# AN IMPROVED DATA PROTECTION TECHNIQUE TO SECURE A PUBLIC CLOUD ENVIRONMENT

S.Hendry Leo Kanickam*
Research Scholar, Department of Computer Science, Bishop Heber College (Autonomous),
Affiliated to Bharathidasan University, Trichy, Tamilnadu, India

Dr.L.Jayasimman
Assistant Professor, Department of Computer Applications, Bishop Heber College (Autonomous),
Affiliated to Bharathidasan University, Trichy, Tamilnadu, India

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

**Keywords:** Elliptic Curve Cryptography; Elliptic Curve Digital Signature; Elliptic-Curve Diffie–Hellman; Polynomial Based Hashing ; 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 by IT professionals to have 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, have emerged 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 disaster management and risk mitigation policies to protect security and privacy. The data transmission process is protected on the internet by the encryption process, which could protect the data. The encryption process converts data using any encryption technique via utilizing key in mixed form. Only the user has the encryption key to obtain the 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 secure authentication. Many hybrid algorithms are developed for security intention in a cloud computing environment like ECC-Md5, Blowfish-MD5, AES-MD5, ECC-SHA. These older techniques are not helpful last year due to their low-security level, and the MAC has been broken effortlessly by hackers [4, 5]. Here, it introduces a new security technique using a hybrid cryptosystem for data security in the cloud environment. The hybrid algorithm is mainly used to implement 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 of the security mechanism. The proposed technique, performance is comparable with other hybrid algorithms, such as ECC-SHA, ECDH-SHA, and ECDSA-SHA Major Headings

## 2. Related Works

To secure the data [8], prevents hackers or unauthorized access in a cloud environment at data transmission by encoding user data. There's a lot of concern now about how to build a better hash algorithm. The main complexity stems from the truth that a hash function’s design principles are not completely understood. Some of the techniques [9] for constructing new hash functions from old functions have already been addressed in the literature survey. Several attempts have been made to consider better design principles for hash functions; however, this work still seems to be ongoing. The purpose of this article is to explain a new hash algorithm that integrates some of the aspects described above. In particular, the output length of the function is a parameter that can be set at the beginning.

Additionally, the level of resistance to collision is based on another parameter 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. The MAC (Message authentication codes) is used with the polynomial evaluation method that generates a tiny key, even for very large messages. An efficient procedure evaluates using polynomials over a finite field to be utilized to construct 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 of the security mechanism. Hence, it presents a hybrid algorithm for enhanced network security. The integrity is ensured by transmitting data, the data are mainly subjected to a new proposed polynomial based Hashing and Elliptic Curve Cryptography (PHECC) hash algorithm. The proposed algorithm uses PH (Polynomial Hashing) to monitor the integrity of the data, collects data, encrypts data, decrypts and installs all of these data into a cloud. The data download and upload 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. Proposed Methodology

### 3.1 How to Hash into Elliptic Curves

Some of the elliptic curve cryptosystems re quire hashing into an elliptic curve, for example, other cryptography techniques. A new super singular elliptical curve (elliptic curve scheme) is used in which a one-to-one mapping f occurs from the base field Fp 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 [6] Shallue et al. Released the first algorithm mapping Fpn into an elliptic curve in deterministic polynomial time. In deterministic polynomial time, the hybrid algorithms provide any elliptic curve E specified over Fpn, maps components of Fpn into E, when pn = two mod 3. The algorithms rely on a rational function, an explicit function from Fpn to E, which can be implemented over Fpn in O (log3q) time and a constant number of operations. In addition, this technique relies on the computation of a cube root and is simpler than the algorithm of Shallue and Woestijne[7].

An algorithm shows how to use two separate constructions to hash efficiently and deterministically into an elliptic curve. If the underlying hash function is one-way [8, 9], the first construction is a one-way function. Additionally, even if the underlying hash function is collision-resistant, the second construction achieves collision resistance. Explain two constructions of hash functions in E, given a function f in an elliptic curve E. Define L as f-1P's maximum size where P is any point on E:

$$L = \underset{P \in E}{\text{Max}}(|f^{-1}(P)| \quad (1)$$

Have L 4 for our encoding feature fa, b. Note that if operating with cofactor r in subgroup E of order n, use function fa, b'=r.fa, b encoding. If r is relative prime to n, then L 4r must be present. The first development is as follows: given a hash function h:

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

$$H(m) = f(h(m)) \quad (2)$$

Into the curve Ea, b(Fp) as a hash function. Show that H is one-direction if h is one direction, in the following. The second construction, therefore, is given as follows. Consider the following family of functions: a securityparameter k and an integer q = pn with q 2k,

$$v_{c,d} : \{0,1\}^M \rightarrow F_q \; x \rightarrow c.x + d \quad (3)$$

$$c, d \in F_q$$

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. Combine the encoding function f with the functions in the vc, d family with an elliptic curve E to get a collision-free family G:

$$G = (f \bullet v_{c,d})_{c,d \in F_q} : \{0,1\}^M \rightarrow E(F_q)\; x \rightarrow f(c.x + d) \quad (4)$$

Given the family H of collision-resistant functions h:{0,1}*{0,1}k, the following hash function family is built into the curve as a final stage *E*:

$$H_E = (f \bullet v_{c,d})_{c,d \in F_{qk \in H}} : \{0,1\}^M \rightarrow E(F_q)\; x \rightarrow f(c.h(m) + d) \quad (5)$$

### 3.2 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 SHA-1 without sharing its weaknesses. A set of six hash functions is considered 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 recent hash functions that use various additive constants and shift amounts and are nearly identical in their architectures, differing only in the number of rounds. The truncated versions of SHA-256 and SHA-512 are called SHA-224 and SHA-384. The truncated versions of SHA-512/256 and SHA-512/224 are SHA-512/256 and SHA-512/224.. At this point, a step summary of SHA-512 is processing as given below [11]:

**Steps Overview:**

- A version of SHA-256 is SHA-512 that controls eight 64-bit words. Initially, the message to be hashed.
- Padded in such a way by its length in such that way the consequence is a multiplicity of a multiple of 1024bits long, and then the next step,
- Parsed into 1024-bit blocks of message $M^{(1)}$, $M^{(2)}$, ….., $M^{(N)}$.
- The blocks of message 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 a number of procedures, processes and standards to ensure the safety of information 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 is #0.00% of the bits differ (0 of 512). Longest identical bit sequence: offset 0, length 512.

### 3.3. 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 Elliptical curves are generalized to other established cryptosystems. The ECC protection pattern is very critical and does not impact side-channel attacks. For encryption, different key lengths are used to provide adequate coverage over the data blocks. In two variables, f(x, y)=0 with a rational point, an elliptical curve is considered to be a unique cubic curve above the K field (a point at infinity stage). K is typically the complex numbers, rational, and algebraic extensions of rational, p-adic numbers, or a finite field [12]. The primary generation is that both public and private key was created. The sender encrypts the message with the public key of the recipient, and the recipient decrypts the private key. In order to pick the number' handle within a 'n' range, construct the public key d = the random number using the following equation, and choose from (1 to n-1). P's the curve stage. Q 'is the public key, and the private key.’

Encryption: Let'm' be the message is transmitted to represent this message on the curve. These have in-depth implementation details. Consider' has a point'M' on the curve 'E.' Randomly select 'k' from [1 – (n-1)]. Two cipher texts will be generated. It is C1 and C2.

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

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

M=c2-d*c1 (3)

M is the original message 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)

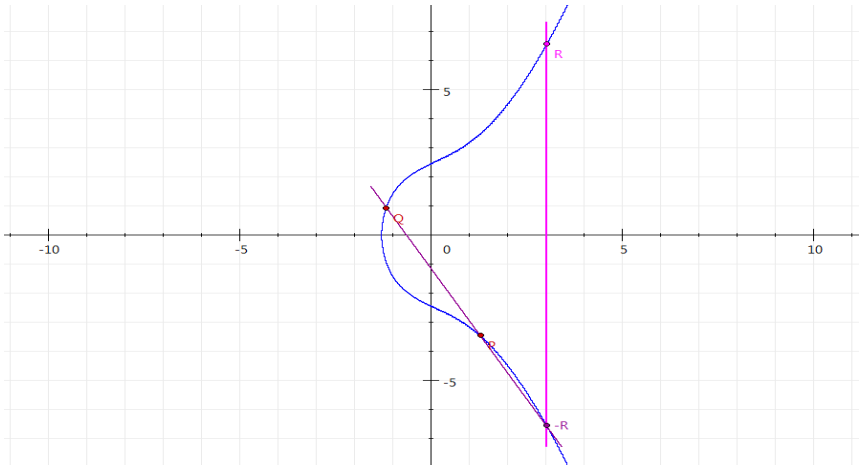


Figure-1 ECC Example Curve

2. Type: F(p), Curve attributes: ANSI X9.62,Curve: prime192v1, radix: 16 hexadecimal, attributes: $y^2 = x^3 + 3x + 6$, and attractions: $y^2 = x^3 + 6$., 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 Type: F(2^m), Pick Curve Attributes: ANSI X9.62, Curve: c2pnb163v1, Radix: 16 hexadecimal attributes

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. Scale of the curve: Small, form of curve: F (2^m), Attributes of curve: m=4. f = x^4+x+1. a=1,b=1. Curve: $y^2 + xy = x^3 + 2 + 1$. P = (g9), Q = (g6), R = P + Q = (g5);

### 3.4. ECDH

Elliptic-curve Diffie–Hellman (ECDH) is considered as an anonymous key agreement (In cryptography, a key-agreement protocol is a procedure which has 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 elliptic curve public-private key pair, a shared secret (A shared secret is a key or data which can only recognized 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 cipher text 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. There are two inputs; it is upfront to compute 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 disclosing it to others, either the other party or an eavesdropper.

Algorithm

For instance, two persons, A and B, 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 A and B. The private key represents dA, and the public key denotes HA = dA G for A and the keys dB and HB = dB G for B. Note that both A and B use the same domain parameters: the same base point G on the same elliptic curve on the same finite field.
- A and B 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.
- A computes S = dA HB (using own private key and Benny's public key), and B evaluates S = dB HA (using own private key and A's public key). Note that S is the same for both A and B, iS = dA HB = dA (dB G) = dB (dAG) = dB HA

**ECDH EXAMPLE:**

Step 1: Set criteria for the public

Form of curve: F(p), Size of curve: Large, Domain parameters: a=3,b=6,p=47, generator G=(15,18)

Step 2: Pick Secrets

Alex= 5

Benny=6

Step 3: Create Common keys

Hidden key (d): Q=d*G,

Alex= (44,8)

Benny= (15,29)

Step 4: Shared keys Exchange

Step 5:Common key generation

Key = sA*QB and key=sB*QA

S= (44, 39)

**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 somewhat different type of signature checks, which Signature authentication more easily [14]. The primary 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 related to DSA, other than the key generation, depend upon the ECC algorithm [15].

Key Pair Generation: ECDSA relies on the domain parameters to construct key pairs. Provided the elliptical curve E over Zp with many divisible points,

1. Choose a P(xp, yp) and a random integer (d[1, n1]) of the curve.
2. Make sure the point Q is also on a curve to determine Q(xq, yq) = dP.
3. The private key is d. The public key is (E, P, n, Q).

Generation of Signature: The input message m and the private key d to be digitally signed

1. Select an integer random k [1, n 1].
2. Compute (x1 , y1) = kP; transform x1 and r = x1 mod n. (Even though you go back to stage 1,

r = zero)

3. Assess s = k1 (SHA512(m)+d). (When s = 0, go back to stage 1)

Signature Generation: The digitally signed message m and the private key d,

1. Select an integer random k [1, n 1].
2. Compute (x1 , y1) = kP; transform x1 and r = x1 mod n. (Even though you go back to stage 1,

r = zero)

3. Assess s = k1 (SHA512(m)+d). (When s = 0, go back to stage 1)
4. Signing is pair (r, s).a

Signature Verification: Due to the pair of signatures (r, s) on m, public keys (E, P, n, Q)

1. See if the r, s [1, n 1] numbers are used.
2. Then determine w = s1 mod n.
3. U1 = RW mod n, U2 = RW mod n. Assessment
4. Compute = u1P + u2Q, transform x0 and v = x0 mod n.
5. Compare v and r, and only accept the signature if v = r.

**ECDSA EXAMPLE:**

Signature originator: Hendri Hendri

EC-prime239v1' domain parameters for use:

Selected algorithm for signature: ECSP-DSA with SHA hash function.

Text size M for signature: 800 bytes

Bit length of d = 477 bits c + length

File Name = instance.txt

Message = Security in the cloud refers to a collection of protocols, processes and standards designed to provide cloud computing information security.

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

Elliptical curve E defined by curve: y^2 = x^3 + ax + b(mod(p)):

a=883423532389192164791648750360308885314476597252960362792450860609699836

b=738525217406992417348596088038781724164860971797098971891240423363193866

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=825279456674134568943983145361858689942353069019325502872831183065044791

Vy=111037200973547883682114900479132970846620462704197883105851475989597207

Use the selected hash feature to measure the hash value 'f of message M.

f=10168359407591437558955011081399370465124679006 99

The following is the ECDSA SIGNATURE:

G has r and k cofactor (r*k is the number of points of E) in the first order:

= 1 k

G-point of E (described by its co-ordinates (x, y))

Gx=110282003749548856476348533541186204577905061504881242240149511594420911

Gy=869078407435509378747351873793058868500210384946040694651368759217025454

r=883423532389192164791648750360308884807550341691627752275345424702807307

The hidden key s is to overcome the W = x*G(x unknown) EC discrete log problem

S= 216287993557651304296725434592457107505247085725787393465198777094360670

**Signature**: Convert the group element Vx (x coordinates of point V on elliptic curve) to the number i:

i=825279456674134568943983145361858689942353069019325502872831183065044791

Calculate the number c = i mod r (c not equal to 0):

c=825279456674134568943983145361858689942353069019325502872831183065044791

Calculate the number d = u^(-1)*(f + s*c) mod r (d not equal to 0):

d=362088769638068375701148647363011188155181123734816615205442352366981827

The following ECDSE VERIFICATION:

When c or d does not fall under the [1, r-1] interval, the signature is invalid: c and d fall under the necessary [1, r-1] interval. The h number = d^(-1) mod r can be calculated::

h=293179453129488923881175802177996442920018646603779301671014432188740 20

Calculate the number h1 = f*h mod r:

h1=772735643357951758524621761137287984709801409036517666412527566768419637

Calculate the elliptic curve point P = h1 G + h2 W The number h1 = f*h mod r can be determined.

h2=631055865605595247577867682252235718210546356295322625385995433503009 15

(If the signature is invalid, then P = (Px, Py) = (inf, inf)

Px=825279456674134568943983145361858689942353069019325502872831183065044791

Py=111037200973547883682114900479132970846620462704197883105851475989597207

Convert the Px (x point P coordinates on the elliptical curve) element group to the i

i=825279456674134568943983145361858689942353069019325502872831183065044791

The number c '= I mod r is calculated:

=825279456674134568943983145361858689942353069019325502872831183065044791

If c '= c, the signature is correct; the signature is invalid if not:

### 3.5 PHECC (Polynomial based Hashing and Elliptic Curve Cryptography)

The research work's scope is to explain a new hash algorithm using polynomials over finite fields. In software procedure, it operates at velocities close to SHA 3. The hash has many attractive characteristics in terms of its flexibility. Specifically, the hash length is a parameter to be calculated at the beginning. Additionally, the degree of collision resistance is estimated using another parameter to compare whose value may be deferred. The hash function that constructs has the following important attributes. Initially, the output length can be modified only adjusted a few calculation steps. Second, the estimate performance is evaluated as a bit-stream procedure to oppose a block procedure. Finally, many construction features, 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 Hash algorithms in construction. But it is, the make use of polynomials is very different.

This paper proposes a method to access and store the data through the internet securely. ECC-based encryption has been used primarily to store encrypted information 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 proposed scheme will ensure data privacy and protection across different schemes in the cloud architecture. The integrity of this is verified by the PH algorithm after storage of encrypted data in the cloud. The integrity of the hash code is verified by comparing data before and after storage in the cloud, the integrity of that is checked using the PH (Polynomial based Hashing) algorithm. The integrity is checked by matching the hash code generated before and after storing data in the cloud. The total 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. ECC-based algorithms 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 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 recomputed 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 is used after that as a reference associated with the stored hash code in the server to prove user authentication attempts because this hash value produced from a one-way function cannot be regenerated. It is compared only with the hash value produced from the server.
- The user assures that the message from an authorized party. The user now sends his hash value of the password encrypted with ECC to the server. The encrypted parameters over the communicating channel securely protect against revision or modification from the unauthorized opponent.
- The server receives the user's parameters and begins the checking process by verifying the user's ID from the server's pre-stored database. 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 user's ID 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 transmitted to the user, which will assure the user that the message originated from the server by comparing it with the generated one.
- The user will check the server's received message by comparing the received value with the stored one. After the check succeeded, the user generates by multiplying the hash by generating the curve's point and sending 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.
- 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 the 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. Using the PH algorithm as an authentication algorithm gives an efficient and scalable tool for authenticating the user with the server due to its speed. The complex multiplication technique is utilized in the curve generation because of smaller space storage, which is stored in the field parameters, improving the computational cost and efficiency. The elliptic scalar multiplication operations and Finite field calculation efficiency is performed using the algorithm proposed support. Also, the known algorithms are used for such calculations. By selecting the related finite field and elliptical curves the ECC output can be speeded
- Three constructions may be of interest to themselves. Therefore, the algorithm can be briefly mentioned below.

- In the beginning, a sequence of k polynomials above F2 of degree < n is obtained from CR building and generated a different sequence.
- Secondly, a binary sequence of k polynomials above the F 2 degree < n is obtained as input in the compression routine., an integer r with $2n < r \leq k$, and a sufficiently large integer λ, and generates r matrices of 2n rows and 1 + λ 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+λ 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 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, with Cantor enumeration, a single integer is rendered by the Ri vectors. A map of adjectives exists for each d 1

Ed: Ed: Nd N

We can consider identity for d = 1, and for d = 2, we can take identity.

ex, y / or e2X, y / or X+Y2 + 3x+y2

Due to en, map

(x1.xn))(e2x1,x2.x3),xn+1) (x3).

He's an en+1 nominee. More optimal options occur on the other side. For the application of the enumeration function to Ri vectors and for 4 n 10 you have to use the em functions m = irr(n). First, in Table 1, record pair values (n, m) Explicit applicants generating numbers.

Table 1. Number of irreducible Polynomials.

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

Appendix is supported with the manageable size n = 4 , 5, 6. For instance, we used for n = 4

$$e_7(s_1, s_2, s_3, s_4, s_5, s_6, s_7) = e(e(e(s_1, s_2)s_3, e(s_4, s_5), e(s_{6,} s_7)$$

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

An output length hash function. Let G be a final group of abelians and pick G .

Let $H: G \rightarrow \{0,1\}^T$ be a function with τ < κ. For a string M ∈ {0, 1}*,

Consider the new function

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

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

ALGORITHM:

PARAMETERS: e, n1, ne, {rj, sj, gj, qj,}, T

INPUT: Message M with a length of k

OUTPUT: Hash value H of bits M of T Compute the stretching and splitting

Calculate the masking For $1 \leq j \leq e$ and $1 \leq i \leq k$.

1. Compute the tables for $1 \leq j \leq e$ and $1 \leq i \leq$ . Each table has entries, and there are SJ bits in each entry.
2. Compute bit strings and related BSinjj integer strings
3. Computing the spectrum from the tables From the tables, compute the spectra and their associated integers .To utilize both sets of integers to evaluate an integer (for $1 \leq j \leq e$).
4. Compute in the group.

5. The final hash value H is the sum of the  in group G.
6. Keys generation (Public Key /Private Key) in ECC:
7. To produce keys to discover one Base Point (G) from the group of Elliptic Curve Points. The Base Point can be produced as follows:
8. Discover the large prime factor (k) for the total number of points (#N) in the elliptic curve group Ep (b4, b6).
9. Calculate kP where P is the elliptic curve point in group Ep (b4, b6).
10. 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 Pm, and then it will be encrypted cipher text 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 Ep (b4, b6). The ECC technique needs a few global parameters. An Elliptic group Ep (b4, b6) and Base Point G are the global encryption parameters. To encrypt the message, the technique performs as follows:

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

- k  N (choose k as a random positive prime number) R
- . Q <— [k] G
- Pk <— [k] PB (PB is receiver's public key)
- Cm<— {Q, Pm + Pk } (Cm is the Cipher Text and sent as a Point)

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

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

Pm + kPB - bB (Q) = Pm + kPB - bB (kG) = Pm + kPB – k (bB G) = Pm + kPB – kPB = Pm

The various operations like multiplication of points, the addition of points, and the negative of a point perform the above process. A has message masked the message is encrypted as Pm by adding kPB to m. Other than A recognizes the value of k, so even though PB is a public key, no one can remove the mask kPB. For an attacker to eliminate the message, the attacker has to compute k from the given group G and [k] G, i.e., Q, which is assumed very complexly.

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 (2160)* using the generator

b957ac049c0efdf70b169150cf39a5763f19134f,

(Presented in hex form) and all the operations in the field were made modulo

d10000000000000000000000000000000000000008.

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) = 67df eb5a9ac7c7bd03776dd3abc8963eb3cbf7db

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) =dc48d7cda46cc9dda71424fef93c4201513187ac

Let M be 500, 000 repetitions of 0 and the same parameters as above. We get

PHECC (M) = fb879ebd36a72ca27e141129bef05cdab24afed5.

## 4. 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 is performed.

- Encryption time: The encryption time means that the Computation Time/Response Time is when an encryption technique acquires to generate a cipher text 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 cipher text.
- Input data size- Different algorithms needed different memory space to execute the operation. The memory space needed by any algorithm is determined based on the number of rounds, input data size. The algorithm is considered as the best that utilizes small memory and performs the best task.
- Throughput-Throughput of the encryption technique is computed by separating the total plaintext in Megabytes (MB) encrypted at each algorithm's total encryption time.

The comparative analysis of the hybrid encryption algorithm is given below Table 2. In the table, the experiment results of the 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. The tabular results shown below show that the experimental results obtained in the context of storage size parameters are better at the proposed technique

The comparative analysis of the hybrid encryption algorithm is given below Table 2. In the table, the experiment results of the 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. The tabular results shown below show that the experimental results obtained in the context of storage size parameters are better at the proposed technique.

Table 1: Proposed and Existing technique compared based on Time

| 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 | 30 |

**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 with accuracy and speed rate.

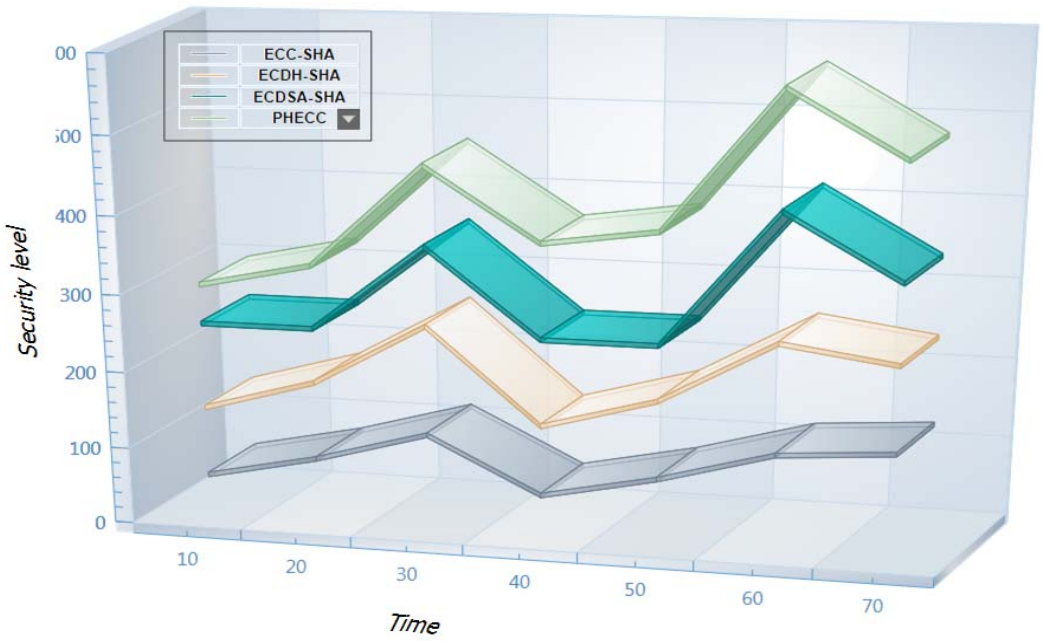


Figure 2: comparison of the proposed algorithm with others by time and security level

**Encryption / Decryption Time**

Encryption and decryption time was based on the processor speed performance and the complexity of the technique. 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 than other hybrid algorithms.

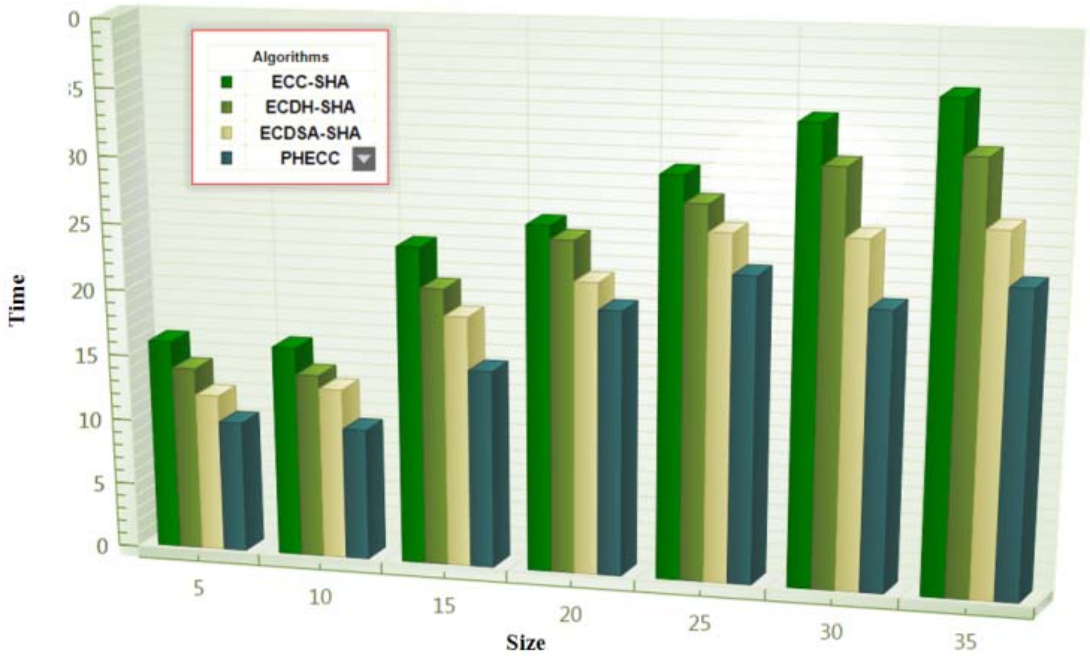


Figure 3: Encryption and Decryption time analysis

**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 below shows 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, the Throughput of the proposed algorithm is higher than the others. The below graph shows that the proposed algorithm outperforms than other methods.

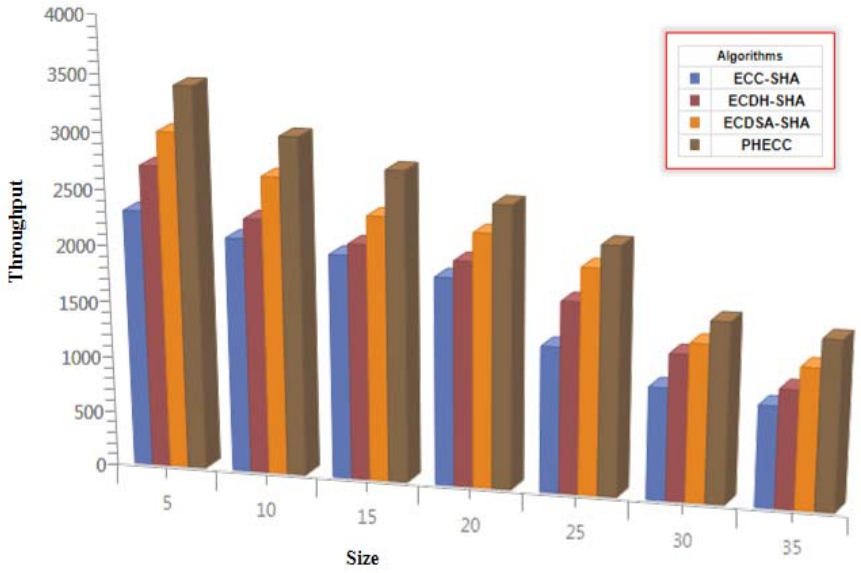


Figure 4: Throughput

## 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 accomplished through a technique of encryption using PH and ECC method. For authentication purposes, use an efficient polynomial evaluation procedure based on algorithms that lead 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. Using polynomial evaluation, it is possible to reduce the key to a size of about as large as the tag. To compare the hybrid algorithms work PHECC with other hybrid algorithms using different parameters. The proposed technique's experimental results show that the encrypted file's size is decreased by approximately 6% compared to that of the existing technique, resulting in the lesser storage space consumption at the cloud. Therefore, the cloud's storage space is used efficiently when the proposed technique is implemented. Moreover, the time is taken by the proposed technique of PH with ECC in the encryption process, and the decryption process of the text file is lesser than that of the pre-existing ECC-SHA based techniques.

## References

[1] Pritesh Jain, Prof. Vaishali Chourey and Prof. Dheeraj Rane (2011): An Analysis of Cloud Model-Based Security for Computing Secure Cloud Bursting and Aggregation in Real Environment" International Journal of Advanced Computer Research, Volume 1, (2011), pp. 23-28.

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

[3] P.Prasad, B.Ojha, R.R Shahi, R.Lal, A.Vaishu, and U.Goel,(20111):3-Dimensional Security in Cloud Computing, 3rd International Conference on Computer Research and Development, Sanghai, , pp.198-201.

[4] Vadym Mukhin, Artem Volokyta (2011): 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,, pp.15-17.

[5] Ashutosh Kumar Dubey, Animesh Kumar Dubey, Mayank Namdev and Shiv Shakti Shrivastava (2012): 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, pp. 1 – 8.
[6] D. Boneh and M. Franklin (2001): Identity-Based Encryption from the Weil Pairing, Advances in Cryptology — CRYPTO 2001, pp. 213-229, In Joe Kilian, Springer.
[7] Dan Boneh, Ben Lynn, and Hovav Shacham(2004) Short signatures from the Weil pairing. J. Cryptology .
[8] D. Boneh and M. Franklin(2001): "Identity-Based Encryption from the Weil Pairing", Advances in Cryptology — CRYPTO, pp. 213-229.
[9] Victor Boyko, Philip D. MacKenzie, and Sarvar Patel(2000): Provably secure password-authenticated key exchange using diffie-hellman, International Association for Cryptographic Research,pp. 157-172.
[10] Gaj, K., Homsirikamol, E., Rogawski, M., Shahid, R., Sharif, M.U(2012): Comprehensive evaluation of high-speed and medium speed implementations of five SHA-3 finalists using Xilinx and Altera FPGAs, 3rd SHA-3 Candidate Conference, Washington, D.C., 22–23..
[11] M. Knutsen, K. A. Martinsen (2010): Java Implementation and Performance Analysis of 14 SHA-3 Hash Functions on a Constrained Device, Norwegian University of Science and Technology, Department of Telematics, pp.1-155.
[12] Archana Singh Parmar, Monika Sharma (2017): Improving Data Storage Security in Cloud Computing using Elliptic Curve"International Journal of Engineering Science and Computing Vol -7, Issue- 4 ,pp.100-104.
[13] B. Glas, S. Sander, V. Vitali Stuckert, D. Muller-Glaser, J. Becker(2011): Prime Field ECDSA Signature Processing for Reconfigurable Embedded Systems,"International Journal of Reconfigurable Computing Volume 2 Issue 7 ,pp.18-25 .
[14] Navneet Randhawa, and Lolita Singh (2011): A Systematic Way to Provide Security for Digital Signature Using Elliptic Curve Cryptography, International Journal of Computer Science and Technology Vol. 2, Issue 3, pp.1-6.
[15] Kun-Lin Tsai, Fang-YieLeu, Tien-Han Wu, Shin-shiuanChiou, Yu-Wei Liu, and Han-Yun Liu(2014):A Secure ECC-based Electronic Medical Record System, Journal of Internet Services and Information Security (JISIS), Volume 4, Issue 1, pp.11-18.