Enhanced Data Security Technique to Protect User's Data in the Public Cloud Environment

S. Hendry Leo Kanickam[1] & Dr. L. Jayasimman[2]
Research Scholar, Department of Computer Science, Bishop Heber College [1]
Assistant Professor, Department of Computer Applications, Bishop Heber College [2]
Affiliated to Bharathidasan University, Trichy, Tamilnadu, India.
henryleo.msc@gmail.com[1], simmanjaysee@gmail.com[2]

Abstract

Cloud computing is a rising commercial infrastructure paradigm that assures to remove the need for maintaining and handling much expensive computing hardware. As the market develops, the threat to data also grows. To secure the data from unauthorized access and to ensure that the data are intact with the proposed scheme, and that solves the issue of privacy, integrity, consistency and unauthorized access problems. This research paper proposed a hybrid algorithm PHECC (Polynomial based Hashing and Elliptic Curve Cryptography), and that includes PH with ECC security algorithms to promise the clients about their data security in the cloud. In the current scenario, the hybrid algorithm almost guarantees for data security. The performance of the proposed technique is compared with other hybrid algorithms.

Keywords:*Key Words: Secure Hash Algorithm -2, Elliptic Curve Cryptography, Elliptic-Curve Diffie–Hellman, Elliptic Curve Digital Signature Algorithm, Polynomial Based Hashing and Elliptic Curve Cryptography.*

1. Introduction

Cloud computing assures to aid organizations, and their IT departments are more efficient and capable of handling cost-effectively deliver new services that allow their businesses to development thrive [1]. However, the assurance of the cloud performance cannot be accomplished until IT professionals have more self-assurance in the privacy, safety and security of the cloud environment. Several privacy and security threats, like risk or the malware of a malicious insider, emerge to be ubiquitous facets of the IT landscape nowadays and should be addressed as part of both the national and international cybersecurity agenda [2]. The security protection faces the challenges by organizations wishing to utilize the cloud services are not completely different from the threats and traditional security issues. The same external and internal threats are present and require proper disastrous management and risk mitigation policies with the purpose of protecting security and privacy.To protect the data transmission process via the internet, data could be protected by the encryption process. The encryption process converts data using any encryption technique via utilizing key in mixed form. Only the user has the encryption key to obtain encrypted data using the deception process. Key-based encryption is classified into two major categories: symmetric key encryption and asymmetric key encryption [3].Besides, there is a third category of encryption process known as the hash process to use secure authentication. Many hybrid algorithms are developed for security intention in a cloud computing environment like ECC-Md5, Blowfish-MD5, AES-MD5, ECC-SHA, and so on. These old techniques are not helpful last year due to their low-security level, and the MAC has been broken effortlessly by hackers [4, 5]. Here, introduces a new security technique by using a hybrid cryptosystem for data security in the cloud environment. The hybrid algorithm is mainly used to implement a combination of two familiar and most extensively used cryptographic Hashing and ECC techniques. The new proposed Hybrid algorithm merges the PH (Polynomial based Hashing) for authentication and ECC operations for the security mechanism. The performance of the proposed algorithm is comparable with other hybrid algorithms example, ECC-SHA, ECDH-SHA, and ECDSA-SHA.

2. Related Work

To secure the data prevents hackers or unauthorized access in a cloud environment at the time of data transmission by encrypting the user data. There is much discussion now about how to construct a better hash algorithm. The main complexity stems from the truth that the design principles of a hash function are not completely understood. Some of the techniques have already been discussed in the literature survey to create new hash functions from old functions. Several attempts have been made to recognize better design principles of hash functions; however, this work seems to be still

developing. A new hash algorithm is this article aims to describe, which incorporates a few of the aspects mentioned above. Particularly, the output length of the function is a parameter that can be set at the beginning.
Additionally, the degree of collision resistance is based on another parameter which can be identified at the outset process. After reviewing relevant polynomial-evaluation algorithms, it launches high-speed cryptography to generating a faster message-authentication, particularly high-speed computation of authenticators that secure messages against falsification. In existing polynomial-evaluation technique has a noticeable effect on cryptographic speed. The new proposed authentication algorithm combines the benefits of a small number of multiplications and a minimal number of variables using Elliptic Curve Polynomial operations. MAC (Message authentication codes) using polynomial evaluation has the benefit of involving (a very short key) even for very large messages. An efficient procedure evaluates using polynomials over a finite field to be utilized in the construction of a fast MAC. The ideas about the Hybrid algorithm are the starting point for this research improvement. The new proposed Hybrid algorithm combines the PH (Polynomial based Hashing) for authentication and ECC operations for the security mechanism. Hence, it presents a hybrid algorithm for enhanced network security. To ensure the integrity of the transmitted data, the data is mainly subjected to a new proposed **polynomial based Hashing and Elliptic Curve Cryptography (PHECC)** hash algorithm. The system takes data, encrypt data, decrypt data and upload all these data into the cloud, and it also creates a hash value using **PH (Polynomial Hashing)** for checking the integrity of the data. The data downloading and uploading will be encrypted/decrypted using the ECC algorithm. After reducing the data, the signature element is sent to authority ECC to be digitally signed, and the message digest obtained by this process is also encrypted/decrypted using the ECC technique. The same process followed to decryption.

3. How to Hash into Elliptic Curve

Some of the elliptic curve cryptosystems require hashing into an elliptic curve, for example, other cryptography techniques. A particular supersingular elliptic curve is used (elliptic curve scheme) in which exists a one-to-one mapping *f* from the base field $\mathbb{F}_p$ to the curve. This allows to hash using *f (h (m))* in which *h* represents a classical hash function. Moreover, the password-based authentication protocols are mainly used to provide another context in which hashing into an elliptic curve is sometimes needed.In deterministic polynomial time was issued by Shallue and Woestijne, the first algorithm mapping $\mathbb{F}_{p^n}$Into an elliptic curve [6]. Thehybrid algorithms provide any elliptic curve *E* defined over $\mathbb{F}_{p^n}$, maps elements of $\mathbb{F}_{p^n}$Into *E* in deterministic polynomial time, when p^n = two mod 3. The algorithms depend on a rational function, explicit function from $\mathbb{F}_{p^n}$ to *E*, and that can be implemented in O ($\log^3 q$) time and a constant number of operations over $\mathbb{F}_{p^n}$Moreover, that technique depends on computing a cube root and simpler than the Shallue and Woestijne algorithm [7].
An application illustrates how to hash efficiently and deterministically into an elliptic curve using two different constructions. The first construction is a one-way function if the underlying hash function is one-way [8, 9]. Additionally, the second construction achieves collision resistance, even if the underlying hash function is collision-resistant.Given a function f into an elliptic curve *E*, explain two constructions of hash functions into *E*. Define *L* as the maximal size of $f^{-1}(P)$ where *P* is any point on *E*:

$$L = \max_{P \in E}(|f^{-1}(P)|) \qquad (1)$$

For our encoding function $f_{a,b,}$ have L ≤ 4. Note that if work in a subgroup of *E* of order *n* with cofactor *r*, use the encoding function$f'_{a,b} = r.f_{a,b}$. If *r* is relatively prime to *n*, then must have L≤ 4r. The first construction is as follows: given a hash function h:

$\{0,1\}^* \rightarrow \mathbb{F}_p$is defined by,

$$\boldsymbol{H\,(m) = f\,(h\,(m))} \quad (2)$$

As a hash function into the curve $E_{a,b}(\mathbb{F}_p)$. In the following, show that *H* is one-way if *h* is one way. **H**ence, the second construction is given as follows. A security parameter *k* and an integer $q = p^n$ with $q \geq 2^k$, consider the following family of functions:

$$v_{c,d_{c,d\in\mathbb{F}_q}} : \{0,1\}^* \rightarrow \mathbb{F}_q \qquad x \rightarrow c.x + d \quad (3)$$

Where *x* is noticed as an element in $\mathbb{F}_p$. It is trouble-free to perceive that this family is 1=q^2-pairwise independent. Given an elliptic curve *E*, merge the encoding function *f* with the functions in the $v_{c,d}$ family to get a collision-free family *G*:

$$G = \left(f \circ v_{c,d}\right)_{c,d\in\mathbb{F}_q} : \{0,1\}^* \rightarrow E(\mathbb{F}_q) \; x \rightarrow f(c.x + d) \quad (4)$$

As a final point, given a family H of collision-resistant functions $h:\{0,1\}^* \rightarrow \{0,1\}^k$build the following family of hash functions into the curve E:

$$H_E = \left(f \circ v_{c,d}\right)_{c,d\in\mathbb{F}_{q_{h\in H}}}:\{0,1\}^* \rightarrow E\left(\mathbb{F}_q\right) \;\; x \rightarrow f(c.h(m)+d)(5)$$

4. Secure Hash Algorithm -2 (Sha-2)

The Secure Hash Algorithm (SHA) is described as a cryptographic hashing algorithm, and this algorithm was introduced by NSA [10]. The new versions SHA-2 bear the same underlying logical binary operations, the resemblance of structure, and modular arithmetic function as that of SHA-1 without sharing its weaknesses. A set of six hash functions are considered as the sequence of 224, 256, 384 or 512 bits used in SHA- 2. In favor of SHA- 256 and SHA-224 bits, every message block has represented 512 bits, which are denoted as a sequence of 32-bit words. For SHA-512 and SHA-384, each message block includes 1024 bits as a sequence of 64-bit words. SHA-256 performs on 32-bit words, and SHA-512 manages on 64-bit words. Both SHA-512 and SHA-256 are new hash functions that utilize different additive constants and shift amounts, and their structures are virtually identical, varying only in the number of rounds. SHA-224 and SHA-384 are the truncated versions of SHA-256 and SHA-512. SHA-512/256 and SHA-512/224 are truncated versions of SHA-512. At this point, a step summary of SHA-512 is processing as given below [11]:

Steps Overview:

- SHA-512 is a variant of SHA-256 that controls eight 64-bit words. Initially, the message to be hashed.
- Padded with its length in such that way the result is a multiple of 1024bits long, and then the next step,
- Parsed into 1024-bit message blocks $M^{(1)}$, $M^{(2)}$, ….., $M^{(N)}$.
- The message blocks process one by one at a time: starting with a fixed first hash value $H^{(0)}$, evaluate sequentially as follows

$$H^{(i)} = H^{(i-1)} + C_M{}^{(i)}(H^{(i-1)}) \qquad (1)$$

  Where C is the SHA-512 compression function and + means word-wise mod 2^{64}addition. $H^{(N)}$ is the hash of M.
- SHA-512 applies over six logical functions. Each function is denoted as x, y, and z and controls on 64-bit words. The result of each function is formed as a new 64-bit word.

SHA example

Hash function: SHA 512

The base of Hash value: hexadecimal

File Name: example.txt

Message: Cloud computing security refers to the set of procedures, processes and standards designed to provide information security assurance in a cloud computing environment.

Hash value of original file: CF 2B 58 43 0E A0 F8 4B 92 8F 6F 82 30 12 B6 5A 12 EF F9 24 67 F3 1F F7 EB 08 FF F5 B6 EE 88 9A 3F 0A 16 1D 41 6D 73 63 B2 30 5E C0 48 C1 F5 5C 81 FA 39 16 BB 57 7A 3F 3D B4 C4 75 40 6E 05 F8

Hash value of modified file: CF 2B 58 43 0E A0 F8 4B 92 8F 6F 82 30 12 B6 5A 12 EF F9 24 67 F3 1F F7 EB 08 FF F5 B6 EE 88 9A 3F 0A 16 1D 41 6D 73 63 B2 30 5E C0 48 C1 F5 5C 81 FA 39 16 BB 57 7A 3F 3D B4 C4 75 40 6E 05 F8

Difference between the hash value of the original and of the modified file:

00000000#0.00% of the bits differ (0 of 512).

Longest identical bit sequence: offset 0, length 512.

5. ECC (Elliptic Curve Cryptography)

ECC is a public-key cryptosystem, and each user has a private key and a public key. The private key is mainly utilized for performing with two operations, such as the decryption process and signature generation process. The public key is mainly used for handling two operations, like the encryption process and signature verification process, as Elliptic curves use an extension to other current cryptosystems. The security pattern of ECC is quite significant, and it does not affect the side-channel attacks. Variable key lengths have been used for the encryption and are varied about the data blocks to supply a sufficient amount of cover the data. An elliptic curve over a field K is considered as a nonsingular cubic curve in two variables, f(x, y) =0 with a rational point (a point at infinity stage). The field K is typically taken to be the complex numbers, rational, and algebraic extensions of rational, p-adic numbers, or a finite field [12]. The key generation is an important part where we have to generate both public key and private key. The sender will be encrypting the message with the receiver's public key, and the receiver will decrypt its private key. To choose a number' handle within the range of 'n.' Using the following equation, generate the public key d = the random number and have to be chosen within the range of (1 to n-1). P is the point on the curve. The public key denotes'Q,'and the private key represents.'

Encryption: Let'm' be the message is transmitted to represent this message on the curve. These have in-depth implementation details. Consider'*m'*has a point'*M'*on the curve *'E.'*Randomly select 'k' from [1 – (n-1)]. Two ciphertexts will be generated. It isC1 and C2.

$$C1=k*P,\ c2=M + k*Q \quad (2)$$

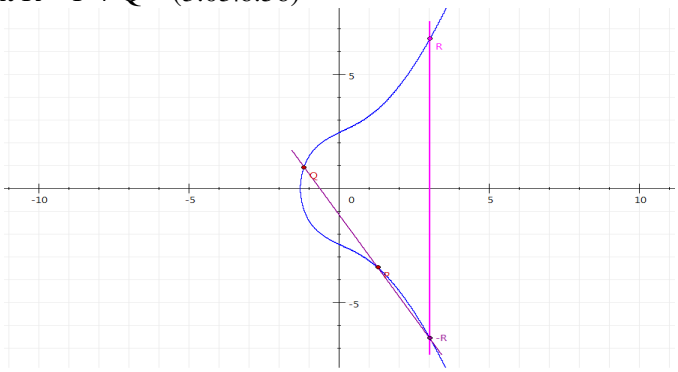
Decryption: Have to get back the message'm' that was send

$$M=c2-d*c1 \quad (3)$$

M is the original message that we have sent.

ECC EXAMPLE

1. Curve Size: Small , Curve Type: Real number, Curve attributes: a=3, b=6, Curve: $y^2 = x^3 + 3x + 6$,Point P = (1.3|-3.47),Point Q = (-1.17|0.92),Point R = P + Q = (3.03|6.56)



2. Curve Size: Large, Curve Type: F(p), Select curve attributes: ANSI X9.62,Curve: prime192v1, Radix: 16 hexadecimal, Curve attributes: $y^2 = x^3 + 3x + 6$, where

a = fffffffffffffffffffffffffffffffefffffffffffffffc

b = 64210519e59c80e70fa7e9ab72243049feb8deecc146b9b1

p = fffffffffffffffffffffffffffffffefffffffffffffff

Base point G: Point P

x = f0a3ab46f69d1a16f3212a9d90ece4582c8bfad13e91db82

y= 36438e499c894d67bca962ec83fb5e2fb4a2646fe4864195

Base point G: point Q

x = cdc910ab77a91d9676b6b05469ea516ae58152b1a7fec53f

y=303f7072446138caab2762f4cc991d82695ca632ba01c234

Point R : R = P + Q

x= 269f63e40da89dba0ab49f9723c77eace1e41bddb44f6a52

y= da20e0400577f70be26e8c7d0d6a8df87483ef7903b0ad25

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 = 00000006 a2adda70 3fd11169 75c07a51 801a06af 992e40f9
y= 00000007 bb784d10 6afdcf77 1b2288b2 ace3dbfd b190a5c5
Base point Q:
X=00000002 310338bb 5da9adae a49d77c9 c0a57d0b ad391353
Y= 00000005 37055f95 b9649133 f6616fde 5a9d7082 dfdc3210
Point R : R = P + Q
X= 00000006 d649e67c 8cad57b1 eb57ad15 20517342 f86aeab8
Y= 00000002 78f1fdcf db30d5f7 cc036fe6 f0600ac3 8b09f818

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 = (g9|g12), Point Q = (g6|g3), Point R = P + Q = (g5|g10)

6. ECDH

Elliptic-curve Diffie–Hellman (ECDH) is considered as an anonymous key agreement (In cryptography, a key-agreement protocol is a protocol in which two or more parties know how to agree on the key protocol in such that method that both influence the outcome and income) [13]. In this two parties, each having an ellipticcurve public-private key pair, a shared secret (A shared secret is a cryptographic key or data that is only known to the parties involved in a secured communication) establish over an insecure channel (to a secure channel, an insecure channel is unencrypted and may be subject to eavesdropping).This shared secret recognizes how to be directly used as a key, or to derive another key (expressed as DK = KDF(Key, Salt, Iterations) in which KDF is the key derivation function, and DK represents the derived key, key denotes the original password or key, a Salt is a random number that performs cryptographic salt, and Iterations refers to the number of iterations of a sub-function). The key or DK can be used to encrypt subsequent communications utilizing a symmetric-key cipher (using that the symmetric ciphers apply the same cryptographic keys for performing both the encryption and decryption of plaintext and ciphertext respectively). ECDH is also used as an analogous scheme, and that depends upon adding of points on an elliptic curve cryptosystem. The basic operation is merged to produce a primitive function called as a keyed one-way function. A keyed one-way function a function that obtains two inputs, one of which is a private key and generates one output. Given the two inputs, it must be straightforward to calculate the output. However, it should be computationally infeasible to evaluate the key, using only the other input and the output. In such a way, every party can utilize their private key without revealing it to anyone else, either the other party or an eavesdropper.

Algorithm

For instance, two persons, Alex and Benny, desire to exchange a secret key with each other as given below in the following steps are used:

- Initially, the public and private keys are generated by Alex and Benny. The private key represents dA, and the public key denotes HA = dA G for Alex, and the keys dB and HB = dB G for Benny. Note that both Alex and Benny are using the same domain parameters: the same base point G on the same elliptic curve on the same finite field.
- Alex and Benny swap over their public keys HA and HB, using an insecure channel. The Man in the Middle would interrupt HA and HB, but will be capable of discovering neither dA nor dB without solving the discrete logarithm problem.
- Alex computes S = dA HB (using own private key and Benny's public key), and Benny evaluates S = dB HA (using own private key and Alex's public key). Note that S is the same for both Alex and Benny, iS = dA HB = dA (dB G) = dB (dAG) = dB HA

ECDH EXAMPLE:
Step 1: Set public parameters
Curve type: F (p), Curve Size: Large, Domain parameters: a=3,b=6,p=47, generator G=(15,18)
Step 2: Choose Secrets
Alex= 5
Benny=6
Step 3: Generate shared keys
Secret key (d): Q=d*G,
Alex= (44,8)
Benny= (15,29)
Step 4: Exchange shared keys
Step 5: Generate common key
Key = sA*QB and key=sB*QA
S= (44, 39)

7. ECDSA (Elliptic Curve Digital Signature Algorithm)

ECDSA (Elliptic Curve Digital Signature Algorithm) is the elliptic curve analog of the DSA (Digital Signature Algorithm). ECDSA is an ECC approach that needs a hash function and a few modular operations. ECDSA is similar to ElGamal's signature technique, but it utilizes a slightly different signature verification method, which creates verification of signatures faster [14]. The main difference between ElGamal's digital signature system and ECDSA is that in ElGamal's system, the verification process needs three multiplications of integer times a point, but ECDSA has only two multiplications of integer times a point are needed. These multiplication operations are performed the most expensive parts of these techniques. The procedure of ECDSA is explained as given below. In ECDSA, the signature generation and verification are related to DSA, other than the key generation depends upon on ECC algorithm [15].
Key Pair Generation: In ECDSA, the key pair generation depends upon the domain parameters. Given the elliptic curve E over Z_p with several points that are divisible by the large prime n,
1. Select a point $P(x_p, y_p)$ on the curve and a random integer $d \in [1, n-1]$.
2. To evaluate $Q(x_q, y_q) = dP$, make sure the point Q is also on the curve.
3. The public key is (E,P, n,Q), and the private key is d.
Signature Generation: Given a message m to be signed and the private key d,
1. Choose a random integer $k \in [1, n-1]$.
2. Compute $(x_1, y_1) = kP$, convert x_1 into integer and $r = x_1 \bmod n$. (Return to step 1 even if $r = 0$)
3. Evaluate $s = k^{-1}(SHA512(m) + dr)$. (Return to step 1 if $s = 0$)
4. Signature is (r, s) pair.

Signature Verification: Given the signature pair (r, s) on message m and public key (E,P, n,Q),
1. Check whether that integers $r, s \in [1, n-1]$.
2. And then evaluate $w = s^{-1} \bmod n$.
3. Evaluate $u_1 = SHA512\,(m)w \bmod n$, $u_2 = rw \bmod n$.
4. Compute $(x_0, y_0) = u_1P + u_2Q$, convert x_0 into integer and $v = x_0 \bmod n$.
5. Finally, compare v and r, accept the signature only if v = r.

ECDSA EXAMPLE:
Signature originator: HendriHendri
Domain parameters to be used 'EC-prime239v1':
Chosen signature algorithm: ECSP-DSA with hash function SHA
Size of message M to be signed: 800 bytes
The bit length of c + bit length of d = 477 bits
File Name = example.txt
Message = Cloud computing security refers to the set of procedures, processes and standards designed to provide information security assurance in a cloud computing environment.

Encrypted Data:
20 43 6C 6F 75 64 20 63 6F 6D 70 75 74 69 6E 67 20 73 65 63 75 72 69 74 79 20 72 65 66 65 72 73 20 74 6F 20 74 68 65 20 73 65 74 20 6F 66 20 70 72 6F 63 65 64 75 72 65 73 2C 20 70 72 6F 63 65 73 73 65 73 20 61 6E 64 20 73 74 61 6E 64 61 72 64 73 20 64 65 73 69 67 6E 65 64 20 74 6F 20 70 72 6F 76 69 64 65 20 69 6E 66 6F 72 6D 61 74 69 6F 6E 20 73 65 63 75 72 69 74 79 20 61 73 73 75 72 61 6E 63 65 20 69 6E 20 61 20 63 6C 6F 75 64 20 63 6F 6D 70 75 74 69 6E 67 20 65 6E 76 69 72 6F 6E 6D 65 6E 74 2E

Elliptic curve E described through the curve equation: y^2 = x^3 + ax + b (mod p):
a=88342353238919216479164875036030888531447659725296036279245086060969983 6
b=73852521740699241734859608803878172416486097179709897189124042336319386 6
Private Key = 1547465310

Create a random one-time key pair (secret key, public key) = (u,V) with the domain parameters of 'EC-prime239v1' (V=(Vx,Vy) is a point on the elliptic curve):

u=252015484171984037028194345951735738458769524510546017515617919140545729
Vx=82527945667413456894398314536185868994235306901932550287283 1183065044791
Vy=11103720097354788368211490047913297084662046270419788310585 1475989597207
Calculate a 'hash value' f (message representative) from message M, using the chosen hash function SHA.
f=10168359407591437558955011081399370465124679006 99
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=11028200374954885647634853354118620457790506150488124224014951159442091 1
Gy=86907840743550937874735187379305886850021038494604069465136875921702545 4
r=88342353238919216479164875036030888480755034169162775227534542470280730 7
The secret key s is the solution of the EC discrete log problem W=x*G(x unknown)
S= 21628799355765130429672543459245710750524708572578739346519877709436067 0
Signature: Convert the group element Vx (x co-ordinates of point V on elliptic curve) to the number i:
i=82527945667413456894398314536185868994235306901932550287283 1183065044791
Calculate the number c = i mod r (c not equal to 0):
c=82527945667413456894398314536185868994235306901932550287283 1183065044791
Calculate the number d = u^(-1)*(f + s*c) mod r (d not equal to 0):
d=36208876963806837570114864736301118815518112373481661520544235236698182 7
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=29317945312948892388117580217799644292001864660377930167101443218874020
Calculate the number h1 = f*h mod r:
h1=77273564335795175852462176113728798470980140903651766641252756676841963 7
Calculate the elliptic curve point P = h1 G + h2 W Calculate the number h2 = c*h mod r:
h2=63105586560559524757786768225223571821054635629532262538599543350300915
(If P = (Px, Py) = (inf, inf) then the signature is invalid):
Px=82527945667413456894398314536185868994235306901932550287283 1183065044791
Py=11103720097354788368211490047913297084662046270419788310585 1475989597207
Convert the group element Px (x co-ordinates of point P on elliptic curve) to the number i:
i=82527945667413456894398314536185868994235306901932550287283 1183065044791
Calculate the number c' = i mod r:
c'=82527945667413456894398314536185868994235306901932550287283 1183065044791
If c' = c then the signature is correct; otherwise the signature is invalid:

8. PHECC (Polynomial based Hashing and Elliptic Curve Cryptography)

The main intension of this research is to explain a new hash algorithm using polynomials over finite fields. In software procedure, it runs at speeds comparable to SHA 3. The hash has many attractive characteristics in terms of its flexibility. Specifically, the length of the hash is a parameter that can be set at the outset. Additionally, the estimated degree of collision resistance is computed in terms of another parameter to compare whose value may be differed. The hash function that constructs has the following important attributes. Initially, the length of the output can be modified simply by changing a few steps of the calculation. Secondly, the computation performance is evaluated as a bit-stream procedure to oppose a block procedure. Finally, many features of the construction, which are the Current Register (CR) construction, the compression function and the exponentiation and truncation, appear to be novel constructs. The construction uses polynomials over finite fields. Note that earlier efforts have used polynomials over finite fields in the construction of hash algorithms. However, the make use of polynomials is very different.

This paper proposes a method to access and store the data through internet securely. ECC based encryption method is mainly used for storing the encrypted data in the cloud since the use of ECC notably reduces the message size, transmission overhead and the computation cost over RSA based PKI because the 160-bit key size in ECC gives similar security with the 1024-bit key in RSA. This scheme can ensure the privacy and security of the data in the cloud architecture using various schemes. After storing the encrypted data in the cloud, the integrity of that is checked using the PH (Polynomial based Hashing) algorithm. The integrity is checked by comparing the hash code, which is generated before and after storing data in the cloud. The overall steps of the proposed model are as follows,

A. The protection against unauthorized access by using the**PH (Polynomial based Hashing)** algorithm to verify that the service request is from an authorized user by comparing the hash of the password of the user by the stored hash in the database of the authentication server.
B. The use of **ECC based algorithm** for encrypting the data between the server and user will securely protect against Man in The Middle attack because the opponent cannot evaluate the key that is used for encryption or decryption.
C. The proposed model provides strong mutual authentication between and the authentication server and authentication user. The user challenges the authentication server in the authentication request message encrypting and identity by using secret-key computed by the user using ECC. Only the server can recompute the secret key and retrieve the user identity.

Authentication steps:

- The user calculates the hash value for the password using a highly secure **PH (Polynomial based Hashing)** algorithm. This hash value will be used after that as a reference when compared with the stored hash code in the server to verify user authentication attempts because this hash value produced from a one-way function so that it cannot be regenerated. It is compared only with the hash value, which will be produced from the server with the function.
- The user can assure that the message from an authorized party. The user then sends his hash value of the password encrypted with ECC to the server. The encryption of these parameters over the communicating channel will securely protect against alteration or modification from the unauthorized opponent.
- The server receives the parameters from the user and begins the checking process by verifying the ID of the user from the pre-stored database in the server. This assures the identity of the transmitter and that the message arrives from the authorized user.
- The server computes the hash value, and it compares with the received value from the authorized user. Because the creation of the hash value in the server from a specific function should be matched with the generated value in the server using the same function.
- The server checks the previous parameters (Checking the ID of the user and matches up to the received hash with the computed one) and if the check succeeded.
- The server will create hashing value using the PH algorithm. The server will transmit the created hash code encrypted with ECC to the user. The hash value is to be transmitted to the user, and that will assure the user that the message is originated from the server by comparing it with the generated one.
- The user will check the received message from the server by comparing the received value with the stored one. After the check succeeded, the user generates by multiplying the hash by the generating point of the curve and send it to the server.

- The server also generates by multiplying the hash with the same generating point of the curve and send to the user. After exchanging the hash value between the user and the server, mutual authentication between the server and the user is achieved.

D. After successful authentication, the data downloading and uploading process will be encrypted/decrypted using the EC algorithm. After reducing the data, the signature element is transmitted to authority EC to be digitally signed, and the message digest obtained by this process is also encrypted/decrypted using the same EC technique. The same process followed to decryption. The use of the PH algorithm as an authentication algorithm gives an efficient and scalable tool for authenticating the user with the server owing to its speed. The complex multiplication technique is utilized in the generation of the curve because of smaller space storage, which is stored in the field parameters, and that improves the computational cost and efficiency. Both the elliptic scalar multiplication operations and the efficiency of finite field computations are performed with the support of the proposed algorithm. Also, the known algorithms are used for these computations. The performance of ECC can be speed by selecting the appropriate finite field and elliptic curves.

To initiate three constructions may be of interest in their own right. The algorithm, then, can be described in brief as below.

- Initially, the CR construction is obtained as input a sequence of k polynomials over $\mathbb{F}_2$ of degree $< n$ and produces another such sequence.
- Secondly, the compression routine receives as input a binary sequence of length k, a sequence of k polynomials over $\mathbb{F}_2$ of degree $< n$, an integer r with $2^n < r \leq k$, and a sufficiently large integer λ, and generates r matrices of 2^nrows and $1 + \lambda$ columns. This construction is invertible. In other words, given the matrices, one can reconstruct both the binary sequence in addition to the sequence of polynomials. The compression function is obtained by eliminating columns 2 to $1+\lambda$ of selected rows of these matrices.
- The third construction is truncation, followed by exponentiation in a finite group. The hash function obtains a message specified as a binary string of length k and executes a preliminary operation on it to convert it into a sequence of k polynomials over $\mathbb{F}_2$ of degree $< n$. After that process, it invokes the CR construction and the compression routine, respectively.

In the Cantor enumeration, the entries of the compressed matrices are combined to generate a single integer. In the truncation-exponentiation routine, this integer is used to generate a hash value.

Enumeration:

Finally, a single integer generates from the vectors $R_i$using Cantor enumeration. For each $d \geq 1$, there is an adjective map

$$e_d : \mathbb{N}^d \rightarrow \mathbb{N}$$

For $d = 1$, we can take the identity, and for $d = 2$, we can take

$$e(x, y) = e_2(x, y) = \frac{(x + y)^2 + 3x + y}{2}$$

Given e_n, the map

$$(x_1, \ldots\ldots, x_{n+1})\ ) \rightarrow e_n\ (e_2(x_1, x_2), x_3\ , \ldots, x_{n+1})$$

It is a candidate for e_{n+1}. On the other hand, there are more optimal choices. To apply the enumeration function to the vectors $R_i$and need the functions $e_m$where $m = irr(n)$ for $4 \leq n \leq 10$. first record the values of the pairs (n,m) in Table 1. Explicit candidates that produce numbers

Table 1: Number of irreducible polynomials

N	4	5	6	7	8	9	10
M	7	10	16	25	43	71	129

of manageable size for $n = 4, 5, 6$ are given in the appendix. For example, for $n = 4$, we used

$$e_7(s_1, s_2, s_3, s_4, s_5, s_6, s_7) = e\left(e\big(e(e(s_1, s_2)s_3), e(s_4, s_5)\big), e(s_6, s_7)\right)$$

This number is the order $(k/10)^{16}$. Especially for a 1MB file, this is about 40 bytes.

Truncation-exponentiation:

The third of the "primitives" (the first two being the CR construction and the compression function) is known as truncation followed by exponentiation in a group and is explained as follows. Let

$$H: \{0,1\}^* \rightarrow \{0,1\}^k$$

A hash function with output length κ. Let G be a finite abelian group and select an element $g \in G$. Let

$$F: G \to \{0,1\}^T$$

be a function with $\tau < \kappa$. For a string $M \in \{0, 1\}^*$, consider the new function

$$H: M \to F\left(g^{int(H(M))}\right) \in \{0,1\}^T$$

Here, *int*is the function that associates to a bit string the integer that it represents in base 2.

ALGORITHM:

PARAMETERS: $e, n_1, \ldots, n_e, \{r_j, s_j, g_j, q_j\}, T$

INPUT: Message M of length k

OUTPUT: Hash value H of M of Tbits

- Compute the stretching and splitting $s(M, n_j)(1 \le j \le e)$
- Calculate the masking $CR_i^{(n_j)}$ for $1 \le j \le e$ and $1 \le i \le k$.
- Compute the tables $T_i^{(n_j)}$ for $1 \le j \le e$ and $1 \le i \le r_j$. Each table has 2^{nj} entries and each entry has s_j bits.
- Compute bit strings and their associated integers $BS_i^{(n_j)}$
- From the tables, compute the spectra $R_i^{(n_j)}$ and their associated integers $C(R_i^{(n_j)})$.
- To utilize both sets of integers to evaluate an integer I_j (for $1 \le j \le e$).
- Compute H_j in the group.
- The final hash value H is the sum of the H_j in group G.

Keys generation (Public Key /Private Key) in ECC:

To produce keys in order to discover one Base Point (G) from the group of Elliptic Curve Points. The Base Point can be produced as follows:

- Discover the large prime factor (k) for the total number of points (**#N**) in the elliptic curve group $E_p(b_4, b_6)$.
- Calculate kP where P is the elliptic curve point in group $E_p(b_4, b_6)$.
- If kP = o, then one can consider that P as a Base Point (G) for the key generation process.

Elliptic Curve Encryption/Decryption:

The first task encodes the plain text message (m) to be transmitted a (x, y) coordinate point P_m, and then it will be encrypted ciphertext and consequently decrypted. Simply encode the message as the x or y coordinates of a point because not all such coordinates are presented in the elliptic curve group $E_p(b_4, b_6)$. The ECC technique needs a few global parameters. An Elliptic group $E_p(b_4, b_6)$ and Base Point G are the global parameters for the encryption. To encrypt the message, the technique performs as follows:

To encrypt and transmit a message to B, A performs the following steps.

- k $\in_R$ N (choose k as a random positive prime number) R
- . Q <— [k] G
- P_k<— [k] P_B (P_B is receiver's public key)
- C_m<— {Q , $P_m + P_k$ } (C_m is the Cipher Text and sent as a Point)

To decrypt the message, the EC Decryption technique works as follows:

The Cipher Text C_m is to decrypt,and then B extracts the first coordinate *'O'*from the ciphertext, then multiply with its Private Key (b_B) and subtract the result from the second coordinate.

$$P_m + kP_B - b_B(Q) = P_m + kP_B - b_B(kG) = P_m + kP_B - k(b_B G) = P_m + kP_B - kP_B = P_m$$

In order to perform the above process, the various operations like multiplication of points, the addition of points, and the negative of a point. A has message m asked the message is encrypted as P_m by adding kP_B to m. Other than A recognizes the value of k, so even though P_B is a public key, no one can remove the mask kP_B. For an attacker to eliminate the message, the attacker has to compute k from the given group G and [k] G i.e. Q, and that is assumed very complex.

Example

Hashing message = Cloud computing security refers to the set of procedures, processes and standards designed to provide information security assurance in a cloud computing environment.

Let p=47, a=3 and b=6 are the characteristic parameters of the digital signature.
The length of the string is n = 166 characters long
Curve: $y^2 = x^3 + 3x + 6$
The coordinates of point P= (1.3|-3.47)
The coordinates of point Q= (-1.17|0.92)
Point R = P + Q = (3.03|6.56)
The private key = 1547465310.
The order of the point P=47.
The signature(r, s) = (44, 39)
Signature is agree
Computed the exponentiation in $GF(2^{160})^*$ using the generator
*b*957*ac*049*c*0*efdf*70*b*169150*cf*39*a*5763*f*19134*f,*
(Presented in hex form) and all the operations in the field were made modulo
*d*10000000000000000000000000000000000000008.
Let *M* be 50000 repetitions of the ten ASCII characters 0123456789. This is a message of length 0.5MB. We computed a 160-bit hash using 10 tables with 5 bit strings per table of length 256 bits each. The result is:
PHECC (*M*) = 67*df eb*5*a*9*ac*7*c*7*bd*03776*dd*3*abc*8963*eb*3*cbf*7*db*
Let *M* be as above except that we changed the last byte (in hexadecimal notation, from 39 to 38). We computed a 160 bit hash with the same parameters as above to get
PHECC (*M*) =*dc*48*d*7*cda*46*cc*9*dda*71424*fef*93*c*4201513187*ac*
Let *M* be 500*,* 000 repetitions of 0 and the same parameters as above. We get
PHECC (*M*) = *fb*879*ebd*36*a*72*ca*27*e*141129*bef*05*cdab*24*afed*5

9. Experimental Result

The implementations are completed using a system having an i5 Intel processor with 2 GB RAM capacity. The proposed method is computed using various parameters like Encryption time, Decryption time, Input data size and Throughput. The text files of different sizes are used to perform experiments in which a comparison of algorithms ECDH with SHA, ECC with SHA, ECDSA with SHA and proposed method are performed.

- Encryption time: The encryption time means that the Computation Time/Response Time as the time that an encryption technique acquires to generates a ciphertext from a plain text.
- Decryption time: The decryption time means that (Computation Time/ Response Time), which is considered the time that an encryption technique obtains to regenerate a plain text from a ciphertext.
- Input data size- Different algorithms needed different memory space to execute the operation. The memory space needed by any algorithm is determined based onthe number of rounds, input data size, and so on. The algorithm is considered best that utilizes small memory and perform the best task.
- Throughput-Throughput of the encryption technique is computed by separating the total plaintext in Megabytes (MB) encrypted on total encryption time for each algorithm.

The comparative analysis of the hybrid encryption algorithm is given below Table 2. In the table, the experiment results fromthe encryption file size comparison between ECC-SHA, ECDH-SHA, ECDSA-SHA and proposed encryption algorithm are clearly illustrated. The efficient performance is analyzed with these four hybrid algorithm comparisons for the cloud environment. From the tabular results shown below, observe that the experimental results obtained in the context of storage size parameters are better in the case of the proposed algorithm.

Table 1: Comparison of the proposed algorithm with other algorithms

FILE SIZE	ENCRYPTION TIME				DECRYPTION TIME				THROUGHPUT (%)			
	ECC-SHA	ECDH-SHA	ECDSA-SHA	PH-ECC	ECC-SHA	ECDH-SHA	ECDSA-SHA	PH-ECC	ECC-SHA	ECDH-SHA	ECDSA-SHA	PH-ECC
20KB	5146	4231	3015	1247	8861	6194	4635	2579	90	80	70	60
40KB	10907	8652	5612	2807	14307	10536	8691	4531	85	70	65	50
60KB	12342	9954	6143	4001	18019	12369	9475	5526	65	55	45	40
80KB	14562	11389	7165	5254	22591	16548	10341	6448	40	30	30	25
100KB	16432	12473	8013	6025	26118	19346	11475	7346	35	25	25	20

Authentication Time and Security level

The performance evaluation of authentication time and security level using ECC-SHA, ECDH-SHA, ECDSA-SHA and proposed encryption algorithm is shown in Figure 1. The new proposed authentication algorithm combines the advantages of a small number of multiplications and a minimal number of variables using Elliptic Curve Polynomial operations. As a result, the proposed polynomial based hashing algorithm performs better than other algorithms compared by accuracy and speed rate.

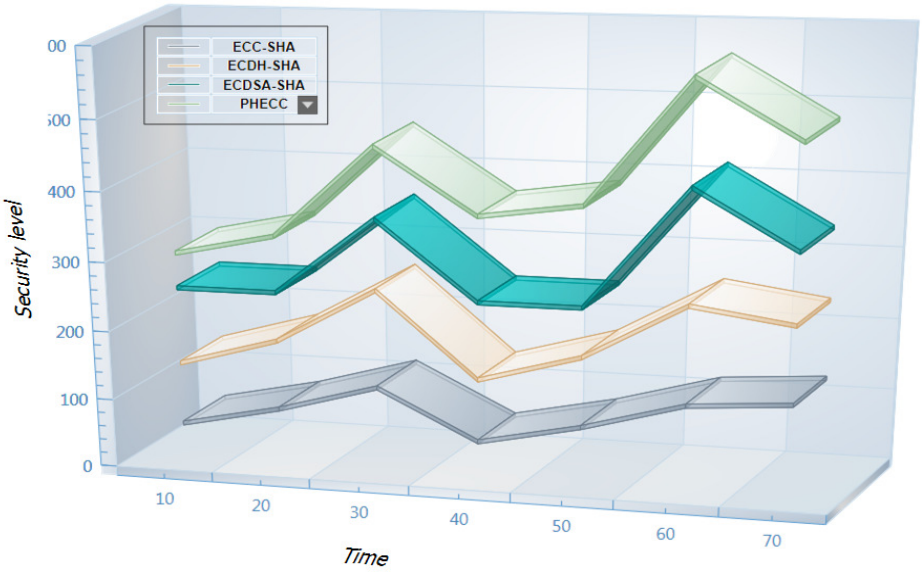


Figure 1: comparison of the proposed algorithm with others by time and security level

Encryption / Decryption Time

Encryption and decryption time was based on the processor speed performance and also the complexity of the technique etc. From the result, the proposed algorithm PHECC gives a much faster encryption/decryption process as compared to other techniques. In ECC-SHA, the time of encryption/decryption process, ECDH-SHA, and ECDSA-SHA is growing exponentially based on their performance. PHECC encryption time differs linearly depending on the input key size. Also, the decryption time remains in the exponential increase. Using the below graph to conclude that the proposed algorithm provides better results based on encryption and decryption time compared by other hybrid algorithms.

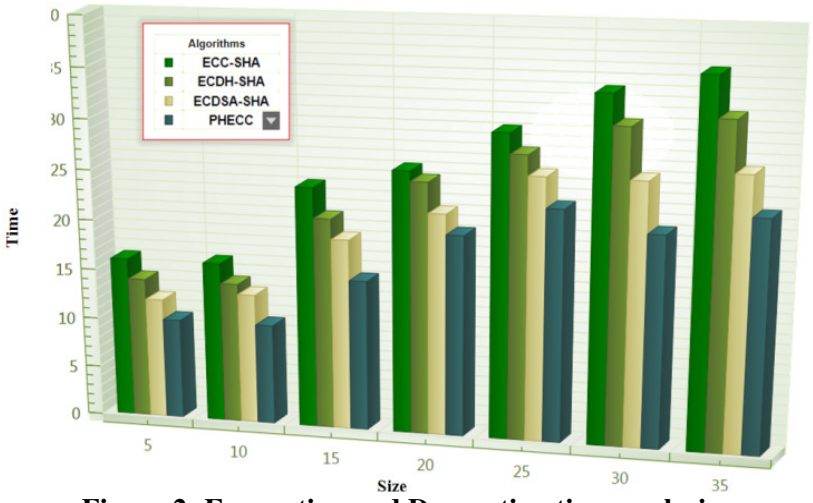


Figure 2: Encryption and Decryption time analysis

Throughput

The Throughput of the algorithm evaluates by dividing the total data in bytes by encryption time. The higher the Throughput higher is the efficiency of the system. Figure 2 given below provides us the comparison between the ECC-SHA, ECDH-SHA, ECDSA-SHA, and proposed algorithm using Throughput. It is essential to understand the size of the input and the size of output, as this is one of the important properties of an avalanche effect in any cryptographic algorithm.Figure 1 demonstrates the Throughput (in operations/second); the corresponding precise measurements are given in figure 1. The polynomial based hashing and ECC algorithm vary from others. The proposed algorithm using polynomial based simple and effective multiplication operation in the elliptic curve. In figure 3 theThroughputof the proposed algorithm is higher than others. The below graph shows that the proposed algorithm outperforms than others.

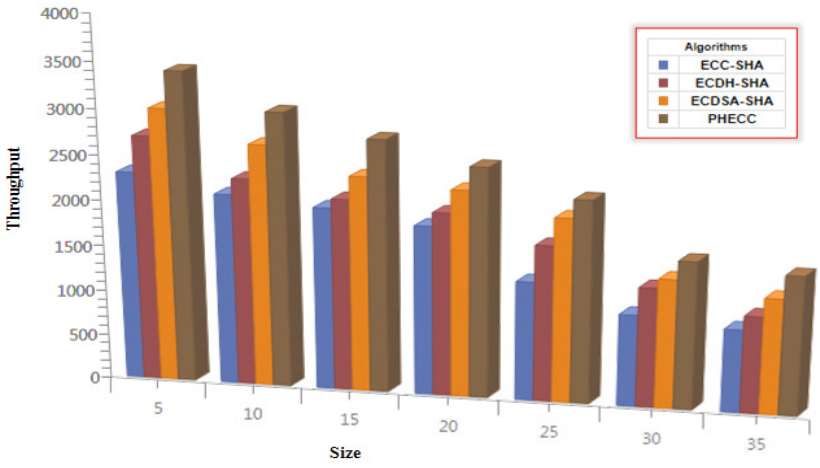


Figure 3: Throughput

X. Conclusion

In this research, a new technique is implemented for ensuring the data security of the files being uploaded to the cloud by clients. This security is achieved through a technique of encryption using PH and ECC method. For authentication purposes, use an efficient polynomial evaluation procedure based on algorithms PH that leads to a fast MAC. To examine how fast the evaluation can be completed using the software. In terms of speed, our procedure can be compared with other algorithms. By using polynomial evaluation, it is possible to reduce the key to a size of about as large as that of the tag. To compare the hybrid algorithms work PHECC with other hybrid algorithms using different parameters. The experimental results of the proposed technique show that the size of the encrypted file is decreased by approximately 6% as compared to that of the existing technique, resulting in the lesser storage space consumption at the cloud. Therefore, the storage space of the cloud is used in a much efficient manner when the proposed technique is implemented. Moreover, the time is

taken by the proposed technique of PH with ECC in the encryption process and decryption process of the text file is lesser than that of the pre-existing ECC-SHA based techniques.

References

[1] Pritesh Jain, Prof. VaishaliChourey and Prof. DheerajRane," An Analysis of Cloud Model-Based Security for Computing Secure Cloud Bursting and Aggregation in Real Environment" published in International Journal of Advanced Computer Research (IJACR), ISSN (Print): 2249-7277, ISSN (Online): 2277-7970 Volume 1,**(2011),** pp. 23-28.

[2] S. Abdul, H. M. Abdul Kader and M. M. Hadhoud"Performance Evaluation of Symmetric Encryption Algorithms" published in Journal Communications of the IBIMA, ISSN: 1943-7765, Volume 8,**(2009)** pp. 58-64.

[3] Parikshit Prasad, BadrinathOjha, Rajeev RanjanShahi, RatanLal"3-Dimensional Security in Cloud Computing" published in IEEE Xplore, 978-1-61284-840-2/11/$26.00 ©2011, **(2011)** ,pp. 198-201.

[4] VadymMukhin, ArtemVolokyta, "Security Risk Analysis for Cloud Computing Systems" The 6th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, Prague, Czech Republic, **(2011),** pp.15-17.

[5] Ashutosh Kumar Dubey, Animesh Kumar Dubey, MayankNamdev and Shiv Shakti Shrivastava, "Cloud-User Security Based on RSA and MD5 Algorithm for Resource Attestation and Sharing in Java Environment" published in CSI Sixth International Conference on Software Engineering (CONSEG), Indore, MP (INDIA), IEEE Xplore, ISBN: 978-1-4673-2174-7, **(2012)**, pp. 1 – 8.

[6] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In Joe Kilian, . Springer,**(2001).**

[7] Dan Boneh, Ben Lynn, and HovavShacham. Short signatures from the weil pairing. J. Cryptology, **(2004).**

[8] Colin Boyd, Paul Montague, and KhanhQuoc Nguyen. Elliptic curve based password authenticated key exchange protocols. In Vijay Varadharajan and Yi Mu, editors, ACISP, volume 2119 of Lecture Notes in Computer Science, Springer, **(2001).**

[9] Victor Boyko, Philip D. MacKenzie, and Sarvar Patel. Provably secure password-authenticated key exchange using di_e-hellman. In EUROCRYPT, **(2000).**

[10] K. Gaj, et al., Comprehensive Evaluation of High-Speed and Medium-Speed Implementations of Five SHA-3 Finalists Using Xilinx and Altera FPGAs, George Mason University

[11] M. Knutsen, K. A. Martinsen, Java Implementation and Performance Analysis of 14 SHA-3 Hash Functions on a Constrained Device, Norwegian University of Science and Technology, Department of Telematics, **(2010).**

[12] Archana Singh Parmar, Monika Sharma," *Improving Data Storage Security in Cloud Computing using Elliptic Curve*" International Journal of Engineering Science and Computing(Vol -7,Issue- 4), **(2017).**

[13] B. Glas, S. Sander, V. VitaliStuckert, D. Muller-Glaser, J. Becker, "Prime Field ECDSA Signature Processing for Reconfigurable Embedded Systems," International Journal of Reconfigurable Computing **(2011).**

[14] Navneet Randhawa, and Lolita Singh, "A Systematic Way to Provide Security for Digital Signature Using Elliptic Curve Cryptography" IJCST Vol. 2, Issue 3, (**2011**).

[15] Kun-Lin Tsai, Fang-YieLeu, Tien-Han Wu, Shin-shiuanChiou, Yu-Wei Liu, and Han-Yun Liu, "A Secure ECC-based Electronic Medical Record System", Journal of Internet Services and Information Security (JISIS), Volume 4, Issue 1, **(2014).**

Authors

Mr.S.Hendry Leo Kanickamworking as an Assistant Professor in Department of Information Technology ,St.Joseph's College (autonomous) Trichy,Tamilnadu, India. He received his M.Phil Degree in Bharathidasan University in 2008 and also He is pursuing Ph.D (Computer Science) in Bharathidasan University.

Dr. L. Jayasimman working as an Assistant Professor in Department of Computer Applications, Bishop Heber College Trichy, Tamilnadu, India. He received his M.Tech Degree in Bharathidasan University, Trichy, India in 2008 and completed his PhD (Computer Science) in Bharathidasan University in 2014.