International Journal of Environmental Sciences
ISSN: 2229-7359
Vol. 11 No. 17s, 2025
https://theaspd.com/index.php

A Novel Access Control Based Privacy Preserving Model to Evaluate Multiuser Authentication Access Control in Secure Cloud Environment

Md. Sameera[1] Dr. K. Usha Rani[2]
[1] Research Scholar, Dept. of Computer Science, Sri Padmavati Mahila Visvavidyalayam (SPMVV) (Women's University), Tirupati
[2] Professor, Dept. of Computer Science, Sri Padmavati Mahila Visvavidyalayam (SPMVV) (Women's University), Tirupati

Abstract:
Cloud computing has become a new standard in computer architecture, allowing multiple users to remotely access scalable services and store data in the cloud. Small and Medium-sized Enterprises (SMEs) benefit from cloud-based solutions for project management, collaboration, and other tasks, offering significant savings and increased productivity, especially with budget constraints. However, concerns over data security and privacy arise when SMEs entrust sensitive information to Cloud Service Providers (CSPs) that may not be within the same trusted domains. A promising solution to address these concerns involves the use of cryptographic techniques, particularly by restricting access to decryption keys solely to authorized users. This effectively safeguards sensitive multi-user data from CSPs. As cloud computing expands, secure and efficient multi-user data sharing has become essential, given the vast amounts of data stored in the cloud. Ensuring data privacy in multi-party storage is critical for protecting shared data, and methods like concealing the server's address sequence during frequent data access can make it harder to trace data access. To meet the growing demand for scalable, adaptable, and reliable access control in distributed work environments, a novel approach called the Novel Access Control-based Privacy Preserving Approach (NACPPA) has been developed. This approach integrates Attribute-Based Encryption (ABE) and the Broadcast Group Multi-Key Policy (BGKM)**,** fulfilling the need for efficient Group Key Management (GKM) while maintaining the integrity of existing user secret shares without requiring modifications. Security evaluations comparing the proposed method to alternative solutions demonstrate its effectiveness in preserving privacy and ensuring secure access control.

Keywords: Privacy Preservation, Data Security, Cloud Data Sharing, Attribute Based Encryption, Broadcast Group Multi Key Management, Multi User Data Sharing, Shamir Secret Sharing Scheme.

1. INTRODUCTION

Cloud services are tasked with storing and processing vast amounts of sensitive data. This centralized handling of critical resources by CSPs exposes businesses to potential data breaches, loss, and unauthorized access. Therefore, data security and privacy must be top priorities when designing cloud-based solutions. The dynamic nature of multi-user environments in the cloud makes it essential to implement effective access control systems that can safeguard sensitive data while accommodating various user roles and permissions [1].

In particular, achieving efficient and secure access control is crucial in distributed computing systems. For instance, in medical services and other organizational settings, it may be more efficient to grant Shared Access Privileges (SAP) to all group members, rather than creating redundant copies for each individual. SAP ensures that group members, with or without explicit permission, can access the information they need. It also optimizes computational resources by confirming only the essential information and restricting unnecessary data access [2].

While Attribute-Based Encryption (ABE) has become a widely used method for secure authentication, it relies on a symmetric key-based cryptographic approach, which is not ideal for large-scale, multi-user cloud environments. ABE is efficient for single-key communication but lacks support for multi-

key generations, making it less suitable for distributed systems. Approaches like those proposed by Zhao et al. [3] and Yang et al. [4] introduce decentralized key management and authentication schemes. However, these solutions do not fully support user anonymity or authenticated access, limiting their application in cloud environments where both privacy and functionality are crucial.

Furthermore, decentralized systems proposed by Ruj et al. [5] fail to provide both read and write access control to files, nor do they restrict access based on file ownership. To address these limitations, there is a clear need to enhance distributed computing systems by incorporating advanced privacy features and refining authentication processes to better support data sharing and access control policies.

In response to these challenges, we propose a novel Broadcast Group Key Management (BGKM) scheme, which is the first to be provably secure and adaptable to dynamic, multi-user environments. Our approach minimizes the need for private communications between team members and multi-key servers while ensuring that secure interactions are maintained. By employing efficient hashing functions and vector-based computations, our method provides scalable, secure multi-key management in distributed systems [6].

Group Multi-Key Management (GKM) plays a critical role in secure group communications (SGC), which include applications such as video conferencing, secure file sharing, online publishing, and secure data broadcasting. In GKM, data is encrypted using a symmetric encryption algorithm and shared among group members via a multi-key. If the group dynamics change—such as when a new member joins or an existing member leaves—a new group multi-key is generated and securely redistributed to ensure that no one can access past or future communications without proper authorization. This process, known as re-multi-keying, ensures both forward and backward secrecy in group communications [7].

To enhance access control in distributed cloud environments, we introduce a Novel Access Control-based Privacy Preserving Approach (NACPPA)**.** This method is resilient to relay attacks and allows for the modification of access permissions, enabling the revocation of multi-user access when needed. The proposed solution is designed to facilitate scalable, adaptable, and trustworthy access control for data sharing in distributed settings.

The remainder section of this paper is structured as follows: Section 2 reviews relevant literature on cloud computing, access control, and cryptographic techniques. In Section 3, we outline the essential requirements and assumptions for our proposed approach. Section 4 details the design and implementation of NACPPA. Section 5 presents the empirical evaluation of our approach, assessing its efficiency and effectiveness. Finally, Section 6 concludes the paper, summarizing our contributions and suggesting avenues for future research.

2. LITERATURE REVIEW

Cloud computing and data sharing features present significant opportunities for addressing complex problems but also introduce new security risks. Securing sensitive data, particularly ensuring that ciphertext remains private, is a major challenge. Furthermore, the issue of many-to-many group data sharing complicates the management and protection of shared information.

Ateniese et al. [8] introduced a fast proxy re-encryption strategy in 2006, designed for distributed file systems. This technique, based on bilinear mapping, offered a way to secure data in the cloud but faced challenges in preventing malicious users from bypassing security and in defending against combinatorial attacks.

In 2010, Yu et al. [9] presented the attribute-based encryption (ABE) protocol, which combines proxy re-encryption with delayed re-encryption. ABE is used for fine-grained access control in cloud computing, maintaining data privacy while offering scalability. However, the one-to-many sharing mechanism employed in ABE lacks support for fine-grained encrypted data exchanges, which is crucial in more complex access control scenarios.

To address these concerns, Liang et al. [10] proposed a privacy-preserving ciphertext multi-sharing mechanism. By combining proxy re-encryption with anonymous data exchange, this approach improves user anonymity and data secrecy in cloud environments. In 2017, Zhang et al. introduced a secure multi-hop re-encryption strategy [11], which further enhanced the protection of sensitive information by applying continual concealing during the re-encryption process. This scheme is robust enough to withstand attacks from malicious or untrusted servers. However, like previous methods, it supports only one-to-one data sharing.

An additional challenge is the protection of user behavior from servers that may leak sensitive data access patterns. Oblivious Random Access Machine (ORAM) technology has been proposed as a solution to this problem, enabling secure and anonymous data retrieval while concealing access patterns [12]. Several ORAM schemes have been developed with the goal of minimizing the communication and storage overhead while ensuring robust security [13].

Hoang et al. [14] introduced a new ORAM technique based on a hierarchical structure. This approach improved data retrieval time to O(log^3n) and reduced server storage requirements to O(n log n), while addressing the issue of access pattern disclosure. Despite these improvements, the protocol's bandwidth and access speed were compromised.

In 2010, Yang et al. [15] proposed a new hierarchical ORAM scheme that combined the cuckoo hash method with random Hill sorting to balance bandwidth and security. While this scheme ensured secure data storage, its computational and storage costs were significant.

Following this, A. K. Das et al. [16] introduced a conventional Tree ORAM technique that utilized binary tree topology for efficient data access. However, the obfuscation process of this method led to a poor hidden access pattern effect, as it did not adequately address the limitations of the Tree ORAM design.

To enhance the security of Tree ORAM, He et al. [17] proposed a Merkel Hash tree-based Path ORAM approach, which included a rigorous security analysis based on hash values to verify data integrity. In 2015, Gentry et al. [18] introduced a new Tree ORAM design that utilized binary search methods to reduce bandwidth and storage costs, making the system more efficient.

Fisch et al. [19] introduced a two-service ORAM system in 2017, which integrated ORAM with privacy path retrieval technology. By leveraging privacy information from both servers, the client could retrieve the path to access the target data block. This method, unlike traditional systems, only required one communication round per access, thus improving efficiency over conventional ORAM implementations.

3. FUNDAMENTAL BACKGROUND

This section outlines the essential concepts and assumptions underlying the proposed system. It describes the roles of the data owner, the cloud server, and the data sharer in the context of secure data storage and sharing in cloud computing.

- **Cloud Server:**
  The cloud server is responsible for storing data and offering various services. It implements efficient access control policies for managing multiple users' data based on their credentials. Acting as a semi-trusted device, the cloud server ensures that encrypted data is securely stored and grants access only to authorized users. While the server can store plain-text data, it only provides access to encrypted data based on the users' credentials. Additionally, the server is responsible for managing user authorization privileges and handling revoked access to data files without notifying the data sharers.
- **Data Owner:**
  The data owner is a multi-user registered on the cloud platform. The owner can store confidential data and exchange files with other registered users based on their allowed privileges. The data owner maintains a registry of authorized credentials for all users and can issue new credentials to users who are authorized to access specific data. Furthermore, the data owner has the authority to revoke access to any user's data at any time, without notifying the data sharer about the revocation.
- **Data Sharer:**
  The data sharer can only access encrypted data. They are granted authorization to access specific files based on their credentials. The sharer interacts with encrypted data, which includes multi-key authorization, ensuring that the data they access is in line with the appropriate access control policies. The data sharer can explore only the encrypted files that match their credentials, and the cloud server verifies this authorization process. Additionally, the sharer identifies sensitive data using encryption masks and boundary code values to determine which files contain authorized credentials.

Table 1 provides a set of notations used throughout the implementation. These notations describe the roles, encryption keys and various functions essential for the secure management and sharing of data in the cloud system.

Table 1: Basic Notations

Notation	Description
Owner/User of v-th Multiuser (V_μ)	Denotes the owner or participant of the v-th multiuser group.
Attribute Relations(X_j , X , L_j)	Corresponds to the j-th attribute or its relationship within the system.
Multi-Key Distribution Center (KDC)	A centralized entity responsible for managing and distributing multi-keys.
Claimed Multiuser Attributes $(l_j = \mid L_j \mid, I[j,u], I_u)$	The total number of attributes claimed by a multiuser for encryption or decryption.
Public Multi-Key / Secret Multi-Key $(PK[j]/SK[j], sk_{i,v})$	Publicly shared or privately held multi-keys used for access control.
H,H,MSG	Hash functions with message

Based on the above parameters, the following concepts are utilized in the proposed implementation: basic formats of access policies, Access tree structure, and Attribute-Based Encryption (ABE).

3.1. Basic Format of Access Policies in Cryptography

Basic access control policies are typically described in the following formats:

- Attributes related to Boolean functions
- Secret sharing linear schema
- Span programs related to monotone

Boolean functions can be represented using an access tree structure. For example, an expression like: $((a_1 \wedge a_2 \wedge a_3) \vee (a_4 \wedge a_5) \wedge (a_5 \wedge a_6))$, $a_1, a_2, \ldots, a_6$ are attributes.

Let us consider $B:\{1,0\}^m \rightarrow \{1,0\}$ be the function relates to Boolean monotone, for every span function i.e. $(a_1, a_2, \ldots\ldots, a_n) \in \{1,0\}^m$, the following function should satisfied all the labelled functions

$b((a_1, a_2, \ldots, a_n)) = 1 \Leftrightarrow \exists v \in \mathbb{Z}^{1xl} : vN = [0, 1, 1, \ldots, 1] \wedge \forall i : (a_x(i) = 0 \Rightarrow v_i = 0)$ **(1)**

Here $b((a_1, a_2, \ldots\ldots, a_n) = 1$ be span function which is indexed with span vector $\{i \mid a_{x(i)} = 1\}$ & span program constructed in Boolean retrieval functions.

3.2. Shamir Secret Sharing Process

It's the bare minimum for protecting sensitive information in a distributed cloud storage system. Algorithm 1 demonstrates the fundamentals of Shamir's secret-sharing procedure [20].

Algorithm 1 Shamir's Secret Sharing Procedure: Message Preprocessing

Require: Information in the form of files with user details

Ensure: Encrypted data blocks with initial buffer states

1: Append padding bits to the input information such that the total length becomes a multiple of 512 bits.

2: Divide the padded message into 64-bit blocks. These blocks are processed in larger units of 512 bits.

3: Append a 64-bit block at the end, representing the original message length (before padding) in binary format.

4: **Define Processing Functions (f_t):**

- For $0 \leq t \leq 19$: $f_t(B, C, D) = (B \wedge C) \vee (\neg B \wedge D)$
- For $20 \leq t \leq 39$: $f_t(B, C, D) = B \oplus C \oplus D$
- For $40 \leq t \leq 59$: $f_t(B, C, D) = (B \wedge C) \vee (B \wedge D) \vee (C \wedge D)$
- For $60 \leq t \leq 79$: $f_t(B, C, D) = B \oplus C \oplus D$

5: **Define Processing Constants ($K(t)$):**

- For $0 \leq t \leq 19$: $K(t) = 0x5A827999$

- For $20 \leq t \leq 39$: $K(t) = 0x6ED9EBA1$
- For $40 \leq t \leq 59$: $K(t) = 0x8F1BBCDC$
- For $60 \leq t \leq 79$: $K(t) = 0xCA62C1D6$

6: **Initialize Word Buffers (H_i):**

- $H_0 = 0x67452301$
- $H_1 = 0xEFCDAB89$
- $H_2 = 0x98BADCFE$
- $H_3 = 0x10325476$
- $H_4 = 0xC3D2E1F0$

7: **Process Message Blocks (Iterative):**

- The message, now in 512-bit blocks, will be processed sequentially.
- For each 512-bit block, a series of 80 rounds of operations will be performed using the functions f_t, constants $K(t)$, and the word buffers H_0 to H_4.
- This process will update the values of the word buffers.

8: **Output State:** The final state of the word buffers (H_0, H_1, H_2, H_3, H_4) after processing all the 512-bit blocks. This output represents an intermediate stage, likely the input to the secret sharing scheme itself.

A novel algorithm is introduced for secure multi-user data sharing. This method divides the original data D into n distinct functional components or "shares": {d1, d2... dn}. Each share's composition is determined by the total number of shares (n) and a knowledge parameter k associated with each individual share Di. Critically, when k is less than n, the values of shared multi-keys—specifically the first (k-1) shares—remain protected. These initial shares can be utilized without requiring specific access credentials. This design draws inspiration from and builds upon the security principles underlying Shamir's Secret Sharing scheme. A key application of this approach lies in multi-key management based on a (k, n) threshold, particularly where n = 2k - 1. This configuration balances the number of shares needed for data reconstruction with the overall number distributed. The primary benefit of this algorithm is the enhanced confidentiality and privacy it affords to outsourced data. It is presented as a mechanism for establishing a secure database service by leveraging secret sharing techniques across multiple service providers.

3.3. Broadcast Group Multi-Key Management (BGKM) Schema

A BGKM scheme is designed to ensure that only authentic team members can access the shared group key. This is crucial in maintaining security in systems that involve multiple participants, such as collaborative cloud-based environments. In a BGKM system, certain security requirements must be met to ensure the integrity and confidentiality of the shared keys:

- **Appropriateness**: The BGKM scheme ensures that authorized team members can access the shared group key with a very low probability of failure. This requirement emphasizes that only legitimate participants can retrieve the key, and the likelihood of failure is kept at a minimal level, making the process reliable and efficient. The condition is typically expressed probabilistically, ensuring that authorized access is virtually guaranteed [21].
- **Soundness**: This requirement ensures that unauthorized individuals, lacking a valid Identify Specific Token (IST)**,** cannot manipulate the system to obtain the correct group key. The IST serves as a form of authentication, and it is crucial for preventing attackers from gaining unauthorized access during the key induction (initialization) phase. Only those possessing a valid IST can be granted access to the group key, thus maintaining the integrity of the access control mechanism[6].
- **Key Concealing**: Any group that does not have a valid IST should not be able to deduce the true group key from any publicly available information. The security system ensures that even if some public information is leaked, malicious actors cannot infer the actual group key. This is essential for maintaining confidentiality and preventing unauthorized decryption of sensitive data [6].
- **Forward/Backward Key Protection**: This ensures that when the group key is updated, former members who no longer belong to the group cannot retrieve the new key. Similarly, current members should not be able to access past keys once they have been updated. This protection is

achieved by using the IST, the previous key (K), and the updated public information. The update mechanism ensures that keys remain secure even as group membership changes, preventing key leakage [22].

- **Overhead of the Computation**: The computational cost associated with the encryption and decryption of data before and after outsourcing to cloud servers is an important consideration. When cloud storage systems handle sensitive data, ensuring the secrecy of the data and the efficient execution of cryptographic operations becomes critical. The BGKM scheme must account for these computational costs to avoid performance bottlenecks [23].
- **Additional Requirements**: In addition to the core security properties, there are other practical requirements that must be considered in the real-world deployment of a BGKM scheme. These include low packet latency, ensuring timely communication within the network, and ensuring service availability, which refers to the system's capacity to provide uninterrupted service even under high demand or during failures. These aspects are especially important in cloud-based environments, where system performance and reliability are critical to user satisfaction [24].

3.4 Attribute Based Encryption

Attribute-Based Encryption (ABE) with multiplier functions is examined. A foundational ABE scenario involves:

Initialization of system: System Initialization: A prime number p is selected, along with a generator g for groups of generative functions G_O and G_T of order q. A mapping function $e: G_0 \times G_0^i \to G_T$ is defined, coupled with a hash function $H: \{1,0\}^* \to G_0$, this hash function is combined with different attributes Lj, disjoint connection $\left(L_i \bigcap L_j = \phi \, for (i \neq j)\right)$, and then secret multi key of generative function is

$$SK \mid j \mid = \{x_i, b_i, i \in L_j\} \quad \textbf{(2)}$$

From secret multi key, generate public multi key from sources.

$$PK[j] = \left\{e(g,g)^{x_i}, g^{b_i}, i \in L_j\right\} \quad \textbf{(3)}$$

Multi-User Key Generation and Distribution: A multi-user v who needs access to content possesses a set of attributes denoted by I(v,j). For each attribute within this set, a corresponding secret multi-key sk is generated. This generation process employs a hash-based mechanism:

$$sk_{v,i} = g^{x_i} \mathrm{H}(v)^{b_i} \quad \textbf{(4)}$$

Here, L_j represents a specific attribute, $H(L_j)$ is the hash of that attribute, and b1, b2, ..., bm are user-specific secret exponents. These secret multi-keys are then securely distributed to the respective users. Decryption operations will also rely on these secret multi-keys. The public multi-keys, derived from the secret multi-keys, are used during the encryption process.

Data Encryption:

The sender, wishing to encrypt data, utilizes an Encrypt function within the Attribute-Based Encryption (ABE) framework. Crucially, the sender constructs an *access tree structure* to define the conditions under which the data can be decrypted. The resulting encrypted message, or ciphertext (CT), takes the following form:

$$CT = \left\langle R, \pi, ct_0 \left\{ct_{0,a}, ct_{1,a}, ct_{2,a}, \forall_a\right\}\right\rangle \quad \textbf{(5)}$$

Here $\pi(a)$ be the mapping connection R_a corresponding matrix with different attributes which are located and associated with access tree of data.

Decryption of data by receiver: Use decryption function i.e. $ABE.Dec\left\langle CT, \{sk_{i,v}\}\right\rangle$, CT be the encrypted text, receiver Vu explore cipher text CT with secret multi key sharing and then obtain output decrypted message with following conditions i.e.

a) For each attribute $a \in A^{'}, dec(a) = \dfrac{C_{1,a} e(H(v), C_{3,a})}{e(sk_{\pi(a)}, v, C_{2,a})}$

b) V_u evaluates $msg = C_0 / \Pi_{a \in A'} dec(a)$

4. IMPLEMENTATION AND CONSTRUCTION OF NACPPA

The implementation procedure and architecture of the proposed approach are mentioned in this section.

4.1 Implementation Procedure

The security implementation in the Novel Access Control- based Privacy Preserving Approach (NACPPA) involves several key computational steps, as described below:

- **System Setup:** This phase is managed by a trusted multi-user authority and coordinated by a third-party trusted entity. The setup process does not require any specific input aside from security parameters, which are defined in the form of attributes. These attributes are associated with the Global Public Key (GPK), which serves as the foundational public parameter for multi- user security.
- **Multi-Key Generation:** In this step, the attribute authority (denoted as AA) generates the required multi-keys. The inputs include a set of attributes (attr), a centre- specific public multi-key ($ucpk_i, k$), a verification key (VK_k), and a master multi-key (*MK*) related to the global public parameters (*GPK*). The output consists of a secret multi-key (SK_{ud}) assigned to the data multi-user (*DU*) and a corresponding secret multi-key pair for the data owner (*DO*).
- **File Encryption:** The encryption process is executed by the domain authority. It accepts inputs such as plaintext data (*m*), the global public key (GPK), the data owner's secret multi-key, and the data user's public key (pk_0). The encryption process transforms the plaintext into an encrypted format, producing cipher text ($CT_{fog\ comm}$), which is further broken down into multiple cipher texts ($CT\,1$, $CT\,2$) and securely stored in the cloud service provider's storage system.
- **Trapdoor Generation (GenTrap):** This method, handled by the data multi-user (DU), is responsible for generating a trapdoor - a cryptographic key used in searchable encryption. The process requires inputs such as search terms, secret multi-keys, and public multi- keys. The result of this computation is a Trapdoor (*tw*), along with a multi-key component related to the re-computation process.
- **File Decryption:** The decryption process is performed jointly by the data multi-user and the domain authority. The primary input is the encrypted cipher text, which contains partially encrypted data. The process also involves retrieving the re-encryption key (*RK*), which is used to convert the encrypted data back into its original plaintext format. The final output of this step is the fully decrypted text that can be accessed by authorized users.

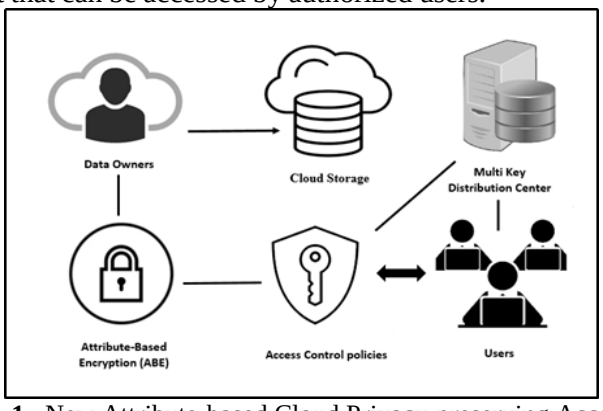

Figure. 1. New Attribute-based Cloud Privacy-preserving Access

4.2 Proposed NACPPA Schema

In this section, we explain the proposed NACPPA Schema, which enhances ABE to handle dynamic hierarchical structures involving multiple user groups. The schema is designed to manage the distribution of multi- keys and facilitate secure access to encrypted data while ensuring privacy and secure multi-user operations within the cloud environment.

Basic Process:

Figure 1 illustrates the proposed Access Control-based Privacy Preserving Model, which ensures secure

multi-user authentication and access control in a cloud environment. In this model, Data Owners encrypt their data using Attribute-Based Encryption (ABE) before uploading it to Cloud Storage. The encrypted data remains protected from unauthorized access, even from the Cloud Service Provider (CSP).

To regulate access, the system enforces Access Control Policies, which define decryption permissions based on user attributes. A Multi-Key Distribution Center (MKDC) manages cryptographic keys and distributes them securely to authorized users. This ensures that only users with the required attributes can decrypt and access the data.

Additionally, the Broadcast Group Multi-Key Policy (BGKM) is integrated with Group Key Management (GKM) to enhance scalability and minimize computational overhead. By combining these techniques, the model maintains security and privacy while supporting efficient multi-user access control.

- **Hierarchical Structure:** The schema operates in a multi-layered hierarchy where top-level domain author- ities control the security of data access and multi-key distribution. Subordinate domain authorities and multi- user groups receive encrypted multi-keys based on their access rights, ensuring that users only access data they are authorized for, as shown in Figure 2.
- **Role of Authorities:** The Domain Authority holds the primary control over the access control policies and is responsible for generating and securely distributing multi-keys to authorized users. Subdomain Authorities manage specific user groups and ensure that multi- key access is appropriately granted based on a user's attributes, credentials, and assigned roles.
- **Multi-Key Distribution:** The system generates multi- keys based on user attributes, which are then securely distributed by the domain and subdomain authorities. Multi-users within the system each have a unique multi- key structure based on their assigned attributes, ensuring fine-grained control over access and data decryption. The multi-key generation ensures that each user receives a decryption key tied to specific attributes, granting access only when the correct set of attributes matches the access policy.
- **Data Decryption:** Decryption of data is possible only for users who possess the correct combination of multi- keys based on their assigned attributes. The multi-key structure ensures that multi-users with different roles or attributes can securely decrypt the data, while unauthorized users are restricted from access, even if they possess one or more partial keys.
- **Trusted Multiuser Evaluation:** Trusted multi-user evaluations ensure that only authorized multi-users, deter- mined through attribute-based encryption (ABE), are able to access sensitive data. The privacy-preserving master multi-key parameters enhance the security model, ensuring that keys remain protected while facilitating safe and efficient access to encrypted resources.
- **Privacy and Security:** The NACPPA schema maintains a high level of security by ensuring that only multi- users with valid and matching credentials (multi-keys) can decrypt and access the data. The privacy-related master multi-key parameters further contribute to the overall security and privacy of the system.
- **parameters:** Help evaluate and enforce the privacy of users and sensitive data, making the system resilient against unauthorized access or data leakage.

The fundamental characteristics of NACPPA encompass the following methods:

1. System setup: Setup calculation method is executed by the authority of domain for the creation of multi key - public (PK) and multi key - master (MK). d is the length of multi key structure, by using relationship between bilinear maps. After that, random parameters are evaluated with C being the prime order p with gen c $\alpha, \beta \in Zp, \forall i\{1, 2\}$. Next, depending on the length of the multi key d, the creation of the master and public multi keys is

$$P_k = (\square, c, h_1 = c^{\beta}1\ , f_1 = c^{1/\beta}1\ , h_2 = \ \ c^{\beta}2\ , f_2 = c^{1/\beta}2\ , e(c, c)^{\alpha}) \qquad \textbf{(6)}$$

$$MK = (\mathcal{B}_1, \mathcal{B}_2, c^{\alpha}) \qquad \textbf{(7)}$$

The above equation shows the description of attributes with respect to cipher text and plain text between multi users in sharing of data.

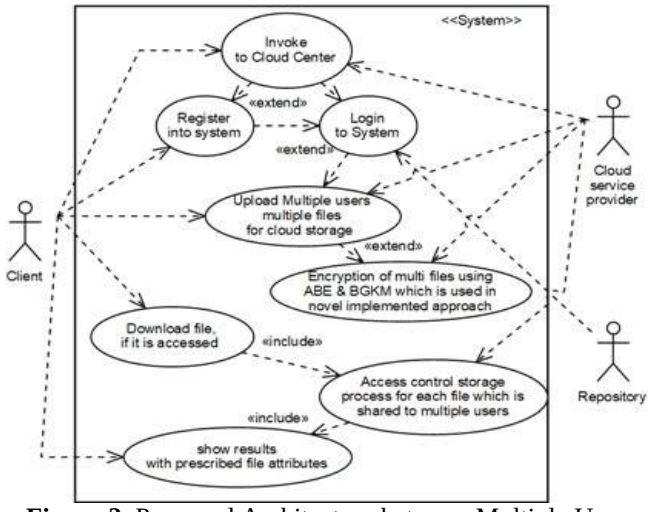

Figure 2. Proposed Architecture between Multiple Users

2. Creation and Grant of Domain Authority: Recursive attribute relations, or ai, j and related attributes {A0, A1, . . . , Am}, Ai = {ai,1, ai,2, . . . , ai,n} are how domain authority is arranged and linked, generates multiple keys for domain authority and chooses the number of multi-keys for the domain authority. Domain authority is organized and associated with recursive attribute relations i.e. with ai and associated attributes. Creates multi key for domain authority, select identity no. of multikeys r[k] for authority of domain. Random selection of identity multi keys for each multi user ai,j, $0 \le i \le m$, $1 \le j \le n$, it is utilized in authority of domain.

$$DA(MK) = \begin{pmatrix} A, D = c^{\frac{\alpha + r^{\{u\}}}{\beta_1}}, D_{i,j} = c^{r_i^{(u)}}.H(a_{i,j})^{r_{i,j}^{\{u\}}}, \\ D_{i,j}^{'} = c^{r_{i,j}^{\{u\}}} \; for(0 \le i \le m, 1 \le j \le n_i), \\ E_i = c^{\frac{r^{\{u\}} + r_i^{\{u\}}}{\beta_2}} \; for(1 \le i \le m) \end{pmatrix} \quad \textbf{(8)}$$

These features are employed once more in the domain authority master multi-key generation $r^{\{u\}}$ above, where E_i is the form of translation $r^{\{u\}}$ unique rule creation relates to attribute set Ai to $E_i / E_{i'}$ with associative translating elements Ei & Ei'. technique of decryption computation.

3. Multi-User Revocation: In cloud-based systems, when a multi-user is revoked, they must no longer have access to data shared by the data owner. To enforce this restriction, our implementation employs a re-encryption approach, ensuring that previously shared files remain inaccessible to revoked users. The NACPPA enhances attribute set-based encryption, allowing for the secure revocation of different multi- users. When a data owner shares files, an updated multi-key is generated specifically for revoked users, in accordance with security policies defined by the domain authority.

4. File Access Operations: When a group of users submits a request to the cloud server, the server encrypts the request and securely transmits it back to the users. The authorized users then apply the decryption process to access the updated data while ensuring that only permitted users can retrieve the shared information.

5. File Decryption Process: The decryption process re- quires the ciphertext and the corresponding multi-key structure as input. Initially, the system verifies the multi-user multi-key (*k*) against the associative access tree structure and the encrypted data received from the data owner. If the decryption conditions align with the required multi-key structure (*u*), the full content is successfully decrypted. Otherwise, the system evaluates and executes the decryption procedure accordingly. The decryption method operates as follows:

- The multi-key verification process ensures that only authorized users can decrypt the data.
- If the verification is successful, the cipher text is decrypted, granting access to the original

content.

- If verification fails, the decryption process is restricted, preventing unauthorized access.

$$De(TC, KS_u, i, t) = e\left(G_{i,j}, D_t\right) / \left(G'_{i,j}, D'_t\right) = e(x,x)^{r_i^{\{u\}}}.q_t(0) \quad \textbf{(9)}$$

The following describes the decryption algorithm used for all stored encrypted content with translated polynomial interpretation F.

$$E_{z)} = e\left(D_z, F_i / F_{i'}\right).E'_z = f\left(c,c\right)^{r^{\{u\}}}.q_z(0) \quad \textbf{(10)}$$

Based on the above decryption procedure message to be evaluated as $M = \hat{G}.F / e\left(G, D\right)$.

5. PERFORMANCE EVALUATION

To assess the effectiveness of the proposed approach, an experimental cloud environment was created, enabling multiple users to access files from different data owners securely. The NACPPA framework was implemented using ABE to ensure privacy-preserving access control. For the cloud environment setup, the implementation was carried out using the latest version of CloudSim, along with Java and NetBeans as development tools. The experimental setup consisted of hosts with the following specifications:

5.1 Setup and Multi – Key Generation in NACPPA

- **Setup_NACPPA:** This phase is responsible for generating both the public key (PK) and the master multi-key (MK), which serve as the foundation for secure access control. These keys are generated based on predefined cryptographic assumptions to ensure secure encryption and decryption processes.
- **NACPPA_MultiKeyGen:** In this step, the previously generated PK and MK are utilized to create multi-keys. These multi-keys are specifically structured to support private operations within the encryption framework. The multi-key structure can be implemented at different levels of depth, typically supporting one or two hierarchical functions, allowing for efficient key management and controlled access.

5.2 NACPPA Key Management and Encryption Processes

- **NACPPA_MultiKeyDeleg:** This process is responsible for delegating certain private multi-key operations to newly generated multi-users. Using the public key (PK) and master key (MK) linked to the domain authority, specific methods of the Domain Authority's (DA) private multi-keys are assigned to new users. The delegated multi-key within the domain authority is then utilized for executing private multi-key operations.
- **NACPPA_Enc:** Encryption is performed based on the conditions set by the access tree policy. Using the public key (PK), the system encrypts files, ensuring that only authorized users with matching attributes can decrypt them.
- **NACPPA_Dec:** This phase involves decrypting the encrypted files using the private multi-key assigned to the respective user. The decryption process ensures that only users who meet the defined access policies can retrieve the original content.
- **NACPPA_Rec:** Re-encryption is carried out in this step to enhance security. Using the public key (PK), all files are encrypted again with the private multi-key, allowing for secure re-encryption of both encrypted and decrypted files. The newly generated private key is then used to decrypt the file through encrypted file operations.

5.3 Experimental Results and performance Comparison

The experimental evaluation of the proposed NACPPA approach was conducted by measuring the time taken for various operations and comparing it with other authentication methods. The following figures illustrate the time taken for different authentication instances in a cloud-based client environment using the proposed approach.

To assess its effectiveness, the NACPPA model was com- pared with alternative authentication techniques, including Ciphertext Policy Attribute-Based Encryption (CP-ABE) [1], Hierarchical

Attribute-Based Encryption (HABE) [2], and the Recursive Matrix-based Oblivious RAM (RM-ORAM) model [25].

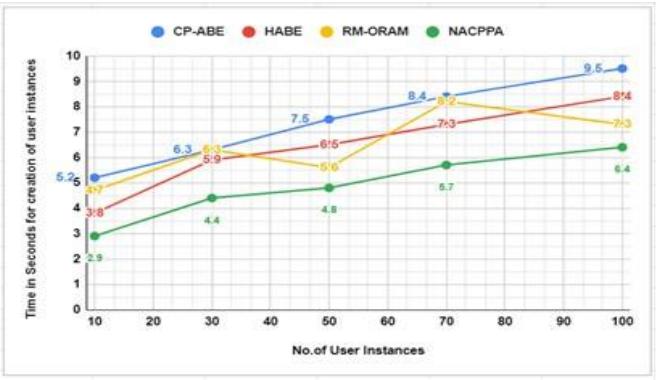

Figure 3. Performance assessments for all multi-user tasks in the cloud setup environment

The results obtained from the proposed approach demonstrate improvements in security response time, file encryption and decryption speeds, and access tree generation time for different multi-users. Additionally, the approach was evaluated for average accuracy and memory utilization in various multi-user operations such as file uploads, request processing, and download operations. These enhancements ensure efficient and secure data storage in a distributed cloud environment.

Table II presents a detailed breakdown of the total processing time required to handle multiple user instances, highlighting the efficiency of the proposed approach.

TABLE II: Total time values (in Seconds) for multi user instances

Users	CP-ABE	HABE	RM-ORAM	NACPPA
10	5.2	3.8	4.7	2.9
30	6.3	5.9	6.3	4.4
50	7.5	6.5	5.6	4.8
70	8.4	7.3	8.2	5.7
100	9.5	8.4	7.3	6.4

Table II and Figure 3 illustrate the time required to process multi-user instances on cloud platforms using traditional methods such as CP-ABE, HABE, and RM-ORAM. The results indicate that as the number of multi-user instances increases, these conventional approaches experience a significant rise in execution time for handling various user operations.

In contrast, the proposed NACPPA approach demonstrates lower execution time compared to existing methods. This improvement is attributed to its efficient access control mechanisms and optimized key management, which reduce processing delays while ensuring secure authentication and data access.

5.3.1 Encryption Time for Multi-User Instance Requests

Table III presents the encryption time required for different multi-user instance requests during the process of securely uploading original content to the cloud. The encryption time varies depending on the number of user requests and the size of the data being uploaded. The results indicate that as the number of multi-user requests increases, the encryption time fluctuates based on computational complexity and security overhead. The proposed NACPPA approach optimizes this process, ensuring efficient encryption while maintaining a high level of security for cloud-stored data.

TABLE III: Multi-user instances' encryption time values (in seconds)

Users	CP-ABE	HABE	RM-ORAM	NACPPA
10	5.3	4.7	4.6	4.5
30	6.4	5.8	6.3	3.1
50	7.4	7.7	6.3	3.3
70	8.3	7.2	8.2	4.7
100	9.6	8.4	9.4	7.6

5.3.2 Encryption Time and Performance Assessment

The evaluation results for encryption time and the performance comparison of various encryption techniques are presented in Table III and Figure 4**.** The findings indicate that as the number of multi-user instances increases across different cloud services, traditional encryption methods such as CP-ABE, HABE, and RM-ORAM require more time to process encryption operations. In contrast, the proposed NACPPA approach demonstrates faster encryption speeds, allowing multiple users to securely upload files with reduced processing delays. This improvement is attributed to optimized key management and efficient encryption algorithms, ensuring both security and performance in cloud-based environments.

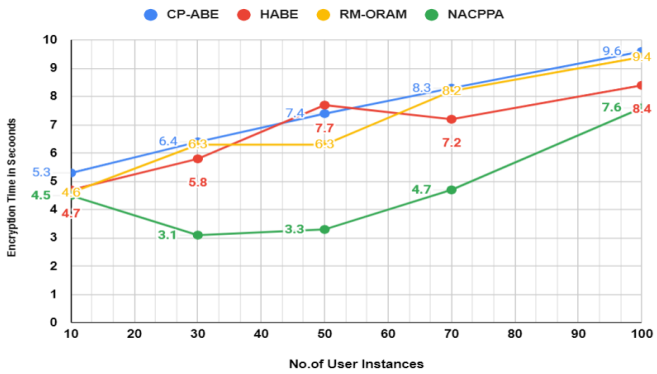

Figure 4. Comparing the effectiveness of various methods with encryption time

5.3.3 Decryption Time Analysis for Multi-User File Access

Table IV presents the decryption time required for multiple users to download shared files from various data owners in a secure cloud storage environment. The evaluation measures how efficiently the proposed NACPPA approach processes decryption requests compared to existing methods.The results demonstrate that as the number of multi-user requests increases, traditional encryption techniques experience higher decryption time due to complex key management and access control mechanisms**.** However, the NACPPA approach significantly reduces decryption time by leveraging optimized multi-key structures and efficient decryption algorithms, ensuring faster and more secure file access for authorized users.

Table IV: Decryption time values (in seconds) for multi user instances

Users	CP-ABE	HABE	RM-ORAM	NACPPA
10	4.8	5.8	5.3	4.7
30	5.3	6.7	5.9	3.7
50	4.8	8.5	6.4	3.3
70	7.4	6.9	5.7	3.9
100	6.8	7.5	7.9	6.4

5.3.4 Decryption Time and Performance Assessment

The evaluation results for decryption time and the performance comparison of different decryption algorithms are presented in Table IV and Figure 5**.** As the number of multi-user instances increases across various cloud services, traditional decryption methods like CP-ABE, HABE**,** and Recursive RM-ORAM require significantly more time due to higher computational complexity and key management overhead.

In contrast, the proposed NACPPA approach achieves faster decryption speeds**,** enabling multiple users to quickly access shared data across different cloud services. This improvement is attributed to optimized key delegation, efficient multi-key structures, and reduced computational overhead, ensuring secure and rapid data retrieval in distributed cloud environments.

International Journal of Environmental Sciences
ISSN: 2229-7359
Vol. 11 No. 17s, 2025
https://theaspd.com/index.php

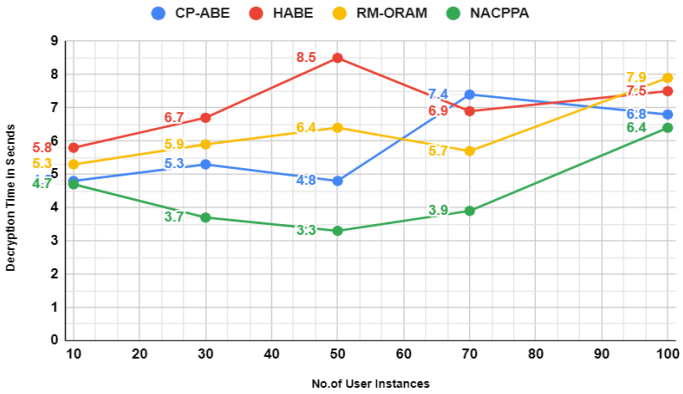

Figure 5. Difference between different approaches in terms of decryption time.

5.3.5 Memory Utilization Performance Evaluation

Table V and Figure 6 illustrate the performance evaluation of various approaches in terms of memory utilization when handling multi-user instance services. The assessment compares traditional methods such as CP-ABE, HABE, and RM-ORAM with the proposed NACPPA approach. As the number of multi-user instances increases, conventional techniques exhibit higher memory consumption due to their complex encryption and key management processes. In contrast, the NACPPA approach demonstrates optimized memory utilization, enabling more efficient handling of multi-user requests while maintaining security and performance in cloud environments.

Table V: Different memory usage values (in bits)

Users	CP-ABE	HABE	RM-ORAM	NACPPA
10	4652	3642	4887	4760
30	5327	4326	5226	5626
50	7356	6974	4745	4026
70	22132	5796	6354	5356
100	24553	8964	6785	4324

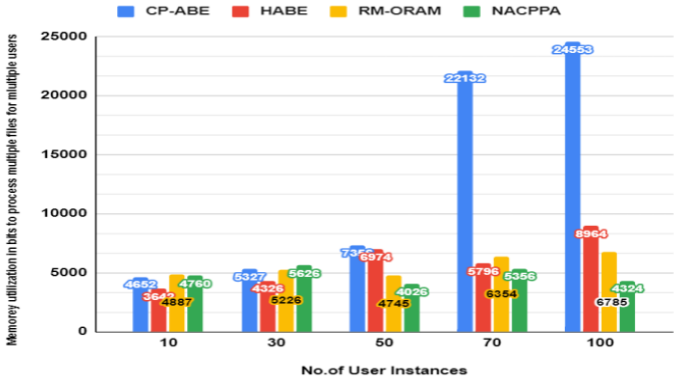

Figure 6. Performance evaluation of proposed approach with traditional approaches in terms of memory

5.3.6 Performance Evaluation of Access Tree Structure

Figure 7 presents the performance evaluation of the access tree structure in relation to the construction of **multi-user trees** for securely storing data. The evaluation compares how different approaches handle access tree generation, considering factors such as scalability, security, and efficiency in managing multi-user access control. As the number of multi-user instances increases, traditional models like CP-ABE, HABE, and RM-ORAM experience higher computational complexity in access tree construction. However, the proposed NACPPA approach optimizes this process, ensuring efficient tree formation, reduced overhead, and secure data organization**,** making it more suitable for dynamic cloud environments.

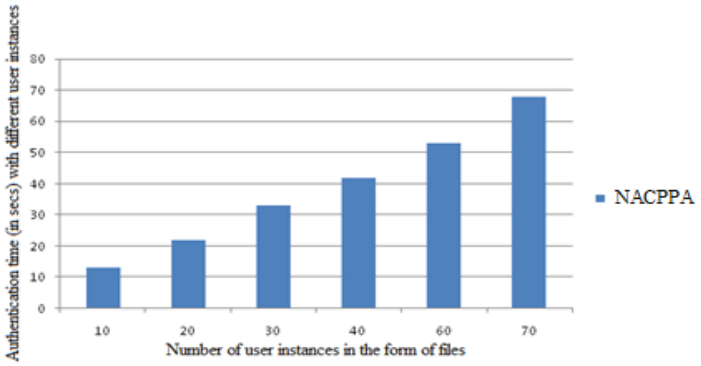

Figure 7. Performance evaluation of generation of access tree structure

5.3.7 Performance Analysis and Conclusion

Figures 3–7 illustrate the performance evaluation of the proposed NACPPA approach compared to conventional methods such as CP-ABE, HABE, and RM-ORAM in multi-file sharing cloud environments. The evaluation includes total multi-user instances, encryption and decryption times, and memory utilization. The results show that the NACPPA approach provides superior performance with enhanced security and efficiency.

6. CONCLUSİON

This article presents a privacy-preserving access control approach, NACPPA, designed to be collusion-resistant and untraceable, making it highly suitable for group data sharing in cloud storage. The proposed method aims to secure shared data and prevent unauthorized collaboration among malicious users. It employs an efficient key exchange mechanism to generate conference keys for authorized users while ensuring robust access control and data security in cloud environments. Experimental results indicate that NACPPA outperforms existing methods in terms of efficiency, scalability, and reliability for cloud-based data sharing. Future research will focus on enhancing the approach for multi-file sharing, dynamic user revocation, and collaborative key management in cloud computing.

7. DECLARATIONS

- Conflict of interest: The authors declare that they have no conflicts of interest related to the content of this article.
- Funding: No external funding was received for conducting this study.

8. REFERENCES

[1]. Bethencourt, J., Sahai, A., & Waters, B. (2007). Ciphertext-policy attribute-based encryption. *In 2007 IEEE Symposium on Security and Privacy (SP'07)* (pp. 321–334). IEEE. https://doi.org/10.1109/SP.2007.11

[2]. Wang, G., Liu, Q., & Wu, J. (2010). Hierarchical attribute-based encryption for fine-grained access control in cloud storage services. *In Proceedings of the 17th ACM Conference on Computer and Communications Security* (pp. 735–737). https://doi.org/10.1145/1866307.1866392

[3]. Zhao, Z., & Wang, J. (2017). Verifiable outsourced ciphertext-policy attribute-based encryption for mobile cloud computing. *KSII Transactions on Internet and Information Systems, 11*(6), 3254–3272. https://doi.org/10.3837/tiis.2017.06.016

[4]. Yang, A., Zhang, K., Liang, X., Fu, X., & Shen, X. S. (2020). Delegating authentication to edge: A decentralized authentication architecture for vehicular networks. *IEEE Transactions on Intelligent Transportation Systems, 23*(2), 1284–1298. https://doi.org/10.1109/TITS.2020.2964277

[5]. Ruj, S., Stojmenovic, M., & Nayak, A. (2013). Decentralized access control with anonymous authentication of data stored in clouds. *IEEE Transactions on Parallel and Distributed Systems, 25*(2), 384–394. https://doi.org/10.1109/TPDS.2013.19

[6]. Sreeram, G., Anandhamala, G. S., & Uma, G. V. (2016). Ensuring an efficient access control security in cloud computing using broadcast group key management. *Sylwan, 160*(6), 219–230.

[7]. Shang, N., Yu, H., Wang, X., & Liu, R. (2010). Broadcast group key management with access control vectors [Technical report].
[8]. Ateniese, G., Burns, R., Curtmola, R., Herring, J., Khan, L., Kissner, L., ... & Peterson, Z. (2007). Provable data possession at untrusted stores. *In Proceedings of the 14th ACM Conference on Computer and Communications Security* (pp. 598–609). https://doi.org/10.1145/1315245.1315318
[9]. Yu, S., Wang, C., Ren, K., & Lou, W. (2010). Achieving secure, scalable, and fine-grained data access control in cloud computing. *In Proceedings of IEEE INFOCOM 2010* (pp. 534–542). https://doi.org/10.1109/INFCOM.2010.5462174
[10]. Liang, K., Susilo, W., & Liu, J. K. (2015). Privacy-preserving ciphertext multi-sharing control for big data storage. *IEEE Transactions on Information Forensics and Security*, *10*(8), 1578–1589. https://doi.org/10.1109/TIFS.2015.2421873
[11]. Zhang, M., Zhang, Y., Wang, L., Wang, X., & Zhou, H. (2017). Obfuscating re-encryption algorithm with flexible and controllable multi-hop on untrusted outsourcing server. *IEEE Access*,*5*,26419–26434. https://doi.org/10.1109/ACCESS.2017.2764902
[12]. Zhang, K., Liang, X., Lu, R., & Shen, X. (2018). Efficient large-universe multi-authority ciphertext-policy attribute-based encryption with white-box traceability. *Science China Information Sciences*, *61*(3), 1–13. https://doi.org/10.1007/s11432-017-9333-6
[13]. Sun, W., Wang, B., Liu, N., Li, M., Lou, W., & Hou, Y. T. (2014). Protecting your right: Attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud. *In Proceedings of IEEE INFOCOM 2014*(pp.226–234). https://doi.org/10.1109/INFOCOM.2014.6847930
[14]. Hoang, T., Zhang, Y., Pham, H., & Guo, Y. (2018). Efficient oblivious data structures for database services on the cloud. *IEEE Transactions on Cloud Computing*, *9*(2), 598–609. https://doi.org/10.1109/TCC.2017.2679720
[15]. Yang, H., Wu, Q., Zhang, J., & Zhang, H. (2019). A privacy-protecting and resource-saving scheme for data sharing in smart home. *Journal of Internet Technology*, *20*(2), 607–615. https://doi.org/10.3966/160792642019032002036
[16]. Das, A. K., Wazid, M., Odelu, V., Kumar, N., Conti, M., & Jo, M. (2018). Biometrics-based privacy-preserving user authentication scheme for cloud-based industrial Internet of Things deployment. *IEEE Internet of Things Journal*, *5*(6), 4900–4913. https://doi.org/10.1109/JIOT.2018.2855188
[17]. He, D., Kumar, N., Kumar, M., & Choo, K. K. R. (2017). Certificateless public key authenticated encryption with keyword search for industrial internet of things. *IEEE Transactions on Industrial Informatics*, *14*(8), 3618–3627. https://doi.org/10.1109/TII.2017.2788009
[18]. Gentry, C., Halevi, S., & Raykova, M. (2015). Private database access with HE-over-ORAM architecture. *In Proceedings of the 13th International Conference on Applied Cryptography and Network Security (ACNS 2015)* (pp. 172–191). https://doi.org/10.1007/978-3-319-28166-7_9
[19]. Fisch, B., Vinayagamurthy, D., Boneh, D., & Popa, R. A. (2017). Iron: Functional encryption using Intel SGX. *In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (pp. 765–782). https://doi.org/10.1145/3133956.3134036
[20]. Pachala, S., Rupa, C., & Sumalatha, L. (2021). An improved security and privacy management system for data in multi-cloud environments using a hybrid approach. *Evolutionary Intelligence*, *14*, 1117–1133. https://doi.org/10.1007/s12065-020-00489-w
[21]. Zou, X., Dai, Y. S., & Bertino, E. (2008). A practical and flexible key management mechanism for trusted collaborative computing. *In Proceedings of IEEE INFOCOM 2008* (pp. 1211–1219). https://doi.org/10.1109/INFOCOM.2008.193
[22]. Wan, Z., & Deng, R. H. (2011). HASBE: A hierarchical attribute-based solution for flexible and scalable access control in cloud computing. *IEEE Transactions on Information Forensics and Security*, *7*(2), 743–754. https://doi.org/10.1109/TIFS.2011.2177720
[23]. Barsoum, A., & Hasan, A. (2012). Enabling dynamic data and indirect mutual trust for cloud computing storage systems. *IEEE Transactions on Parallel and Distributed Systems*, *24*(12), 2375–2385. https://doi.org/10.1109/TPDS.2012.285
[24]. Nabeel, M., Shang, N., & Bertino, E. (2012). Privacy preserving policy-based content sharing in public clouds. *IEEE Transactions on Knowledge and Data Engineering*, *25*(11), 2602–2614. https://doi.org/10.1109/TKDE.2012.109
[25]. Gordon, S., Gordon, M. J., Choi, J., & Kang, M. S. (2016). A matrix based ORAM: Design, implementation and experimental analysis. *IEICE Transactions on Information and Systems*,*99*(8),2044–2055. https://doi.org/10.1587/transinf.2015FCP0002