Implementation of Optimizing Point Multiplication in Elliptic Curve Cryptography Using Binary Method

[1]Bh.Padma [2]P.Prapoorna Roja [3]C.Srinivas [4]N.Venugopal

Abstract

Public key cryptography has become a mainstay for secure communications. They lay the foundation of secure transmission of documents, key management and digital signatures. Elliptic curve cryptography is an asymmetric key cryptographic tool that has gained a lot of attention in the industry. Elliptic Curve Cryptography provides greater security and more efficient performance than the first generation public key techniques now in use. The performance of elliptic curve cryptosystem heavily depends on an operation called point multiplication or scalar multiplication. The speed of scalar multiplication dominates the efficiency of the system. Fast multiplication is particularly more crucial for some environments such as central servers, where large numbers of key agreements or signature generations occur, and in handheld devices with low computational power. Because of such importance of speed of scalar multiplication, several methods have been developed. This paper describes implementations and test results of Elliptic Curve Cryptography (ECC) The paper deals with the problem of improving the performance of point multiplication using binary representation. This method reduces the point addition as well as the point doubling. This paper insists the application of Binary Method for point multiplication as it is faster than the widely used original method.

Keywords : Encryption/Decryption, Elliptic Curve cryptography, Elliptic curves, Discrete logarithmic problems.

Introduction

Elliptic Curve Cryptosystem was proposed by Miller [6] and Koblitz[7], which relies on the difficulty of elliptic curve discrete logarithm problem(ECDLP). The vast majority of the products and standards that use public key cryptography for encryption use RSA[8]. But the bit length for secure RSA use has increased over recent years and this has put a heavier processing load on applications using RSA. A competing system known as Elliptic curve Cryptography (ECC) has begun to challenge RSA [5]. The principal attraction of ECC compared to RSA is that it offers equal security for a smaller bit size, thereby reducing processing overhead. Another advantage that makes EC more attractive is the possibility of optimizing the arithmetic operations in the underlying field [2]. Now ECC is very popular for many information security applications [10].

Elliptic Curve Cryptography

Elliptic Curve Cryptography (ECC) is a public key technology that offers performance advantages at higher security levels cryptography. Every user taking part in public key cryptography will take a pair of keys, a public key and a private key. Only the particular user knows the private key whereas the public keys are distributed to all users taking part in the communication. Some public key algorithm may require a set of predefined constants to be known by all the devices taking part in the communication. In ECC we call these predefined constants as 'Domain parameters' [12].

The mathematical operations of ECC is defined over the elliptic curve $y^2 = x^3 + ax + b$, where $4a^3 + 27b^2$ “ 0. Each

value of the 'a' and 'b' gives a different elliptic curve. All points (x, y) which satisfies the above equation plus a point at infinity lies on the elliptic curve. The public key is a point in the curve and the private key is a random number. The public key is obtained by multiplying the private key with the generator point G in the curve [13].

The security of ECC depends on the difficulty of Elliptic Curve Discrete Logarithm Problem [14]. Let P and Q be two points on an elliptic curve such that kP = Q, where k is a scalar. Given P and Q, it is computationally infeasible to obtain k, if k is sufficiently large. k is the discrete logarithm of Q to the base P. Hence the main operation involved in ECC is point multiplication. i.e. multiplication of a scalar k with any point P on the curve to obtain another point Q on the curve.

The most crucial operation in elliptic curve cryptography is the scalar multiplication using a large integer *k*. In elliptic curve cryptosystems, scalar multiplication is involved in main operations such as key agreement and signing/ verifying. The speed of scalar multiplication dominates the efficiency of the system. The Binary Method not only reduces the total number of point addition, but also reduces the number of point doubling [3, 4, and 15]. The observations that are found while using both the methods for encrypting and decrypting a file have shown that the binary method is faster than the widely used original method.

Scalar Multiplication

Given an elliptic curve point P and an integer, scalar multiplication is finding another point. i.e. Finding kP = P+P+P+.....+ k times. Here the point P is a fixed point that generates a large, prime order subgroup. It is also known as point multiplication and dominates the execution times of elliptic curve cryptographic scheme. This is the original method we generally use while implementing ECC.

Binary Method

Elliptic curve cryptosystem will be the cryptosystem for the future. One way to improve the performance of such cryptosystem is to use an efficient method for point multiplication that is Binary Method[1]. For computing kP, the simplest method is finding binary representation of k[9].

i.e $k = \sum_{j=0}^{l-1} k_j \, 2^j$ where each $k_j \varepsilon \{ 0, 1 \}$.

then kp can be computed as

i.e $kP = \sum_{j=0} k_j \, 2^j * P$

$kP = 2(.....2(2k_{l-1} P + k_{l-2} P)+.....)+ k_0 P$

This method requires l -1 doublings and W_k - 1 addition, where W_k is the weight (the number of one's)[3].

As for example let k=15.

Binary representation of K is $(1111)_2$.

Thus Q=15P can be obtained as

Q = 15P = 2(2(2P+P) +P) +P.

similarly the others

Q = 22P = 2(2(2(2P) +P) +P). - $(10110)_2$

Q=45P= 2(2(2(2(2P)+P)+P)+P- $(101101)_2$

Q=63P=2(2(2(2(2P+P) +P) +P) +P) +P - $(111111)_2$

CPU Time for Encryption and Decryption and Key Exchange in ECC

The time taken for scalar multiplication varies according to the following factors:

- **1. In Encryption** - Value of K chosen By Sender.
- **2. In Decryption** - The Private Key value chosen by Receiver.
- 3. Both Encryption time and Decryption time linearly increase with the **size of the file.**

➢ **4. Diffie Hellmann Key Exchange** Private Key values of Sender and Receiver [11].

The comparison of CPU Times for encryption times and decryption times in both the scalar multiplication methods with elliptic curve parameters: p=2011, a=9, b=7, G= (2010, 1600) was recorded. The encryption and decryption times are specific to the processor. The above observations are recorded on a machine with 1GB RAM and 1.6 GHz processor speed on Win XP platform.

Case 1 : Comparison of CPU time in General Method and Binary Method for Scalar Multiplication in ECC (for fixed values of parameters: private key of receiver is 35 and k=1000) for different file sizes. The graphs below represents the variations in the encryption and decryption times for both the binary and the general method.

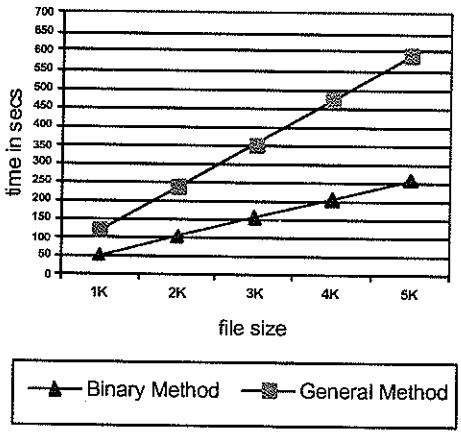

Figure1: Encryption

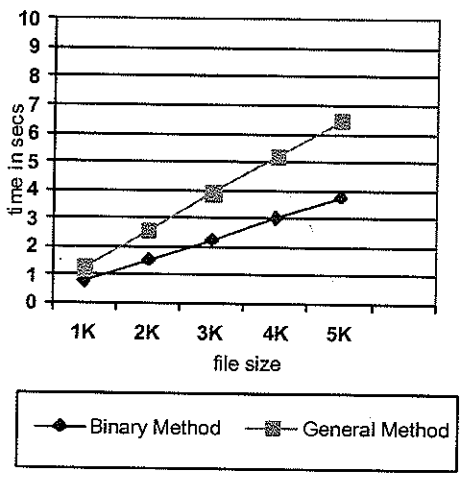

Figure 2 : Decryption

Case 2: Comparison of CPU time in General Method and Binary Method for Scalar Multiplication in ECC, for different k values in encryption process where the private key value of receiver is 35 and file size is 1k. The graph below represents the Encryption times in both the methods.

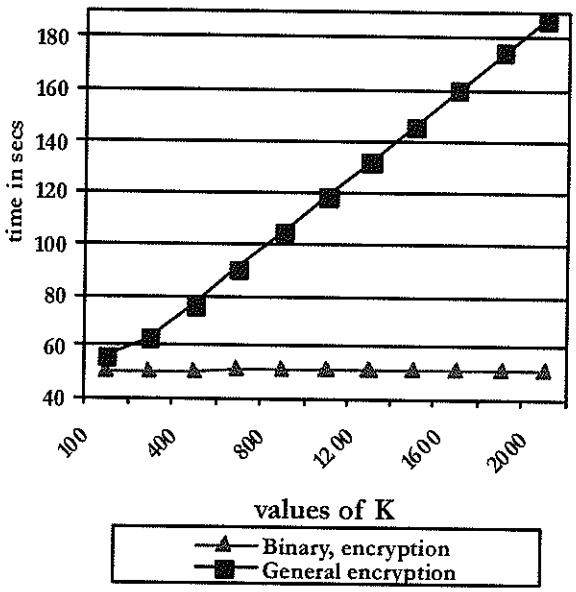

Figure 3 : Encryption

Case 3 : Comparison of Decryption Times when the private key values of receiver varies. The graph below represents the decryption times for both the binary and the general method.

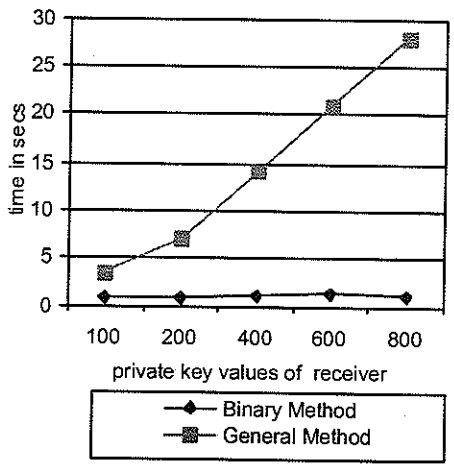

Figure 4 : Decryption

Case 4: Effect of Generator Point And Public Key Values in Encryption and Decryption Times in both the methods: k=1000, file size = 1k, private key of receiver is 35. The table below shows the execution time in both the methods for encryption and decryption.

Table 1 : Effect Of Generator Point

CPU Time In Seconds		Binary Method		General Method	
Generator Point	Public Key Of B	Encryption	Decryption	Encryption	Decryption
(2, 523)	(1864, 131)	70.78	0.8	102.37	1.93
(2, 1488)	(1867, 1880)	70.77	0.8	102.36	1.91
(5, 359)	(1203, 11)	70.75	0.8	101.96	1.92
(5, 1652)	(1203, 2000)	70.88	0.8	102.42	1.92

Case 5: Comparison of CPU Time in Key Exchange when both the methods are implemented. Private Key of sender is kept constant and Private key value of receiver is 35. Encryption and Decryption times for both the methods for key exchange is shown in the following graph.

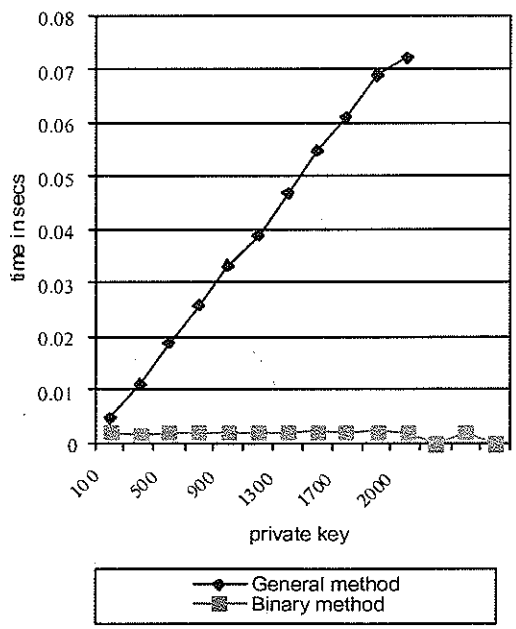

Figure 5 : Key Exchange

Conclusions

For efficient implementation of ECC, it is important for the point multiplication algorithm and the underlying field arithmetic to be efficient. Implementation of ECC using Binary Method on point multiplication has shown considerable improvement in efficiency compared to the original implementation. The following can be concluded when Binary method is used on point multiplication for ECC.

When the size of the input files grows:

- The Execution time taken for Encryption in Binary method reduces to half that of the General Method.
- Also Decryption time in Binary Method reduces to 40% that of the General Method.

When the value of k increases:

There is a negligible growth in the Execution time for Encryption in Binary method where as a large growth in the Execution time for Encryption is found in General method. There is no significant change in decryption times.

- Also as the k value grows the following holds good:

X*tB = tG x=1, 2 ...where

tB= encryption, time in binary Method

tG=encryption time in General Method

When the value of Private Key of Receiver nB increases:

- There is a negligible growth in the Decryption time in Binary Method where as a large and significant growth (multiples of ten) in the Execution time for Decryption is found in General Method.

x * tB = tG x=10,20,30 where

tB= decryption time in binary Method

tG= decryption time in General Method

When different Generator Point and Public Key are used:

- There is no impact is found on Encryption Time and Decryption Time for different values of Generator Point and Public Key values of the receiver.

When different Generator Point and Public Key are used:

- There is a very small growth in the CPU time for key exchange in binary method, compared to that of general method.

References

[1] Md. Rafiqul Islam, Md. Sajjadul Hasan, Ikhtear Sharif Muhammad Asaduzzaman, *"A New Point Multiplication Method for Elliptic Curve Cryptography Using Modified Base Representation"*, International Journal of the Computer, the Internet and Management Vol.16. No.2, PP. 9-16, May-August 2008.

[2] E.Al-Daoud, R.Mahmod, Md. Rushdan, A. Kilicman , *"A new addition formula for Elliptic curve over GF (2n)"*, IEEE Transactions on Computers, Vol. 51, No.8, PP. 972-975, Aug.2002.

[3] J. Lopez, R. Dahab , *"An overview of elliptic curve cryptography"*, Technical report, IC-00-10, May 22. Available at *http:// www.dcc.unicamp.br/ic-main/public-cation - e.html.* 2000

[4] J. Solinas (2000), *"Efficient arithmetic on Koblitz curves"*, Designs, odes and Cryptography, Vol. 19, PP.195-249.

[5] W. Stalling , *Cryptography andNetwork Security*, Prentice Hall, New Jersey, USA, Third Edition, Chapter 10, 2003.

[6] V.S. Miller, *"Use of elliptic curves in cryptography"*, Advances in Cryptology, Proceedings of CRYPTO'85, LNCS, Springer-Verlag, 218,PP. 417–426, 1986.

[7] N. Koblitz, *"Elliptic curve cryptosystem"*, Mathematics of Computation 48 , 203–209, 1987.

[8] R.L. Rivest, A. Shamir, L.M. Adleman, *"A Method for obtaining digital signatures and public key cryptosystem"*, Communications of the ACM 21,120–126, 1978.

[9] Standard Specifications for Public Key Cryptography, IEEE Standard 1363, 2000.

[10] A.J. Menezes, S. A Vanstone, *"Elliptic curve cryptosystem and their implementations"*, Journal of Cryptology 6 (4) , 209–224, 1993.

[11] W. Diffie, M.E. Hellman, *"New directions in cryptography"*, IEEE Transactions on Information Theory 22 (6) , 644–654, 1976.

[12] *"Elliptic Curve Cryptography"*, Standards for Efficient Cryptography Group, September, 2000.Working Draft. Available from: http:// www.secg.org

[13] A. K. Bhandari, D.S. Nagraj, B.Ramkrishna, T. N. Venkataramana, (editors). *"Elliptic Curves, Modular Forms and Cryptography"*, New Delhi, India: Hindustan Book Agency, 2003.

[14] Dummit, S.David and Foote, M.Richard, Abstract Algebra, New York, NY: John Wiley and Sons, Inc., 1999.

[15] M.Brown, D.Hankerson, J.L_opez, and A.Menezes, *"Software implementation of the NIST elliptic curves over prime _elds"*. In Progress in Cryptology {CT-RSA 2001 (2001), D. Naccache, Ed., Vol. 2020 of Lecture Notes in Computer Science, pp. 250- 265.

Author's Biography

Bh.Padma is an assistant professor working in G.V.P.College of P.G. and Degree Courses, with an experience of 3 years in teaching and 3 years in IT industry.She has an aptitude for research. Her topics of interest are Data Mining, Cryptography and Analysis of Algorithms.

P.Prapoorna Roja is a professor working in Jerusalem college of engineering, having 15 years of experience in teaching, and research. She has guided a number of P.G students for their thesis works, and at present has a research scholar associated with her. Her areas of interests include Computer Networks, Network security, Computer organization and architecture.

C.Srinivas is an associate professor in G.V.P.College of engineering for women. He has an experience of 11years both in industry and teaching. His international exposure has instigated in him the zeal and interest in computer networks and cryptography. Added to this he is CISCO's certified instructor for CCNA programs.

N.Venugopal is an associate professor in G.V.P.College of Engineering. His 10 years of experience includes his experience both in industry and teaching. He is CISCO's certified instructor for CCNA programs. His research interests include algorithm analysis, DBMS and computer networks.