Chebyshev Chaotic Map with Attribute Based Encryption on Session Based Data Sharing in Fog Environment

THUSHARA G A (thusharagopinath25@gmail.com)
National Institute of Technology Tiruchirappalli
S. Mary Saira Bhanu (msb@nitt.edu)
National Institute of Technology Tiruchirappalli

Research Article

Keywords:

DOI: https://doi.org/

Additional Declarations: No competing interests reported.

Chebyshev Chaotic Map with Attribute Based Encryption on Session Based Data Sharing in Fog Environment

Thushara G.A[1*] and S.Mary Saira Bhanu[2]

[1*]Department of Computer Science and Engineering, National Institute of Technology Tiruchirappalli, Tamil Nadu, 620015, India.

[2]Department of Computer Science and Engineering, National Institute of Technology Tiruchirappalli, Tamil Nadu, 620015, India .

*Corresponding author(s). E-mail(s): thusharagopinath25@gmail.com;
Contributing authors: msb@nitt.edu;

Abstract

Fog computing is a feasible approach that can alleviate the tremendous pressure on cloud-based medical data processing. Despite the benefits of low latency, storage, and computational power that service IoT applications and devices, fog computing has few privacy and security concerns. Attribute Based Encryption is a powerful cryptographic technique for ensuring the secure sharing of medical data. Unfortunately, the problem of collusion induced by compromised users, trusted third party attack, bottleneck of key generation center, ineffective key distribution, and poor data-sharing remains unsettled in the research in attribute based encryption systems. This work proposes a user-specific and file-specific attribute-based encryption that combines Chebyshev chaotic maps for generating hash function to provide efficient session-based key setup in fog-based data-sharing. The computational cost spent by fog nodes to carry out session management is extremely minimum in the proposed work. Furthermore, the work can minimize third-party processing overheads and protect data-sharing from a number of cryptographic assaults. According to experimental and theoretical analysis, the proposed work is secure and extremely effective for sharing data in a fog environment.

Keywords: Fog Computing, Data-Sharing, CP-ABE, Chebyshev chaotic map, Security, Cryptanalysis

1 Introduction

Due to the large number of valuable assets at risk, information security is a critical concern for smart hospitals management systems (SHMS). A SHMS mainly focus on remote care system, inter connected clinical information system, networking equipment, co-related organizations, different identification system, mobile client devices, and effective data-sharing. Figure- 1 shows the different functional areas of a SHMS. The growing flow of data within and between hospital-related organizations introduces hazards that SHMS must address. The hazards include potential harm to patient care or loss of personal health information, which can be caused not only by despicable behaviors, but also by human mistake, system or third-party failures, and natural events. Data are frequently seen as valuable assets in the context of information security. The majority of decisions made by smart devices in the SHMS are based on data processing. In recent times, numerous significant medical systems have been created, researched, and implemented. Several technologies in e-healthcare focus on data-sharing and transmission along with traditional hospital assets, the security and privacy of sensitive information, and end user accessibility.

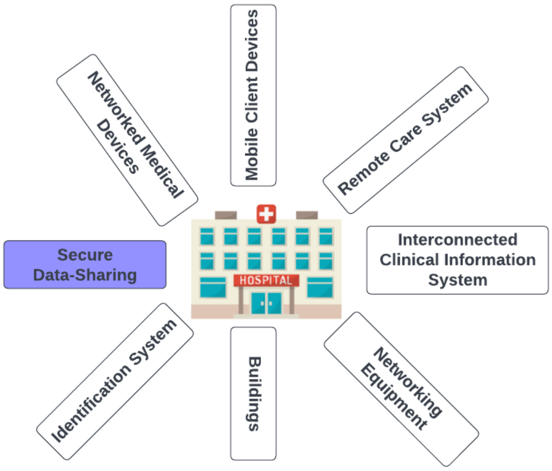

Fig. 1: A smart hospital management system

Atzori et al. [1] mentioned that, since smart gadgets are constantly linked to the internet, the probability of vigorous attacks increases. The attack potential rises exponentially as the threat landscape expands with the concern of data-sharing. In a SHMS, there are primarily 5 sorts of attack scenarios that could occur, according to Elizabeth Snell and Kyle Murphyal [2], and figure- 2 shows that the percentage of possible attacks in a SHMS.

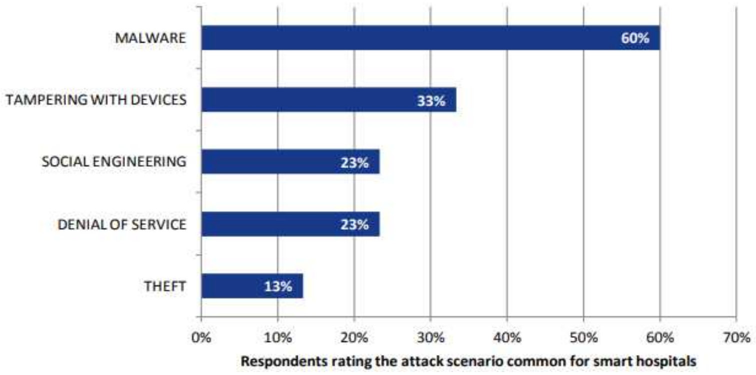

Fig. 2: Attack scenarios common for smart hospitals from [2]

The poll found that human errors were the most likely to happen, whereas a natural phenomenon had the lowest possibility of happening. Operator errors, policy and procedure violations, and hardware failure, were considered to pose a significant risk to SHMS in terms of human error. However, a more considerable proportion of respondents than expected believed that hostile behaviors, such as threats from ransomware, social engineering, stealing, Denial of Service (DoS), and cryptographic attacks, were particularly crucial for SHMS. According to Elizabeth Snell and Kyle Murphyal [2], figure-3 shows the percentage of threats occurring in a SHMS.

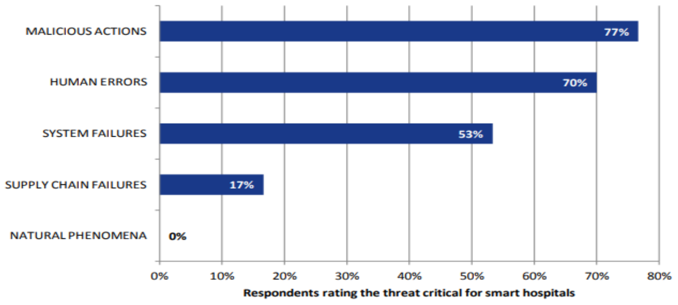

Fig. 3: Threats critical to SHMS from [2]

When cloud computing and internet of things (IoT) support fundamental hospital functions, the concept of SHMS get advanced. The cloud environment enables users to employ scalable, dispersed computing environments. A third-party provides the cloud's computational resources, which leads to several security threats during data-sharing [3]. Due to the shared environment aspect of the cloud, users won't know the precise location of personal data or the other data resources that are generally stored [4]. As a result, under the Internet-based computer paradigm, users must generally accept the basic assumption of trust [5]. Due to limited supplies such as storage, power, processing, and transmission, IoT devices are also subject to numerous network threats. Significant privacy problems arise when maintaining and transferring medical data in a cloud scenario where computer resources, including security, are managed by a third-party service provider [6]. The primary barrier in these cloud frameworks is their latency issues, threat handling, and inability to overcome the single point of failure when configuring with IoT-based systems. The main reason for the loss in system stability is the migration of latency-sensitive applications to centralized databases and from databases to other health-related organizations for better analysis. Centralized cloud server design is constantly vulnerable to many kinds of cryptographic assaults. The cryptographic attacks during data-sharing is categorized into location based and router based attacks as shown in figure- 4.

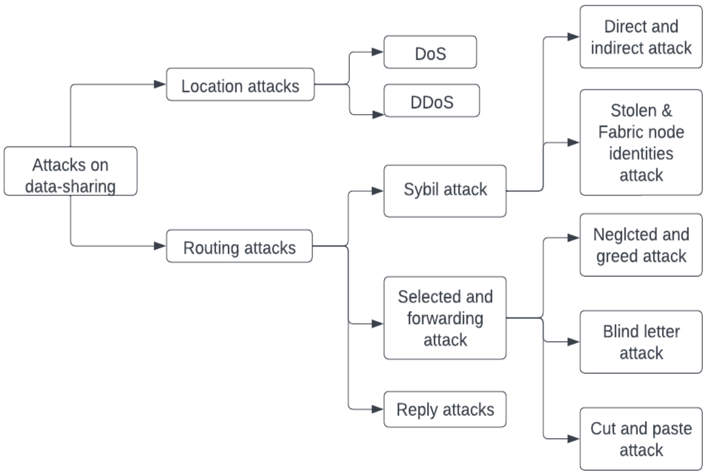

Fig. 4: Cryptographic threats during data-sharing

Low-latency applications benefit significantly from distributed data processing. Cisco created the concept of a fog layer between the cloud and IoT devices to accomplish latency sensitive scenarios. Before transferring to the fog layer, secure data-sharing requires both an effective access control mechanism and encryption. In cryptographic approach, Attribute Based Encryption (ABE) [7] is the most excellent option for fine-grained access control. But ABE is vulnerable to routing- and location-based attacks since it must manage numerous attributes and permissions. The Ciphertext Policy- Attribute Based Encryption (CP-ABE) [8], an improvement of ABE, is a preferable alternative for secure data access when it places control in the hands of the data owner. The proposed work presented both file and user specific CP-ABE technique to overcome the cryptographic threats during fog-based data-sharing. CP-ABE provides several advantages over traditional encryption systems, where access can be restricted to specific attributes rather than a single key, and scalability, where a large number of users can be given access to data without the need to manage a large number of keys. However, CP-ABE also has some limitations, such as the need for a trusted authority to manage the attribute assignments and the keys, and it is vulnerable to several security threats like key leakage, collusion attacks, and attribute loss.

A session-based approach is a better solution for resolving security problems during data-sharing. The proposed work introduced a session-based data-sharing between fog nodes in a fog environment to reduce the burden of unwanted queries. Also, the proposed work needs a robust authentication protocol in a session-based method for mutual authorization, for that, the proposed work presents Chebyshev's chaotic map-based authentication protocol as reliable. Chebyshev chaotic maps are mathematical functions that exhibit chaotic behavior, which makes them useful for generating unpredictable and complex sequences of numbers. Since the generated chaotic sequences are unpredictable and difficult to replicate, they can protect the data from unauthorized access and tampering. Chaotic maps give a one-way hash function, and Chebyshev polynomials are employed to provide confidentiality while sharing data between users during personal and session key formation. The importance of Chebyshev chaotic map-based authentication in data-sharing is its ability to provide strong security, resistance against attacks, efficiency, and scalability.

In the data access technique of the proposed work, data owner (DO) defines the access policy using user-specific attributes from distributed environment. Each authenticated user has access to the data with regard to their attributes. By using the CP-ABE encryption method, DO encrypts data before sending it to the relevant fog server (FS). The encrypted file will then be sent to the cloud server (CS) over a secure channel followed by a re-encryption process using file-specific CP-ABE by FS. The temporary data storage utilizes the appropriate FS during the storage phase. CS are for permanent data storage and processing if the data is not accessed shortly. The time difference between permanent and temporary data is one week. Each FS in the session-based architecture can communicate with nearby FS and the cloud platform. Contribution of the proposed work is outlined below:

- Composite Access Policy: The proposed work combine the user-specific attribute and file-specific attributes for CP-ABE technique. User-specific attributes might result in collusion and guessing attacks. The guessing rate will increase, and seizures may decrease if file-specific properties are also considered by the proposed work when encrypting the data.
- Session-based data-sharing: In session-based data-sharing, data is shared between sessions by the same process, rather than being transferred over a different process, which can be especially significant for large datasets or when data needs to be updated frequently. Storing data within a session can reduce the memory usage required to hold the data, as the data can be shared across multiple requests or tasks within the session.
- Reduced Response Time: Since session-based data-sharing can reduce data transfer and memory usage, it can also lead to faster processing times. In addition, it can enable more efficient caching of data, as the data can be cached within the session and reused across multiple requests or tasks. This can reduce the need to re-fetch or re-compute the same data multiple times, which can save time and computation cost.
- Mutual-Authentication: Chebyshev chaotic map protocol provide the session-based mutual-authentication. It is an important security measure for ensuring the confidentiality, integrity, and availability of shared data. It can help to prevent unauthorized access, ensure data integrity, provide accountability, and build trust between the parties involved in the data- sharing process.

The remainder of this article is organized as follows. The section "Related Work" introduces collection of existing literature. The section "Preliminaries" provides some math functions required for the system development. The section "System Overview" presents the system model in detail. The security analysis and discussions are given in sections "Security Analysis" and "Experimental Results and Discussion" respectively. The proposed work is concluded in the "Conclusion" section.

2 Related Work

2.1 Cloud-based data-sharing

Current medical data-sharing methods are commonly recommended, with two main goals: accessibility and security. While medical data are sensitive to attacks, privacy preservation is an important study topic in existing data-sharing schemes. Chen et al. [9] introduced cloudlet-based medical data-sharing, which used the Number Theory Research Unit (NTRU) to encrypt a user's data from wearable devices, and offered a trust model to let comparable individuals communicate about their conditions with one another. There is a chance of a man-in-the-middle attack since the user's data is transmitted to the cloud without encryption. Shaobo et al. [10] developed an augmented privacy-preserving data-sharing method using Caching and Spatial K-Anonymity (CSKA) in continuous location-based service, which limits the danger of users' data being exposed to untrusted location service providers. They examined only choosing cells to build a masking region based on a single user's predicted location. Yang et al. [11] introduced a centralized cloud computing medical record-sharing strategy based on the classification of medical record attributes, which used vertical divisions of a specific dataset for different portions of medical data to achieve varying anonymity settings. Besides the centralized cloud bottleneck, remote cloud queries still need to be solved in this approach. Cheng et al. [12] presented a fine-grained electronic medical document-sharing technique in cloud-assisted e-healthcare systems using similarity-based recommendations enhanced by Locality Sensitive Hashing (LSH). Jian et al. [13] suggested a traceable group data-sharing technique to accommodate anonymous simultaneous people in public clouds by utilizing the key agreement and the group signature. Group members can interact privately concerning the group signature in this work, and the true identities of participants can identify if required. An interface between users and a public cloud is introduced by Jin et al. in [14] together with a private key management system for users' access to the private cloud.

2.2 Cloud-Crypto-based data-sharing

Yang et al. [15] presented a data-sharing technique to a particular set of users in cloud-based multimedia platforms to achieve privacy and security in a specific period, resulting in a single point of failure. To address the high computing issue in secure data-sharing, Li et al. [16] recommended eliminating a large portion of the computation burden by adding system public variables and moving partial encryption processing offline. Furthermore, before the decryption phase, a public ciphertext test phase is done, which avoids most of the calculation cost caused by illegitimate ciphertexts. Furthermore, a shared conference key is produced from the key agreement to allow group members to communicate and save their data securely. Hui et al. [17] developed a data-sharing technique that improves privacy by enabling the data to construct a random search trapdoor. This approach can enable the cloud to undertake data-sharing without gaining any necessary details by utilizing the bloom filter and bilinear pairing operation to generate certain indexes for each data file. In this scenario, trusted storage servers and key generating centers are vulnerable to assault. Jiawen et al. [18] employed consortium blockchain and intelligent contract systems to achieve approved data-sharing in vehicular edge networks quickly; These technologies efficiently prevent second-hand data-sharing without authorization. In addition, they presented a reputation-based data-sharing scheme to ensure high-quality data-sharing among vehicles.

The reliability of medical data-sharing is also a significant challenge for resource-constrained e-healthcare equipment. Chu et al. [19] suggested a new public-key crypto scheme for secure data-sharing that generates constant-size ciphertexts and can provides an optimal allocation of decryption rights for any group of ciphertexts by combining secret keys into a unique key. Since the volume of ciphertexts in cloud storage typically rises rapidly, the main disadvantage of this strategy is a predefined limit on the number of maximum ciphertext classes. In cloud computing, Wang et al. [20] presented an efficient hierarchical attribute-based encryption technique that combined access trees with multiple security levels into one for various types of medical data. This technique allows users to decrypt all permission files by computing the secret key once, but it adds an insider threat. Ming et al. [21] separated PHR system users into different security sectors to reduce the key management burden for owners and users while achieving fine-grained and scalable data access control. In this case, the centralized server is vulnerable to cryptographic attack and serves as a bottleneck. To reduce the expense of encryption in mobile healthcare systems, Yi et al. [22] recommended online and offline ABE for medical data-sharing. When electronic health records (EHR) are known, [22] executed the majority of computational tasks on an offline and online phase that may quickly generate the final ciphertext. Hybrid clouds have been suggested in some proposals as a means of moving the encryption effort to a private cloud. A few studies advocate for private, mixed programs. The availability of acceptable healthcare services and effective data-sharing, both of which are major obstacles to the effective use of medical data in e-healthcare systems, must be taken into consideration by current data-sharing schemes. The proposed work provides a unique, effective, and privacy-preserving fog-assisted data-sharing technique in medical procedures, taking advantage of the prospective application of fog computing in cutting-edge medical systems. Advantages and limitations of existing data-sharing techniques are shown in Table- 1.

Table 1: Comparison of existing works

Reference No	**Methods employed**	**Environment**	**Merits**	**Demerits**
[23]	Selective encryption, AES	Cloud	Resistant to both key and data leakage	Additional security and computation needed for data fragmentation
[24]	A sensitive and energetic access control mechanism, AES	Cloud	Secure access control	Varying computation time with ciphertext size.

[25]	CP-ABE, partially hidden access policy	Cloud	Attribute privacy, large universe attributes	Inefficient attribute revocation.
[26]	NTRU encryption, Additive homomorphism	Cloud	Employed a trust model to share data	Ciphertext by NTRUEncrypt can fail to decrypt
[27]	RSA, secret sharing scheme	Cloud	Lower key distribution and computation costs	Incapable of dynamic group management
[28]	ABE with bloom filter	Cloud	Hidden attributes	High encryption cost
[29]	CP-ABE, PRE	Cloud	Extensible library compatible for android devices	Unrectified attribute revocation
[30]	CP-ABE, Signcryprion	Cloud	Reduced ciphertext size with minimal pairing operations	Restricted message space
[31]	CP-ABE, PRE, Chaotic map	Fog and Cloud	Revocation of attributes via dynamic policy updating	Dynamic access policy operations are challenging
[32]	CP-ABE, chain encryption	Fog and Cloud	Reduced the cost of the network and computational decryption	Attacks are more likely to target a single authority
[33]	CP-ABE, decryption testing scheme	Fog and Cloud	Policy for verifiable hidden access	Outsourcing decryption uncovered

3 Preliminaries

3.1 Bilinear Pairing

Many cryptographic systems, such as ABE, use bilinear pairings as a core mathematical tool. Bilinear pairings are employed in ABE to offer fine access control over encrypted data. In ABE, a mapping between two groups, typically identified as G1 and G2, is created using bilinear pairings. This mapping has the following properties:

- Bilinearity: For any elements $P, Q \in G1$ and $R, S \in G2$, the bilinear pairing $e(P+Q, R+S) = e(P, R) * e(Q, S)$, where e is the bilinear pairing function.

- Non-degeneracy: The bilinear pairing $e(P,Q)$ is non-zero $\forall P \in G1 \ \ and \ \ Q \in G2$.
- Computability: The bilinear pairing function e can be efficiently computed.

In ABE, bilinear pairings are used to map attributes to elements in $G1$ or $G2$, and to map a user's secret key to an element in $G2$. By using the bilinear pairing, the system can enforce fine-grained access control policies over encrypted data based on the attributes associated with the user's secret key. For example, in the proposed work, a patient's medical record could be encrypted and accessed only by authorized users with the appropriate attributes, such as being a licensed doctor or nurse.

3.2 Elliptic Curve Cryptography

Pairing-based ABE schemes use elliptic curve pairings as the underlying mathematical structure. These pairings are used to map attributes to points on an elliptic curve, and to perform cryptographic operations such as encryption, decryption, and key generation. There will be $G1$ and $G2$ to represent two groups, and e to represent the bilinear pairing function between these two groups.

- Key generation
  1. Choose a random generator point $P \in G1$.
  2. Choose a random scalar s and compute the public key $Q = sP \in G1$.
  3. The private key is s.
- Encryption:
  1. Let M be the message to be encrypted, and let A be the set of attributes that must be satisfied to decrypt the message.
  2. Choose a random scalar r.
  3. Compute $C1 = rP \in G1$ and $C2 = M * e(rQ, H(A)) \in G2$, where $H(A)$ is a hash of the attribute set A.
  4. The ciphertext is $(C1, C2)$
- Decryption:
  1. Let sk be the private key associated with the user trying to decrypt the ciphertext.
  2. Let A be the attribute set associated with sk.
  3. Compute $H(A)$ and compute $e(sk, C1)$.
  4. If $e(sk, C1) \ \ \neq \ \ e(P, H(A))$, the decryption fails.
  5. Otherwise, compute the plaintext M as $C2/e(sk, C1)$.

3.3 Access Tree Construction

In CP-ABE, an access tree defines the policy for decrypting a ciphertext. An access tree is a tree-like structure where each node in the tree represents a logic date, and the leaves represent attributes. The access tree generates in such a way that the attributes required to decrypt a ciphertext define by the path from the tree's root to the leaf node corresponding to the ciphertext. For example, consider an access tree as shown in figure-5 with attributes "Patient id (P)," "Doctor id(D)," "Medical history (M)," and "Progress note (Pn)". In this example, a ciphertext encrypted under the access policy $P \wedge (D \wedge (C \vee Pn))$ would be associated with the leaf node that corresponds to the given attributes. Users can only decrypt the ciphertext if they have all of the attributes in the path from the tree's root to the corresponding leaf node. However, the security of CP-ABE depends on the assumption that the attributes used to encrypt the ciphertext hidden from the attacker and the design of the access tree can significantly impact the system's security.

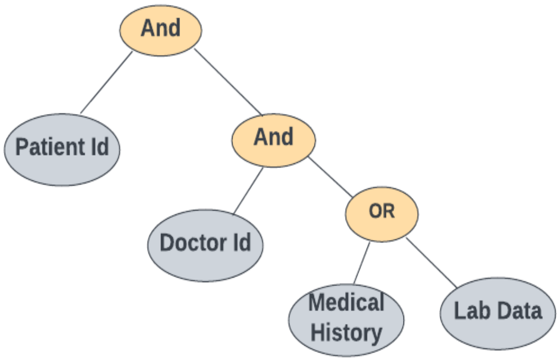

Fig. 5: Example of an access tree

3.4 CP-ABE construction

1. CPABESetup $((1^{\alpha}, h) \rightarrow (MPK, MSK))$: KGC generates a pairing-friendly elliptic curve group G of order q and choose a bilinear pairing e between G and a multiplicative group G_T of the same order. Also generate a random generator g of G, two hash functions $H1$ and $H2$, and a random secret key $\alpha \in Z_q$. A hash value h of the string AT computed using $H1$.

$$MPK = (g, e(g,g)^{\alpha}, h) \quad and \quad MSK = \alpha \tag{1}$$

2. CPABEKeygen $(MPK,\ h) \rightarrow (S_{key},)$: The MPK and a list of user attributes, h that generated from the set of attributes are inputs to the key generation phase by KGC. Secret key pair (S_{key_i}, S_{key_j}) generates as the result by utilizing a random value r_3

$$S_{key_i} = g^{r_1 + r_3} \quad and \quad S_{key_j} = g^{r_3/(r_2 . r_1)} \tag{2}$$

3. CPABEEncrypt ($\mathbb{D}$, h, MPK)$\rightarrow$ (S_{key},): The transmitting data $\mathbb{D}$, the access tree h, and the MPK are all inputs to the encryption phase by the DO. Only the authorized DU who have a set of specific attributes that satisfy the h can decrypt the $\mathbb{D}$ once the phase encrypts $\mathbb{D}$ and generates a ciphertext ($CT^{'}$).
$$CT^{'} = \mathbb{D}.e(\mathbb{G}, \mathbb{G})^{r_1}.MPK \tag{3}$$
4. CPABEDecrypt (S_{key},h,SPK,$CT^{'}$)$\rightarrow \mathbb{D}$: The MPK, $CT^{'}$, h, and the S_{key}, which is a private key for a set S of attributes, are all inputs to the decryption phase. h must be satisfied by the set S of attributes in order for the phase to decrypt the $CT^{'}$ and provides a actual data $\mathbb{D}$.

$$\begin{aligned}\mathbb{D} = CT^{'}/(e(S_{key_j}, CT_1)/S_{key_i})\\ = CT^{'}/[(e(\mathbb{G}, \mathbb{G})^{(r_1+r_3)S}/e(\mathbb{G}, \mathbb{G})^{r_3.MPK})]\\ = CT^{'}/[(e(\mathbb{G}, \mathbb{G})^{r_1.h}.e(\mathbb{G}, \mathbb{G})^{r_3.h)/e(\mathbb{G},\mathbb{G})^{r_3}.MPK}]\end{aligned} \tag{4}$$

3.5 Chebyshev Chaotic Maps

In the proposed work, the Chebyshev chaotic map is used to generate a shared secret key ($T^{'}$) between the FS. The FS use a $T^{'}$ to authenticate each other's identities. One FS sends a challenge message to the FS, which is encrypted using the $T^{'}$. The other FS decrypts the message using the same key and sends a response message back to the first FS, also encrypted with the $T^{'}$. If both FS can successfully encrypt and decrypt each other's messages, then they have successfully authenticated each other's identities and can establish a secure communication channel.

Let $n \in \mathbb{Z}$, $x \in [-1, 1]$ and an n-order Chebyshev polynomial map $Q_n(x) : [-1, 1] \rightarrow [-1, 1]$ is defined as follows:

$$Q_n(x) = \cos(n * \arccos(x)) \tag{5}$$

The formulation also allows for a recursive formula of the Chebyshev polynomial pattern, which is as follows:

$$Q_n(x) = 2 * x * Q_{n-1}(x) - Q_{n-2}(x),\ \ whereQ_0(x) = 1\ \ and\ \ Q_2(x) = x, n \geq 2 \tag{6}$$

Two characteristics of the Chebyshev polynomial map are as follows:

- Semi Group property:

$$\begin{aligned}Q_r(Q_s(x)) = \cos(r * \cos^{-1}(s * \cos^{-1}(x)))\\ = \cos(r * s * \cos^{-1}(x)) = Q_{sr}(x)\\ = Q_s(Q_r(x)),\ \ where\, r, s \in \mathbb{Z}\ \ and\ \ x \in [-1, 1]\end{aligned} \tag{7}$$

- Chaos property: When $n > 1$, an n-degree Chebyshev polynomial map $Q_n(x) : [-1, 1] \rightarrow [-1, 1]$ has the constant measure $q(x) = 1/\pi\sqrt{1-(x^2)}$ and positive Lyapunov exponent $\lambda = \log n > 0$.
  Multiple x must be connected with the same y for the equation to hold, according to $y = \cos(x)$ periodicity.

$$\begin{aligned} Q_n(x) = 2xQ_{n-1}(x) - Q_{n-2}(x) \;\; mod \;\; P \\ where \;\; n \geq 2, x \in [-1,1] \;\; and \;\; P \in \mathbb{Z_P} \end{aligned} \tag{8}$$

$$Q_r(Q_s(x)) = Q_{sr}(x) = Q_s(Q_r(x)) \;\; mod \;\; P \tag{9}$$

Table 2: Table of acronyms

Notation	**Description**
DO	Data owner of the system
DU	Data users of the system
KGC	Key generation center of the system
CS	Cloud server of the system
FS	Fog server of the system
AA	Attribute authorities of the system
G_1, G_2, G, G_T	Multiplicative cyclic groups
g	group generator
MPK	Master public key generated by CP-ABE
MSK	Master secret key generated by CP-ABE
$Q_n(x)$	Chebyshev polynomial range for n
SPK	System public key generated b KGC
SMK	System master key generated b KGC
λ	Security parameter for system setup
α, β	Random variables for system setup
DO_{sk}	Data owner's secret key
DO_{pk}	Data owner's public key
FS_{pk}	Fog server's public key
FS_{sk}	Fog server's secret key
DU_{sk}	Data users' secret key
DU_{pk}	Data users' public key
$\mathbb{A}_p$	User-specific attribute access policy
$\mathbb{A}_q$	File-specific attribute access policy
A_1	Array of sensitive attributes
A_2	Array of non-sensitive attributes
$t_s, t_s^{'}$	Time stamp of sender and receiver
ps	Password of group admin
$H(M)$	Message hash value

reg_m	Registration message for authentication
req_m	Request message for authentication
CT	Encrypted data using $\mathbb{A}_p$
$CT^{'}$	Re-encrypted data using $\mathbb{A}_q$

4 System Overview

This section introduces the framework, system model, design goals and description of the proposed work. As shown in the figure 6, the proposed work consists of six entities, which include a DO, FS, CS, KGC, data users (DU), and attribute authorities (AA). Here, the hospital accumulates patient data, which needs to share with research labs and pharmaceutical institutes for medical analysis. Before transmitting the data (D) to the appropriate FS, the DO encrypts it ($D \rightarrow CT$) by the CP-ABE scheme. The DO develops user-specific access policy (T_p) along with CT. The FS utilizes file-specific attribute policy (T_q) to re-encrypt the CT ($CT \rightarrow CT^{'}$). Therefore, the DU must provide both user and file specific attributes to access the D.

The FS retains CT temporarily before sending it to the CS. Any malpractices that happen to the CS data make changes to the metadata on the FS. The DO can define an encryption policy that requires user-specific attributes and encrypt the (D) using (T_p). DU can only decrypt the (D) if they possess the required attributes. The use of CP-ABE scheme enables fine-grained access control and provides more flexibility and expressiveness than traditional encryption methods. FS handles a session-based data-sharing technique for quickly sharing data between different users or applications. In the proposed work, (D) is stored in a session associated with a particular DU or application during a specific period. The (D)in a session can be accessed and modified by different DU or applications, but it is only available for the duration of the session. Once the session is closed or expires, the (D) is no longer accessible. Session-based data-sharing provides flexible and secured data transmission to the proposed work.

Chebyshev chaotic maps utilize as a part of a session authentication procedure by introducing chaotic features into the mutual authentication phase. In the proposed work, a dynamic password gets generated using the Chebyshev map's chaotic behavior, making it more challenging for an attacker to gain unauthorized access to the system. The $T^{'}$ generated by the Chebyshev map exchanges across each FS during each session authentication process. The DU generates the $T^{'}$ based on a set of initial conditions, referred to as the seed values, selected randomly. Then DU utilize the private key as a part of the authentication procedure, either as a distinct secret key or as a component of the password to encrypt a challenge. When a DU attempts to authenticate, the system uses the same seed values to regenerate the secret key and compares it to the key provided by the DU. If the keys match, the DU is authenticated and allowed access to the session. Table-2 provides a list of all the notations used in this work.

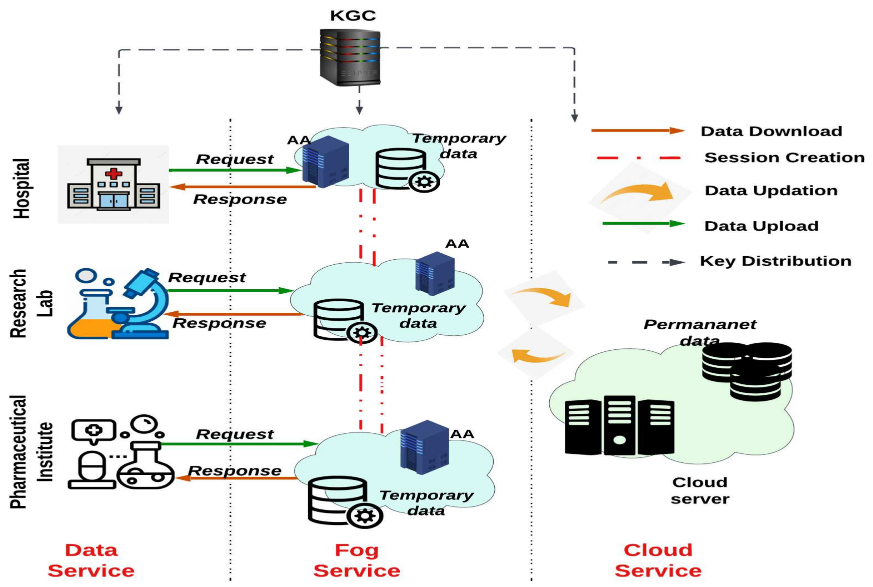

Fig. 6: Session-based fog-cloud framework for data-sharing

4.1 System Model

1. KGC: Create master and private keys for the system's remaining entities, facilitate registration, and activate the system. KGC performs the key creation and key management procedures whenever DO seeks to upload a file. Every session key generation is managed by the KGC, which also updates the key for every session update.
2. DO: The DO pre-process the medical data, gathered by medical IoT devices or personally provided by patients. Encrypted medical data is shared and sent to a respective FN with appropriate access policy and list of attributes. Together with the AA, the DO creates an access policy for an organization. Encrypts the data in line with the access rules before delivering it to the FN. Since FNs are semi-trusted entities, DO issues access policies in an encrypted format. The DU asserts that the FN might be able to alert unauthorized users in the absence of this. As a result, the user can only decode the ciphertext when the attribute complies with the embedded access policy.
3. FS: Fog nodes are serves that can broadcast and store medical data. FS has a strong basis in data processing technology. The shared ciphertext is processed, re-encrypted, and sent to the CS after that. Data re-encryption using file attributes and data storage after obtaining the necessary data

from the DO constitute the fundamental capability of FS. The DO encrypts data before sending it to the proper FS. The system uses the FS and adheres to the double CP-ABE scheme. FS interacts with the other entities of the system and is semi-trusted by them.

4. CS: CS is a distant third-party server with significant storage capacity. The shared ciphertext sent from the FS is managed and stored by CS. Only temporarily can a FS hold data before automatically sending to the CS. Any modified file on the CS will impact the corresponding FS.
5. DU: Doctors, academicians, insurance providers, and drug manufactures are just a few example of Du. Users access the ciphertext using their user-specific attributes and file-specific attributes. The sharing of data across institutions is essential for gathering medical information and providing medical services. Each DU must sign up for the system using their unique id (uid). They can access the ciphertext from the closest FS using their user attributes and file attributes. DO provides file-specific attributes while transferring the data to the FS. When accessing data, the user must give a list of file-specific attributes. The ciphertext can only access when both the DU's attribute and its group key meet the access restrictions of the ciphertext.
6. AA: Users' authenticity is checked by the AA, who also issues an attribute key(A') to those DU, who pass the authentication process. When a DU reaches the AA, the AA will evaluate the user's validity qualities either directly or via an authentication mechanism, and will then produce a key based on the legally approved attributes. Regarding the user revocation, the AA tells the KGC to delete the DU or attribute from the list of active users.

4.2 System Description

The proposed work provides effective data-sharing in SHMS using a fog environment, which offers efficient authentication and prone to different attacks. The data gathers via medical IoT devices (such as wearable technology and biosensors) or the DO's entries. The DO then encrypt the acceptable data using CP-ABE scheme and sends the ciphertext (CT) to the appropriate FS. The FS is also in charge of secure data transmission. For data transfer and analysis, FS establishes a session with cooperative FS (to which data-sharing is needed). Each session uses Chebyshev chaotic maps based mutual authentication to protect user groups and session-based key management. After a predetermined amount of time, the FS sends the CT to CS for storage.

The proposed work consists of following phases: Setup, KeyGen, Preprocessing, Encryption, Re-encryption, Session creation, and Decryption.

- $Setup : (\lambda) \rightarrow (SPK, SMK)$
  This is the beginning step of them proposed work. The system public key

(SPK) and the system master key (SMK) are output by the KGC in response to the security parameter λ. KGC selects a security parameter λ for a multiplicative cyclic group G from Z_p^* of an elliptic curve E. g be the group generator. Pairing generates the unique global value (U_{val}) to each entity of the system as per the Algorithm-1.

Algorithm 1 System Setup Algorithm

```
    Input : The security parameter $\lambda$
    Output : System parameters ($SPK$, $SMK$)
1: Select a multiplicative cyclic group $G$ with generator $g$.
2: Ensure that the $G$ resides on the elliptic curve E with prime order $p$.
3: Select two random values $\alpha$ and $\beta$
4: for all entity in the system do Compute $g^{\alpha}$ and $e(g,g)^{\beta}$ Assign unique
   identity ($U_{val}$).
5: end for
6: $SPK = g, g^{\alpha}, e(g,g)^{\beta}, \quad SMK = g^{\alpha}$
7: Share the system parameters.
```

- $KeyGen : (SPK, SMK, Pos, S) \rightarrow (SK, PK_F, SK_F)$
  The KGC configure the system and create keys for the FS and the DO during the initial phase of the system. The KGC outputs the secret key, SK, for the DO, the public and secret key for the FS (FS_{pk}, FS_{sk}) after taking as input the SPK, SMK, attribute set S, and geographical position data (Pos) of the FS. The system's attributes, S, are provided as input to the KGC. It selects a bilinear map $e : G \times G \rightarrow G_T$ between two multiplicative groups G and G_T of prime order P. Attributes of DO are not kept secret by KGC. They are components of public keys and connect to the system wide properties. Additionally, KGC selects exponents $\alpha, \beta \in Z_P$ at random. The KGC produces the SMK and SPK using following equation.

  To create secret keys for DO, the KGC uses the KeyGen algorithm. The DO delivers the KGC its set of attributes S, including the user-specific attributes and file-specific attributes. The KGC computes the secret key for DO, DO_{sk}, using the public keys that correspond to the attributes for the DO, DO_{pk} . KGC receiving the SMK and a random number $r \in Z_P$ as input.

$$DO_{sk} = (\gamma, \gamma_s), \quad s \in S \tag{10}$$

$$\gamma = g^{\alpha} g^{\beta r}, \quad \forall s \in S, \quad \gamma_s = g_s^r . g^r \tag{11}$$

The KGC generates public key and private key for each FS. It creates the FS_{pk} and FS_{sk} after receiving the FS's position (Pos). The FS_{pk} of the nearest FN is sent to the DO by the KGC after getting the location of the

DO.

$$FS_{pk} = e(r, g), \quad r \in G, \qquad and \;\; FS_{sk} = r \tag{12}$$

- Data Pre Processing:
  The DO executes the pre-processing algorithm which generates the normalized data hided file, and its matching encrypted list of attributes as shown in the figure-7. It accepts the raw file transmitted from the patient as the input and generates a normalized file as output as conveyed in Algorithm-2. Based on the security objective, attributes from the input file categorize into sensitive and non-sensitive. Sensitive attributes are those attributes that shouldn't be linkable to an individual. DO is using sensitive attributes to generate access policies. FS generates the access policy regarding the attributes specific to each file. With the pandas package, Algorithm-2 converts all categorical values into numerical values and normalise each numerical value in the array to be between zero and one Using the min and max methods from the NumPy library.

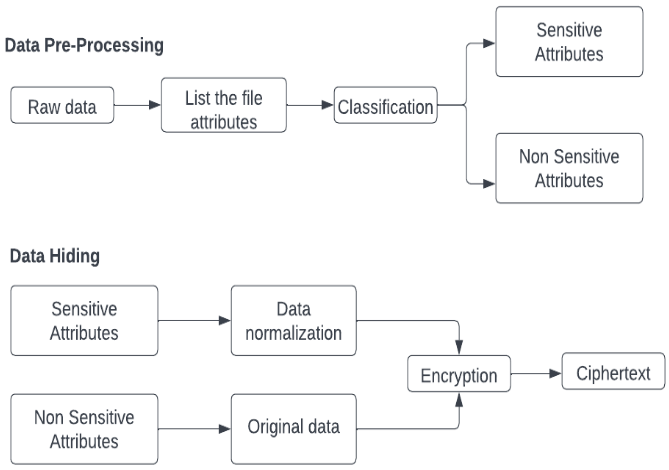

Fig. 7: Data preparation

- $Encrypt : (M, (\mathbb{A}_p), FS_{pk}) \rightarrow CT, SA$
  The DO uses the CP-ABE encryption technique to encrypt the medical data D. After that, DO outputs the ciphertext (CT) with an DO defined access policy $(\mathbb{A}_p)$, and FS public key (FS_{pk}). For secure and fine-grained access control, DO incorporates CP-ABE scheme to encrypt the data. The DO also establishes a user-specific access policy based on their experiences and preferences. The proposed work assumes that the DO keeps a regular medical item table. The obtained data compared with the average value

Algorithm 2 Data Pre-Processing Algorithm

```
    Input : Raw data (CSV file)
    Output : Normalized data
 1: Load file
 2: Initialize array A1, A2
 3: Classify the attributes, A, into A1 and A2
 4: if A is sensitive then
 5:     $A1 \leftarrow A$
 6: else
 7:     $A2 \leftarrow A$
 8: end if
 9: $Numerical_{val} \leftarrow \forall\ Categorical_{vaue}$
10: $MinMax(Numerical_{val})$
```

that considering unusual medical conditions. DO obtain the encrypted list of sensitive attributes (SA) along with the ciphertext. FS can create an access tree ($\mathbb{A}_q$) form SA.

$$CT, SA \leftarrow Enc(D, (\mathbb{A}_p), (FS_{pk})) \tag{13}$$

- $Re-encrypt : (CT, (\mathbb{A}_q)) \rightarrow CT^{'}$
  According to the registered users, the FS defines a new access policy with file-specific attributes ($\mathbb{A}_q$). The CT is then re-encrypt with the ($\mathbb{A}_q$). The new ciphertext $CT^{'}$ is generate by the FS and send to the CS for safekeeping. The FS creates a developing access tree $\mathbb{T}$, as shown in figure-8, to combine the file-specific attributes and user-specific attributes into a unique access tree. Two of the root nodes in $\mathbb{T}$, ($\mathbb{T}_p$), and $\mathbb{T}_q$ serve as the root nodes for the user-specific access policy ($\mathbb{A}_p$) and the file-specific access policy ($\mathbb{A}_q$) respectively. The root node T oversees the entire access policy. The DO and FS may both precisely build the ($\mathbb{A}_p$) and ($\mathbb{A}_q$) respectively. In the figure- 8, yellow coloured circles representing the AND gates, grey coloured circles representing the user-specific attributes and pink coloured circles representing the file-specific attributes. To protect the collaborative DU's access to shared data, the FS builds $\mathbb{A}_q$ tree. The parent node $\mathbb{T}_q$) of an $\mathbb{A}_q$ tree includes N file-specific attributes for the required file. There is a threshold minimum for attribute matching for data-sharing. Since the threshold of root node is one, the $\mathbb{A}_q$ can be satisfied by a DU with attributes, that can only engage with one of the N attributes.

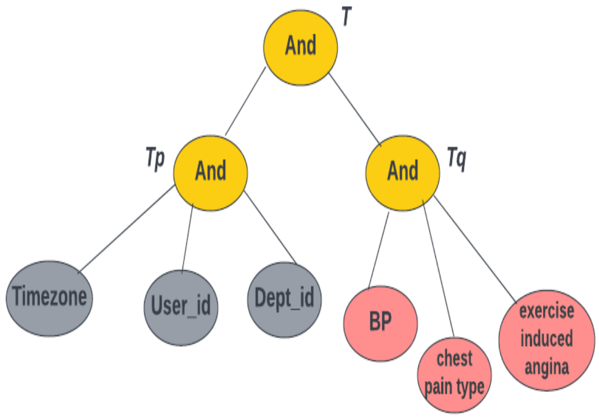

Fig. 8: Combined access tree

- FS to FS Session Creation
  The proposed work appropriates for group-based data-sharing in SHMS, research labs pharmaceutics, and corporate tracking, where a lot of users register with unique group identifiers. A group admin (the initial member of the group) can form a group by considering the medical scenario and medical IoT devices. The proposed work has used powerful and inexpensive Chebyshev chaotic map for mutual authentication for the system. Chaotic maps are utilized in the proposed work to offer one-way hash operations. To guarantee anonymity for users sharing data, the proposed work utilizes the same Chebyshev polynomials employed during the formation of hidden and session keys. An FS that distributes keys organizes the creation of secure keys for group user based data-sharing in multi-party key generation. In the initial stage of group admin authentication, DU authenticates from the FS in addition to the formation of the session key.

  1. Group Admin Authentication: This section initially looked at the process for group admin (GA) authentication at FS before moving on to distributed multiparty keying, where various (GA) participate by sharing their security credentials with FS, as mentioned in the Algorithm-3. The authentication process starts when a GA sends a chaotic map-based one-way hash to FS, which subsequently checks the security credentials and creates CP. FS also challenges the group administrator to prove their identity. By stacking the polynomials, FS utilizes the semi-group feature of CP to produce a generic polynomial.
  2. Session key establishment: This section provides the session key establishment protocol for GA and FS to secure the communications for a

Algorithm 3 Group Admin Authentication

```
1: GA selects a random number $r$ at range $[0, 1]$
2: Compute $Q_r(x)$ as the Chebyshev polynomial range for $r$
3: GA calculates the hash value: $GA_h = H(GA_{id}, Q_r(x))$
4: GA generate and send an authentication message:
5: $GA_{msg} \leftarrow GA_{ts}, GA_h, H(GA_{msg})$ to FS
6: Transmits the encrypted message, $E_{FS_{pk}}(GA_{msg})$ to FS
7: for $GA_{ts'} - GA_{ts} \leq \Delta ts \ \ and \ \ GA_h == GA_{h'}$ do
8:     Accept the message $GA_{msg}$
9:     FS generate $Q_s(x)$ using a random number $s$
10:    for $Q_{rs}(x) == Q_r(Q_s(x))$ do
11:        FS generate and send an authentication message
12: $FS_{msg} = FS_{id}, r^{'}, H(FS_{id})$ to GA
13:        GA compute the Chebyshev polynomial range for $r^{'}$ and
14: send to FS: $(GA_{id}, Q_{r't}(x))$
15:    end for
16:    if $Q_r(x) == Q_{r'}(x)$ then
17:        Session authenticated
18:    else
19:        FS Reject the session
20:    end if
21: end for
```

specific session. Each new session requires a renewal of session key. It is safer and more dependable to set up these keys by combining Chebyshev chaotic maps in a multiparty setting. All *GA* contribute by providing the FS with their security credentials to produce the session key in this situation. The message M is independently prepared and sent to FS by each of the *GA*. From the extracted hash values, FS randomly choose m values.The FS then generates a new secret that all *GA* can share as conveyed in Algorithm- 4. Each *GA* verifies the time threshold and hash values getting from the secret message from FS.

3. Group-User Authentication
Each authorized DU must first register with a certain *GA*, after which the *GA* produces a chaotic-based token with a random number, user IDs, and timestamps as shown in the Algorithm-5. With the help of a one-way chaotic map-based hash function, *GA* share these values securely. DU establish a standard session key for secure group user data-sharing using these shared variables. The proposed work set up a secure interaction process between the *GA* and other DU. The FS verifies each DU before group user network creation for secure data-sharing between DU to prevent malicious actions. The *GA* can authenticate each DU to produce a typically shared key. *GA* initially shares the key by sharing a standard Chebyshev polynomial with all members of the same group of DU.

Algorithm 4 Session Key Establishment

```
Require: Authenticated group admin (GA)
Ensure: There are $n$ number of GA and each having unique password $ps$
 1: $\forall\ GA \in N$ send a message to FS
 2: $M = H(ps) \oplus H(Q_{ps}(x)), H(M)$
 3: FS extract the hash value $h^{'} = M \oplus H(pw) = H(Q_{ps}(x))$ and generate
    $Q_q(x)$ with a random number $q$
 4: $FS : H_{new} = H(Q_{ps}(x))_i \oplus H(Q_{ps}(x))_{i+1} \oplus H(Q_{ps}(x))_{i+2} \ldots\ldots \oplus H(Q_{ps}(x))_n$
 5: FS generate a secret $M^{'} = Q_q(x), H_{new}, ts_q, H(M^{'})$
 6: $\forall\ GA : H^{'}_{new} = H(GA_{id} \oplus Hps \oplus H_{new} \oplus H(Q_r(Qq(x))))$
 7: $\forall\ GA$ send confirmation message $M^{''} = GA_{id} \oplus H(ps) \oplus Q_p(x), ts, H(M^{''})$
 8: FS calculate $H(Q_p(Q_q(x)))$ to obtain $H_{new}$ and decode the $M^{''}$
 9: FS send the confirmation message to GA
```

By sending a registration message (reg_m), all DU register with GA. GA validates and compares (reg_m) to confirm; if not, (reg_m) is discarded. Using a secret and previously shared credential, DU computes a request message (req_m) for authentication for safe data-sharing following a specific group certification. The GA initially checks identities by calculating ts and checking hash values after receiving DU's (reg_m). GA adds authenticate DU's data after verification is complete to resist dictionary and passive attacks and removes extra primary security credentials from the routing table. GA calculates a secure session key using the received parameters to share data with group users for a predetermined time window or necessary session. Using the secure channel, GA delivers the secret key to the DU and validates with previously transmitted private credentials and verified IDs for authorized DU.

Algorithm 5 Group User Authentication

```
 1: $DU \xrightarrow{M} GA$: $(DU_{id}, req_m, DU_{ts}, H(Q_r(y)))$
 2: for $(DU^{'}_{ts} - DU_{ts} \leq \Delta ts$ and $H(req_m) == H^{'}(req_m))$ do
 3:     GA verifies DU authenticity from the respective FS
 4:     FS add DU in to the requested group
 5:     GA calculate $ssk : (H(GA_{id}) \oplus H(DU_{id}) \oplus req_m \oplus DU_{ts} \oplus Q_r(y))$
 6:     GA transmit the $ssk$ to DUs for each session-based data-sharing
 7: end for
```

- Decrypt:
  The GA runs the decrypt algorithm and accesses the $CT^{'}$ at this step. Only when the shared ciphertext's properties comply with the user and file specific access policies can the GA decrypt it. In the meantime, the GA with file-specific attributes that adhere to the access policy generated by the FS can

decrypt $CT^{'} \rightarrow CT$. When a GA has a user-specific attribute that matches the access policy created by the DO, the GA can use its secret key to decrypt the CT and obtain the plaintext. To decrypt CT, a GA with attribute set S requires the public key (PK) and a secret key (SK).

$$\begin{aligned} CT &= Decrypt(CT^{'}, \mathbb{A}_q, DU_{sk}, DU_{pk}) \\ M &= Decrypt(CT, \mathbb{A}_p, DU_{sk}, DU_{pk}) \end{aligned} \tag{14}$$

4.3 Cryptanalysis Discussion

- DoS and DDoS Attack:
  A DoS attack aims to take down a computer system or network so its targeted recipient cannot reach it. DoS attacks achieve this by providing the victim with excessive traffic or data that causes a collapse. The DoS attack strips both times' genuine consumers of the service or resource they anticipated. DoS attacks can take the victim a lot of time and effort to deal with, even while they usually do not lead to the theft or loss of crucial data or other resources. The Distributed Denial of Service (DDoS) assault is another DoS attack. When several systems coordinate a synchronized DoS attack on a single target, the result is a DDoS attack. The proposed work leverages time-specific session-based data-sharing to prevent DOs and DDoS attacks.
- Sybil Attack:
  In cybersecurity, a reputation system is undermined by generating several identities in the Sybil attack. The ease of users with which identities can be used, the extent to which the reputation system accepts input from users, and whether the reputation system treats all users equally determine how vulnerable a reputation system is to a Sybil attack. The values of $H(Q_r(y))$ will change in the proposed work if an unauthorised user tries to enter the session. After that, the entire authentication message is changed. Access will only be granted by GA if $H(req_m) = H^{'}(req_m)$ as shown in Algorithm- 5.
- Reply Attack:
  A replay attack occurs when an attacker listens on a secure network connection, detects it, and then purposefully freezes or re-sends it to trick the recipient into performing what the attacker wants. In the proposed work, user validation and session creation only occur within a predetermined time window. If there is a delay in the data transfer, FS or GA has the right to terminate the session and deactivate the user. The encrypted message $E_{FS_{pk}}(GA_{msg})$, according to the Algorithm- 3, will be modified and transmitted to the FS if an attacker attempts to change the random value while accessing the session as GA. When FS attempts to calculate $Q_{rs}(x) == Q_r(Q_s(x))$ as per the Algorithm- 5, it is unable to do so and recognizes that the message is tainted. The request can be immediately rejected and blocked by FS. Without FS' approval, GA will not supply a DU id if Eve attempts to pose as a DU. Since the adversary cannot attack the strategy to replay the message in session-enabled data-sharing, the proposed work is secure from a reply attack.

- Secret key and Attribute guessing Attacks:
  The user's secret key DU_{sk}, which is generated from both user and file specific attributes, is the only secret on the part of the DU. The secret key is strong and has sufficient bits to withstand brute force attacks in the event that a server is compromised. Even if there is a probability of collecting user-specific attributes, it can be challenging for non-expert people to get file-specific attributes. As a result, the proposed work is resistant to attacks that try to guess a secret key or attribute.
- Modify Attack:
  The message $(DUid, reqm, DUts, H(Qr(y)))$ delivered by the registered user has a signature req_m. the GA can determine whether the message gets modified by confirming the req_m and user timestamp DU ts as indicated in Algorithm- 5. Since the adversary cannot attack the technique to alter the message, the proposed work is safe from modification attacks. If shared data is moderated by a user simultaneously, the revised version will appear on the relevant FS. The DO can determine whether or not a registered authority gets altered.
- Man-in-the-Middle Attack:
  Considering a threat that positions itself as a node in the middle of the FS and DU networks. This becomes new to DU_{sk} which is generated from the user and file specific attributes. This threat may attempts to misrepresent each party to the other. Then the opponent cannot determine the secret ssk, created locally by the GA and stored in FS as indicate in Algorithm- 5, because they need to learn DU_{sk}. The adversary cannot produce good encryption by concurrently guessing the user and file specific attributes. Thus, the man-in-the-middle attack is unsuccessful.

5 Experimental Results and Findings

Simulated and compared it to the [34], [35] and [36] schemes to expressly demonstrate the effectiveness of the proposed work. It uses the Charm cryptographic framework for the implementation. Charm offers a comprehensive set of cryptographic tools that can be used to build a variety of applications such as secure communication systems, data storage systems, and authentication systems. The "SS512" curve, a super-singular symmetric elliptic curve over a 512-bit base field with a 160-bit curve group order, was employed in the proposed work.The CP-ABE scheme pairing-based cryptography (PBC) library is the foundation for the system's implementation. The Flask framework was taken into consideration for the REST API implementation between fog servers. Traffic control tool sets the maximum rate of data transmission for the network interface eth0 to 1000 kilobytes per second (kbps) and 100GB storage space. Numpy library used for Chebyshev chaotic map based hash generation. "Hospital Triage and Patient History Dataset" [37] is considered for the experiment in the proposed work.

The experimental results shown in figure- 9 implies a clear relationship between the size of the attribute universe and the setup computation time for various schemes ([34], [35], [36]). The computation times for all systems are constant to the number of attributes. It means that larger attribute universes do not impact the system setup time. Comparing the proposed work to the Li et al. [34], Hur et al. [35], and Jang et al. [36] schemes shows that the proposed work generates more setup calculation time. The proposed work takes into account the list of attributes from 5 to 25. Setup time takes 120 ms with 5 attributes as well as 25 attributes. Even though this would initially appear to be a drawback, the extra parameters brought on by the proposed work might increase security in data processing regions. On the other hand, the [34] scheme has a lower computational time as it generates the least number of parameters during setup. It may make it a more appealing option for applications with limited computational resources. Still, it's essential to consider if the reduced number of parameters could negatively impact system security. According to the experiment's findings, the setup phase is unaffected by scaling the attribute universes.

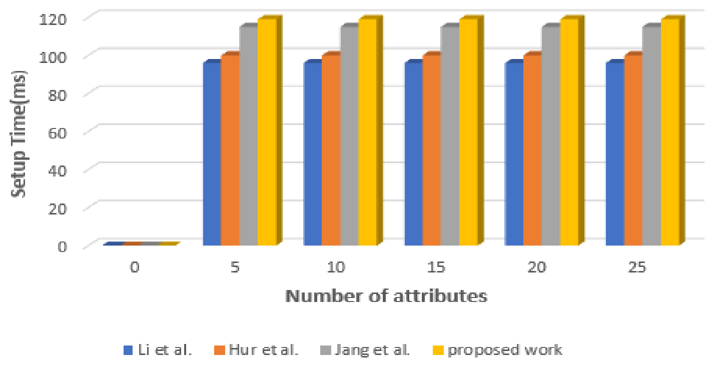

Fig. 9: System setup time with respect to number of attributes

Figure- 10 shows that the computation time required for key generation to be influenced by the number of user attributes used in the process. Key generation time is linearly dependent on the number of attributes in the attribute universe. As depicted in the figure, the proposed work has a computation time comparable to that of the [34] scheme. However, the [35] and [36] methods perform better in computation time during key generation due to their minimal exponentiation operation requirements. These results suggest that using a smaller number of user attributes can result in faster and more efficient key generation, as seen in the case of the [35] and [36] scheme.

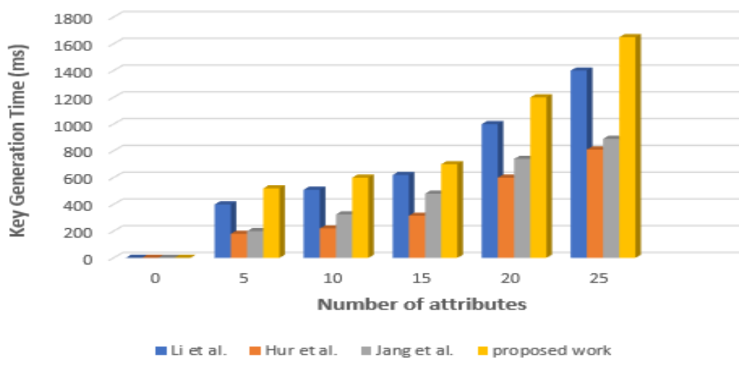

Fig. 10: Key generation time with respect to number of attributes

Figure- 11 compares the preceding DO encryption time variation with the number of ciphertext attributes. The computation time for the proposed work and the [35] technique varies greatly about the fluctuating ciphertext attribute number during regional encryption. The encryption time of the [34] and [36] methods is approximately constant with the degree of attributes in the ciphertext. The proposed work is superior to the [35] scheme; however, computation time grows as the number of ciphertext attributes during encryption increases. For each cycle, the proposed work took the 100MB ciphertext into account.

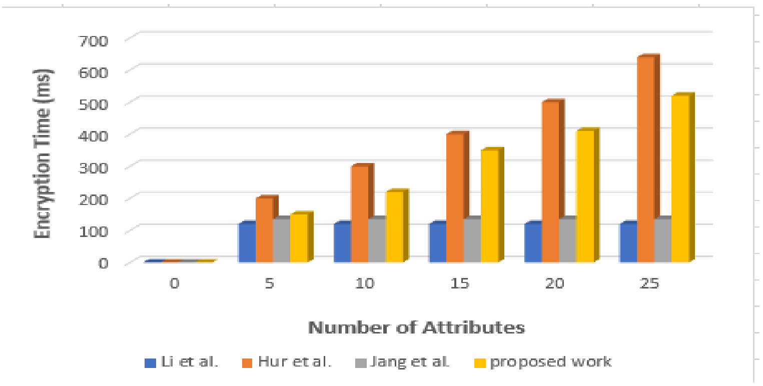

Fig. 11: Encryption time with respect to number of attributes

In figure- 12, the re-encryption time of the proposed work compares to that of other techniques, [34] and [35]. The graph shows that as the number of attributes in the ciphertext increases, the re-encryption time of the proposed work remains relatively stable. In contrast, the re-encryption time of [34] increases significantly. The FS has a faster computation speed than the DO used in the proposed work. The FS can re-encrypt the ciphertext much more quickly, leading to a significant reduction in re-encryption time compared to [34]. This result suggests that the FS is a more efficient and effective solution for re-encryption, mainly when dealing with ciphertexts with many attributes.

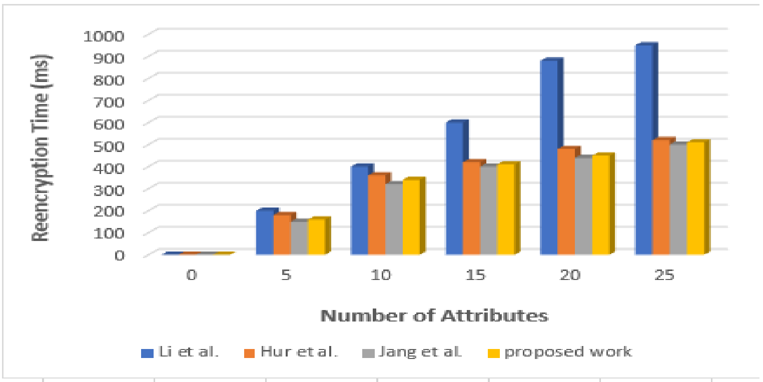

Fig. 12: Re-encryption time with respect to number of attributes

Figure-13 plots the computation duration for decryption against the various variables used in the decryption process. Decryption time includes both FS decryption and DU decryption. Except for the [35] scheme, which does not perform cloud decryption, the computation time for FS decryption grows as the number of attributes involved in decryption increases. Aside from the [35] method, all schemes display constant computation times during the local decryption. The proposed work has a lower calculation time than the [34] scheme because it uses fewer multiplication and exponentiation operations for local decryption.

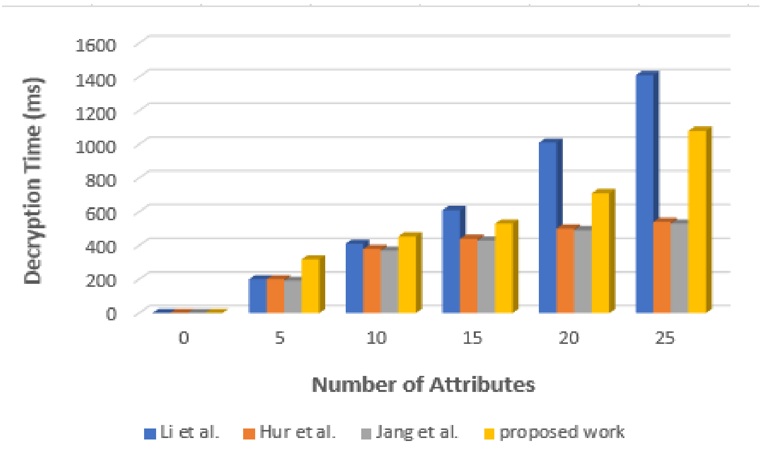

Fig. 13: Decryption time with respect to number of attributes

6 Conclusion

The proposed work focused on addressing the challenges of medical data-sharing in a fog-cloud environment using both user and file specific CP-ABE scheme. In order to provide secure data-sharing, proposed work emphasises the following crucial elements: limitlessness, flexibility, efficiency, and collusion resistance. The proposed work uses data users' and file attributes to implement secure fine-grained access control. Additionally, using Chebyshev's chaotic map functions for mutual authentication security provides a cutting-edge method for session-based data-sharing that reduces the risks involved during data transmission. By reducing the risk of data loss or theft, the proposed work style will increase the security of medical data-sharing. There are suggestions for future improvements, including incorporating threat analysis using blockchain technology and policy concealing. That would allow for a more secure and efficient way of managing medical data-sharing in the fog-cloud environment while ensuring the privacy and security of sensitive medical data.

Statements and Declarations

6.1 Ethical Approval and Consent to participate

Authors are responsible for correctness of the statements provided in the manuscript.

6.2 Human Ethics

Not Applicable

6.3 Consent for publication

Additional informed consent was obtained from all individual participants for whom identifying information is included in this article.

6.4 Availability of supporting data

Not Applicable

6.5 Competing interests

The authors have no relevant financial or non-financial interests to disclose.

6.6 Funding

The authors did not receive support from any organization for the submitted work.

6.7 Authors' contributions

Conceptualization: [Thushara G.A],[S.Mary Saira Bhanu]; Methodology: [Thushara G.A],[S.Mary Saira Bhanu]; Formal analysis and investigation: [Thushara G.A]; Writing - original draft preparation: [Thushara G.A]; Writing - review and editing: [Thushara G.A],[S.Mary Saira Bhanu]; Resources: [Thushara G.A],[S.Mary Saira Bhanu]; Supervision:[S.Mary Saira Bhanu].

6.8 Acknowledgements

Not Applicable

6.9 Authors' information

*[1]Thushara G.A
[2]S.Mary Saira Bhanu

Appendix A Not applicable

References

[1] A. Iera L. Atzori and G. Morabito. The internet of things: A survey. *Computer Networks*, 54:2787–2805, 2010.
[2] Elizabeth Snell and Kyle Murphyal. Malware most common smart hospital data security threat.
[3] D Chen L Wang, Y Ma Y Hu, and J Wang. Towards enabling cyber-infrastructure as a service in clouds. *Comput. Electr. Eng.*, 39:3–14, 2013.
[4] L.M. Kaufman. Data security in the world of cloud computing. *IEEE Secur. Priv*, 7:61–64, 2009.
[5] D. Lekkas D. Zissis. Addressing cloud computing security issues. *Future Gener. Comput. Syst.*, page 583–592, 2012.
[6] J.J. Yang, J.Q. Li, and Y. Niu. A hybrid solution for privacy preserving medical data sharing in the cloud environment. *Future Generation Computer Systems*, 43:74–86, 2015.
[7] A. Sahai and B. Waters. Fuzzy identity based encryption. *In Advances in Cryptology – Eurocrypt*, 3494:457–473, 2005.
[8] A. Sahai J. Bethencourt and B. Waters. Ciphertext-policy attribute-based encryption. *Security and Privacy*, pages 321–334, 2007.
[9] Yongfeng Qian Min Chen, Kai Hwang Jing Chen, and Long Hu Shiwen Mao. Privacy protection and intrusion avoidance for cloudlet-based medical data sharing. *IEEE Trans. Cloud Comput*, 43:74–86, 2016.
[10] Xiong Li Shaobo Zhang, Tao Peng Zhiyuan Tan, and Guojun Wang. A caching and spatial k-anonymity driven privacy enhancement scheme in continuous location-based services. *IEEE Trans. Cloud Comput*, 43:74–86, 2016.

[11] JianQiang Li JiJiang Yang and Yu Niu. A hybrid solution for privacy preserving medical data sharing in the cloud environment. *Fut. Gener. Comput. Syst.*, 43:74–86, 2015.

[12] Rongxing Lu Cheng Huang, Jun Shao Hui Zhu, and Xiaodong Lin. Fssr: Fine-grained ehrs sharing via similarity-based recommendation in cloud-assisted ehealthcare system. In *In Proceedings of the Annual Conference of the ACM ASIA Conference on Computer and Communications Security (AisaCCS'16)*, pages 95–106, 2016.

[13] Tianqi Zhou Jian Shen, Jin Li Xiaofeng Chen, and Willy Susilo. Anonymous and traceable group data sharing in cloud computing. *IEEE Trans. Inf. Forens. Secur.*, 13:912–925, 2017.

[14] YanKit Li Jin Li and Wenjing Lou. Xiaofeng Chen, PatrickPC Lee. A hybrid cloud approach for secure authorized deduplication. *IEEE Trans. Parallel Distrib. Syst.*, 26, 2015.

[15] Xuemin Shen Kan Yang, Zhen Liu adn Xiaohua Jia. Time-domain attribute-based access control for cloud-based video content sharing: A cryptographic approach. *IEEE Trans. Multimed.*, 18:940–950, 2016.

[16] Yinghui Zhang Jin Li and Yang Xiang Xiaofeng Chen. Secure attribute-based data sharing for resource-limited users in cloud computing. *Comput. Secur*, 72:1–12, 2018.

[17] Zheng Qin Hui Yin and Keqin Li Lu Ou. A query privacy-enhanced and secure search scheme over encrypted data in cloud computing. *J. Comput. Syst.*, 90:14–27, 2017.

[18] Rong Yu Jiawen Kang, Maoqiang Wu Xumin Huang, and Yan Zhang. Sabita Maharjan, Shengli Xie. Blockchain for secure and efficient data sharing in vehicular edge computing and networks. *IEEE IoT J*, 6:4660–4670, 2018.

[19] ShermanSM Chow ChengKang Chu and RobertH Deng. WenGuey Tzeng, Jianying Zhou. Key-aggregate cryptosystem for scalable data sharing in cloud storage. *IEEE Trans. Parallel Distrib. Syst*, 25:468–477, 2014.

[20] Junwei Zhou Shulan Wang, Jianping Yu Joseph K. Liu, and Weixin Xie. Jianyong Chen. An efficient file hierarchy attribute-based encryption scheme in cloud computing. *IEEE Trans. Inf. Forens. Secur*, 11:1265–1277, 2016.

[21] Shucheng Yu Ming Li and Wenjing Lou. Yao Zheng, Kui Ren. Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption. *IEEE Trans. Parallel Distrib. Syst*, 24:131–143, 2013.

[22] Yi Liu, Yinghui Zhang, and Zhusong Liu. Jie Ling. Secure and fine-grained access control on e-healthcare records in mobile cloud computing. *Future Generation Computer Systems.*, 78:1020–1026, 2018.

[23] L. Zhang. Cryptanalysis of the public key encryption based on multiple chaotic systems. *Chaos, Solitons and Fractals.*, 37, 2008.

[24] Han Qiu, Meikang Qiu, Meiqin Liu, and Gerard Memmi. Secure health data sharing for medical cyber-physical systems for the healthcare 4.0.

IEEE Journal of Biomedical and Health Informatics, 24(9):2499–2505, 2020.

[25] Khaled Riad, Rafik Hamza, and Hongyang Yan. Sensitive and energetic iot access control for managing cloud electronic health records. *IEEE Access*, 7:86384–86393, 2019.

[26] Min Chen, Yongfeng Qian, Jing Chen, Kai Hwang, Shiwen Mao, and Long Hu. Privacy protection and intrusion avoidance for cloudlet-based medical data sharing. *IEEE Transactions on Cloud Computing*, 8(4):1274–1283, 2020.

[27] Si Han, Ke Han, and Shouyi Zhang. A data sharing protocol to minimize security and privacy risks of cloud storage in big data era. *IEEE Access*, 7:60290–60298, 2019.

[28] Dong Zheng, Axin Wu, Yinghui Zhang, and Qinglan Zhao. Efficient and privacy-preserving medical data sharing in internet of things with limited computing power. *IEEE Access*, 6:28019–28027, 2018.

[29] Hui Ma, Rui Zhang, Guomin Yang, Zishuai Song, Kai He, and Yuting Xiao. Efficient fine-grained data sharing mechanism for electronic medical record systems with mobile devices. *IEEE Transactions on Dependable and Secure Computing*, 17(5):1026–1038, 2020.

[30] Y. Sreenivasa Rao. A secure and efficient ciphertext-policy attribute-based signcryption for personal health records sharing in cloud computing. *Future Generation Computer Systems*, 67:133–151, 2017.

[31] Yuanfei Tu, Geng Yang, Jing Wang, and Qingjian Su. A secure, efficient and verifiable multimedia data sharing scheme in fog networking system. *Cluster Computing*, 24(1):225–247, 2021.

[32] Ahmed Saidi, Omar Nouali, and Abdelouahab Amira. Share-abe: an efficient and secure data sharing framework based on ciphertext-policy attribute-based encryption and fog computing. *Cluster Computing*, 25(1):167–185, 2022.

[33] Yang Zhao, Xing Zhang, Xin Xie, Yi Ding, and Sachin Kumar. A verifiable hidden policy cp-abe with decryption testing scheme and its application in vanet. *Transactions on Emerging Telecommunications Technologies*, 33(5):e3785, 2022.

[34] Jiguo Li, Wei Yao, Yichen Zhang, Huiling Qian, and Jinguang Han. Flexible and fine-grained attribute-based data storage in cloud computing. *IEEE Transactions on Services Computing*, 10(5):785–796, 2017.

[35] Junbeom Hur and Dong Kun Noh. Attribute-based access control with efficient revocation in data outsourcing systems. *IEEE Transactions on Parallel and Distributed Systems*, 22(7):1214–1221, 2011.

[36] Kim JW Edemacu K, Jang B. Cescr:cp-abe for efficient and secure sharing of data in collaborative ehealth with revocation and no dummy attribute. *PLoS One*, 16(5), 2021.

[37] Adrian Daniel Haimovich Woo Suk Hong and R. Andrew Taylor. Hospital triage and patient history data. https://www.kaggle.com/datasets/maalona/hospital-triage-and-patient-history-data. Accessed: 2017.