International Journal of Computer Networks and Applications (IJCNA)
DOI: 10.22247/ijcna/2025/21 Volume 12, Issue 3, May – June (2025)

RESEARCH ARTICLE

Geometric Cryptosystem Based Elliptic Curve Cryptography and Advanced Encryption Standard with Galois Counter Mode for Secure Cloud Storage

Pruthvi Kumar K R
Department of Computer Science and Engineering, BMS Institute of Technology and Management, Visvesvaraya Technological University, Belagavi, India.
✉ pruthvikumar31@gmail.com

Usha B A
Department of Information Science and Engineering, BMS Institute of Technology and Management, Visvesvaraya Technological University, Belagavi, India.
ushaba@bmsit.in

Received: 22 February 2025 / Revised: 16 May 2025 / Accepted: 30 May 2025 / Published: 30 June 2025

Abstract – Cloud Computing (CC) offers data storage facility for data management, maintenance and remote backup. However, the existing methods suffer from high memory and time consumption for encryption and decryption, particularly when handling large datasets, and also struggled to maintain data integrity and prevent unauthorized access. Therefore, geometric cryptosystem based Elliptic Curve Cryptography and Advanced Encryption Standard with Galois/Counter Mode (ECC-AESGCM) is proposed in this research for secure cloud storage. Here, ECC is employed for key generation, while AESGEM is deployed for data encryption and decryption. By combining ECC and AESGCM, a robust and effective security strategy for optimizing memory and time consumption is ensured during encryption and decryption processes. The Secure Hash Algorithm-256 (SHA-256) is employed for hash verification to maintain the integrity and authenticity of cloud data. The ECC point validation is used for similarity checking to detect unauthorized changes, thereby maintaining reliability and consistency in the stored data. The effectiveness of ECC-AESGCM is analyzed based on the metrics of execution time, computation time, encryption time, and decryption time with file sizes being varied between 5KB and 25KB. The proposed ECC-AESGCM consumes 54.83ms and 27.89ms of encryption and decryption time for a file size of 10KB, thereby exhibiting commendable effectiveness than the existing Modified ECC (MECC) technique.

Index Terms – Advanced Encryption Standard, Elliptic Curve Cryptography, Galois/Counter Mode, Geometric Cryptosystem, Point Validation, Secure Cloud Storage.

1. INTRODUCTION

The term ‘Cloud Computing’ (CC) denotes any online service hosted by a third party that enables access computing resources and services over the internet. These services commonly include servers, analytics, databases, networks, tools and software [1, 2]. The rapid advancement of CC has transformed organizational processes and the way data is distributed and stored [3]. Cloud servers offer immense storage abilities and on-demand computational sources, allowing businesses to effectively manage huge data [4]. Users outsource data to the cloud for storage and processing due to its flexibility and scalability. Cloud Service Providers (CSP) ensure data privacy and equip users with data encryption, prior to uploading it into the cloud [5, 6]. CSPs are responsible for handling cloud servers and generating services for users to effectively preserve and handle their data [7]. CC offers a wide range of services such as computing power, applications, and storage over the internet, which allow users to manage and access resources on-demand, without requiring a local infrastructure [8]. The cloud storage system generates flexible and on-demand storage services to businesses or individuals through third parties over the internet. It empowers users with access to data everywhere, thus supporting both pay-per-use and on-demand models [9].

CC enables users to access data and resources from anywhere through the internet, offering three levels of services: Platform as a Service (PaaS), Infrastructure as a Service (IaaS) and Software as a Service (SaaS) [10, 11]. IaaS allows for consistent data storage at lesser cost when compared to PaaS and SaaS [12]. The IaaS minimizes cost with improved availability and technical advancements, widely beneficial in cloud server deployment for computation and storage [13-15]. Cloud servers are preferred for computing-intensive tasks due to their reliability, low cost and high performance which ensure effective operations, with minimal investments [16].

RESEARCH ARTICLE

However, the security of cloud systems remains a primary concern that prompts the development of new techniques with significant security advancements. Current security issues in the cloud include lack of user access control and guaranteed cloud data security [17-19]. Additionally, optimizing memory and time consumption during data encryption and decryption, mainly in huge datasets and real-time processing, remains a critical issue.

Efficiency is crucial in a cloud storage system, alongside high memory usage and processing time, altogether impacting overall system performance and user experience. In order to overcome these problems, this research proposes a robust and secure Geometric Cryptosystem integrating Elliptic Curve Cryptography (ECC) with the Advanced Encryption Standard in Galois/Counter Mode (AES-GCM) based on a geometric cryptosystem for optimizing memory and time consumption during encryption and decryption. The ECC is selected for its ability to provide strong security with shorter key lengths, resulting in robust key generation and lower computational demands. AES-GCM delivers high performance, enabling efficient encryption and decryption, while the Secure Hash Algorithm-256 (SHA-256) ensures data integrity.

1.1. Research Motivation

With the rapid growth of data storage requirements, CC has become a major paradigm that provides flexible, scalable, and cost-effective services. However, the protection of data in the cloud environment remains one of the most pressing challenges, as the level of threats such as information leakage and performance inefficiencies, continues to rise.

Conventional encryption techniques struggle to balance optimal time and memory utilization with effective security for large datasets and real-time processes. Therefore, there is an urgent need for advanced cryptographic methods that ensure both data security and efficient resource utilization.

1.2. Problem Statement

Existing cloud storage encryption approaches, such as MECC and multi-stage methods, suffer from high encryption and decryption time, complex key management, and heavy memory consumption when handling large volumes of data. These shortcomings degrade system performance and negatively impact user experience in real-time cloud environments. Therefore, there is a need to develop a cryptographic procedure that ensures low-latency encryption and decryption, reduced memory usage, and strong data integrity and confidentiality.

1.3. Objectives

Thus, study seeks to build a secure and efficient cryptographic model for cloud storage, with its key objectives specified as follows:

- To build a hybrid cryptosystem that is a synergy of Elliptic Curve Cryptography (ECC) and Advanced Encryption Standard with Galois/Counter Mode (AES-GCM) in order to increase security and time decrease consumption for reduced computational complexity.
- To incorporate an ECC point validation for integrity checks and unauthorized data modification detection.
- To evaluate the proposed ECC-AESGCM method in terms of the measures: encryption time, decryption time, memory consumption, and general execution time.

This study's contributions to research are listed as follows:

- ECC is adopted for key exchange, offering greater efficiency in key generation by requiring shorter key lengths while providing equivalent security compared to traditional methods.
- AES-GCM is utilized to enhance security and efficiency, delivering high performance in symmetric data encryption while ensuring authentication, data integrity, and confidentiality.
- Point validation is implemented to preserve ECC security by ensuring that every point involved in cryptographic operations is valid. ECC point validation also aids in detecting unauthorized modifications, thereby maintaining the reliability and consistency of stored data.

The research manuscript is further organized as follows: Section 2 explains the literature review of existing models, and Section 3 outlines the proposed method. Section 4 discusses the results and performance evaluation in detail, while Section 5 concludes the research.

2. LITERATURE REVIEW

This section discusses notable studies focused on secure data transmission in cloud environments.

Vengala et al. [20] suggested a three-factor authentication system using MECC for data transfer in the cloud environment. Three factors of authentication, data compression and secure data transfer were included in this model, and the SHA-512 with Cued Clock Point (CCP) was used for authentication. Then, user-uploaded data was compressed using the Compliments Hashing Algorithm (CHA) at the server's end. Furthermore, the MECC encrypted data and uploaded it to a cloud server, offering a vigilant security system with smaller key sizes which reduced the memory space required for accessing large datasets in the cloud. However, when dealing with large data in the cloud, the MSA-OBA demanded increased memory requirements during the encryption and decryption processes.

Ahamed and Duraipandian [21] developed a secure data storage setup through ECC deduplication in CC. The Multi-

RESEARCH ARTICLE

Layer Storage (MLS) based security model based on ECC focused on providing secure cloud storage and eliminated replicates at the primary levels. The model operated in three stages, where the initial two stages were saved in the local system, while the third stage of the encrypted data was stored in the cloud. The model required the use of the same device that originally generated the data. However, dual encryption suffered from extensive overhead and resource requirements due to additional computational tasks which increased the model's execution time.

Singh and Jha [22] suggested a security enhancement technique using African Buffalo-based Elapid Crypto Model (AB-ECM) to efficiently handle and protect data during transmission to the receiver. In AB-ECM, a large volume of data from the cloud was reduced through a two-stage technique. During the auditing stage, system security was strengthened, and data was decrypted only when the secret keys of both the sender and receiver matched. However, the model faced issues related to complexity and increased encryption time due to a lack of coordination among multiple points.

Seth et al. [23] introduced a secure data storage system in the cloud by integrating dual encryption techniques, and data fragmentation to secure data distribution in multi-cloud environments. It offered an improved degree of security for data outsourcing in a cloud environment when connecting multiple independent service providers. However, this approach suffered from time complexity due to point multiplication leading to high execution times.

Shyla and Sujatha [24] presented an effective secure data retrieval model using Multi-Stage Authentication (MSA) and an Optimized Blowfish Algorithm (OBA). To enhance system security, an optimal key value was selected through Binary Crow Search Algorithm (BCSA). The MSA-OBA consisted of three stages: MSA, data security and retrieval. Initially, cloud users registered data on the cloud through MSA, after which the data was encrypted by the OBA. MSA-based data retrieval was performed following the encryption process. However, the model suffered from limited adaptability and scalability due to its complex structure, which negatively affected overall performance.

Ahmad and Mehfuz [25] suggested an efficient Time-oriented Latency Approximation-Based Data Encryption (TLADE) for cloud storage. This method focused on selecting an optimal encryption technique at different times based on latency approximation. In this method, an optimal approach was selected for date encryption based on latency approximation, following which an optimal technique was selected for encryption based on the Quality of Service (QoS) values. To enable different encryption methods, each technique was evaluated based on its latency-related QoS Support Values (QoSV).

Yi et al. [26] developed a Deterministic Storage and Deletion Mechanism for Trusted Cloud Service Environments (DSDM-TCSE). This approach provided a three-layer cloud data framework and adopted blockchain as the transmission layer, enabling techniques such as key negotiation and encryption. It supported fine-grained storage, deletion control, and verification of cloud data to protect data privacy. The method was effective through the use of data noise vectors and a cuckoo filter, which facilitated faster data construction and verification. It offered system scalability and modularity, enabling efficient data management across layers. However, the integration of blockchain and a multi-layered structure was computationally expensive, requiring a higher level of expertise.

From the overall analysis, it is noted the abovementioned models have the following limitations, as identified and addressed in this research. The existing techniques are affected by encryption time complexity due to a lack in coordination between multiple points. The previous models suffer from high time complexity due to point multiplication, leading to higher execution time. Some models also suffered from restricted scalability and adaptability due to complex structures, which affected the model's overall performance. Dual encryption required extensive resources and overhead due to additional computational tasks, which further increased execution time. When handling large data, the MSA-OBA method also increased memory usage during encryption and decryption. To overcome these limitations, this research proposes an ECC-AES-GCM-based geometric cryptosystem for securing cloud data. Table 1 summarizes the existing research models in a concise manner.

Table 1 Summary of the Existing Research

Author(s) & Year	Method	Advantages	Limitations
Vengala et al. [20] (2023)	MECC with 3FA and CCP-SHA512-CHA	Reduced memory space required for accessing large datasets in the cloud	Demanded increased memory usage during encryption and decryption
Ahamed and Duraipandian [21] (2022)	ECC with Multi-layer Deduplication	Required the same device that generated the data for efficient processing	Required extensive overhead and resources due to additional computational tasks which increased the model's execution time.

RESEARCH ARTICLE

Singh and Jha [22] (2023)	AB-ECM with auditing	Data was decrypted when the secret key of both the sender and receiver auditing stages were similar.	Affected by time and overall model complexity due to the lack of coordination among multiple points.
Seth et al. [23] (2022)	Dual encryption and data fragmentation	Improved degree of security for data outsourcing in cloud when connecting many independent service providers.	High time complexity due to point multiplication and high execution time.
Shyla and Sujatha [24] (2022)	MSA with Optimized Blowfish & BCSA	Data retrieval was accomplished after performing encryption, thereby enhancing security and key selection.	Suffered from limited adaptability and scalability due to the model's complex structure which affected the overall performance.
Ahmad and Mehfuz [25] (2024)	TLADE	Optimized model selection based on latency.	Limited validation among varied cloud scenarios, thereby affecting model generalizability.
Yi et al. [26] (2025)	DSDM-TCSE	System scalability and modularity enabled effective data management among layers.	Incorporating blockchain and multi-layered structure made the model computationally expensive, thereby requiring higher-level expertise.

3. PROPOSED METHOD

A detailed explanation of the proposed geometric cryptosystem-based ECC-AESGCM for secure data sharing is presented in this section.

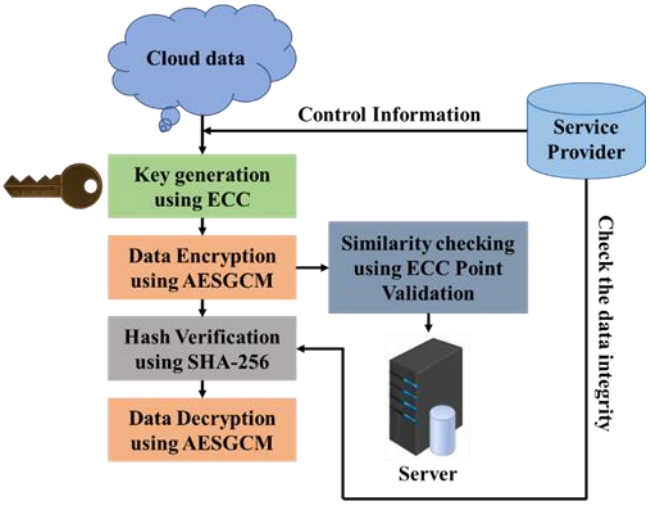

Figure 1 Flow Diagram of the Proposed Method

Primarily, the user produces a key using ECC, while the data encrypted and decrypted using AESGCM, after which hash verification is performed through SHA-256. The encrypted data is stored in the server for similarity checking which is performed by using ECC point validation technique. If the data is same as the encrypted data, then a hash function is produced for specific data; else, data is stored in the cloud. The developed approach is used for minimizing storage space in the cloud. By incorporating ECC and AESGCM, a more robust and effective security for optimizing memory and time consumption in encryption and decryption process is worked at. Figure 1 represents the data security and integrity verification process.

3.1. System Model

The cloud customer transfers control information to the service provider, including cloud data and instructions for managing secure data. Using a generated key, the cloud customer's data is encrypted and passed on for similarity checking to validate its integrity. The server then stores the encrypted and validated data. Additionally, the server manages data access requests and ensures data security. During data retrieval, hash verification calculates the retrieval data's hash value. The hash value is then compared with the actual hash value to ensure the data has not been altered. If the hash verification is successfully completed, the data is decrypted using a decryption algorithm.

3.2. Elliptic Curve Cryptography (ECC)

ECC is a key-based technique for encrypting data, focused on private and public key pairs for generating keys. In the key

RESEARCH ARTICLE

generation, the receiver decrypts data using the private key, while the sender encrypts it using the public key. ECC is based on an elliptic curve theory which is utilized to create smaller, faster and efficient cryptographic keys. The elliptic curve is expressed as given in equation (1).

$$y^2 = x^3 + ax + b \quad (1)$$

Where, x and y are coordinates of curve points, and a and b are constants. The elliptic curve points have special properties, and it is possible for them to obtain double of a point by adding the point itself to a certain point P. This process is repeated to achieve arbitrary multiples of n of this point, denoted as nP. As the addition process continues, it eventually returns to the initial point after m iterations. When P is multiplied by n to obtain a point X, it is easier to define X based on P and n. While the public key is known already, the private key is produced through n and public key is produced using P and X. The ECC delivers better efficiency in key generation, requiring short key lengths while providing comparable security to traditional methods. ECC point validation is used to check for similarities and detect unauthorized changes, thereby maintaining data reliability and consistency.

3.3. Encryption and Decryption Using AESGCM

The AES is a symmetric block encryption method that is obtained through a replacement variation network. It utilizes a similar key for both encryption and decryption. The size of the cipher key in the AES block is 128 bits which is enhanced to 192 and 256 bits based on the application. The key length and rounds are provided as 10 for 128 bits, 12 for 192 bits and 14 for 256 bits, in accordance with the to block size. Generally, 128 bits are organized into a matrix of 4×4. The AES process is separated into 4-byte substitution, mix columns, shift rows and an added round key. At first, an individual byte is swapped through an innovative byte, while the plain text scores are swapped through replacement box scores. The shift operation is a transformation procedure where data is moved to the left end in a progressive rule for every portion. In case of a shift operation being executed for the primary row, then one block is moved to the left. If the shift is executed in the second row, then dual blocks are moved to the left. Likewise, this procedure is continued until n th row elements are moved into n times in the shift operation. Following this, the columns are integrated into the next stages. The first matrix of the first column integrates with the initial column in the second matrix, and is repeated until completed for all columns. After attaining a mixed column operation, the add-around key is integrated through matrix scores. The XOR operation is accomplished in the addition procedure where XORs 16-byte extend keys and plain text. After the XOR operation, the resulting matrix represents the encrypted data.

The GCM is a block-cipher configuration mode that integrates encryption and authentication effectively. It is based on dual essential operations of counter mode encryption and Galois field multiplication. The GCM utilizes universal hashing with the benefits of high speed, low cost and latency. The table-driven field operations help in software implementation and theoretic establishment of block ciphers, which generate logical presumptions on the block cipher. During encryption and decryption, the counter mode is applied to process input data. A tag of up to 128 bits is generated in the authentication process by hashing the data using a Galois field accumulator. This tag is then used to validate the integrity of the data. Decryption is permitted only if the generated tag matches the actual tag; otherwise, authentication fails, and no data is decrypted from the database. The GCM is defined as a function of the secret key k_s, initial vector v_{ini}, plain text p_t, and addition authentication data Ad_{data}, as given in equation (2).

$$(T, c) = GCM(k_s, v_{ini}, p_t, Ad_{data}) \quad (2)$$

The entire function is defined by the parameters T and c, where T denotes the authenticated tag utilized in decryption, which validates the authenticity of the encrypted information. The equations for GCM encryption and authentication are provided in equations (3) to (5).

$$y_{i+1} = inc(y_i) \qquad for\ i = 1,2,\dots,n \quad (3)$$

$$c_i = p_i \oplus e(k_s, y_i) \qquad for\ i = 1,2,\dots,n \quad (4)$$

$$T = MSB_t\left(G_{hash}(H, A, C) \oplus e(k_s, y_0)\right)\ for\ i = 1,2,\dots,n \quad (5)$$

Where, y is a counter block, $inc(y_i)$ is a function that increments the counter value, c_i is a cipher text, and $\oplus$ is a XOR operation. $e(k_s, y_i)$ is an encryption function of AES counter mode, MSB_t is the most significant bit used for hashing function, T is an authentication tag, and G_{hash} is a hashing function used for authentication. The hashing algorithm is stated in equation (6).

$$x_{i+1} = (input + x_i) \times H \quad (6)$$

Where, x_i is an intermediate hash value and H is a hash subkey. The hashing function uses a 128-bit XOR and Galois multiplication as response to limit G_{hash} functions. The Galois multiplication procedure is varied from traditional base 10 addition and multiplication. In Galois addition or multiplication, carry-less addition and multiplication are used, which do not increase the bit size. The system must be initialized when an innovative key and initialization vector are established. Therefore, it is necessary to generate a hash key derived from the output of AES encryption. Once initialization is complete, the system is ready to process encryption inputs. AES-GCM ensures both security and efficiency, offering higher performance in symmetric data

RESEARCH ARTICLE

encryption with authentication, as well as data integrity and confidentiality, compared to traditional methods.

3.4. Hash Verification

The SHA-256 is a cryptographic hash algorithm that generates a 256-bit (32byte) hash value which is used for hash verification to maintain the integrity and authenticity of cloud data. It is intended to generate various hash if a single character in input data changes. It helps identify any alterations made to the actual data. The user verifies every single letter to verify if it has changed by comparing it to the actual hash digests as the digits differ. It considers plain text input length L and produces an output of fixed length $n = 32$ bytes hash scores. Additionally, it takes input message M of the highest length 32 bytes [27]. The M is padded through ρbit to create the input length L, in which the last 64 bits are preserved for actual input M. The padded bits are initiated with 1 based on the essential numbers. The last input of SHA-256 is at a key length separated into 512 same-size blocks of 512 bits. This model is an aggregation of three cryptographic methods: Elliptic Curve Cryptography (ECC) for safe and efficient key generation, Advanced Encryption Standard in Galois/Counter Mode (AES-GCM) for rapid and verified encryption, and the Secure Hash Algorithm-256 (SHA-256) for integrity verification. ECC is chosen due to its potential to offer high security with reduced key sizes, resulting in lower computational power requirements and memory usage compared to traditional techniques such as RSA. AES-GCM provides both cryptographic services in a single efficient operation and supports parallel processing, which offers significantly higher throughput and reduced latency. SHA-256 ensures a reliable integrity check by generating deterministic, collision-resistant hashes capable of detecting even slight changes in data. The geometric cryptosystem is implemented using a modular and streamlined data flow, minimizing computational overhead. The pseudocode 1 of the proposed method is given as follows for reproducibility.

$\left(data, user_{public_{key}}, user_{private_{key}}\right)$:

Step 1: ECC Key Generation

$shared_{secret} = ECC_{Key}{}_{Generation\left(user_{public_{key}}, user_{private_{key}}\right)}$

$AES_{key} = Derive_{AES_{Key(shared_{secret})}}$

Step 2: AES-GCM Encryption

$iv = Generate_{IV}$

$aad = Prepare_{Additional_{Authenticated_{Data}}}$

$enc_{data}, auth_{tag} = AES_{GCM}{}_{Encrypt\left(data, aes_{key}, iv, aad\right)}$

Step 3: SHA-256 Hashing for Integrity

$original_{hash} = SHA256_{Hash(data)}$

Step 4: ECC Point Validation for Similarity Check

$valid_{point} = ECC_{Point}{}_{Validation\left(enc_{data}, user_{public_{key}}\right)}$

if $valid_{point} == True$:

$Upload_{To}{}_{Cloud\left(enc_{data}, auth_{tag}, iv, original_{hash}\right)}$

else:

Raise Error ("Invalid ECC point, Upload aborted")

return data securely encrypted and uploaded

Function retrieve and decrypt data $\left(\begin{matrix} enc_{data}, auth_{tag}, iv, original_{hash}, user_{private_{key}}, \\ sender_{public_{key}} \end{matrix}\right)$

Step 1: ECC Key Regeneration

$shared_{secret} = ECC_{Key}{}_{Generation\left(sender_{public_{key}}, user_{private_{key}}\right)}$

$aes_{key} = Derive_{AES_{Key(shared_{secret})}}$

Step 2: AES-GCM Decryption

$decrypted_{data} = AES_{GCM}{}_{Decrypt\left(enc_{data}, aes_{key}, iv, auth_{tag}\right)}$

Step 3: Verify Integrity using SHA-256

$retrieved_{hash} = SHA256_{Hash(decrypted_{data})}$

if $retrieved_{hash} = original_{hash}$:

return $decrypted_{data}$

else:

Raise Error ("Data integrity check failed")

Pseudocode 1 Function Secure Cloud Storage Process

4. EXPERIMENTAL RESULTS

The effectiveness of the geometric cryptosystem based ECC-AESGCM is simulated in Python 3.7 with the system configuration encompassing 8GB RAM, an Intel i5 processor, and Windows 10 OS.

The execution time, computation time, encryption time, and decryption time are considered as performance metrics for analyzing the ECC-AESGCM model's effectiveness. The proposed ECC-AESGCM provides a robust and secure ecosystem for optimizing memory and time consumption for both encryption and decryption.

The ECC point validation is employed for detecting unauthorized changes, thereby maintaining reliability and consistency of the stored data.

RESEARCH ARTICLE

4.1. Performance Analysis

The effectiveness of ECC-AESGCM is analyzed based on the execution time, computation time, encryption time, and decryption time. The various file sizes of 5KB, 10KB, 15KB, 20KB and 25KB are considered to evaluate the effectiveness of the proposed ECC-AESGCM. Table 2 and Table 3 present the effectiveness of the proposed ECC-AESGCM for execution time and computation time with different file sizes. Figures 2, 3, 4 and 5 illustrate the effectiveness of ECC-AESGCM in terms of encryption time, memory usage for encryption, decryption, and memory usage during decryption.

Table 2 presents the execution time (ms) for different cryptographic methods with varying file sizes of 5KB, 10KB, 15KB, 20KB and 25KB.

The existing cryptographic methods such as RSA (Rivest-Shamir-Adleman), DH (Diffie-Hellman), DSA (Digital Signature Algorithm), RC4 (Rivest Cipher 4), and ECC are compared with the proposed ECC-AESGCM. The proposed ECC-AESGCM demands an execution time of 108.81ms, 109.70ms, 108.71ms, 107.71ms, and 108.71ms for the file sizes of 5KB, 10KB, 15KB, 20KB and 25KB, respectively. Table 3 presents the assessment of computation time (ms) for different cryptographic methods for the file sizes of 5KB, 10KB, 15KB, 20KB and 25KB. The different cryptographic methods such as RSA, DH, DSA, RC4 and ECC are compared with the proposed ECC-AESGCM. The proposed ECC-AESGCM requires a computation time of 79.76ms, 82.75ms, 81.77ms, 81.78ms, and 81.77ms, correspondingly for file sizes being 5KB, 10KB, 15KB, 20KB and 25KB. Table 4 presents the performance evaluation outcomes of the cryptographic models in terms of compression time (ms) for varying file sizes of 5KB, 10KB, 15KB, 20KB and 25KB. The existing cryptographic methods such as RSA, DH, DSA, RC4 and ECC are compared with the proposed ECC-AESGCM. The proposed ECC-AESGCM demands a compression time of 133.645ms, 273.592ms, 388.472ms, 520.234ms, and 649.781ms, correspondingly for the file sizes of 5KB, 10KB, 15KB, 20KB and 25KB.

Table 2 Execution Time (ms) with Different File Size

Method	File Size (KB)				
	5	10	15	20	25
RSA	230.71	237.77	253.24	225.07	294.19
DH	185.92	203.93	196.10	198.10	189.47
DSA	207.03	232.05	208.46	215.16	224.45
RC4	252.84	258.45	255.92	253.06	258.14
ECC	215.71	215.86	207.76	206.71	208.73
ECC-AESGCM	108.81	109.70	108.71	107.71	108.71

Table 3 Computation Time (ms) with Different File Size

Method	File Size (KB)				
	5	10	15	20	25
RSA	109.57	102.13	107.52	106.45	102.57
DH	123.56	129.35	132.76	122.78	127.86
DSA	115.65	119.25	121.25	118.56	119.56
RC4	131.25	134.12	135.42	137.24	133.33
ECC	232.73	234.81	235.85	237.28	239.18
ECC-AESGCM	79.76	82.75	81.77	81.78	81.77

Table 5 presents a performance evaluation in terms of compression rate (%) for different cryptographic methods for various file sizes 5KB, 10KB, 15KB, 20KB and 25KB. The different cryptographic methods such as RSA, DH, DSA, RC4 and ECC are compared with the proposed ECC-AESGCM. The proposed ECC-AESGCM acquires a compression rate of 20.5%, 24.8%, 29.1%, 33.4% and 37.5%, correspondingly for the file sizes of 5KB, 10KB, 15KB, 20KB and 25KB.

Table 6 presents the compressed file size (in bytes) for different cryptographic methods across various file sizes:

RESEARCH ARTICLE

5KB, 10KB, 15KB, 20KB, and 25KB. The cryptographic methods compared include RSA, DH, DSA, RC4, and ECC, with the proposed ECC-AESGCM. The proposed ECC-AESGCM results in the following compressed file sizes: 5,678,321 bytes, 10,876,543 bytes, 15,987,654 bytes, 21,123,456 bytes, and 26,345,678 bytes, corresponding to the file sizes of 5KB, 10KB, 15KB, 20KB, and 25KB, respectively.

Table 4 Compression Time (ms) with Different File Size

Method	File Size (KB)				
	5	10	15	20	25
RSA	144.321	291.456	423.987	564.123	704.563
DH	145.762	300.472	419.238	568.943	709.654
DSA	139.849	290.576	420.193	559.462	711.21
RC4	150.178	304.903	429.345	572.789	726.314
ECC	156.487	301.918	420.876	568.309	714.231
ECC-AESGCM	133.645	273.592	388.472	520.234	649.781

Table 5 Compression Rate (%) with Different File Size

Method	File Size (KB)				
	5	10	15	20	25
RSA	12.34	15.67	18.92	22.11	25.47
DH	13.5	16.2	19.6	23	26.5
DSA	11.9	14.8	17.95	21.15	24.3
RC4	13	16.5	19.8	23.2	26.7
ECC	12.75	15.85	18.95	22.3	25.55
ECC-AESGCM	20.5	24.8	29.1	33.4	37.5

Table 6 Compressed File Size (bytes) with Different File Size

Method	File Size (KB)				
	5	10	15	20	25
RSA	5,244,616	10,489,096	15,733,576	20,978,056	26,222,536
DH	5,345,121	10,567,982	15,890,034	21,112,788	26,336,872
DSA	5,123,789	10,345,876	15,678,902	20,890,453	26,123,567
RC4	5,389,457	10,678,432	15,890,754	21,123,900	26,345,678
ECC	5,278,904	10,564,317	15,890,211	21,112,456	26,334,987
ECC-AESGCM	5,678,321	10,876,543	15,987,654	21,123,456	26,345,678

Table 7 presents the file upload time (in milliseconds) for different cryptographic methods across various file sizes: 5KB, 10KB, 15KB, 20KB, and 25KB. The cryptographic methods compared include RSA, DH, DSA, RC4, and ECC, with the proposed ECC-AESGCM. The proposed ECC-AESGCM achieves the following file upload times: 58.816 ms, 63.419 ms, 63.427 ms, 63.399 ms, and 62.835 ms, corresponding to the file sizes of 5KB, 10KB, 15KB, 20KB, and 25KB, respectively.

Table 8 presents the file download time (in milliseconds) for different cryptographic methods across various file sizes: 5KB, 10KB, 15KB, 20KB, and 25KB. The cryptographic methods compared include RSA, DH, DSA, RC4, and ECC, with the proposed ECC-AESGCM. The proposed ECC-

RESEARCH ARTICLE

AESGCM results in the following file download times: 63.439 ms, 63.859 ms, 62.862 ms, 63.424 ms, and 63.374 ms, corresponding to the file sizes of 5KB, 10KB, 15KB, 20KB, and 25KB, respectively.

Table 7 File Upload Size (ms) with Different File Size

Method	File Size (KB)				
	5	10	15	20	25
RSA	60.125	65.213	67.345	69.456	70.678
DH	62.145	66.478	68.329	70.234	72.451
DSA	61.897	64.752	66.892	68.456	69.982
RC4	63.326	67.014	68.876	71.129	73.247
ECC	59.89	64.123	66.245	68.435	70.547
ECC-AESGCM	58.816	63.419	63.427	63.399	62.835

Table 8 File Upload Size (ms) with Different File Size

Method	File Size (KB)				
	5	10	15	20	25
RSA	68.239	69.52	71.345	72.678	74.299
DH	67.435	68.93	70.456	71.789	73.234
DSA	65.89	67.123	68.567	69.945	71.678
RC4	66.732	68.459	70.123	71.89	73.678
ECC	64.89	66.234	67.789	69.345	70.89
ECC-AESGCM	63.439	63.859	62.862	63.424	63.374

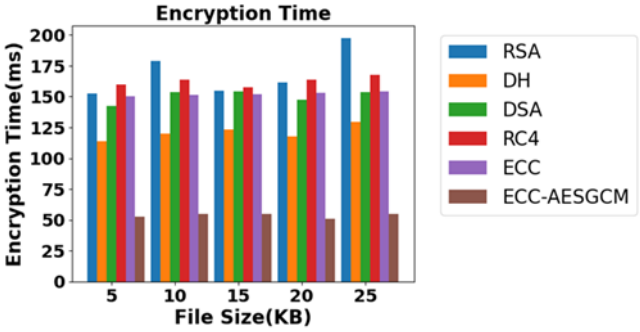

Figure 2 Encryption Time (ms) with Different File Sizes

Figure 2 presents the encryption time (in milliseconds) for different cryptographic methods across various file sizes: 5KB, 10KB, 15KB, 20KB, and 25KB. The cryptographic methods compared include RSA, DH, DSA, RC4, and ECC, with the proposed ECC-AESGCM. The proposed ECC-AESGCM achieves the following encryption times: 52.83 ms, 54.85 ms, 54.88 ms, 50.86 ms, and 54.85 ms, corresponding to the file sizes of 5KB, 10KB, 15KB, 20KB, and 25KB, respectively.

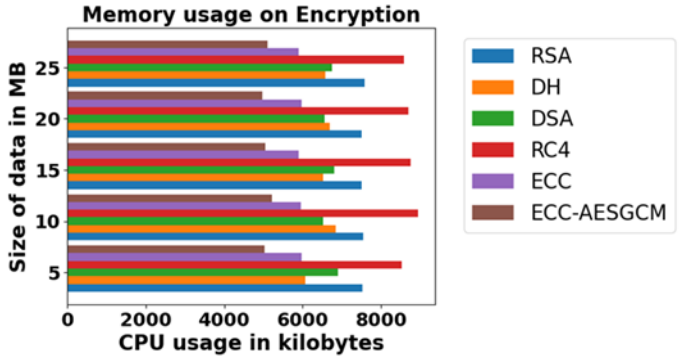

Figure 3 Memory Usage for Encryption with CPU Usage

Figure 3 presents evaluation outcomes in terms of memory usage for encryption and CPU usage for different cryptographic methods across various data sizes (MB). The cryptographic methods compared include RSA, DH, DSA, RC4, and ECC, with the proposed ECC-AESGCM. The proposed ECC-AESGCM demonstrates lower memory usage for encryption compared to the existing techniques. The ECC-AESGCM requires the following memory for encryption:

RESEARCH ARTICLE

5023 KB, 5220 KB, 5050 KB, 4980 KB, and 5100 KB for file sizes of 5KB, 10KB, 15KB, 20KB, and 25KB, respectively.

Figure 4 depicts the decryption times (ms) for different cryptographic methods across various file sizes: 5KB, 10KB, 15KB, 20KB, and 25KB. The cryptographic methods, including RSA, DH, DSA, RC4, and ECC, are compared with the proposed ECC-AESGCM. The proposed ECC-AESGCM requires decryption times of 26.93 ms, 27.89 ms, 26.92 ms, 26.89 ms, and 26.92 ms for file sizes of 5KB, 10KB, 15KB, 20KB, and 25KB, respectively.

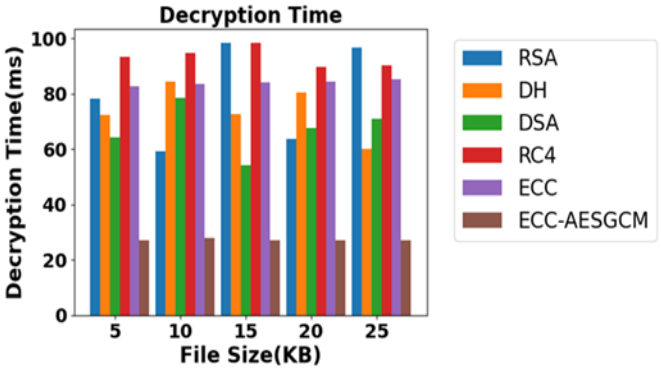

Figure 4 Decryption Time (ms) with Different File Sizes

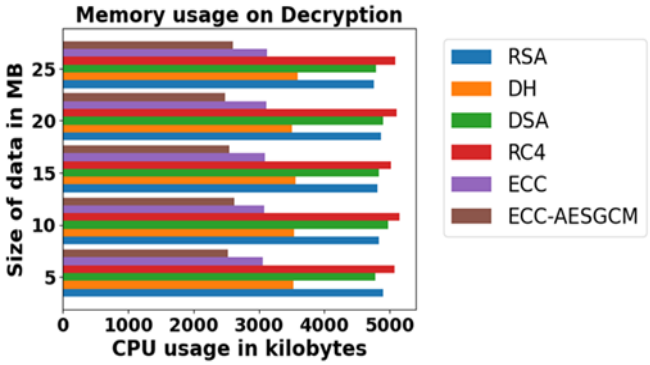

Figure 5 Memory Usage for Decryption with CPU Usage

Figure 5 presents the model's memory usage for decryption with CPU usage for different cryptographic methods among various sizes of data (MB). The different cryptographic methods: RSA, DH, DSA, RC4 and ECC are studied against the proposed ECC-AESGCM.

The proposed ECC-AESGCM requires less memory for decryption compared to the existing techniques. The ECC-AESGCM requires 2523 KB, 2620 KB, 2550 KB, 2480 KB, and 2600 KB of memory for decryption for file sizes of 5KB, 10KB, 15KB, 20KB, and 25KB, simultaneously.

4.2. Comparative Analysis

In this section, the proposed ECC-AESGCM is based on a geometric cryptosystem which is comparatively analyzed with the existing approach, MECC [20]. The performance metrics of encryption time (ms) and decryption time (ms) are analyzed with various file sizes 5-25KB to analyze the effectiveness of the proposed ECC-AESGCM, as shown in Table 9. The proposed ECC-AESGCM requires encryption times of 52.83 ms, 54.85 ms, 54.88 ms, 50.86 ms, and 54.85 ms for the file sizes of 5KB, 10KB, 15KB, 20KB, and 25KB, respectively. The proposed ECC-AESGCM also requires decryption times of 26.93 ms, 27.89 ms, 26.92 ms, 26.89 ms, and 26.92 ms for the file sizes of 5KB, 10KB, 15KB, 20KB, and 25KB, respectively.

4.3. Discussion

The achievements of the proposed ECC-AESGCM model are based on several critical design decisions that lead to reduced computation time, encryption and decryption times, and improved memory efficiency. ECC provides the same level of security as RSA and other traditional methods, but with much smaller key sizes. This results in faster key generation, reduced computational overheads, and lower memory consumption.

The fact that shorter keys are more suitable for processing large volumes of data is particularly important, as they facilitate encryption and decryption procedures. AES-GCM ensures both confidentiality and integrity in a single, efficient operation. GCM supports parallel processing and eliminates expensive block cipher chaining, leading to high throughput. It also speeds up encryption while maintaining data integrity through authenticated tags, reducing the need for separate integrity checks.

The geometric cryptosystem framework combines ECC and AES-GCM securely in a modular and streamlined data path, minimizing redundant operations and keeping data transformation to a minimum. These algorithms enhance the speed and responsiveness of the system, most notably in the reduction of encryption and decryption times as file sizes increase from 5KB to 25KB, especially for smaller files. By incorporating ECC point validation, the system rapidly detects unauthorized changes to stored data. This feature improves security while maintaining a reasonable computational load, which is essential for reliability and optimizing the validation process.

The use of SHA-256 guarantees that data integrity is quickly and reliably verified. Its deterministic output, which is highly sensitive, can immediately detect data tampering, enhancing the model's robustness against state changes and minimizing verification overhead. Compared to MECC and other conventional methods such as RSA, DSA, and RC4, the ECC-AESGCM model reduces encryption and decryption times, demonstrating its computational efficiency. Memory optimization is also achieved, making the system scalable and suitable for real-time cloud applications.

RESEARCH ARTICLE

Table 9 Comparative Analysis of ECC-AESGCM with MECC

Performance Metrics	Methods	File Size (KB)				
		5	10	15	20	25
Encryption time (ms)	MECC [20]	502	1312	2147	2689	4122
	Proposed ECC-AESGCM	52.83	54.85	54.88	50.86	54.85
Decryption time (ms)	MECC [20]	821	1123	2114	2885	4122
	Proposed ECC-AESGCM	26.93	27.89	26.92	26.89	26.92

5. CONCLUSION

The ECC-AESGCM is proposed in this research based on a geometric cryptosystem for optimizing memory and time consumption in the encryption and decryption processes. The ECC provides superior effectiveness in key generation and requires shorter key length, while offering the same level of security. The AES-GCM is applied to enhance efficiency and security, providing higher performance in symmetric data encryption with authentication, ensuring both the confidentiality and integrity of cloud data. ECC point validation is used to detect unauthorized changes, thereby maintaining the consistency and reliability of data. The performance of ECC-AESGCM is evaluated through execution time, computation time, encryption time, and decryption time across various file sizes ranging from 5KB to 25KB. The proposed ECC-AESGCM achieves 109.70ms, 82.75ms, 273.592ms, 63.419ms, and 63.859ms for execution time, computation time, compression time, file upload time, and file download time, respectively, for a file size of 10KB. In the future, different encryption algorithms can be considered to analyze file sizes ranging from 100MB to 1000MB, further enhancing the security of cloud data.

REFERENCES

[1] B. R. Rao and B. Sujatha, "A hybrid elliptic curve cryptography (HECC) technique for fast encryption of data for public cloud security," Measurement: Sensors, vol. 29, p. 100870, 2023.

[2] M. A. Hossain and M. A. Al Hasan, "Improving cloud data security through hybrid verification technique based on biometrics and encryption system," International Journal of Computers and Applications, vol. 44, no. 5, pp. 455–464, 2022.

[3] A. E. Adeniyi, K. M. Abiodun, J. B. Awotunde, M. Olagunju, O. S. Ojo, and N. P. Edet, "Implementation of a block cipher algorithm for medical information security on cloud environment: using modified advanced encryption standard approach," Multimedia Tools Applications, vol. 82, no. 13, pp. 20537–20551, 2023.

[4] M. Jan, Q. Shahzad, and S. Afsar, "Securing the Cloud Storage by Using Different Algorithms of Cryptography," International Journal of Scientific Research in Computer Science and Engineering, vol, 10, no. 2, pp.38-45, 2022.

[5] M. I. Reddy, P. V. Rao, T. S. Kumar, and S. R. K, "Encryption with access policy and cloud data selection for secure and energy-efficient cloud computing," Multimedia Tools Applications, vol. 83, no. 6, pp. 15649–15675, 2023.

[6] K. S. K. Maathavan and S. Venkatraman, "A Secure Encrypted Classified Electronic Healthcare Data for Public Cloud Environment," Intelligent Automation & Soft Computing, vol. 32, no. 2, pp. 765–779, 2022.

[7] R. Bhat, N. R. Sunitha, and S. S. Iyengar, "A probabilistic public key encryption switching scheme for secure cloud storage," International Journal of Information Technology, vol. 15, no. 2, pp. 675–690, 2023.

[8] R. Priyadarshini, A. Quadir Md, N. Rajendran, V. Neelanarayanan, and H. Sabireen, "An enhanced encryption-based security framework in the CPS Cloud," Journal of Cloud Computing, vol. 11, no. 1, p. 64, 2022.

[9] A. Mohiyuddin, A. R. Javed, C. Chakraborty, M. Rizwan, M. Shabbir, and J. Nebhen, "Secure Cloud Storage for Medical IoT Data using Adaptive Neuro-Fuzzy Inference System," International Journal of Fuzzy Systems, vol. 24, no. 2, pp. 1203–1215, 2022.

[10] J. S. Jayaprakash, K. Balasubramanian, R. Sulaiman, M. Kamrul Hasan, B. D. Parameshachari, and C. Iwendi, "Cloud Data Encryption and Authentication Based on Enhanced Merkle Hash Tree Method," Computers, Materials & Continua, vol. 72, no. 1, pp. 519–534, 2022.

[11] P. Sharma, R. Jindal, and M. D. Borah, "Blockchain-based cloud storage system with CP-ABE-based access control and revocation process," Journal of Supercomputing, vol. 78, no. 6, pp. 7700–7728, 2022.

[12] A. E. Sunday, and O. E Olufunminiyi, "An Efficient Data Protection for Cloud Storage through Encryption," International Journal of Advanced Networking and Applications, vol. 14, no. 5, pp.5609-5618, 2023.

[13] M. Suganya and T. Sasipraba, "Stochastic Gradient Descent long short-term memory based secure encryption algorithm for cloud data storage and retrieval in cloud computing environment," Journal of Supercomputing, vol. 12, no. 1, p. 74, 2023.

[14] U. Narayanan, V. Paul, and S. Joseph, "Decentralized blockchain based authentication for secure data sharing in Cloud-IoT: DeBlock-Sec," Journal of Ambient Intelligence and Humanized Computing, vol. 13, no. 2, pp. 769–787, 2022.

[15] P. Yan, S. Nan, X. Feng, Y. Wu, J. Yang, and H. Zhang, "Compression and Cryptography Algorithms for 3D Remote Sensing Point Cloud Data Based on 3D-TCICM and ECC," IEEE Photonics Journal, vol. 15, no. 5, pp. 1–23, 2023.

[16] A. A Parveen and P. S. S. Akilashri, "Secured Authentication Using ECC Based Fractal Fuzzy in Cloud," International Journal of Intelligent Systems and Applications in Engineering, vol. 12, no. 17s, pp.184-194, 2024.

[17] S. Kautish, S. Juneja, K. Mohiuddin, F. K., Karim, H. Elmannai, S. Ghorashi, and Y. Hamid, "Enhanced Cloud Storage Encryption Standard for Security in Distributed Environments," Electronics, vol. 12, no. 3, p. 714, 2023.

[18] G. Swetha and K. Janaki, "Cloud based secure multimedia medical data using optimized convolutional neural network and cryptography mechanism," Multimedia Tools Applications, vol. 81, no. 23, pp. 33971–34007, 2022.

[19] F. Sajid, M. A. Hassan, A. A. Khan, M. Rizwan, N. Kryvinska, K. Vincent, and I. U. Khan, "Secure and Efficient Data Storage Operations by Using Intelligent Classification Technique and RSA Algorithm in IoT-Based Cloud Computing," Scientific Programming, vol. 2022, pp. 1–10, 2022.

RESEARCH ARTICLE

[20] D. V. K. Vengala, D. Kavitha, and A. P. S. Kumar, "Three factor authentication system with modified ECC based secured data transfer: untrusted cloud environment," Complex & Intelligent Systems, vol. 9, no. 3, pp. 2915–2928, 2023.
[21] N. Niyaz Ahamed and N. Duraipandian, "Secured Data Storage Using Deduplication in Cloud Computing Based on Elliptic Curve Cryptography," Computer Systems Science and Engineering, vol. 41, no. 1, pp. 83–94, 2022.
[22] K. K. Singh and V. K. Jha, "Security enhancement of the cloud paradigm using a novel optimized crypto mechanism," Multimedia Tools Applications, vol. 82, no. 11, pp. 15983–16007, 2023.
[23] B. Seth, S. Dalal, V. Jaglan, D. Le, S. Mohan, and G. Srivastava, "Integrating encryption techniques for secure data storage in the cloud," Transactions on Emerging Telecommunications Technologies, vol. 33, no. 4, p. e4108, 2022.
[24] S. I. Shyla and S. S. Sujatha, "Efficient secure data retrieval on cloud using multi-stage authentication and optimized blowfish algorithm," Journal of Ambient Intelligence and Humanized Computing, vol. 13, no. 1, pp. 151–163, 2022.
[25] S. Ahmad and S. Mehfuz, "Efficient time-oriented latency-based secure data encryption for cloud storage," Cyber Security and Applications, vol. 2, p. 100027, 2024.
[26] W. Yi, C. Wang, J. Chen, S. Kuzmin, I. Gerasimov and X. Cheng, "DSDM-TCSE: Deterministic storage and deletion mechanism for trusted cloud service environments," Future Generation Computer Systems, vol. 165, p. 107611, 2025.
[27] R. R. Suman, B. Mondal, and T. Mandal, "A secure encryption scheme using a Composite Logistic Sine Map (CLSM) and SHA-256," Multimedia Tools and Applications, vol. 81, no. 19, pp. 27089–27110, 2022.

Authors

Pruthvi Kumar K R is a highly proficient and experienced professional with an unwavering passion for technological advancement and personal growth. With a solid background in Artificial Intelligence and Machine Learning, AWS Cloud Computing Basics, and Python syntax and semantics, Pruthvi Kumar has proven himself to be a valuable asset to any organization. His proficiency in AWS Academy Machine Learning Foundations and Associate level SAN Certification from EMC2 speaks volumes of his expertise in these areas. Pruthvi Kumar's long-standing association with Karnataka State Council for Science and Technology (KSCST) bearing the project title 'WAAS', showcases his exceptional leadership and technical skills. He attended various workshops and training programs and holds a Master's degree in Digital Communication & Networking and a Bachelor's degree in Information Science & Engineering. Currently, Pruthvi Kumar is serving as an SME in Microland PVT Ltd. In his spare time, he enjoys spending time with family and friends.

Usha B A is a highly experienced professional with a Ph.D. in Computer Science and Engineering from RVCE, Bangalore. With 17 years of teaching experience, they have held various roles at prestigious institutions like R V College of Engineering and BMSIT, Bangalore. Currently serving as a Professor in Cyber Security at BMSIT&M, Bengaluru, Dr. Usha has expertise in subjects like Software Engineering, Database Management System, and Cloud Computing. They have a strong technical skill set in languages like HTML, Java, and databases such as SQL Server and Oracle. Dr. Usha has also authored books and contributed to book chapters on topics like Big Data Analytics and Cyber Security. Additionally, they hold multiple online certifications in areas like Machine Learning, Data Science, and Python programming. Dr. Usha is actively involved as an Editorial Board Member for various international journals and conferences, showcasing their commitment to academic excellence and research contributions.

How to cite this article:

Pruthvi Kumar K R, Usha B A, "Geometric Cryptosystem Based Elliptic Curve Cryptography and Advanced Encryption Standard with Galois Counter Mode for Secure Cloud Storage", International Journal of Computer Networks and Applications (IJCNA), 12(3), PP: 324-335, 2025, DOI: 10.22247/ijcna/2025/21.