International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Traitement du Signal
Vol. 42, No. 2, April, 2025, pp. 915-931
Journal homepage: http://iieta.org/journals/ts

Enhancing Cloud Data Security Through Functional-Based Stream Cipher and Attribute-Based Access Control with Multiparty Authorization

Keshav Sinha[1*], Saad Aldosary[2], Moustafa H. Aly[3], Walid El-Shafai[4]

[1] School of Computer Science, University of Petroleum and Energy Studies, Dehradun 248007, India
[2] Department of Computer Science, Community College, King Saud University, Riyadh 11437, Saudi Arabia
[3] Electronics and Communications Engineering Department, College of Engineering and Technology, Arab Academy for Science, Technology and Maritime Transport, Alexandria 1029, Egypt
[4] Department of Electronics and Electrical Communications Engineering, Faculty of Electronic Engineering, Menoufia University, Menouf 32952, Egypt

Corresponding Author Email: keshav.sinha@yandex.com

https://doi.org/10.18280/ts.420227

Received: 21 January 2024
Revised: 31 May 2024
Accepted: 20 August 2024
Available online: 30 April 2025

Keywords:
functional-based stream cipher (FBSC), distributed authorization, attribute-based encryption (ABE), NIST, Lagrange interpolation, involutory function

ABSTRACT

In the present-day context, data owners store and share sensitive records on cloud servers to ensure the confidentiality and integrity of their data. There are challenges related to storing data on cloud servers, including ensuring data security, privacy, compliance, and data sovereignty concerns. Another problem is data request frequency (increasing the server's computational overhead). An Attribute-Based Access Control Scheme (ABCS) was implemented, allowing authorized users secure access to data from the cloud server. The proposed work is divided into two objectives: The first objective is enhancing data security through an XOR and Functional-Based Stream Cipher (FBSC) for secure storage and sharing. The second objective entails leveraging data owner attributes to create a polynomial for distributing the symmetric secret key. The Symmetric Secret Key is segmented into 'n' shares using Shamir's Secret Sharing Scheme to provide the multiparty Secret Key Points, which are then shared among the authorized users via asymmetric encryption. The Attributed-based hierarchical tree structure scheme AB-HTS-S stores the Secret Key Points. Lagrange interpolation is used to reconstruct symmetric secret keys and provide access to privileged users. AB-HTS-S scheme defines an authorized user threshold ($T \geq 3$) to reconstruct the symmetric secret key for decryption. Data encryption is evaluated using statistical methods such as the NIST Statistical Test Suite, Correlation Coefficient, and Histogram analysis. Performance analysis, a key aspect of our research, demonstrates that our proposed scheme offers significant computational efficiency, ensuring rapid encryption/decryption and high throughput. The experimental results show that our scheme requires minimal storage and communication overhead. Security analysis proves that our scheme resists collusion and chosen plaintext attacks. Therefore, the proposed schemes can offer secure and efficient mechanisms for cloud storage, instilling confidence in its practical application.

1. INTRODUCTION

In the modern era, the cloud has become the potential destination for data storage. At the same time, handling data has become challenging for any large organization. Opting for remote storage services gives individuals access to nearly limitless storage space, which can result in substantial savings on data storage and management expenses [1]. The convenience of uploading and retrieving files from any location via the internet is a significant advantage of cloud storage. This flexibility has driven more users to deploy their applications on cloud-based servers. In a typical cloud storage setup, the responsibility for safeguarding data lies with the cloud servers. However, these servers are sometimes managed by cloud service providers who may not be fully trustworthy, posing a potential risk of private data being accessed by unauthorized entities. To mitigate this vulnerability, it is common practice for users to encrypt their files before uploading them to the cloud. This encryption helps ensure the security of their data, providing an additional layer of protection against unauthorized access. Data owners often outsource their massive chunks of data to cloud servers using pre-designed Access Control Policies (ACP) [2]. In a cloud environment, data owners have limited control over their stored data. To address issues related to authorization, confidentiality, and integrity, both ACPs and cryptographic algorithms are utilized. ACPs and cryptographic algorithms provide a robust framework for protecting sensitive information in cloud storage environments. The traditional cryptosystem uses complex key management such as (Random Number Generation (RNG), Key Derivation Functions (KDF), Key Exchange Protocols, Key Wrapping, and Key Pair Generation) with high storage overhead [3]. Researchers have utilized Attribute-Based Encryption (ABE) in response to encryption challenges. ABE empowers them to tackle encryption issues by enabling access control based on predefined attributes or characteristics, thus enhancing the flexibility and granularity of data protection. The ABE is the public-key cryptosystem technique that uses access control and authorization policy for data sharing. Sahai and Waters [4] explored the intricate interplay between users and stored data, shedding light on their relationship dynamics. The encryption scheme was formulated as a representation of the identity (I) and its ability or inability (θ) to operate on a Resource (R) within the ecosystem, encompassing all possible combinations of 'I', 'θ', and 'R' [5]. ABE has several disadvantages, such as substantial computation costs, encryption/decryption time, and key management [6]. The fined-grained Access Control List (ACL) is used to overcome the mentioned problem. In today's scenario, few systems have been implemented based on ACL and ABE in cloud environments [7]. However, using ABE provides flexibility but increases overhead during multi-linear mapping and public key transmission [8]. This study aims to minimize encrypted data access and solely provide the decryption key to authorized individuals. The system is designed based on three factors: (i) Key Management, (ii) Authorization Policy, and (iii) Privacy Prevention. The key management works on the Access Control List (ACL), which uses the functional-based Stream cipher for data encryption/decryption and improved RSA uses for secure key sharing. The authorization policy uses simple scalar points over a parabolic curve for key generation. The access control policy will be updated at every ΔT sec to verify additional attributes, which helps protect confidential data leakage [9].

1.1 Main idea

The proposed system integrates an authorization and access control scheme to enhance the security of data storage and sharing. Using a Functional-Based Stream Cipher (FBSC), data is encrypted and stored on an organization's server. The distribution of the symmetric secret key relies on data owner attributes to construct the polynomial. Shamir's Secret Sharing divides the symmetric secret key into 'n' shares. Asymmetric encryption is used to exchange these secret shares among multiple privileged data users to maintain the confidentiality and integrity of the secret. Secret key points are reconstructed using Lagrange interpolation, ensuring that users can only access and decrypt data if they possess the necessary access privileges. The system employs a threshold value (k=3), where the Organization server share, Data Owner share, and Data User share are used to reconstruct the symmetric secret key, each bound to a polynomial. For example, a healthcare organization adopts the proposed system to store and share sensitive patient records among its staff securely. Using a Functional-Based Stream Cipher (FBSC), the organization encrypts patient data before storing it on the server. The symmetric secret key for encryption is divided into 'n' shares using Shamir's Secret Sharing based on attributes such as staff roles and departments. Asymmetric encryption ensures secure exchange of these shares among senior doctors, department heads, and IT administrators. To access the encrypted records, the system employs a threshold value (k=3), requiring the shares from the organization server, a data owner (doctor), and a privileged data user (nurse) to reconstruct the key using Lagrange interpolation. This process guarantees that only authorized users with the necessary access privileges can decrypt and access the patient records, maintaining data security and confidentiality.

1.2 Main contributions

Implementation of an authorization and access control scheme combined with a Functional-Based Stream Cipher (FBSC) to ensure secure data storage and sharing.

Use a Functional-Based Stream Cipher (FBSC) to encrypt data before storing it on the organization's server.

Creation of secret key shares from symmetric secret key using Shamir's Secret Sharing scheme.

Utilization of asymmetric encryption to exchange secret key shares among privileged users.

Application of Lagrange interpolation to reconstruct the symmetric secret key.

Only users with the appropriate access privileges can access and decrypt the data.

1.3 Organization of this paper

The paper is organized as follows: Section 2 reviews access control schemes for cloud data storage and permission revocation of privileged users. Section 3 introduces the mathematical foundation, including bilinear map and language interpolation. Section 4 presents the proposed system model for functional-based encryption, Secure key sharing, privileged uses, Access control list, and Data user authorization. Section 5 presents the performance and security analysis. Finally, we summarize the entire work and future scope in Section 6.

2. ACCESS CONTROL SCHEME AND REVOCATION

In a cloud environment, the role of the cloud service provider (CSP) is to manage the storage and sharing of data. Data security is determined based on cryptographic techniques and Access control mechanisms [10]. An encryption technique is used to encrypt data and securely store it on a server. The encryption key is only shared with authorized users [11]. On the other hand, access control schemes ensure that only authorized individuals will gain entry or interact with specific resources and restrict access to unauthorized users.

2.1 Access control on encrypted data

An access control scheme is implemented on encrypted data to restrict unauthorized access. The problem with this approach is trust and the relationship between data owner and users. Trust and relationships change dynamically in different scenarios, and to maintain the desired level of security, the decryption key must be updated regularly [12]. The sharing of the symmetric key with all participants is granted based on the access control list (ACL). However, this method has drawbacks, such as (i) where the data owner creates various groups to share data and (ii) the size of the data owner's key increases exponentially [13]. The alternative approach is to combine symmetric and public-key cryptography. The data owner shares the public key with authorized users, and decryption is performed using the access control list [14]. However, as the number of users increases, it becomes expensive and time-consuming. The effective way is to encrypt data and provide the necessary keys to users at every ΔT time interval [15]. Attribute-based encryption (ABE) has recently undergone significant developments, where it recognizes users based on attributes rather than a single identity. In the ABE system, data is encrypted using a specified access control list. The authorization of users is established by aligning the attributes of the user with the ACL [16]. Recent literature demonstrates that Attribute-Based Encryption (ABE) is extensively used for secure cloud computing data storage. However, it is typically complex and requires substantial computation, making it challenging to implement in practical situations. To address this, Lai et al. [17] introduced Shamir's Secret Sharing algorithm to maintain confidentiality and integrity in a multi-cloud environment. It involves verification of each participant's secret share point, which is used for authorization.

Additionally, the hash-based signature shares generate the point for the authorized personnel. Building on this concept, Yang et al. [18] introduced a data access control scheme named DAC-MACS. It employs an efficient access control list to provide authorization for users and security for storing data in the cloud environment. The approach uses the decryption and revocation of the user's permission by using trust between the user and the data owner. DAC-MACS uses the independent global certificate authority (CA) and multiple attribute authorities (AAs) to authorize and identify global users and overcome the issue of collusion attacks.

2.2 Permission revocation for data owner

It refers to the process by which the original data owner can withdraw previously granted permissions to access, manipulate, or share that data. This concept is highly relevant to user authorization data security and privacy. It enables the data owner to respond to changing circumstances, such as changing user roles, discovering a security breach, or non-compliance with data use policies [19]. The data owner retains the right to revoke access granted to a user or application at any point. Revocation mechanisms can vary depending on the system's design. Some systems may implement immediate revocation, where access is revoked as soon as the command is issued, ensuring that unauthorized users cannot continue to access the data. Other systems might use periodic checks or time-based revocation, where access is revoked after a certain period or during regular intervals. These mechanisms can involve updating access control lists, redistributing encryption keys, or employing more complex cryptographic techniques such as proxy re-encryption, which allows a third party to revoke permissions without revealing the data.

The data owner withdraws access permissions from individuals in the access control list who no longer maintain group affiliation or due to a lack of trustworthiness on the user's part [20]. User revocation is a well-studied yet challenging problem. The main concern is that revoked users still have access to the previously mentioned keys, allowing them to decipher ciphertext. As a result, anytime a user is revoked, the data owner must perform re-keying and re-encryption activities to prevent the revoked user from accessing forthcoming data [21]. ABE uses data owner attributes to encrypt the data and distribute keys to authorized users regularly to facilitate this. However, this approach is inefficient due to the significant workload on the data owner [22]. An alternative, superior approach allows the data owner to assign computationally costly processes, such as re-encryption, to a third party while minimizing information exposure. The Proxy Re-Encryption (PRE) is a valuable option since it allows a semi-trusted proxy to transform a ciphertext that decodes into another ciphertext. It is decrypted without knowing the underlying data or using secret keys [23]. To integrate the Key Policy Attribute Based Encryption (KP-ABE) with PRE, we will outsource most calculation operations involved in user revocation to the CSP. It first integrates PRE with a CP-ABE system in the cloud to provide a scalable revocation mechanism. The attribute revocation is supported by the work stipulating that if a user is removed from a system, the data owner must transfer PRE keys to the CSP [24]. The biggest flaw with this solution is that it requires the data owner to be online to submit the PRE keys to the CSP on time, preventing the revoked user from accessing the data. The delay in issuing PRE keys might endanger system security [25]. To the best of our knowledge, there are trusted authorities at the local level for the authorization and revocation of permission of data users. The proposed model introduces the trusted authority center representing the data owner and in charge of re-encryption, authorization, and revocation. The trust-based insurance mechanism decreases the risks of data leakage. Meanwhile, Organization servers, owing to large-scale data handling, reduce the computational cost and data requests directly to the Cloud server, improving the efficiency of data storage and sharing. The proposed model uses the re-encryption process of encrypted data, whereas the re-encryption key is not stored in the cloud servers to reduce the storage and computational cost [26].

3. PRELIMINARIES

This section presents the mathematical understanding of bilinear maps and Lagrange interpolation, and Table 1 presents the various symbols used in this paper.

Table 1. Notations

ACL	Access Control List
C_{Server}	Cloud Server
D_i	Data
D_{Owner}	Data Owner
DUs	Data Users
Dx	Privileged Users
FK_i	File Authorization
$Organization_{Server}$	Organization Server
KC_i	File Binding Code
PK_{sk}	Symmetric Secret Key
Pb_{key}	Public Key
Pv_{key}	Private Key
R_n	Randomized Secret Power
RG_{Phase}	Rand Generation Phase
N_k	Generated Random Sequence
N_1	Fixed Padding
N_2	Repeated Padding
UT_i	User_Type
U_i	User_ID
UCi	User_Credentials
$Skey_{Points}$	Secret Key Points
TU	Trusted Authority
ΔT	Time Intervals

3.1 Bilinear map

A bilinear map $e: G \times G \rightarrow G_T$ with specific properties used for constructing the cryptographic protocols. The components and properties are as follows: Let $GP=\{e, G, G_T, g, p\}$ define the group parameters. Here, G and G_T are multiplicative cyclic groups with the same prime order 'p', g=a generator of the group G, e=the bilinear map, and p=prime order of the groups. The properties of bilinear map 'e' are:

(1) Bilinearity: For all $a, b \in Z_p$ and $g_1, g_2 \in G$. we have $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$. It means that the bilinear map is linear in both arguments. Specifically, if you scale the input by some scalars a and b, the output is scaled by the product ab.

(2) Non-degeneracy: For all $(g_1, g_2) \in G$, $e(g_1, g_2) \neq 1$ (the identity element of G_T). This property ensures that the bilinear map does not map every pair of elements to the identity element, meaning some pair (g_1, g_2) exist such that $e(g_1, g_2)$ is not the identity in G_T.

(3) Computability: For all $(g_1, g_2) \in G$, $e(g_1, g_2)$ can be computed in polynomial time. This means that there is an efficient algorithm for computing the bilinear map 'e' for any pair of elements from G.

3.2 Lagrange interpolation

Lagrange interpolation is a method to find a polynomial that passes through a given set of points. Given a set of points $\{(a_0, b_0), (a_1, b_1), \ldots, (a_n, b_n)\}$ where $(a_i \neq a_j)$ for $(i \neq j)$, and $(i, j) \in \{0, 1, \ldots, n\}$ the Lagrange interpolation polynomial $Q_n(x)$ is given by:

$$Q_n(x)=\sum_{i=0}^{n} \Delta_i(x) b_i \quad (1)$$

where, $\Delta_i(x)$ is the Lagrange basis polynomial, defined as:

$$\Delta_i(x) = \sum_{i=0}^{n} \left(\prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x-a_j}{a_i-a_j} \right) \times b_i \quad (2)$$

where, a_i=x-coordinates of the given points, b_i=y-coordinates (function values) of the given points and $\Delta_i(x)$ is the Lagrange basis polynomial associated with the i^{th} point in the set of $\in\{0, 1, \ldots, n\}$. If there are $(n+1)$ points, a unique polynomial of degree 'n' passes through all these points. Lagrange interpolation provides a direct way to construct this polynomial. The Lagrange interpolation formula gives an explicit polynomial for the given points, making it easy to understand and compute. Unlike other interpolation methods like Newton interpolation, Lagrange interpolation does not require solving a system of linear equations. It can simplify the computation and reduce potential numerical issues. Lagrange interpolation can be applied to any set of distinct points. It does not impose any constraints on the distribution of the points.

4. FORMAL DEFINITION

The system model comprises an Organization Server, cloud server, Trusted Authority, Rand Generation Phase, Data Owner, and Data User.

A Data Owner (D_{Owner}) is an organization's registered user who desires to store the data in a cloud server (C_{Server}). The D_{Owner} creates an Access Control List (ACL) and shares it with the Organization Server ($Organization_{Server}$). The owner is authorized to perform read/write and update/delete operations on the file and access control list (ACL). The D_{Owner} has attributes such as User_Id (Ui), User_Type (UT_i), and User_Credentials (UC_i). Data sharing requires authorization credentials, which are issued by the organization server.

A Cloud Server (C_{Server}) stores encrypted data and backup ACL. The D_{Owner} can upload the ACL and encrypted image to C_{Server}. It is considered a trusted platform for performing read/write operations on ACL at every encrypted data request. The organization server requires an ACL to authorize every user. The cloud uses the repository structure to organize the data files.

Organization Server ($Organization_{Server}$) is an authorization center between the D_{Owner} and Data User (DUs). It is responsible for file encryption and decryption using an Involutory function and Functional-Based Stream Cipher (FBSC) and generating shared authorization points using Shamir Secret Sharing. Afterward, D_{Owner} established the multiparty authorization key tree. Access to each file is linked to users' attributes, which are stored at C_{Server} and managed by $Organization_{Server}$. The work of $Organization_{Server}$ is to assign, revoke, and manage the User_Credentials (UC_i) from the ACL. It also records and indexes all the valid credentials to avoid redundancy. Finally, $Organization_{Server}$ sends the encrypted data, Symmetric Secret Key (PK_{sk}), and ACL to the C_{Server}.

Rand Generation Phase (RG_{Phase}) is the component of $Organization_{Server}$ that generates random bits to pad with the original file. The padding is performed when D_{Owner} uploads the file to $Organization_{Server}$. Optimal Asymmetric Encryption Padding (OAEP) is used to convert the deterministic encryption scheme into a probabilistic scheme and prevent the partial decryption of cipher data, and the adversary cannot invert the trapdoor function.

Trusted Authority (TU) is the $Organization_{Server}$ component used to generate the Symmetric and Asymmetric keys. The Symmetric Secret Key (PK_{sk}) is used in FBSC, and the Asymmetric key (Pb_{key} and Pv_{key}) is used to share secret key points ($Skey_{Points}$) with the DUs. It is considered a secure and authorized component of the proposed model.

Data Users (DUs) are registered users whose credentials are stored in the $Organization_{Server}$ access list. Suppose the user wants to access the shared file provided by the D_{Owner}. It required Secret points for $Organization_{Server}$ according to the attributes. Lagrange Interpolation is used to reconstruct the polynomial which holds the PK_{sk}. Then, DUs use the key to decrypt the encrypted file received from the C_{Server}.

Figure 1 presents the System model of the Attribute-Based Access Control Scheme and Functional-Based Stream Cipher (FBSC) for secure data storage and sharing. The proposed system consists of four participants: the Data Owner (D_{Owner}), Cloud Server (C_{Server}), Data User (DUs), Trusted Authority (TU), Rand Generation Phase (RG_{Phase}), and Organization Server ($Organization_{Server}$). Our model uses the symmetric encryption technique to encrypt the data (Di), whereas an asymmetric algorithm is used to secure PKsk sharing between $Organization_{Server}$ and DUs. The proposed system model has several steps:

Step 1: The D_{Owner} gets registered with $Organization_{Server}$. The sensitive credentials information like User_Id (U_i), User_Type (UT_i), and User_Credentials (UC_i) is stored in the $Organization_{Server}$ to maintain the ACL.

Step 2: The D_{Owner} shares the Data (D_i) with the organization server.

Step 3: $Organization_{Server}$ uses the RG_{Phase} and TU to generate the random bits and PK_{sk} for padding and encryption.

Step 4: The next step of $Organization_{Server}$ is to create the multiparty authorization key tree for PK_{sk} based on the user's attributes. Another work is to generate the Asymmetric key Pb_{key} and Pv_{key} for the secure sharing of secret key points ($Skey_{Points}$) and Randomized Secret Power (R_n). The distributor is the component in the proposed model that shares the secret key points ($Skey_{Points}$) with DUs.

Step 5: Data decryption is performed based on the reconstruction of PK_{sk} using Lagrange Interpolation. It also checks the user attribute from ACL for authorized users. The interpolation requires $Skey_{Points}$ from three different users: (i) $Organization_{Server}$ Point gets stored in $Organization_{Server}$ at the time of generation and sharing, (ii) D_{Owner} point is shared once the share is created for authorization of DUs, and (iii) DUs point is shared with multiple privileged users to reconstruct the polynomial.

Step 6: The PK_{sk} is used to decrypt and share the encrypted data with DUs.

It should be noted that the authorized users are only responsible for key corrections and modifications, and D_{owner} is responsible for maintaining the ACL in $Organization_{Server}$.

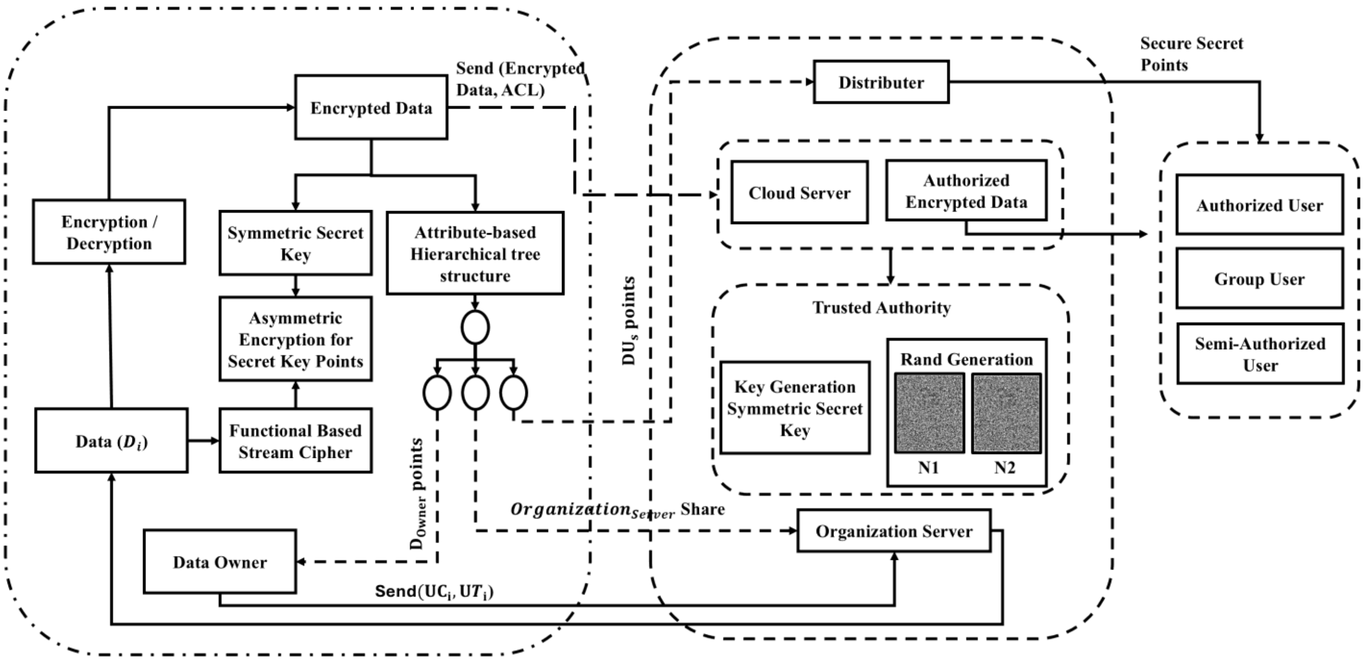


Figure 1. Proposed system model

4.1 System functionality

Here, we present the proposed system functionality of encryption and key sharing. The section is consist of (i) the Rand Generation Phase (RG_{Phase}) and (ii) Trusted Authority (TU).

4.1.1 Rand generation phase (RG_{Phase})

In RG_{Phase}, the random bits are generated using the KM Generator. The mathematical foundation has already been discussed in my previous work, Sinha et al. [27], and an extended application of the generator is used here for optimal asymmetric encryption padding, expressed as Eq. (3).

$$N_{k+1}=(N_k \times M \times I) \bmod n \tag{3}$$

where, I=Non-Integral Number, M=Maddy Constant, n=Moduli. The N_k generates the sequence that belongs to the I^+ range. The generated sequence is probabilistic and non-uniform. The fixed random bit [$N_k \rightarrow N_1$] and repeated random bit [$N_k \rightarrow N_2$] are generated based on the N_k. The D_{Owner} uploads the file to $Organization_{Server}$, at every ΔT time N_1 and N_2 is padded with D_i.

4.1.2 Trusted authority (TU)

The proposed model uses two distinct types of keys: (i) Symmetric Secret and (ii) Asymmetric key. The organization server uses the Attributed based hierarchical tree structure, where the tree leaf stores the Asymmetric key (Pb_{key} and Pv_{key}) information, and the node stores the Symmetric Secret key (PK_{sk}). The distributor component handles the secret key points ($Skey_{Points}$). The file system's directory structure is used to manage the multiparty authorization keys, whereas one key is used for one file.

4.2 Functional-based stream cipher (FBSC)

The data is encrypted using XOR using $RG_{Phase,}$ and then the Functional-Based Stream Cipher (FBSC) is on the data before being stored on the C_{Server}. The traditional cryptographic algorithm requires huge computational time, a complex permutation-combination system, and is vulnerable to cryptanalytic attacks. Here, the FBSC uses the Involutory function for encryption and decryption. An involutory function, as mentioned, is one were applying the function twice return the original value. Formally, a function 'f' is involutory if:

$$f(f(x))=x \tag{4}$$

where, ∀x in the domain of 'f'. Such a function is necessarily bijective, meaning it is both injective (one-to-one) and surjective (onto).

4.2.1 Bijective function

A bijective function f: A→B is a function where each element in set A is mapped to a unique element in set B, and each element in set B is mapped from a unique element in set

A. This means every element has a unique pre-image and image, ensuring a perfect pairing between the sets.

4.2.2 Anti-homomorphism

In the context of group theory, an anti-homomorphism is a map between two groups that reverses the order of the group operation. If G and H are groups, a map f:$G \rightarrow H$ is an anti-homomorphism if for all (A, $B \in G$):

$$f(AB)=f(\mathrm{B})f(\mathrm{A}) \tag{5}$$

The contrasts with a homomorphism, where the operation order is preserved:

$$f(AB)=f(A)f(B) \tag{6}$$

4.2.3 Involution and anti-involution

An involutory function 'f' satisfies:

$$f(f(x))=x \tag{7}$$

If we denote this function as:

$$f(x)=X \tag{8}$$

Then the inverse, also an involution, satisfies:

$$f(f(x))=X \tag{9}$$

An anti-involution (or anti-homomorphism) has the property that it reverses the order of application, which is a characteristic of anti-homomorphic functions. The proposed FBSC uses the mathematical foundation of involutory function, Bijective Function, and Anti-Homomorphism. The involution function takes the data as a stream for encryption and is represented as Eq. (10).

$$\text{Encryption\_Data}_\text{i} = f\ (x) = \left(PK_{sk} - {D_i}^{1/R_n}\right)^{R_n} \tag{10}$$

The $\text{Encryption\_Data}_\text{i}$ and User attribute is stored in the C_{Server}. The DUs retrieve the data from the C_{Server} for decryption. For decryption, we compute the anti-evolution function $f\,(f(x))$, which is generated by Eq. (11).

$$Original\_Data_i = f\big(f(x)\big) = \left(PK_{sk} - \left(\left(PK_{sk} - {D_i}^{1/R_n}\right)^{R_n}\right)^{1/R_n}\right)^{R_n} = x \tag{11}$$

where, PK_{sk}=Symmetric Secret Key and R_n=Randomized Secret Power, D_i is the original data where [i=number of data is uploaded to the $\text{Organization}_{\text{Server}}$]. The key storage system uses a multiparty authorization tree to manage keys. When a user changes a file name, the file's authorization key FKi changes accordingly. Eq 12 presents the authorization key transfer mechanism:

$$FK_i = \begin{cases} a_i \oplus x_i \\ a_i{}' \oplus x_i \end{cases} \tag{12}$$

where, a_i=Old secret key associated with the file, $a_i{}'$=New secret key associated with the file after the name change, x_i=full name of the file 'i' and FK_i is the authorization key for file 'i'.

4.3 Attribute-based multiparty authorization key

This section presents the implementation of Attribute-based multiparty authorization key computation using Shamir secret sharing. The distribution of each key uses the 2-degree langrage interpolation function over the finite field 'Z_p'. A finite field Z_p consists of the set of integers {0, 1, 2, …, $p-1$} with arithmetic operations (addition and multiplication) performed modulo a prime number 'p'. This field ensures that every non-zero element has a multiplicative inverse, making it suitable for cryptographic applications.

4.3.1 2-degree Lagrange interpolation

For our key distribution system, a 2-degree Lagrange interpolation means we are constructing a polynomial $P(x)$ of degree 2 that passes through three points (x_0, y_0), (x_1, y_1), (x_2, y_2). The polynomial is:

$$P(x) = y_0 l_0(x) + y_1 l_1(x) + y_2 l_2(x) \tag{13}$$

With the basic polynomials $l_i(x)$ defined as:

$$l_0(x) = \frac{x-x_1}{x_0-x_1} \times \frac{x-x_2}{x_0-x_2} \tag{14}$$

$$l_1(x) = \frac{x-x_0}{x_1-x_0} \times \frac{x-x_2}{x_1-x_2} \tag{15}$$

$$l_2(x) = \frac{x-x_0}{x_2-x_0} \times \frac{x-x_1}{x_2-x_1} \tag{16}$$

4.3.2 Shamir secret sharing (SSS)

It is an ideal cryptographic method for dividing a Symmetric Secret Key into multiple shares, ensuring that only a specified number of shares (threshold k) are needed to reconstruct the key. This method provides both security and fault tolerance. In a (k, n) threshold scheme, k=minimum number of shares required to reconstruct the Symmetric Secret Key, and n = total number of shares distributed. The symmetric secret key 'PK_{sk}' is divided into 'n' shares (PK_{sk_1}, PK_{sk_2}, …, PK_{sk_n}) pieces and distributed to privileged DUs. A polynomial of degree (k-1) is used to generate the shares. To divide the secret PK_{sk} into n shares such that any k or more shares can reconstruct the secret. To construct the polynomial $f(x)$ of degree (k-1) where the secret PK_{sk} is the constant term:

$$f(x)=a_0 + a_1 x + a_2 x^2 + \cdots + a_{k-1} x^{k+1} \tag{17}$$

where, a_0=PK_{sk} and {a_1, a_2, … a_{k-1}} are user attribute tokens. To compute the shares PK_{sk_i} by evaluating the polynomial at 'n' distinct non-zero points x_i.

$$PK_{sk_i} = f(x_i) \tag{18}$$

For i=1, 2, …, n. Now distribute the shares (x_i, PK_{sk_i}) to n users, ensuring that each user receives one unique share. The distribution of keys is based on the number of users with privileged access to $\text{Organization}_{\text{Server}}$.

Algorithm 1: Key Distribution

```
1: Start
2: Input:
3: k: threshold number of shares required to reconstruct
   the secret.
4: n: total number of shares to generate.
```

```
5:  $PK_{sk}$: The secret key to distribute.
6:  //Generate a Polynomial
7:  Choose a random polynomial $f(x)$ of degree $k-1$ where the constant term is the secret key $PK_{sk}$ and the other coefficients are randomly chosen.
8:  //Generate Shares
9:  for $i$ =1 to $n$
10:     Choose a random $xi$ in a finite field $Z_p$ where $p$ is a large prime number.
11:     Compute the corresponding $y_i=f(x_i)$.
12:     Share the $Skey_{Points}$ as $(x_i, y_i)$.
13: End
14: //Key Distribution Algorithm
15: $D_X$ = the number of privileged users: Define $D_X$ as the number of privileged users who will receive shares of the $Skey_{Points}$.
16: Point generation from polynomials: For each privileged user, '$i$' from 1 to $D_{X-1}$: Generate a point $(X, G(X))$ on a polynomial.
17: Point calculation: Calculate the $Y$-value for each point: $Y=G(X)$
18: $D_{X-1}$=(X, Y), where [X=1, 2, …, $D_X$]: Store the generated points as pairs $(X, Y)$ where $X$ ranges from 1 to $D_X$ (inclusive).
19: End
```

Each participant performs single point sharing $(X, G(X))$. Here the $Skey_{Points}$ start from the (1, G(1)) and not from (0, G(0)) because G(0) consists of secret key information [28-32]. The distributor component distributes the $Skey_{Points}$ only to the privileged users. The points are present in the hash formed (0, KC_i). Here, KC_i is the file binding code of the 'X_i', is calculated using Eq. (19).

$$KC_i = (PK_{sk_i}, f(x_i)) \bmod p \tag{19}$$

The file binding code KC_i can be used to associate 'x_i', to verify the integrity/authenticity of 'x_i' within a system. The points (0, KC_i) contain hash pairs of points where the first coordinate is always '0', and the second coordinate is the file binding code KC_i.

4.3.3 Symmetric secret key (PK_{sk}) reconstruction

To reconstruct the PK_{sk}, any 'k' users can use their shares to interpolate the polynomial $f(x)$ based on the Lagrange interpolation. The proposed method stores the $Organization_{Server}$, D_{Owner}, and DUs points. Given k shares (x_1, PK_{sk_1}), (x_2, PK_{sk_2}), …, (x_k, PK_{sk_n}). To compute the polynomial $f(x)$ by combining the k shares. The Lagrange basis polynomials $l_i(x)$ is given by Eq. (2). The polynomial f(x) can be reconstructed as Eq. (20).

$$f(x) = \sum_{i=1}^{k} PK_{sk_i} \times l_i(x) \tag{20}$$

The secret PK_{sk} is the constant team a_0 of the polynomial, which is $PK_{sk}=f(0)$. Let us consider PK_{sk} that needs to be divided into 5 shares (n=5) with a threshold of 3 (k=3). The polynomial is given as Eq. (21).

$$f(x) = a_0 + a_1X + a_2X^2 \tag{21}$$

where, $a_0=PK_{sk}$, and a_1, a_2 are UC_i of D_{Owner}, which are retrieved from the $Organization_{Server}$. To create the shares, evaluate $f(x)$ at five distinct points (eq. x=1, 2, 3, 4, 5). Provides each user with their corresponding share (x_i, PK_{sk_i}). Afterward, any three users can use their shares to perform Lagrange interpolations, reconstruct f(x), and retrieve PK_{sk}. Based on the ACL, the organization server maintains the multiparty authorization key tree for $Skey_{Points}$.

4.4 $Skey_{Points}$ sharing using asymmetric encryption

The secure sharing of key $Skey_{Points}$ and Randomized Secret Power (R_n) is performed based on the public key infrastructure (PKI).

4.4.1 Encrypted key sharing

In this context, the $Skey_{Points}$ are securely shared with participants using an asymmetric cryptosystem (specifically RSA) within a Public Key Infrastructure (PKI). The improved RSA cryptosystem ensures enhanced security through additional measures Sinha et al. [27].

Algorithm 2: $Skey_{Points}$ Sharing using RSA Cryptosystem

```
Step 1. The system generates two prime numbers, denoted as '$P_i$' and '$Q_i$', for each user have indexed by 'i'.
Step 2. The modulus '$m_i$' is calculated ($m_i = \sum_{i=0}^{n} P_i \times Q_i$).
Step 3. Calculate $\Phi(m)=(P-1)(Q-1)$
Step 4. $\lambda(m)$ represent the $\Phi(m)$.
Step 5. A value (e) is selected as:
            $(1 < e < \Phi(m))$ and gcd(e, $\Phi(m)$=1.
Step 6. $Pb_{key} \times Pv_{key} \equiv 1 \bmod \Phi(m)$
Step 7. $Enc_{key} = (R_n, SKey_{Points})^{Pb_{key}} \bmod n.$
```

where, $(Enc_{key}, m) \rightarrow Organization_{Server}$, for secure distribution to the DUs. The privileged user gets the secure key $Skey_{Points}$ and R_n points. Encryptions enhance the security of the key management process while sending it to the insured communication channel/network.

4.4.2 Decryption process

The decryption of the Enc_{key} is performed using the RSA and Chinese remainder technique to reduce the computation time of the decryption process. Consider the set of pairwise co-prime (m_1, m_2,…, m_k) and sequence of positive integer (a_1, a_2, …, a_k), (Sinha et al. [27]).

$$a^{(\Phi(\mathrm{m}))} = 1 \bmod \mathrm{N} \tag{22}$$

The standard solution of CRT is present in the form of ($X=X_0+kN$), where 'k' and $N=(n_1 \cdot n_2 \cdot \ldots \cdot n_r)$ (Sinha et al. [27]). The standard form of CRT is expressed as Eq. (23).

$$\mathrm{X} \equiv \sum_{i=0}^{m} a_i \bar{X_\iota} N_i \equiv 1 \bmod N \tag{23}$$

where, $\bar{X_\iota}$ is the modular inverse of 'X_i' is calculated as Eq. (24).

$$\bar{X_\iota} N_i \equiv 1 \bmod N \tag{24}$$

Then, $\left[N_i = \frac{N}{n_i}\right]$ is reduced as:

$$X = X_0 \bmod N \tag{25}$$

where, $[X_0 = a_i \bar{X_\iota} N_i]$ and $\Phi(m)$ is used for decryption. If 'm' is prime, then it is represented as Eq. (26) [27].

$$\Phi\,(\mathrm{m})=\mathrm{m}\left(1-\frac{1}{m}\right) \tag{26}$$

If 'm' is prime, then [$C^{\Phi(m)} \equiv 1 \bmod m$], which is expressed as Eq. (27) [27].

$$\begin{cases} D_p = Pv_{key}\ mod\ (p-1) \\ D_q = Pv_{key}\ mod\ (q-1) \end{cases} \tag{27}$$

where, 'p' and 'q' are relatively prime, reducing computational time.

$$\begin{cases} X_p = (Enc_{key})^{D_p}\ mod\ p \\ X_q = (Enc_{key})^{D_q}\ mod\ q \end{cases} \tag{28}$$

The unique solution of CRT-RSA is given as:

$$M \equiv \left[\left(X_p \times q \times q^{-1}\right) + \left(X_q \times p \times p^{-1}\right)\right] mod\ n \tag{29}$$

Here, $M \equiv Enc_{key}$ which is decrypted and gets $\mathrm{Skey}_{\mathrm{Points}}$ that are used in Lagrange basis polynomials to interpolate the polynomial $f(x)$.

5. PERFORMANCE AND SECURITY EVALUATION

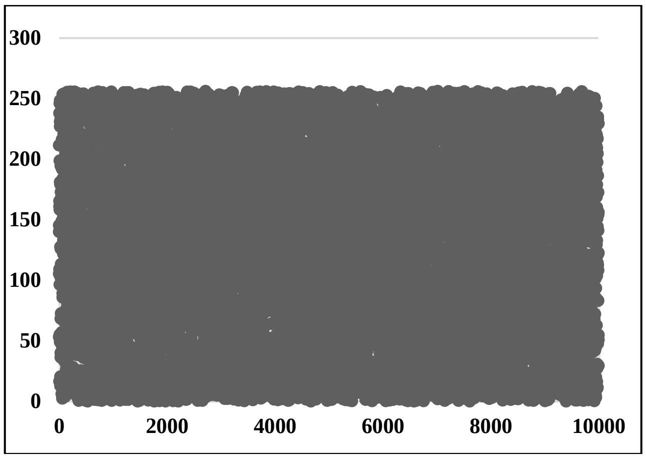
The section is divided into three sub-sections. Performance Analysis assesses the encryption/decryption process and overall system performance. Throughput measures the rate at which encrypted messages are successfully transmitted to the $\mathrm{Organization}_{\mathrm{Server}}$. Improving the throughput will enhance the overall performance during the encryption process. Statistical Analysis is conducted on an encrypted image to assess the uniformity and deviation between the image pixels. The tests have confirmed that the encrypted data is cryptographically secure and suitable for sharing and storage purposes. Security Analysis focuses on the Attribute-Based Access Control Scheme. The test is divided into two subsections: (i) computational overhead for authorization key generation for every user required to access and share files properly, and (ii) Security analysis is performed where an Adversary (A) performs two types of attacks: chosen plaintext attack and Collusion attack. The subsequent section evaluates the performance of the proposed method and compares it to similar work.

5.1 Performance analysis

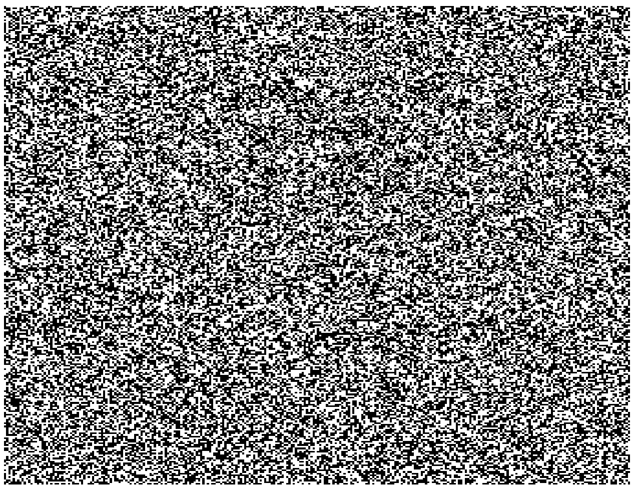
The proposed model is simulated on the Linux Mint platform, equipped with a 64-bit processor, 4GB of RAM, an Intel Core i5, and a single-core processor running at a CPU frequency of 1.70GHz. The organizational server setup utilizes the Apache server, allowing the data owner to upload and retrieve data, manage the access control list (including updates, deletions, and modifications), and generate random bits for encryption.

5.1.1 Rand generation phase (RGPhase)

This $\mathrm{RG}_{\mathrm{Phase}}$ uses a KM-Generator to generate cryptographically secure random bits [27]. Figure 2 represents the random bit generation, which is uniformly distributed throughout the region.



(a) Pictorial representation of random variable



(b) Pictorial pixel representation

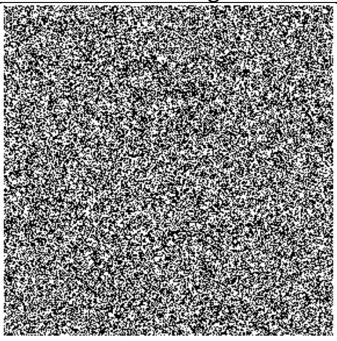
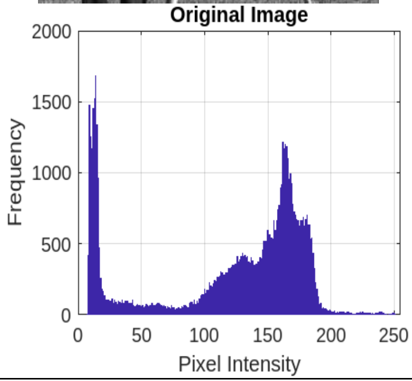
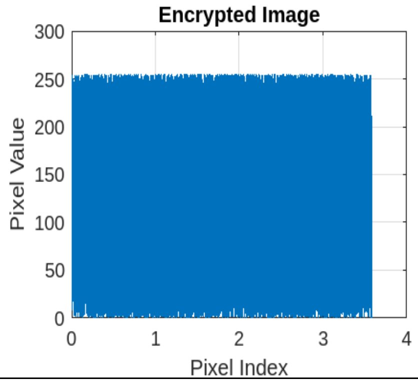
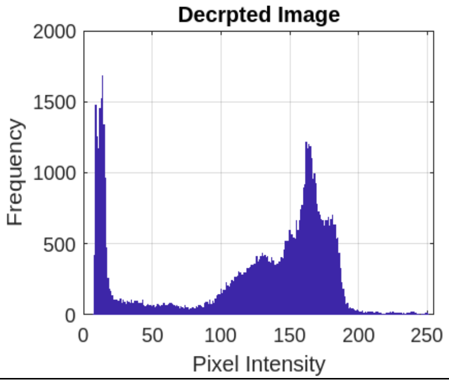
Figure 2. Generation of pseudorandom random bit

Figure 2(a) illustrates the binary output frame of (256×256) with randomness properties. Figure 2(b) shows the histogram plot of the generated sequence in (256×256) window frames. The pixel plot visualizes the uniform distribution of 0's and 1's and has no pattern in generation.

5.1.2 Evaluation of encryption/decryption

We examined the encryption time of image files (.tiff) using XOR and Functional-Based Stream Cipher. The image is downloaded from the website (https://links.uwaterloo.ca/Repository.html, accessed on 23-4-2024). The test was conducted on nine different images (tiff) (Lena, Barb, Boat, Goldhill, Mandrill, Mountain, Washsat, Peppers, and Cameraman). A random bit is generated and stored in the matrix 'A'. The random sequence matches the original image's matrix 'B' (256×256). The XOR is performed between matrix (A ⊕ B), and then the Functional-Based Stream Cipher is used for bit-level encryption. Table 2 represents the encryption and decryption of the cameraman image, and histogram analysis represents the uniformity of pixels. The proposed XOR and Functional-Based Stream Cipher (FBSC) performance was assessed by comparing the encryption times for cameraman images with those of traditional encryption algorithms. The results of this comparison can be found in Table 3.

Table 2. XOR and functional-based stream cipher encryption and decryption

Original Image	XOR Image	Decrypted Image
		
		

The concept of throughput is utilized to evaluate the efficiency of an algorithm during data transmission. Eq. (30) provides a guide to compute throughput; a higher value indicates superior performance.

$$\text{Throughput} = \frac{\text{Plaintext}}{\text{Encryption Time}} \tag{30}$$

In Table 4, a comparison of throughput reveals that the proposed XOR and FBSC techniques outperform other methods. It suggests that the new technique has a lower transfer rate and is suitable for integrating with Organization$_{\text{Server}}$.

Table 3. Encryption time comparison

Image	Technique	Time (Unit: Second)
Cameraman (256×256)	Sinha et al. [27]	0.006542
	Sun and Lv [28]	0.005362
	Arab et al. [29]	0.025781
	Proposed XOR and FBSC	0.001542

Table 4. Comparison of throughput

Algorithm	Throughput (kb/sec)
Sinha et al. [27]	183.34
Sun and Lv [28]	180.4
Arab et al. [29]	181.89
Proposed XOR and FBSC	184.67

5.2 Statistical analysis

Statistical Analysis is used to evaluate the uniformity of encrypted images.

Figure 3 represents the encryption and decryption process of the cameraman's image with a different set of keys. Figure 3(a) presents the original image. Figure 3(b) illustrates the encrypted image using the symmetric secret key 'K1'. Figure 3(c) represents the decrypted image using the symmetric secret key 'K1'. Now, the sensitivity of FBSC is evaluated by slightly changing the decryption key 'K2', presented in Figure 3(d). The re-encryption process with 'K2' is presented in Figure 3(e). The difference between the two different encryptions with 'K1' and 'K2' is present in Figure 3(f). It is observed that the proposed Function-Based Stream Cipher is extremely sensitive to the initial parameter for key generation and doesn't reveal any sensitive information. We execute the same operation with eight benchmark images (Lena, Barb, Boat, Goldhill, Mandrill, Mountain, Washsat, and Peppers), each with a resolution of (256×256) pixels.

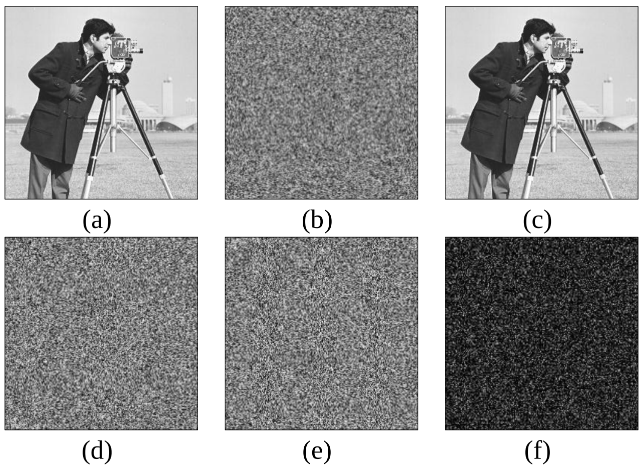


Figure 3. Image encryption using functional-based stream cipher

5.2.1 NIST (SP) 800-22 test

We evaluate the randomness of encrypted images using the NIST Special Publication (SP) 800-22 statistical test suite [30]. We transform encrypted image data into binary sequences and then administer a series of tests to confirm that the encryption process generates outputs practically indistinguishable from random sequences. The evaluation is essential for validating the efficacy and security of cryptographic algorithms employed in image encryption. The input parameter for the test is (i) 128 bits of block length for the block frequency test, (ii) 9 bits for the non-overlapping and overlapping test, (iii) 500 bits for Linear Complexity test, (iv) 16 bits of block length for Sequential test, and (v) 10 bits length for approximation of entropy test. The NIST SP 800-22 test suite offers a comprehensive set of statistical tests to evaluate the randomness of binary sequences, crucial for ensuring the security of cryptographic algorithms. The Frequency Test estimates the ratio of 0s to 1s in the generated stream, with significant deviations from a 50/50 ratio indicating non-randomness. The Block Frequency Test checks the number of 1s in fixed-size blocks, while the Cumulative Sum (Cusum) Test determines whether the sequence stays positive or negative for extended periods, signaling potential non-random behavior.

The Runs Test assesses the oscillation rate of 0s and 1s, ensuring the frequency of runs of consecutive 0s and 1s falls within expected limits. The Longest Run of 1s in a Block Test evaluates the length of the longest run of 1s in each block, comparing it to expected values to gauge randomness. The Rank Test checks for linear dependencies among fixed-length substrings by examining the ranks of matrices derived from the sequence.

The Discrete Fourier Transform (Spectral) Test applies FFT to the sequence to detect repetitive patterns, while the Non-Overlapping Template Matching Test identifies occurrences of predefined non-periodic patterns. The Overlapping Template Matching Test looks for overlapping patterns, and the Universal Statistical Test examines the distribution of bits between matching patterns.

The Approximate Entropy Test compares the frequency of overlapping blocks of two consecutive lengths, and the Random Excursion Test evaluates the number of cycles returning to zero for different lengths. The Random Excursion Variant Test extends this by examining cycles at every state. The Linear Complexity Test measures the complexity of the sequence by counting the number of distinct linear feedback shift registers (LFSRs) needed to reproduce it.

Finally, the Serial Test comprises two parts: the Serial 1 Test, which examines the frequency of all possible length m patterns, and the Serial 2 Test, which detects overlapping patterns of length 2m. Together, these tests ensure that a sequence demonstrates the properties of true randomness, which is vital for cryptographic security.

The statistical hypothesis for the testing of encrypted images works on two possible outcomes; H_0=accept the uniformity of encrypted image pixels and H_1=reject the null hypothesis. The fixed significance level (α>0.01) defines the randomness of encrypted image pixels. Two error types, Type I and Type II, consider the encrypted image's randomness. If the pixels are random, the null hypothesis is rejected, concluding that it is non-random (Type I error). If the stream non-random accepts the null hypothesis, it determines that it is random (Type II error). Table 5 presents the hypothesis of accepting the uniformity of pixels in an encrypted image.

The NIST test applies to nine benchmark encrypted images, and the p-value of each test is present in Table 6. The result of p-value indicates that all the pixels of encrypted images are uniformly distributed. When the p-value is less than 0.001 under the NIST SP 800-22 standard, it suggests the absence of a random pattern in the sequence of bits. Conversely, a p-value of 0.001 or higher indicates that the tested bits demonstrate a uniform distribution and possess statistical reliability.

Table 5. Hypothesis testing scenario

Situation	**Scenario 1**	**Scenario 2**
Uniformity in encrypted image pixel (H_0)	H_0 is accepted No error	H_0 is rejected (Accept H_1) Type I error
Non-uniformity in encrypted image pixels (H_1)	Type II error	No error

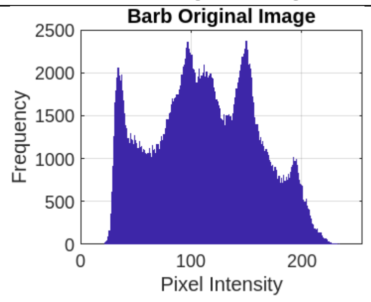
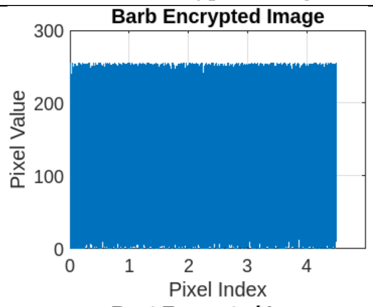
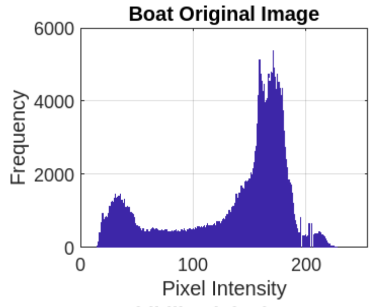
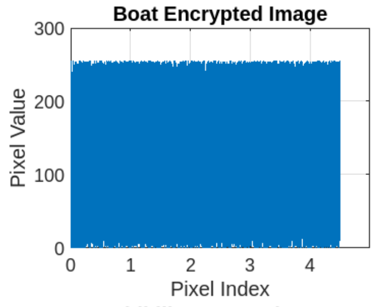
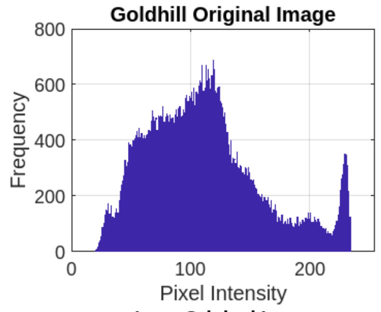
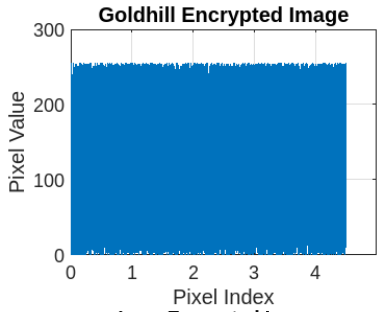
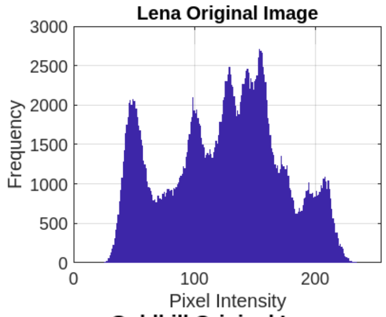
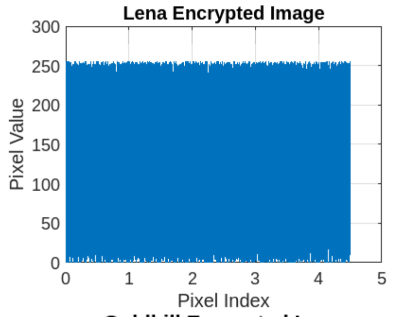
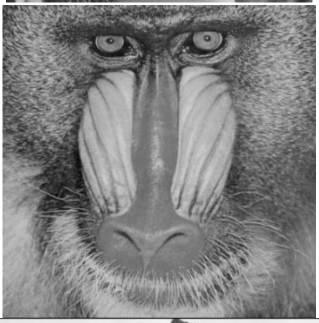
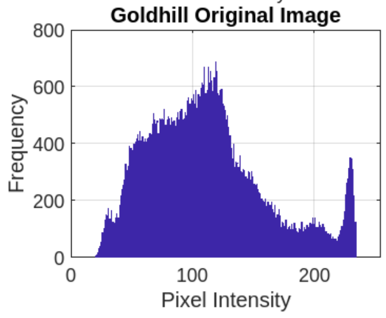
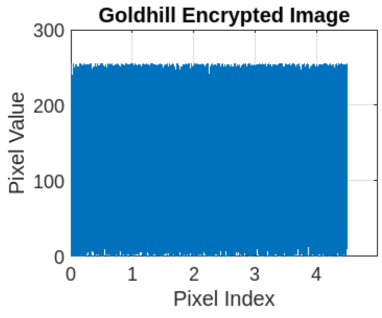
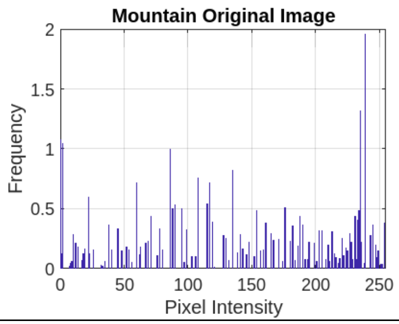
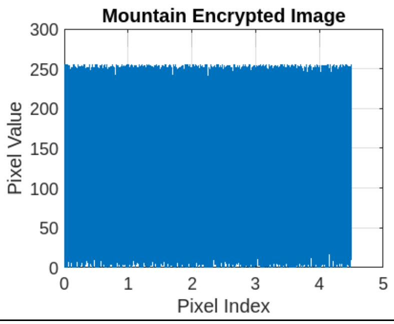
5.2.2 Histogram analysis

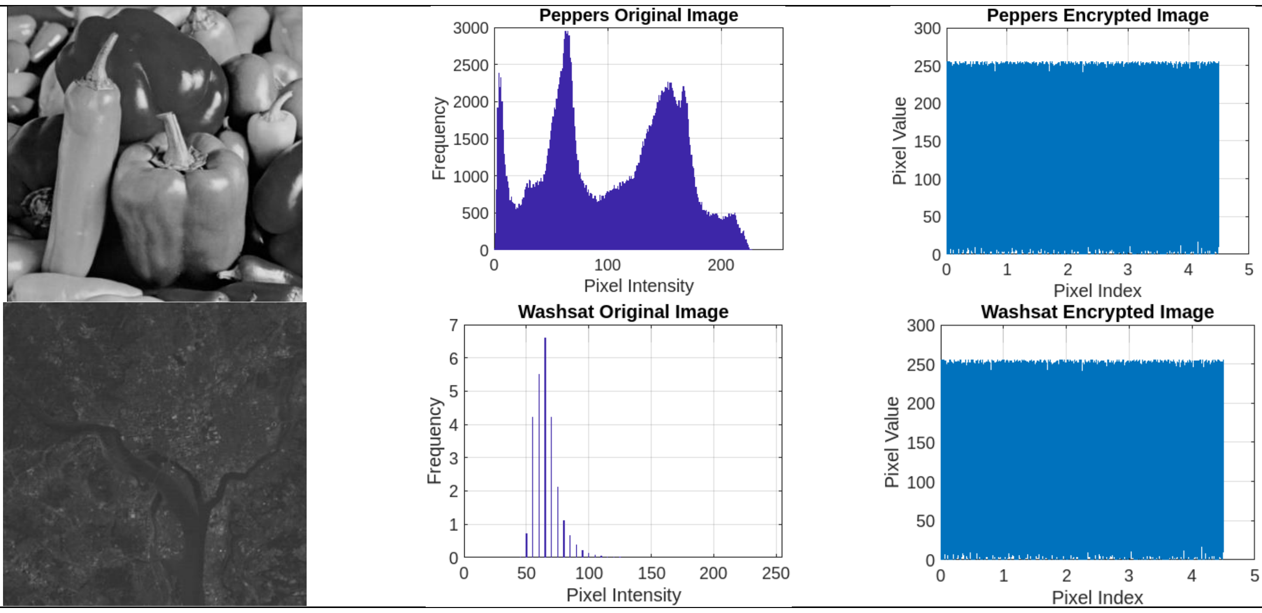
This section compares the encrypted and original image pixels using a histogram plot. Table 7 presents the original image pixels distributed non-uniformly throughout the region, while the encrypted image shows a uniform distribution of pixels in the histogram graph plot.

Table 6. NIST statistical test on encrypted image generated from Proposed XOR and FBSC

Type of Test	**Lena**	**Boat**	**Barb**	**Goldhill**	**Mandrill**	**Mountain**	**Washsat**	**Peppers**
Frequency Test	0.4120	0.9606	0.6140	0.1815	0.1327	0.4770	0.0257	0.1384
Frequency Test within a Block	0.4227	0.9000	0.7167	0.7484	0.6990	0.5227	0.2836	0.8312
Run Test	0.7315	0.2578	0.1093	0.6254	0.8976	0.3402	0.5621	0.9247
Longest Run of Ones in a Block	0.6318	0.8852	0.4275	0.7094	0.9136	0.0158	0.7462	0.3985
Binary Matrix Rank Test	0.2036	0.5903	0.8315	0.1089	0.4462	0.7084	0.1975	0.5243
Discrete Fourier Transform (Spectral) Test	0.8319	0.7642	0.4198	0.6310	0.2894	0.5021	0.6193	0.8801
Non-Overlapping Template Matching Test	0.7432	0.1839	0.9136	0.2874	0.5726	0.4261	0.9583	0.7310
Overlapping Template Matching Test	0.8196	0.1753	0.6402	0.5049	0.7284	0.8319	0.4627	0.6951
Maurer's Universal Statistical test	0.1742	0.6932	0.2783	0.5229	0.7401	0.6192	0.9240	0.4715
Linear Complexity Test	0.5903	0.8315	0.1089	0.4462	0.7084	0.1975	0.5243	0.9621
Serial Test 1	0.7683	0.4906	0.9157	0.2314	0.6728	0.9285	0.1396	0.8183
Serial Test 2	0.1794	0.7362	0.3049	0.9128	0.6523	0.4418	0.7195	0.5871
Approximate Entropy Test	0.5412	0.8629	0.1754	0.7083	0.4367	0.2984	0.8271	0.6315
Cumulative Sums (Forward)	0.4026	0.9531	0.6872	0.8194	0.3658	0.7841	0.5927	0.4103
Cumulative Sums (Reverse)	0.6472	0.2895	0.8102	0.5398	0.7321	0.1754	0.9832	0.6291

Table 7. Histogram analysis

Original Image	Pictorial Original Image	Pictorial Encrypted Image
		
		
		
		
		
		



5.2.3 Correlation coefficient (CF) analysis

The correlation coefficient analysis uses the mean and variance of pixels within the nearest pixel and is computed as Eq. (31).

$$Corr\,(X,Y) = \frac{\left|\frac{1}{N}\sum_{i=1}^{n}(X_i - E(X))(Y_i - E(Y))\right|}{\sqrt{\frac{1}{N}\sum_{i=1}^{n}(X_i - E(X))^2 \times (Y_i - E(Y))^2}} \tag{31}$$

Here, $E(x)$ and $E(y)$ are the mean of the pixel value, and it is computed as Eq. (32).

$$E(X) = \frac{1}{N}\sum_{i=1}^{n} X_i \;\&\; E(Y) = \frac{1}{N}\sum_{i=1}^{n} Y_i \tag{32}$$

Table 8 summarizes the comparison results of correlation coefficient analysis for benchmark grayscale images of size (256×256). The test includes 16,384 pairs of neighboring pixels to perform the CF analysis. The correlation coefficient values for the original images are closer to '1', indicating a higher degree of correlation between neighboring pixels. In contrast, the coefficient correlation values for the encrypted data are more relative to '0', meaning a lower degree of correlation between neighboring pixels. The observation suggests that the proposed XOR and Involution Function-Based Stream Cipher generates high entropy among neighboring pixels.

Table 8. Comparing correlation values of different methods

Methods	Image	Horizontal	Vertical	Diagonal
Hua and Zhou [31]	Lena	-0.0685	0.0857	0.0059
	Goldhill	-0.0351	0.0556	0.0330
Gao et al. [32]	Cameraman	0.0159	0.0093	0.0097
	Lena	0.0069	0.0047	0.0056
Ye and Huang [33]	Cameraman	0.0063	-0.0099	-0.0076
	Baboon	-0.0063	0.0070	0.0051
	Boat	0.0033	-0.0069	0.0025
Nematzadeh et al. [34]	Boat	0.0014	0.0012	0.0029
	Cameraman	0.0014	0.0014	0.0022
	Lena	0.0015	0.0019	0.0012
Proposed XOR and FBSC	Boat	-0.0025	-0.0016	-0.0005
	Lena	-0.0011	-0.0012	-0.0017
	Goldhill	0.0022	0.0045	0.0023
	Cameraman	0.0028	0.0025	0.0017

Here, ciphered image values are nearer to '0' for all the images. The proposed XOR and function-based stream cipher technique results indicate that the encryption process has successfully obscured the original image's patterns.

5.2.4 Differential attack analysis

Differential Attack Analysis is a method used to test the robustness of cryptographic algorithms, particularly in how small changes in the input (such as altering a single pixel in an image) can affect the output (the encrypted image).

NPCR (Number of Pixels Change Rate) is a measure used in Differential Attack Analysis to evaluate the extent of pixel change across an encrypted image when a single pixel in the original image is altered.

$$NPCR = \frac{\sum_{i=1}^{R}\sum_{j=1}^{C} E_d(i,j)}{R \times C} \tag{33}$$

where, 'R' and 'C' are the images' dimensions (rows and columns). $E_d(i, j)$ is a binary indicator function.

$$E_d(i,j) = \begin{cases} 0, \text{if } E_1(\text{i, j}) = E_2(\text{i, j}) \\ 1, \text{if } E_1(\text{i, j}) \neq E_2(\text{i, j}) \end{cases} \tag{34}$$

where, E_1 is the encrypted image of the original image, and E_2 is the encrypted image of the original image with one pixel altered. A high NPCR value indicates that the encryption algorithm causes significant changes in the encrypted image when a single pixel in the original image is modified, demonstrating strong sensitivity and resistance to differential attacks. The ideal NPCR value is around 99.609%.

UACI (Unified Average Changing Intensity) measures the average intensity difference between two encrypted images, providing insight into how uniformly the encryption algorithm spreads the changes.

$$UACI = \frac{\sum_{i=1}^{R}\sum_{j=1}^{C} |E_1(i,j) - E_2(i,j)|}{255 \times R \times C} \tag{35}$$

where, $|E_1(i,j)\text{-}E_2(i,j)|$ is the absolute difference in pixel values at coordinate (i, j) between the two encrypted images, and 255 is the maximum possible pixel value for an 8-bit image. A high UACI value indicates that the encryption algorithm effectively distributes the changes across the image, ensuring that the differences in the input are diffused throughout the encrypted image. The ideal UACI value is around 33.46%.

Table 9. Comparison of average NPCR and UACI values

S. No.	Method	NPCR	UACI
1	Proposed XOR and FBSC	99.6334	33.4954
2	Hua and Zhou [31]	99.6166	33.5033
3	Gao et al. [32]	99.6110	33.4430
4	Ye and Huang [33]	99.6405	33.5175
5	Mahalingam et al. [35]	99.6230	33.4935
6	Chen et al. [36]	99.64	33.50
7	Guan et al. [37]	99.62	33.44
8	Belazi et al. [38]	99.63	33.61
9	Haghighi et al. [39]	99.61	33.47
10	Stalin et al. [40]	99.61	33.46
11	Yang et al. [41]	99.65	33.5078
12	Zefreh [42]	99.64	33.54
13	Zhang and Hu [43]	99.61	33.50
14	Ravichandran et al. [44]	99.6060	33.5126
15	Patel et al. [45]	99.6067	33.47
16	Zheng and Liu [46]	99.6366	33.4586
17	Li et al. [47]	99.62	33.47
18	Zhang et al. [48]	99.6094	33.4635
19	Lone and Qureshi [49]	99.6056	33.4758
20	Teng et al. [50]	99.6060	33.4689
21	Ye et al. [51]	99.6132	33.4601
22	Zhang et al. [52]	99.6199	33.4773

In Table 9, the NPCR and UACI values of the proposed algorithm are compared to those of other algorithms. The results demonstrate that the proposed algorithm surpasses the ideal NPCR and UACI benchmarks, indicating high resistance to differential attacks.

5.3 Security analysis

The section is divided into two parts: (i) Computational Overhead, (ii) Storage Overhead, and (iii) Security Analysis.

5.3.1 Computation overhead

The computational overhead evaluates the performance of proposed Attributed-based hierarchical tree structure scheme (AB-HTS-S). The performance of the scheme is related to decryption of the encrypted data using shared secret points.
Here the AB-HTS-S is compared with existing schemes such as CP-WABE-CA, CC-ABE, and CP-ABE-SD. Our scheme uses an Attributed-based hierarchical tree structure with Lagrange interpolation to enable efficient multi-user collaborative decryption. To our knowledge, no existing scheme supports hierarchical tree structure access. The parameter for evaluation is as follows:

- δ: exponential operations overhead.
- τ: hash operations calculational effort.
- ρ: calculational overhead for pairing operations within group H.
- ν: number of users cluster.
- ν_a: attributes connected to access tree.
- ν_c: connected node count.
- ν_u: number of attributes a user required for decryption.
- $\nu_{c,r}$: number of non-leaf nodes required to regenerate root node.
- $\nu_{l,c}$: count of non-leaf nodes required to compute leaf node.
- |H| size of an element in group H.
- $|H_T|$ size of an element in the H_T group.

Table 10 present the comparison of computation overhead for each phase across different schemes establishes the proposed scheme's efficiency and performance advantage by requiring fewer computational resources, making it well-suited for cryptographic operations.

The next experiment is performed to observe the number of attributes used to construct the polynomial using a multiparty authorization key. Figure 4 represents the reconstruction time of $Skey_{Points}$ and performs the encryption and decryption using XOR and Functional-Based Stream Cipher (FBSC). The results indicate that as the number of attributes increases, the time required for encryption also increases. It implies a clear relationship between the complexity of the attribute set and the computational burden during the encryption process. Consequently, more attributes lead to longer encryption times, underscoring the influence of attribute count on encryption efficiency.

Figure 5 presents the decryption cost on both the $Organization_{Server}$ and Data User (DUs) sides. The cost of decryption increases linearly with an increase in the user attributes. The server-side costs encompass computational resources and time required to manage, process, and decrypt data, which increase as user attributes grow. The increases in server costs result from the heightened complexity and increased data handling associated with each additional attribute. Decryption costs also rise with the number of attributes on the DUs side. It encompasses the time and computational power needed for the user's device to decrypt the data. As the number of attributes expands, the decryption process becomes more resource-intensive, leading to longer decryption times and higher energy consumption on the user's device.

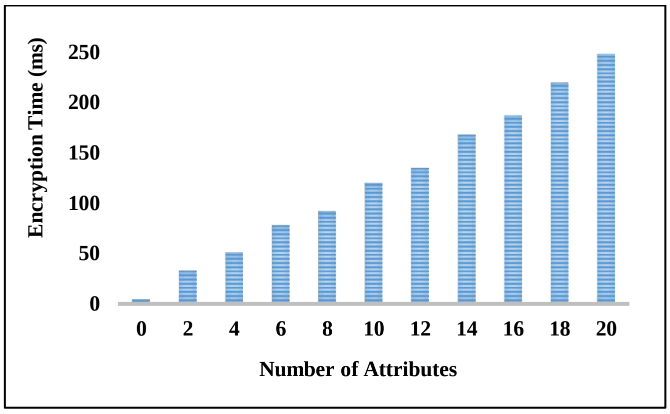


Figure 4. Encryption time

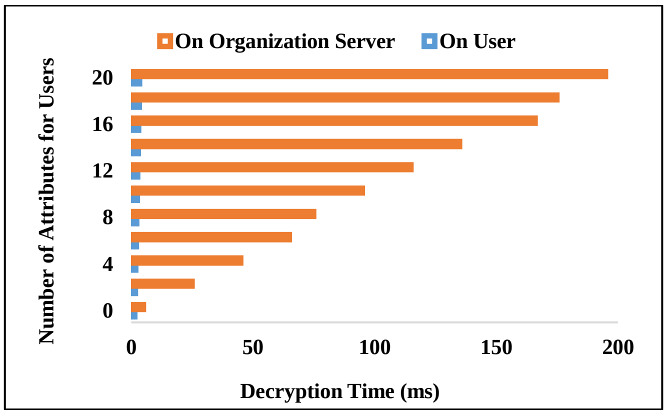


Figure 5. Decryption time

Table 10. Comparison of computational overhead

Phase	CC-ABE [1]	CP-ABE-SD [53]	CP-WABE-CA [54]	Proposed AB-HTS-S
Setup	$(\nu+5)\delta+\rho$	$2\delta+\rho$	$(\nu+1)\delta+\rho$	$\nu\delta+2\rho$
Encrypt	$(4\nu_a+2\nu_c+3)\delta+2\nu_u\tau$	$(4\nu_u+2\nu_c+1)\delta+2\nu_a\tau$	$(2\nu_a+3\nu_c+1)\delta+\nu_a\tau$	$(3\nu_a+2\nu_c+1)\delta+\nu_a\tau$
Decrypt	$(4\nu_u+2)\rho+(\nu_u+\nu_{l,c}+\nu_{c,r})\delta$	$(4\nu_u+1)\rho+(\nu_u+\nu_{l,c}+\nu_{c,r})\delta$	$(2\nu_u+1)\rho+(\nu_u+\nu_{l,c}+\nu_{c,r})\delta$	$(3\nu_u+1)\rho+(3\nu_u+2\nu_{l,c}+\nu_{c,r})\delta$
Collaborative decrypt	$(4\nu_u+\nu_c+2)\rho+(\nu_u+\nu_{l,c}+\nu_{c,r})\delta$	$(4\nu_u+\nu_c)\rho+(\nu_u+\nu_{l,c})\delta$	$(2\nu_u+\nu_c+1)\rho+(\nu_u+\nu_{l,c}+\nu_{c,r})\delta$	$(3\nu_u+2\nu_c+1)\rho+(3\nu_u+2\nu_{l,c}+\nu_{c,r})\delta$

5.3.2 Storage overhead

The experiment is performed to evaluate the impact of attribute size on the storage overhead of an Organization$_{\text{Server}}$. The owner and user points are required for key reconstruction in the proposed system. The size of each parameter in the scheme is represented by $|P_i|$, while 'm' and 'm_j' denote the total number of attributes for owners and users, respectively. Table 11 compares the storage overhead of different parameter sizes used for key generation.

m_{AA}=the number of elements in the management domain.

K_c=the number of elements rooted in the ciphertext.

Table 11. Comparison of storage overhead

Party	Proposed Model	Yang's DAC-MACS [18]
On User	$(m+1)\|P_i\|$	$(m_{AA}+3)\|P_i\|$
Organization$_{\text{Server}}$	$(K_c+1)\|P_i\|$	$(3K_c+3)\|P_i\|$

Our proposed model stores authorization Skey$_{\text{Points}}$ generated using a polynomial on the Organization$_{\text{Server}}$. At the same time, the DAC-MACS model [17] utilizes the secret authority component for key reconstruction. In contrast, our proposed scheme only requires the local Organization$_{\text{Server}}$ to store the authorization points as tokens for key reconstruction. After sharing the encrypted data and Skey$_{\text{Points}}$, the PK_{sk} and Encryption_Data$_{\text{i}}$ are deleted from the Organization$_{\text{Server}}$. The organization server manages all attributes, while the cloud solely stores an Access Control List (ACL) and Encryption_Data$_{\text{i}}$ .Our system holds fewer parameters compared to Yang's DAC-MACS. When the owner's attributes exceed the number of user identities, it increases storage overhead for data consumers and impacts 'm_i' with the same 'mj'. By merging the same user identities, we can reduce storage overhead. Thus, in our scheme, a user's storage overhead is not influenced by the number of user identities.

5.3.3 Security analysis

(1) Partial authorization key exposure attack

In our system, if an attacker possesses 'j' shares of the Skey$_{\text{Points}}$, it is equivalent to the adversary having no secret key share.

Proof: To reconstruct the PK_{sk} In our proposed system, we require a minimum of three shares (Organization$_{\text{Server}}$, D$_{\text{Owner}}$, and DUs points) to generate the polynomial. Let us assume that 'P_i' denote the probability of adversary guessing an available Skey$_{\text{Points}}$ with 'i' shares. The authentication is performed based on the Organization$_{\text{Server}}$ and D$_{\text{Owner}}$ share, which means that an adversary cannot access the encrypted file unless the D$_{\text{Owner}}$ doesn't provide the authorization Skey$_{\text{Points}}$ share to the system. It involves two requirements: (i) the set of attribute points must satisfy encryption, and (ii) the D$_{\text{Owner}}$ must approve the access of the data file containing the attributes and share the key.

(2) Collusion attack resistance

Our system has been designed with a robust security feature to resist collusion attacks.

Proof: In our proposed model, each D$_{\text{Owner}}$ is associated with attributes such as User_Id (U$_{\text{i}}$), User_Type (UT$_{\text{i}}$), and User_Credentials (UC$_{\text{i}}$). The UC$_{\text{i}}$ is integrated into polynomials alongside PK_{sk}, which is utilized in Function-Based Stream Cipher. Organization$_{\text{Server}}$ and D$_{\text{Owner}}$ are required to access the encrypted file, ensuring that only one recipient can access the file at any time. The attribute integration is given as Eq 36.

$$UC_i = f(PK_{sk}, U_i, UT_i) \tag{36}$$

Only the Organization$_{\text{Server}}$ and D$_{\text{Owner}}$ are authorized to access the encrypted file, ensuring it is given as Eq. 37.

$$Access_{Requirements} = (\text{Organization}_{\text{Server}} + \text{D}_{\text{Owner}}) \tag{37}$$

The attachment of the D$_{\text{Owner}}$ shares to generate PK_{sk} is prohibited, preventing unauthorized access as Eq 38.

$$PK_{sk} = f(U_i, UT_i, UC_i) \tag{38}$$

If multiple DUs attempt to perform the task independently, the decryption of Encryption_Data$_{\text{i}}$ cannot be jointly carried out simultaneously, as represented in Eq. (39).

$$\text{Decryption\_Task}_{\text{i}} = DUs_1 + DUs_2 + \cdots + DUs_n \tag{39}$$

(3) Security resilience against compromised owner points

Our system maintains security even when some owner points are compromised.

Proof: In previous studies on related schemes [17, 23], the scheme constraints stipulate that Organization$_{\text{Server}}$, D$_{\text{Owner}}$, and DUs points can collectively generate the symmetric secret key PK_{sk}. This mechanism assumes that if any points belonging to DUs are compromised, the adversary may utilize those points for key generation. The proposed system employs a two-way authentication process, where Organization$_{\text{Server}}$ authenticates the adversary's credentials. Once authentication is completed, the Organization$_{\text{Server}}$ requires the D$_{\text{Owner}}$ share to generate the symmetric secret key PK_{sk}. Subsequently, D$_{\text{Owner}}$ validates the recipient using the access control list. With these properties in place, the construction of PK_{sk} using the proposed scheme remains resilient, even when some D$_{\text{Owner}}$ points are compromised. The constraint on key generation is given as Eq. (40).

$$PK_{sk} = f(\text{Organization}_{\text{Server}} + \text{D}_{\text{Owner}} + DUs) \tag{40}$$

The OrganizationServer authenticates the adversary's credentials using Eq. (41).

$$Auth_{OS} = g(Adversary\_Credentials) \tag{41}$$

Once authenticated, the Organization$_{\text{Server}}$ requires the

$\mathrm{D_{Owner}}$ share to generate the PK_{sk}:

$$Auth_{DO} = h(\mathrm{D_{Owner}}\_Share) \tag{42}$$

The $\mathrm{D_{Owner}}$ validates the recipient using the access control list.

$$Validation = i(Recipient, \mathrm{Access}\_Control\_List) \tag{43}$$

Even if some $\mathrm{D_{Owner}}$ points are compromised, the construction of PK_{sk} using the proposed scheme remains robust:

$$Resilience = j(Compromised\_\mathrm{D_{Owner}}\_Points) \tag{44}$$

The proposed scheme ensures the integrity of the PK_{sk} generation process.

(4) Chosen plaintext attack (CPA)

The chosen plaintext attack (CPA) is mitigated in our proposed scheme through a robust security model involving dual-layer authentication and an Attribute-based hierarchical tree structure scheme. When an adversary attempts to perform a CPA by selecting plaintexts and obtaining their corresponding ciphertexts, they cannot decrypt the ciphertexts without proper authentication. The encryption process uses the PK_{sk}, which is derived through a two-way authentication mechanism: the $\mathrm{Organization_{Server}}$ first authenticates credentials and then requires the $\mathrm{D_{Owner}}$ share for PK_{sk}. Even if some $\mathrm{D_{Owner}}$ points are compromised, the adversary cannot generate the PK_{sk} due to the need for both $\mathrm{Organization_{Server}}$ and $\mathrm{D_{Owner}}$ validation. It ensures that any decryption attempt by the adversary fails, as they cannot bypass the authentication process or obtain PK_{sk}. Thus, the attributed-based hierarchical tree structure with Lagrange interpolation in our scheme provides robust security against chosen plaintext attacks, maintaining the integrity and confidentiality of the encrypted data.

6. CONCLUSION AND FUTURE WORK

This paper presents a robust encryption and secure data storage system leveraging XOR and Functional-Based Stream Cipher techniques. Our proposed model employs the organization server to securely encrypt data, which is then stored on the cloud server. Using Shamir's Secret Sharing Scheme, $\mathrm{Skey_{Points}}$ are distributed among data users. The Secret Key Points are stored using the Attributed-based hierarchical tree structure scheme (AB-HTS-S), ensuring efficient initial computation during key reconstruction. The Attribute-Based Access Control Scheme (ABCS) with Lagrange interpolation is employed for symmetric secret key reconstruction, which is shared with privileged users, ensuring secure and controlled access. By applying these techniques, our scheme effectively mitigates chosen plaintext attacks. Even if some authorization points are compromised, adversaries cannot decrypt the ciphertext without passing through the dual-layer authentication process. The $\mathrm{Organization_{Server}}$ and $\mathrm{D_{Owner}}$ validation mechanisms prevent unauthorized access and maintain data integrity and confidentiality. This mechanism addresses critical vulnerabilities in existing models, providing a reliable and efficient method for protecting sensitive information against advanced cryptographic attacks. Future work will focus on optimizing computational efficiency, particularly for large-scale implementations with numerous attributes and users. We aim to explore advanced cryptographic techniques to reduce encryption and decryption overheads further. Additionally, we plan to integrate our scheme with emerging technologies like blockchain and edge computing to enhance decentralized security and data integrity. Supporting dynamic attribute updates and revocation will be another key development area, ensuring continuous adaptability and robustness. Comprehensive real-world testing and validation will be conducted to assess performance and security in diverse scenarios.

ACKNOWLEDGMENT

The authors would like to extend their gratitude to King Saud University (Riyadh, Saudi Arabia) for funding this research through Researchers supporting (Project No.: RSP2025R260).

REFERENCES

[1] Xue, Y., Xue, K., Gai, N., Hong, J., Wei, D.S., Hong, P. (2019). An attribute-based controlled collaborative access control scheme for public cloud storage. IEEE Transactions on Information Forensics and Security, 14(11): 2927-2942. https://doi.org/10.1109/tifs.2019.2911166

[2] Mansouri, Y., Toosi, A.N., Buyya, R. (2017). Data storage management in cloud environments: Taxonomy, survey, and future directions. ACM Computing Surveys (CSUR), 50(6): 91. https://doi.org/10.1145/3136623

[3] Talbi, J., Haqiq, A. (2016). A broker framework for selecting secure cloud service provider using security risk approach. International Journal of Cloud Computing, 5(3): 175-186. https://doi.org/10.1504/ijcc.2016.10000904

[4] Sahai, A., Waters, B. (2005). Fuzzy identity-based encryption. In Advances in Cryptology-EUROCRYPT 2005: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, pp. 457-473. https://doi.org/10.1007/11426639_27

[5] Zhang, G., Liu, Q. (2011). A novel image encryption method based on total shuffling scheme. Optics communications, 284(12): 2775-2780. https://doi.org/10.1016/j.optcom.2011.02.039

[6] Wang, S., Liang, K., Liu, J.K., Chen, J., Yu, J., Xie, W. (2016). Attribute-based data sharing scheme revisited in cloud computing. IEEE Transactions on Information Forensics and Security, 11(8): 1661-1673. https://doi.org/10.1109/tifs.2016.2549004

[7] Lee, H.W. (2021). Design of multi-functional access control system. IEEE Access, 9: 85255-85264. https://doi.org/10.1109/access.2021.3087917

[8] Li, Q., Ma, J., Li, R., Liu, X., Xiong, J., Chen, D. (2016). Secure, efficient and revocable multi-authority access control system in cloud storage. Computers & Security, 59: 45-59. https://doi.org/10.1016/j.cose.2016.02.002

[9] Oberko, P.S.K., Obeng, V.H.K.S., Xiong, H. (2022). A survey on multi-authority and decentralized attribute-based encryption. Journal of Ambient Intelligence and

Humanized Computing, 13: 515-533. https://doi.org/10.1007/s12652-021-02915-5
[10] Cayirci, E., Garaga, A., Santana de Oliveira, A., Roudier, Y. (2016). A risk assessment model for selecting cloud service providers. Journal of Cloud Computing, 5(1): 14. https://doi.org/10.1186/s13677-016-0064-x
[11] Xie, M., Ruan, Y., Hong, H., Shao, J. (2021). A CP-ABE scheme based on multi-authority in hybrid clouds for mobile devices. Future Generation Computer Systems, 121: 114-122. https://doi.org/10.1016/j.future.2021.03.021
[12] Gu, J., Shen, J., Wang, B. (2021). A robust and secure multi-authority access control system for cloud storage. Peer-to-Peer Networking and Applications, 14: 1488-1499. https://doi.org/10.1007/s12083-020-01055-5
[13] Huang, K. (2021). Revocable large universe decentralized multi-authority attribute-based encryption without key abuse for cloud-aided IoT. IEEE Access, 9: 151713-151728. https://doi.org/10.1109/access.2021.3126780
[14] Liu, Z., Jiang, Z.L., Wang, X., Yiu, S.M. (2018). Practical attribute-based encryption: Outsourcing decryption, attribute revocation and policy updating. Journal of Network and Computer Applications, 108: 112-123. https://doi.org/10.1016/j.jnca.2018.01.016
[15] Ling, J., Chen, J., Chen, J., Gan, W. (2021). Multiauthority attribute-based encryption with traceable and dynamic policy updating. Security and Communication Networks, 2021(1): 6661450. https://doi.org/10.1155/2021/6661450
[16] Hwang, Y.W., Lee, I.Y. (2020). CP-ABE access control that block access of withdrawn users in dynamic cloud. KSII Transactions on Internet and Information Systems (TIIS), 14(10): 4136-4156. https://doi.org/10.3837/tiis.2020.10.012
[17] Lai, J., Deng, R.H., Guan, C., Weng, J. (2013). Attribute-based encryption with verifiable outsourced decryption. IEEE Transactions on Information Forensics and Security, 8(8): 1343-1354. https://doi.org/10.1109/tifs.2013.2271848
[18] Yang, K., Jia, X., Ren, K., Zhang, B., Xie, R. (2013). DAC-MACS: Effective data access control for multiauthority cloud storage systems. IEEE Transactions on Information Forensics and Security, 8(11): 1790-1801. https://doi.org/10.1109/tifs.2013.2279531
[19] Li, J., Sha, F., Zhang, Y., Huang, X., Shen, J. (2017). Verifiable outsourced decryption of attribute-based encryption with constant ciphertext length. Security and Communication Networks, 2017: 1-11. https://doi.org/10.1155/2017/3596205
[20] Setapa, S., Suhilah, T.P. (2014). An access control list for role-based system: An observation and recommendation. International Journal of Information and Education Technology, 4(6): 468. https://doi.org/10.7763/ijiet.2014.v4.452
[21] Li, W.M., Li, X.L., Wen, Q.Y., Zhang, S., Zhang, H. (2017). Flexible CP-ABE based access control on encrypted data for mobile users in hybrid cloud system. Journal of Computer Science and Technology, 32: 974-990. https://doi.org/10.1007/s11390-017-1776-1
[22] Yin, H., Zhang, J., Xiong, Y., Ou, L., Li, F., Liao, S., Li, K. (2019). CP-ABSE: A ciphertext-policy attribute-based searchable encryption scheme. IEEE Access, 7: 5682-5694. https://doi.org/10.1109/access.2018.2889754
[23] Shen, J., Deng, X., Xu, Z. (2019). Multi-security-level cloud storage system based on improved proxy re-encryption. EURASIP Journal on Wireless Communications and Networking, 2019(1): 277. https://doi.org/10.1186/s13638-019-1614-y
[24] Zhong, H., Zhu, W., Xu, Y., Cui, J. (2018). Multi-authority attribute-based encryption access control scheme with policy hidden for cloud storage. Soft Computing, 22: 243-251. https://doi.org/10.1007/s00500-016-2330-8
[25] Zhang, L., Hu, G., Mu, Y., Rezaeibagha, F. (2019). Hidden ciphertext policy attribute-based encryption with fast decryption for personal health record system. IEEE Access, 7: 33202-33213. https://doi.org/10.1109/access.2019.2902040
[26] Rao, K.D., Kumar, K.P., Murali Krishna, P.V. (2011). A new and secure cryptosystem for image encryption and decryption. IETE Journal of research, 57(2): 165-171. https://doi.org/10.4103/0377-2063.81745
[27] Sinha, K., Paul, P., Amritanjali, A. (2022). An improved pseudorandom sequence generator and its application to image encryption. KSII Transactions on Internet and Information Systems (TIIS), 16(4): 1307-1329. https://doi.org/10.3837/tiis.2022.04.012
[28] Sun, F., Lv, Z. (2022). A secure image encryption based on spatial surface chaotic system and AES algorithm. Multimedia Tools and Applications, 1-21. https://doi.org/10.1007/s11042-021-11690-6
[29] Arab, A., Rostami, M.J., Ghavami, B. (2019). An image encryption method based on chaos system and AES algorithm. The Journal of Supercomputing, 75: 6663-6682. https://doi.org/10.1007/s11227-019-02878-7
[30] Rukhin, A., Soto, J., Nechvatal, J., Smid, M., Barker, E., Leigh, S., Levenson, M., Vangel, M., Banks, D., Heckert, A., Dray, J., Vo, S. (2001). A statistical test suite for random and pseudorandom number generators for cryptographic applications. National Institute of Standards and Technology. https://doi.org/10.6028/nist.sp.800-22
[31] Hua, Z., Zhou, Y. (2017). One-dimensional nonlinear model for producing chaos. IEEE Transactions on Circuits and Systems I: Regular Papers, 65(1): 235-246. https://doi.org/10.1109/tcsi.2017.2717943
[32] Gao, S., Wu, R., Wang, X., Wang, J., Li, Q., Wang, C., Tang, X. (2023). A 3D model encryption scheme based on a cascaded chaotic system. Signal Processing, 202: 108745. https://doi.org/10.1016/j.sigpro.2022.108745
[33] Ye, G., Huang, X. (2016). A secure image encryption algorithm based on chaotic maps and SHA-3. Security and Communication Networks, 9(13): 2015-2023. https://doi.org/10.1002/sec.1458
[34] Nematzadeh, H., Enayatifar, R., Yadollahi, M., Lee, M., Jeong, G. (2020). Binary search tree image encryption with DNA. Optik, 202: 163505. https://doi.org/10.1016/j.ijleo.2019.163505
[35] Mahalingam, H., Veeramalai, T., Menon, A.R., Amirtharajan, R. (2023). Dual-domain image encryption in unsecure medium-a secure communication perspective. Mathematics, 11(2): 457. https://doi.org/10.3390/math11020457
[36] Chen, Y., Tang, C., Ye, R. (2020). Cryptanalysis and improvement of medical image encryption using high-speed scrambling and pixel adaptive diffusion. Signal

Processing, 167: 107286. https://doi.org/10.1016/j.sigpro.2019.107286

[37] Guan, M., Yang, X., Hu, W. (2019). Chaotic image encryption algorithm using frequency-domain DNA encoding. IET Image Processing, 13(9): 1535-1539. https://doi.org/10.1049/iet-ipr.2019.0051

[38] Belazi, A., Talha, M., Kharbech, S., Xiang, W. (2019). Novel medical image encryption scheme based on chaos and DNA encoding. IEEE Access, 7: 36667-36681. https://doi.org/10.1109/access.2019.2906292

[39] Haghighi, B.B., Taherinia, A.H., Mohajerzadeh, A.H. (2019). TRLG: Fragile blind quad watermarking for image tamper detection and recovery by providing compact digests with optimized quality using LWT and GA. Information Sciences, 486: 204-230. https://doi.org/10.1016/j.ins.2019.02.055

[40] Stalin, S., Maheshwary, P., Shukla, P. K., Maheshwari, M., Gour, B., Khare, A. (2019). Fast and secure medical image encryption based on non linear 4D logistic map and DNA sequences (NL4DLM_DNA). Journal of Medical Systems, 43: 1-17. https://doi.org/10.1007/s10916-019-1389-z

[41] Yang, F., Mou, J., Ma, C., Cao, Y. (2020). Dynamic analysis of an improper fractional-order laser chaotic system and its image encryption application. Optics and Lasers in Engineering, 129: 106031. https://doi.org/10.1016/j.optlaseng.2020.106031

[42] Zefreh, E.Z. (2020). An image encryption scheme based on a hybrid model of DNA computing, chaotic systems and hash functions. Multimedia Tools and Applications, 79(33-34): 24993-25022. https://doi.org/10.1007/s11042-020-09111-1

[43] Zhang, X., Hu, Y. (2021). Multiple-image encryption algorithm based on the 3D scrambling model and dynamic DNA coding. Optics & Laser Technology, 141: 107073. https://doi.org/10.1016/j.optlastec.2021.107073

[44] Ravichandran, D., Banu S.A., Murthy, B.K., Balasubramanian, V., Fathima, S., Amirtharajan, R. (2021). An efficient medical image encryption using hybrid DNA computing and chaos in transform domain. Medical & Biological Engineering & Computing, 59: 589-605. https://doi.org/10.1007/s11517-021-02328-8

[45] Patel, S., Thanikaiselvan, V., Pelusi, D., Nagaraj, B., Arunkumar, R., Amirtharajan, R. (2021). Colour image encryption based on customized neural network and DNA encoding. Neural Computing and Applications, 33(21): 14533-14550. https://doi.org/10.1007/s00521-021-06096-2

[46] Zheng, J., Liu, L. (2020). Novel image encryption by combining dynamic DNA sequence encryption and the improved 2D logistic sine map. IET Image Processing, 14(11): 2310-2320. https://doi.org/10.1049/iet-ipr.2019.1340

[47] Li, J., Chen, L., Cai, W., Xiao, J., Zhu, J., Hu, Y., Wen, K. (2022). Holographic encryption algorithm based on bit-plane decomposition and hyperchaotic Lorenz system. Optics & Laser Technology, 152: 108127. https://doi.org/10.1016/j.optlastec.2022.108127

[48] Zhang, F., Zhang, X., Cao, M., Ma, F., Li, Z. (2021). Characteristic analysis of 2D lag-complex logistic map and its application in image encryption. IEEE MultiMedia, 28(4): 96-106. https://doi.org/10.1109/mmul.2021.3080579

[49] Lone, M.A., Qureshi, S. (2022). RGB image encryption based on symmetric keys using Arnold transform, 3D chaotic map and affine hill cipher. Optik, 260: 168880. https://doi.org/10.1016/j.ijleo.2022.168880

[50] Teng, L., Wang, X., Xian, Y. (2022). Image encryption algorithm based on a 2D-CLSS hyperchaotic map using simultaneous permutation and diffusion. Information Sciences, 605: 71-85. https://doi.org/10.1016/j.ins.2022.05.032

[51] Ye, G., Wu, H., Liu, M., Shi, Y. (2022). Image encryption scheme based on blind signature and an improved Lorenz system. Expert Systems with Applications, 205: 117709. https://doi.org/10.1016/j.eswa.2022.117709

[52] Zhang, Y., Xie, H., Sun, J., Zhang, H. (2022). An efficient multi-level encryption scheme for stereoscopic medical images based on coupled chaotic system and Otsu threshold segmentation. Computers in Biology and Medicine, 146: 105542. https://doi.org/10.1016/j.compbiomed.2022.105542

[53] Chen, N., Li, J., Zhang, Y., Guo, Y. (2020). Efficient CP-ABE scheme with shared decryption in cloud storage. IEEE Transactions on Computers, 71(1): 175-184. https://doi.org/10.1109/tc.2020.3043950

[54] Li, X., Wang, H., Ma, S. (2025). An efficient ciphertext-policy weighted attribute-based encryption with collaborative access for cloud storage. Computer Standards & Interfaces, 91: 103872. https://doi.org/10.1016/j.csi.2024.103872