Hierarchical Access Structure-based Scheme with Outsourcing and Revocation Mechanism for Cloud Environment

Tabassum N. Mujawar[1], Lokesh B. Bhajantri[2], and Ashok V. Sutagundar[2]

[1]*Ramrao Adik Institute of Technology, D Y Patil deemed to be University, Navi Mumbai, India,*
[2]*Basaveshwar Engineering College, Bagalkote, India*

https://doi.org/10.26636/jtit.2023.4.1299

Abstract — Ciphertext policy attribute-based encryption (CPABE) is one of the efficient implementations of encrypted access control scheme for cloud computing. Though multiple implementations of CPABE exist, there are some issues that need to be addressed, including efficient revocation approach, decryption time, storage cost etc. In this paper, an efficient scheme that incorporates a hierarchical access structure, outsourced decryption, as well as user and attribute revocation is presented. The hierarchical access structure is utilized to encrypt multiple data using one common access structure and makes the encryption process more efficient. The outsourcing server is used to perform partial decryption, so that all heavy computations are performed by this server and less overhead is incurred by the data user. The proposed framework also integrates the evaluation of trustworthiness of data users and service providers to ensure trusted and encrypted access control procedures. The paper also presents an analysis of the time required for performing different operations. Simulation results show that the proposed scheme outperforms the existing approaches.

Keywords — attribute revocation, cloud computing, hierarchical access control, outsourced decryption, trust

1. Introduction

The various benefits of cloud computing, such as lower cost, simpler management, on-demand services, and pay-per-use have convinced many organizations to adopt cloud technologies. There are certain security issues that need to be dealt with while utilizing cloud services, as traditional security mechanisms are not applicable in this particular case. The data is stored in a cloud environment that remains outside the organization's control and is handled by untrusted service providers. In such a case, maintaining the security of the data stored in the cloud environment is the most important issue. It is equally important to ensure that the data is accessed by the authorized entities only. Suitable access control methods are applied to enforce the required access policies, so that the data is available only to legitimate users.

Attribute-based encryption (ABE) [1] is a widely used approach relied upon to implement access control procedures for cloud environments. The ABE scheme is basically a variety of the identity-based encryption scheme which provides access based on specific attributes and access policies relying on those attributes. The ciphertext that is encrypted using ABE is decrypted only if the attributes satisfy the requirements of the access structure. Ciphertext policy attribute-based encryption (CPABE) [2] and key policy attribute-based encryption (KPABE) [3] are two different ways of implementing this scheme. In the case of CPABE, the attributes and the secret key are associated with each other, with the access structure and the ciphertext being tied up as well. As far as the KPABE scheme is concerned, the attributes are associated with the ciphertext and the access structure is related with the secret key. CPABE is used most widely, as it is efficient, collusion-resistant and most secure. The key issues that need to be addressed include the complexity of paring operations, the time required for encryption and decryption, efficient revocation mechanisms, and storage cost. The decryption operation used in the CPABE scheme is very complex, as many pairing operations are involved therein. Also, if large amounts of data are to be stored, then the time required for encryption and the storage cost are considerable as well. The required changes need to be introduced, if any user revokes the system. Also, if any attributes are revoked, the system's setup must be updated as well. The existing revocation mechanism is not efficient, as major updates need to be performed to support user and attribute revocation processes.

In a cloud environment, users are fully dependent on service providers. Therefore, it is essential to build a trusted relationship between those two groups. This helps establish a certain level of trust that the providers and users are interacting with trustworthy entities only. Traditional access control models fail to take the aspect of trust into consideration while setting up the access control mechanism. Hence, in this paper, an approach to measure the level of trustworthiness of specific cloud service providers, i.e. CSPs, and data users, is embedded within the proposed access control scheme.

1.1. Our Contribution

This paper presents an efficient, revocable, outsourcing, hierarchical access structure and trust-based access control (ROHAS-TBAC) scheme. The major contribution of the paper is described below.

– An access control technique incorporating hierarchical access structures is developed.
– Complex paring operations are outsourced to a semi-trusted server. A verification process is also embedded to test the correctness of transformations performed by the outsourcing server. The hashing and encapsulation-based mechanism follows to verify whether the intermediate ciphertext generated by the outsourcing server is correct.
– A direct and immediate user revocation mechanism relying on maintaining two separate revocation lists for revoked users is developed. The revocation information is integrated in the ciphertext as its distinct component, so that ciphertext update processes become efficient. Only revocation-related components need to be modified during the ciphertext update process.
– The attribute revocation mechanism relying on maintaining an attribute revocation list is proposed. The revocation information is integrated with the key, as a separate component.
– The proposed work ensures trusted access to the data by integrating the trust computation scheme. Trust estimation is performed before granting access to the data.

The remaining part of this paper is organized as follows. The existing methods relying on outsourcing mechanisms and revocation are presented in Section 2. The proposed scheme is described in Section 3, and the experimental analysis is presented in Section 4. The conclusion is given in Section 5.

2. Related Work

The revocable and multiple authority-based CPABE scheme is presented in [4]. The scheme also supports multi-valued attributes. In order to implement the revocation mechanism, a binary tree construction and unique identities for all users are utilized. The multiple authorities are responsible for the key distribution process and the scheme supports attribute revocation based on these multiple authorities. Generally, the access structure associated with ciphertext is presented in plaintext form. Hence, it is possible for other entities to know the values of attributes. This may harm the privacy of both data owners and users.

A CPABE scheme with partially hidden access policies is proposed in [5]. In this scheme, the issue of attacks guessing the attribute values is addressed. The scheme is implemented in such a way that no third party can guess any attribute values. The paper presents an outsourced decryption testing mechanism that also preserves the privacy of attribute values. During the decryption test phase, the cloud server cannot obtain any knowledge about the attribute values. Thus, the scheme maintains the privacy of attributes and reduces the overhead by outsourcing the decryption operation to a cloud server.

In [6], an ABE scheme suitable for edge computing environments and supporting healthcare systems is proposed. Here, the costly encryption and decryption operations are outsourced to the edge nodes. The method also supports the attribute update functionality. The key update algorithm is invoked whenever any attribute value is modified. The ciphertext update process is also implemented, so that the recent values of attributes can be incorporated. Due to the outsourcing approach, the overall burden on resource-constrained devices is reduced.

A CPABE scheme integrated with the broadcast mechanism is presented in [7]. This method is based on the standard identity-based encryption mechanism, where ensuring complete anonymity and preserving privacy are the two major issues addressed. The proposed solution utilizes type 3 bilinear paring and supports constant ciphertext sizes, as well as constant decryption times and key sizes. In general, the decryption process requires some information about the data user and access policy while decrypting the corresponding ciphertext. This approach reveals the information about the data user and does not preserve anonymity – as step that is ensured by eliminating access structure- and receiver-related information from the ciphertext.

A CPABE scheme that incorporates such key features as complete outsourcing, multiple keyword search and verification, is presented in [8]. In this scheme, the majority of the complex and time-consuming computations are outsourced to third party cloud servers. These computations include key generation, key verification, encryption, keyword encryption, test algorithms and trapdoor generation. The scheme also utilizes a verification mechanism to check the correctness of computations performed by cloud servers while generating the private keys. The facility to apply multiple keyword searches is also supported by maintaining privacy of the keywords.

Trapdoor generation, keyword encryption and test algorithm processes are employed to facilitate multiple keyword searches. Thus, the scheme reduces the overhead by supporting outsourced decryption and ensures that the verification of key generation computations is carried out. The basic construction of CPABE allows to decrypt the ciphertext upon satisfaction of access structure requirements. Unfortunately, it is not possible to decrypt the ciphertext in a collaborative manner. In certain scenarios, the data owner allows collaborative decryption and, hence, a need to build a scheme supporting collaborative decryption exists.

A CPABE scheme that supports collaborative decryption with the help of blockchain technology is described in paper [9]. In this method, various user groups are considered. If any user cannot decrypt the ciphertext alone, then the attributes of the other users can be used collaboratively to decrypt the ciphertext in question. Collaborative decryption is possible only when the owner of the data has given relevant permission. The scheme also supports multiple authority-based key distribution. Thus, single authority-related issues, such as a single point failure or compromised authority, etc., may be resolved.

In [10], an efficient and flexible access structure based CPABE scheme that supports attribute revocation is presented. Here, the attribute revocation list and binary tree representation are used to support the attribute revocation mechanism. An indi-

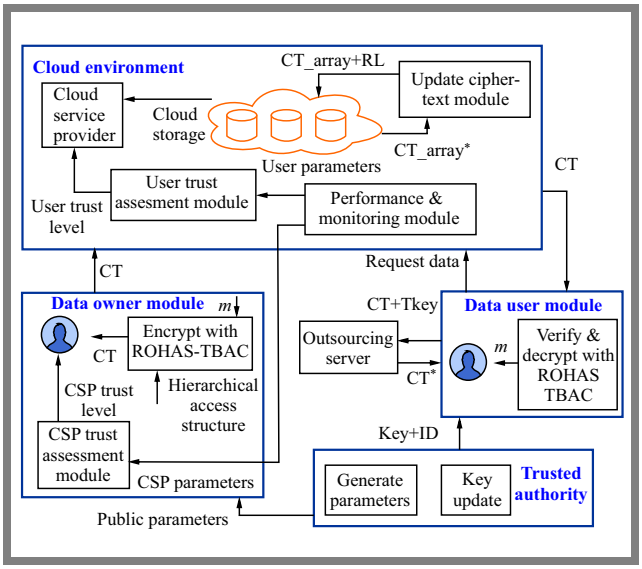

Fig. 1. Architecture of the proposed system.

rect approach is implemented, where the key update process is handled by a third-party key authority. An efficient key update algorithm for non-revoked users is implemented to support the attribute revocation process. The binary tree representation makes the scheme more efficient with regards to the key update process and shortens the overall time needed for key generation. The access policies generated by this method are more expressive, as threshold gates are used as well. Paper [11] presents a CPABE scheme with an effective policy and file update functionality. In this proposal, the majority of the policy update procedure is performed by the proxy cloud service provider. The file update model and the policy update functionality are utilized to mitigate security-related risks and the overall cost. The initially computed encryption components are sent to the service provider and then the remaining part of the policy update process is completed on the service provider's side. The scheme reduces communication overhead and computation cost.

The majority of the existing CPABE implementations rely on outsourcing the decryption process. The encryption process also incurs certain amount of overhead. Hence, an approach to outsource both these processes to fog nodes is proposed in [12]. In this scheme, the fog nodes are considered as computationally efficient and, hence, all major computations are moved thereto. This reduces overhead for both the data owner and data user. The scheme also implements a method for immediate revocation of attributes by utilizing a group key. It supports multiple authorities and a large set of attributes. In the basic CPABE construction, it is not possible to create groups of users those possesses with similar attribute sets.

The scheme presented in article [13] proposes a way to incorporate unique identities along with the private key and then utilizes these IDs, combined with attributes, to form certain groups. The scheme includes a mechanism for attribute grouping key. As soon as any attribute is revoked, the secret keys are updated. The functionality of outsourcing certain complex decryption and decryption operations to the fog nodes is included as well. The existing CPABE implementations support any of revocation, traceability, and hidden policy features. If the scheme supports revocation and traceability, then the privacy of the policy is not maintained. Also, if it supports hidden access policies, then the tracking of malicious entities or revocation are not possible.

The scheme that incorporates the revocation functionality and hidden access policies is described in [14]. The revocation mechanism is designed using a revocation list which is attached to the ciphertext. Revocation information is represented in the form of a binary tree. This scheme is beneficial as far as updating the ciphertexts is concerned, as fewer computations are needed for this process. A method to identify the malicious users also developed as well.

A user revocation method which relies on the validity of the users' secret keys and maintains the revocation list is presented in [15]. This scheme relies on direct revocation for users. The size of the revocation list is also reduced by excluding users with an expired validity period. Some other implementations of the CPABE scheme are also presented in [16]–[21].

3. Proposed Work

The key components elements of the proposed system include the following: cloud environment, data owner, data user, outsourcing server, and trusted authority. The architecture of the proposed system is shown in Fig. 1.

Data owner. The data owner utilizes the hierarchical access structures and computes the ciphertexts. The data owner verifies the trust level of the CSP before sending any data to cloud storage. The ciphertext is sent to a trusted service provider for storage. The additional components related to the revocation mechanism, such as the revocation list and the encryption time, are integrated with the ciphertext.

Data user. The actual data is recovered by the data user if the user's secret key holds the required attributes capable of satisfying the hierarchical access structure. Before decryption, care is also taken to make sure that the data user and/or any attributes of that user are not revoked from the system. The data user module includes also a verification and decryption component. The outsourcing server computes an intermediate form of the ciphertext that is sent to data user. The verification and decryption module checks the correctness of the transformation process. If the transformation is performed correctly, the decryption algorithm is applied to generate the plaintext.

Trusted authority. A trusted server is also included in the system, offering two major functionalities, namely generating the necessary parameters, and handling the revocation process. It issues public parameters, keys and identities to the data owner and data user. The authority includes the key update module for modifying user's secret key after a specific time interval. Whenever any attribute of any user is revoked, a new attribute revocation list is prepared and the secret key of the user is updated after a specific time interval.

Cloud environment. The cloud environment includes CSP, user trust assessment module, performance and monitoring

module, storage servers, and ciphertext update module. The previously generated ciphertexts are updated after a specific time interval by the ciphertext update module. The user revocation list is modified when any revocation event occurs. The old ciphertexts are updated according to this modified revocation list. The CSP handles all requests from data users and data owners. The data is accessible to authorized and trusted users only. The performance and monitoring module is responsible for maintaining all the necessary parameters related to CSP and data users that are relied upon for trust assessment. The user trust assessment module takes the necessary user behavior-related parameters as input and generates trust levels for the data user. If the data user is trustworthy, then only the requested ciphertext is sent to that data user.

Outsourcing server. This module performs partial decryption of the received ciphertext. Before performing partial decryption, the user revocation list and the attribute revocation list are checked. If the user's ID or any attribute are not present in any of the revocation lists and the user's secret key holds the required attributes, then the transformed ciphertext is generated.

3.1. Proposed ROH-TBAC Scheme Overview

On the data owner side, a different hierarchical access structure is generated by combining the access structures which have a mutual hierarchical relationship. The advantage of generating such a hierarchical access structure is that the same access structure may be used while encrypting multiple data. There is no need to build separate access structures for each file that the data owner wants to store in a cloud environment. This approach may reduce the encryption lead time simultaneously reducing the amount of storage required for multiple ciphertexts. A specific part of the ciphertext is decrypted depending on whether the data user's attributes satisfy the access structure fully or partially.

The proposed scheme facilitates the revocation schemes sed both for users and attributes. The proposed scheme utilizes the user revocation list and the attribute revocation list to implement the revocation mechanism. Whenever any attribute is revoked by the user, it is immediately added to the attribute revocation list (ARL) associated with that specific user. Then, the binary tree presented in [15] is created to represent all attributes present in the system. The attribute revocation list is embedded with the user's secret key and the attribute revocation mechanism is implemented. The revocation information is embedded as a separate component in the user's secret key. Hence, while implementing the key update functionality, only the revocation-related part is updated and the rest of the components remain unchanged. This approach helps reduce the time required to update the user's secret key.

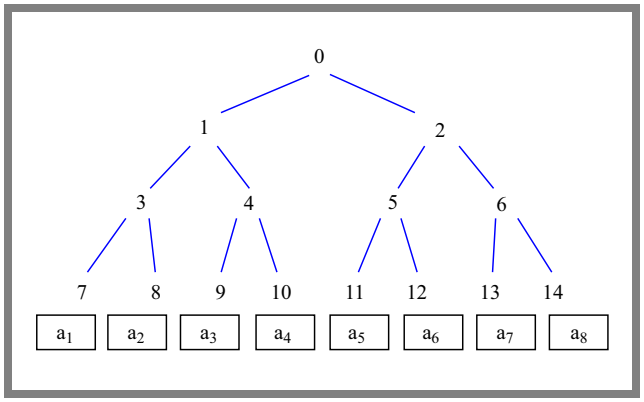

Fig. 2. Tree representing attribute revocation-related information.

If there are total n attributes present, then the revocation tree consists of $2n-2$ nodes. The leaf nodes in the tree represent the individual attributes. The nodes of the tree are labeled in breadth first approach starting with root node with label 0. The minimum cover approach is followed to store the attribute revocation information. Two functions – path(k) and min_cover(ARL) – are used to maintain the revocation information and implement the attribute revocation scheme. Path(k) returns the path for leaf node k, starting from the root of the tree.

For any user ($user_i$), the min_cover(ARL_i) function generates a list of the minimum number of nodes that are needed to represent the non-revoked attributes. To check whether the particular attribute has been revoked or not, the intersection of path(a_j) and min_cover(ARL_i) for each attribute a_j associated with the ciphertext is computed. If there is only one node present at this intersection, then that particular attribute is not revoked. For example, the tree shown in Fig. 2 means that there are eight attributes in the system.

The path(9) function returns the path of leaf node 9 from the root node, i.e. path$(9) = \{0, 1, 4, 9\}$. Consider that $user_i$ holds, initially, four attributes a_1, a_2, a_3, and a_4. Then, attribute a_3 is revoked. Then, the function min_cover$(ARL_i) = \{3, 10\}$. This set represents the non-revoked attributes of $user_i$. $User_i$ wants to decrypt the ciphertext with the access structure that includes attributes a_1, a_2, and a_3. To check whether attribute a_1 is revoked or not, let us compute path(7). The function path$(7) = \{0, 1, 3, 7\}$. Now, we compute, path$(7) \cap$ min_cover$(ARL_i) = \{3\}$. As there is only one node present at the intersection, attribute a_1 is not revoked. To check for attribute a_3, let us compute path$(9) \cap$ min_cover$(ARLi) = \emptyset$. Hence, attribute a_3 is considered to be revoked. The ciphertext is not decrypted by $user_i$.

The proposed scheme supports also the direct and immediate user revocation mechanisms from [22]. The key features of this scheme are the validity times applicable to the key, the revocation list (RL) and the immediate revocation list (IRL). Here, each user is assigned a unique identity during the system's setup phase. Whenever a given user is revoked from the system, their ID added in the revocation list. Therefore, the user cannot access any newly generated ciphertexts. In order to keep the revocation list shorter, a validity period is associated with the user's secret key. The users with an expired key can be removed from the revocation list. Such an approach helps keep the list short. The ciphertext update process is implemented to ensure that the revoked users cannot access any previously generated ciphertexts.

The revocation information is integrated with ciphertext, constituting its separate part. Hence, the ciphertext update

process needs to modify the revocation-related part only and the rest of the components of ciphertext remain unchanged. This approach can reduce the time required for updating all previously generated ciphertexts and reduces overhead for the entire system. The proposed method also maintains a separate revocation list for users who left the system before their key validity expired. This list is useful to prevent users from accessing old ciphertexts, until they are updated.

The decryption algorithm checks the user's identity using both revocation lists. If it is present in either of the lists, the ciphertext is not decrypted. If the user's ID is not present in both revocation lists, then the validity period of the user's key is compared with the period of time associated with a given ciphertext. If the validity period is expired, the ciphertext is not decrypted. The time period is represented using a tree that is similar to that shown in [22].

The proposed method incorporates also a mechanism for outsourcing the decryption operation to a third-party server. This server performs all the major computations that include complex pairing operations. This approach helps minimize the overhead incurred on the data user side due to the complexity of decryption-related operations. A verification mechanism is implemented as well to ensure that the transformation performed by the outsourcing server is correct. The trust computation scheme for CSP and data user that is based on behavior and feedback, as presented in [23], [24], ensures that the data is handled by trusted providers and accessed by trusted users.

3.2. ROH-TBAC Scheme Details

The proposed scheme includes different modules responsible for initial setup, encryption, key generation, decryption, partial decryption, verification, ciphertext update and key update. The hierarchical access structure is used while encrypting any message. The liner secret sharing scheme (LSSS) matrix is used to represent the access policies. In LSSS, the set of parties P is present and each party shares a secret. In an access structure utilizing the LSSS scheme, the attributes play the role of parties.

Consider that there is a given attribute set such as $A = \{\mathrm{a}_1, \mathrm{a}_2, \mathrm{a}_3, \ldots, \mathrm{a}_n\}$. There exists a function $\rho(i)$ that maps the i-th row of the access matrix to a particular attribute. Consider that secret s is to be shared, such that $s \in Z_p$. Then, other values $v_2, v_3, \ldots, v_n \in Z_p$ are selected randomly and a vector $v = \{s, v_2, v_3, \ldots, v_n\}$ is generated. Then, the matrix $\lambda = M.v^T$ is computed. λ_i represents the share corresponding to the attribute returned by $\rho(i)$. At the receiver's end, the secret can be reconstructed if the required numbers of shares are available. If there is set of authorized attributes S and $I = \{i : \rho(i) \in S\}$, then there exists a constant w_i such that:

$$s = \sum_{i \in I} w_i \lambda_i.$$

If the user's secret key possesses the required attributes, then the necessary secret value can be reconstructed and the imposed access structure can be satisfied.

Phase 1 – initial setup. This phase computes the public key $Pkey$ and the master key $Mkey$ along with the necessary revocation-related components. Consider that there are n users and m attributes present in the system. Let R represent a tree that holds information about user revocation and let T represent a tree that holds the details about the time period. The attribute revocation information is also represented using a binary tree AR. The necessary components for each node of all these trees are computed. Here, bilinear group $\mathbb{G}$ that has a generator g and a prime order p is selected. Also, a bilinear map $e : \mathbb{G} \times \mathbb{G}$ is considered. Algorithm 1 shows the entire process.

Phase 2 – encryption. This phase takes a public key and the LSSS matrix as input. The LSSS matrix is represented by M and contains l rows and n columns. Along with the generated ciphertext, additional information, such as revocation list (RL) and time period T_e, is embedded. Algorithm 2 depicts the entire process.

Phase 3 – key generation. The set of attributes A, $Pkey$ and $Mkey$ are used for generating the secret key. The attribute revocation list ARL is maintained and the attribute revocation information is embedded in the key. Algorithm 3 depicts this phase. Let T represent the time period. The secret key components are computed by randomly selecting element b. The component for each element of the minimum cover of ARL is also computed by using the components computed for each attributes during the initial setup phase. Similarly, components for every member of T are also computed using time period components computed during the initial setup phase.

Phase 4 – transformation key generation. This phase generates the transformation key $Tkey$, which is needed to apply the outsourced decryption mechanism. This phase also generates the retrieving key ReK that is used to perform complete decryption on the user side. The third-party outsourcing server utilizes $Tkey$ to generate a partly decrypted ciphertext. This phase takes the public key and the secret key as input and produces $Tkey$ and Rek as output, as shown in Algorithm 4.

Phase 5 – partial decryption. This phase generates a partially decrypted ciphertext. The input provided to this phase includes transformation key, ciphertext, and all revocation lists: RL, IRL, and ARL. This phase does not perform partial decryption if any of the following conditions become true:

- the user's secret key does not hold sufficient attributes to satisfy the access structure,
- the RL or IRL includes the user's identity,
- the user's secret key validity period is expired,
- any of the user's attributes are present in ARL.

If none of the above-mentioned conditions are true, then this phase generates the transformed ciphertext. The corresponding secret is reconstructed after satisfying the access structure (Algorithm 5).

Phase 6 – decryption. The transformed ciphertext $CT_i^{''}$ and the retrieving key ReK are provided as input. The plaintext is recovered as output. The transformed ciphertext is completely

decrypted by applying a simple operation as:

$$m_i = \frac{T_{1,i}}{(T_{2,i})^z} \ . \tag{1}$$

Phase 7 – updating ciphertext. The ciphertext update process ensures that old ciphertexts are not accessible to the revoked users. For this purpose, the modified revocation information is embedded in all previously computed ciphertexts. The component that is related to the revocation is updated in this phase. This phase takes an array of previously computed ciphertexts and a modified revocation list RL' as input (Algorithm 6).

Phase 8 – updating the key. This phase updates the user's secret key if any attribute is revoked or if the validity period for a specific key expires. The secret key $SKey$ and the updated attribute revocation list ARL' are the input for this phase. Min_cover(ARL') is computed for the updated attribute revocation list. The Algorithm 7 shows the entire process.

Phase 9 – verification. The array $M = \{m_1, m_2, \ldots, m_k\}$ represents the symmetric encryption keys that are randomly chosen by the data owner for encrypting each actual message. Algorithm 2 is used to encrypt this array and apply the proposed scheme. The symmetric encryption algorithm (i.e., AES) is applied to encrypt each actual message using the secret keys in M and an array of ciphertexts is generated as $F\_CT = \{fCT_1, fCT_2, \ldots, fCT_k\}$. Thus, the CT generated by Algorithm 2 and F_CT are stored in the cloud. Whenever any data user requests data from the cloud, a partial decryption of the ciphertext of symmetric key CT_i is done by the outsourcing server and the transformed ciphertext $CT_i^{''}$ is sent to the data user.

The data user applies the proposed decryption operation mentioned in phase 6 and computes m_i. Then, the data user computes $Tag = H(m_i) \| fCT_i$, which is used to check the correctness of the transformation performed by the outsourcing server. If the received tag and the newly computed one are similar, then only the recovered symmetric key is correct. Then, this key is used to decrypt the original message. Otherwise, the algorithm returns a null value.

4. Performance Analysis

The proposed scheme is implemented in Java and the JPBC pairing library is utilized [26]. While implementing the scheme, the scenario of an organization maintaining all data in cloud storage is considered. The different access policies are designed by considering the importance of data. The proposed CPABE scheme is used to encrypt the secret key and this key is used to encrypt the actual data by applying a public key cryptography algorithm, such as AES. The employees can access data from cloud storage only if they have sufficient attributes to satisfy the hierarchical access structure. The various attributes are associated with all employees of the organization depending on their roles, responsibilities, positions, etc. If the attributes of any employee satisfy the access structure associated with any ciphertext, then that specific person will obtain the secret key required for decryption. In order to analyze the performance of the proposed scheme, the traditional CPABE scheme and other two existing schemes are implemented as well.

Algorithm 1. ROHAS-TBAC initial setup

Input: system parameters
Output: $Pkey$ and $Mkey$
Start

Randomly choose α and β over $\mathbb{Z}p$
for each attribute a_l in U **do**
    Randomly choose $h_l \in \mathbb{G}$
end for
Compute $X = e(g,g)^{\alpha}$ and $Y = g^{\beta}$
for $i = 1$ to T **do**
    Randomly select vt_i over $\mathbb{Z}p$
    Compute $vt_i^{'} = g^{vt_i}$
end for
for $j = 1$ to $2n-2$ **do**
    Randomly select r_j over $\mathbb{Z}p$
    Compute $r_j^{'} = g^{r_j}$
end for
for $k = 1$ to $2m-2$ **do**
    Randomly choose ar_k over $\mathbb{Z}p$
    Compute $ar_k^{'} = g^{ar_k}$
end for
Compute $Pkey = \left\{\mathbb{G}, g, Y, X, h_l, vt_i^{'}, r_j^{'}, ar_k^{'}\right\}$
    $\forall l \in (1,u)$,
    $\forall j \in (1, 2n-2)$,
    $\forall i \in (1,T)$,
    $\forall k \in (1, 2m-2)$
Compute $Mkey = \{g^{\alpha}, vt_i, r_j, ar_k\}$
    $\forall j \in (1, 2n-2)$,
    $\forall i \in (1,T)$,
    $\forall k \in (1, 2m-2)$

End

Unique identities are also associated with every employee of the organization for implementing the revocation mechanism. User revocation and attribute revocation processes are performed by maintaining a list of revoked users and attributes, respectively. A separate immediate revocation list is maintained to support immediate revocation. The ciphertext update module updates all the generated ciphertexts in one year intervals. The secret keys of all employees also have a validity period assigned, usually lasting one year. The key update module is also implemented to update all secret keys after one year.

The proposed scheme maintains a shorter user revocation list by associating the validity time with the key. The users are removed from the revocation list as soon as their validity period expires. Such an approach helps reduce the encryption lead time. The different scenarios are implemented by considering access policies with ten attributes and with 50% of the revoked users having expired keys. The total number of users considered for the purpose of this evaluation is 50. A comparative analysis of the encryption time is performed by changing the number of revoked users to 10, 15, 20, 25, and 30. Figure 3 shows that the proposed scheme requires a short-

er encryption lead time as the revocation list becomes shorter. Figure 4 shows an assessment of the time needed to update the secrets of users whenever their attributes are revoked. The revocation information is embedded as a separate component in the user's secret key. Therefore, whenever the key is updated, only the revocation related part needs to be updated. In other existing systems, such as the one described in [15], the entire key needs to be updated whenever any attribute is revoked. Thanks to this, the proposed method requires less amount of time as compared with [15]. The total number of 15 attributes and 30 users are considered for this comparison. Figure 5 shows a comparison of the time required to update the previously generated ciphertexts. The proposed scheme integrates the revocation information separately into the generated ciphertext. Therefore, the ciphertext update module needs to modify the revocation specific components of ciphertext only. There is no need to modify the access structure-related components. Hence, the time required by the proposed scheme is much shorter compared with the existing scheme.

Algorithm 2. ROHAS-TBAC encryption

Input: $Pkey$, *RL*, message array m, and LSSS matrix *M*
Output: CT
Start

Compute $v = \{s, v_2, v_3, \ldots, v_n\}$
Compute s as, $\lambda = M.v^T$
for each message m_j **do**
    Compute $Ct_j^{'} = m_j.e(g,g)^{\alpha s_j}$
    Compute $Ct_j^{*} = g^{s_j}$
end for
for each attribute a_i **do**
    Randomly select $y_i \in \mathbb{Z}$p
    Compute $C_i = g^{\beta\lambda_i}.h_{\rho(a_i)}^{-y_i}$
    Compute $C_i^{'} = g^{y_i}$
end for
for each element $c \in$ min_cover(RL) **do**
    Compute $R_{i,c} = \left(r_c^{'}\right)^{s_i}$
end for
Compute $CT = \left\{M, Ct_j^{'}, Ct_j^{*}, C_i, C_i^{'}, R_{i,c}\right\}$
    $\forall j \in (1,n)$,
    $\forall i \in (1,l)$,
    $\forall c \in$ min_cover(RL)

End

Algorithm 3. ROHAS-TBAC key generation

Input: A, $Pkey$, $Mkey$, *ARL*, and T
Output: $Skey$
Start

Randomly select an element *b* over $\mathbb{Z}p$
Compute $Sk^{'} = g^{\alpha}g^{\beta b}$
Compute $Sk^{*} = g^{b}$
for each attribute $a_j \in A$ **do**
    Compute $S_j = h_{a_j}^{b}$
end for
for each component in T **do**
    Compute $T_i = \left(vt_i^{'}\right)^{b}$
end for
for each $k \in$ min_cover(ARL) **do**
    Compute $AR_k = \left(ar_k^{'}\right)^{b}$
end for
Compute $Skey = \left\{Sk^{'}, S_j, Sk^{*}, T_i, U_{id}, AR_k\right\}$
    $\forall j \in (1,u)$,
    $\forall i \in (1,T)$,
    $\forall k \in$ min_cover(ARL)

End

Algorithm 4. OHAS-TBAC transformation key generation

Input: $Skey : \left\{Sk^{'}, Sk^{*}, S_j, \forall j \in attribute\ set\right\}$
Output: $Tkey$ and ReK
Start

Randomly choose z over $\mathbb{Z}p$
Compute $D_1 = \left(Sk^{'}\right)^{\frac{1}{z}}$
Compute $D_2 = (Sk^{*})^{\frac{1}{z}}$
for each attribute $\forall a_j \in A$ **do**
    Compute $D_{a_j} = \left(h_{a_j}^{b}\right)^{\frac{1}{z}}$
end for
Compute $Tkey = \left\{D_1, D_2, D_{a_j}, \forall a_j \in A\right\}$
Compute $ReK = \{z, Tkey\}$

End

Algorithm 5. ROHAS-TBAC partial decryption

Input: $Tkey$, CT, ARL, IRL, and RL
Output: $CT^{''}$
Start

Reconstruct the secret s_i associated with CT_i
Set $T_{1,i} = Ct_i^{'}$
Compute $T_{2,i} = \frac{e(Ct_i^{*}, D_1)}{e(g,g)^{\frac{\beta b s_i}{z}}}$
Compute $CT_i^{''} = \{T_{1,i}, T_{2,i}\}$

End

Algorithm 6. ROHAS-TBAC update ciphertext

Input: CT and $RL^{'}$
Output: CT^{*}
Start

Randomly select σ over $\mathbb{Z}p$
for $j = 1$ to $2n-2$ **do**
    Compute $r_j^{*} = \sigma * r_k$
    Compute $r_j^{'} = g^{r_j^{*}}$
end for
for each element $c \in$ min_cover(RL') **do**
    Compute $R_c^{*} = \left(r_c^{'}\right)^{s}$
end for
Compute $CT^{*} = \left\{M, Ct^{'}, Ct^{*}, C_j, C_j^{*}, R_c^{*}, RL'\right\}$
    $\forall j \in (1,l)$,
    $\forall c \in$ min_cover(RL')

End

Algorithm 7. ROHAS-TBAC update key

Input: $Skey$ and $ARL^{'}$
Output: $Skey^{''}$
Start

1: Randomly choose γ over $\mathbb{Z}p$
2: **for** $i = 1$ to $2m - 2$ **do**
3: Compute $ar_i^* = \gamma * ar_i$
4: Compute $ar_i^{'} = g^{ar_i^*}$
5: **end for**
6: **for** each element $c \in$ min_cover(ARL') **do**
7: Compute $AR_c^* = (ar_c)^s$
8: **end for**
9: Compute $Skey^{''} = \left\{Sk^{'}, Sk^*, S_j, T_k, U_{id}, AR_c^*\right\}$,
$\forall k \in (1, T)$,
$\forall i \in (1, 2m - 2)$,
$\forall c \in$ min_cover(ARL')

End

A comparison of the encryption time needed while relying on the traditional CPABE approach and the proposed work is shown in Fig. 6. The encryption time is measured for different counts of attributes. The time required by the proposed scheme is always shorter than in the case of the existing method. In the proposed scheme, there is no need to encrypt every message separately. It is possible to apply a single hierarchical access structure for multiple messages. Hence, less time is required for encryption. For the purposes of this comparison, the number of attributes varies between 4, 8, 12, 16, and 20. Similarly to the previous comparison, total number of users considered is 30 as well, and in each iteration 50% of users are considered to be revoked users.
As the proposed scheme outsources the decryption process to a third-party server, the time required to decrypt the ciphertext is very short as compared to the traditional approach. Figure 7 shows the comparison of the decryption time required, separately for the proposed scheme, the traditional CPABE approach and the traditional CPABE with an outsourcing mechanism. The proposed method takes less time as all heavy computations are performed by a third-party outsourcing server and a hierarchical access structure is used. The traditional CPABE approach with outsourced decryption also takes less time compared to the traditional CPABE solution.

5. Conclusion

In this paper, an access control scheme for a cloud environment that facilitates encryption with a hierarchical access structure, trust computation, outsourced decryption, as well as an attribute and user revocation mechanism is presented. The hierarchical access structure concept helps reduce the burden associated with maintaining separate access structures for various data. The hierarchical access structure can be built by combining other related access structures and may be used to encrypt multiple data together. The overhead of the complex decryption process is reduced by outsourcing

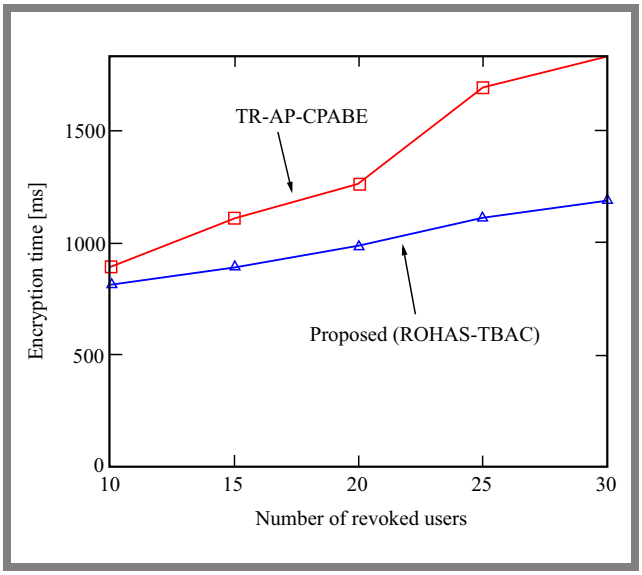

Fig. 3. Encryption time vs. number of revoked users.

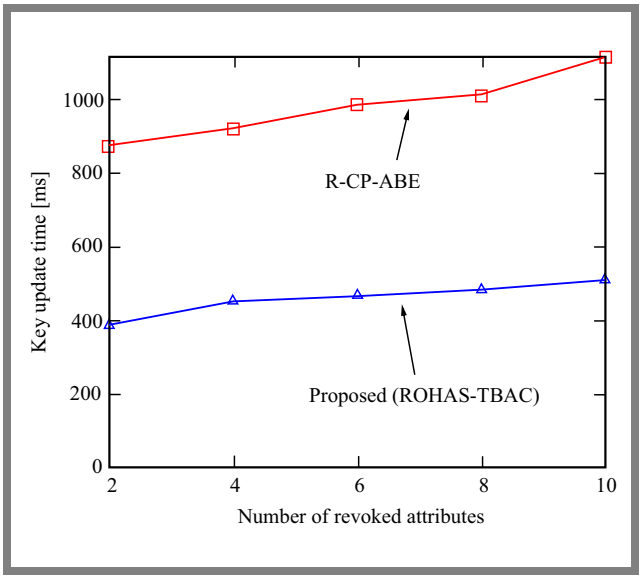

Fig. 4. Key update time vs. number of revoked attributes.

complex computations. This makes the decryption process very simple at the data user level and only small computations are required to get the original data.
The direct and immediate user revocation scheme is implemented by using a revocation list and the validity time period concept. The validity time period helps keep the revocation list shorter. Similarly, an attribute revocation list is maintained to implement the attribute revocation scheme. The proposed scheme integrates the revocation information as a separate component and, hence, the ciphertext update and key update processes take less time to complete.
The results presented in this paper show that the proposed method is more efficient than the currently know solutions in terms of the time required for encryption, decryption, ciphertext update and key update.

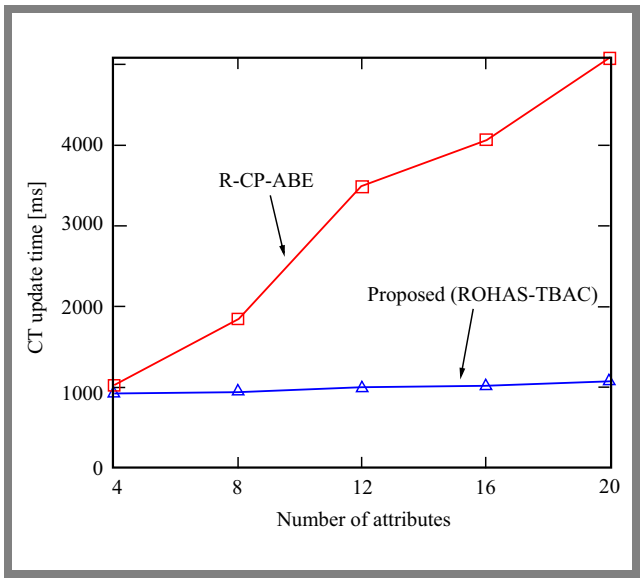

Fig. 5. CT update time vs. number of attributes.

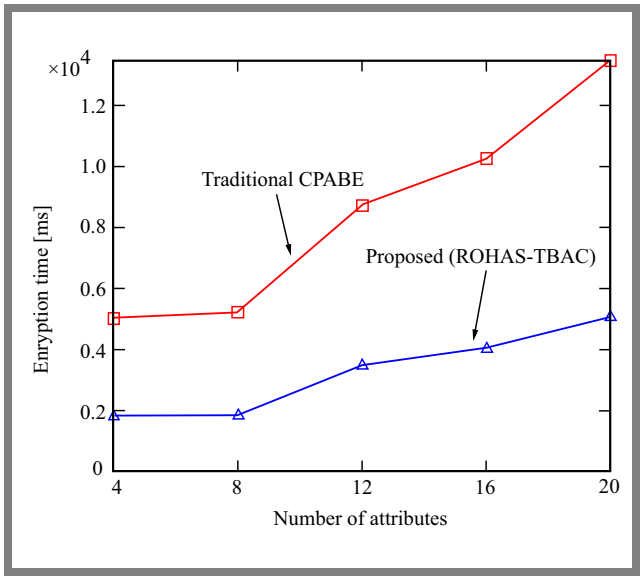

Fig. 6. Encryption time vs. number of attributes.

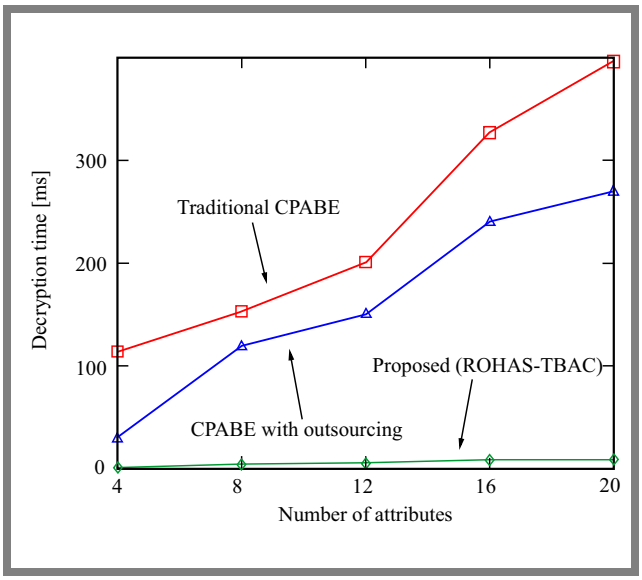

Fig. 7. Decryption time vs. number of attributes.

References

[1] A. Sahai and B. Waters, "Fuzzy Identity Based Encryption", in: *Advances in Cryptology – EUROCRYPT 2005*, vol. 3494, pp. 457–473, 2005 (https://doi.org/10.1007/11426639_27).

[2] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext Policy Attribute based Encryption", *IEEE Symposium on Security and Privacy*, Berkeley, USA, pp. 321–334, 2007 (https://doi.org/10.1109/SP.2007.11).

[3] V. Goyal V, O. Pandey, A. Sahai, and B. Waters, "Attribute Based Encryption for Fine-Grained Access Conrol of Encrypted Data", *Proceedings of the 13th ACM Conference on Computer and Communications Security*, Alexandria, USA, pp. 89–98, 2006 (https://doi.org/10.1145/1180405.1180418).

[4] Y. Yang, J. Sun, Z. Liu, and Y. Qiao, "Practical Revocable and Multi-authority CP-ABE Scheme from RLWE for Cloud Computing", *Journal of Information Security and Applications*, vol. 65, no. 6, art. no. 103108, 2022 (https://doi.org/10.1016/j.jisa.2022.103108).

[5] Z. Zhang, W. Zhang, and Z. Qin, "A Partially Hidden Policy CP-ABE Scheme Against Attribute Values Guessing Attacks with Online Privacy-protective Decryption Testing in IoT Assisted Cloud Computing", *Future Generation Computer Systems*, vol. 123, pp. 181–195, 2021 (https://doi.org/10.1016/j.future.2021.04.022).

[6] H. Zhong, Y. Zhou, Q. Zhang, Y. Xu, and J. Cui, "An Efficient and Outsourcing-supported Attribute-based Access Control Scheme for Edge-enabled Smart Healthcare", *Future Generation Computer Systems*, vol. 115, pp. 486–496, 2021 (https://doi.org/10.1016/j.future.2020.09.021).

[7] M. Mandal, "Privacy-preserving Fully Anonymous Ciphertext Policy Attribute-based Broadcast Encryption with Constant-size Secret Keys and Fast Decryption", *Journal of Information Security and Applications*, vol. 55, art. no. 102666, 2020 (https://doi.org/10.1016/j.jisa.2020.102666).

[8] S. Wang, S. Jia, and Y. Zhang, "Verifiable and Multi-keyword Searchable Attribute-based Encryption Scheme for Cloud Storage", *IEEE Access*, vol. 7, pp. 50136–50147, 2019 (https://doi.org/10.1109/ACCESS.2019.2910828).

[9] Y. He *et al.*, "An Efficient Ciphertext-policy Attribute-based Encryption Scheme Supporting Collaborative Decryption With Blockchain", *IEEE Internet of Things Journal*, vol. 9, no. 4, pp. 2722–2733, 2022 (https://doi.org/10.1109/JIOT.2021.3099171).

[10] K. Yang *et al.*, "Attribute Based Encryption with Efficient Revocation from Lattices", *International Journal of Network Security*, vol. 22, no.1, pp. 161–170, 2020 (https://doi.org/10.6633/IJNS.202001122(1).18).

[11] J. Li *et al.*, "An Efficient Attribute-based Encryption Scheme With Policy Update and File Update in Cloud Computing", *IEEE Transactions on Industrial Informatics*, vol. 15, no. 12, pp. 6500–6509, 2019 (https://doi.org/10.1109/TII.2019.2931156).

[12] S. Tu, M. Waqas, F. Huang, G. Abbas, and Z.H. Abbas, "A Revocable and Outsourced Multi-authority Attribute-based Encryption Scheme in Fog Computing", *Computer Networks*, vol. 195, art. no. 108196, 2021 (https://doi.org/10.1016/j.comnet.2021.108196).

[13] W. Wang, Z. Wang, B. Li, Q. Dong, and D. Huang, "IR-CP-ABE: Identity Revocable Ciphertext-policy Attribute-based Encryption for Flexible Secure Group-based Communication", *IACR Cryptology ePrint Archive*, vol. 1100, pp. 1–14, 2017 (https://ia.cr/2017/1100).

[14] D. Han, N. Pan, and K.-C. Li, "A Traceable and Revocable Ciphertext-policy Attribute-based Encryption Scheme Based on Privacy Protection", *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 1, pp. 316–327, 2020 (https://doi.org/10.1109/TDSC.2020.2977646).

[15] Z. Liu, F. Wang, K. Chen, and F. Tang, "A New User Revocable Ciphertext-Policy Attribute-Based Encryption with Ciphertext Update", *Security and Communication Networks*, vol. 2020, art. no. 8856592, 2020 (https://doi.org/10.1155/2020/8856592).

[16] Z. Li *et al.*, "An Efficient ABE Scheme with Verifiable Outsourced Encryption and Decryption", *IEEE Access*, vol. 7, pp. 29023–29037, 2019 (https://doi.org/10.1109/ACCESS.2018.2890565).

[17] J. Yu, G. He , X. Yan, Y. Tang, and R. Qin, "Outsourced Ciphertext-policy Attribute-based Encryption with Partial Policy Hidden", *International Journal of Distributed Sensor Networks*, vol. 16, no. 5, 2020 (https://doi.org/10.1177/1550147720926368).

[18] J. Li, Y. Zhang, X. Chen, and Y. Xiang, "Secure Attribute-based Data Sharing for Resource-limited Users in Cloud Computing", *Computers and Security*, vol. 72, pp. 1–12, 2018 (https://doi.org/10.1016/j.cose.2017.08.007).

[19] K. Fan, J. Wang, X. Wang, H. Li, and Y. Yang, "A Secure and Verifiable Outsourced Access Control Scheme in Fog-cloud Computing", *Sensors*, vol. 17, no. 7, pp. 1695–1710, 2017 (https://doi.org/10.3390/s17071695).

[20] R. Zhang, H. Ma, and Y. Lu, "Fine-grained Access Control System Based on Fully Outsourced Attribute-based Encryption", *Journal of Systems and Software*, vol. 125, no. 3, pp. 344–353, 2017 (https://doi.org/10.1016/j.jss.2016.12.018).

[21] J. Zhao, P. Zeng, and K.-K.R. Choo, "An Efficient Access Control Scheme With Outsourcing and Attribute Revocation for Fog-Enabled E-Health", *IEEE Access*, vol. 9, pp. 13789–13799, 2021 (https://doi.org/10.1109/ACCESS.2021.3052247).

[22] T.N. Mujawar and L.B. Bhajantri, "Efficient Direct and Immediate User Revocable Attribute based Encryption Scheme", in: *Proceedings of 5th International Conference on Intelligent Sustainable System (ICISS 2022)*, Tirunelveli, India, 2022 (https://link.springer.com/chapter/10.1007/978-981-19-2894-9_38).

[23] T.N. Mujawar and L.B. Bhajantri, "The Trusted Hierarchical Access Structure-Based Encryption Scheme for Cloud Computing", *International Journal of Cloud Applications and Computing*, vol. 12, no. 1, pp. 1–17, 2022 (https://doi.org/10.4018/IJCAC.308273).

[24] T.N. Mujawar and L.B. Bhajantri, "Behavior and Feedback Based Trust Computation in Cloud Environment", *Journal of King Saud University – Computer and Information Sciences*, vol. 34, no. 8, pp. 4956–4967, 2022 (https://doi.org/10.1016/j.jksuci.2020.12.003).

[25] A. Lewko and B. Waters, "Decentralizing Attribute-based Encryption", in: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Tallinn, Estonia, 2011 (https://doi.org/10.1007/978-3-642-20465-4_31).

[26] A. De Caro and V. Iovino, "jPBC: Java Pairing Based Cryptography", in: *2011 IEEE Symposium on Computers and Communications*, Kerkyra, Greece, 2011 (https://doi.org/10.1109/ISCC.2011.5983948).

Tabassum N. Mujawar, Ph.D., Assistant Professor
Department of Computer Engineering
https://orcid.org/0000-0002-8797-2894
E-mail: tabbu3002@gmail.com
Ramrao Adik Institute of Technology, D Y Patil deemed to be University, Navi Mumbai, India
https://www.dypatiledu.com

Lokesh B. Bhajantri, Ph.D., Associate Professor
Department of ISE
https://orcid.org/0000-0002-3947-4292
E-mail: lokeshcse79@gmail.com
Basaveshwar Engineering College, Bagalkote, India
https://www.becbgk.edu

Ashok V. Sutagundar, Ph.D., Associate Professor
Department of ECE
https://orcid.org/0000-0003-1494-7150
E-mail: sutagundar@gmail.com
Basaveshwar Engineering College, Bagalkote, India
https://www.becbgk.edu