International Journal of Scientific & Engineering Research, Volume 6, Issue 8, August-2015
ISSN 2229-5518

An Efficient Approach to Elliptic Curve Cryptography

Rabindra Bista and Gunendra Bikram Bidari

Abstract— This paper has analyzed a method for improving scalar-multiplication in cryptographic algorithms based on Elliptic Curves owing to the fact that has established the superiority of the Elliptic Curve next generation cryptographic algorithms over the present day cryptographic algorithms. For the implementation of the method, the algorithms have been implemented and discussed with the relevant values such the results obtained are recorded, analyzed and compared with contemporary algorithms. More specifically, this paper carried out research on various aspects of elliptic curves and alternatives to reduce the cost incurred for scalar multiplication in Elliptic Curve Cryptography thereby making it possible to construct easy reluctant additive sequence which backtracks whenever the anomalies are encountered thereby making the operation of scalar multiplication operation efficient both in terms of the required operations to be performed and the number of bits to be recorded. Although research in questions related to elliptic curves was pursued earlier for aesthetic reasons, these questions have placed themselves as prominent in several applied areas such as pseudorandom number generation, coding theory and cryptography.

Index Terms— Elliptic Curve Cryptography (ECC), Elliptic Curve Discrete Logarithm Problem, Elliptic Elgamal, Elliptic Curve Digital Signature Algorithm, Scalar Multiplication, Cryptanalysis.

1 Introduction

THE public key encryption scheme based on elliptic curve theory for the creation of smaller, faster and more efficient cryptographic keys through the implementation of curve equation therby offering an unique potential security mechanism is elliptic curve cryptography which provides efficient usefulness in the resource constraint devices that deliver extremely low resource consumption and cryptographic countermeasured offered in software.

1.1 Background and Motivation

An elliptic curve E, which is defined over a field K can be defined explicitly by the Weierstrass equation of the form

$$E: y^2 + a_1xy + a_3y += x^3 + a_2x^2 + a_4x + a_6 \quad (1)$$

where $a_1, a_2, a_3, a_4, a_6 \epsilon K$. If characteristic of K is not 2 or 3 which is $\text{char}(k) \neq 2 \; or \; 3$, then following is the form of the elliptic curve E,

$$y^2 = x^3 + ax + b \quad (2)$$

One of the alternatives to this form is,

$$y^2 + ax = x^3 + bx^2 + c \quad (3)$$

where $a, b \epsilon K$. The discriminant of the curve is $\Delta = -16\,(4\,a^3 + 27b^2)$, which specifies: if the polynomial $x^3 + ax + b$ has a double root in the curves, those curves are not used being vulnerable to the attacks as the double roots may lead to the destination point from multiple source points [1].

1.2 Problem Domain

The scalar point multiplication is the main workload of ECC algorithms. Researchers have shown that scalar point multiplication accounts up to 80% of the total operations performed on Elliptic Curve Cryptosystems [2]. The inefficient implementation of scalar point multiplication hampers the performance thereby slowing down the resource constraint devices like smart phones, ATM machines and others.

1.3 Objectives

The primary objective of this research to study the different algorithms implemented being based on Elliptic Curves. The research modifies one of the existing algorithms such that the operation of scalar multiplication becomes efficient enabling faster ECC computation. The research carries out the comparative analysis with other contemporary approaches in terms of types of operations and CPU cycles. Finally, the research cryptanalyzes the Elliptic Curve Cryptosystems.

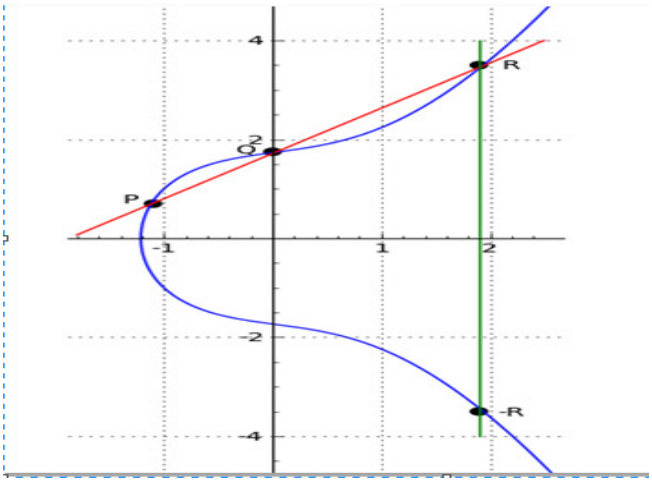

Fig. 1. An Elliptic Curve

1.4 Research Questions

- Exploring the possibilities to figure out some sort of alternatives to improve Multi-Scalar Multiplication.
- It is possible to eliminate the doubling operation?

- *Rabindra Bista, PhD, is an Assistant Professor in Dept.of CSE, Kathmandu University, Nepal, PH-9849476579, E-mail: rbista@ku.edu.np*
- *Gunendra Bikram Bidari is a Graduate Student (ME) in Dept. of CSE, Kathmandu University, Nepal, E-mail: gunendrabidari@gmail.com*

- What is the necessary tradeoff to be made for replacing doubling operation with only addition and subtraction?

2 ECC ALGORITHMS

We discuss three major implementations of Elliptic Curve Cryptographic algorithms namely: Elliptic Curve Diffie-Hellman Key Exchange, Elliptic Curve Elgamal and Elliptic Curve Signature and Signature verification Algorithms.

2.1 Elliptic Curve Diffie-Hellman Key Exchange

Classical Diffie-Hellman Key Exchange Algorithm which forms the foundation of Elliptic Curve Diffie-Hellman Key Exchange Algorithm is used when Alice and Bob communicating in insecure channel agree to share the secrete key. With the establishment of Key Distribution Centers (KDC), the keys were delivered to the concerned communicating parties [3]. Although the fundamental principal remains same as that of classical Diffie-Hellman Key Exchange Algorithm, Elliptic Curve Diffie-Hellman Key Exchange gets the strength from the property of Elliptic Curves which is the operations to be performed are very easy to perform and extremely difficult to revert. This property of Elliptic Curve is extremely useful in cryptographic use [4].

The algorithm for Elliptic Curve Diffie-Hellman Key Exchange is given as:

1. Alice and Bob publicly agree on an Elliptic Curve E over a large finite field $\mathbb{F}$ and a point P on that curve.
2. Alice and Bob each chose privately large random integers, denoted by a and b.
3. Using elliptic curve point-addition, Alice computes aP on E and sends it to Bob. Bob computes bP on E and sends it to Alice.
4. Both Alice and B can now compute the point abP: Alice by multiplying the received value of bP by her secret number a & b and vice versa.
5. Alice and Bob agree that the x-coordinate of this point will be their shared secret value.

Algorithm 1: Elliptic Curve Diffie-Hellman Key Exchange

2.2 Elliptic Elgamal

Elgamal is one of the asymmetric algorithms consisting of three components, key generator, encryption algorithm and decryption algorithm. With the security equivalent to that of RSA, the security of Elgamal is also based on the discrete logarithm problem which demands a discrete power to be executed to encrypt and decrypt the message [5].

A point $G \in E$ of large order and E are public information. The private key is an integer K_2 which is less than the order of G and the public key is $K_2 = GK_1$.

Being based on these basic Elgamal encryption and decryption schemes ECC Elgmal algorithm has been devised. The Elliptic ElGamal cryptosystem is formed by changing the multiplicative structure of F $*$ p by the group law in an elliptic curve E over Fp.

Message expansion factor and the points on the curve E consisted in plaintext are two practical issues in implementation of Elliptic ElGamal for the lack of point generating methods which are convenient and deterministic inherently in nature by themselves.

Setup: Choose an Elliptic Curve E over $\mathbb{F}_q$along with the point P on the curve. A Cyclic subgroup G is generated with order n. The private key for each user is an integer $l \in \{0,1,2 \dots n-1\}$ and public key lP is formed. Let, $m \in G$ is a message and Alice wants to send m to Bob.

Encryption: Alice generates a random integer $k \in \{0,1,2 \dots n\}$ and computes kP.

Alice looks up B's public key lP and computes $m + k(lP)$.

Alice sends to B the pair $(kP, m + klP)$.

Decryption: Bob Computes $(m + klP) - l(kP) = m$ and recovers the message.

Algorithm 2: Elgmal Encryption & Decryption for ECC

2.3 ECDSA

Elliptic Curve Digital Signature Algorithm (ECDSA) allows us to verify the authenticity of its security without being compromised by creating the digital signatures that like the real world signatures.

Signer A has domain parameters $D = (q, FR, a, b, G, n, h)$, private key d, and public key $Q = dG$. B has authentic copies of D and Q.

To sign a message m, A does the following:

1. Select a random integer k from $[1, n-1]$.
2. Compute $kG = (x1, y1)$ and $r = x1 \ mod \ n$.
3. Compute $e = SHA - 1(m)$.
4 Compute $s = k - 1\{e + dr\} \ mod \ n$.
5. A's signature for the message m is (r, s).

The computationally expensive operation is the scalar multiplication Kg in step 2, for a point G which is known a priori.

Algorithm 3: Elliptic Curve Digital Signature Signing

The presumption to use the real signatures is that we can figure out the signatory but we cannot modify the signature without the knowledge of the signatory [6]. Using ECDSA when we want to sign a file, we use a private key which is a random number generated applied on random point on curve known as point of origin.

The signature is generated by applying the private key with the hash of the file where hash is a mathematical equation that is applied on every bite of data which generates a

number unique to the number such that the hash gets changed every time the data is changed ensuring that the signature is not valid anymore.

The point of origin is a point on the Elliptic Curve which generates a second point on the curve as public key when multiplied with a randomly generated number. Elliptic Curve Cryptographic Digital Signature Algorithm (ECDSA) uses private key and the Cryptographic Hash Function (like SHA1) to sign the file [7].

To verify A's signature (r, s) on m, B does:

1. Verify that r and s are integers in $[1, n-1]$
2. Compute $e = SHA-1(m)$.
3. Compute $w = s - 1 \; mod \; n$.
4. Compute $u1 = ew \; mod n \; and \; u2 = rw \; mod \; n$.
5. Compute $u1G + u2Q = (x1, y1)$.
6. Compute v = x1 mod n.
7. Accept the signature if and only if $v = r$.

The computationally expensive operation is the scalar multiplications $u1G$ and $u2Q$ in step 5 where only G is known a priori

Algorithm 4: Elliptic Curve Digital Signature Verification

Private Key is used to generate the digital signatures whereas public key is used to validate the signature thus generated. The cryptographic security of ECC depends upon Discrete Logarithm Problem. It is necessary for a cryptanalyst to find the scalar which is the secret random number k from kP and P.

Till known computational complexity needed to code 150 bits key size Elliptic Curve Cryptosystem is given by Pollard's ρ which is equal to 3.8 $\times 10^{10}$ MIPS-yr. (number of years required when executed with millions of instructions per second)[8-9].

3 DESIGN AND IMPLEMENTATION

Modular reduction and scalar multiplication operations are the most heavily used operations in Elliptic Curve Cryptography. Scalar multiplication is the most resource demanding operation in Elliptic Curve Cryptography and consumes up to 85% of execution time. The operation of multiplying point P in scalar operation with itself is achieved through point addition and point doubling represented as:

$$kP = (P + P + P + \cdots .. + P) \; ------ \; k \text{ times}$$

As a consequence the choice of algorithm determines the operational efficiency of kP and efficiency of the whole cryptographic procedure itself. Mathematically it is represented as:

$$\sum_{i=1}^{n} k_i P_i \qquad (4)$$

where k_i denotes scalar and P_i are the point on Elliptic Curve. It is mandatory for the operations to be efficient in the case of multi-scalar multiplications as they are the most time consuming operations for Elliptic Curve Cryptosystems especially in the devices with constrain memory and computational power. Improvement in efficiency of multi-scalar multiplication is highlighted by the fact that overall efficiency of the system is dependent on the multiplication itself [10].

Slight modification to the SGRAC algorithm [11] provides us the capability to find the variance with the main theme and help us in our step of retreat by recording the differences while executing the algorithm. By modifying the algorithm we have not only simplified the scalar generating procedures but also reduced the number of bits to be stored.

Three flags 0, 1 and 2 are set in list *TrackItem* for indicating presence, absence of abnormalities and readjustment of the values. Starting from the given prime number the values are reduced until a small threshold is reached.

3.1 Devise of Algorithm

The existence of non abnormality is indicated with 0 in list *TrackItem*. Whenever the difference between two successive terms is larger than the predefined Fibonacci ratio, the value is multiplied with the ratio itself which is indicated by the addition of value 1 into the list *TrackItem* and abnormality is corrected by subtracting the recent list item from its penultimate predecessor and the corresponding adjustment is indicted by adding value 2 into the list *TrackItem* and the differences are stored in the list *Store*.

1.. Read the first value of scalar as $ListItem_0$

2.. Initialize the second and third values of series as

$ListItem_1 = ListItem_0 * \frac{(\sqrt{5}-1)}{2}$ & $ListItem_2 = ListItem_0 - ListItem_1$

3.. do until: $ListItem_i \leq \min\_val$

if $min\_val < abs(\, ListItem_i - ListItem_{i-1} * \frac{(\sqrt{5}-1)}{2})$

$ListItem.append(ListItem_{1i-2} * \frac{(\sqrt{5}-1)}{2})$

$TrackItem.append\ (1)$

$ListItem.append(ListItem_{i-2} - ListItem_i)$

$TrackItem.append\ (2)$

$\text{Store.append}\ (\text{ListItem}_i - \text{ListItem}_{i+1})$

else

$ListItem.append(ListItem_{i-2} - ListeItem_{i-1})$

$TrackItem.append\ (0)$

4. Return two lists $ListItem$ and $TrackItem$

Algorithm 5: Variation Tracking Algorithm

The difference between product of previous dataItem & Fibonacci ratio and current dataItem if exceeds the threshold value, the situation is termed as variation one and the values are readjusted. For the flag value 0, the next term is generated adding two immediate previous values. For flag value 1, Store valued is added with current *ListItem* and for the flag value 2,

the previous values are retreated one step and added up.

1. $do\ until\ cont\ <= i$
2. $if\ (TrckItem[cnt] == 0)$
   $RevItm.append(RevItm[cnt-1]$
   $+RevItm[cnt-2]$
3. $if\ (TrckItem[cnt] == 1)$
   $RevItm.append(RevItm[count-1]$
   $+Store[strcnt])$
   $strcount = strecnt + 1$
4. $if\ (TrckItm[cnt] == 2)$
   $RevItm.append(RevItm[cnt-2]$
   $+RevItm[cnt-3]$

Return $RevItm$

Algorithm 6: Scalar Generation

3.2 Results

The number of bit to be recorded for the increment in bit size of the scalar rises linearly thereby supporting the idea of maximum efficient utilization of storage.

Table 1: Timing Performance and bits to be stored

Bits	Time(ms)			Record (Bits)
	Worst	Average	Best	
64	6.28	5.32	3.72	48
96	9.5	8.47	6.96	48
128	8.67	8.51	8.48	56
160	12.25	9.93	6.67	56
192	20.03	12.29	8.94	64
224	14.23	12.3	11.03	72
256	18.05	14.08	12.48	72
288	19.12	16.04	14.72	80
320	26.01	25.31	18.89	88
352	29.54	25.72	19.11	96
384	32.37	26.85	23.72	96
416	35.96	29.74	25.9	104
448	46.71	36.65	28.06	104
480	51.63	41.09	32.57	120
512	59.04	48.37	36.88	128

As shown in the Table 1, starting from 64 bits upto 512 bits, the time for finding the value of scalar k are recorded along with the bits required for the storage.

Once, the first two precomputed values are figured out, the rest of the values obtained from Variation Tracking Algorithm can be discarded and only the values of the list *TrackItem* are to be preserved. The desired prime number is obtained with repeated addition and adjustment which completely eliminates the doubling operation.

Table 2: Number of Addition Operations along with series length

Bits	Operation Stages			Sequence Length
	worst	Average	Best	Average
64	71	69	68	37
96	115	112	109	52
128	161	160	156	81
160	254	247	239	114
192	351	341	330	157
224	438	435	429	205
256	532	530	517	248
288	633	628	605	301
320	726	706	697	343
352	806	793	781	392
384	895	884	869	409
416	974	961	947	436
448	1090	1067	1045	498
480	1193	1172	1156	531
512	1308	1279	1267	589

Moreover, the largest variation in terms of percentage is within three for the worst case when compared with average ensuring the efficient utilization of both time and memory resources.

Table 3 shows the comparison of the computation in terms of recorded time and number of operations involved with result of other algorithms studied earlier in the Section 2.

4 Analysis

The comparative analysis is obtained using the system environment as: Processor: Intel Core i5 CPU M 460 @ 2.53 GHz

with 2 GB RAM, System: 32-bit Operating System, x64-based processor and Tool: Sage-5.11 installed with Oracle VM VirtualBox Manager using Operating System Fedora.

4.1 Comparative Analysis

The methods were selected as the cost-consuming operation of scalar multiplication and inverse were substituted with addition, subtraction and double operation in all the cases.

The results obtained are compared against Extended Euclidean Algorithm which is based on Modular Reduction and Squaring, Generalized Continued Fraction based on Window Method, Shamir's Trick which performs addition and multiplication in blocks and Interleave method which works in bit level manipulation of binary numbers. The parameters can be referenced with values from Table 4.

Table 3: Comparison of Results

Methods	Operation	No of Operations
EEA	Modular Reduction and Squaring	80Add +260DBL
GCF	Window Method	232ADD
Shamir' Trick	Multiple Blocks in Operation	121ECADD +160ECDBL
Interleave Method	Bits manipulation	160 ECADD + 160 ECDBL
Our Scheme	Addition-Retreat-Addition	247ADD +3SUB

Table 4: Comparison of different operations of ECC [12]

Operations	CPU Cycles	Time Taken
Addition in GF(p)	315	1.97µs
Subtraction in in GF(p)	357	2.23 µs
Montgomery Multiplication	2860	17.88 µs
Point Addition	33049	207 µs
Point Doubling	40737	254 µs
Scalar Multiplication	10,148,863	63.4 ms

The results are compared with Extended Euclidian Algorithm which employs Modular Reduction and Squaring Method, Generalized Continued Fraction which employs Window Method, Shamir's Trick which finds results of operation of simultaneous Addition and Multiplication in blocks and Interleave Method which adjusts the Multiplication. Table 3 shows the results obtained are efficient in terms of time and CPU cycle when referenced with results from Table 4. It is evident that the costly operations of repeated multiplication and doubling have been completely replaced.

Table 3 shows the comparison of the computation in terms of recorded time and number of operations involved with result of other algorithms studied in literature review. Performance of the scheme is compared with other methods under consideration and is depicted in Fig. 3 where the performance of the scheme consists of 247 Add operations and 3 Subtract operations.

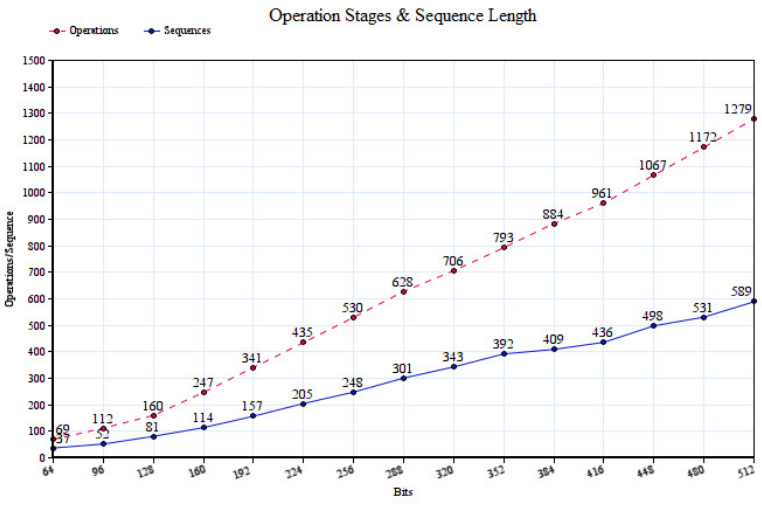

Fig. 2. Number of stages in operations and Sequence Length

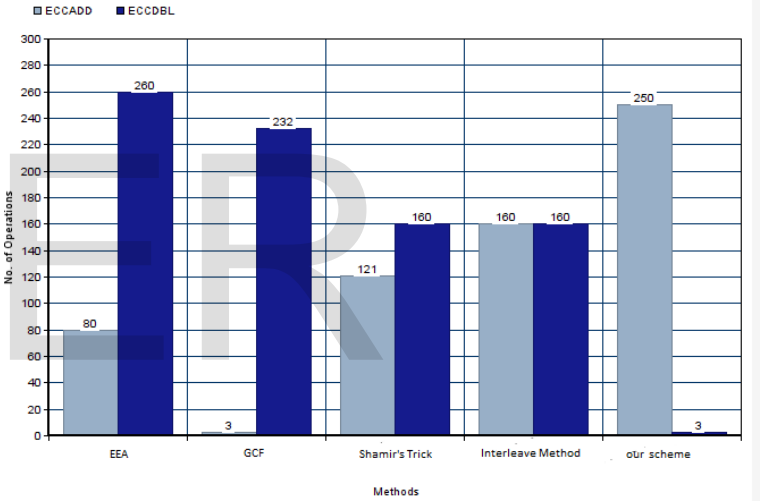

Fig. 3. Comparison of Performance

It is evident that Subtraction operation is mere sign inversion of Addition Operation. Fig. 3 clearly shows that the cumbersome operation of ECCDBL has been completely eliminated in new scheme.

Table 4 provides the costs in terms of CPU cycles and time taken for the completion associated with different operations used in ECC.

4.2 Cryptanalysis

As the cryptosystems based on Elliptic Curve Cryptography are robust, attempts have been made to figure out the secret key from the leaked information like the analysis different amount of electromagnetic emanations, error message, noise and even recording of CPU cycles. This analysis is called information leakage analysis and falls under the category of side channel analysis [13].

According to Roberts & Zobehi [14-15] there are both active and passive ways to attack a cryptographic system. In passive attack, the attacks records and studies the information

like timing leakages, power consumption leakages and error messages as the attack have no direct access to the devices. Hamming weight leakage and transition count leakage revel the information easily as they are associated with binaries [16].

Clever outsiders, funded organizations and knowledgeable insiders are some of the active attackers who might have sophisticated tools to carry out the analysis. The best way to protect from these types of attacks is inserting several dummy activities such that when the leaked information is analyzed no useful information is revealed [17].

Moreover, in Elliptic Curve Cryptography overwhelmingly depends upon the private key r which is a random number and some of the adversaries even figure out the pair (r, R) without even solving the DLP. As shown in [18-19], many applications like low power devices and high volume serves precompute the pair (r, R) for late use to enhance their performance.

The pairs are generally stored on disks making them extremely vulnerable. Storing in hardware protected media is one of the best ways to protect the forging. The pairs are not only vulnerable for the current use of the private key but also for the long run as the future keys are derived from the current pair as most predominant ways to generate large future key pairs are dependent on the current pair.

4 Conclusion

4.1 Findings

For the testing purpose, 100 random numbers for each of the corresponding bits were generated and tested. Simple analysis shows that the number of operations required during the execution have a linear correlation with number of bits employed. The average maximum variation between the worst case and average case is less than 4%.

Table 4 show the comparison of the computation in terms of recorded time and number of operations involved with result of other algorithms studied in literature review. The methods were selected as the cost-consuming operation of scalar multiplication and inverse were substituted with addition, subtraction and double operation in all the cases. The findings can be summarized as:

- It is possible to improve the efficiency of scalar multiplication in Elliptic Curve Cryptography
- Fibonacci pattern is the key for the implementation of Addition-Retreat-Addition scheme.
- The values obtained show that the number of bits to be stored and the operation time increases linearly for the increased bits to be manipulated.
- It is possible to eliminate the costly operation of doubling in ECC.

4.2 Contributions

The contributions of this research paper are as follows:

- Analysis of Algorithmic approach to Elliptic Curve Cryptography
- Modification of an existing algorithm such that it becomes suitable for efficient implementation of scalar multiplication in Elliptic Curve Cryptosystems.
- Comparative analysis and results.

4.3 Limitations

There are few limitations associated with this research work which are:

- The research work has been primarily being focused only into the field of Elliptic Curve Cryptographic schemes.
- Only the software implementation has been carried out in this research work and no hardware considerations were made.
- Low computational processing devices like 8-bit, 16-bit processors were not considered.

4.4 Recommendations for Future Work

Following could be the recommendations for researchers to explore in future:

- The researchers are recommended to extend the scheme discussed in pairing based cryptography and examine if comparable results can be obtained.
- Implementation of the discussed scheme in the hardware devices.

This research explored and examined various algorithms associated with Elliptic Curve Cryptography. It is figured out that the scalar multiplication operation is the most cost incurring operation in terms of resources in Elliptic Curve Cryptography. A scheme is developed, discussed and implemented to make the scalar multiplication in finite field efficient and comparable results have been presented.

Acknowledgment

The authors are indebted to Department of Computer Science and Engineering, Kathmandu University for providing the technical support to carry out this research.

References

[1] Koblitz, N., "Elliptic curve cryptosystems". Mathe- matics of Computation, 48(177):203-209, Jan. 1987.
[2] Lenstra, A. K & Verheul, E. R. "Selecting cryptographic key sizes". Journal ofCryptology, 14(4):255–293, 2001
[3] Certicom, Corp. ECC: Elliptic Curve Cryptography. [Online]. https://www.certicom.com/index.php/ecc. Accessed on June 12, 2012.
[4] Hankerson, D., Menezes, A., and Vanstone, S., "Guide to Elliptic Curve Cryptography". Springer- Verlag, 2004.
[5] Toorani, M., Meheshti, A. & Shirazi, L. Cryptanalysis of an Elliptic Curve-based Signcryption Scheme: International Journal of Network Security, Vol.10, No.1, PP.51–56, 2010.
[6] Singh, K & Chaudhary, M.K. "Scalar Multiplication Algorithms of Elliptic Curve Cryptography over GF, International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-3, Issue-1, 2013.
[7] Nguyen, M. V. (2009), Exploring Cryptography Using the Sage Computer Algebra System, School of Engineering and Science, Victoria University, 2009.
[8] Carella, N.A. Topics in Elliptic Curves Cryptography: The Groups of Points, Technical Report: School of Computer Science and Information Systems. Oxford University. 2010.

[9] Galbhiy, S., Karmakar, S., Sharma, M., Sharma, S. & Kowar, M.K. Application of Elliptic Curve Method in Cryptography: A Literature Review, (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 3 (3), 4499 – 4503 2012.

[10] Goundar, R.R. (2008). "Addition Chains in Application to Elliptic Curve Cryptosystems," Ph.D. dissertation, Kochi University, Japan, 2008.

[11] Malik, M. Y. (2009). "Efficient Implementation of Elliptic Curve Cryptography Using Low-power Digital Signal Processor," National University of Science and Technology Press (NUST) Press, Pakistan.

[12] Marnat, A. Speed Up for Scalar Multiplication on Elliptic Curves in Characteristic 2, 2010.

[13] Robert, A. "Elliptic Curves," class notes from postgraduate lectures, Department of Electrical Engineering, Neuchatel University, 2003.

[14] Moller, B. Algorithms for Multi-exponentiation. Selected Areas in Cryptography, volume 2259 of Lecture Notes in Computing Science, pages 165-180. Springer- Verglag, 2012.

[15] Toorani, M. & Beheshti, A. Cryptanalysis of an efficient signcryption scheme with forward secrecy based on Elliptic Curve. IEEE. Reprinted from the Proceedings of International Conference) on Computer and Electrical Engineering (ICCEE'08).

[16] Bajo, J., Luis, A. Gonzalez, A., and M.Corchado, J. "A Shopping Mall Multiagent System: Ambient Intelligence in Practice". As on March 2013. http://bisite.usal.es/webisite/archivos/publicaciones.

[17] Khatua, S., Das, A. and Yuheng, Z. "Agent Based Secured e-Shopping Using Elliptic Curve Cryptography," International Journal of Advanced Science and Technology Vol. 38, 2012.

[18] Courtney McT. and Sankaranarayanan, S., "Intelligent Agent based Hotel Search & Booking System", Proceedings of the 9th WSEAS International Conference on TELECOMMUNICATIONS and INFORMATICS, (2009).