Advances in Computer Science and Information Technology (ACSIT)
Print ISSN: 2393-9907; Online ISSN: 2393-9915; Volume 2, Number 2; January-March, 2015 pp. 144-150
© Krishi Sanskriti Publications
http://www.krishisanskriti.org/acsit.html

Proposed Model for Network Security Issues using ECC Based Threshold Cryptography

Garima Kaushik[1], Amit Kathuria[2] and Samruddha Patil[3]

[1]Graduate (B.Tech – CSE) BPIT, GGSIPU, Delhi
[2]Graduate (Masters – ECE) University of Southern California Los Angeles, CA
[3]Student(B.Tech-CSE) BPIT, GGSIPU, Delhi
E-mail: [1]garimakaushik13@gmail.com, [2]k.amit764@gmail.com, [3]samruddha1401@gmail.com

Abstract—*Elliptic Curve Cryptography (ECC) plays an important role in today's public key based security systems. ECC is a faster and more secure method of encryption as compared to other Public Key Cryptographic algorithms. This paper focuses on the performance advantages of using ECC in the wireless network. Firstly, in this paper its algorithm has been implemented and analyzed for various bit length inputs. To increase the level of security further, a new concept of threshold cryptography is introduced and implemented based on the already implemented ECC algorithm. In threshold cryptography we not only rely on one person but on several people for the decryption of our message.*

1. INTRODUCTION

In a secure cryptosystem, an individual should be able to send an encrypted message to an organization without knowing the public key for every person within the receiving company. The destination organization should be able to set up its own security policyto determine who can read the messages it receives. The cryptosystem is designed such that sender cannot circumvent the security policy and the individual can send the message without knowing the policy.

2. ELLIPTIC CURVE CRYPTOGRAPHY (ECC)

A: Basics of Elliptic Curve

Elliptic Curve (EC) systems as applied to cryptography were first proposed in 1985 independently by Neal Koblitz and Victor Miller.[1]ECC was basically designed to run on small, constrained devices especially embedded devices which have less storage space capacity, less processing capabilities, less power consumption.

Elliptic Curve has a unique property which is why it is used in cryptography.This uniquenessenables us to take any two points on a specific curve, add them together, and get a third point on the same curve .The problem lies that which two points were added together to obtain the third point.[2] [3]

Basic Concepts

An elliptic curve is defined by an equation in two variables, with coefficient. For the purpose of cryptography, the variable and coefficient are limited to a special kind of set called a FINITE FIELD. [3] The general equation for an elliptic curve is:

$$y^2 + axy + by = x^3 + cx^2 + dx + e$$

wherea,b,c,d and e are real numbers and x, y take values from real number .

ECC is considered as the one which has the highest security quality in per bit key among current public key cryptosystems. It's characterized by small key, small system parameter, small public key, saving bandwidth, fast implementation, low power, and low hardware requirements. [4]. ECC is "an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields" [5].For the purpose of cryptography the variables and coefficients are limited to a special kind of set called a FINITE FIELD.

The security of ECC depends on the difficulty of Elliptic Curve Discrete Logarithm Problem (ECDLP). Let P and Q be the two points on the curve such that kP = Q, where k is a scalar and is called elliptic curve discrete logarithm of Q to the base P. Given P and Q, it is computationally infeasible to obtain k, if k is sufficiently large. [6]

The implementation of ECC mainly relies on the operations at three levels: the scalar multiplication, the point addition / doubling, and the finite field modulo arithmetic. The ECC system based on $GF(2n)$ is widely utilized for its simple field arithmetic and efficient scalar multiplication algorithms. Two different coordinates: the affine coordinate and the projective coordinate can be used for the ECC where the curve is defined over $GF(2n)$. It was shown in [3][7][8] that the projective coordinate is more desirable for hardware implementation because it avoids the costly field inversion operation

3. THRESHOLD CRYPTOGRAPHY

Threshold cryptography involves sharing of a key by multiple individuals engaged in encryption or decryption or splitting of message either before or after encryption. It avoids trusting and engaging only one node for doing the job. Hence, the primary objective is to share this authority in sucha way that each individual node performs computation on the message without revealing any secret information about its partial key or the partial messgae.

Another objective is to have distributed architecture in a hostile environment. A certain number of nodes called threshold t, are required to encrypt or decrypt a message. Let n be the number of parties. Such a system is called *(t,n)*-threshold, if at least *t* of these parties can efficiently decrypt the cipher text, while less than *t* have no useful information. Similarly it is possible to define *(t,n)*-threshold signature scheme, where at least *t* parties are required for creating a signature. So, the Threshold schemes generally involve key generation, encryption, share generation, share verification, and share combining algorithms.Thus, the TC enhances security till compromising nodes are less than t since It is difficult to decode partial messages if the number is less than the threshold.

Threshold cryptography achieves the needs such as confidentiality and integrity against malicious nodes. It also provides data integrity and availability in a hostile environment and can also employ verification of the correct dat sharing. All this is achieved without revealing the secret key. [10]

4. SHAMIR'S SECRET SHARING SCHEME

A secret sharing scheme is a means for *n* parties to carry *shares* or *parts* *si*of a message *s*, called the *secret*, such that the complete set *s*1*, . . .sn*of the parts determines the message. The secret sharing scheme is said to be *perfect* if no proper subset of shares leaks any information regarding the secret.

Multiple party secret sharing. Let *s*be a secret to be shared among *n* parties. Generate the first

n − 1 shares *s*1*, . . . , sn*−1 at random and set

$$s_n = s - \sum_{i=1}^{n-1} .$$

The secret is recovered as

$$s = \sum_{i=1}^{n} s_i .$$

A (*t, n*) *threshold* secret sharing scheme is a method for *n* parties to carry shares *si*of a message *s*such that any *t* of the them to reconstruct the message, but so that no *t* – 1 of them can easy do so. The threshold scheme is *perfect* if knowledge of *t* − 1 or fewer shares provides no information regarding *s*.

Shamir's (*t, n*)-threshold scheme. A scheme of Shamir provide an elegant construction of a perfect (*t, n*)-threshold scheme using a classical algorithm called Lagrange interpolation. First we introduce Lagrange interpolation as a theorem.In Lagrange Interpolation , we have tdistinct points ($x_{i\,,}\, y_i$) of the form (xi , f(xi)) , where f(x) is a polynomial of degree less than t , then f(x) is given by :

$$f(x) = \sum_{i-1}^{t} y_i \prod_{\substack{1 \le j \le t \\ i \ne j}} \frac{x - x_j}{x_i - x_j}.$$

In Shamir Secret Sharing scheme , we put a_0 = M (message) and a_1........a_{t-1} coefficients are set randomly for this polynomial. We get

$$f(x) = \sum_{k=0}^{t-1} a_k x^k ,$$

Now in order to generate the shares we put different values of x. We then get n number of f(x) values which are our shares. In general , we compute f(i) , where $1 \le i \le n$. The shares (i , f(i)) are then distributed to different parties.

Now for the recovery of the message , our constant term a_0 holds our message .We can easily get our message back from t number of shares (i ,f(i)) by putting i=0 , that is

f(0) = a_0 = M

$$s = \sum_{i \in I} c_i f(i), \text{ where each } c_i = \prod_{\substack{j \in I \\ j \ne i}} \frac{i}{j - i}.$$

5. ALGORITHMS/ METHODOLOGY USED

i) ECC algorithm

At the Sender End –

Step 1 - The sender will take a point P on the elliptic curve equation given above.

Step 2 – A random number'd' is selected within the range of 1-(n-1). 'd' is the private key.

Step 3 – The sender will generate a public key Q by private key and point P.

Q = d*P

Step 4 – The message to be sent has point 'M' on curve E.

Step 5 – Randomly select 'k' from 1 to (n-1).

Step 6 – Generate two cipher text strings C1 and C2.

$$f(x) = \sum_{k=0}^{t-1} a_k x^k,$$

C1 = k *P and C2 = M+K*Q

Step 7 – Send C1 and C2. C1 and C2 are encrypted texts.

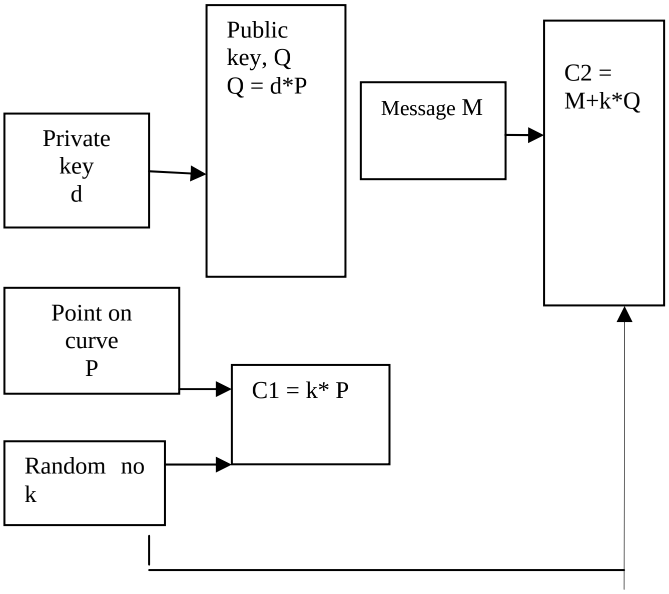


Fig. 1: Encryption at sender site

At the Receiver End –

Step 1 – The receiver uses the cipher texts C1 and C2 to decrypt the message M.

Step 2 – The receiver uses the private key to decrypt the message M.

Step 3 – The receiver has private key 'd'.

M = C2 – d*C1

Step 4 – 'M' is the original message.

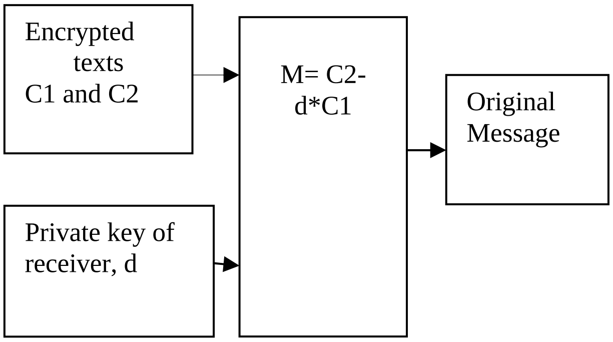


Fig. 2: Decryption at the receiver site

ii) Threshold Cryptography Algorithm

For generating shares :-

1. Sender enters the message M.
2. Sender also enters the number of shares (say n) he wants to generate and the minimum number (threshold number say t) of shares he want to use to recover the message back.
3. Using Lagrange interpolation algorithm, we compute a lagrange polynomial

   (that is f(x)) whose constant term is our message M that is a_0 = M . Other constant terms (a_1, a_2 , ….. , a_{t-1}) are randomly set .The degree of this polynomial is one less than the threshold number that is t-1.
4. Now we generate shares by substituting x with the values 1 to n.
5. Now we have our n number of shares in the form (i , f(i)) where $1 \le i \le n$
6. We send these shares to n different number of people.

For recovering the message :-

1. In order to recover the message M , we atleast need threshold number of shares.
2. We combine these shares using a formula

$$s = \sum_{i \in I} c_i f(i), \text{ where each } c_i = \prod_{\substack{j \in I \\ j \neq i}} \frac{i}{j-i}.$$

   and as the constant term of our lagrange polynomial was our message M, we substitute x with 0 as f(0) = M.

3. Finally, we get our message back.

iii) Threshold cryptography based ecc algorithm

At the sender's end :-

1. The sender will take a point P on the elliptic curve equation given .

2. A random number‘d’ is selected within the range of 1- (n-1). ‘d’ is the private key.
3. The sender will generate a public key Q by private key and point P.

   Q = d*P
4. Now the sender enters the message M.
5. This message is then processed by the threshold cryptographic algorithm in order to generate the shares.
6. Now sender enters the number of shares (say n) he wants to generate and the minimum number (threshold number say t) of shares he want to use to recover the message back.
7. Using Lagrange interpolation algorithm, we compute a lagrange polynomial (that is f(x)) whose constant term is our message M that is a_0= M . Other constant terms (a_1, a_2 , ….. , a_{t-1}) are randomly set .The degree of this polynomial is one less than the threshold number that is t-1.

$$f(x) = \sum_{k=0}^{t-1} a_k x^k,$$

8. Now we generate shares by substituting x with the values 1 to n.
9. Now we have our n number of shares in the form (i , f(i)) where 1≤i≤n.
10. Now we encrypt theses individual shares using the Elliptic curve cryptography algorithm.
11. For each share we generate two cipher texts C1 and C2 with the help of another random number (that is K).
12. C1 is generated as : C1 = K.P
13. C2 is generated as : C2 = M + K.Q
14. As we can see in the above steps , we are multiplying a scalar K with co-ordinate points (X,Y). This is not a normal multiplication. This process involves point multiplication and point addition algorithms. An algorithm called point doubling is also used.
15. Now these cipher texts are transmitted to n number of people.

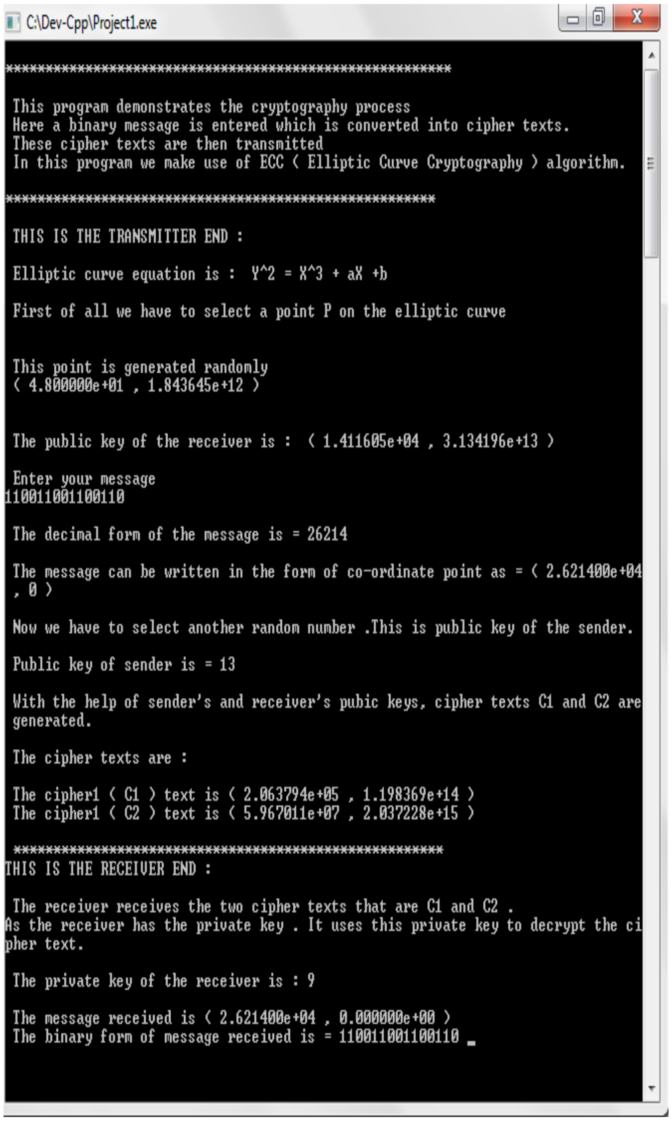
For recovering message back :-

1. Now n people receive encrypted shares.
2. These encrypted shares are then decrypted using ECC decryption algorithm.
3. For each share we have two cipher texts C1 and C2.
4. Each person decrypts the share using the private key that is *d* by
   a. Share = C2 – d.C1
5. In this way all the shares can be decrypted.
6. Now inorder to recover the original message M , we atleast need threshold number of shares.
7. We combine these shares using a formula and as the constant term of our lagrange polynomial was our message M, we substitute x with 0 as f(0) = M.
8. Finally, we get our message back.

$$s = \sum_{i\in I} c_i f(i), \text{ where each } c_i = \prod_{\substack{j\in I\\ j\neq i}} \frac{i}{j-i}.$$

6. OUTPUT

i) ECC algorithm



```
*********************************************************

 This program demonstrates the cryptography process
 Here a binary message is entered which is converted into cipher texts.
 These cipher texts are then transmitted
 In this program we make use of ECC ( Elliptic Curve Cryptography ) algorithm.

*******************************************************

 THIS IS THE TRANSMITTER END :

 Elliptic curve equation is :  Y^2 = X^3 + aX +b

 First of all we have to select a point P on the elliptic curve


 This point is generated randomly
 ( 4.800000e+01 , 1.843645e+12 )


 The public key of the receiver is :  ( 1.411605e+04 , 3.134196e+13 )

 Enter your message
110011001100110

 The decimal form of the message is = 26214

 The message can be written in the form of co-ordinate point as = ( 2.621400e+04
 , 0 )

 Now we have to select another random number .This is public key of the sender.

 Public key of sender is = 13

 With the help of sender's and receiver's pubic keys, cipher texts C1 and C2 are
 generated.

 The cipher texts are :

 The cipher1 ( C1 ) text is ( 2.063794e+05 , 1.198369e+14 )
 The cipher1 ( C2 ) text is ( 5.967011e+07 , 2.037228e+15 )

 *******************************************************
THIS IS THE RECEIVER END :

 The receiver receives the two cipher texts that are C1 and C2 .
As the receiver has the private key . It uses this private key to decrypt the ci
pher text.

 The private key of the receiver is : 9

 The message received is ( 2.621400e+04 , 0.000000e+00 )
 The binary form of message received is = 110011001100110 _
```

Fig. 3: Output of 15bit number

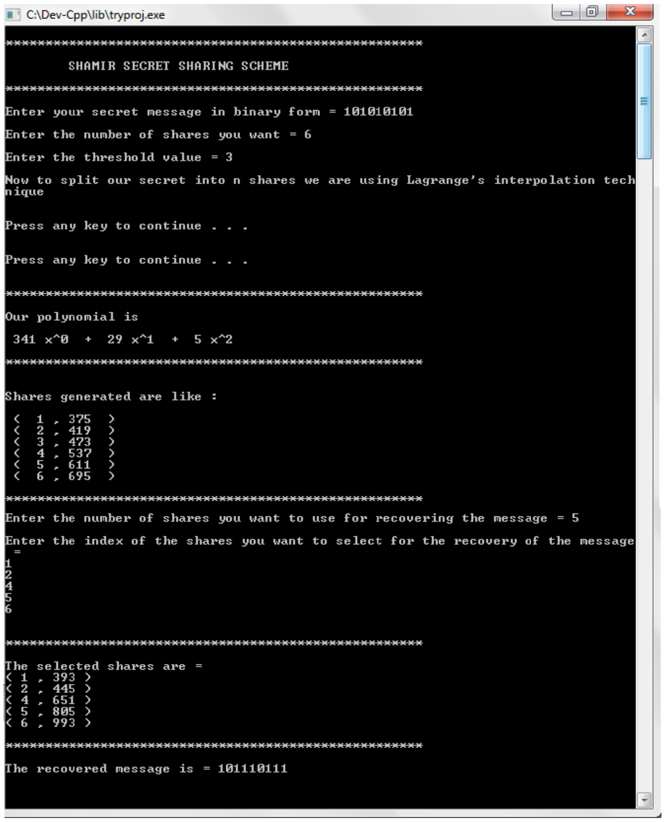
Table 1: Encryption at the sender end :

POINT ON CURVE (P)	PRIVATE KEY (d)	ORIGINAL MESSGAE (M)	RANDOM NO (k)	CIPHER TEXT (C1)	CIPHER TEXT (C2)
(14368,1.80201e+006)	17	0	23	(5380.57,-510939)	(98.5765,-44140.4)
(14227,1.77708e+006)	14	1	14	(72216.8,1.944e+007)	(92.5932,-42562.4)
(14505,1.82636e+006)	16	10	6	(9469.52,1.017e+006)	(5259.61,498212)
(14622,1.84725e+006)	18	11	2	(2763.97,274246)	(63.851,-34778.6)
(14276,1.87487e+006)	11	101	21	(48105.6,-1.06e+007)	(2477.67,252020)
(14959,1.9079e+006)	10	111	13	(147.409,53907.5)	(398.062,87918.4)
(15089,1.93149e+006)	2	1011	9	(89.7893,42145.3)	(53840.4, -1.25311e+007)
(15194,1.95062e+006)	15	1111	10	(23138.5, -3.58342e+006)	(13243.4, -1.60426e+006)
(15605,2.02618e+006)	12	11011	22	(3720.01,-352533)	(117076, -4.00739e+007)
(15716,2.04676e+006)	8	11101	21	(3.17266,-8992.35)	(18309.4, 2.54292e+0006)
(15850,2.07171e+006)	4	101010	1	(15850,2.07171e+006)	(302.217,-71606.6)
(16017,2.10296e+006)	10	110011	23	(6854.3,675386)	(2195.48,-227677)
(16147,2.12741e+006)	2	1101011	9	(854.854,131790)	(5210.94,-482529)
(16432,2.18136e+006)	11	11001100	10	(179.994,59549.7)	(2810.77,-261485)
(16670,2.22679e+006)	19	111100001	16	(2154.07,228380)	(350456, 2.07057e+008)
(16918,2.27448e+006)	14	1010101011	14	(59.1795,-34305)	(531019, 3.86226e+008)

Table 2: Decryption at the receiver end :

CIPHER TEXT (C1)	CIPHER TEXT (C2)	PRIVATE KEY (d)	MESSAGE RECEIVED
(5380.57,-510939)	(98.5765,-44140.4)	17	0
(72216.8,1.944e+007)	(92.5932,-42562.4)	14	1
(9469.52,1.017e+006)	(5259.61,498212)	16	10
(2763.97,274246)	(63.851,-34778.6)	18	11
(48105.6,-1.06e+007)	(2477.67,252020)	11	101
(147.409,53907.5)	(398.062,87918.4)	10	111
(89.7893,42145.3)	(53840.4,-1.25311e+007)	2	1011
(23138.5,-3.58342e+006)	(13243.4,-1.60426e+006)	15	1111
(3720.01,-352533)	(117076,-4.00739e+007)	12	11011
(3.17266,-8992.35)	(18309.4,2.54292e+0006)	8	11101
(15850,2.07171e+006)	(302.217,-71606.6)	4	101010
(6854.3,675386)	(2195.48,-227677)	10	110011
(854.854,131790)	(5210.94,-482529)	2	1101011
(179.994,59549.7)	(2810.77,-261485)	11	11001100
(2154.07,228380)	(350456,2.07057e+008)	19	111100001
(59.1795,-34305)	(531019,3.86226e+008)	14	1010101011

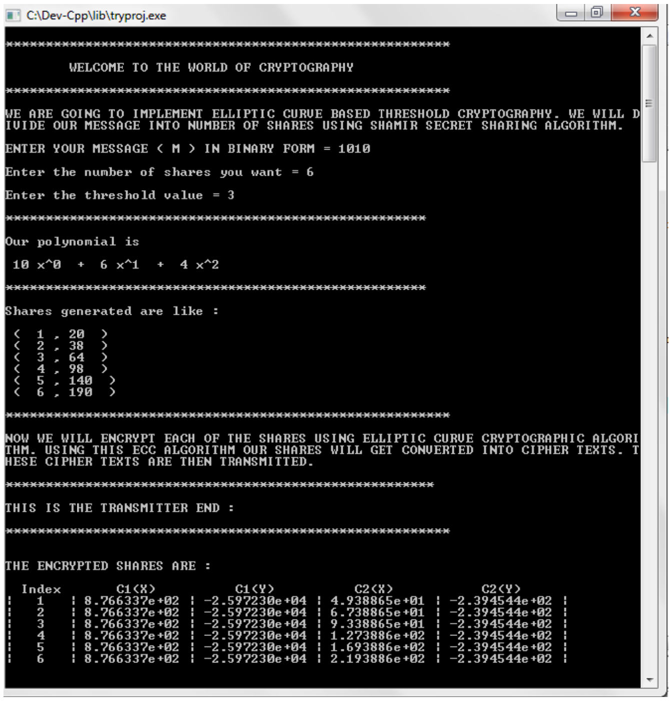
ii) Threshold Cryptography Algorithm



```
*****************************************************

        SHAMIR SECRET SHARING SCHEME

*****************************************************

Enter your secret message in binary form = 101010101

Enter the number of shares you want = 6

Enter the threshold value = 3

Now to split our secret into n shares we are using Lagrange's interpolation tech
nique


Press any key to continue . . .


Press any key to continue . . .


*****************************************************

Our polynomial is

 341 x^0  +  29 x^1  +  5 x^2

*****************************************************


Shares generated are like :

 (  1 , 375  )
 (  2 , 419  )
 (  3 , 473  )
 (  4 , 537  )
 (  5 , 611  )
 (  6 , 695  )

*****************************************************

Enter the number of shares you want to use for recovering the message = 5

Enter the index of the shares you want to select for the recovery of the message
 =
1
2
4
5
6


*****************************************************

The selected shares are =
( 1 , 393 )
( 2 , 445 )
( 4 , 651 )
( 5 , 805 )
( 6 , 993 )

*****************************************************

The recovered message is = 101110111
```

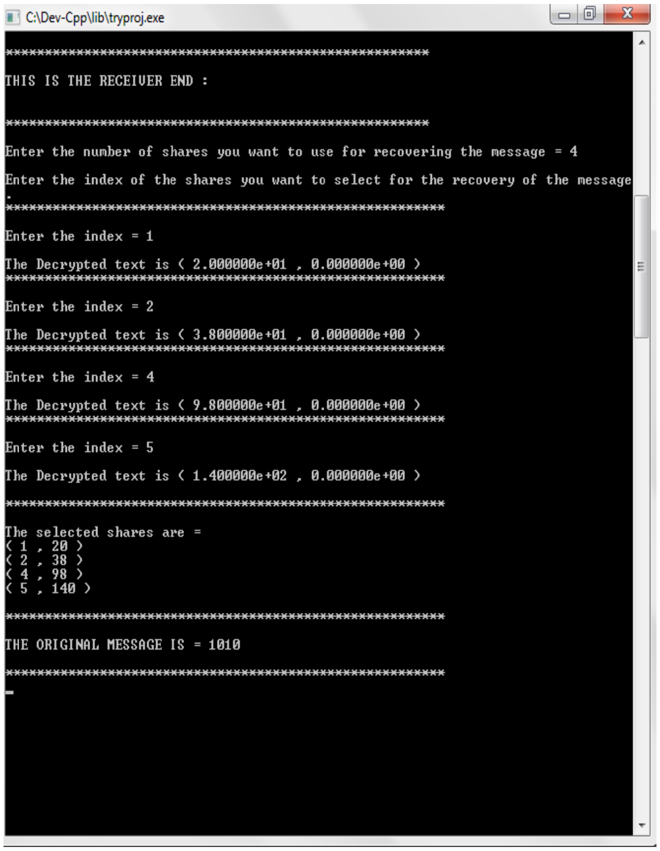
Fig. 4: Output of 9 bit number

iii) Threshold cryptography based ecc algorithm



```
*********************************************************

        WELCOME TO THE WORLD OF CRYPTOGRAPHY

*********************************************************

WE ARE GOING TO IMPLEMENT ELLIPTIC CURVE BASED THRESHOLD CRYPTOGRAPHY. WE WILL D
IVIDE OUR MESSAGE INTO NUMBER OF SHARES USING SHAMIR SECRET SHARING ALGORITHM.

ENTER YOUR MESSAGE ( M ) IN BINARY FORM = 1010

Enter the number of shares you want = 6

Enter the threshold value = 3

******************************************************

Our polynomial is

 10 x^0  +  6 x^1  +  4 x^2

******************************************************

Shares generated are like :

 (  1 , 20  )
 (  2 , 38  )
 (  3 , 64  )
 (  4 , 98  )
 (  5 , 140  )
 (  6 , 190  )

*********************************************************

NOW WE WILL ENCRYPT EACH OF THE SHARES USING ELLIPTIC CURVE CRYPTOGRAPHIC ALGORI
THM. USING THIS ECC ALGORITHM OUR SHARES WILL GET CONVERTED INTO CIPHER TEXTS. T
HESE CIPHER TEXTS ARE THEN TRANSMITTED.

*******************************************************

THIS IS THE TRANSMITTER END :

*********************************************************


THE ENCRYPTED SHARES ARE :

  Index        C1(X)          C1(Y)           C2(X)            C2(Y)
|   1    | 8.766337e+02 | -2.597230e+04 | 4.938865e+01 | -2.394544e+02 |
|   2    | 8.766337e+02 | -2.597230e+04 | 6.738865e+01 | -2.394544e+02 |
|   3    | 8.766337e+02 | -2.597230e+04 | 9.338865e+01 | -2.394544e+02 |
|   4    | 8.766337e+02 | -2.597230e+04 | 1.273886e+02 | -2.394544e+02 |
|   5    | 8.766337e+02 | -2.597230e+04 | 1.693886e+02 | -2.394544e+02 |
|   6    | 8.766337e+02 | -2.597230e+04 | 2.193886e+02 | -2.394544e+02 |
```

Fig. 5: Threshold based ECC for input 1010



```
******************************************************

THIS IS THE RECEIVER END :

******************************************************

Enter the number of shares you want to use for recovering the message = 4

Enter the index of the shares you want to select for the recovery of the message
.
********************************************************

Enter the index = 1

The Decrypted text is ( 2.000000e+01 , 0.000000e+00 )
********************************************************

Enter the index = 2

The Decrypted text is ( 3.800000e+01 , 0.000000e+00 )
********************************************************

Enter the index = 4

The Decrypted text is ( 9.800000e+01 , 0.000000e+00 )
********************************************************

Enter the index = 5

The Decrypted text is ( 1.400000e+02 , 0.000000e+00 )

********************************************************

The selected shares are =
( 1 , 20 )
( 2 , 38 )
( 4 , 98 )
( 5 , 140 )

********************************************************

THE ORIGINAL MESSAGE IS = 1010

********************************************************
_
```

Fig. 6: Threshold based ECC for input 1010 continued

Analysis of Output-

We have plot two graphs. One for the shares generated for the input 1010 and one graph illustrating the encrypted shares which are encrypted using ECC algorithm. It took 9 seconds for the algorithm to generate this output.

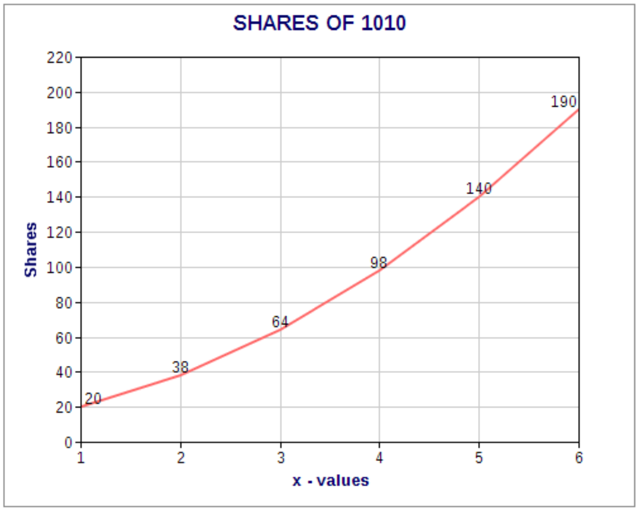


Fig. 7: Generated shares of 1010

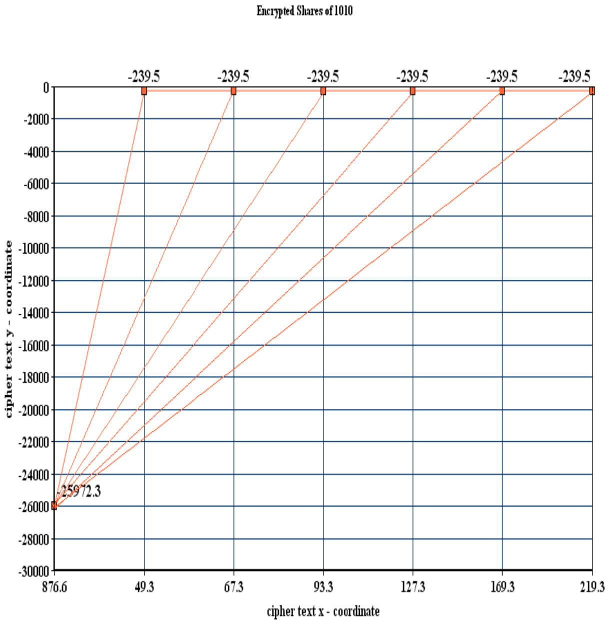


Fig. 8: Encrypted shares for 1010

7. CONCLUSION

The digital signature based Elliptic Curve Cryptography and Threshold covers all four aspects of security - Integrity, Authentication,Non-repudiation and Confidentiality.

They promise for the better and secure data transmission is opening new dimensions of its application in every field of communication. Mobile computing, wireless sensor net-works, server based encryption, image encryption, government and financial communication protocols and many other.

8. FUTURE WORK

This is a completely new domain and has tremendous scope of research. ECC can be used to provide authentication and enhanced security. ECC can be used in Mobile computing, wireless sensor net-works, server based encryption, image encryption, government and financial communication protocols and many other areas.

We are going to further introduce the concept of Cloud Computing with ECC. We are going to get the biometric values and then encrypt them using our Elliptic Curve Cryptography based Threshold cryptographic algorithm. Then we will store these encrypted values in our cloud and whenever we want , we can retrieve them from our cloud , decrypt them and get the original values.

REFERENCES

[1] An Elliptic Curve Cryptography Primer, Certicom “Catch the curve” White paper series, June 2004

[2] Elliptic Curve Cryptography A new way for Encryption, Rahim Ali , Department of Computer Science and Engineering, Bahria University, 13 National Stadium Road Karachi

[3] IEEE Standard P 1363-2000, IEEE standard specification for public key cryptography’, Aug2000.

[4] The Study and Application of Elliptic Curve Cryptography Library on Wireless Sensor Network, 2008 11th IEEE International Conference on Communication Technology Proceedings.

[5] D. R. Hankerson, S. A. Vanstone, and A. J. Menezes, *Guide to Elliptic Curve Cryptography:*Springer, 2004.

[6] “Design of a Private Credentials Scheme Based on Elliptic Curve Cryptography” , 2009 First International Conference on Computational Intelligence, Communication Systems and Networks.

[7] G. Agnew, R. Mullin, and S. Vanstone, “On the development of a fast elliptic curve processor chip”, *Advances in Cryptology CRYPTO’91*, pp. 482-487, New York, Springer-Verlag, 1991.

[8] D. V. Chudnovsky and G. V. Chudnovsky, “Sequences of numbers generated by addition in formal groups and new primality and factorization tests.” *Advances in Applied Mathematics* vol. 7, no. 4, pp. 385-434, May 1986.

[9] “New Low Complexity Key Exchange and Encryption protocols for Wireless Sensor Networks Clusters based on Elliptic Curve Cryptography”, NRSC2009.

[10] “ECC based threshold cryptography Implementation for MANETs” by Levent Ertaul and Nitu J. Chavan.