IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-7, Issue-2, Feb 2019 E-ISSN: 2347-2693

Time Complexity Analysis of Cloud Data Security: Elliptical Curve and Polynomial Cryptography

D.Pharkkavi[1*], D. Maruthanayagam[2]

[1]Sri Vijay Vidyalaya College of Arts & Science, Dharmapuri, Tamilnadu, India
[2]PG and Research Department of Computer Science, Sri Vijay Vidyalaya College of Arts & Science, Dharmapuri, Tamilnadu, India

[*]*Corresponding Author*: pharkkavimca2012@gmail.com

DOI: https://doi.org/10.26438/ijcse/v7i2.321331 | Available online at: www.ijcseonline.org

Accepted: 10/Feb/2019, Published: 28/Feb/2019

Abstract- Encryption becomes a solution and different encryption techniques which roles a significant part of data security on cloud. Encryption algorithms is to ensure the security of data in cloud computing. Because of a few limitations of pre-existing algorithms, it requires for implementing more efficient techniques for public key cryptosystems. ECC (Elliptic Curve Cryptography) depends upon elliptic curves defined over a finite field. ECC has several features which distinguish it from other cryptosystems, one of that it is relatively generated a new cryptosystem. Several developments in performance have been found out during the last few years for Galois Field operations both in Normal Basis and in Polynomial Basis. On the other hand, there is still some confusion to the relative performance of these new algorithms and very little examples of practical implementations of these new algorithms. Efficient implementations of the basic arithmetic operations in finite fields GF(2^m) are need for the applications of coding theory and cryptography. The elements in GF(2^m) know how to be characterized in a choice of bases. A variety of basis used to represent field elements has a major impact on the performance of the field arithmetic. The multiplication techniques that make use of polynomial basis representations are very efficient in comparison to the best techniques for multiplication using the other basis representations. In this paper, focuses on user confidentiality and security protection in cloud, and that uses ***enhanced ECC technique over Galois Field GF(2^m).*** The strong point of the ***proposed ECPC (Elliptical Curve and Polynomial Cryptography) algorithm*** is based on the complexity of computing discrete logarithm in a large prime modulus, and the ***Galois Field*** allows mathematical operations to mix up data effectively and easily. ***This technique is mainly used to encrypting and decrypting data to ensure security protection*** and user confidentiality in the cloud computing. Results show that the performance of ECPC over Galois Field, in two region of evaluation, it has better than the other technique that is used for comparison purpose.

Keywords- Cloud Computing, Security, Cryptography, ECC, RSA, ECDH, ECDSA, Elliptical Curve and Polynomial Cryptography (ECPC)

I. INTRODUCTION

Security protection in cloud computing involves concepts like network security, control and equipment strategies deployed to protect data, infrastructure and applications correlated with cloud computing [1]. A significant feature of cloud is the notion of interconnection with a choice of materials that makes it necessary and difficult to securing these environments. Security problems in a cloud platform can lead to economic loss, also a bad reputation if the platform is oriented large public and are the cause behind the huge adoption of this new solution. In the cloud, the data stored for customers are represented as very important information. This is issue of the infringement of such data by an unauthorized third party is unacceptable to access information. There are two ways to attack data in Cloud [2]. First one is outsider attack and the second one is insider attack. The insider attack is an administrator in which they can have the possibility to hack the user's data. The insider attack is very complex to be recognized. So, the users must be very careful to analyze while saving their data in cloud storage. Therefore, the need to think of techniques which delay the use of data even though the data is accessed by the third party, they shouldn't obtain the actual data. As a result, the entire data must be encrypted before it is sent to the cloud storage [3]. Security allows the integrity, confidentiality, availability and authenticity of information. The improvement of technologies and their standardization makes available a set of protocols and algorithms for responding to these problems [4]. Several encryption techniques have been implemented and developed in order to give more secured data transmission process in cloud, like, RSA, Knapsack and DH for asymmetric category, Blowfish, RC4, AES, DES 3DES for symmetric category. [5, 6].

In this paper, ***focuses on user confidentiality and security protection in cloud,*** and that uses enhanced ECC (Elliptic Curve Cryptography) technique over Galois Field GF(2^m) [7]. The strong point of the proposed ***ECPC algorithm is based on the complexity of computing discrete logarithm*** in a large prime modulus, and the Galois Field allows mathematical operations to mix up data effectively and easily. ECC Algorithm affords message authentication and secure message integrity, with data confidentiality and non-repudiation of message. ECC is considered as a public key encryption method in which depends upon elliptic curve theory that can be utilized to generate more effective cryptographic keys [8]. ECC generates keys by using the properties of the elliptic curve equation rather than the traditional technique of key generation as the product of very large prime numbers. Because ECC aids to establish equivalent security protection with less power consumption and battery resource usage, it becomes broadly used for cloud applications.

In ECC, efficient finite field squaring is essential for quick implementation in software environments. In the binary finite field GF (2^m) Finite field arithmetic is a significant arithmetic operation Squaring is needed for several cryptographic techniques which are based on the DLP (Discrete Logarithm Problem) in the additive group of points on an Elliptic Curve defined over a finite field or the multiplicative group of a finite field [9]. ***In this paper, presents a new technique for performing binary finite field arithmetic in Polynomial Basis***.

Elliptic Curve

In Elliptic Curve, it has considered as a unique property that generates main utilization cryptography (i.e. its capability to obtain any two points on a specific curve), add them together and obtain a third point on the same curve [10]. The main operation of ECC is involved point multiplication which is multiplication of a scalar K with any point P on the curve to obtain another point Q on the same curve. An elliptic curve is termed by an equation, is of two variables, with coefficients. For the main intention of cryptography, the variable and coefficients are limited to a special type of set known as a **FINITE FIELD**. The general equation for an elliptic curve is:

$$\mathbf{y^2 + axy + by = x^3 + cx^2 + dx + e}$$

Where a, b, c, d and e are real numbers and x and y also take their values from real number. A simplified elliptic curve equation is given as:

$$\mathbf{y^2 – x^8 + dx + e}$$

In **ECC**, the Elliptic Curve is used to describe the members of the set over that the group is evaluated i.e. an operation on any two elements of the set will provide a result that is the member of the same set in addition to operations between them which defines how math work in the group. In real time scenario, the ECC is implemented over a finite prime field and in hardware, in which binary number are used; the finite field is represented GF (2^m). The GF is termed as a finite field namely Galois Field. The field of finite primes provides the ECC which allows encipher to encrypt the data very easily however, the cryptologist the process of initiating to attack the encrypted message is very complex. The GF (p) is termed as the field of integers module p, in which p is a very large prime number, and that comprise all the integers from 0 to p-1 in such case of square graph, p*p in size. To implement the ECC in software to handle prime numbers in addition to other number, ECC allows the improvement of potable chips which can be deployed over the mobile devices and provide a processor friendly encryption. ***ECC is implicated on hardware, point doubling and adding on elliptic curve*** and scalar multiplication over binary finite fields.

1.1 Polynomial Basis Representation

Galois fields are described as fields with a finite *field order q* that is also the number of elements in the field [11]. Here, a Galois field of order *q* is represented as *GF*(*q*). The order *q* of the field is a prime *p* or a power of a prime *pm* [12]. In general, Galois fields are used for cryptography purposes, other than there are several applications using Galois fields also (for example: error-correcting codes). Field orders (field sizes) used for cryptography are usually massive (e.g. $m > 150$) with the intention of make crypto analysis harder. The security offered by the cryptosystem usually increases exponentially when *m* becomes larger. For cryptography applications *m* in *GF*(2^m) must be a prime, in order to prevent a crypto analysis attack. Galois fields with a polynomial basis are most commonly used in ECC and therefore they are presented in detail. The other commonly used basis is called optimal normal basis (ONB), Polynomial basis has proven to be faster and easier to implement than optimal normal basis and it is therefore usually preferred to ONB. ***Galois field with a polynomial basis is generated with an irreducible polynomial over GF (2^N).***

For the polynomial basis representation, each element of the field represents a polynomial, f(x) of the form:

$$f(x) = a_0 + a_1x + a_2 x^2 + a_3x^3 + a_ix^i$$

All coefficients, a_i, of the polynomial are either one or zero. Therefore, each operation consists of subtraction, addition, and multiplication that are defined by means of polynomial arithmetic with the coefficients reduced modulo 2. For instance, the sequence of the bit 01100101 would signify the polynomial:

$$x^6 + x^5 + x^2 + 1$$

In view of the fact that operations like multiplication and squaring acquire inputs of size m bits and result in values 2m-1 bits, it should be a technique for reducing the result of these operations into elements of the Galois field. Accordingly, these values are minimized by a reduction polynomial of order m for an m bit field. Therefore taking the reduction polynomial raised to any power of x and

adding it to a field element will result in a value congruent to the original field element modulo the reduction polynomial which is already raised.

Field operations [13]: the arithmetic operations are termed on the elements of GF(2^m) while using a polynomial basis representation with reduction polynomial *p(x) as* follows:

Addition Operation

It can be done only using one n-bit XOR operation (equal to bit wise addition module 2). The sum of two elements A, B ∈ GF(2^m) is given by below equation.

$$C(x) = A(x)\ Xor\ B(x) = \sum_{i=0}^{m-1} (a_i\ Xor\ b_i)x_i$$

Square Operation

The binary representation of element's square is done by inserting a 0 bit between consecutive bits of the binary representation. The square of A ∈ GF(2^m) is given by below equation.

$$A^2(x) = \sum_{i=0}^{m-1} a_i x^{2i}$$

Multiplication Operation

In such case, two elements A(x), B(x) belongs to binary field GF(2^m) with irreducible polynomial P(x). Field multiplication done by two steps [14]:

1. Polynomial multiplication of A(x) and B(x)

$$C'(x) = A(x).\ B(x)$$

2. Reduction using irreducible polynomial p(x)

$$C(x) = C'(x) \bmod P(x)$$

Reduction Operation

Square and Multiplication operations require as above a reduction process, which is the process to decrease the order of resulting values from larger than m to less or equal to m.

$$C(x) = C'(x) \bmod P(x)$$

Inversion Operation

Inversion process is described the most time-consuming process while computing the scalar multiplication in ECC using Montgomery technique. In binary finite field, inversion process is obtaining a^{-1} for performing a non-zero element a ϵ GF(2^m):

$$(A.\ A^{-1}) = 1 \bmod f(x)$$

II. SECURITYALGORTIHMS

A. Elliptic Curve Cryptography (ECC)

ECC is considered one of the public key encryption methods that produce best cryptographic keys in accordance with the elliptic curve theory. It generates smaller keys within a short time. Before using the large prime numbers for key generation, ECC uses the properties of elliptic curves to produce keys. Elliptic curve is a nonsingular cubic curve with two variables in a certain field and an infinite rational point [15, 16]. Each and every user produces a public- private key pair, where the public key is applied for performing the signature verification and encryption process and the private key is also applied for signature generation and decryption process. In ECC, the high level of data security recognizes how to be achieved using a 164 bit key, where the traditional techniques require 1024 bit key. Data security using ECC algorithm

Key generation

- A selects random integer dA, which is A's private key
- A generates a public key PA = dA * B
- B selects a private key dB and generates a public key PB = dB * B
- A generates the security key Key = dA * PB
- B generates the security key Key = dB * PA

Signature Generation

- For signing a message m by sender of cloud A, using A's private key dA
- Calculate e = HASH(m), where HASH is a cryptographic hash function, such as SHA-1
- Select a random integer k from [1, n-1]
- Calculate r = x1 (mod n), where (x1, y1) = k * B. If r = 0, go to step III
- Calculate s = k^{-1}(e + dA * r) (mod n). If s = 0, go to step III
- The signature is the pair (r, s)
- Finally, send signature (r, s) to B

Encryption algorithm

Assume A sends an encrypted message to B

- A takes plaintext message m, and encodes it onto a point, pm,
- from the elliptic group
- A chooses another random integer, k from interval [1, p-1]
- The cipher text is a pair of points pc = [(kB), (pm + k * PB)]
- Send cipher text pc to B

Decryption algorithm

B will decrypt cipher text pc

- B computes the product of the first point from pc and its private key dB, which is kB * dB
- B takes this product and subtracts it from the second point from pc, (pm + k * PB) – kB * dB, since PB = dB * B, so the difference is pm
- Finally, B decodes pm to get the message m

Signature Verification

If B wants to authenticate A's signature, B must have A's public key pA

- Verify that r and s are integers in [1, n-1]
- Calculate e = HASH(m), where HASH is the same function used in the signature generation

- Calculate w = (s – 1) % n
- Calculate u1 = e*w % n and u2 = r*w % n
- Calculate (x1,y1) = u1 * B + u2 * PA
- The signature is valid if x1 = r % n, otherwise invalid

EXAMPLE:

1. Curve Size: Small , Curve Type: Real number, Curve attributes: a=6, b=20, Curve: $y^2 = x^3 + 6x + 20$,Point P = (1.3|5.47),Point Q = (-1.97|0.66),Point R = P + Q = (2083|-7.72)

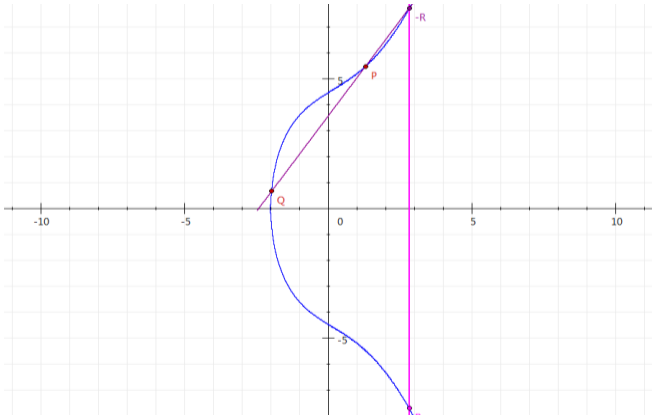

Fig. 1

2. Curve Size: Large, Curve Type: F(p), Select curve attributes: ANSI X9.62,Curve: prime192v1, Radix: 16 hexadecimal, Curve attributes: $y^2 = x^3 + 6x + 20$, where
a = fffffffffffffffffffffffffffffffefffffffffffffffc
b = 64210519e59c80e70fa7e9ab72243049feb8deecc146b9b1
p = fffffffffffffffffffffffffffffffefffffffffffffff
Base point G: Point P
x = 188da80eb03090f67cbf20eb43a18800f4ff0afd82ff1012
y= 7192b95ffc8da78631011ed6b24cdd573f977a11e794811
Base point G: point Q
x = 188da80eb03090f67cbf20eb43a18800f4ff0afd82ff1012
y=7192b95ffc8da78631011ed6b24cdd573f977a11e794811
Point R : R = P + Q
x= dafebf5828783f2ad35534631588a3f629a70fb16982a888
y= dd6bda0d993da0fa46b27bbc141b868f59331afa5c7e93ab

3. Curve Size: Large, Curve Type: F(2^m), Select curve attributes : ANSI X9.62, Curve: c2pnb163v1, Radix : 16 hexadecimal
a = 72546b5435234a422e0789675f432c89435de5242
b = c9517d06d5240d3cff38c74b20b6cd4d6f9dd4d9
m = 163

Base Point P:
x = 00000007 af699895 46103d79 329fcc3d 74880f33 bbe803cb
y= 00000001 ec23211b 5966adea 1d3f87f7 ea5848ae f0b7ca9f

Base point Q:
X=00000007 af699895 46103d79 329fcc3d 74880f33 bbe803cb
Y= 00000001 ec23211b 5966adea 1d3f87f7 ea5848ae f0b7ca9f
Point R : R = P + Q
X= 00000007 ee35173a 4ae9f401 c42fe4f6 01338998 bb745a37
Y= 00000003 6639922b a4e6c208 dc1f73b5 b137fc51 4a275c7d

4. Curve Size: Small, Curve Type: F(2^m), curve attributes : m=4, f = x^4+x+1,a=1,b=1,Curve: $y^2 + xy = x^3 + x^2 + 1$, Point P = (g12|g16), Point Q = (g6|g3), Point R = P + Q = (1|g10)

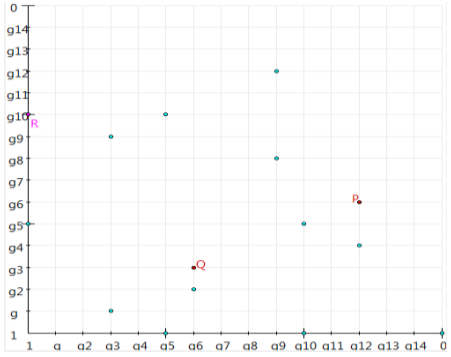

Fig. 2

B. ECDSA (Elliptic Curve Digital Signature Algorithms)

Signature algorithm is mainly used for authenticating a device or a message transmitted by the device. For instance, two devices are considered as A and B. The device A signs the message using the private key, to authenticate a message transmit by A. The device A sends the message and the signature to the device B. This signature can be verified only by using the public key of device A. Because the device B recognizes A‟s public key, it can be verified that the message is transmitted by A or not. ECDSA is a variant of the DSA (Digital Signature Algorithm) that works on elliptic curve groups. A signed message is sent from A to B, both have to agree up on Elliptic Curve domain parameters. Transmitter “A” have a key pair consisting of a private key dA (a randomly chosen integer less than n, where n is the order of the curve, an elliptic curve domain parameter) and a public key QA = dA * G (G is the generator point, an elliptic curve domain parameter) [17]. An overview of ECDSA process is defined below:

Signature Generation

For signing a message m by sender A, using A‟s private key dA

- Calculate e = HASH (m), where HASH is a cryptographic hash function, such as SHA-1
- Select a random integer k from [1,n − 1]
- Calculate r = x1 (mod n), where (x1, y1) = k *
- If r = 0, go to step 2
- Calculate s = k − 1(e + dAr)(mod n). If s = 0, go to step 2
- The signature is the pair (r, s)

Signature Verification

For B to authenticate A's signature, B must have A‟s public key QA

- Verify that r and s are integers in [1,n − 1]. If not, the signature is invalid
- Calculate e = HASH (m), where HASH is the same function used in the signature generation
- Calculate w = s −1 (mod n)
- Calculate u1 = ew (mod n) and u2 = rw (mod n)
- Calculate (x1, y1) = u1G + u2QA
- The signature is valid if x1 = r(mod n), invalid otherwise

EXAMPLE:

Signature originator: parkavi parkavi
Domain parameters to be used 'EC-prime239v1':
Chosen signature algorithm: ECSP-DSA with hash function SHA-1
Size of message M to be signed: 800 bytes
Bit length of c + bit length of d = 478 bits
Message = “Cloud – The technology of distributed data processing in which some **scalable** information resources and capacities are provided as a service to multiple external customers through Internet technology”.
Encrypted Data:
20 43 6C 6F 75 64 20 96 20 54 68 65 20 74 65 63 68 6E 6F 6C 6F 67 79 20 6F 66 20 64 69 73 74 72 69 62 75 74 65 64 20 64 61 74 61 20 70 72 6F 63 65 73 73 69 6E 67 20 69 6E 20 77 68 69 63 68 20 73 6F 6D 65 20 73 63 61 6C 61 62 6C 65 20 69 6E 66 6F 72 6D 61 74 69 6F 6E 20 72 65 73 6F 75 72 63 65 73 20 61 6E 64 20 63 61 70 61 63 69 74 69 65 73 20 61 72 65 20 70 72 6F 76 69 64 65 64 20 61 73 20 61 20 73 65 72 76 69 63 65 20 74 6F 20 6D 75 6C 74 69 70 6C 65 20 65 78 74 65 72 6E 61 6C 20 63 75 73 74 6F 6D 65 72 73 20 74 68 72 6F 75 67 68 20 49 6E 74 65 72 6E 65 74 20 74 65 63 68 6E 6F 6C 6F 67 79 2E
Elliptic curve E described through the curve equation: y^2 = x^3 + ax + b (mod p) :
a =
8834235323891921647916487503603088853144765972529603627924508606096998 36
b =
7385252174069924173485960880387817241648609717970989718912404233631938 66
Private key = 1545029212
Public key W=(Wx,Wy) (W is a point on the elliptic curve) of the signature originator:
W =
3392142815974640325568473898875509281419788613894901922436493589770267 94
Wx =
4849914437968509803607981758089212472773654934754196297821164601338157 56
Wy =
1128871025637972423124276818941332962152234778336249919670416479272048 75
Calculate a 'hash value' f (message representative) from message M, using the chosen hash function SHA-1.
f =
1229244187867772930448275680459591187106462377917

- ECDSA SIGNATURE as follows:

G has the prime order r and the cofactor k (r*k is the number of points on E):
k = 1
Point G on curve E (described through its (x,y) coordinates):
Gx =
1102820037495488564763485335411862045779050615048812422401495115944209 11
Gy =
8690784074355093787473518737930588685002103849460406946513687592170254 54
r =
8834235323891921647916487503603088848075503416916277522753454247028073 07
The secret key s is the solution of the EC discrete log problem W=x*G(x unknown)
S=
8673944514982004407568146807130793250216774525651397603746405115332395 20
Signature:
Convert the group element Vx (x co-ordinates of point V on elliptic curve) to the number i:
i =
8748447953424042422429704068790309821096255297759787159758029896391903 78
Calculate the number c = i mod r (c not equal to 0):
c =
8748447953424042422429704068790309821096255297759787159758029896391903 78
Calculate the number d = u^(-1)*(f + s*c) mod r (d not equal to 0):
d =
4997382264525469665605113156645278610879577473887696570203503575168128 89

- ECDSA VERIFICATION as follows:

If c or d does not fall within the interval [1, r-1] then the signature is invalid:
c and d fall within the required interval [1, r-1].
Calculate the number h = d^(-1) mod r:
h =
3917029456354149178221609422892610420329131968407031794154631242308962 39

Calculate the number h1 = f*h mod r:
h1 =
3280515274205410127575032472827991378063512478894545844364811239211490 57
Calculate the elliptic curve point P = h1 G + h2 W
Calculate the number h2 = c*h mod r:
h2 =
2509457036573211864438791338378101335036090875869887369994509709829843 7
(If P = (Px, Py) = (inf, inf) then the signature is invalid):
Px =
8748447953424042422429704068790309821096255297759787159758029896391903 78
Py =
5339203640627069202536364845901493435126414440978750925238673785381931 85
Convert the group element Px (x co-ordinates of point P on elliptic curve) to the number i:
i =
8748447953424042422429704068790309821096255297759787159758029896391903 78
Calculate the number c' = i mod r:
c' =
8748447953424042422429704068790309821096255297759787159758029896391903 78
If c' = c then the signature is correct; otherwise the signature is invalid

C. ECDH (Elliptic Curve Diffie Hellman Algorithm)

ECDH is considered into a key agreement protocol which allows two parties communication to establish using a shared secret key that can be used for performing private key techniques. Both parties exchange some public information to each other [18]. Using this public data and their own private data these parties calculates the shared secret. Any third party, who doesn't have access permission to the private details of each device, will not be capable to analyze the shared secret from the available public information. An overview of ECDH process is defined below. A shared secret between A and B is generated using ECDH, both have to agree up on Elliptic Curve domain parameters. Both the sender and receiver have a key pair consisting of a private key d (a randomly chosen integer less than n, in which n is the order of the curve, an elliptic curve domain parameter) and a public key = d * G (G is the generator point, an elliptic curve domain parameter).

Algorithm: Elliptical Curve Diffie-Hellman

- Alex and Benny agree on the elliptic curve E and base point $G(x_1,y_1)$
- Alex generates a random integer a $\in \{1, \ldots\ldots\ldots, n-1\}$ where n is the order of the group and number a is called private key of the Alex.
- Alex sends to Benny her public key $Q_a=aG=a(x_1,y_1)=(x_a,y_a)$
- Benny generates the random integer b $\in \{1, \ldots\ldots\ldots, n-1\}$ and b number is called private key of the Benny
- Benny sends to Alex his public key . $Q_a=bG=B(x_1,y_1)=(x_b,y_b)$
- Alex can then compute $(x_k,y_k)=aQ_b=a(bG)=abG$.
- Likewise, Benny can compute $(x_k,y_k)=bQ_a=b(aG)=abG$.
- The shared session key is x_k which is the x – coordinate of the point.

EXAMPLE:

Step 1: Set public parameters
Curve type: F (p), Curve Size: Small, Domain parameters: a=2,b=2,p=29, generator G=(1,11)
Step 2: Choose Secrets
Alex= 3
Benny=5
Step 3: Generate shared keys
Secret key (d): Q=d*G ,
Alex=(3,8)
Benny=(4,4)
Step 4: Exchange shared keys
Step 5: Generate common key
Key = sA*QB and key=sB*QA
S= (23,8)

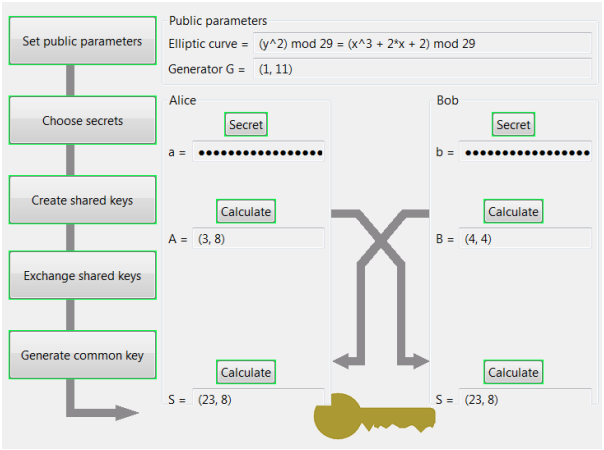

Fig.3

D. Proposed ECPC (Elliptical Curve and Polynomial Cryptography)

Polynomial basis multiplication is based on two main arithmetic operations over the binary polynomials. There are modulo reduction and polynomial multiplication irreducible polynomials. In this work, the so-called Mastrovito matrix is constructed from the coefficients of the first multiplicand and the irreducible polynomial defining the field. After that, the polynomial multiplication and modulo reduction steps are implemented together using matrix. Irreducible polynomials with special structures and low hamming

weights have been used to design efficient finite field multipliers.

In this section, illustrate the representation and multiplication of GF(2^m) elements in the polynomial basis.

The finite field GF(2^m) is an extension field of GF(2) and constitutes a dimension m vector space over it. The finite field GF(2) has only the elements 0 and 1. ***In this binary field, the addition and the subtraction are defined as XOR operation*** while the multiplication is defined as AND operation.

Step 1:

Let x ∈ GF(2^m) and be a root of the degree m irreducible polynomial over GF(2)

$$w(x) = x^m + w_{m-1}x^{m-1} + \cdots w_1 x + w_0 = 0$$

Then, the following set constitutes the polynomial basis in GF(2^m):

$$\{1, x, \ldots\ldots\ldots, x^{m-1}\}$$

With polynomial basis, GF(2^m) elements can be signified as degree m-1 polynomials as follows:

$$GF(2^m) = \{a(x)|a(x) = a_{m-1}x^{m-1} + \cdots . + a_1 x + a_0, \quad a_i \in GF(2)\},$$

Where the coefficients a_i are the polynomial basis coordinates in GF(2). When the elements of GF(2^m) are represented as ***polynomials over GF(2), their addition and subtraction are equivalent to the coefficient-wise XOR, denoted dy "+" in this paper.*** Also, because of the above equation, all arithmetic operations in GF(2^m) are performed modulo the irreducible polynomial w(x) chosen to construct the field. Let a(x) and b(x) be two field elements and c(x) be their product. Then,

$$c(x) = a(x)b(x) mod\ w(x).$$

Therefore, the polynomial basis multiplication has two steps: reduction modulo an irreducible polynomial and polynomial multiplication.

Step 2: Polynomial multiplication

Let d(x)=a(x)b(x) be the product of the polynomials representing the field elements. D(x) is the degree 2m-2 polynomial

$$d(x) = a(x)b(x) = \left(\sum_{i=0}^{m-1} a_i x^i\right)\left(\sum_{i=0}^{m-1} b_j x^j\right) = \sum_{k=0}^{2m-2} d_k x^k$$

Where

$$d_k = \sum_{i+j=k} a_i b_j, \quad 0 \le i, j, \le m-1, 0 \le k \le 2m-2.$$

Step 3: Modular Reduction

In the modular reduction c(x)=d(x) mod w(x), the degree 2m-2 polynomial d(x) is reduced by the degree m irreducible polynomial w(x) iteratively. The partial remainder after each reduction can be computed by the following iteration:

$$d^{(2m-2)}(x) = d(x), d^{(k-1)}(x) = d^{(k)}(x) + w(x)d_k^{(k)}x^{k-m}, m \le k \le 2m-2,$$

Here, $d^{(k)}(x)$ is a partial remainder of degree k and $d^{m-1}(x)=c(x)$. the iteration in above equation reduces $d^{(k)}(x)$ from degree k to k-1, since adding (coefficientwise XORing) $d^{(k)}(x)$ with polynomial

$$w(x)d_k^{(k)}x^{k-m} = \left(x^m + \sum_{i=0}^{m-1} w_i x^i\right) d_k^{(k)}x^{k-m} = d_k^{(k)}x^k + \sum_{i=0}^{m-1} d_k^{(k)} w_i x^{i+k-m} = d_k^{(k)}x^k + \sum_{i=k-m}^{k-1} d_k^{(k)} w_{i-(k-m)} x^i$$

Cancels its term with the order k.

Step 4:

The range of the irreducible polynomial w(x) may ease the modular reduction. Sparse irreducible polynomials having fewer nonzero terms are usually preferred for efficiency. A degree m irreducible polynomial over GF(2) which has r nonzero terms are in the form

$$x^m + x^{m_1} + x^{m_2} + \ldots . + x^{m_{r-3}} + x^{m_{r-2}} + 1.$$

Step 5:

At this point, r > 1 should be an odd number like 3 and 5. The sparse polynomials with three or five nonzero are named as shown below are known as trinomial and pentanomial respectively:

$$x^m + x^{m_1} + 1, x^m + x^{m_1} + x^{m_2} + x^{m_3} + 1.$$

Step 6:

Equally spaced irreducible polynomials are another option for efficient modular reduction. An equally spaced polynomial is represented in the form

$$x^{n_s} + x^{(n-1)_s} + \cdots . . x^s + 1,$$

Where ns=m.

Step7: The proposed polynomial interpolation technique in the elliptic curve ElGamal cryptosystem:

To discuss the algorithm of the modified elliptic curve ElGamal cryptosystem is performed. It will be transmitted the set of encrypted points as two polynomials that are constructed using Lagrange polynomial interpolation technique. The first ***polynomial will be encrypted as well to ensure the additional steps in this modified algorithm*** is meaningful for implementation. Here the algorithm:

- Alex selects her secret key, kA such that$1 \le k_A < n$. The gcd (k_A, n) = 1. She publishes her public key as $k_A P$
- Benny selects k_B such that$1 \le k_B < n$. The gcd (k_B, n) = 1. He encrypts each points such that (x_E, y_E) = P_m + k_B ($k_A P$).
- Benny makes polynomial A(x) based on the points (1, x_{E1}), (2, x_{E2}), (3, x_{E3}), (I, x_{EI}) where l indicates the number of encrypted points. Another polynomial B(x) is constructed based on the encrypted points (x_{E1}, y_{E1}), (x_{E2}, y_{E2}), (x_{E3}, y_{E3}),…, (x_{EI}, y_{EI}). Both polynomials are constructed using Lagrange polynomial interpolation.

- Benny adds the x-coordinate of k_B (k_AP) to each of the coefficients modulo p of the polynomial A(x) while the coefficients of polynomial B(x) remain unchanged. The encrypted polynomial A(x) denoted as A' (x). Benny transmits (k_BP, A'(x), B(x)) to Alex.
- Alex decrypts by multiplying her secret key such that $k_A(k_B P)$. Polynomial A(x) is obtained from A'(x) by deducting each coefficients using the x coordinate of $k_A(k_B P)$.
- Alex achieves x-coordinate of encrypted points by substituting x = 1, 2,…,I into A(x). Then x coordinate of encrypted points obtained is substituting into B(x) to get the y-coordinate of encrypted points.
- Alex obtains Pm such that $P_m + k_B (k_A P) - k_A (k_B P)$

EXAMPLE:

The exponent can be indicated by preceding it by the character E or e, as you can see in the example. Data must consist of two columns, *x* and *y*, to get the polynomial regression $y=a_n x^n+a_{n-1}x^{n-1}+...+a_2x^2+a_1x+a_0$.

Number of Data Points: 3

Polynomial Degree: 2

	x	y	Calculated y	Error
1.	12	16	16	0
2.	6	3	3	0
3.	1	10	10	0

Result: $y = 3.242424242 \cdot 10^{-1} x^2 - 3.66969697 x + 13.34545455$

Residual Sum of Squares: rss = 0

Coefficient of Determination: $R^2 = 1$

Chart

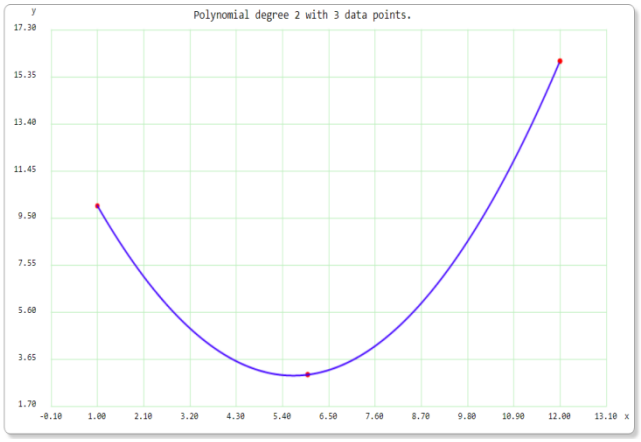

Fig. 4

Result

Mode: normal x,y analysis

Polynomial degree 2, 3 x, y data pairs.

Correlation coefficient (r^2) = 0.9999999999999987

Standard error = 3.162713435991701e-14

Coefficient output form: mathematical function:

f(x) = 1.3345454545454507e+001 * x^0
+ -3.6696969696969535e+000 * x^1
+ 3.2424242424242311e-001 * x^2

Table

x,	y,	%
1.00,	10.00,	0.00
1.55,	8.44,	5.00
2.10,	7.07,	10.00
2.65,	5.90,	15.00
3.20,	4.92,	20.00
3.75,	4.14,	25.00
4.30,	3.56,	30.00
4.85,	3.17,	35.00
5.40,	2.98,	40.00
5.95,	2.99,	45.00
6.50,	3.19,	50.00
7.05,	3.59,	55.00
7.60,	4.18,	60.00
8.15,	4.97,	65.00
8.70,	5.96,	70.00
9.25,	7.14,	75.00
9.80,	8.52,	80.00
10.35,	10.10,	85.00
10.90,	11.87,	90.00
11.45,	13.84,	95.00
12.00,	16.00,	100.00

III. EXPREMENTAL RESULTS

In this experimental performance analysis, the given algorithms perform on the basis of the following parameters on local system at different input size. In this phase, it describes the experimental parameters, platforms and key management of experimental algorithms.

Evaluation Parameters Performance of encryption algorithm is evaluated the following parameters.

- ***Key Generation time:*** In Key Generation Time considered the time that a key generation produces a keys.
- ***Encryption Time:*** The encryption time considered the time that an encryption technique generates a cipher text from a plain text.
- ***Decryption Time:*** The decryption time considered the time that a decryption technique generates a plain text from a cipher text.

Evaluation Platforms Performance of encryption algorithm is evaluated the following system configuration.

- Software Specification: Experimental evaluation on Eclipse Jee Mars with Java Development Kit (JDK) 8 Update 65, Matlab version 2014, Windows 8.1 Pro 64 bit Operating System.

- Hardware Specification: The entire algorithms are tested on Intel Core i5 (2.40 GHz) (fourth generation processor with 4GB of RAM) with 1 TB-HDD.

Experimental result for encryption algorithm ECC, ECDH, ECDSA, and ECPC are shown in table-1 that has been implemented many input file sizes: 889 bytes, 430 bytes, and 3159 bytes. In this experiment, key size of each algorithm that is used also mentioned in the below table. All the results are obtained with double time, for achieving higher accuracy hundred (100) examples of total execution time were taken afterward an average of hundred samples were taken for the comparative analysis and measurement among techniques and for the graph plotting as well. ***Encryption and Decryption time is evaluated in millisecond and the input size is taken in kilobytes.*** All the respective observation readings and graph are shown for all the analyzed algorithms on Cloud systems.

Table 1: Performance Comparison of Different algorithms

Algorithm	Key Size (bit)	File Size(bytes)	Key Generation Time(Milliseconds)	Encryption Time(Milliseconds)	Decryption Time(Milliseconds)
ECC	521	430	272	215	300
		889	303	237	352
		3159	345	288	381
ECDH	384	430	224	183	257
		889	233	229	301
		3159	256	277	338
ECDSA	256	430	185	124	212
		889	211	164	258
		3159	207	228	291
ECPC	163	430	110	97	175
		889	175	134	210
		3159	197	198	271

Key Generation Time:

The results of the experimentation of key generation time for ***different node*** sizes, threshold values and key sizes for ECC, ECDSA, ECDH, and ECPC are completed in efficient manner. The key sizes of ECPC 163 are comparable with other algorithms. This originally appears that ***ECPC is storage efficient.*** From the experimental results, it is the key generation time that gradually increases for a given key size with increase in threshold't' and node 'n'. ***As the key size increases, the generation time also exponentially increases***. The key generation time for all the four algorithms are illustrated in Figure 1**.** ***ECPC demonstrates desirable results compared to ECC.***

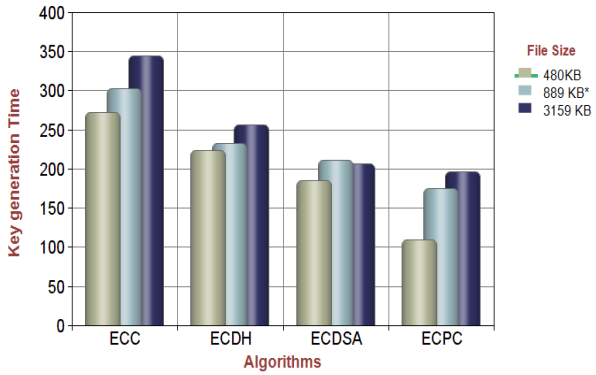

Figure 1: Comparison of Key Generation Time

Encryption Time:

Encryption throughput is capable to be determined by evaluating ***the total plaintext encrypted on total encryption time*** of various encryption techniques. Increased throughput results in ***decrease of power consumption***. The encryption time for different size of files are illustrated in Table -1. The corresponding curves for the encryption time illustrate in figure 2. In encryption graph (figure 2***), analyze that ECC curve consume time than other encryption techniques. ECDH, ECDSA curve*** is related to other method for performing small amount of load however, while the load is high then it obtain much time to encrypt data. ***ECPC curves show linearity while the data load increases.***

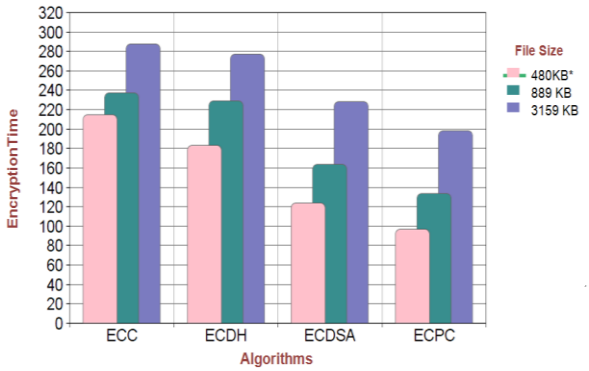

Figure 2: Comparison of Encryption Time

Decryption Time:

Decryption throughput can be determined ***by evaluating the total cipher text decrypted over the decryption time.*** It is format from the figure which ECPC algorithm is superior of all other algorithms. ECDSA has benefit over others as it can be seen over 256 bit key size and ECC has low performance in terms of power consumption and throughput than others. Also, in ***decrypting the different document files ECPC algorithm is superior of all other algorithms*** in terms of throughput and power consumption. ECPC has performed ***double throughput*** as it is compared to others so that it takes ***less time*** to decrypt the document file.

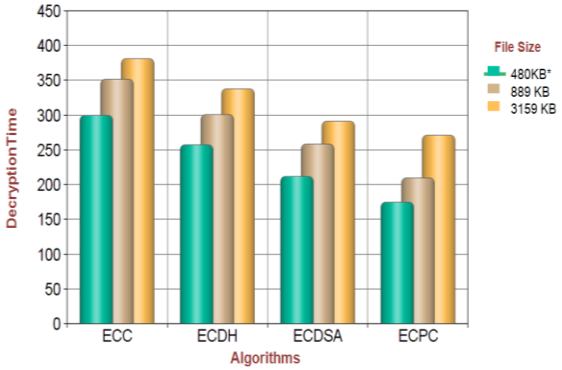

Figure 3: Comparison of Decryption Time

IV. CONCLUSION

Encryption and decryption algorithm keeps very significant contribution in security communication. This research paper emphasizes on the security of cloud user's information confidentiality protection using enhanced ECC (elliptic curve cryptography) algorithm over Galois Field GF (2^m). The strong point of ***the proposed ECPC algorithm is based on the complexity of computing discrete logarithm in a large prime modulus,*** and the Galois Field allows mathematical operations to mix up data effectively and easily. The Galois Field permits mathematical operations to merge data easily and effectively. From this paper illustrates***, the security performance comparison for broadly used encryption and decryption techniques such as ECC, ECDSA, ECDH, with our proposed ECPC algorithms.*** Depending on the text files used and the experimental result it has been decided that ECPC algorithm consumes ***least encryption*** time and ECC and other algorithms consume longest encryption time. Moreover, that ***least decryption*** time of ECPC algorithm is performed better than other algorithms.

REFERENCES

[1]. Wolf Halton, "Security Solutions for Cloud Computing", July 15, 2010.

[2]. Wolf Halton, Opensource and security on "Security Issues and Solutions in Cloud Computing", July 25, 2010, wolf in cloud computing, Tech Security.

[3]. L. Arockiam, S. Monikandan « Data Security and Privacy in Cloud Storage using Hybrid Symmetric Encryption Algorithm » International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 8, August 2013

[4]. Ponemon Institute and CA "Security of cloud computing Users: A study of Practitioners in the US & Europe". May 12, 2010.

[5]. Ryan K L Ko, Peter Jagadpramana, Miranda Mowbray, Siani Pearson, Markus Kirchberg , Qianhui Liang , Bu Sung Lee, "TrustCloud: A Framework for Accountability and Trust in Cloud Computing" 2011 IEEE World Congress on Services.

[6]. Muhammad Rizwan Asghar, Mihaela Ion, Bruno Crispo, "ESPOON Enforcing Encrypted Security Policies in Outsourced Environment", 2011 Sixth International Conference on Availability, Reliability and Security.

[7]. Xu Huang, Pritam Gajkumar Shah and Dharmendra Sharma, "Multi-Agent System Protecting from Attacking with Elliptic Curve Cryptography," the 2nd International Symposium on Intelligent Decision Technologies, Baltimore, USA, 28-30 July 2010.

[8]. Xu Huang, Pritam Shah, and Dharmendra Sharma, "Minimizing hamming weight based on 1"s complement of binary numbers over GF (2m)," IEEE 12th International Conference on Advanced Communication Technology, Phoenix Park, Korea Feb 7-10, 2010. ISBN 978-89-5519-146-2, pp.1226-1230.

[9]. Xu Huang, Pritam Shah, and Dharmendra Sharma, "Fast Algorithm in ECC for Wireless Sensor Network," The International MultiConference of Engineers and Computer Scientists 2010, Hong Kong, 17-19 March 2010. Proceeding 818-822.

[10]. Pritam Gajkumar Shah, Xu Huang, Dharmendra Sharma, "Analytical study of implementation issues of elliptical curve cryptography for wireless sensor networks," The 3rd International Workshop on RFID & WSN and its Industrial Applications, in conjunction with IEEE AINA 2010, April 20-23, 2010, Perth, Australia.

[11]. Omura, J.K., Massey, J.L.: Computational method and apparatus for finite field arithmetic, United States Patent 4,587,627 (1986)

[12]. Robert, J., McEliece: Finite Fields for Computer Scientists and Engineers. The Kluwer International Series in engineering and computer science. Kluwer Academic Publishers, Dordrecht (1987)

[13]. Karatsuba, A., Ofman, Y.: Multiplication of multidigit numbers on automata. Sov. Transaction Info. Theory 7(7), 595–596 (1963)

[14]. Rodriguez-Henriquez, F., Kog, Q.K.: On Fully Parallel Karatsuba Multipliers for GF (2m).In: International Conference on Computer Science and Technology (CST), pp. 405–410 (2003)

[15]. Setiadi, I., Kistijantoro, A.I. and Miyaji, A. (2015) Elliptic Curve Cryptography: Algorithms and Implementation Analysis over Coordinate Systems. 2015 2nd International Conference on Advanced Informatics: Concepts , Theory and Applications, Chonburi, 19-22 August 2015, 1-6. https://doi.org/10.1109/icaicta.2015.7335349

[16]. J. Krasner "Using Elliptic Curve Cryptography (ECC) for Enhanced Embedded Security –Financial Advantages of ECC over RSA or Diffie-Hellman (DH)" Embedded Market Forecasters American Technology International, Inc. November 2004

[17]. D. Hankerson, A. Menezes, S. Vanstone, "Guide to Elliptic Curve Cryptography"Ch- , Pp 76-78 Springer, 2004.

[18]. Garg, V. and Ri, S.R. (2012) Improved Diffie-Hellman Algorithm for Network Security Enhancement. International Journal of Computer Technology and Applications , 3, 1327-1331.

[19]. Setiadi, I., Kistijantoro, A.I. and Miyaji, A. (2015) Elliptic Curve Cryptography: Algorithms and Implementation Analysis over Coordinate Systems. 2015 2nd International Conference on Advanced Informatics: Concepts , Theory and Applications, Chonburi, 19-22 August 2015, 1-6. https://doi.org/10.1109/icaicta.2015.7335349

Authors Profile

D.Pharkkavi received her **M.Phil** Degree from Tiruvalluvar University, Vellore in the year 2013. She has received her **M.C.A** Degree from Anna University, Chennai in the year 2012. She is pursuing her Ph.D (Full-Time) Degree at Sri Vijay Vidyalaya College of Arts & Science, Dharmapuri, Tamilnadu, India. Her areas of interest include Cloud Computing and Mobile Computing.

Dr.D.Maruthanayagam received his **Ph.D** Degree from Manonmaniam Sundaranar University, Tirunelveli in the year 2014. He received his **M.Phil** Degree from Bharathidasan University, Trichy in the year 2005. He received his **M.C.A** Degree from Madras University, Chennai in the year 2000. He is working as **HOD Cum Professor**, PG and Research Department of Computer Science, Sri Vijay Vidyalaya College of Arts & Science, Dharmapuri, Tamilnadu, India. He has around **18 years** of experience in academic field. He has published **4 books**, **31 papers** in International Journals and **30 papers** in National & International Conferences so far. His areas of interest include Computer Networks, Grid Computing, Cloud Computing and Mobile Computing.

