# Probabilistic Encryption Based ECC Mechanism

**A.V.N.Krishna**
Pujyasri Madhavanji College of Engg. & Tech., Hyderabad, A.P., India.
hari_avn @rediffmail.com

## Abstract

Elliptic Curve Cryptography provides a secure means of exchanging keys among communicating hosts using the Diffie Hellman Key Exchange algorithm. Encryption and Decryption of texts and messages have also been attempted. In the paper[15], the authors presented the implementation of ECC by first transforming the message into an affine point on the EC, and then applying the knapsack algorithm on ECC encrypted message over the finite field GF(p). The kanp sack problem is not secure in the present standards and more over in the work the authors in their decryption process used elliptic curve discrete logarithm to get back the plain text. This may form a computationally infeasible problem if the values are large enough in generating the plain text. In the present work the output of ECC algorithm is provided with probabilistic features which make the algorithm free from Chosen cipher text attack. Thus by having key lengths of even less than 160 bits, the present algorithm provides sufficient strength against crypto analysis and whose performance can be compared with standard algorithms like RSA.

***Key words:*** *ECC, Probabilistic encryption, Multiple ciphers, chosen cipher text attack.*

## 1. Introduction

Historically, encryption schemes were the first central area of interest in cryptography[18]. They deal with providing means to enable private communication over an insecure channel. A sender wishes to transmit information to a receiver over an insecure channel that is a channel which may be tapped by an adversary. Thus, the information to be communicated, which we call the plaintext, must be transformed (encrypted)to a cipher text, a form not legible by anybody other than the intended receiver. The latter must be given some way to decrypt the cipher text, i.e. retrieve the original message, while this must not be possible for an adversary. This is where keys come into play; the receiver is considered to have a key at his disposal, enabling him to recover the actual message, a fact that distinguishes him from any adversary. An encryption scheme consists of three algorithms: The encryption algorithm transforms plaintexts into cipher texts while the decryption algorithm converts cipher texts back into plaintexts. A third algorithm, called the key generator, creates pairs of keys: an encryption key, input to the encryption algorithm, and a related decryption key needed to decrypt. This work mainly deals with the algorithm which generates sub keys which provides sufficient strength to the encryption mechanism.

Any symmetric encryption scheme uses a private key for secure data transfer. In their work on “ A simple algorithm for random number generation [7], the authors presented a simple algorithm which generates random numbers. In [8-11] the authors presented a probabilistic algorithm which generates multiple cipher texts for one plain text which is relatively free from chosen cipher text attack.

Most of the products and standards that use public-key cryptography for encryption and digital signatures use RSA today. Recently, Elliptic Curve Cryptography [1-6, 12-20] has begun to challenge RSA. The principal attraction of ECC, compared to RSA, is that it appears to offer better security for a smaller key size, thereby reducing processing overhead. Elliptic curve cryptography makes use of elliptic curves in which the variables and coefficients are all restricted to elements of a finite field. In ECC we normally start with an affine point called Pm(x,y). These points maybe the Base point (G) itself or some other point closer to the Base point. Base point implies it has the smallest x,y co-ordinates, which satisfy the EC. A character in a message is first transformed into an affine point of the elliptic curve by using it as a multiplier of Pm. That is, if the ASCII value of a character is A, then we determine P0 m=A(Pm). This is one step towards introducing sophistication and complexity in the encryption process. The newly evaluated P0 m is a point on the EC, determined by applying the addition and doubling strategy of ECC technique. Then as per ECC algorithm, P0 m is added with kPB, where k is randomly chosen secret integer and PB is the public key of user B, to yield (P0 m+kPB). This now constitutes second part of the encrypted version of the message. The other part, namely, kG, which is the product of the secret integer and the Base point, constitutes the first part. Thus the encrypted message is now made up of two sets of coordinates, namely, (kG, P0 m+ kPB). In this paper we have assigned kG=(x1,y1) and (Pm+kPB)=(x2,y2). Not satisfied with the complexity involved in determining the encryption, we wish to introduce further complexity by applying time stamp, a variable nonce value concept to the encrypted version. The whole idea behind these rigorous exercises is to make decryption totally impossible, even if the Base Point G, secret integer k, the affine Point Pm is known to the crypt analyst. Now to recover the information from the encrypted version, first the new model with time-stamp has to be reversed. Then we apply the decryption process of ECC, by applying the private key of recipient (nB) on the first element (kG). This is subtracted from the second element to recover P0 m. This promises to afford maximum security from intruders and hackers. Another public key algorithm, namely RSA, is used to encrypt/decrypt the same message. Unlike the ECC procedure, this yields only one integer for each character of the message. The time and space implications for both the schemes are discussed and analyzed. The paper justifies that despite the harsher requirements of time and space for the ECC methods, it is far superior due to the resistance it offers to any brute force attack.

Some recent works on application of ECC are cited here.Aydos et al. [1] Discusses the results of implementation of ECC over the field GF(p) on an 80 MHz, 32 bit RAM microprocessor. Kristin et al. [8] provides an overview of ECC for wireless security. It focuses on the performance advantages in the wireless environment by using ECC instead of the traditional RSA cryptosystem. Ray et al. [3] explains the design of generator, which automatically produces a customized ECC hardware that meets user-defined requirements. Cilardo et al. [4] explains the engineering of ECC as a complex interdisciplinary research field encompassing such fields as mathematics, computer science and electrical engineering. [2] presents a high performance EC cryptographic process for general curves over GF(p). The standard specifications for public key cryptography are defined in [5 ].

## 2. Proposed Method Description

The Weiestrass equation defining an elliptic curve over GF(p), for $q > 3$, is as follows: :

$$y2 = x3 + ax + b, \qquad (1)$$

where x, y are elements of GF(p), and a, b are integer modulo p, satisfying

$$4a3 + 27b2\ 6= 0 \bmod p \qquad (2)$$

Here p is known as modular prime integer. An elliptic curve E over GF(p) consist of the solutions (x, y) defined by Equations (1) and (2), along with an additional element called O, which is the point of EC at infinity. The set of points (x, y) are said to be affine coordinate point representation. The basic Elliptic curve operations are point addition and point doubling. Elliptic curve cryptographic primitives [13] require scalar point multiplication. Say, given a point P(x, y) on an EC, one needs to compute kP, where k is a positive integer. This is achieved by a series of doubling and addition of P. Say, given k = 386, entails the following sequence of operations P, 2P, 3P, 6P, 12P, 24P, 48P, 96P, 192P, 193P, 386P.
Let us start with P(xP , yP ). To determine 2P, P is doubled. This should be an affine point on EC. Use the following equation, which is a tangent to the curve at point P.

$S = [(3x2P + a)/2yP\ ] \bmod p.$
Then 2P has affine coordinates xR, yR given by

$xR = (S2 - 2xP\ ) \bmod p,$
$yR = [S(xP - xR) - yP\ ] \bmod p.$

Now to determine 3P, we use addition of points P and 2P, treating 2P = Q. Here P has coordinates (xP , yP ) and Q = 2P has coordinates (xQ, yQ). Then

$xR = (S2 - xP - xQ) \bmod p,$
$yR = (S(xP - xR) - yP\ ] \bmod p.$

Therefore we apply doubling and addition depending on a sequence of operations determined for k. Every point xR, yR evaluated by doubling or addition is an affine point (points on the Elliptic Curve).

We add probabilistic features to output of ECC which generates multiple cipher texts for one plain text which makes the cipher text free from Chosen Cipher text attack.

### *2.1 Implementation Details of the Proposed Algorithm*

Once the defining EC is know, we can select a base point called G. G has [x, y] coordinates which satisfy the equation $y^2 = x^3 + ax + b$. The Base point has the smallest x, y values which satisfy the EC.

The ECC method requires that we select a random integer k(k < p), which needs to be kept secret. Then kG is evaluated, by a series of additions and doublings, as discussed above. For purpose of this discussion we shall call the source as host A, and the destination as host B. We select the private key of the host B, called nB. k and nB can be generated by random number generators to give credibility. That would be digressing away from the main discussion. Hence we make suitable assumptions for these two parameters. The public key of B is evaluated by PB = nBG. (3)

Suppose A wants to encrypt and transmit a character to B, he does the following. Assume that host A wants to transmit the character ‘S’. Then the ASCII value of the character ‘S’ is used to modify Pm as follows:

P0m= SPm

Pm we said is an affine point. This is selected different from the Base point G, so as to preserve their individual identities. P0m is a point on the EC. The coordinates of the P0m should fit into the EC. This transformation is done for two purposes. First the single valued ASCII is transformed into a x,y co-ordinate of the EC. Second it is completely camouflaged from the would-be hacker. This is actually intended to introduce some level of complexity even before the message is encrypted according to ECC. As the next step of ECC, we need to evaluate kPB, here PB is a public key of user B. Determining this product involves a series of doubling and additions, depending on the value of k. For a quick convergence of the result, we should plan for optimal number of doubles and additions. The encrypted message is derived by adding P0m with kPB, that is, P0m+kPB. This yields a set of x2, y2 coordinates. Then kG is included as the first element of the encrypted version. kG is another set of x1, y1 coordinates. Hence the entire encrypted version for purposes of storing or transmission consists of two sets of coordinates as follows: Cm = (kG, P0m + kPB) kG −! x1, y1 P0 m + kPB −! x2, y2.

### *2.2 Probabilistic features*

Probabilistic features are added to output of ECC to make free from Chosen Cipher text attack [7-11]. It involves following steps.

1. Algorithm for generating sequence.
2. Generating Basins with unequal values based on equality of values.
3. Mapping the basins to output of ECC.

#### *2.2.1 Algorithm for generating the sequence*

1. Consider the sequence for 0 to n values where n is a positive integer.
2. Convert each element of the sequence into ternary form of a given digit number.
3. Represent the values of step 2 in a matrix form of (n+1) * (digit number).
4. Subtract 1 from each element of the matrix specified in step 3.
5. Consider a random matrix key of size (digit number*digit number).
6. Multiply the output of step 4 with the output of step 5.
7. Convert all positive values of matrix to 1, negative values to -1 and zero by 0.
8. Add 1 to each element of output of step 7.

9. Convert ternary values of step 8 into decimal form. A sequence is generated.

### *2.2.2 Algorithm for generating Basins from sequence generated*

1. Consider the sequence of values starting from 0 to n where n be an integer.
2. Read the sequence generated from algorithm 1.
3. Read the starting element of step 1 and store the first element of step 1 and the corresponding first element of step 2 in a separate basin.

**4.1** Compare the element of step 3 with the elements of step 2. If there is a match, store the corresponding elements of step 1 in the basin specified in step 3. Neglect already visited elements.

**4.2** Repeat step 4.1 with the remaining elements of the basin of step 3 and store them in the same basin. This will form one basin.

5. Go to next element of step 1 which is not visited earlier.

## 3. Implementation of the Proposed Algorithm

**3.1:** The Elliptic Curve is y2 mod 487 = (x3 − 5x + 25) mod 487. The base point G is selected as (0, 5). Base point implies that it has the smallest x, y co-ordinates which satisfy the EC. Pm is another affine point, which is picked out of a series of affine points evaluated for the given EC. We could have retained G itself for Pm. However for the purpose of individual identity, we choose Pm to be different from G. Let Pm =(1,316). The choice of Pm is itself an exercise involving meticulous application of the ECC process on the given EC, the secret integer k, and the private key nB of the recipient B. We have at our disposal a series of random number generators. But that would be digressing from the main path of thought. Hence we shall assume that k = 225, and nB = 277. Plaintext is "S", whose ASCII value is 83. Therefore,
PB = nBG = 277(0, 5) = (260, 48)
P0m = 83(1, 316) = (475, 199)
kPB = 225(260, 48) = (212, 151)
P0m + kPB = (475, 199)+ (212, 151) = (51, 58)
kG = 225(0, 5) = (99, 253).
Encrypted version of the message is: ((99, 253),(51, 58)), where x1 = 99, y1 = 253, x2 = 51, and y2 = 58.

**3.2:**
**Step1:**
Consider the sequence for n= 0 to 26 values.
**Step2:**
Convert the sequence to ternary form of a 3 digit number
i.e. 0 ------- 000
1-------- 001
2-------- 002
.

.

.

26-------- 222

**Step3**:

Represent above ternary form in 27x3 matrix

$$R = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 2 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 2 \\ 0 & 2 & 0 \\ 0 & 2 & 1 \\ 0 & 2 & 2 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 2 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 2 \\ 1 & 2 & 0 \\ 1 & 2 & 1 \\ 1 & 2 & 2 \\ 2 & 0 & 0 \\ 2 & 0 & 1 \\ 2 & 0 & 2 \\ 2 & 1 & 0 \\ 2 & 1 & 1 \\ 2 & 1 & 2 \\ 2 & 2 & 0 \\ 2 & 2 & 1 \\ 2 & 2 & 2 \end{bmatrix}$$

**Step 4:**

Subtract 1 from each element of the above matrix and the resulting matrix R is

$$R=\begin{bmatrix} -1 & -1 & -1 \\ -1 & -1 & 0 \\ -1 & -1 & 1 \\ -1 & 0 & -1 \\ -1 & 0 & 0 \\ -1 & 0 & 1 \\ -1 & 1 & -1 \\ -1 & 1 & 0 \\ -1 & 1 & 1 \\ 0 & -1 & -1 \\ 0 & -1 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & -1 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & -1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & -1 & -1 \\ 1 & -1 & 0 \\ 1 & -1 & 1 \\ 1 & 0 & -1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & -1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

**Step5:**

Consider a random matrix

$$A= \begin{bmatrix} 2 & 5 & -6 \\ 3 & 1 & 3 \\ 4 & -2 & -3 \end{bmatrix}$$

**Step6:**

R= R X A

$$R = \begin{bmatrix} -1 & -7 & 1 \\ -7 & -4 & -2 \\ -13 & -1 & -5 \\ 4 & -6 & -1 \\ -2 & -31 & -4 \\ -8 & 0 & -7 \\ 9 & -5 & -3 \\ 3 & -2 & -6 \\ -3 & 1 & -9 \\ 1 & -4 & 5 \\ -5 & -1 & 2 \\ -11 & 2 & -1 \\ 6 & -3 & 3 \\ 0 & 0 & 0 \\ -6 & 3 & -3 \\ 11 & -2 & 1 \\ 5 & 1 & -2 \\ -1 & 4 & -5 \\ 3 & -1 & 9 \\ -3 & 2 & 6 \\ -9 & 5 & 3 \\ 8 & 0 & 7 \\ 2 & 3 & 4 \\ -4 & 6 & 1 \\ 13 & 1 & 5 \\ 7 & 4 & 2 \\ 1 & 7 & -1 \end{bmatrix}$$

**Step7:**

Convert all positive values to 1, negative values to -1 and zero to 0 of the resulting matrix of step 6.

$$R = \begin{bmatrix} -1 & -1 & 1 \\ -1 & -1 & -1 \\ -1 & -1 & -1 \\ 1 & -1 & -1 \\ -1 & -1 & -1 \\ -1 & 0 & -1 \\ 1 & -1 & -1 \\ 1 & -1 & -1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \\ 0 & 0 & 0 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \\ 1 & 1 & -1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \\ -1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \\ -1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & -1 \end{bmatrix}$$

**Step8:** Add 1 to each element of the matrix R

$$R = \begin{bmatrix} 0 & 0 & 2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 2 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 2 & 0 & 0 \\ 2 & 0 & 0 \\ 0 & 2 & 0 \\ 2 & 0 & 2 \\ 0 & 0 & 2 \\ 0 & 2 & 0 \\ 2 & 0 & 2 \\ 1 & 1 & 1 \\ 0 & 2 & 0 \\ 2 & 0 & 2 \\ 2 & 2 & 0 \\ 0 & 2 & 0 \\ 2 & 0 & 2 \\ 0 & 2 & 2 \\ 0 & 2 & 2 \\ 2 & 1 & 2 \\ 2 & 2 & 2 \\ 0 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 0 \end{bmatrix}$$

**Step9:**

Convert each row of the matrix R to decimal form to generate sequence i.e. 0 0 2 will form $0*3^2+0*3^1+2*3^0 = 2$

The sequence formed is = 2 0 0 18 0 3 18 18 6 20 2 6 20 13 6 20 24 6 20 8 8 23 26 8 26 26 24.

**3.3 For example**

1. n [27]= 0 1 2 3 4 5……………………………………………...26
2. r [27] = 2 0 0 18 0 3 18 18 6 20 2 6 20 13 6 20 24 6 20 8 8 23 26 8 26 26 24.
3. Read n[0]=0. Store the values of n[0],r[0] in a basin. ie b(0)=(0,2).

4.1. n[0] ie. '0' is compared with r[27] values. There is a match at r[1],r[2] & r[4]. Neglect already visited elements. Thus b(0)=[0,1,2,4).

4.2 Step 4.1 is repeated with other elements of basin ie. 1, 2 & 4 values. For elements 1& 4, there is no match of values in r[27]. For element 2, there is a match at r[10]. Thus the basin b(0) is updated to (0, 1, 2, 4&10).

5. The procedure is repeated for the next element of the sequence of step 1 which is not visited earlier. The other basins formed are
b(1)=(3,5,6,7,8,9,11,12,14,15,17,18,19,20,21,23)
b(2)=(13)

b(3)=(16,22,24,25,26)

### *3.4 Mapping the basins on to the output of ECC*

Since 4 Basins are formed in the given example, each value of ECC output is represented by a 2 digit number with a base vlaue of 4.

For example
01 is represented as b(0)b(1), 02 as b(0)b(2), 03 as b(0)b(3), 04 asb(1)b(1),......................................................8 as b(2)b(2), 9 as b(2)b(3) and so on.
The output for S from ECC[15] for the given example is (99 253), (51 58). Each of the values like 9 9, 2 5 3 and so on are repaced with random values from the corresponding basins which generates multiple cipher texts from one plain text.
9 is represented as b(2)b(3) Taking random values from basins this can form (13 16) or (13 22) or (13 24) or (13 26). Similarly each value of output of ECC can be mapped to multiple values of basins in random fashion.
This procedure generates multiple cipher texts for plain text.

for S=83 , Cipher text bu ECC = (99 253), (51 58)
=b(2)b(3) b(2)b(3) b(0)b(2) b(1)b(2) b(0)b(3), b(1)b(2) b(0)b(1) b(1)b(2) b(2)b(2).
Cipher text 1= (13 22 10 13 06 13 02 22), (09 13 00 22 15 13 13 13).
Cipher text 2= (13 26 01 13 11 13 10 16), (17 13 10 16 03 13 13 13).
Cipher text 3= (13 16 01 13 05 13 00 22), (20 13 01 16 19 13 13 13).
Similarly by randomly selecting values from basins multiple cipher texts can be generated for one plain text. Any one cipher text is used for transmission.
During the decryption process, every value of generated cipher text is mapped to the basin values which in turn can be converted to output of Encryption of ECC algorithm. By using the private key of the receiver , Plain text can be retrieved back.

## 4. Conclusion & Future Work

ECC itself is a very secure algorithm for encryption. However, not satisfied with it the algorithm is provided with Probabilistic features which help to generate multiple cipher texts for one plain text. The advantage with this model is it is not only free from linear and differential cryptanalysis but also free from chosen cipher text attacks. But the data overhead of the plain text to cipher text is increased by 1:2. Thus the given model supports the important properties like authentication, security and confidentiality at less computing resources when compared with algorithm like RSA. The model can be used not only for text it can also be used for different media like audio, image & video.

## References

[1] M. Aydos, T. Yanik, and C. K. Kog, “High-speed implementation of an ECC-based wireless authentication protocol on an ARM microprocessor,” IEEE Proceedings of Communication, vol. 148, no. 5, pp.273-279, Oct. 2001.

[2] G. Chen, G. Bai, and H. Chen, “A high-performance elliptic curve cryptographic processor for general curves Over GF(p) based on a systolic arithmetic unit,” IEEE Transactions on Circuits System - II: Express Briefs, vol. 54, no. 5, pp. 412-416,May. 2007.

[3] R. C. C. Cheng, N. J. Baptiste, W. Luk, and P¿ Y.K. Cheung, “Customizable elliptic curve cryptosystems,” IEEE Transactions on VLSI Systems, vol. 13, no. 9, pp. 1048-1059, Sep. 2005.

[4] A. Cilardo, L. Coppolino, N. Mazzocca, and L. Romano, “Elliptic curve cryptography engineering,” Proceedings of the IEEE, vol. 94, no. 2, pp. 395-406,Feb. 2006.

[5] W. Diffie, “The first ten years of Public Key cryp-tography,” Proceddings of IEEE, vol. 76, no. 5, pp.560-577, May 1988

[6] K. M. Finnigin, B. E. Mullins, R. A. Raines, and H. B. Potoczny, “Cryptanalysis of an elliptic curve cryptosystem for wireless sensor networks,” International Journal of Security and Networks, vol. 2, no. 3/4, pp. 260-271, 2006.

[7] Krishna A.V.N : A simple algorithm for random number generation, Journal for Scientific & Industrial research, Vol 64, Oct 2005, pp 794-796

[8] Krishna A.V.N., S.N.N.Pandit: A new Algorithm in Network Security for data transmission, Acharya Nagarjuna International Journal of Mathematics and Information Technology, Vol: 1, No. 2, 2004 pp 97-108

[9] Krishna A.V.N., A.Vinaya Babu: Web and Network Communication security Algorithms, Journal on Software Engineering, Vol 1,No.1, July 06, pp12-14

[10] Krishna A.V.N, A.Vinaya Babu: Training of a New Probabilistic Encryption Scheme Using an Optimal Matrix Key, Georgian Electronic & Scientific Journal, 2009, No. 2(19)

[11] Krishna A.V.N etal: A Generalised scheme for data encryption technique using a randomized matrix key, Journal of Discrete Mathemetical sciences and Cryptography, Vol 10(2007) No. 1, pp 73-81

[12] K. Lauter, “The advantages of elliptic cryptography for wireless security,” IEEE Wireless Communications, pp. 62 - 67, Feb. 2006.

[13] J. Lee, H. Kim, Y. Lee, S. M. Hong, and H. Yoon “Parallelized scalar multiplication on elliptic curves defined over optimal extension field,” International Journal of Network Security, vol. 4, no. 1, PP. 99-106, Jan. 2007.

[14] S. Moon, “A binary redundant scalar point multipli-cation in secure elliptic curve cryptosystems,” International Journal of Network Security, vol. 3, no. 2,PP. 132-137, Sep. 2006.

[15] R. Rajaram Ramasamy, M. Amutha Prabakar, M. Indra Devi, and M. Suguna, "Knapsack based ECC Encryption and Decryption', Inter-national Journal of Network Security, vol. 9, no. 3,PP. 218-226, Nov. 2009.

[16] Z. J. Shi, and H. Yan, “Software implementation of elliptic curve cryptography,” International Journal of Network Security, vol. 7, no. 2, pp. 157-166, Sep.2008.

[17] Standard Specifications for Public Key Cryptography, IEEE Standard p1363, 2000.

[18] W. Stallings, Cryptography and Network Security,Prentice Hall, 4th Edition, 2006.

[19] H. Wang, B. Sheng, and Q. li, “Elliptic curve cryptography-based access control in sensor net-works,” International Journal of Security and Net-works, vol. 1, no. 3/4, pp. 127-137, 2006.

[20] L. Yongliang, W. Gao, H. Yao, and X. Yu, “Ellip-tic curve cryptography based wireless authentication protocol,” International Journal of Network Security, vol. 4, no. 1, PP. 99-106, Jan. 2007.