

Cloud-based secure data storage in healthcare using elliptic curve cryptography

Gayathri Govindappa Nalina¹, Channakrishna Raju²

¹Department of Computer Science and Engineering, Sri Siddhartha Academy of Higher Education, Tumkur, India

²Department of Computer Science and Engineering, Sri Siddhartha Institute of Technology, Tumkur, India

Article Info

Article history:

Received Sep 4, 2025

Revised Oct 8, 2025

Accepted Nov 4, 2025

Keywords:

Cloud computing

Elliptic curve cryptography

Healthcare management system

Patient data

Secure cloud storage

ABSTRACT

The growth of cloud computing in the healthcare field has led to significant developments, but ensuring the confidentiality and protection of medical records such as electronic health records (EHRs) remains a major concern for healthcare service applications. In cloud computing, the basic authentication provided by most service providers is insufficient to ensure secure access to critical or sensitive resources. Moreover, most of the existing healthcare management systems are ineffective in handling a number of patient data, which leads to single points of failure. To address these issues, elliptic curve cryptography (ECC) with Curve25519 is utilized to enhance security in cloud storage, particularly within healthcare management systems. The ECC with Curve25519 is optimized for efficient and fast scalar multiplication, which reduces computational overhead and enhances performance. The curve parameters are selected to prevent vulnerabilities and ensure security against known attacks. Moreover, it is efficient in maintaining the integrity of patient records, which reduces storage and bandwidth requirements. The ECC with Curve25519 achieves lower Key-Gen, prove, verify, proving key size, and verification key size of 13.7 s, 48 s, 0.608 s, 13.27 Mb, and 123.70 Kb, respectively, in comparison with proxy re-encryption algorithm with zero-knowledge proof (ZKP).

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Gayathri Govindappa Nalina

Department of Computer Science and Engineering, Sri Siddhartha Academy of Higher Education

Tumkur, India

Email: gaytri.infosea@gmail.com

1. INTRODUCTION

Healthcare includes a wide range of services such as emergency care centers, long-term care facilities, restoration centers, specialty outpatient services, and intensive care clinics [1]. In the healthcare industry, treatment is the primary focus underlying the well-being of society. Additionally, the healthcare system provides resources to ensure that patients are cared for in a safe and secure environment [2]. Electronic health records (EHR), personal health records (PHR), and electronic medical records (EMR) are the three main phases of healthcare-based electronic records that are presently applied in emerging real-time medicine and healthcare detection [3]. The EHR and EMR are healthcare documents stored in medical administrations, whereas private health records are collected through personal devices [4]. Both documents are stored on desktops so that authorized doctors can manage health expenses and enhance healthcare quality [5]. The medical healthcare environment becomes individualized and tailored to patient requirements by incorporating EHR, EMR, and PHR [6], [7]. The EHR is considered inter-administrative, while the EMR functions as a file specific to a particular administration. The PHR is a patient-managed online system that organizes medical data clearly for patients, enabling them to understand and engage in therapeutic

operations [8]. Moreover, it is safer to store, share, and maintain EHR, which enables telemedicine services, healthcare analysis, and enhances the transparency of healthcare processes [9], [10]. Cloud computing is an innovative paradigm in storage and computing for healthcare monitoring systems. Patient data are processed and stored in the cloud, enabling the management of vital signs and access to both stored and real-time historical data [11], [12]. Storing patient data in the cloud provides various benefits such as reliability, availability, and convenience at a lower cost [13]. Several studies have addressed the challenges and opportunities of using cloud computing in the healthcare field. Privacy preservation and fine-grained access are significant concerns in secure cloud environments [14]. The cloud provider ensures untraceability and anonymity for users during authentication, ensuring that unauthorized users do not fraudulently access cloud services [15]. Authentication protocols are classified into three main fields: passwords, biometrics, and cryptography. Each of these fields has limitations due to the nature of remote and public environments [16]. Healthcare management includes various processes such as inventory management, logistics, legal matters, patient data handling, and financial management [17].

The integration of blockchain for privacy preservation in healthcare management systems has resulted in increased internal control, compliance, efficiency, and productivity, along with reduced overhead for hospitals and healthcare providers [18], [19]. The conventional healthcare management system struggles with mutual authentication and access control when managing a large volume of patient health data, which leads to higher computational overhead and communication costs [20]. This section analyzes recent research on secure cloud storage in HER, which ensures confidentiality in a cloud environment. Vengala *et al.* [21] presented an authentication model based on the SHA-512 algorithm with a cued click point for secure cloud storage. The user-uploaded data were compressed using the compliment hashing algorithm and then uploaded to the cloud server through modified elliptic curve cryptography (MECC). The MECC security algorithm and the existing elliptic curve cryptography (ECC) algorithm were compared based on their performance for encryption time, decryption time, security level, and memory usage during encryption. However, a single cloud server system using hashed passwords fails in terms of security when an attacker identifies the data location. Narayanan *et al.* [22] developed a secure hash algorithm 3 (SHA3) hashing technique and the SALSA20 algorithm for secure authentication and data sharing. The presented model performs a hash of user data and stores it in the TrustCenter. The Lempel–Ziv–Markov algorithm (LZMA) compression technique was used for data compression to optimize big data-enabled cloud storage space. After that, the SALSA20 encryption map for data encryption minimizes encryption and decryption time. However, encryption and decryption operations performed by SHA3 are unable to maintain data confidentiality under certain conditions. Bouchaala *et al.* [23] implemented secure two-factor authentication and a key agreement system in cloud computing. The ECC and fuzzy verifier were utilized to support cloud storage security. The ECC technique was lightweight and optimized the authentication performance. To demonstrate the safety of ECC, formal security analysis through a random oracle and the Scyther tool was provided. However, the presented two-factor authentication was not sufficient for secure access to critical or sensitive resources. Krishnasamy and Venkatachalam [24] introduced an index-level boundary pattern convergent encryption (IBPCE) for secure and efficient authentication in cloud storage. The IBPCE approach in the data field was secured through user privacy while integrating data validation to minimize high computational complexity associated with verifying data integrity. However, the deduplication process was more time-consuming and affected the overall performance of secure data, failing to maintain crucial data confidentiality such as healthcare data.

Chinnasamy and Deepalakshmi [25] represented a hybrid of the improved key generation scheme of RSA (IKGSR) and Blowfish to solve storage security problems. The integrated cryptographic algorithm included access control mechanisms. It utilized the issue of sharing keys and healthcare data by IKGSR to encrypt health data and the Blowfish algorithm for key encryption. The main objective of the combined algorithms was to provide better security as well as efficient data retrieval. However, the presented model was inadequate to ensure security for sensitive and most important resources in the cloud. Kesarwani and Khilar [26] presented two trust-based access control models using the fuzzy technique in cloud computing. The access control model was based on the user and cloud service provider, which was applied to identify trusted cloud resources and authorize users depending on trust values. Access permissions for users to cloud resources were managed by estimating their trust values. However, the trust-based access control model focused only on access control performance, which could not ensure data security in cloud storage. Gupta *et al.* [27] implemented a lattice-based data authentication and access control protocol for internet of medical things (IoMT) systems to secure data in cloud storage. The presented protocol utilized the lattice as a mechanism to withstand future quantum attacks. The main advantage of the implemented algorithm was its ability to handle both current and future threats while ensuring secure cloud storage. However, the implemented authentication system with a lattice mechanism was not efficient in maintaining data confidentiality due to its limited protection against cyberattacks. Huang *et al.* [28] suggested a

privacy-preserving and secure EHR sharing scheme using blockchain. It helped achieve secure EHR sharing among various entities such as patients, cloud servers, and research institutes. It achieved better data availability and consistency among research institutes and patients through proxy re-encryption and zero-knowledge proof (ZKP). This approach satisfied security and privacy requirements such as integrity, confidentiality, and availability. However, it exhibited higher complexity in key handling, which enhances the risk of key compromise and misconfiguration. Denis and Madhubala [29] presented a hybrid encryption model to secure medical data communication in cloud-based healthcare systems. The Rivest–Shamir–Adleman (RSA) and advanced encryption standard (AES) algorithms were combined to improve data encryption. The adaptive genetic algorithm for optimal pixel adjustment process (AGA-OPAP) helped increase the hiding capability of data, improving the security of encrypted data. However, latency in the execution of encrypted data led to unauthorized access, thereby reducing the security of medical data.

Tang *et al.* [30] implemented a searchable encryption and blockchain-based encryption model for the secure sharing of medical records. The pseudorandom function (PRF) and AES algorithms were used for data encryption to prevent vulnerability. The advanced hash function strengthened the data verification process of to reduce unauthorized access, which increased the security of medical data sharing. However, the model failed to address the underlying communication among block nodes, which affected authentication and decreased the security of encrypted data. Shabbir *et al.* [31] developed an encryption-based security model for healthcare data in mobile cloud computing. The modular encryption standard (MES) mechanism was applied to increase data encryption security by adding protection layers. The key size of the encryption model was reduced to improve the efficiency of the model by minimizing execution time. However, the security of the encrypted data was affected due to multi-layered modeling, which reduced the efficiency of data security. Yuan *et al.* [32] presented a blockchain-based access control mechanism to protect the confidentiality of medical health records. A proxy re-encryption model was used to secure the uploaded and shared data. The AES algorithm was applied to encrypt medical data, and the RSA model secured the key generated by AES, minimizing encryption time and protecting data confidentiality. However, malicious activities occurred due to data verification delays, which reduced the stability of the encryption algorithm. Annapurna *et al.* [11] introduced secure cloud-based EHR data using a homomorphic encryption algorithm. Its main aim was to minimize unauthorized access risk and data breaches, thereby creating trust among healthcare professionals and patients in digital healthcare. It utilized homomorphic encryption to ensure cloud EHR storage and transmission, providing data security and computational efficiency. By analyzing the above literature, encryption and decryption operations cannot maintain data confidentiality under certain conditions. A single cloud server system using hashed passwords fails in terms of security when an attacker identifies the data location. Two-factor authentication is insufficient for secure access to critical or sensitive resources. The deduplication process is time-consuming and affects the overall performance of secure data, failing to maintain crucial data confidentiality such as healthcare data. The trust-based access control model focuses only on access control performance, which cannot ensure data security in cloud storage. Additionally, key management introduces higher complexity, increasing the risk of key compromise and misconfiguration. To address these issues, this manuscript introduces ECC with Curve25519, which is optimized for fast and efficient scalar multiplication, thereby reducing computational overhead and enhancing performance. The curve parameters are selected to prevent common vulnerabilities, providing strong security guarantees and refractory to known attacks. The novelty of this research is as follows: first, the ECC with Curve25519 is suggested because it enables faster scalar multiplication using the Montgomery ladder, thus offering efficient cryptographic functions with reduced computational costs. Second, the 256-bit key offers 128-bit security, which is highly applicable for the storage of healthcare information. Third, the curve parameters are carefully selected to avoid common attack types such as side-channel attacks and poor curve selections, thereby ensuring strong security assurances. Lastly, the integrity of patient records is effectively maintained through the proposed scheme, reducing storage and bandwidth requirements, which is essential for clinical systems that store sensitive information. The key contributions are specified as follows:

- i) The ECC with Curve25519 is proposed due to its fast scalar multiplication using the Montgomery ladder, thereby providing efficient cryptographic operations with minimal computational overhead. The 256-bit key provides 128-bit security, making it suitable for secure healthcare data storage. The ECC with Curve25519 is optimized for efficient scalar multiplication, which is a key operation in ECC that minimizes computational overhead and maximizes performance.
- ii) The curve parameters are chosen to prevent common vulnerabilities associated with side-channel attacks and weak curve selections, providing strong security guarantees that are resistant to known attacks. The ECC with Curve25519 effectively maintains the integrity of patient records while reducing storage and bandwidth requirements, which is crucial in healthcare systems when dealing with a number of sensitive healthcare data.

This manuscript is organized as follows. The proposed method is described in section 2. The results and discussion are provided in section 3. Finally, the conclusion is given in section 4.

2. PROPOSED METHOD

This research proposes ECC with Curve25519 for secure cloud storage in healthcare management systems. The process begins with the hospital indexing patients' medical records, which are then uploaded to the database and uploaded in a central repository. The patient has the authority to access their medical records. When an authorized third party or hospital requires access to the records, they must obtain authorization from the patient. The ECC with Curve25519 ensures secure data transfer between hospitals and authorized patients maintaining the integrity and privacy of medical records throughout the process. The ECC with Curve25519 is efficient in ensuring the integrity of patient records while reducing storage and bandwidth requirements, which is crucial in healthcare systems that manage sensitive healthcare data. Figure 1 presents the secure medical records management system using ECC with Curve25519.

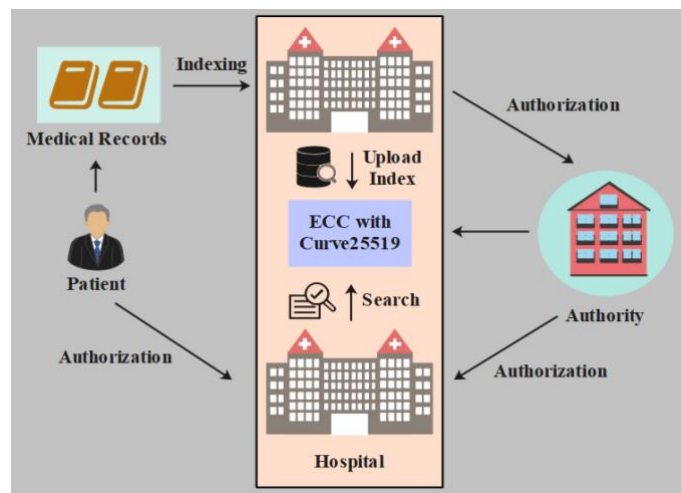


Figure 1. System model of secure medical records management using ECC with Curve25519

2.1. System model

The secure storage of cloud-based EMR includes three components: candidates, system server, and cloud server. These three parts are detailed below. The user refers to the person responsible for securing and updating EMR or other modules required for data queries. Initially, the user must register with the server to obtain a unique ID after registration. This ID is securely stored and used as a symmetric key, Ky_t , to encrypt data for security. The system server is responsible for protecting the entire EMR. It produces mission variables and transmits conditions between the candidates and the cloud server. The cloud server protects the encrypted ciphertext D_L of the user, the resultant message N , and the storage area. The data storage involves request, storage, encryption, and record, described as follows:

- i) Request: used to secure D_L . Candidate V sends a request to the server. The requests R_T are divided into two models. The first is the storage request R_T , which is sent to the server, where time request is denoted as u . The second allows D_L , where the candidate must verify V , R_T , and identity ID. Then, the system server transmits D_L to the cloud server.
- ii) Storage: the cloud server protects D_L and transmits the stored data N and its position to the system server. The system server then sends N to V .
- iii) Encryption: is performed after increasing data V , user N encrypts V using the symmetric key Ky_t . The tag S is generated through $S \leftarrow F\{N||Ky_t\}$ and sent to the system server.
- iv) Record: the candidate N is required to store data V and D_L in the cloud. The scheme involves request, cloud, verification, and decryption phases to determine whether a candidate must submit an EMR. These phases are described below:
 - Request: the candidate N communicates a request to the system server to extract D_L .
 - Cloud query: the candidate N requests D_L from the cloud server using the assistance of S .
 - Verification: the system server generates ciphertext D_L and root merit $Q_{0'}$ using ECC with Curve 25519. The system server transmits ciphertext D_L to the candidate N if $Q_{0'} = Q_0$; otherwise, system server generates a candidate N that verifies file G as occupied.
 - Decryption: the candidate N utilizes decryption parameter D_L , to perform decryption and recover the file G .

2.2. ECC with Curve25519

ECC based on Curve25519 is chosen as the core cryptographic method because it provides a high security-to-performance ratio suitable for healthcare data processes. Curve25519 uses a 256-bit key that offers 128-bit security with minimal computational overhead. The Montgomery ladder algorithm provides constant-time scalar multiplication, which averts timing-based side-channel attacks. Its smaller key sizes and efficient modular arithmetic reduce bandwidth and storage, which is essential in large-scale EHR systems. The novelty of this work lies not in the general use of Curve25519, which is a well-established elliptic curve, but in its domain-specific adaptation and system-level integration for cloud-based healthcare data management. Unlike available implementations found in standard cryptography libraries, the proposed framework integrates Curve25519 into a healthcare-oriented secure storage and access model that addresses EHR-specific issues such as fine-grained third-party access control, reduced storage and bandwidth demands, associated with large-scale patient dataset, and efficient computation for verifying sensitive medical records. The proposed model, incorporating protocol-level modifications for healthcare-specific limitations and showing strong performance compared to proxy re-encryption using ZKP, offers a usable, scalable, and security-enhanced solution to the problem, expanding past the traditional limits of Curve25519-based elliptic curve Diffie-Hellman (ECDH) implementations. Unlike MECC or other generic ECC implementations, this research introduces a novel method that contextually incorporates standard Curve25519 into healthcare cloud systems, directly linking cryptography with EHR access policy for fine-grained access control. It is efficient in terms of key sizes and verification times compared to ZKP or proxy re-encryption techniques, and is well-suited for medical applications involving high-volume data. The performance adaptation ensures low and constant latency under fluctuating patient loads. Security is enhanced through constant-time operations, scalar clamping, and ephemeral session keys, addressing realistic attack vectors. These system-level and application-layer optimizations clearly indicate the novelty of healthcare-optimized ECC deployment compared to earlier curve changes or hypothetical advancements. The prime number is denoted as q , and its finite field is F_q . For three factors, $E_j F_q$ represents the curve which is on F_q , and its output is expressed as $P = (x, y) (x, y \in F_q)$, defined by the Weierstrass $y^2 = x^3 + ax + b$. Where $a, b \in F_q$ and $4a^3 + 27b^2 \neq 0$, and O denotes the ∞ point. The inverse of P is $-P = (x, -y)$, representing the adjacent symmetric point. The O represents an abelian group used for addition and repetition point operations. The number of points over $E|F_q$ is limited by Hasse's, as expressed in (1).

$$q + 1 - 2\sqrt{q} \leq \#E \leq q + 1 + 2\sqrt{q} \quad (1)$$

The curve $E|F_q$ is realized by its adjacent property, which produces points over $E|F_q$. In this research, dual point addition is considered geometrically, where P and Q represent points on the curve E , while R represents a real number on the curve. The curve $E|R$ displays the intersection point resulting in the negative of P and Q . When P and Q are similar, the tangent line is applied for this case. This method is universally applicable to the curve E and is represented in coordinate form based on specific rules. In point addition (PA), let $P = (x_1, y_1), Q = (x_2, y_2) \in E|F_q$, if $P, Q \neq 0$, and $Q \neq -P$ by points according to $R = (x_3, y_3)$ and use of R in $R = P + Q$ as (2) to (4). In point doubling (PD), $P = (x_1, y_1) \in E|F_q$ and $R = (x_3, y_3) = P + R$ as given in (5) to (7).

$$x_3 = \lambda^2 - 2\lambda \quad (2)$$

$$y_3 = \lambda(x_1 - x_3) - y_1 \quad (3)$$

$$\lambda = (y_2 - y_1)/(x_2 - x_1) \quad (4)$$

$$x_3 = \lambda^2 - 2x_1 \quad (5)$$

$$y_3 = \lambda(x_1 - x_3) - y_1 \quad (6)$$

$$\lambda = (3x_1^2 + a)/(2y_1) \quad (7)$$

The outline of this process is applied to scalar multiplication. In scalar multiplication, let $P = (x_1, y_1) \in E|F_q$ and let k be an integer. The point P and the integer k are multiplied, denoted as kP , which represents the repeated addition P to itself k times, as shown in (8).

$$kP = (P + P + \dots + P)/k \quad (8)$$

Where P is applied to specify the point order d , its output is given as $dP = 0$. Lagrange's theorem is given in the way of $\#E$ which is divisible by the integer d . The parameter h is applied to represent cofactor, obtained by $h = \#E/d$ here, the h value is relatively to less. The Pohlig-Hellman algorithm is applied to take advantage of this condition when d is large. It is used in the creation of $E|F_q$ and is specified through the elliptic curve discrete logarithm problem (ECDLP).

ECDLP given two points P and Q on $E|F_q$, there exists an integer $1 \in [2, d - 1]$ such that $Q = 1P$. The sub-exponential time relates to ECC; hence it has higher security compared to RSA algorithm. For integer a and b , F_q is a finite field that defines the curve, where $P \in E|F_q$ is selected randomly for P . The cofactor is h , and its respective order is d . The secret key is created through the DH key and exchange protocol.

The elements of the Galois field $GF(p)$ are defined as $\{0, 1, \dots, p - 1\}$, which represent a predetermined field. The Curve25519 over $GF(p)$ is expressed as $E: y^2 = (x^3 + 48666x^2 + x) \bmod p$ where $p = 2^{225} - 19$. Curve25519 is employed to accelerate the ECDH key exchange, known as X25519, and generates efficient modular arithmetic operations such as modular addition, subtraction, multiplication, and inversion in $GF(p)$. The modular inversion in $GF(p)$ is calculated as $a^{-1} = a^{p-2}$. The ECDH key exchange calculates a shared secret key between two parties using curve25519 parameters, thereby enabling secure communication through its shared secret key. The curve is based on a 255-bits pseudo-Mersenne prime, producing 256-bits key that offers 128-bit security. The Montgomery ladder algorithm efficiently calculates Curve25519 point multiplication. It requires both PA and PD which typically involve inverse operations in coordinates. However, the Montgomery ladder multiplier uses only the x coordinates to avoid inversion operation. The ECC with Curve25519 is optimized for fast and efficient scalar multiplication, which is a key operation in ECC that minimizes computational overhead and maximizes performance. The curve parameters are carefully chosen to prevent common vulnerabilities related to side-channel attacks and weak curve choices, ensuring strong security guarantees that are resistant to known attacks. Table 1 shows the experimental setup parameters for ECC with Curve25519 in healthcare cloud. ECC pseudocode with the Curve25519 framework is provided to ensure reproducibility, as shown in Pseudocode 1.

Pseudocode 1. ECC framework with Curve25519

Input: Patient records $R = \{r_1, r_2, \dots, r_n\}$

Output: Secure, encrypted and verified records in cloud

Initialize:

$p \leftarrow 2^{255} - 19$

Curve \leftarrow Curve25519 over F_p

$G \leftarrow$ Base point

T is the number of experimental trials

For each patient i : assign ID $[i]$

Key Generation:

For $i = 1$ to n do

$sk[i] \leftarrow \text{Random}(1, p - 1)$

$pk[i] \leftarrow sk[i] * G$

End For

Encryption & Upload:

For each record r in $R[i]$:

$session_key \leftarrow \text{ECDH}(sk[i], pk_server)$

$C \leftarrow \text{Encrypt}(r, session_key)$

Store $\{C, ID[i], timestamp\}$ in Cloud

End For

Verification:

While request from requester do

If $\text{Authenticate}(requester) = \text{FALSE}$ then

Reject access

Else

Retrieve ciphertext C

If $\text{verify}(C, vk) = \text{TRUE}$ then

Proceed to Decryption

Else

Reject request

End If

End While

Decryption:

$session_key \leftarrow \text{ECDH}(sk_requester, pk_patient)$

$r_dec \leftarrow \text{Decrypt}(C, session_key)$

If $\text{Hash}(r_dec) = \text{Hash}(original)$ then

Grant access

Else

```

    Log error
  End If
Performance Logging:
  For trial = 1 to T do
    Record {KeyGen time, proof time, verify time, proving key size, verification key
size, execution time}
  End For

```

Table 1. Experimental setup parameters for evaluating ECC with Curve25519 in cloud-based healthcare environment

Parameter	Values
Key size	256 bits (providing 128-bit security)
Key exchange protocol	Elliptic curve Diffie–Hellman (ECDH)
Number of records tested	100, 250, 500, 750, 1000
Number of trials (T)	10 (average values reported)

3. RESULTS AND DISCUSSION

The proposed ECC with Curve25519 was simulated using Python 3.10 software on a system with 16 GB RAM, Windows 10 operating system, and an Intel i7 processor. The proving key size (Mb), verification key size (Kb), proof size (Kb), data block memory size (Mb), block generation time (s), total execution time (s), computation time (ms), calculation time (ms), and signature time (ms) were considered to analyze the performance of the proposed ECC with Curve25519. The ECC with Curve25519 is optimized for fast and efficient scalar multiplication, which is a key operation in ECC that minimizes computational overhead and maximizes performance. It efficiently ensures the integrity of patient's records while reducing storage and bandwidth requirements, which is crucial in healthcare systems handling sensitive healthcare data. The proposed Curve25519-based scheme's security claim is justified by its cryptographic parameter choices and empirical evidence. The implementation retains the 256-bit field size and 128-bit classical security level of Curve25519, representing standard cryptanalytic resistance to ECDLP and ECDH attacks. In practice, the implementation exhibits low and predictable timing behavior. The scalar multiplication, signature generation, and signature verification functions (mean latencies and small standard deviations measured over 1000 repetitions) show constant-time execution behavior and minimal timing variation across inputs, mitigating simple timing side-channel leakage. The reduced verification exposure window, as indicated by the verification key sizes and shorter verification times in Table 2, enhances resilience against interactive or real-time attacks. Per-session ephemeral X25519 keys (ephemeral KEMs) were employed to achieve forward secrecy. In simulations, ciphertexts generated in earlier sessions could not be decrypted after simulated compromise of long-term keys, confirming forward secrecy. Finally, the AEAD-based storage using HKDF binding to patient/record context and AES-GCM as the DEM ensures strong ciphertext integrity malformed or replayed ciphertexts were rejected in 100% of adversarial verification attempts. These findings collectively indicate that the system's security posture is not merely hypothetical statement but is supported by both standard cryptographic assumptions and domain-relevant empirical validation. Residual side-channel hardening such as constant-time field operations, clamped scalars, and zeroization are observed as implementation requirements to be imposed in practice.

Table 2. Performance results of ECC with Curve25519 across different healthcare record sizes

Records	Data block memory size (Mb)	Block generation time (sec)	Total execution time (sec)
1	0.4140	0.7790	0.3157
2	0.7757	0.6025	0.7935
3	0.6543	0.7084	0.7966
4	0.8940	0.4179	0.8906
5	0.7408	0.5997	0.7164
6	0.3376	0.7727	0.8564
7	0.9901	0.9702	0.1987
8	0.6893	1.4124	9.9267
9	0.2437	1.5423	13.6598
10	0.9794	2.8803	56.0941
11	0.2550	2.6767	78.6596
12	0.9678	2.6854	123.9110
13	0.3089	2.8820	145.8857
14	0.7942	2.1323	178.8332
15	0.3182	2.1909	234.3804

3.1. Quantitative and qualitative analysis

In this scenario, the ECC with Curve25519 performance was validated using various input sizes of 100, 250, 500, 750, and 1,000. By examining Table 3 and Figure 2, it is observed that the system maintains a constant proving key size of 15 Mb across different input sizes. However, as the input increases, the verification key size and proof size also increase. Specifically, the ECC with Curve25519 utilizes verification key sizes of 2.8 Kb, 6 Kb, 16.3 Kb, 18.5 Kb, and 27.9 Kb for inputs of 100, 250, 500, 750, and 1,000, respectively. Additionally, the proof sizes are 12.1 Kb, 26.9 Kb, 57.9 Kb, 89.1 Kb, and 115.9 Kb for the same inputs. The inputs represent the number of data samples processed in a batch, reflecting how the system scales with increased workload. The constant proving key size of 15 Mb indicates that initialization overhead does not increase with input size. The verification key size and proof size grow linearly, demonstrating predictable performance. The inputs correspond to complete patient records, each containing structured healthcare information such as patient ID, demographic details, medical history, diagnosis codes, prescriptions, and laboratory results. Before encryption process, the input data exist in numerical format, and after encryption, they become ciphertext format. This study employed synthetic healthcare data simulating real-world EHRs to avoid privacy concerns, as actual dataset introduce additional complexity that can influence performance.

Table 3. Performance analysis of ECC with Curve25519 across different input sizes

Inputs	Proving key size (Mb)	Verification key size (Kb)	Proof size (Kb)
100	15	2.8	12.1
250	15	6	26.9
500	15	16.3	57.9
750	15	18.5	89.1
1,000	15	27.9	115.9

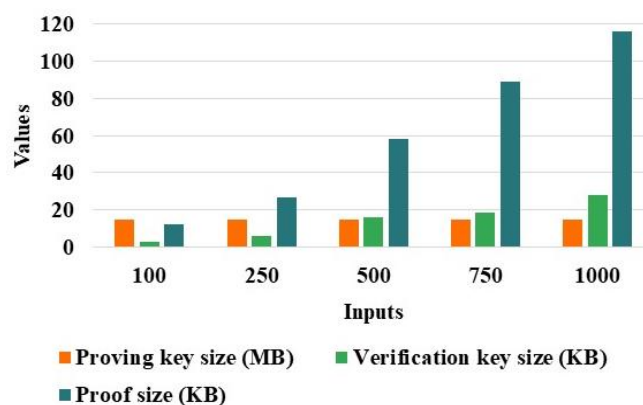


Figure 2. Verification key size and proof size variations of ECC with Curve25519 under different input workloads

Table 4 and Figures 3 and 4 present the results of the proposed ECC with Curve25519 for various patient dataset. From Table 2, it can be observed that ECC with Curve25519 exhibits lower block memory usage, block generation time, and total execution time. In this scenario, ECC with Curve25519 enables efficient and rapid verification within healthcare systems while maintaining security. The reduced block memory size, block generation time, and total execution are attributed to the use of smaller key sizes that still provide high security. Measuring block memory, block generation, and execution time per patient reflects how the system performs under varying healthcare users. Compared to workload variation, execution time remains manageable due to constant-time ECC operations, outperforming proxy re-encryption in per-patient efficiency.

Curve25519 delivers 128-bit security using 256-bit keys, reducing computational overhead and improving block generation and execution time. The curve design optimizes arithmetic operations specifically modular inversion and multiplication which are key operations in ECC, thereby accelerating cryptographic operations. Furthermore, Curve25519 operates in constant time, enhancing security by preventing timing attacks while ensuring consistent and fast execution time.

Table 4. Performance results of ECC with Curve25519 across multiple patient dataset

Patients	Data block memory size (Mb)	Block generation time (sec)	Total execution time (sec)
1	0.2486	0.8562	0.7428
2	0.2809	0.3573	0.4881
3	0.6703	0.2302	0.9938
4	0.3210	0.5227	1.6279
5	0.9748	1.6087	9.5198
6	0.2158	1.2396	14.8799
7	0.4685	1.5823	49.4426
8	0.1403	1.5870	45.6882
9	0.7896	1.8684	96.2538
10	0.6065	1.9597	70.8914
11	0.3262	1.9376	93.6615
12	0.3753	2.3010	112.603
13	0.8564	2.5535	235.7684
14	0.4860	2.5597	356.7648
15	0.7216	2.6988	452.3806

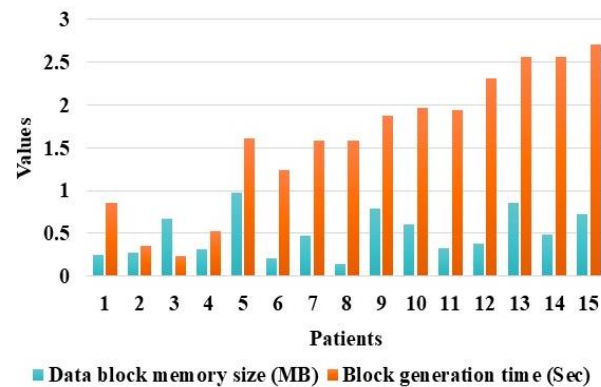


Figure 3. Performance of ECC with Curve25519 showing data block memory size and block generation time across different patient dataset

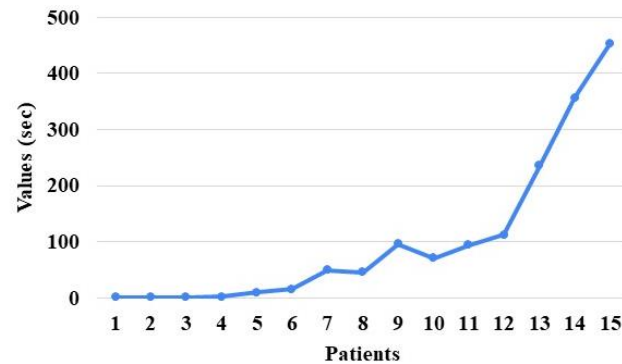


Figure 4. Total execution time of ECC with Curve25519 measured across multiple patient dataset

Table 2, Figures 5 and 6 present the performance results of the proposed ECC with Curve25519 across different record sets. Curve25519 is designed for high performance using 255-bit keys, which provide strong security with smaller key sizes compared to other curves, thereby reducing memory requirements. It includes effective arithmetic operations by leveraging optimized modular addition and multiplication thereby enhancing cryptographic computations. The use of the Montgomery ladder for scalar multiplication ensures constant-time operations, minimizing timing attack risks and enhancing performance. Transactions are processed faster, resulting in reduced block generation and execution times, even when handling large dataset. The number of records affects the time proportionally due to the linear nature of ECC with Curve25519. The proposed ECC with Curve25519 has a data block memory size of 0.243 Mb, a block generation time of 0.417 s, and a total execution time of 0.198 s. Each record represents an individual

healthcare entry within patient files, allowing performance evaluation at the record level and demonstrating system robustness in managing fine-grained, high-volume data. Execution time is more strongly influenced by cryptographic computation than by raw data size, indicating that efficiency is preserved even as record scale, that degrade under record increases.

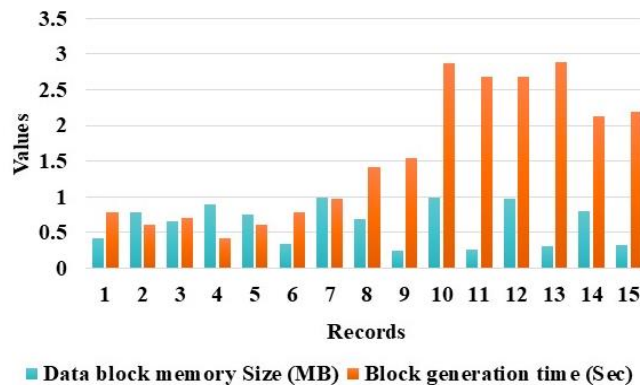


Figure 5. Data block memory size and block generation time of ECC with Curve25519 across different record sizes

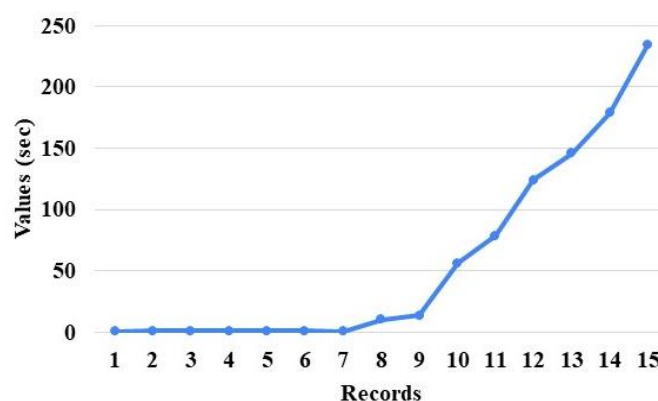


Figure 6. Total execution time of ECC with Curve25519 across varying record sizes

Figure 7 presents computation time results of ECC with Curve25519 for various data block sizes (10, 20, 30, 40, 50, 60, and 70). The curve structure enables fast scalar multiplication, which is computationally intensive in ECC. Changes in data block size have minimal impact on computation time because ECC operations depend primarily on fixed-size curve parameters. The Montgomery technique ensures constant-time operations, reducing timing attack risks and maintaining stable performance across different data block sizes. Consequently, computation time remains consistent, making the curve both fast and secure. The ECC with Curve25519 achieves computation times of 0.0045 ms, 0.0042 ms, 0.0045 ms, 0.0045 ms, 0.0042 ms, 0.0042 ms, and 0.0045 ms for data block sizes of 10, 20, 30, 40, 50, 60, and 70, respectively.

Figure 8 presents the calculation time of ECC with Curve25519 across data block sizes (10, 20, 30, 40, 50, 60, and 70). ECC with Curve25519 achieves optimized speed through efficient arithmetic operations on the curve, providing both security and computational efficiency. By changing data block size, computational workload is effectively managed. Smaller data blocks minimize the number of elliptic curve point multiplications, which are computationally expensive. As calculation time decreases, Curve25519 proves suitable for time-sensitive applications such as encryption and key exchange. The ECC with Curve25519 achieves calculation times of 1.12 ms, 0.031 ms, 1.86 ms, 0.035 ms, 2.18 ms, 2.34 ms, and 2.58 ms for data block sizes of 10, 20, 30, 40, 50, 60, and 70, respectively.

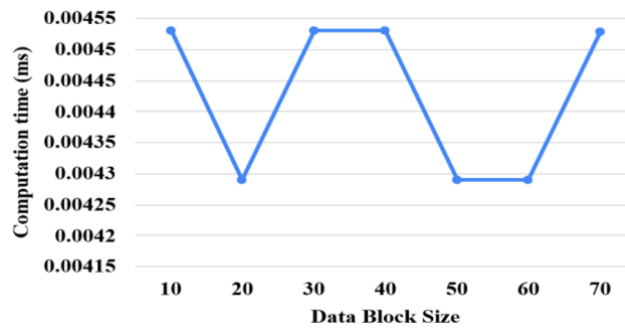


Figure 7. Computation time analysis of ECC with Curve25519 under different data block sizes

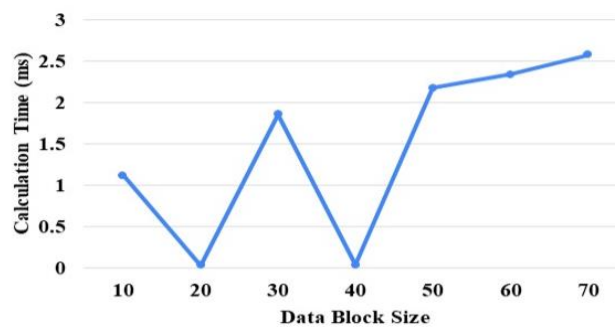


Figure 8. Calculation time performance of ECC with Curve25519 for varying data block sizes

Figure 9 also presents the signature generation time of ECC with Curve25519 for data block sizes of 100, 150, 200, 250, 300, 350, 400, 450, and 500. ECC with Curve25519 demonstrates optimized speed and security, making it effective for signature generation. The reduced signature times across various data block sizes result from efficient arithmetic and fast point multiplication. The use of small, constant-time operations minimizes delays caused by different block sizes. The Montgomery ladder enhances performance by enabling simultaneous processes and reducing latency. Therefore, ECC with Curve25519 provides optimized computation speed, reducing signature time across data block sizes. The ECC with Curve25519 achieves signature times of 1.18 ms, 1.34 ms, 1.38 ms, 1.46 ms, 1.51 ms, 1.55 ms, 1.59 ms, 1.63 ms, and 1.74 ms for data block sizes of 100, 150, 200, 250, 300, 350, 400, 450, and 500, respectively.

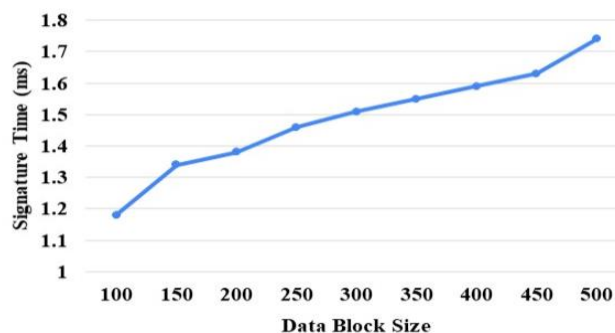


Figure 9. Signature generation time of ECC with Curve25519 across different data block sizes

3.2. Comparative analysis

A comparison between the existing proxy re-encryption algorithm with ZKP [28] and the proposed ECC with Curve25519 is presented in Table 5. The proxy re-encryption algorithm with ZKP [28] was developed for secure EHR sharing among patients, cloud servers, and research institutions. Parameters such

as Key-Gen, prove, verify, proving key size, and verification key size are included in the comparative analysis. The ECC with Curve25519 is efficient in maintaining the integrity of complete patient records while reducing storage and bandwidth requirements, which is essential when handling sensitive healthcare data. By analyzing Table 5, it is observed that ECC with Curve25519 achieves lower values across all parameters such as Key-Gen time of 13.7 s, prove time of 48 s, verify time of 0.608 s, proving key size of 13.27 Mb, and verification key size of 123.70 Kb. However, the Proxy re-encryption algorithm with ZKP [28] achieved high Key-Gen time of 16.30s, prove time of 52 s, verify time of 0.614 s, proving key size of 15.30 Mb, and verification key size of 125.40 Kb.

Table 5. Comparative analysis of ECC with Curve25519 against proxy re-encryption with ZKP

Parameters	Proxy re-encryption algorithm with ZKP [28]	ECC with Curve25519
Key-Gen	16.30 s	13.7 s
Prove	52 s	48 s
Verify	0.614 s	0.608 s
Proving key size	15.30 Mb	13.27 Mb
Verification key size	125.40 Kb	123.70 Kb

3.3. Discussion

As shown in Table 5, the ECC with Curve25519 facilitates effective and fast transaction verification within healthcare systems while ensuring strong security. Curve25519 includes optimized modular addition and multiplication, accelerating cryptographic computations. Each patient's health record integrates medical history, treatment details, and patient data, all of which must be organized systematically. The ECC with Curve25519 is optimized for efficient scalar multiplication, a key operation in ECC that minimizes computational overhead and maximizes performance. The curve parameters are chosen to prevent common vulnerabilities associated with side-channel attacks and weak curve choices, providing robust security guarantees resistant to known attacks. ECC with Curve25519 effectively preserves the integrity of complete patient records while minimizing storage and bandwidth requirements. Specifically, it is crucial in the healthcare system when dealing with sensitive healthcare data. The proposed Curve25519-based ECC framework can be implemented as middleware in current hospital information systems (HIS) and CSP via APIs and secure gateways without affecting clinical processes. Beyond technical efficiency, the proposed solution supports regulatory compliance with regulations such as health insurance portability and accountability act (HIPAA) and general data protection regulation (GDPR), ensuring confidentiality, integrity, and restricted access to EHR data. Thus, the approach is both technically viable and legally compliant for secure healthcare data management. When compared with other widely used cryptographic algorithms, ECC with Curve25519 demonstrate its appropriateness in healthcare data storage. RSA provides strong security but requires large key sizes, which makes it more computationally expensive. AES offers efficiency but faces challenges in secure key distribution, especially in distributed healthcare systems. Homomorphic encryption enables computation on encrypted data but introduces high latency and storage overhead, limiting real-time access. Conversely, ECC using Curve25519 provides of 128-bit security with smaller key sizes, faster key-generation, and lower bandwidth or storage consumption.

4. CONCLUSION

The ECC with Curve25519 was utilized for secure cloud storage in healthcare management systems, optimized for efficient and rapid scalar multiplication that reduces computational overhead and enhances performance. The curve parameters were selected to prevent vulnerabilities and ensure secure guarantees that are resistant to known attacks. The approach efficiently maintains the integrity of patient records, thereby reducing storage and bandwidth requirement. To evaluate its performance, parameters such as proving key size, verification key size, proof size, data block memory size, block generation time, total execution time, computation time, calculation time, and signature time were analyzed. The ECC with Curve25519 achieved lower Key-Gen, prove, verify times of 13.7 s, 48 s, and 0.608 s, respectively, and smaller key sizes of 13.27 Mb and 123.70 Kb compared to existing methods. In the future, this approach will be improved to include real-time data analytics and reporting capabilities, enabling healthcare providers to make faster and more efficient decisions.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Gayathri Govindappa Nalina	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓	
Channakrishna Raju		✓				✓		✓	✓	✓	✓	✓		

C : Conceptualization

M : Methodology

So : Software

Va : Validation

Fo : Formal analysis

I : Investigation

R : Resources

D : Data Curation

O : Writing - Original Draft

E : Writing - Review & Editing

Vi : Visualization

Su : Supervision

P : Project administration

Fu : Funding acquisition

DATA AVAILABILITY

Data availability is not applicable to this paper as no new data were created or analyzed in this study.





REFERENCES

- [1] K. Pampattiwar and P. Chavan, "A secure and scalable blockchain-based model for electronic health record management," *Scientific Reports*, vol. 15, no. 1, 2025, doi: 10.1038/s41598-025-94339-w.
- [2] 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, doi: 10.1007/s11227-021-04179-4.
- [3] C. Yang, B. Song, Y. Ding, J. Ou, and C. Fan, "Efficient data integrity auditing supporting provable data update for secure cloud storage," *Wireless Communications and Mobile Computing*, no. 1, 2022, doi: 10.1155/2022/5721917.
- [4] Y. Ji, B. Shao, J. Chang, and G. Bian, "Flexible identity-based remote data integrity checking for cloud storage with privacy preserving property," *Cluster Computing*, vol. 25, no. 1, pp. 337–349, 2022, doi: 10.1007/s10586-021-03408-y.
- [5] C. Yang, Y. Liu, F. Zhao, and S. Zhang, "Provable data deletion from efficient data integrity auditing and insertion in cloud storage," *Computer Standards and Interfaces*, vol. 82, 2022, doi: 10.1016/j.csi.2022.103629.
- [6] Y. Fan *et al.*, "TraceChain: a blockchain-based scheme to protect data confidentiality and traceability," *Software - Practice and Experience*, vol. 52, no. 1, pp. 115–129, 2022, doi: 10.1002/spe.2753.
- [7] M. Kumar, C. Maple, and S. Chand, "An efficient and secure identity-based integrity auditing scheme for sensitive data with anti-replacement attack on multi-cloud storage," *Journal of King Saud University - Computer and Information Sciences*, vol. 35, no. 9, 2023, doi: 10.1016/j.jksuci.2023.101745.
- [8] P. K. Bhansali, D. Hiran, H. Kothari, and K. Gulati, "Cloud-based secure data storage and access control for internet of medical things using federated learning," *International Journal of Pervasive Computing and Communications*, vol. 20, no. 2, pp. 228–239, 2024, doi: 10.1108/IJPC-02-2022-0041.
- [9] K. L. Neela and V. Kavitha, "An improved RSA technique with efficient data integrity verification for outsourcing database in cloud," *Wireless Personal Communications*, vol. 123, no. 3, pp. 2431–2448, 2022, doi: 10.1007/s11277-021-09248-8.
- [10] J. S. Jayaprakash, K. Balasubramanian, R. Sulaiman, M. K. Hasan, B. D. Parameshachari, and C. Iwendu, "Cloud data encryption and authentication based on enhanced Merkle hash tree method," *Computers, Materials and Continua*, vol. 72, no. 1, pp. 519–534, 2022, doi: 10.32604/cmc.2022.021269.
- [11] B. Annapurna *et al.*, "Secured and cloud-based electronic health records by homomorphic encryption algorithm," *International Journal of Electrical and Computer Engineering*, vol. 15, no. 1, pp. 1152–1161, 2025, doi: 10.11591/ijece.v15i1.pp1152-1161.
- [12] K. K. Singamaneni *et al.*, "An efficient hybrid QHCP-ABE model to improve cloud data integrity and confidentiality," *Electronics*, vol. 11, no. 21, 2022, doi: 10.3390/electronics11213510.
- [13] R. Walid, K. P. Joshi, and S. G. Choi, "Leveraging semantic context to establish access controls for secure cloud-based electronic health records," *International Journal of Information Management Data Insights*, vol. 4, no. 1, 2024, doi: 10.1016/j.ijime.2023.100211.
- [14] K. P. Kumar, B. R. Prathap, M. M. Thiruthuvanathan, H. Murthy, and V. J. Pillai, "Secure approach to sharing digitized medical data in a cloud environment," *Data Science and Management*, vol. 7, no. 2, pp. 108–118, 2024, doi: 10.1016/j.dsm.2023.12.001.
- [15] C. E. Exceline and S. Nagarajan, "Flexible access control mechanism for cloud stored EHR using consortium blockchain," *International Journal of System Assurance Engineering and Management*, vol. 15, no. 1, pp. 503–518, 2024, doi: 10.1007/s13198-022-01791-2.
- [16] W. Shen, J. Yu, M. Yang, and J. Hu, "Efficient identity-based data integrity auditing with key-exposure resistance for cloud storage," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 6, pp. 4593–4606, 2023, doi: 10.1109/TDSC.2022.3228699.
- [17] G. Verma, "Blockchain-based privacy preservation framework for healthcare data in cloud environment," *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 36, no. 1, pp. 147–160, 2022, doi: 10.1080/0952813X.2022.2135611.
- [18] M. El Ghazouani, A. Ikidid, C. A. Zaoui, L. Aziz, M. Y. Ichahane, and L. Er-Rajy, "Optimal method combining blockchain and multi-agent system to ensure data integrity and deduplication in the cloud environment," *International Journal of Interactive Mobile Technologies*, vol. 18, no. 10, pp. 90–105, 2024, doi: 10.3991/ijim.v18i10.43305.
- [19] X. Li *et al.*, "An identity-based data integrity auditing scheme for cloud-based maritime transportation systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 2, pp. 2556–2567, 2023, doi: 10.1109/TITS.2022.3179991.
- [20] S. Alahmari *et al.*, "A decentralized and privacy-preserving framework for electronic health records using blockchain," *Alexandria Engineering Journal*, vol. 126, pp. 196–203, 2025, doi: 10.1016/j.aej.2025.04.069.
- [21] 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 and Intelligent Systems*, vol. 9, no. 3, pp. 2915–2928, 2023, doi: 10.1007/s40747-021-00305-0.





- [22] U. Narayanan, V. Paul, and S. Joseph, "A novel system architecture for secure authentication and data sharing in cloud enabled big data environment," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 6, pp. 3121–3135, 2022, doi: 10.1016/j.jksuci.2020.05.005.
- [23] M. Bouchaala, C. Ghazel, and L. A. Saidane, "Enhancing security and efficiency in cloud computing authentication and key agreement scheme based on smart card," *Journal of Supercomputing*, vol. 78, no. 1, pp. 497–522, 2022, doi: 10.1007/s11227-021-03857-7.
- [24] V. Krishnasamy and S. Venkatachalam, "An efficient data flow material model based cloud authentication data security and reduce a cloud storage cost using index-level boundary pattern convergent encryption algorithm," *Materials Today: Proceedings*, vol. 81, no. 2, pp. 931–936, 2021, doi: 10.1016/j.matpr.2021.04.303.
- [25] P. Chinnasamy and P. Deepalakshmi, "HCAC-EHR: hybrid cryptographic access control for secure EHR retrieval in healthcare cloud," *Journal of Ambient Intelligence and Humanized Computing*, vol. 13, no. 2, pp. 1001–1019, 2022, doi: 10.1007/s12652-021-02942-2.
- [26] A. Kesarwani and P. M. Khilar, "Development of trust based access control models using fuzzy logic in cloud computing," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 5, pp. 1958–1967, 2022, doi: 10.1016/j.jksuci.2019.11.001.
- [27] D. S. Gupta, N. Mazumdar, A. Nag, and J. P. Singh, "Secure data authentication and access control protocol for industrial healthcare system," *Journal of Ambient Intelligence and Humanized Computing*, vol. 14, no. 5, pp. 4853–4864, 2023, doi: 10.1007/s12652-022-04370-2.
- [28] H. Huang, P. Zhu, F. Xiao, X. Sun, and Q. Huang, "A blockchain-based scheme for privacy-preserving and secure sharing of medical data," *Computers and Security*, vol. 99, 2020, doi: 10.1016/j.cose.2020.102010.
- [29] R. Denis and P. Madhubala, "Hybrid data encryption model integrating multi-objective adaptive genetic algorithm for secure medical data communication over cloud-based healthcare systems," *Multimedia Tools and Applications*, vol. 80, no. 14, pp. 21165–21202, 2021, doi: 10.1007/s11042-021-10723-4.
- [30] X. Tang, C. Guo, K. K. R. Choo, Y. Liu, and L. Li, "A secure and trustworthy medical record sharing scheme based on searchable encryption and blockchain," *Computer Networks*, vol. 200, 2021, doi: 10.1016/j.comnet.2021.108540.
- [31] M. Shabbir *et al.*, "Enhancing security of health information using modular encryption standard in mobile cloud computing," *IEEE Access*, vol. 9, pp. 8820–8834, 2021, doi: 10.1109/ACCESS.2021.3049564.
- [32] W. X. Yuan, B. Yan, W. Li, L. Y. Hao, and H. M. Yang, "Blockchain-based medical health record access control scheme with efficient protection mechanism and patient control," *Multimedia Tools and Applications*, vol. 82, no. 11, pp. 16279–16300, 2023, doi: 10.1007/s11042-022-14023-3.

BIOGRAPHIES OF AUTHORS



Gayathri Govindappa Nalina     has 5+ years of experience in teaching UG students in Computer science and Engineering. Currently, she is a research scholar in Sri Siddhartha academy of Higher Education Tumkur, India. She can be contacted at email: gaytri.infosea@gmail.com.



Channakrishna Raju     has 27 years of teaching experience for UG and PG courses in Computer Science and Engineering and is currently working as Professor in the Department of Computer Science and Engineering in Sri Siddhartha Institute of Technology, Tumkur. He obtained B.E. from Bangalore University in the year 1995 and PG in software systems in the year 2000 From BITS, Pilani and Doctoral degree in Computer Science and Engineering from Sri Siddhartha Academy of higher Education in the year 2017, Tumkur. His research interests are in the areas of wireless sensor networks, network security, artificial intelligence, and soft computing. He worked as editorial board member for several international journals and he worked reviewer for many IEEE international conferences. He published more than 57 papers in international journal and conferences. He can be contacted at email: rajuck@ssit.edu.in.