Points corresponding to point (7,6) are C_1= kG= (13,7) and C_2= P_m+ kP_B+ $\alpha\mathrm{B}_\mathrm{A}$= (5,16).
- Alice represents the letter ‘n’ of daring as (9,1). He selects a random number k = 9 and encrypts the point (9,1). Cipher Points corresponding to point (9,1) are C_1= kG= (7,6) and C_2= P_m+ kP_B+ $\alpha\mathrm{B}_\mathrm{A}$= (13,7).
- Alice represents the letter ‘g’ of daring as (0,6). He selects a random number k = 10 and encrypts the point (0,6). Cipher Points corresponding to point (0,6) are C_1= kG= (7,11) and C_2= P_m+ kP_B+ $\alpha\mathrm{B}_\mathrm{A}$= (9,16).

C_1	C_2
(16,13)	(9,16)
(3,1)	(16,4)
(9,16)	(3,16)
(13,7)	(5,16)
(7,6)	(13,7)
(7,11)	(9,16)

4.3. Encryption Stage 2

- Step 1: He performs XOR operation between binary form of x-coordinate of C_1, binary form of x-coordinate of A_B, binary form of x-coordinate of $\mathrm{B}_\mathrm{A}$and binary form of a randomly chosen integer t = 8, that is,

  00010000 ⊕ 00001001 ⊕ 00000110 ⊕ 00001000 = 00010111.
  Decimal value corresponding to 00010111 is 23.

- Step 2: He performs XOR operation between binary form of x-coordinate of C_2, binary form of x-coordinate of A_B, binary form of x-coordinate of $\mathrm{B}_\mathrm{A}$and binary form of a randomly chosen integer t = 8, that is,

  00001001 ⊕ 00001001 ⊕ 00000110 ⊕ 00001000 = 00001110.
  Decimal value corresponding to 00001110 is 14.

- Step 3: He performs XOR operation between binary form of y-coordinate of C_1, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B}_\mathrm{A}$and binary form of a randomly chosen integer t = 8, that is,

  00001101 ⊕ 00010000 ⊕ 00001110 ⊕ 00001000 = 00011011.
  Decimal value corresponding to 00011011 is 27.

- Step 4: He performs XOR operation between binary form of y-coordinate ofC_2, binary form of y-coordinate of A_B, binary form of y-coordinate of$A_B$andbinary form of a randomly chosen integer $t = 8$, that is,

  00010000 ⊕ 00010000 ⊕ 00001110 ⊕ 00001000 = 00000110
  Decimal value corresponding to 00000110 is 6.

Therefore, cipher text corresponding to first character of the message, that is, ‘d’ is 23,14,27,6.

- Step 1: He performs XOR operation between binary form of x-coordinate of C_1, binary form of x-coordinate of A_B, binary form of x-coordinate of $B_A$and binary form of preceding plain text, ‘d’, that is,

  00000011 ⊕ 00001001 ⊕ 00000110 ⊕ 01100100 = 01101000.
  Decimal value corresponding to 01101000 is 104.

- Step 2: He performs XOR operation between binary form of x-coordinate of C_2, binary form of x-coordinate of A_B, binary form of x-coordinate of $B_A$and binary form of preceding plain text ‘d’, that is,

  00010000 ⊕ 00001001 ⊕ 00000110 ⊕ 01100100 = 01111011.
  Decimal value corresponding to 01111011 is 123.

- Step 3: He performs XOR operation between binary form of y-coordinate of C_1, binary form of y-coordinate of A_B, binary form of y-coordinate of $B_A$and binary form of preceding plain text ‘d’, that is,

  00000001 ⊕ 00010000 ⊕ 00001110 ⊕ 01100100 = 01111011.
  Decimal value corresponding to 01111011 is 123.

- Step 4: He performs XOR operation between binary form of y-coordinate of C_2, binary form of y-coordinate of A_B, binary form of y-coordinate of $B_A$and binary form of preceding plain text ‘d’, that is,

  00000100 ⊕ 00010000 ⊕ 00001110 ⊕ 01100100 = 01111110
  .

Therefore, cipher text corresponding to second character of the message, that is, ‘a’ is 104,123,123,126.

- Step 1: He performs XOR operation between binary form of x-coordinate ofC_1, binary form of x-coordinate of A_B, binary form of x-coordinate of $B_A$binary form of preceding plain text ‘a’, that is,

  00001001 ⊕ 00001001 ⊕ 00000110 ⊕ 01110010 = 01110100
  Decimal value corresponding to 01110100 is 116.

- Step 2: He performs XOR operation between binary form of x-coordinate of C_2, binary form of x-coordinate of A_B, binary form of x-coordinate of B_A and binary form of preceding plain text ‘a’, that is,

  00000011 ⊕ 00001001 ⊕ 00000110 ⊕ 01110010 = 01111110.

Decimal value corresponding to 01111110 is 126.

- Step 3: He performs XOR operation between binary form of y-coordinate of C_1, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B_A}$ and binary form of preceding plain text ‘a’, that is,

  00010000 ⨁ 00010000 ⨁ 00001110 ⨁ 01110010 = 01111100.
  Decimal value corresponding to 01111100 is 124.

- Step 4: He performs XOR operation between binary form of y-coordinate of C_2, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B_A}$ and binary form of preceding plain text ‘a’, that is,

  00010000 ⨁ 00010000 ⨁ 00001110 ⨁ 01110010 = 01111100.
  Decimal value corresponding to 01111100 is 124.

Therefore, cipher text corresponding to third character of the message, that is, ‘r’ is 116,126,124,124.

- Step 1: He performs XOR operation between binary form of x-coordinate of C_1, binary form of x-coordinate of A_B, binary form of x-coordinate of $\mathrm{B_A}$and binary form of preceding plain text ‘r’, that is,

  00001101 ⨁ 00001001 ⨁ 00000110 ⨁ 01110010 = 01110000.
  Decimal value corresponding to 01110000 is 112.

- Step 2: He performs XOR operation between binary form of x-coordinate ofC_2, binary form of x-coordinate of A_B, binary form of x-coordinate of $\mathrm{B_A}$ and binary form of preceding plain text ‘r’, that is,

  00000101 ⨁ 00001001 ⨁ 00000110 ⨁ 01110010 = 01111000.
  Decimal value corresponding to 01111000 is 120.

- Step 3: He performs XOR operation between binary form of y-coordinate of C_1, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B_A}$and binary form of preceding plain text ‘r’, that is,

  00000111 ⨁ 00010000 ⨁ 00001000 ⨁ 01110010 = 01101101.
  Decimal value corresponding to 01101101 is 109.

- Step 4: He performs XOR operation between binary form of y-coordinate of C_2, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B_A}$and binary form of preceding plain text ‘r’, that is,

  00010000 ⨁ 00010000 ⨁ 00001000 ⨁ 01110010 = 01111010.
  Decimal value corresponding to 01111010 is 122.

Therefore, cipher text corresponding to fourth character of the message, that is, ‘i’ is 112,120,109,122.

- Step 1: He performs XOR operation between binary form of x-coordinate of C_1, binary form of *x*-coordinate of A_B, binary form of *x*-coordinate of $\mathrm{B}_\mathrm{A}$and binary form of preceding plain text 'i', that is,

  $00000111 \oplus 00001001 \oplus 00000110 \oplus 01101001 = 01100001$.
  Decimal value corresponding to 01100001 is 97.

- Step 2: He performs XOR operation between binary form of x-coordinate of C_2, binary form of *x*-coordinate of A_B, binary form of *x*-coordinate of $\mathrm{B}_\mathrm{A}$and binary form of preceding plain text 'i', that is,

  $00001101 \oplus 00001001 \oplus 00000110 \oplus 01101001 = 01101011$
  Decimal value corresponding to 01101011 is 105.

- Step 3: He performs XOR operation between binary form of y-coordinate ofC_1, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B}_\mathrm{A}$andbinary form of preceding plain text 'i', that is,

  $00001101 \oplus 00010000 \oplus 00001000 \oplus 01101001 = 01111100$
  Decimal value corresponding to 01111100 is 124.

- Step 4: He performs XOR operation between binary form of y-coordinate of C_2, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B}_\mathrm{A}$and binary form of preceding plain text 'i', that is,

  $00000111 \oplus 00010000 \oplus 00001000 \oplus 01101001 = 01110110$.
  Decimal value corresponding to 01110110 is 118.

Therefore, cipher text corresponding to fifth character of the message, that is, 'n' is 97,105,124,118.

- Step 1: He performs XOR operation between binary form of x-coordinate of C_1, binary form of *x*-coordinate of A_B, binary form of *x*-coordinate of $\mathrm{B}_\mathrm{A}$and binary form of preceding plain text 'n', that is,

  $00000111 \oplus 00001001 \oplus 00000110 \oplus 01101110 = 01100110$.
  Decimal value corresponding to 01100110 is 102.

- Step 2: He performs XOR operation between binary form of x-coordinate ofC_2, binary form of *x*-coordinate of A_B, binary form of *x*-coordinate of $\mathrm{B}_\mathrm{A}$and binary form of preceding plain text 'n', that is,

  $00001001 \oplus 00001001 \oplus 00000110 \oplus 01101110 = 01101000$.
  Decimal value corresponding to 01101000 is 104.

- Step 3: He performs XOR operation between binary form of y-coordinate of C_1, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B}_\mathrm{A}$and binary form of preceding plain text 'n', that is,

  $00001011 \oplus 00010000 \oplus 00000100 \oplus 01101110 = 01110001$.
  Decimal value corresponding to 01110001 is 113.

- Step 4: He performs XOR operation between binary form of y-coordinate ofC_2, binary form of y-coordinate of A_B , binary form of y-coordinate of $\mathrm{B_A}$ and binary form of preceding plain text ‘n’, that is,

  00010000 ⊕ 00010000 ⊕ 00000100 ⊕ 01101110 = 01101010.
  Decimal value corresponding to 01101010 is 106.

Therefore, cipher text corresponding to sixth character of the message, that is, ‘g’ is 102,104,113,106.

Alice sends the cipher text 23, 14, 27, 6;104, 123, 123, 126; 116, 126, 124, 124; 112, 120, 109, 122; 97, 105, 124, 118; 102, 104, 113, 106 to Bob.

4.4. Decryption Stage 1

To get (C_1,C_2) corresponding to the first character of the message, he performs following steps:

- Step 1: He performs XOR operation between binary form of first decimal value of cipher text, binary form of x-coordinate of A_B, binary form of x-coordinate of $\mathrm{B_A}$and binary form of a randomly chosen integer t = 8, that is,

  00010111 ⊕ 00001000 ⊕ 00000110 ⊕ 00001001 = 00010000.
  Decimal value corresponding to 00010000 is 16 and it serve as x-coordinate of C_1.

- Step 2: He performs XOR operation between binary form of second decimal value of cipher text, binary form of x-coordinate of A_B, binary form of x-coordinate of $\mathrm{B_A}$and binary form of a randomly chosen integer t = 8, that is,

  00001110 ⊕ 00001000 ⊕ 00000110 ⊕ 00001001 = 00001001.
  Decimal value corresponding to 00001001 is 9 and it serve as x-coordinate of C_2.

- Step 3: He performs XOR operation between binary form of third decimal value of cipher text, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B_A}$and binary form of a randomly chosen integer t = 8, that is,

  00011011 ⊕ 00001000 ⊕ 00001110 ⊕ 00010000 = 00001101.
  Decimal value corresponding to 00001101 is 13 and it serve as y-coordinate of C_1.

- Step 4: He performs XOR operation between binary form of fourth decimal value of cipher text, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B_A}$and binary form of a randomly chosen integer t = 8, that is,

  00000110 ⊕ 00001000 ⊕ 0001110 ⊕ 00010000 = 00010000.
  Decimal value corresponding to 00010000 is 16 and it serve as y-coordinate of C_2.

To get (C_1,C_2) corresponding to the second character of the message, he performs following steps:

- Step 1: Performs XOR operation between binary form of fifth decimal value of cipher text, binary form of x-coordinate of A_B, binary form of x-coordinate of $\mathrm{B_A}$and binary form of preceding plain text d, that is,

01101000 ⊕ 01100100 ⊕ 00000110 ⊕ 00001001 = 00000011.
Decimal value corresponding to 00000011 is 3 and it serve as *x*-coordinate of C_1.

- Step 2: Performs XOR operation between binary form of sixth decimal value of cipher text, binary form of *x*-coordinate of A_B, binary form of *x*-coordinate of $\mathrm{B_A}$and binary form of preceding plain text *d*, that is,

  00001001 ⊕ 00000110 ⊕ 01100100 ⊕ 01111011 = 00010000.
  Decimal value corresponding to 00010000 is 16 and it serve as *x*-coordinate of C_2.

- Step 3: Performs XOR operation between binary form of seventh decimal value of cipher text, binary form of *y*-coordinate of A_B, binary form of *y*-coordinate of $\mathrm{B_A}$and binary form of preceding plain text *d*, that is,

  01111011 ⊕ 01100100 ⊕ 00001110 ⊕ 00010000 = 00000001.
  Decimal value corresponding to 00000001 is 1 and it serve as *y*-coordinate of C_1.

- Step 4: Performs XOR operation between binary form of eighth decimal value of cipher text, binary form of *y*-coordinate of A_B, binary form of *y*-coordinate of $\mathrm{B_A}$and binary form of preceding plain text *d*, that is,

  01111110 ⊕ 01100100 ⊕ 00001110 ⊕ 00010000 = 00000100.
  Decimal value corresponding to 00000100 is 4 and it serve as *y* C_2.

To get (C_1,C_2) corresponding to the third character of the message, he performs following steps:

- Step 1: Performs XOR operation between binary form of ninth decimal value of cipher text, binary form of *x*-coordinate of A_B, binary form of *x*-coordinate of $\mathrm{B_A}$and binary form of preceding plain text, that is, *a*, that is,

  01110100 ⊕ 01110010 ⊕ 00000110 ⊕ 00001001 = 00001001.
  Decimal value corresponding to 00001001 is 9 and it serve as *x*-coordinate of C_1.

- Step 2: Performs XOR operation between binary form of tenth decimal value of cipher text, binary form of *x*-coordinate of A_B, binary form of *x*-coordinate of $\mathrm{B_A}$and binary form of preceding plain text *a*, that is,

  01111110 ⊕ 01110010 ⊕ 00000110 ⊕ 00001001 = 00000011.
  Decimal value corresponding to 00000011 is 16 and it serve as *x*-coordinate of C_2.

- Step 3: Performs XOR operation between binary form of eleventh decimal value of cipher text, binary form of *y*-coordinate of A_B, binary form of *y*-coordinate of $\mathrm{B_A}$and binary form of preceding plain text *a*, that is,

  01111100 ⊕ 01110010 ⊕ 00001110 ⊕ 00010000 = 00010000.
  Decimal value corresponding to 00010000 is 16 and it serve as *y*-coordinate of C_1.

- Step 4: Performs XOR operation between binary form of twelfth decimal value of cipher text, binary form of *y*-coordinate of A_B, binary form of *y*-coordinate of $\mathrm{B_A}$and binary form of preceding plain text *a*, that is,

01111100 ⊕ 01110010 ⊕ 00001110 ⊕ 00010000 = 00010000.
Decimal value corresponding to 00010000 is 16 and it serve as *y*-coordinate of C_2.

To get (C_1,C_2) corresponding to the fourth character of the message, he performs following steps:

- Step 1: Performs XOR operation between binary form of thirteenth decimal value of cipher text, binary form of *x*-coordinate of A_B, binary form of *x*-coordinate of $\mathrm{B_A}$and binary form of preceding plain text, that is, *r*, that is,

  01110000 ⊕ 01110010 ⊕ 00000110 ⊕ 00001001 = 00001101.
  Decimal value corresponding to 00001101 is 13 and it serve as *x*-coordinate of C_1.

- Step 2: Performs XOR operation between binary form of fourteenth decimal value of cipher text, binary form of *x*-coordinate of A_B, binary form of *x*-coordinate of $\mathrm{B_A}$and binary form of preceding plain text *r*, that is,

  01111000 ⊕ 01110010 ⊕ 00000110 ⊕ 00001001 = 00000101.
  Decimal value corresponding to 00000101 is 5 and it serve as *x*-coordinate of C_2.

- Step 3: Performs XOR operation between binary form of fifteenth decimal value of cipher text, binary form of *y*-coordinate of A_B, binary form of *y*-coordinate of $\mathrm{B_A}$and binary form of preceding plain text *r*, that is,

  01101101 ⊕ 01110010 ⊕ 00001000 ⊕ 00010000 = 00000111.
  Decimal value corresponding to 00000111 is 7 and it serve as *y*-coordinate of C_1.

- Step 4: Performs XOR operation between binary form of sixteenth decimal value of cipher text, binary form of *y*-coordinate of A_B, binary form of *y*-coordinate of $\mathrm{B_A}$and binary form of preceding plain text *r*, that is,

  01111010 ⊕ 01110010 ⊕ 00001000 ⊕ 00010000 = 00010000.
  Decimal value corresponding to 00010000 is 16 and it serve as *y*-coordinate of C_2.

To get (C_1, C_2) corresponding to the fifth character of the message, he performs following steps:

- Step 1: Performs XOR operation between binary form of seventeenth decimal value of cipher text, binary form of *x*-coordinate of A_B, binary form of *x*-coordinate of $\mathrm{B_A}$and binary form of preceding plain text*i*, that is,

  01100001 ⊕ 01101001 ⊕ 00000110 ⊕ 00001001 = 00000111.
  Decimal value corresponding to 00000111 is 7 and it serve as *x*-coordinate ofC_1.

- Step 2: Performs XOR operation between binary form of eighteenth decimal value of cipher text, binary form of *x*-coordinate of A_B, binary form of *x*-coordinate of $\mathrm{B_A}$and binary form of preceding plain text *i*, that is,

  01101011 ⊕ 01101001 ⊕ 00000110 ⊕ 00001001 = 00001101.
  Decimal value corresponding to 00001101 is 13 and it serve as *x*-coordinate of C_2.

- Step 3: Performs XOR operation between binary form of nineteenth decimal value of cipher text, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B}_\mathrm{A}$and binary form of preceding plain text *i*, that is,

  01111100 ⊕ 01101001 ⊕ 00001000 ⊕ 00010000 = 00001101.
  Decimal value corresponding to 00001101 is 6 and it serve as *y*-coordinate of C_1.

- Step 4: Performs XOR operation between binary form of twentieth decimal value of cipher text, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B}_\mathrm{A}$and binary form of preceding plain text *i*, that is,

  01110110 ⊕ 01101001 ⊕ 00001000 ⊕ 00010000 = 00000111.
  Decimal value corresponding to 00000111 is 7 and it serve as *y*-coordinate of C_2.

To get (C_1,C_2) corresponding to the sixth character of the message, he performs following steps:

- Step 1: Performs XOR operation between binary form of twenty first decimal value of cipher text, binary form of x-coordinate of A_B, binary form of x-coordinate of $\mathrm{B}_\mathrm{A}$and binary form of preceding plain text, that is, *n*, that is,

  01100110 ⊕ 01101110 ⊕ 00000110 ⊕ 00001001 = 00000111.
  Decimal value corresponding to 00000111 is 7 and it serve as *x*-coordinate of C_1

- Step 2: Performs XOR operation between binary form of twenty second decimal value of cipher text, binary form of x-coordinate of A_B, binary form of x-coordinate of $\mathrm{B}_\mathrm{A}$and binary form of preceding plain text *n*, that is,

  01101000 ⊕ 01101110 ⊕ 00000110 ⊕ 00001001 = 00001001.
  Decimal value corresponding to 00001001 is 9 and it serve as *x*-coordinate of C_2.

- Step 3: Performs XOR operation between binary form of twenty third decimal value of cipher text, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B}_\mathrm{A}$and binary form of preceding plain text *n*, that is,

  01110001 ⊕ 011101110 ⊕ 00000100 ⊕ 00010000 = 00001011.
  Decimal value corresponding to 00001011 is 11 and it serve as *y*-coordinate of C_1.

- Step 4: Performs XOR operation between binary form of twenty fourth decimal value of cipher text, binary form of y-coordinate of A_B, binary form of y-coordinate of $\mathrm{B}_\mathrm{A}$and binary form of preceding plain text *n*, that is,

  01101010 ⊕ 01101110n ⊕ 00000100 ⊕ 00010000 = 00010000.
  Decimal value corresponding to 00010000 is 16 and it serve as *y*-coordinate of C_2.

4.5. Decryption Stage 2

P_m= C_2− $C_1 n_B$− βA_B= (3,1) corresponding to the character ‘d’.
P_m= C_2− $C_1 n_B$− βA_B= (5,1) corresponding to the character ‘a’.
P_m= C_2− $C_1 n_B$− βA_B= (5,16) corresponding to the character ‘r’.
P_m= C_2− $C_1 n_B$− βA_B= (7,6) corresponding to the character ‘i’.
P_m= C_2− $C_1 n_B$− βA_B= (9,1) corresponding to the character ‘n’.

$P_{\mathrm{m}}= C_2- C_1 n_B- \beta A_{\mathrm{B}}=$ (0,6) corresponding to the character ‘g’.

5. CONCLUSION

For Bank transactions and mobile computing security of encrypted text is of prime importance. We proposed a new double fold authenticated encryption scheme which provides confidentiality along with authentication. This scheme makes the encrypted message more secure using two stage encryption process.

REFERENCES

[1] D. Hankerson, A. J. Menezes and S. Vanstone, Guide to Elliptic Curve Cryptography, Springer Professional Computing, First ed., Springer -Verlag New York (2004).

[2] A. M. Johnston and P. S. Gemmell, Authenticated Key Exchange Provably Secure Against the Man-in-Middle Attack, Journal of Cryptology. 2 (2002) 139148.

[3] A. Joux, A One Round Protocol for Tripartite Diffie-Hellman, Journal of Cryptology. 17 (2004) 263-276.

[4] N. Koblitz, Elliptic Curve Cryptosystem, Mathematics of Computation. 48 (1987) 203-209.

[5] N. Koblitz, A. J. Menezes and S. Vanstone, The State of Elliptic Curve Cryptography, Design, Codes and Cryptography. 19 (2000) 173-193.

[6] D. S. Kumar, CH. Suneetha and A. Chandrasekhar, Encryption of Data Using Elliptic Curve Over Finite Fields, International Journal of Distributed and Parallel Systems (IJDPS). 3 (2012) 301-308.

[7] A. K. Lenstra and E. R. Verheul, Selecting Cryptographic Key Size, Journal of Cryptology. 14 (2001) 255-293.

[8] M. Machhout, Z. Guitouni, K. Torki and L. Khriji, Coupled FPGA/ASIC Implementation of Elliptic Curve Crypto-Processor, International Journal of Network Security & Its Applications. 2 (2010).

[9] V. S. Miller, Uses of Elliptic Curves in Cryptography, In Advances in Cryptology-CRYPTO’85 Proceedings, Crypto 1985. 218 (1986) 417-426.

[10] G. L. Mullen and D. Panario, Handbook of Finite Fields, First ed., Chapman and Hall/CRC (2013).

[11] K. R. C. Pillai and M. P. Sebastian, Elliptic Curve Based Authenticated Session Key Establishment Protocol for High Security Applications in Constrained Network Environment, International Journal of Network Security and Its Applications (IJNSA). 2 (2010).

[12] J. H. Silverman, The Arithmetic of Elliptic Curves, Graduate Texts in Mathematics. 106 (2009).

[13] L. D. Singh and K. M. Singh, Implementation of Text Encryption Using Elliptic Curve Cryptography, Procedia Computer Science. 54 (2015) 73-82.

[14] CH. Suneetha, D. S. Kumar and A. Chandrasekhar, Secure Key Transport in Symmetric Cryptographic Protocols Using Elliptic Curves Over Finite Fields, International Journal of Computer Applications. 36 (2011).

[15] L. C. Washington, Elliptic Curves Number Theory and Cryptography, Second ed., Chapman and Hall/CRC (2008).