Control Theory and Informatics
ISSN 2224-5774 (print) ISSN 2225-0492 (online)
Vol.3, No.2, 2013- National Conference on Emerging Trends in Electrical, Instrumentation & Communication Engineering

ECC Cipher Processor Based On Knapsack Algorithm

Jitendra Sharma and Prashant Shukla (Corresponding author)
Department of Electronics & Communication Engineering, G.L.A. University, Mathura, U.P., India-281406
E-mail: jitendrasharma_786@yahoo.com

Abstract
Elliptical Curve Cryptography (ECC) provides a secure means of exchanging keys among communicating hosts using the Diffie Hellmen Key Exchange algorithm. This paper presents the implementation of ECC by first transforming the message into an affine point on the elliptical curve (EC), and then applying the knapsack algorithm on ECC encrypted message over the finite field gF(p). In ECC we normally start with an affine point called Pm(x,y). This point lies on the elliptic curve. In this paper we have illustrated encryption/decryption involving the ASCII value of the characters constituting the message, and then subjecting it to the knapsack algorithm. Thus the modified plain text has been encrypted by application of the ECC method. The modification of the plain text in conjunction with P_m and application of Knapsack algorithm is the new innovation of this paper. The security of ECC relies on the difficulty of solving the Elliptic Curve Discrete Logarithm Problem (ECDLP), i.e. finding k, given P and Q = kP. The problem is computationally intractable for large values of k.
Keywords: Discrete logarithm, elliptic curve cryptography (ECC), knapsack algorithm, public key cryptography, RSA algorithm

1. Introduction

In 1976, Whitfield Diffie and Martin Hellman introduced the concept of Public Key Cryptography (PKC). Since then, many implementations of it have been proposed, and many of these cryptographic applications are based on their security on the intractability of hard mathematical problems, namely the Integer Factorization Problem (IFP) and the finite field Discrete Logarithm Problem (DLP). Over the years, sub-exponential time algorithms were developed to solve these problems. As a result, key sizes grew to more than 1000 bits, so as to attain a reasonable level of security. In constrained environments where computing power, storage and bandwidth are limited, carrying out thousand-bit operations becomes an impractical approach to providing adequate security. In order to protect or exchange confidential data, the cryptography plays an important role in the security of the information. Therefore, it is necessary to implement efficient cryptosystems, which can support applications economically feasible. In this context, public key cryptography based on elliptic curves is widely used because it presents higher security per key bit, and their main application is the private key exchange. Additionally, the Elliptic Curve Cryptosystems (ECC) can be used in applications where the computation resources are limited such as smart cards and cellular telephones. The ECC systems are included in the NIST (National Institute of Standards and Technologies) and ANSI (American National Standard Institute) standards, and the principle advantage over other systems of public key like RSA is the size of the parameters, which are very small, however the ECC systems provide the same level of computational security. 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 a 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. McIvor et al. [7] introduces a novel hardware architecture for ECC over GF(p). Chen et al. [2] presents a high performance EC cryptographic process for general curves over GF(p). The standard specifications for public key cryptography are defined in [10]. The idea for extending knapsack algorithm to encryption/decryption was derived by Diffie [5]. The ECC concept is very well documented and illustrated by Williams Stallings [11]. In [12], Access control in sensor networks is used to authorize and grant users the right to access the network and data collected by sensors. This paper describes a public key implementation of access control in a sensor network. The paper, Moon [8] proposed a more efficient and novel approach of a scalar point multiplication method than existing double and add by applying redundant recoding, which originates from radix-4 Booths algorithm. Shi et al. [9], investigates whether some architectural parameters such as word size may affect the choice of algorithms when implementing ECC with software.

2. Method Used:

ECC Encryption:

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

$y^2 = x^3 + ax + b$ ---------- (1)

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

$4a^3 + 27b^2 \neq 0 \ mod(p)$ ---------- (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 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.

Algorithm1: General Key Flow (U_{Alice}, U_{Bob}, P_m)

1: Begin
2: Initiate the connection with U_{Alice} and U_{Bob}
3: if (U_{Alice} and U_{Bob} is legitimate user) then
4: U_{Bob} = {P_{Bob}, n_{Bob}}//Key pair for U_{Bob}
5:U_{Alice} = {P_{Alice}, n_{Alice}}//Key pair forU_{Alice}
6:U_{Bob} = Send (P_{Bob}, U_{Alice}) ;//Send the Public key of U_{Bob} to U_{Alice}
7: U_{Alice} =Send (P_{Alice}, U_{Bob});//Send the Public key of U_{Alice} to U_{Bob}
8: U_{Bob} = Send (U_{Alice}, Public information);
9: U_{Alice} = Send (U_{Bob}, Public information);
10: endif

Algorithm 2: Encryption

1: //Encryption– Alice encrypts the message and sends it to Bob
2: if (Encryption=TRUE) then
3: Calculate the P_m ': P_m' = G.P_m, where P_m is the ASCII value of Plain Text and G is the base point of EC
4: Calculate the kP_{Bob} value
5: Calculate the P_m' + kP_{Bob} = (x2, y2)
6: Calculate the kG = (x1, y1)

Hence the entire encrypted version for purposes of storing or transmission consists of two sets of coordinates as follows:

$C_m = (kG, P_m' + kP_B)$
$kG \rightarrow x_1, y_1$
$P_m + kP_B \rightarrow x_2, y_2$

Thus the modified plain text has been encrypted by application of the ECC method. The modification of the plain text in conjunction with P_m and application of Knapsack algorithm is the new innovation of this paper.

We start with P(x_P, y_P). To determine 2P, P is doubled. This should be an affine point on EC. Using the following equation, which is a tangent to the curve at point P.

$S = [(3x_P^2 + a)/2y_P \ ] mod p$

For the encryption process, a number of additions are required to take place. The formulae used to calculate the value of the coordinates x_R and y_R are given as

$x_R = (S^2 - x_P - x_Q) mod p$
$y_R = [S(x_P - x_R) - y_P] mod p$

Now to determine 3P, we use addition of points P and 2P, treating 2P=Q here P has coordinates (x_P, y_P) and Q = 2P has coordinates (x_Q,y_Q).

In these two formulas of point addition and point doubling a variable S is calculated which is used for computing the values of x_R and y_R for both point addition and point doubling calculation.

Therefore we apply doubling and addition depending on a sequence of operations determined for k. Every point x_R, y_R evaluated by doubling or addition is an affine point.

For example a series of additions and doubling of P

P	2P	3P	6P	12P	24P	48P	96P
-	Doubling	Addition	Doubling	Doubling	Doubling	Doubling	Doubling

Once the ECC encrypted version of the text message is obtained, the knapsack encryption is forced over it. This is done to further secure the message from the hackers.

The most famous of the fallen algorithms is Ralph Merkle's Knapsack problem.

Algorithm 3: Knapsack Encryption

1: S[x1] = Knapsack value(x1);
2: S[y1] = Knapsack value(y1);

3: S[x2] = Knapsack value(x2);
4: S[y2]=Knapsack value(y2);
5: Send(U_{Bob} , C_m =((S[x1], S[y1]),(S[x2], S[y2])));
As per the knapsack algorithm we calculate a cumulative sum S[x_1],

$$S[X1] = \sum_{i=0}^{m} aixi$$

In the final encrypted version the co-ordinate x is replaced by its equivalent S[x_1]. Similarly other coordinates like y_1, x_2, y_2 are transformed by the knapsack algorithm, so that the encrypted message is now represented as
C_m = ((S[x_1],S[y_1]),(S[x_2],S[y_2])).
Depending on the number of characters in the message, there will be as many such pairs of integers. This is either stored in archival device like CD/DVD or transmitted to a beneficiary through the Net.

Algorithm 4: Knapsack Decryption

1: //Decryption– Bob decrypts the message received from Alice
2: if (Decryption=TRUE) then
3: x1= Inverse Knapsack value (S[x1]);
4: y1 = Inverse Knapsack value (S[y1]);
5: x2 = Inverse Knapsack value (S[x2]);
6: y2 = Inverse Knapsack value
7: endif

The recipient B has all relevant information for reversing the knapsack procedure and to recover bit pattern of the coordinates. For example B knows the a_i series, his own secret key n, the base point G, a, b, p values of the EC. B receives the encrypted message.
The x_1 value is recovered in an iterative fashion as follows:
S[x_1]- n^m .
If this value is positive i.e., S[x_1] - n^m > 0, then a binary bit1 is assigned at the (m) position. The current value is S[x_1] = S[x_1] - n^m. If however, the value is negative, then a bit0 is assigned and the S[x_1] remains unchanged.
Now subtract n^{m-1} from the current S[x_1]. Depending upon whether it is +ve or –ve, assign 1 or 0 at the relevant bit position. Continue this subtraction until the series is exhausted. This will recover the binary bit pattern of x_1.
These procedures are repeated for y_1, x_2, and y_2.

Algorithm 5: ECC decryption

1: kG = (x1, y1);
2: P_m' + kP_{Bob} = (x2, y2);
3: Calculate n_{Bob} kG = n_{Bob} (x1, y1);
4: Calculate P_m' = P_m' + kP_{Bob} –n_{Bob} kG;
5: Calculate the P_m value from P_m' using discrete logarithm.

The decryption involves the following procedures:
kG is represented by x_1, y_1, and P'_m+ kP_B is represented by x, y.
In order to pull out P'_m from P'_m +kP_B , B applies his secret key n_B and multiplies kG so that,
n_B kG = kP_B .
Subtracting this from P'_m + kP_B , to get P'_m as follows:
P'_m = P'_m + kP_B - n_BkG
This subtraction is another ECC procedure involving doubling and addition. But, the only difference is that the negative term will have its y coordinate preceded by a minus sign. With this subtle change in mind, the expression of determining the slope, new values of x_R, y_R are the same. Wherever y figures, it is substituted as -y. This will yield P'_m. Using the discrete logarithm concept the ASCII value of 'S' can be retrieved as follows, P'_m= S P_m.

3. PERFORMANCE COMPARISON USING VARIOUS TARGET DEVICES

The synthesis for various bit orders was done using different target devices and results were obtained and the corresponding graphs were plotted. It is found that the target device VIRTEX-4 is more efficient compared to SPARTAN-3E and VIRTEX-2 as it consumes less time to encrypt and decrypt a given data. Given below are the graphs, Fig.1 and Fig.2, comparing the efficiency performance of the various target devices- SPARTAN-3E, VIRTEX-2 and VIRTEX-4.

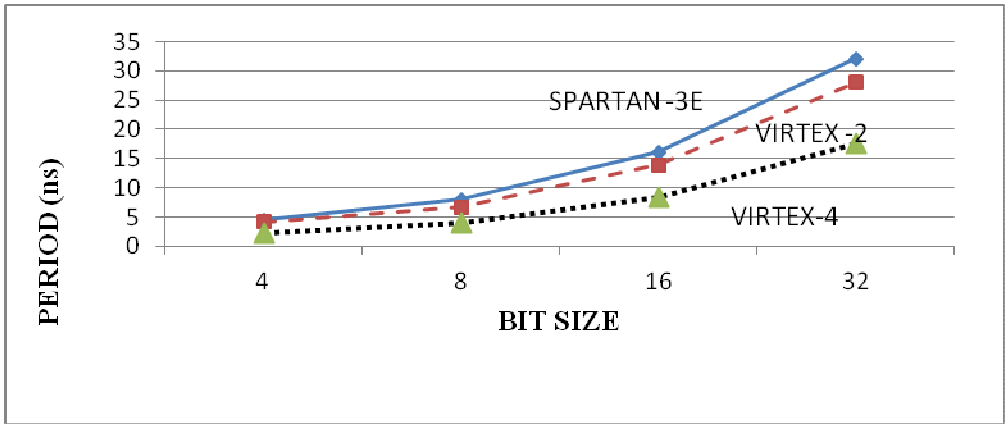

Fig.1 bit order Vs time period (ns) for encryption

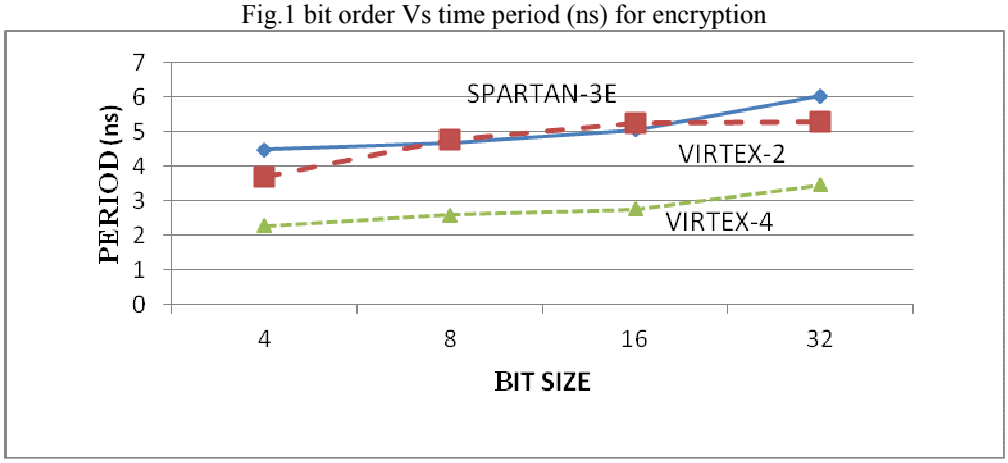

Fig.2 bit order Vs time period (ns) for decryption

4. APPLICATIONS OF ECC

Wireless communications, where there is a speed and memory tradeoff, ECC allows for smaller keys and faster processing times. This significantly improves the protocol execution speeds, and reduces power consumption. This is significant in portable communication devices such as cell-phones and pagers. An ECC implemented in this environment would be beneficial in ATM Machine as it has the reduced key sizes and computations should significantly reduce transaction times, making it more convenient for the user. Phone cards, and more generally Smart Cards, in most instances, important data is stored on the card, such as the amount of money available on it.

5. CONCLUSION

The realization of ECC based cipher processor has been attempted. Scalar multiplication is the main operation used to convert the plain text to cipher text, performed using point addition and point doubling. Once the ECC encryption is done, further complexity is added to the cipher text by using Knapsack Algorithm.

The operations of point addition and point doubling include full adder, adder, carry save adder, squarer and divider modules. The language used to code these modules is VHDL and that of the test benches is Verilog. The modules are integrated to obtain ECC Encryption, Knapsack encryption, knapsack decryption and ECC decryption. The software used to simulate the modules is Modelsim 6.1c. For synthesis, XILINX software is used. From the synthesis result, it is obtained that the optimum target device is VERTIX -4. The performance of encryption/ decryption on the target device with respect to time is shown in Fig.11 and Fig.12. The efficiency with respect to area is shown in Fig.13 and Fig.14.

Thus, after a four stage process of ECC encryption, knapsack encryption, knapsack decryption, ECC decryption, the ciphered text from the sender is deciphered at the receiver.

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, pp273-279.Oct.2001.

2. G.Chen,G.Bai,and H.Chen," A High-performance elliptic curve cryptographic procressor 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.Chenug,"Customzablei elliptic curve cryptosystms," 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 cryptography," Proceedings of IEEE, vol.76, no.5, pp.560-577, May 1998.
6. K.Lauter,"The advantage of elliptic curve cryptography for wireless security," IEEE Wireless Communications, pp.62-67, Feb 2006.
7. C.J.Mclvor, M.McLoone, and J.V.McCanny, "Hardware elliptic curve cryptographic processor over GF(p)," IEEE Transactions on Circuits Syst. I: Reg. Papers, vol.53, no.9, pp.1946-1957, Sep 2006.
8. S.Moon,"A Binary Reduandant Scalar point multiplication in secure elliptic curve cryptosystems," International Journel of Network Security, vol.3, no.2, pp.132-137, Sep 2006.
9. Z.J.Shi, and H.Yan, "Software implementation of elliptic curve cryptography," International Journel of Network Security, vol.7, no.2, pp.157-166, Sep 2008.
10. Standard specifications for Public Key Cryptography, IEEE Standard p1363, 2000.
11. W.Stallings, Cryptography and Network Security, Prentice Hall, 4th Edition,2006.
12. H.Wang, B.Sheng, and Q.Li,"Elliptic curve cryptography-based access control in sensor networks," International Journel of security and networks, vol.1, no.3/4, pp.127-137, 2006.