Journal of Theoretical and Applied Information Technology
31st August 2023. Vol.101. No 16
© 2023 Little Lion Scientific

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

ANALYSIS OF ECC AND ZKP BASED SECURITY ALGORITHMS IN CLOUD DATA

[1]E.JANSIRANI, [2]DR.N.KOWSALYA,

[1]Research Scholar, Sri Vijay Vidyalaya College of Arts & Science(Affiliated to Periyar University), Dharmapuri, Tamilnadu, India.
[2] Assistant Professor, PG and Research Department of Computer Science, Sri Vijay Vidyalaya College of Arts & Science (Affiliated to Periyar University), Dharmapuri, Tamilnadu, India.
E-mail: [1]e.jansirani2020@gmail.com, [2]kowsisara2003@gmail.com

ABSTRACT

Cloud computing is now used to store massive amounts of data in various areas such as business, military universities, and so on. We can obtain data from the cloud based on the user's request. Clients use cloud storage services to upload their files as well as authentication information to a cloud storage server. Cloud server (CS) must demonstrate to a verifier that he is actually storing all of the client's data unchanged in order to ensure the availability and integrity of the data that is being saved for clients. To save data on the cloud, many obstacles must be surmounted. There are several methods that can be taken to address these issues. For data protection, cryptography methods are gaining popularity. In cloud computing, a singular method is ineffective for providing high-level data security. In this paper, we establish a new security method based on Hybrid Elliptic Curve Cryptography and Zero Knowledge Proof (HECCZKP) Algorithm, which is used in this suggested system to provide data protection when compared to Elliptic Curve Cryptography (ECC) and Zero Knowledge Proof (HECCZKP).

Keywords: *Cloud Computing, Data Security, Elliptic Curve Cryptography, Zero Knowledge Proof, Discrete Logarithm Problem*

1. INTRODUCTION

Cloud computing has taken the internet by storm over the last decade, with many big tech companies rolling out cloud services, but it has also been characterised by a slew of high-profile data leaks. Cloud computing enables computing to be both environmentally and fiscally sustainable. Sustainability is described as "creating and maintaining conditions in which humans and nature can coexist productively to support current and future generations." [1] To guarantee the habitability of our world for future generations, sustainability should be regarded as a requirement of technological development. Environmental, economic, and social factors can all be used to determine sustainability in cloud computing. Utilizing resources effectively emphasises environmental sustainability, while meeting consumer requirements effectively without jeopardising the economy itself is the focus of economic sustainability [2]. [3]. The promotion of social stability and equal standard of life is defined as social sustainability. Each of these characteristics can be found in cloud computing. CC's power, flexibility, and simplicity of use come with a slew of security risks. Despite the fact that CC is a new intuitive way to access applications and simplify work, there are a number of challenges/issues that may prevent its widespread usage. A non-exhaustive search in this field shows some issues. These are their names: SLAs (Service Level Agreements)[4], What to transfer, how to secure it, and so on. [5]. Because CC supports automatic updates, a single change made by an administrator to a programme affects all users. This implies that any flaws in the software are instantly visible to a large number of users, which is a major risk for any organisation with little security. Many researchers also concur that security is a major worry for cloud computing adoption. Nowadays, cloud computing is a well-known tool. Companies such as Google, Microsoft, and Amazon are improving the services they provide to their customers. Users' adoption of online systems is hampered by security concerns. The cloud service providers are worried about inadequate security measures, and it is recommended that additional factors like data integrity, control, audit, secrecy, and availability be considered [6].

However, cloud performance assurance cannot be achieved until IT pros have greater confidence in the cloud environment's privacy, safety, and security. A number privacy and security threats, such as risk or malicious insider malware, have emerged as pervasive aspects of the IT environment in recent years, and should be addressed as part of both the worldwide and national digital safety agendas [7]. Organizations looking to use cloud services encounter security challenges that are not entirely dissimilar from threats and conventional security issues. For the purpose of preserving the security and privacy, the same internal and external threats exist today and necessitate appropriate disaster management and risk mitigation procedures. Data can be encrypted to protect the data transmission procedure over the internet. Encryption is the process of converting data using any type of encryption method by utilising a mixed key. The encryption key is only available to the user in order to acquire the encrypted data via the deception process. Key-based encryption is divided into two types: symmetric key encryption and asymmetric key encryption [8].

Within zero knowledge proofs, the framework is proven to be resistant to data privacy leakage. The use of a probabilistic technique aids in the reduction of computation and communication overhead[9]. The following step is to Elliptical curve cryptography (ECC) is a public key encryption method based on elliptic curve theory that can be used to generate cryptographic keys that are smaller, quicker, and more effective [8]. Instead of the conventional technique of generation as the product of very large prime numbers, ECC generates keys using the properties of the elliptic curve equation. ECC is being utilised more frequently for cloud applications because it helps to create equivalent security with less battery and computing resource consumption. Existing security algorithms that can be applied in the cloud to provide security are discussed. As previously discussed, there are numerous security algorithms accessible in cloud computing today. Aside from these, there is an urgent need to create many more efficient algorithms to improve cloud computing security. The current algorithms can be improved further to increase the security of data in the cloud. As the number of cloud computing users grows rapidly, making their data completely secure becomes a significant issue. To increase security in the cloud computing world, we therefore need some better algorithms.

Cryptographic Cloud Services: A variety of services are now available to users thanks to recent advancements in cloud storage that enable proficient data protection and decryption without the need for a third party. This improves the system's capacity for storage efficiency and security while also enabling quick and safe data retrieval. The sharing of cloud resources by various kinds of users is simple with this service, and it improves the system's capability by securing more storage after the implementation of cryptographic techniques [10].

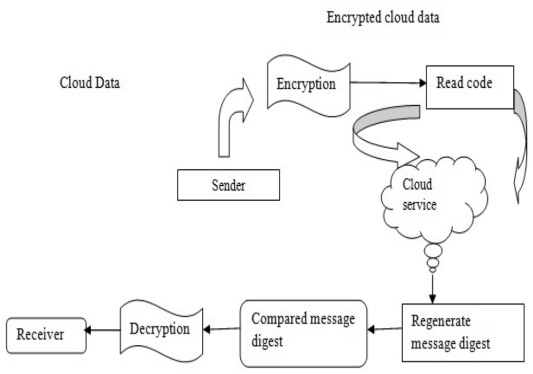


Figure 1: Cryptography Cloud Services

The various types of cloud storage systems are depicted in Figure 1. It can be seen that the data between the sender and the receiver undergoes encryption and decryption before being saved on the cloud server. It also symbolises secure data transmission via cloud storage.

Computational Security: A number of complex computational problems have been inspired by the subject of cryptography. The assumption that these problems are generally computationally difficult to answer in polynomial time forms the foundation of today's cryptosystems.

Discrete Logarithm Problem (DLP): The logarithm function for real numbers has a group counterpart known as the discrete logarithm. Discrete logarithm is a mathematical issue defined by the equation axe mod n = b, where a, n, b, and x are all positive integers and n is prime [11]. It is thought that it is still very challenging to perform discrete logarithmic calculations. There isn't a computationally effective way to solve this issue, so the commonly used discrete logarithm algorithm for public-key cryptography is used.

Elliptic Curve Discrete Logarithm Problem (ECDLP): The Elliptic Curve Discrete Logarithm Problem (ECDLP) is an additive group's extension of the DLP. Example: Let G be a subset of E(Fq) generated by the point P of prime order p. The Elliptic Curve Discrete Logarithm Problem is named after a generator P of an additive group G of rank p and the public element Q = x:P $\in$ G,The

ECDLP can be solved in O () steps using either the baby-step giant-step algorithm or the Pollard' Rho algorithm O ($\sqrt{p}$) steps ([12]).

The ECDLP is a popular method for generating EC credentials. That is, given the public key Q and the generator P, the EC key derivation function depends on the hardness of this problem to protect the chosen secret key x.

2. SECURITY ALGORITHMS

The primary emphasis is on cryptography in order to safeguard data transmission over the network. Cryptography is the practise and study of methods for securing communication and data in the presence of adversaries. Encryption and decryption methods are employed in cryptography. Some of the security methods are Eliptic curve cryptography and zero knowledge proof.

2.1. Elliptic Curve Cryptography

An elliptic curve (EC) is a smooth, projective algebraic curve of genus one with a defined point O. An elliptic curve is an abelian variety, which means that it has an algebraically defined multiplication with regard to which it is a (necessarily commutative) group, and O is the identity element. Without O defined, the curve is frequently referred to as an elliptic curve. Any elliptic curve can be expressed as a plane algebraic curve described by the following equation: (Where a and b are elements of R). However, the complete curve should have the following properties: It is an abelian group. Symmetric about the x-axis, The identity element is a point at infinity." [13]

Elliptic curve cryptography (ECC) is a public key encryption method that uses elliptic curve theory over finite fields. It was used to reduce the size, speed, and efficiency of encryption keys. In security applications where integrated circuit space and computational capacity are constrained, such as PC (personal computer) cards, smart cards, and wireless devices, the ECC algorithm is one of the most potent asymmetric algorithms for a given key length. ECC's security pattern is quite exceptional and is unaffected by side channel attacks. Variable lengths of keys have been used for encryption, and they vary in line with the data blocks to provide enough coverage for the data. An elliptic curve over a field K is a nonsingular cubic curve in two variables with a reasonable point, f(x, y) =0. (which may be a point at infinity). The complex numbers, reals, rationals, and algebraic generalisations of rationals, p-adic numbers, or a finite field are commonly assumed to be the field K [16]. We must generate both the public and private keys during this stage. The sender will encode the message using the recipient's public key, and the receiver will decrypt it using its private key. Now we must choose a number'd' from the range 'n'. We can create the public key using the following equation: d = the random number we chose from a range of (1 to n-1). P is the curve's centre. 'Q' represents the public key, and 'd' represents the secret key.

Encryption: Let'm' be the message we're conveying. We must represent this statement on the curve. These contain extensive application information. Consider the fact that'm' has the point 'M' on the curve 'E'. Choose 'k' at random from the range [1 - (n-1)]. Let C1 and C2 be the cypher texts that are produced.
C1=k*P, c2=M + k*Q
Decryption: Have to get back the message ‘m’ that was send

$$M=c2-d*c1$$

M is the original message that we have send.

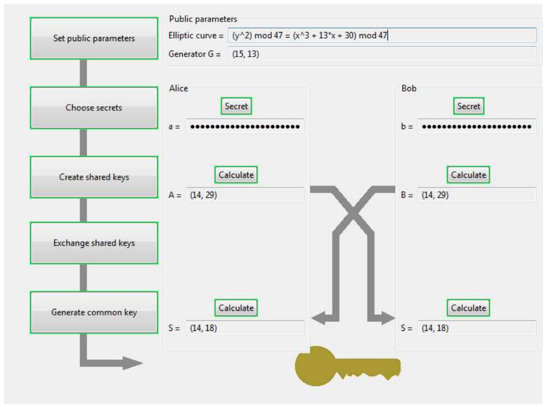
2.2.Elliptic-curve Diffie–Hellman (ECDH) :

It constitutes an secret key agreement (In cryptography, a key-agreement protocol is a protocol that allows two or more parties to agree on a key in such a way that both parties influence the result) [17]. In this case, two parties create a shared secret (a cryptographic key or data that is only known to the parties engaged in a secured communication) over a channel that is not secure, each with an elliptic-curve public-private key pair. (In contrast to a secure channel, an insecure channel is unencrypted and may be subject to eavesdropping). This discussed secret may be used directly as a key or to derive another key (Such use may be expressed as DK = KDF(Key, Salt, Iterations) where Key is the original key or password, Salt is a random number that serves as cryptographic salt, and Iterations refers to the number of iterations of a sub-function. The key, or the derived key, can then be used to encrypt future communications with a symmetric-key cypher. (Symmetric ciphers use the same cryptographic keys for encryption of plaintext and decryption of cipher text.). ECDH is a similar scheme built on the addition of points on an elliptic curve. The fundamental operation is combined to form a primitive function known as a keyed one-way function in this case. A function that accepts two inputs, one of which is secret (such as the key), and outputs only one is known as a keyed one-way function. The output must be easy to compute given the two inputs. However, calculating the key using only the other input and result must be computationally infeasible. In this manner, each party can use their secret key without disclosing it to anyone else, whether the other party or an eavesdropper.

Algorithm: Assume Alice and Bob want to trade a confidential key. The following methods are used to accomplish this:

- To begin, Bob and Alice create their own secret and public keys. For Alice, we have the secret key dA and the public key HA = dA G, and for Bob, we have the keys dB and HB = dB G. Both Alice and Bob are using the same domain parameters: the same basis point G on the same elliptic curve on the identical finite field..
- Alice and Bob trade their public keys, HA and HB, over an insecure channel. The Man in the Middle will intercept HA and HB but will not be able to determine dA or dB unless the discrete logarithm issue is solved.
- Alice computes S = dA HB (with her private key and Bob's public key), and Bob computes S = dB HA (with his private key and Alice's public key). It's worth noting that S is the same for both Alice and Bob: S = dA HB = dA (dB G) = dB (dAG)

ECDH Example:
Step 1: Set public parameters
Curve type: F (p), Curve Size: Large, Domain parameters: a=3,b=6,p=47, generator G=(15,13)
Elliptic Curve: (y^2) mod 47 = (x^3 + 13*x + 30) mod 47
Step 2: Choose Secrets
Alice= 8
Bob=8
Step 3: Generate shared keys
Secret key (d): Q=d*G,
Alice= (14,29)
Bob= (14,29)
Step 4: Exchange shared keys
Step 5: Generate common key
Key = sA*QB and key=sB*QA
S= (14, 18)



Public parameters
Elliptic curve = (y^2) mod 47 = (x^3 + 13*x + 30) mod 47
Generator G = (15, 13)

Alice
a = ••••••••••••••••••••••
A = (14, 29)
S = (14, 18)

Bob
b = ••••••••••••••••••••••
B = (14, 29)
S = (14, 18)

2.3.Elliptic Curve Digital Signature Algorithms (ECDSA)

- The Federal Information Processing Standard (FIPS) for the United States Government, referred to as the Digital Signature Standard, contains the specifications for the Digital Signature Algorithm (DSA). (DSS). The prime-order subsets of Z*p's discrete logarithm problem is the foundation of DSA's security. Vanstone (1992) proposed ECDSA in answer to the National Institute of Standards and Technology's (NIST) request for public feedback on their first proposal of a Digital Signature Standard. (DSS). It was approved in 1998 as an ISO standard (ISO 14888-3, 1998), an ANSI standard (ANSI X9.62, 1999), an Institute of Electrical and Electronics Engineers (IEEE) standard (IEEE 1363-2000), and a FIPS standard (FIPs 186-2), all of which were approved in 2000. (NIST, 2000).
- The elliptic curve counterpart of the DSA is ECDSA. That is, instead of working in a subgroup of rank q in Z *p, we work in an elliptic curve group E.(Zp). A signature algorithm is used to validate a device or a communication sent by a device. Consider the following two devices: A and B. To verify a message sent by A, the device A uses its private key to sign the message. The communication and signature are sent to the device B by device A. Only the public key of device A can be used to verify this signature. Because device B is aware of A's public key, it can determine whether or not the communication was sent by A. ECDSA is an elliptic curve group-based variation of the Digital Signature Algorithm (DSA). Both parties must concur on Elliptic Curve domain parameters before sending a signed message from A to B. The private key dA (a chosen number at random less than n, where n is the rank of the curve, an elliptic curve domain

parameter) and the public key QA = dA * G belong to sender "A." (G is the generator point, an elliptic curve domain parameter)

Signature Generation

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

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

Signature Verification

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

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

Example:

Signature originator: Jaya Jaya
Domain parameters to be used 'EC-prime239v1':
Chosen signature algorithm with hash function SHA-1
Size of message M to be signed: 800 bytes
Bit length of c + bit length of d = 478 bits
Message = Jansirani,Sri Vijay Vidyalaya College of Arts & Science,Dharmapuri.
Encrypted Data:

```
20 43 6C 6F 75 64 20 96 20 54 68 65 20 74 65 63
68 6E 6F 6C 6F 67 79 20 6F 66 20 64 69 73 74 72
69 62 75 74 65 64 20 64 61 74 61 20 70 72 6F 63
65 73 73 69 6E 67 20 69 6E 20 77 68 69 63 68 20
73 6F 6D 65 20 73 63 61 6C 61 62 6C 65 20 69 6E
66 6F 72 6D 61 74 69 6F 6E 20 72 65 73 6F 75 72
63 65 73 20 61 6E 64 20 63 61 70 61 63 69 74 69
65 73 20 61 72 65 20 70 72 6F 76 69 64 65 64 20
61 73 20 61 20 73 65 72 76 69 63 65 20 74 6F 20
6D 75 6C 74 69 70 6C 65 20 65 78 74 65 72 6E 61
6C 20 63 75 73 74 6F 6D 65 72 73 20 74 68 72 6F
75 67 68 20 49 6E 74 65 72 6E 65 74 20 74 65 63
68 6E 6F 6C 6F 67 79 2E
```

Elliptic curve E described through the curve equation: y^2 = x^3 + ax + b (mod p) :
a = 883423532389192164791648750360308885314476597252960362792450860609699836
b = 738525217406992417348596088038781724164860971797098971891240423363193866
Private key = 1545029212
Public key W=(Wx,Wy) (W is a point on the elliptic curve) of the signature originator:
W = 339214281597464032556847389887550928141978861389490192243649358977026794
Wx = 484991443796850980360798175808921247277365493475419629782116460133815756
Wy = 112887102563797242312427681894133296215223477833624991967041647927204875
Calculate a 'hash value' f (message representative) from message M, using the chosen hash function SHA-1.
f = 12292441878677729304482756804595911871064 62377917

- **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 = 110282003749548856476348533541186204577905061504881242240149511594420911
Gy = 869078407435509378747351873793058868500210384946040694651368759217025454
r = 883423532389192164791648750360308884807550341691627752275345424702807307
The secret key s is the solution of the EC discrete log problem W=x*G(x unknown)
S= 867394451498200440756814680713079325021677452565139760374640511533239520
Signature:
Convert the group element Vx (x co-ordinates of point V on elliptic curve) to the number i:
i = 874844795342404242242970406879030982109625529775978715975802989639190378
Calculate the number c = i mod r (c not equal to 0):

c =
874844795342404242224297040687903098210962
552977597871597580298963919037 8
Calculate the number d = u^(-1)*(f + s*c) mod r (d not equal to 0):
d =
499738226452546966560511315664527861087957
7473887696570203503575168128 89

- **ECDSA VERIFICATION as follows:**

If c or d does not fall within the interval [1, r-1] then the signature is invalid:
c and d fall within the required interval [1, r-1].
Calculate the number h = d^(-1) mod r:
h =
391702945635414917822160942289261042032913
1968407031794154631242308962 39
Calculate the number h1 = f*h mod r:
h1 =
328051527420541012757503247282799137806351
2478894548544364811239211490 57
Calculate the elliptic curve point P = h1 G + h2 W
Calculate the number h2 = c*h mod r:
h2 =
250945703657321186443879133837810133503609
0875869887369994509709829843 7
(If P = (Px, Py) = (inf, inf) then the signature is invalid):
Px =
874844795342404242224297040687903098210962
552977597871597580298963919037 8
Py =
533920364062706920253636484590149343512641
4440978750925238673785381931 85
Convert the group element Px (x co-ordinates of point P on elliptic curve) to the number i:
i =
874844795342404242224297040687903098210962
552977597871597580298963919037 8
Calculate the number c' = i mod r:
c' =
874844795342404242224297040687903098210962
552977597871597580298963919037 8
If c' = c then the signature is correct; otherwise the signature is invalid:

3. ZERO KNOWLEDGE PROOF

In general, a zero-knowledge permits a proof of an assertion's truth while conveying no other information about the assertion other than its actual truth [18]. Typically, such a protocol includes two entities: a prover and a verifier. A Zero-Knowledge Proof enables the prover to demonstrate knowledge of a secret while revealing no information that the verifier can use to convey this demonstration of knowledge to others. The examples of interactive and non-interactive proof systems that will be addressed are zero-knowledge systems. In the initial group, a prover and a verifier exchange multiple messages (challenges and responses), usually based on random numbers which they may keep private, whereas in the second, the person who proves sends only one message. In both systems the prover's objective is to convince the verifier about the truth of an assertion, e.g. the claimed knowledge of a secret. The verifier either accepts or rejects the proof.

A zero-knowledge proof is described formally by three conditions: complete, sound, and perfect (also known as zero-knowledge). To be deemed zero knowledge, a proof must completely satisfy these three conditions. A proof is considered full when the individual who is verifying agrees that a fact is being verified and is correct at every step. The second condition, sound, is met if the individual verifying the data is confident that the information is correct. Lastly, evidence is deemed perfect or zero-knowledge if the individual verifying the information has gained no relevant knowledge about the subject under scrutiny. The accompanying example was created to demonstrate these abstract concepts.

3.1. Interactive Proof

An interactive proof is a protocol that allows a prover P and a verifier V to communicate. The protocol is divided into rounds, with V asking questions in each round based on P's responses in earlier rounds. We formalise this by imagining P attempting to persuade V that (x) = y, and we provide formal formulations below. Consider C to be a function. If the following conditions apply, a pair of interactive machines P,V is an interactive proof for C with soundness:Completeness. For every x such that $(x) = y$ it holds that $\Pr[\langle P,V\rangle(x) = accept\ ] = 1$.

- ϵ-Soundness. For any x with $(x) \neq y$ and any P∗ it holds that $\Pr[\langle P*,V\rangle = accept\ ] \leq \epsilon$

We say an interactive proof scheme has succinct proof size and verifier time if they are (polylog($|C|$, $|x|$)).

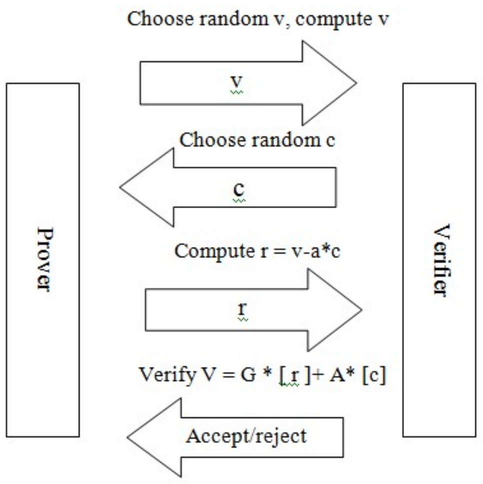


Figure 2 : Hand Shake Protocol

Proof of Zero Knowledge that is interactive. Before the protocol starts, the Prover has chosen a private key and computed and published the public key. Throughout the protocol, the Prover and the Verifier exchange messages as the Prover tries to authenticate himself/herself.

Example: Interactive proof

- The Interactive Zero Knowledge Proof steps are described below. In a handshake protocol manner, the Prover and the Verifier exchange information. Figure 1 summarises this procedure.
- Prover selects private key an at random from [1,n- 1].
- The public key A = G [a] is computed and published by the prover.
-
- The prover selects v at random from [1,n-1] and computes V = G [v] before passing V to the validator.
- Verifier chooses challenge c at random from [0; 2^{t-1}] where t is the bit length of the challenge. Verifier sends c to prover.
- Prover computes r = v- a * c and sends it to verifier.
- Alice performs the following checks:
- Verifies A is a valid point on the curve (i.e. not the point at infinity)
- Verifies G × [r] + A × [c] is equal to the original V prover sent

As we can see, prover never had to give Alice his private key. To complete the identification process, the prover and verifier used ECC to make calculations using only publicly available information.

3.2. Non-interactive zero-knowledge proofs:

The prover and verifier interact over numerous rounds in interactive zero knowledge proofs, and their responses can vary based on the messages they have received so far. Non-interactive zero knowledge proofs, on the other hand, comprise of a single message sent by the prover to the verifier. Non-interactive proofs are usually more difficult to construct than interactive proofs, and they frequently rely on more shaky assumptions. They are, however, helpful in situations where interaction is not possible or should not be permitted, such as digital signatures and encryption schemes. A proof for the 3-colorability of graphs under a number-theoretic assumption was generated using non-interactive zero-knowledge proofs [19].

There are two steps to a non-interactive zero knowledge proof system for a language L. The first step, which may be interactive, establishes some information that is shared by both the prover and the verifier, as well as (possibly) some private information for each. This pre-processing occurs irrespective of the theorems to be proved. In a subsequent step, the prover selects and proves

Based on the information from the first step, theorems are presented to the verifier with zero knowledge. This step of theorem proving is not interactive. P = (P1; P2) and V = (P1; P2) are the prover and validator, respectively. (V1; V2). The levels are completely separated from one another. P1 and P2 could, for example, be entirely distinct: the former a trusted centre and the latter a prover in the traditional sense who sees only information provided by the centre.

Let k be a protection parameter. Assume that P1, P2, and V1 are all polynomial time probability interactive TMs, and V2 is a deterministic interactive TM. The pair P = (P1; P2); V = (V1; V2) constitutes a non-interactive proof system for a language L if the conditions of the following two stages are satisfied for all suitably large k:

Pre-Processing Stage: When P1 interacts with V1, three outcomes are produced: p; SP; and SV, where p is shared by P1 and V1, SP is shared by P1, and SV is shared by V1. If P1 does not follow the procedure, V1 will almost certainly output "cheating."

1. **Theorem Proving Stage:** P2 proves theorems to V2 without interaction (the communication on any input is a single message from P2 to V2).

Completeness: For every $x \in L \cap \{0; 1\}^k$

P(V2(1^k; p; S_V ; x , β) = accept : (p, SP , SV) (P1 \$ V1)(1^k) ;
$\beta \leftarrow$ P2(1^k, p, SP , x)) ≥ 1 $^{-2k}$:

Soundness: For every interactive TM $\overset{\frown}{p2}$ For every x $\in \overset{\frown}{L} \cap \{0; 1\}^k$
P(V2(1^k; p; S_V ; x , β) = accept : (p, SP , SV) (P1 \$ V1)(1^k) ;
$\beta \leftarrow$ P2(1^k, p, SP , x)) ≥ 1 $^{-2k}$:
In some cases we will also allow the proof system to have as additional input a history of previous theorems and their proofs.

Private Data Possession Scheme We define an elliptic curve EC over an additive subgroup G1 of prime order q in our method. Assume P is a G1 generator. When a client wishes to store a file data D on the cloud, he first decomposes D into two blocks s and n, where n represents the quotient and s is the remainder, using the Euclidean Division (ED) on the file D with the divisor b. It should be noted that b is kept secret by the customer and is used in the decomposition of several outsourced file data. That is, b symbolises the one-of-a-kind secret information that the client should keep for all requests for data possession verification evidence. It is important to note that b is inextricably linked to the security of our remote verification system. As a result, the definition of multiple data divisors can be used to broaden our proposal. That is, depending on the sensitivity of the data that the data owner wants to share on the cloud, the data owner may rely on various secrets.
Then, in terms of the ECDLP, the published elements are bP, nP, and sP, represented by pk, 1, and 2, respectively. The public key is pk, and the public parts of the file D are 1 and 2. In the following, we will use R, B, and K to meet the requirements in [20]. We additionally indicate scalar point multiplication in an additive group by • and two element multiplication in a multiplicative group by *. The first algorithm illustrates the general idea of the private data possession scheme. This plan is divided into two stages. The keyGen Setup processes are carried out in the initial stage. This stage is only carried out after the content has been uploaded to the cloud. The second phase happens when the client wishes to validate the file's authenticity. It generates a new challenge to obtain proof of data possession from the cloud server for this reason. This latter executes the Proof method, which is a three-step process. The actions that are taken in each of the two aforementioned phases are detailed below.
Preprocessing D=nb+s
Public parameters (EC,+) an elliptic curve
P a generator of Ec
(n.P,b.P,s.P}
R,B and k three integers such that R>>BK
Secret key sk=b where b $\in$ [0,R]
Client (c) Storage server(csp)

Choose $b^{'} \in_R$ [0,R] $\xrightarrow{\mathrm{Re}\,quest(b^{'},ID)}$

$CalculateD_{1} = mb'\_z, choose(r,t) \in_R [o,B]^2$

$\xrightarrow{x1,x2}$

$Calculate(r.P,t.P) = (x1,x2)$

$generate \;\; c \in_R [0,K]$ $\xrightarrow{C}$

Calculate(r+cz).p,(t+cm).P) = $(\gamma 1.P, \gamma 2.P) = (y1, y2)$

$\xrightarrow{y1,y2}$

$check \;\; y1.P - x1 - c.\sigma_2 = c.sk.\sigma 2 - b'.(y2.P - x2)$

4.INTEGRATION OF ZERO KNOWLEDGE ENCRYPTION AND CLOUD COMPUTING

To begin, any data uploaded by a user to a cloud computing network should be encrypted. Current cryptography techniques are extremely secure and dependable. By using encryption, the user ensures that their files are completely and thoroughly protected. This process can be finished by obtaining a public and private key for encrypting and decrypting user files using a third-party application. Under no conditions should a cloud computing provider also provide user data encryption. If the provider distributes private and public keys, it means they have access to the keys and may keep them, making the encryption ineffective. Once encrypted, the data can be uploaded to the cloud network. The cloud network is the most vulnerable to cyber attacks. Because of the complexity of the discrete logarithm issue, even if the cloud network is breached and user data is leaked, there is no significant risk to the user as long as the data is encrypted and the cloud service does not have user keys.

As a user, you'd like to get those items back. PDP algorithms were implemented to enable users to verify the authenticity of their files, but following a

PDP to retrieve files is also recommended. To make the process even safer, the cloud computing network should verify that the user is who they say they are, lest a hacker gain access to the user's personal device. A zero-knowledge proof is an efficient and secure method for the user to verify their identity and the cloud to verify its identity.

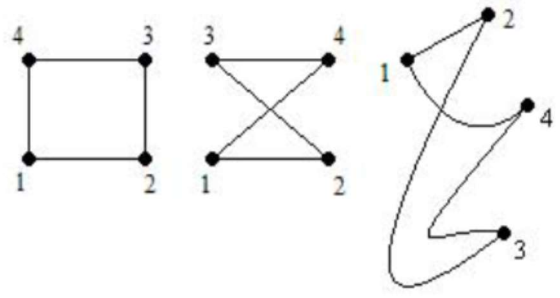


Figure 3: Example Of Two Isomorphic Graphs

The application of the Zero Knowledge Proof is done through mathematical computations, one of which is the calculation of discrete logarithms. "Implementing Zero-Knowledge Authentication with Zero Knowledge (ZKA)" [21] discusses the methods used for web authentication using discrete logarithms. Using SPK1 (x): Y = goX, the algorithm is built on [22], Zero Knowledge Proof of Knowledge Sigma Protocol. This algorithm contains three steps, which are as follows:

a. Initialization:

1. Given a group G and g0, g1 obtained from a random element of G.
2. Then G and g0 be the public key.

b. Registration:

1. The user enters a username and password.
2. Password was hashwd by a hash function. Obtained x= hashing (password).
3. Users do calculations for Y = g_o^x.
4. Then the user sends (username Y) to the server.
5. Server stores the username Y.

c. Authentication Process:

1. Server generates a random variable a, save it and send it to the client.
2. The user enters a username and password.
3. Client hash password by hash function, and compute x = hash (password).
4. Client calculate Y = g_o^x.
5. Client generates random rx 2 G and compute T1 = g_o^{rx} .
6. Client compute c = hash (Y, T1, a) and zx = rx - c x.
7. Client sends (c, zx) to the server.
8. Server calculates T1 = Yc g0zx and matching c2 = hash (Y, T2, a)
9. If appropriate, the user has been authenticated. The technique of Zero Knowledge Proof is a common technique used to prove knowledge of the variables associated with the password.

Each of the components used in Zero knowledge authentication algorithm are,

- G (Public Value) -> A set of numbers based on a formula.
- g_0 (Public Value) -> A number that is obtained from G and an element of G.
- X (Clients Secret Value) -> the result of hashing password which inputted by the prover.
- Y (Clients Secret Value) -> An alias of provers password used by verifier to calculate and proof of knowledge.
- a (Servers Secret Value) -> Random tokens that are generated when prover login.
- T_1,rx,zx,c (Calculation Result) -> Other variable used in calculation.

The algorithm can function if the variables linked with the password are known. The method began by generating G from a number such as 1,2,3,4,..., and g0 is derived from the outcomes of random numbers in G. Because the algorithm was derived using g0 from G, obtaining a logarithmic discrete calculation is challenging. In the client and server systems, each component of the Zero Knowledge Proof algorithm is applied. We can compute as follows, as shown in Table 1:

Table 1: Authentication Process

No	User(prover)	Verifier(server)
1	-	Random a
2	Receive a	Send a
3	Calculate x= hash(password)	-
4	Calculate Y= g_o^x	-
5	Get random rx	-
6	Calculate T_1= g_o^{rx}	-
7	Calculate c=hash(Y,T_1,a)	-
8	Calculate zx=rx-cx	-
9	Send c,zx	Receive c,zx
10	-	Calculate T-2=Y^c g_o^{zx}
11	-	Compare if c=hash(y,T_2,a

Key Generation: After generating the elliptic curve's points, a basis point of order n is chosen from among them. This is the starting place for generating private and public keys. It proceeds like this: To act as the private key, a number "k" is chosen at random from 1 to (P-1) where P is a large prime number. This private key is then used to calculate the public key by using the base point (generating point) of the formed elliptic curve. The above criteria are used to generate all of the public and private keys in the suggested scheme. The private key is used to encrypt the communication, and the public key is used to decrypt it.

Proving and Signing: After getting the decrypted message, the signer computes the message's actual value. The signer computes using zero knowledge notions. The signer employs zero knowledge concepts for verification during this phase. The person who signed requests the number of e from the sender, which can be either 1 or 0. The person who signed confirms the message to be the same as the sender's message based on this value.

Verification: In this case, the blind digital signature with requester/sender can be validated by any verifying authority using the signer's public key, which can be considered a simple electronic signature.

5. PROBLEM STATEMENT

Cloud is designed to support both robustness and efficiency while taking into account the limited storage and computing capabilities of user devices. It must meet the following criteria:

- Security of information confidentiality - our plan must safeguard the security of outsourced data contents from both curious providers and malicious users.
- Adjustable access control - The cloud should provide adaptable security policies for users with varying granted privileges who pertain to various groups. These access control rules should ensure the confidentiality of outsourced data contents both backward and forward.
- Effective user removal - the removal of a group member should have no effect on the surviving users. In other words, unlike conventional fine-grained access control schemes, the challenge is to define a smooth group revocation that does not necessitate changing the secret keys of managed to overcome-revoked members.
- Low computation overhead - On the one hand, the amount of computation at the cloud storage server should be kept to a minimum for scalability reasons, as the server may be engaged in concurrent interactions. However, the proposed algorithms should have a minimal processing complexity on the client side.
- Low transmission costs should be used by the cloud to reduce bandwidth consumption.
- Low storage costs – the restricted storage capacities of user devices were crucial in the design of our solution. As a result, low client storage costs are highly encouraged.
- Privacy addresses the confidentiality of data for particular entities, and it carries legal and liability concerns. It should be regarded as a technical challenge as well as a legal and ethical concern. Protecting privacy in any computing system is a technical challenge, and in a cloud setting, the challenge is exacerbated by the distributed nature of clouds and the potential lack of user knowledge about where data is kept and who has or can access to it. The following two cloud computing features appear to be the most essential in terms of security and privacy: Outsourcing of data services and processing. The primary issue inherent in the cloud computing paradigm is the secure outsourcing of sensitive as well as critical data and processes.
- In contrast, ECC employs asymmetric key encryption, which employs two keys for encryption and decoding, a public and a private key. This is why its security level is greater, and it is difficult for hackers to crack both keys at once. ECC is also notable for having a smaller key area. ECC can provide the same degree of security as other algorithms while using a smaller key size. There is a need to create a system that secures data in the cloud while requiring less computational expense and time for the encryption/decryption process.

6. RESEARCH CONTRIBUTIONS

The following are the major contributions of this paper:

- We suggest a hybrid model that combines two algorithms, ZKP and ECC, and uses ECC to generate ZKP keys. In other words, we are not using the key produced by the ZKP algorithm; instead, ECC is used to generate the key, resulting in a smaller key size.
- For data encryption and decoding, a public key or a private key is used. As a result, this procedure necessitates a large key size and a lot of computational power. The suggested hybrid algorithm (ZKP-ECC) is used to improve system security in less time by addressing the issue of key size, and also to

reduce computational power for memory optimisation.

- In addition, we present an algorithm for our proposed framework that explains how the public key is generated using the ECC algorithm and how encryption/decryption is performed using ZKP.

7. HYBRID ELLIPTIC CURVE CRYPTOGRAPHY ZEROKNOWLEDGE PROOF (HECCZKP)

The figure below clearly shows how ZKP, in conjunction with ECC, successfully secures data stored in the cloud. Zero-Knowledge Encryption is a more secure method to manage this delicate procedure.Without going into too much detail, The Zero-Knowledge is based on three major requirements:Fullness is the ability of an honest prover to persuade the verifier that he has the password by carrying out some process in the necessary manner.The verifier will almost definitely detect when the prover is lying.Zero-knowledge — if the prover has a password, the verifier only gets the knowledge which the assertion is true. The new suggested diagram clearly shows the novelty of the proposed method, in which there is secure transmission of user data to server and then storage mechanism is even secured due to encrypted data. Furthermore, novelty can be calculated in terms of computational expense and time. The suggested cloud system works from both the data user and the service provider perspectives. The service supplier is in charge of the entire cloud environment. The dataset is uploaded to the cloud with greater anonymity in this flow. To accomplish the highest level of security, the proposed system explains the idea of cryptography. Encryption is performed using Elliptic curve cryptography and zero knowledge proof in this idea, which is referred to as hybrid elliptic curve cryptography zero knowledge proof.The secret key is generated using a hybrid method. In order to create the two input points (referred to as elliptical curve points) needed to run the elliptical curve, our suggested algorithm creates a random 128 bit hash. The overall data is stored in cloud storage after the encryption procedure. Decryption with a secret key is needed to retrieve the file from the data owner's sider. Finally, the data is downloaded and sent to the data proprietor.

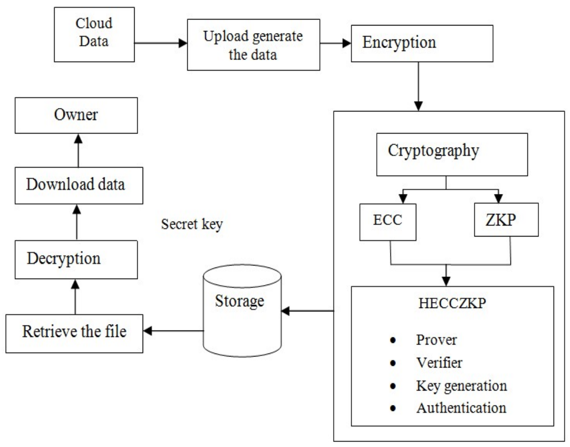


Figure 4: Proposed Methodology Of HECCZKP Cloud Storage Technique

When storing data on a cloud server, the primary limitation is security. Data loss, data leakage, user authentication, malicious user handling, incorrect use of cloud computing and its services, insider threats, outsider malicious assaults, data loss, loss of control, and service disruption are some of the security dangers in cloud computing. As a result, improving _ the security for multimedia data storage in a cloud facility is critical. The primary goal is to create an architecture that assures users that their data is secure. Current methods require some tweaking to improve the security and accuracy of data storage and access among multiple users.Decryption takes a long time as well.

Proposed Algorithm:

Step1: Initialization: In this phase, the EC is formed by providing the value of coefficients a and b of the equation of the elliptic curve i.e

y2 = *x*+ + b

Generation of ECC :y2 = x3 + ax + b

Condition: 4a3 + 27b2 ≠ 0

Compute: y = sqrt(x3 + ax + b)

Step2:Encryption/Decryption

In this section, we present the methodology for encryption and decryption. The message ‘m’ that is to be sent will be encoded as a series of points represented as (u(x),v(x)). The encoded message is referred as Em. The algorithm works as follows: To encrypt and send a message to B, A performs the following steps.

$k \in_R N$ (choose k as a random positive prime number in N)

$Q \leftarrow [k]D$ (D is the Divisor of the EC & The form of Q is (u(x),v(x)))

$P_k \leftarrow [k]P_B$ (P_B:(u(x),v(x)) is receiver’s(B’s) public key)

$C_m \leftarrow \{ Q , E_m + P_k \}$ (C_m :(u(x),v(x)) is the Cipher Text to be sent)

To decrypt Cipher text message, the Decryption algorithm works as follows: To decrypt the Cipher Text Cm , B extracts the first coordinate 'Q' from the cipher text then multiply with its Private Key (aB) and subtract the result from the second coordinate.

Step 3: Completeness of verification: In our scheme, the completeness property implies public verifiability, which allows any entity, not just the client (data owner), to challenge the cloud server for data possession or data integrity without the need for any secret information. That is, the public verification components required in the verification process are publicly known. As a result, any authorised user can challenge the server storage and efficiently verify data possession evidence.

8. EXPERIMENTAL RESULTS

The experiments in this paper were intended to put the proposed model to the test. According to the suggested security model, the computed execution time was for encryption and decryption. All tests were carried out on an Intel core 2.26 GHz processor with 512 KB cache and 2 GB of RAM. The system was operating Microsoft OS "Windows 7," and Python 2.7.6 was used for mathematical calculations, with the NumPy Python plugin. In addition, all of the graphs in this article were created in Python with the matplotlib plugin. Algorithms used the charm cryptography framework's crypto library, with keys based on each of the security settings.

Encryption time, decryption time, input size and throughput, energy consumption, uploading speed, and security level are all used to assess the suggested method. Files of various sizes are used to run experiments in which the algorithms ECC, ZKP, and the proposed method are compared.

- Encryption time (Computation Time/Response Time): The encryption time is defined as the amount of time it takes an encryption method to generate a cypher text from plain text.
- Decryption time (Computation Time/Response Time): The decryption time is the amount of time it takes an encryption method to recreate plain text from cypher text.
- Input data size- Each algorithm requires a different amount of memory space to execute the operation. The amount of memory needed by any algorithm is decided by the size of the input data, the number of rounds, and so on. The finest algorithm is one that uses little memory while performing well.
- Throughput-For each algorithm, the throughput is determined by dividing the total plaintext in Megabytes encrypted by the total encryption time.

The charts below provide a comparison of hybrid encryption algorithms. In the table, the trial outcome encryption file size comparison between ECC, ZKP, and proposed algorithm is efficient performance is analysed for cloud environment. The findings shown below demonstrate that the proposed algorithm produces better results in terms of the storage size parameter..

Energy Consumption: Figure 5 compares the proposed HECCZKP and ECC,ZKP algorithms for encryption and decryption at various plain text sizes. The results demonstrate the superiority of the HECCZKP algorithm over other algorithms in terms of encryption and decryption power processing (when we encrypt the same data using ECC,ZKP, and HECCZKP algorithms, we discovered that HECCZKP requires approximately 30% of the energy consumed by the least of the other algorithms). Despite what has been stated earlier, tasks sent to the Cloud are expected to be independent of one another. When compared to ECC and ZKP, HECCZKP's energy usage has been admirably low. In order to provide a detailed energy production in terms of ratio.

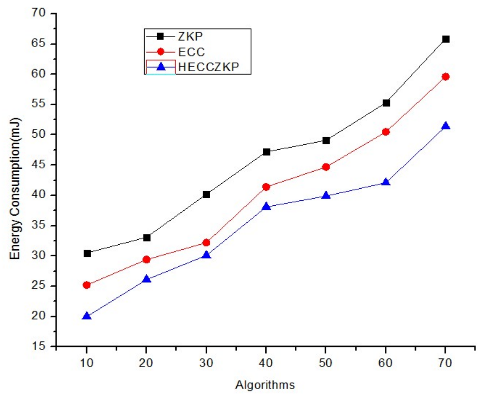


Figure 5: Energy Consumption

Throughput : The throughput of an encryption algorithm is calculated using encryption time. It denotes the encryption pace. The encryption scheme's throughput is determined as follows:

Encryption throughput = Tp (Bytes) / Et (Sec)

Tp denotes the entire plain text (in bytes), and Et denotes the encryption time. (second). The energy consumption of this encryption method decreases as the throughput value increases.

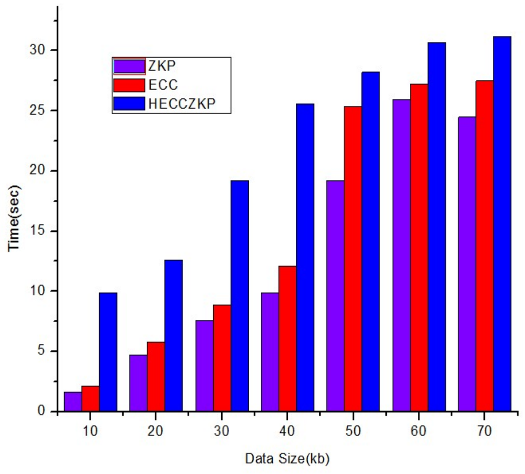


Figure 6: Throughput

Figure 6 compares the output of the HECCZKP algorithm to that of the ECC, ZKP algorithm for various sizes of plain text. It is demonstrated that HECCZKP achieves the same outcomes and the highest values.

Performance analysis for uploading speed and security overhead: Uploading speed is generally defined as the time it takes to send data files from a computer to the internet. In the research, uploading speed is evaluated by sending encrypted data to the cloud. The uploading speed fluctuates according to the bandwidth rate (100 mpbs) and traffic load. The upload speed is approximated for file sizes ranging from 0.1 to 500 MB. The upload speed of the proposed system is anticipated to range between 9.05 and 14.9 megabits per second (Mb/s). Figure 7 clearly shows the preceding description.

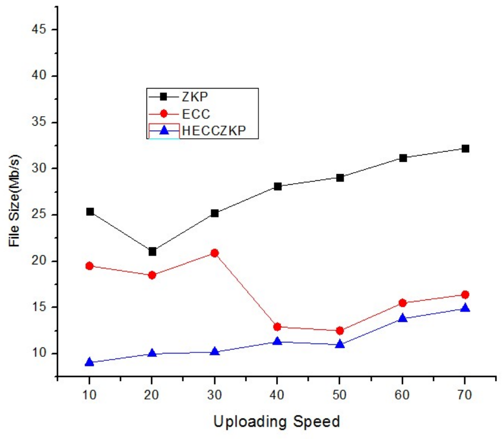


Figure 7: Uploading Speed Security Overhead

Figure 7 shows an evaluation of our hybrid model's performance in comparison to that of various algorithms used independently. The performance measurements show that ZKP is the slowest of the three algorithms, ECC is marginally quicker, and ZKP is the fastest of all three. When comparing the efficacy of our model to that of ECC, it is clear that it outperforms. Our suggested HECCZKP works slightly faster, despite the small difference.

Security level: The security of the ECC, ZKP, and proposed HECCZKP algorithms is investigated. A high degree of security is required when storing data in the cloud. Figure 3 compares the security levels of the proposed HECCZKP, ECC, and ZKP algorithms. The level of security changes depending on the number of records. When the number of records is 60, the proposed HECCZKP algorithm has a security level of 31, but in the ECC, ZKP has a security level of 14 and 8, which is lesser than the proposed HECCZKP algorithm. When compared to the existing ECC, ZKP algorithm, the proposed HECCZKP algorithm provides better record security. This graph and explanation show that the proposed HECCZKP algorithm is more secure than the ECC and ZKP algorithms.

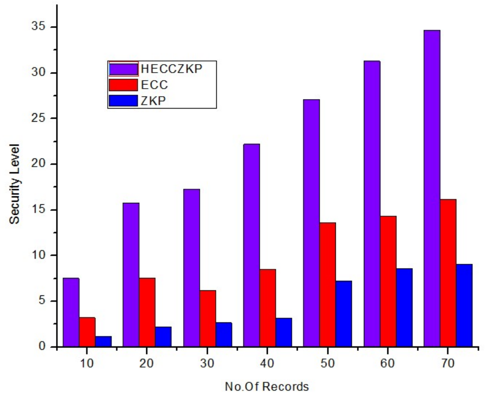


Figure 8:Security Level

Encryption Time Based On Different Input Size: While dealing with cloud users, encryption time was used to estimate the speed of the proposed system. The shorter period demonstrates fast communication between the user and the cloud server.The encryption time is the amount of time it takes a programme to generate ciphertext from plaintext. We analyse the encryption times of ECC, ZKP, and the proposed HECCZKP algorithm in this section. In the graph below, we demonstrate the encryption time performance of ECC, ZKP, and the proposed HECCZKP algorithm. Based on the results, the suggested algorithm outperforms other algorithms.

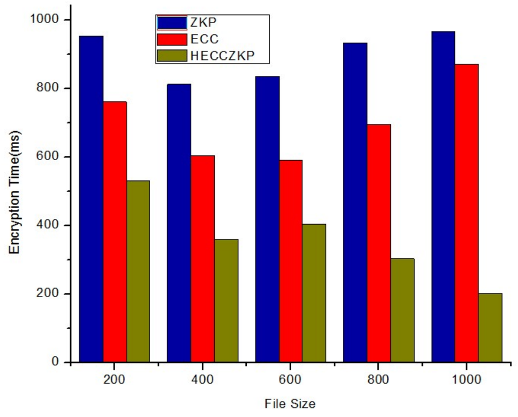


Figure 9: Encryption Time Vs File Size

Decryption Time Based On Different Input Size: The time it takes a programme to generate plaintext from cypher text is referred to as the decryption time. In this section, we examine the decryption times of ECC, ZKP, and the proposed HECCZKP algorithm. In the graph and table below, we compare the decryption times of ECC, ZKP, and the suggested HECCZKP algorithm. In accordance with the results, the suggested algorithm outperforms others.

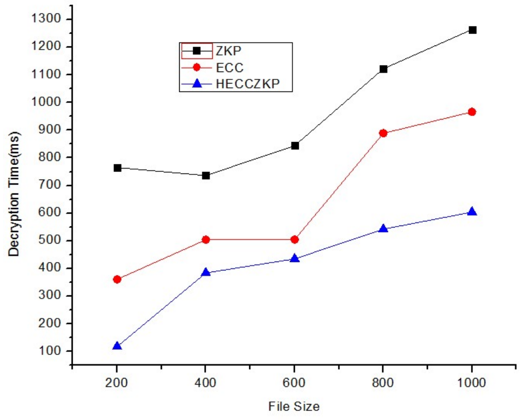


Figure 10: Decryption Time Vs File Size

Key Generation Time: The results of the key generation time experimentation for various node sizes, key sizes for ECC,ZKP, and the proposed HECCZKP algorithms have been completed. HECCZKP Key generation outperforms ECC and ZKP at all key lengths, and this is particularly noticeable as key length increases.

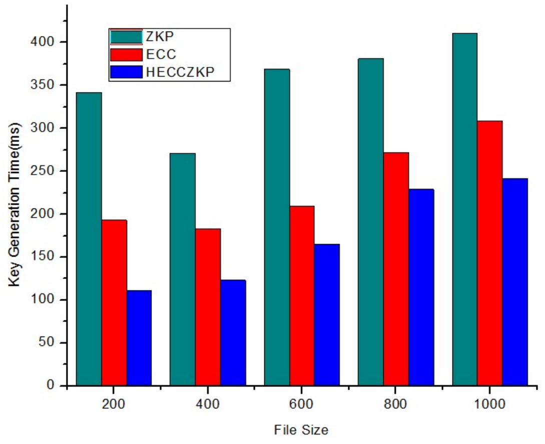


Figure 11: Comparison Of Key Generation Time

HECCZKP is faster at producing the public/private key using comparable lengths than ECC, ZKP, despite the latter's lack of resources devoted to the computationally demanding generation of prime numbers. Because ECC keys have varying lengths, HECCZKP key generation time was faster than other algorithms. According to the experimental findings, key generation time rises gradually for a given key size as node 'n' and threshold 't' increase. The generation time grows rapidly as the key size grows. Figure 10 depicts the time required to generate a key for each of the four methods. When compared to ECC and ZKP, HECCZKP produces better outcomes.

9.. CONCLUSION

The primary client worries in the cloud storage network are security and data integrity.Because cloud storage is one of the most widely used services in almost every industry, one of the most significant user concerns is the security of data saved on it. Encryption is a critical security method for maintaining confidentiality because it is done before transferring data from the device to the cloud via networks. Combining two encryption methods, one of each type, results in a superior cloud storage cryptosystem enhancement. In addition to preserving confidentiality, usability, and scalability, the hybrid elliptic curve cryptography zero knowledge proof algorithms provide a high degree of security. The data is stored in the cloud in ciphertext form rather than its original shape. The use of a random asymmetric key makes recovering the original content of the ciphertext challenging. As a result of the practical application of the algorithms described here, information on the cloud is protected from being stolen via side-channel attacks. As a result, the material in the cloud is safe. In this proposed system, a novel security method based hybrid elliptic curve cryptography zero knowledge proof (HECCZKP) algorithm is used to provide data authentication.

REFERENCES

[1]. United States Government. US EPA. "Learn About Sustainability." EPA Website. 10.16.2016. Accessed 3.28.2018. https://www.epa.gov/sustainability/learn-about-sustainability#what

[2]. J. Morelli. "Environmental Sustainability: A Definition for Environmental Professionals." Journal of Environmental Sustainability. 11.2011. Accessed 03.28.2018. http://www.environmentalmanager.org/wp-content/uploads/2011/09/Article2Morelli1.pdf

[3]. A.D. Basiago. "Economic, Social, and Environmental Sustainability in Development Theory and Urban Development Practice." Environmentalist. 06.1998. Accessed 03.28.2018. https://www.amherst.edu/system/files/media/0972/fulltext.pdf

[4]. Bruce Schneir: Applied Cryptography, 2nd edition, John Wiley & Sons, 1996

[5]. Abrams, M., and Podell, H. “Cryptography” Potentials, IEEE Page No 36-38. Issue: 1, Volume: 20, Feb-Mar, 2001

[6]. Eskiciogiu, A. Litwin,L “ Cryptography and Network Security” LOS Alamitos, A: IEEE computer society press,1987

[7]. 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, pp. 58-64, 2009.

[8]. Parikshit Prasad, Badrinath Ojha, Rajeev Ranjan Shahi, Ratan Lal “3-Dimensional Security in Cloud Computing” published in IEEE Xplore, 978-1-61284-840-2/11/$26.00 ©2011, pp. 198-201, 2011.

[9]. N. Kaaniche, E. Moustaine, M. Laurent. “A Novel Zero-Knowledge Scheme for Proof of Data Possession in Cloud Storage Applications.” IEEE. 08.06.2014. Accessed http://ieeexplore.ieee.org.pitt.idm.oclc.org/document/6846488/

[10]. Manaa, M.E.; Hadi, Z.G. Scalable and robust cryptography approach using cloud computing. *J. Discret. Math. Sci. Cryptogr.* **2020**, *23*, 1439–1445.

[11]. Shor PW. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J Comput 1997;5:1484–509.

[12]. Alfred J. Menezes, Scott A. Vanstone, and Paul C. Van Oorschot. Handbook of Applied Cryptography. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 1996.

[13]. Amaresh Patil, Valluripalli Srinath, Sudheer Shetty (2014,February), “Survey on Efficient Secure Storage Authentication in Cloud Storage System” International Journal of Engineering Research & Technology (IJERT) p. 3,4,5

[14]. Ravi Gharshi and Suresha. 2013. Enhancing Security in Cloud Storage using ECC Algorithm. International Journal of Science and Research (IJSR), Volume 2 Issue 7.

[15]. Chester Rebeiro. 2009. Architecture Explorations for Elliptic Curve Cryptography on FPGAS. M.Sc. thesis. Department of Computer Science and Engineering, Indian Institute of Technology, Madras.

[16]. Padma Bh, D.Chandravathi , P.Prapoorna Roja, “Encoding And Decoding of a Message in the Implementation of Elliptic Curve Cryptography using Koblitz‟s Method ”, International Journal on Computer Science and Engineering Vol. 02, No. 05, 2010, 1904-1907

[17]. B. Glas, S. Sander, V. Vitali Stuckert, D. Muller-Glaser, J. Becker, "Prime Field ECDSA Signature Processing for Reconfigurable Embedded Systems," International Journal of Reconfigurable Computing Volume 2011.

[18]. S. Almuhammadi and C. Neuman. Security and privacy using one-round zero-knowledge proofs. In CEC, pages 435{438, 2005.

[19]. Manuel Blum, Paul Feldman, and Silvio Micali. Non-Interactive Zero Knowledge and Its Applications (Extended Abstract) MIT. pages 103-112, 1988.

[20]. M. Girault, G. Poupard, and J. Stern. On the fly authentication and signature schemes based on groups of unknown order. Journal of Cryptology, 19:463–487, 2006.

[21]. Jun LJ, Brandon. Implementing zero-knowledge authentication with zero knowledge. Pyhton Papers Monograph, Proc PyCon Asia-Pacific 2010;2:1–19..

[22].
amenisch JL. Group signature schemes and payment systems based on the discrete logarithm problem, ETH series in information security an cryptography. Germany: Hartung-Gorre-Verlag; 1998.