International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Decentralized Firewall for Attribute-Based Encryption with Verifiable and Revocable Cloud Access Control

Shintomon Mathew[1], George T. Vadakkumcheril[2], T. Justin Jose[3]

[1]Final Year M.Tech Cyber Security, KMP College of Engineering, Pin: 683549 Perumbavoor, Kerala, India,

[2]Assistant Professor, Department of Computer Science, KMP College of Engineering, Pin: 683549 Perumbavoor, Kerala, India,

[3]Head, Department of Computer Science and Engineering, KMP College of Engineering, Pin: 683549 Perumbavoor, Kerala, India,

Abstract: ***The most difficult issues in data outsourcing are the requirement of authorization policies and the revocation of policy updates. In all current Ciphertext-Policy Attribute-Based Encryption (CP-ABE) schemes, it is expected that there is an Attribute Authority (AA) in the system to issuing attributes to the users. CP-ABE is a promising cryptographic answer for these issues for authorizing access control approaches characterized by a data owner on outsourced data. Be that as it may, it is hard to specifically apply existing CP-ABE plans to data access control for cloud storage frameworks on account of the attribute revocation issue. In numerous applications, there are various authorities exist together in a framework and every authority has the capacity to issue attributes independently. In this paper, we outline an access control structure for multi-authority frameworks and propose an efficient and secure multi-authority access control scheme for cloud storage. We first outline an efficient multi-authority CP-ABE scheme that does not oblige a global authority and can bolster any Linear Secret Sharing Scheme (LSSS) access structure. At that point, we demonstrate its security in the random oracle model. In particular, we propose a revocable multi-authority CP-ABE scheme, and apply it as the fundamental systems to outline the data access control scheme. The analysis results show that the proposed multi-authority access control scheme is scalable, efficient and secure in the data outsourcing systems.***

Keywords: Access Control, Multi-Authority, CP-ABE, Attribute Revocation, Cloud Storage, Data Sharing, Decryption Outsourcing, Verifiability, Multi-receiver Identity-based Encryption**.**

1. Introduction

CP-ABE has been a very active research area in recent years. In the construction of CP-ABE [2], each attribute is a descriptive string and each entity may be tagged with multiple attributes. Many entities may share common attributes, which allows message encryption‟s to specify a secure data access policy over the shared attributes to reach a group of receivers. A decryption‟s attributes need to satisfy the access policy in order to recover the message. These unique features make CP-ABE solutions appealing in many systems, where expressive data access control is required for a large number of users.

CP-ABE is an identity-based encryption approach. There are three roles in CP-ABE. Data owner is the role who wants to share a message with specific access control. Data user is the role who attempts to access the message. Authority is the trust third party responsible for key management. By applying CP-ABE in encryption-based access control, it is easy to provide a fine-grained access control. CP-APE has four algorithms such as setup, key generation, encryption, and decryption. The operational concept of CP-APE is shown in Fig.1.

However, the problem of applying the Attribute-Based Encryption (ABE) to the data outsourcing architecture introduces several challenges with regard to the attribute and user revocation. The revocation issue is even more difficult especially in ABE systems, since each attribute is conceivably shared by multiple users. This implies that revocation of any attribute or any single user in an attribute group would affect the other users in the group. It may result in bottleneck during rekeying procedure, or security degradation in the system.

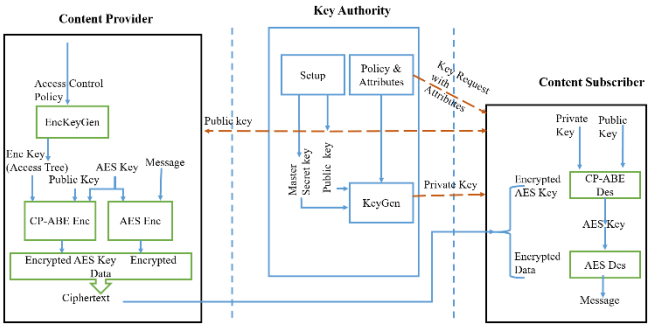

Figure 1: operational concept of CP-ABE

In this paper, we address the aforementioned privacy issue to propose a shared authority based revocable access control scheme for the cloud data storage[5], which realizes authentication and authorization without compromising a user‟s private information. The main contributions are as follows.

1) To identify a new privacy challenge in cloud storage and address a subtle privacy issue during a user challenging the cloud server for data sharing, in which the challenged request it self cannot reveal the user‟s privacy it could obtained by the access authority.
2) To propose an authentication protocol to enhance a user‟s access request related privacy, and the shared access

authority is achieved by anonymous access request matching mechanism.
3) To apply ciphertext-policy attribute based access control to realize that a user can reliably access its own data fields and adopt the proxy re-encryption to provide temp authorized data sharing among multiple users.
4) To propose an Certificate signing, that used in message sending to ensure forward and backward secrecy.
5) To apply the firewall function was initially performed by Access Control Lists (ACL).
6) To apply Encrypted fuzzy keyword search which used to maintain keyword privacy.
7) To improve cloud security assessment and audit in the data outsourcing systems.

2. Related Work

The wide adoption of cloud storage is raising several concerns about the data stored in cloud. Among which, confidentiality, integrity and access control of the data are the most significant and urgent issues [7], [8]. In many situations, when a user encrypts sensitive data, it is imperative that she establish a specific access control policy on who can decrypt this data.

Kan Yang et al. [1] first developed a revocable multi-authority CP-ABE scheme, in where an efficient and secure revocation method is proposed to solve the attribute revocation problem in the system. Ciphertext may be associated with the attribute in a previous version, while the newly joined user may be issued an attribute in a new version. Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [2]-[3] is a promising technique that is designed for access control of encrypted data. There are two types of CP-ABE systems: single authority CP-ABE [2], [3], [16], [18] where all attributes are managed by a single authority, and multi-authority CP-ABE [1] where attributes are from different domains and managed by different authorities. Multi-authority CP-ABE is more appropriate for the access control of cloud storage systems, as users may hold attributes issued by multiple authorities and the data owners may share the data using access policy defined over attributes from different authorities. However, due to the attribute revocation problem, these multi-authority CP-ABE schemes cannot be directly applied to data access control for such multi-authority cloud storage systems.

Eric Zavattoni et al. developed a Ciphertext-Policy attribute-based encryption protocol uses bilinear pairings to provide control access mechanisms, where the set of user‟s attributes is specified by means of a linear secret sharing scheme. In this paper they present the design of a software cryptographic library that achieves record timings for the computation of a 126-bit security level attribute-based encryption scheme. They developed all the required auxiliary building blocks and compared the computational weight that each of them adds to the overall performance of this protocol. In particular, their single pairing and multi-pairing implementations achieve state-of-the-art time performance at the 126-bit security level. For the confidentiality of the outsourced data, Agudo suggest several encryption schemes that can be adopted in cloud storage environment [9]. Xu et al. adopt the traditional AES encryption for their scheme and introduce an access policy on top of this encryption [10].

As to integrity, several researchers suggest to adopt a third party auditor (TPA) [11], [12], [24]. Shacham and Waters suggest a TPA leveraging the homomorphic linear authenticator to reduce the communication and computation overhead compared to the straightforward data auditing approaches [25]. Erway et al. present a definitional framework and efficient constructions for dynamic provable data possession, which supports provable updates to stored data with a low slowdown in practice [15]
.
A series of new access control schemes and solutions have been investigated and devised for cloud environment based on the general access control solutions. Due to its scalability and security, attribute-based encryption (ABE) [16] gains the most popularity in the schemes for access control. A distinguished work Fuzzy identity based encryption (IBE) [17] was introduced by Sahai and Waters in 2005. In a Fuzzy IBE scheme, a private key for an identity set v can be used to decrypt a cipher-text encrypted with a slightly different identity set v0. Fuzzy IBE realizes error tolerance by setting the threshold value of root node smaller than the size of identity set. Based on Fuzzy IBE, Goyal et al. present key policy-attribute based encryption (KP-ABE) [16] and Bethencourt et al. introduce a complementary scheme to KP-ABE, called CP-ABE [2]. There are more concrete and general CP-ABE constructions in a later paper [18]. On the other hand, Boneh and Boyen constructed BB1 and BB2 approaches [19] to build identity-based encryption. Both CP-ABE and KP-ABE can be easily adapted to cloud environment, which has gained extensive researches along this line, say[4], [20], [21], just to list a few.

Tassanaviboon and Gong propose an OAuth and ABE based authorization in semi-trusted cloud computing called AAuth [4]. Their authorization method enables an owner to-consumer encryption and supports encrypted file sharing without revealing owner‟s secret key to consumers by introducing a third party authority. Based one ABE, Yu et al. introduce a way to enable the authority to revoke user attributes with minimal effort [22] and a method to achieve secure, scalable, and fine-grained data access control in cloud computing [23].A cryptographic-based access control [20] for owner write-user-read applications is introduced by Wang et al. in 2009. Their access control system encrypts every data block of cloud storage and adopts a key derivation method to reduce the number of keys. Yu also addresses fine-grained data access control, efficient key/user management, user accountability and etc., for cloud storage in his dissertation [21]. Moreover, a novel decentralized access control with anonymous authentication is introduced by Ruj et al. [13].

Different from the existing researches, we propose FA in this paper which not only maintains the confidentiality and integrity of the data, but also provides a scalable, efficient and flexible access control by modifying the general CP-ABE to adapt to the cloud storage environment. Through the integration of fuzzy functionality into the system, we enhance the scalability and flexibility of the secure authorization.

3. Proposed System

In this paper, first proposing a revocable Multi-authority CP-ABE scheme, where an efficient and secure revocation method is proposed to solve the attribute revocation problem in the system [14]. Here attribute revocation method is efficient in the sense that it incurs less communication cost and computation cost, and is secure in the sense that it can achieve both backward security and ensure forward security. The revoked user cannot decrypt any new Ciphertext that requires the revoked attribute to decrypt. The newly joined user can also decrypt the previously published Ciphertexts, if it has sufficient attributes. The scheme does not require the server to be fully trusted, because the key update is enforced by each attribute authority not the server. Even if the server is not semi-trusted in some scenarios, the scheme can still guarantee the backward security. Then applying the proposed revocable multi-authority CP-ABE scheme as the underlying techniques to construct the expressive and secure data access control scheme for multi-authority cloud storage systems as shown in Fig 2.

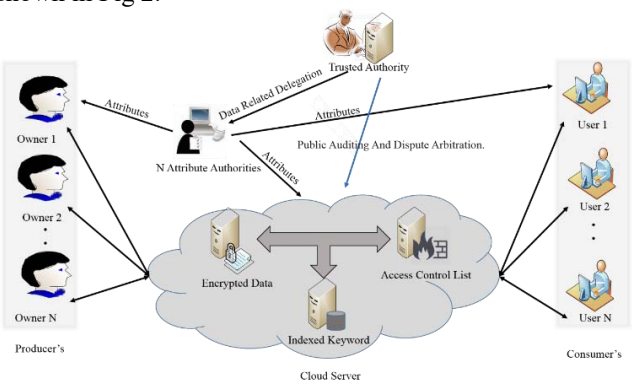

Figure 2: Modified System Architecture

Compared to the existing system proposed systems have following improvements:

- It modifies the framework of the scheme and make it more practical to cloud storage systems, in which data owners are not involved in the key generation.
- It greatly improves the efficiency of the attribute revocation method.
- It also highly improves the expressiveness of access control scheme, where remove the limitation that each attribute can only appear at most once in a Ciphertext.
- Certificate signing is used in message sending, between owner and certificate authority in order to apply for a digital identity certificate.
- The firewall function was initially performed by Access Control Lists (ACLs) in order to perform Context-based access control (CBAC) which intelligently filters TCP and UDP packets.
- It Formalize and solve the problem of effective fuzzy keyword search over encrypted cloud data while maintaining keyword privacy.

3.1 System Model

They consider a data access control system in multi-authority cloud storage, as described in Fig. 3. There are five types of entities in the system: a certificate authority (CA), attribute authorities (AAs), data owners (owners), the cloud server (server) and data consumers (users).

The CA is a global trusted certificate authority in the system. It sets up the system and accepts the registration of all the users and AAs in the system. For each legal user in the system, the CA assigns a global unique user identity to it and also generates a global public key for this user. However, the CA is not involved in any attribute management and the creation of secret keys that are associated with attributes. For example, the CA can be the Social Security Administration, an independent agency of the United States government. Each user will be issued a Social Security Number (SSN) as its global identity.

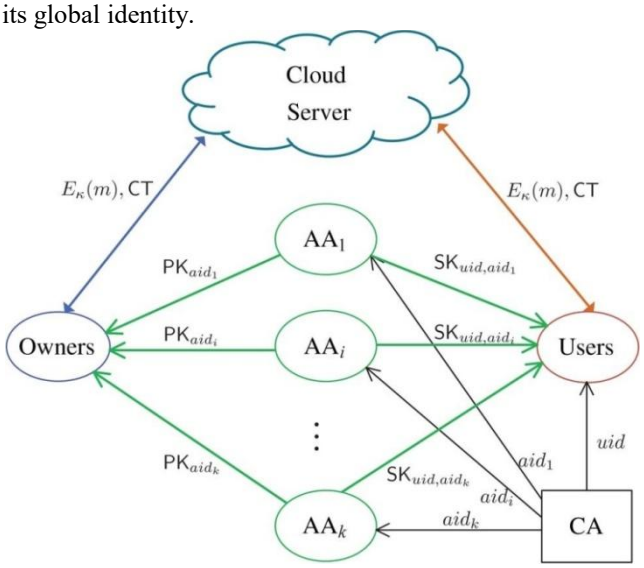

Figure 3: System model of data access control in multi-authority cloud storage.

Every AA is an independent attribute authority that is responsible for entitling and revoking user‟s attributes according to their role or identity in its domain. In their scheme, every attribute is associated with a single AA, but each AA can manage an arbitrary number of attributes. Every AA has full control over the structure and semantics of its attributes. Each AA is responsible for generating a public attribute key for each attribute it manages and a secret key for each user reflecting his/her attributes.

The cloud stores the data owners‟ shared files and provides access service to the users. The data owners define access control policies and under which encrypt their data files before outsourcing them to the cloud. The attribute authorities are responsible for issuing secret keys to the users according to their valid attributes, and they are also in charge of revoking and updating users‟ attribute keys within the authority‟s domains or organizations. The users can request their private keys from the relevant authorities. After downloading any encrypted data file shared on the cloud, only the users whose private keys satisfy the access control policy can decrypt it.

Each user has a global identity in the system. A user may be entitled a set of attributes which may come from multiple attribute authorities. The user will receive a secret key associated with its attributes entitled by the corresponding attribute authorities. Each owner first divides the data into

several components according to the logic granularities and encrypts each data component with different content keys by using symmetric encryption techniques. Then, the owner defines the access policies over attributes from multiple attribute authorities and encrypts the content keys under the policies. Then, the owner sends the encrypted data to the cloud server together with the Ciphertexts. In their paper, they simply use the Ciphertext to denote the encrypted content keys with CP-ABE.

They do not rely on the server to do data access control. But, the access control happens inside the cryptography. That is only when the user"s attributes satisfy the access policy defined in the Ciphertext, the user is able to decrypt the Ciphertext. Thus, users with different attributes can decrypt different number of content keys and thus obtain different granularities of information from the same data.

3.2 Framework

The framework of their data access control scheme is defined as follows.

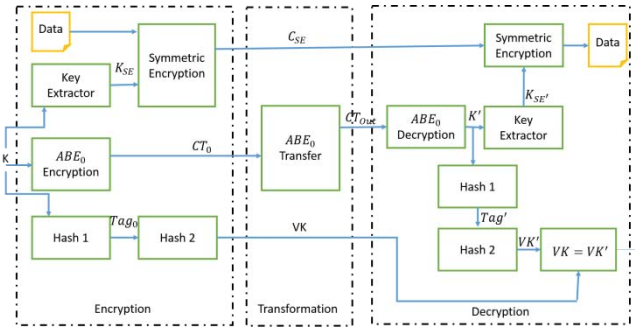

Figure 4: The framework of Verifiable and Revocable Cloud Access Control

Definition (Framework of Multi-Authority Access Control Scheme). The framework of data access control scheme for multi-authority cloud storage systems contains the following phases:

Phase 1: System Initialization. This phase consists of CA setup and AA setup with the following algorithms:

- $CASetup(1^{\lambda}) \rightarrow (GMK, GPP, (GPK_{uid}, GPK'_{uid}), (GSK_{uid}, GSK'_{uid}), Certificate(uid)\})$ The CA setup algorithm is run by the CA. It takes no input other than the implicit security parameter λ. It generates the global master key GMK of the system and the global public parameters GPP. For each user uid, it generates the user"s global public keys (GPK_{uid}, GPK'_{uid}), the user"s global secret keys (GSK_{uid}, GSK'_{uid}), and a certificate $Certificate(uid)$ of the user.
- $AASetup(u_{aid}) \rightarrow (SK_{aid}, PK_{aid}, \{VK_{x_{aid}}, PK_{x_{aid}}\}_{x_{aid} \in u_{aid}})$. The attribute authority setup algorithm is run by each attribute authority. It takes the attribute universe u_{aid} managed by the AA_{aid} as input. It outputs a secret and public key pair (SK_{aid}, PK_{aid}) of the AA_{aid} and a set of version keys and public attribute keys $\{VK_{x_{aid}}, PK_{x_{aid}}\}_{x_{aid} \in u_{aid}}$ for all the attributes managed by the AA_{aid}.

Phase 2: Secret Key Generation by AAs.

- $SKeyGen\left(\begin{matrix} GPP, GPK_{uid}, GPK'_{uid}, GSK_{uid}, SK_{uid}, \\ S_{uid,aid}\{VK_{x_{aid}}, PK_{x_{aid}}\}_{x_{aid} \in S_{uid,aid}} \end{matrix}\right) \rightarrow SK_{uid,aid}$. The secret key generation algorithm is run by each AA. It takes as inputs the global public parameters GPP, the global public keys (GPK_{uid}, GPK'_{uid}) and one global secret key GSK_{uid} of the user uid, the secret key SK_{uid} of the AA_{aid}, a set of attributes $S_{uid,aid}$ that describes the user uid from the AA_{aid} and its corresponding version keys $\{VK_{x_{aid}}\}$ and public attribute keys $\{PK_{x_{aid}}\}$. It outputs a secret key $SK_{uid,aid}$ for the user uid which is used for decryption.

Phase 3: Data Encryption by Owners. Owners first encrypt the data m with content keys by using symmetric encryption methods, then they encrypt the content keys by running the following encryption algorithm:

- Encrypt(GPP, $\{PK_{aid_k}\}_{aid_k \in I_A}, k, A) \rightarrow$ CT. The encryption algorithm is run by the data owner to encrypt the content keys. It takes as inputs the global public parameters GPP, a set of public keys $\{PK_{aid_k}\}_{aid_k \in I_A}$ for all the AAs in the encryption set I_A[3], the content key and an access policy A.[4] The algorithm encrypts according to the access policy and outputs a Ciphertext CT. They will assume that the Ciphertext implicitly contains the access policy A.

Phase 4: Data Decryption by Users. Users first run the decryption algorithm to get the content keys, and use them to further decrypt the data.

- Decrypt$(CT, GPK_{uid}, GSK'_{uid}, \{SK_{uid,aid_k}\}_{aid_k \in I_A}) \rightarrow$ k. The decryption algorithm is run by users to decrypt the Ciphertext. It takes as inputs the Ciphertext CT which contains an access policy A, a global public key GPK_{uid} and a global secret key , GSK'_{uid} of the user uid, and a set of secret keys $\{SK_{uid,aid_k}\}_{aid_k \in I_A}$ from all the involved AAs. If the attributes $\{S_{uid,aid_k}\}_{aid_k \in I_A}$ of the user uid satisfy the access policy A, the algorithm will decrypt the Ciphertext and return the content key.

Phase 5: Attribute Revocation. This phase contains three steps: Update Key Generation by AAs, Secret Key Update by Non-revoked Users[5] and Ciphertext Update by Server.

$$UKeyGen(SK_{aid'}, \tilde{x}_{aid'}, VK_{\tilde{x}_{aid'}}) \rightarrow (\widetilde{VK}_{\tilde{x}_{aid'}}, \widetilde{UK}_{s,\tilde{x}_{aid'}}, UK_{c,\tilde{x}_{aid'}}).$$

- The update key generation algorithm is run by the corresponding $AA_{aid'}$, that manages the revoked attribute $\tilde{x}_{aid'}$. It takes as inputs the secret key $SK_{aid'}$ of $AA_{aid'}$, the revoked attribute $\tilde{x}_{aid'}$ and its current version key $VK_{\tilde{x}_{aid'}}$. It outputs a new version key $\widetilde{VK}_{\tilde{x}_{aid'}}$ and the update key $\widetilde{UK}_{s,\tilde{x}_{aid'}}$ (for secret key update) and the update key $UK_{c,\tilde{x}_{aid'}}$ (for Ciphertext update).
- $SKUpdate(SK_{uid,aid'}, UK_{s,\tilde{x}_{aid'}}) \rightarrow \widetilde{SK}_{uid,aid'}$ .The secret key update algorithm is run by each non-revoked user uid. It takes as inputs the current secret key of the non-revoked user $SK_{uid,aid'}$ and the update

key$UK_{s,\tilde{x}_{aid'}}$. It outputs a new secret key $\widetilde{SK}_{uid,aid'}$ for each non-revoked user uid.

Note that not all the AAs are involved in the encryption. They use encryption set I_A to denote the set of those AAs involved in the encryption. The access policy is a LSSS structure (M^*, ρ^*), which is defined in the supplemental file available online. They denote those users who possess the revoked attributes $\tilde{x}_{aid'}$ but have not be revoked as the non-revoked users.

$CTUpdate(CT, UK_{c,\tilde{x}_{aid'}}) \rightarrow \widetilde{CT}$. The Ciphertext update algorithm is run by the cloud server. It takes as inputs the Ciphertext which contain the revoked attribute$\tilde{x}_{aid'}$, and the update key $UK_{c,\tilde{x}_{aid'}}$. It outputs new Ciphertexts $\widetilde{CT}$ which contain the latest version of the revoked attribute$\tilde{x}_{aid'}$.

3.3 Security model

In multi-authority cloud storage systems, they make the following assumptions:

- The CA is fully trusted in the system. It will not collude with any user, but it should be prevented from decrypting any Ciphertexts by itself.
- Each AA is trusted but can be corrupted by the adversary.
- The server is curious but honest. It is curious about the content of the encrypted data or the received message, but will execute correctly the task assigned by each attribute authority.
- Each user is dishonest and may collude to obtain unauthorized access to data.

3.3.1 Data Access Control Scheme

In this section, they first give an overview of the challenges and techniques. Then, they propose the detailed construction of their access control scheme which consists of five phases: System Initialization, Key Generation, Data Encryption, Data Decryption and Attribute Revocation.

a) Overview

To design the data access control scheme for Multi-authority cloud storage systems, the main challenging issue is to construct the underlying Revocable Multi-authority CP-ABE protocol. In [6], Chase proposed a multi-authority CP-ABE protocol, however, it cannot be directly applied as the underlying techniques because of two main reasons: 1) Security Issue: Chase"s multi-authority CP-ABE protocol allows the central authority to decrypt all the Ciphertexts, since it holds the master key of the system; 2) Revocation Issue: Chase"s protocol does not support attribute revocation. They propose a new revocable multi-authority CP-ABE protocol based on the single-authority CP-ABE proposed by *Lewko* and Waters in [16]. That is they extend it to Multi-authority scenario and make it revocable.

They apply the techniques in Chase"s multi-authority CP-ABE protocol [6] to tie together the secret keys generated by different authorities for the same user and prevent the collusion attack. Specifically, they separate the functionality of the authority into a global certificate authority (CA) and multiple attribute authorities (AAs). The CA sets up the system and accepts the registration of users and AAs in the system.

It assigns a global user identity $uid$to each user and a global authority identity aid to each attribute authority in the system. Because theuid is globally unique in the system, secret keys issued by different AAs for the sameuid can be tied together for decryption. Also, because each AA is associated with an aid, every attribute is distinguishable even though some AAs may issue the same attribute.

To deal with the security issue in [6], instead of using the system unique public key (generated by the unique master key) to encrypt data, their scheme requires all attribute authorities to generate their own public keys and uses them to encrypt data together with the global public parameters. This prevents the certificate authority in their scheme from decrypting the Ciphertexts. To solve the attribute revocation problem, they assign a version number for each attribute. When an attribute revocation happens, only those components associated with the revoked attribute in secret keys and Ciphertexts need to be updated.

When an attribute of a user is revoked from its corresponding AA, the AA generates a new version key for this revoked attribute and generates an update key. With the update key, all the users, except the revoked user, who hold the revoked attributes can update its secret key (Backward Security). By using the update key, the components associated with the revoked attribute in the Ciphertext can also be updated to the current version.

To improve the efficiency, they delegate the workload of Ciphertext update to the server by using the proxy re-encryption method, such that the newly joined user is also able to decrypt the previously published data, which are encrypted with the previous public keys, if they have sufficient attributes (Forward Security). Moreover, by updating the Ciphertexts, all the users need to hold only the latest secret key, rather than to keep records on all the previous secret keys. Note that the CA is not involved in any attribute management and any secret key generation.

b) Secret Key Generation

Each user $uid$is required to authenticate itself to the AA_{aid} before it can be entitled some attributes from theAA_{aid}. The user submits its certificate $Certificate(uid)$to theAA_{aid}. The AA_{aid} then authenticates the user by using the verification key issued by the CA.

If it is a legal user, the AA_{aid} entitles a set of attributes $S_{uid,aid}$ to the user uidaccording to its role or identity in its administration domain. Otherwise, it aborts. Then, the AA_{aid} generates the user"s secret key $SK_{uid,aid}$ by running the secret key generation algorithm $SKeyGen$. It chooses a random number $t_{uid,aid} \in Z_p$and computes the user"s secret key as

$$\mathsf{SK}_{uid,aid} = \Big(K_{uid,aid} = g^{\alpha_{aid}} g^{au_{uid}} g^{bt_{uid,aid}}, K'_{uid,aid} = g^{t_{uid,aid}},$$
$$\forall x_{aid} \in S_{uid,aid} : K_{x_{aid},uid} = g^{u'_{uid} t_{uid,aid} \beta_{aid}} H(x_{aid})^{v_{x_{aid}} \beta_{aid}(u_{uid}+\gamma_{aid})} \Big).$$

If the user uiddoes not hold any attribute fromAA_{aid}, the secret key $SK_{uid,aid}$ only contains the first component $K_{uid,aid}$.

c) Data Encryption

Before hosting data m to cloud servers, the owner processes the data as follows.

1. It divides the data into several data components as $m = \{m_1, \dots, m_n\}$according to the logic granularities. For example, the personal data may be divided into {name, address, security number, employer, salary}.
2. It encrypts data components with different content keys $\{k_1, \dots, k_n\}$ by using symmetric encryption methods.
3. It then defines an access structure $M_i$for each content key $k_i (i = 1, \dots, n)$and encrypts it by running the encryption algorithm Encrypt.

The encryption algorithm Encrypt takes as inputs the global public parameters GPP, a set of public keys $\{PK_{aid_k}\}_{aid_k \in I_A}$for all the AAs in the encryption set I_A, the content key k and an access structure (M^*, ρ^*)over all the involved attributes. Let Mbe a $l \times n$matrix, where l denotes the total number of all the attributes. The function maps each row of M to an attribute. In this construction, they remove the limitation that ρshould be an injective function (i.e., an attribute can associate with more than one rows of M).

To encrypt the content key k, the encryption algorithm first chooses a random encryption exponent $s \in Z_p$and chooses a random vector$\vec{v} = (s, y_2, \dots, y_n) \in z_p^n$, where $y_2, \dots, y_n$are used to share the encryption exponent s. For $i = 1$ to l, it computes $\lambda_i = \vec{v}.M_i$, where $M_i$is the vector corresponding to the $i-th$ row of M. Then, it randomly chooses $r_1, r_2, \dots, r_l \in Z_p$and computes the Ciphertext as

$$\mathsf{CT} = \left(C = \kappa \cdot \left(\prod_{aid_k \in I_A} \mathsf{PK}_{aid_k} \right)^s, C' = g^s, C'' = g^{bs}, . \right.$$
$$\forall 1 \le i \le \ell, \rho(i) \in S_{aid_k} :$$
$$C_i = g^{a\lambda_i} \cdot \left(\mathsf{PK}_{1,\rho(i)}\right)^{-r_i}, C_i' = g^{r_i},$$
$$\left. D_i = g^{\frac{r_i}{\beta_{aid_k}}}, D_i' = \left(\mathsf{PK}_{2,\rho(i)}\right)^{r_i} \right).$$

After that, the owner sends the data to the server in the format as described below.

CT_1	$E_{\kappa_1}(m_1)$	⋯	⋯	CT_n	$E_{\kappa_n}(m_n)$

d) Data Decryption

All the legal users in the system can freely query any interested encrypted data. Upon receiving the data from the server, the user runs the decryption algorithm Decrypt to decrypt the Ciphertext by using its secret keys from different AAs. Only the attributes the user possesses satisfy the access structure defined in the Ciphertext CT, the user can get the content key.

The decryption algorithm
$Decrypt(CT, GPK_{uid}, GSk'_{uid}, \{Sk_{uid,aid_k}\}_{aid_k \in I_A}) \rightarrow k$can be constructed as follows. It takes as inputs the Ciphertext CT which contains an access policy(M^*, ρ^*), a global public key $GPK_{uid}$and a global secret key $GSk'_{uid}$of the useruid, and a set of secret keys $\{Sk_{uid,aid_k}\}_{aid_k \in I_A}$from all the involved AAs. If the user"s attributes can satisfy the access structure, then the user uidproceeds as follows.

Let Ibe $\{I_{aid_k}\}_{aid_k \in I_{A'}}$, where $I_{aid_k} \subset \{1, 2, \dots, l\}$is defined as$I_{aid_k} = \{i : \rho(i) \epsilon S_{aid_k}$. Let $n_A = |I_A|$be the number of AAs involved in the Ciphertext. Then, it chooses a set of constants $\{w_i \in Z_p\}_{i \in I}$and reconstructs the encryption exponent as $s = \sum_{i \in I} w_i \lambda_i$ if λ_i are valid shares of the secret s according to M. The decryption algorithm first computes

$$\prod_{aid_k \in I_A} e(C', K_{uid,aid_k}) e\left(C'', K'_{uid,aid_k}\right)^{-1}$$
$$= e(g,g)^{au_{uid} n_A s} \cdot \prod_{aid_k \in I_A} e(g,g)^{s\alpha_{aid}}.$$

For each $i \in I$, suppose $\rho(i) \in S_{aid_k'}$, it computes

$$e(C_i, \mathsf{GPK}_{uid})\, e(D_i, K_{\rho(i),uid})\, e\left(C_i', K'^{-\mathsf{GSK}'_{uid}}_{uid,aid_k}\right) e(g, D_i')^{-1}$$
$$= e(g,g)^{au_{uid}\lambda_i}.$$

Then, it computes

$$\prod_{aid_k \in I_A} \prod_{i \in I_{aid_k}} \left(e(g,g)^{au_{uid}\lambda_i} \right)^{w_i n_A} = e(g,g)^{au_{uid} n_A s}.$$

Thus, the user can obtain $\prod_{k \in I_A} e(g,g)^{\propto_{aid_k} s}$ and use it to decrypt the Ciphertext as

$$\kappa = C / \prod_{aid_k \in I_A} e(g,g)^{\alpha_{aid_k} s}.$$

Then, the user can use the decrypted content key kto further decrypt the encrypted data component.

e) Attribute Revocation

As They described before, there are two requirements of the attribute revocation: 1) The revoked user (whose attribute is revoked) cannot decrypt new Ciphertexts encrypted with new public attribute keys (Backward Security); 2) the newly joined user who has sufficient attributes should also be able to decrypt the previously published Ciphertexts, which are encrypted with previous public attribute keys (Forward Security). For example, in a university, some archive documents are encrypted under the policy „„CS Dept. AND (Professor OR PhD Student)"", which means that only the professors or PhD students in CS department are able to decrypt these documents. When a new professor/PhD student joins the CS department of the university, he/she should also be able to decrypt these documents. Their attribute revocation methods can achieve both forward security and backward security.

Suppose an attribute $\tilde{x}_{aid'}$ is revoked from the user $uid'$by the $AA_{aid'}$. The attribute$\tilde{x}_{aid'}$ is denoted as the Revoked

Attribute and the user uid' is denoted as the Revoked User. They also use the term of Non-revoked Users to denote the set of users who possess the revoked attribute $\tilde{x}_{aid'}$ but have not been revoked.

3.3.2 Ciphertext Update by Cloud Server

To ensure that the newly joined user who has sufficient attributes can still decrypt those previous data which are published before it joined the system (Forward Security), all the Ciphertexts associated with the revoked attribute are required to be updated to the latest version. Intuitively, the Ciphertext update should be done by data owners, which will incur a heavy overhead on the data owner. To improve the efficiency, we move the workload of Ciphertext update from data owners to the cloud server, such that it can eliminate the huge communication overhead between data owners and cloud server, and the heavy computation cost on data owners. The Ciphertext update is conducted by using proxy re-encryption method, which means that the server does not need to decrypt the Ciphertext before updating.

4. Result Analysis

4.1 Security Analysis

During the secret key update phase, the corresponding AA generates an update key for each non-revoked user. Because the update key is associated with the user"s global identity uid, the revoked user cannot use update keys of other non-revoked users to update its own secret key, even if it can compromise some non-revoked users. After each attribute revocation operation, the version of the revoked attribute will be updated. When new users join the system, their secret keys are associated with attributes with the latest version. However, previously published Ciphertext are encrypted under attributes with old version. Although the CA holds the global master key GMK, it does not have any secret key issued from the AA. Without the knowledge of$g^{\propto_{aid}}$, the CA cannot decrypt any Ciphertext in the system. Table 1 shows comparison of security.

Table 1: The Comparison of Security

Schemes	KP/CP-ABE	Support of Access Structure	Central Authority	Collaborative Computation	User Revocation	Encrypted Search	Policy Firewall
Chase	KP-ABE	AND	YES	YES	NO	NO	NO
MKE	CP-ABE	LSS	YES	NO	NO	NO	NO
CC	KP-ABE	AND	NO	YES	NO	NO	NO
LCLS	KP-ABE	LSS	NO	YES	NO	NO	NO
LW	CP-ABE	LSS	NO	NO	NO	NO	NO
LCHWY	CP-ABE	LSS	Multiple	NO	NO	NO	NO
EER DAC MACS [1]	CP-ABE	LSS	NO	NO	NO	NO	NO
Our Scheme	CP-ABE	LSS / AND	Multiple	YES	YES	YES	YES

4.2 Performance Analysis

In this section, we analyses the performance of our scheme by comparing with the Kan jang"s CP-ABE scheme [1] and our previous scheme in the conference version [3]], in terms of storage overhead, communication cost and computation efficiency.

4.2.1 Storage Overhead

The storage overhead is one of the most significant issues of the access control scheme in cloud storage systems. AA Each AA needs store the information of all the attributes in its domain. The public parameters contribute the main storage overhead on the owner. The storage overhead on each user in our scheme comes from the secret keys issued by all the AAs. The Ciphertext contribute the main storage overhead on the server rather than considering the encrypted data which are encrypted by the symmetric content keys. Table 2 shows storage overhead on each entity.

n_c: The total number of Ciphertexts stored on the Cloud.
$n_{c,x}$: The number of Ciphertexts contain the revoked attribute x.

Table 2: Storage Overhead on Each Entity

Entity	[13]	[14]	[1]	Our
$AA_{aid}(\|p\|)$	$2n_{aid}$	$n_{aid} + 2n_O + 1$	$n_{aid} + 2n_U + 3$	$n_{aid} + 2n_U + 3$
Owner ($\|p\|$)	$n_c + 2n_a$	$n_c + n_A + n_a + 2$	$3n_A + n_a + 3$	$2n_A + n_a + 1$
User ($\|p\|$)	$n_{c,x} + n_{a,uid}$	$n_O(n_A + n_{a,uid})$	$2n_A + n_{a,uid}$	$2n_A + n_{a,uid}$
Server ($\|p\|$)	$3l + 1$	$L + 2$	$4l + 2$	$3l + 2$

4.2.2 Communication Cost

The communication cost of the normal access control is almost the same. Here, we only compare the communication cost of attribute revocation. The communication cost of attribute revocation in [1] [3] is linear to the number of Ciphertext which contain the revoked attribute. Table 3 shows communication cost for attribute revocation.

Table 3: Communication Cost for Attribute Revocation

Operation	[13]	[14]	[1]	Our
Key Update	None	$n_{non,x}\|p\|$	$n_{non,x}\|p\|$	$n_{c,aid}\|p\|$
CT Update	$n_{c,x} \times n_{non,x} + 1\|p\|$	$n_{c,aid}\|p\|$	$2\|P\|$	$2\|P\|$

$n_{non,x}$: Number of non-revoked user hold x. $n_{c,x}$:Number of Ciphertext contains x. $n_{c,aid}$: Number of attributes from the AA_{aid} in all Ciphertexts.

4.3 Computation Efficiency

We implement our scheme and Kan Yang"s scheme [1] on a Linux system with an Intel Core 2 Duo CPU at 3.16GHz and 4.00 GB RAM. The code uses the Pairing-Based Cryptography (PBC) library version 1.2 to implement the access control schemes. We use a symmetric elliptic curve α-curve, where the base field size is 512-bit and the embedding degree is 2. The α-curve has a 160-bit group order, which means p is a 160-bit length prime. All the simulation results are the mean of 11 trials.

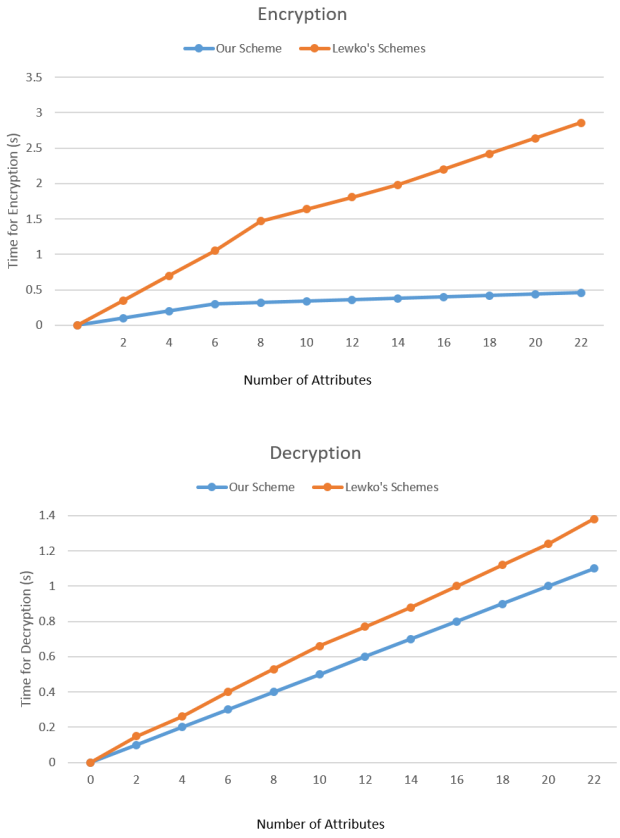

Figure 5: Comparison of Time Consumption with Different Number of Authorities

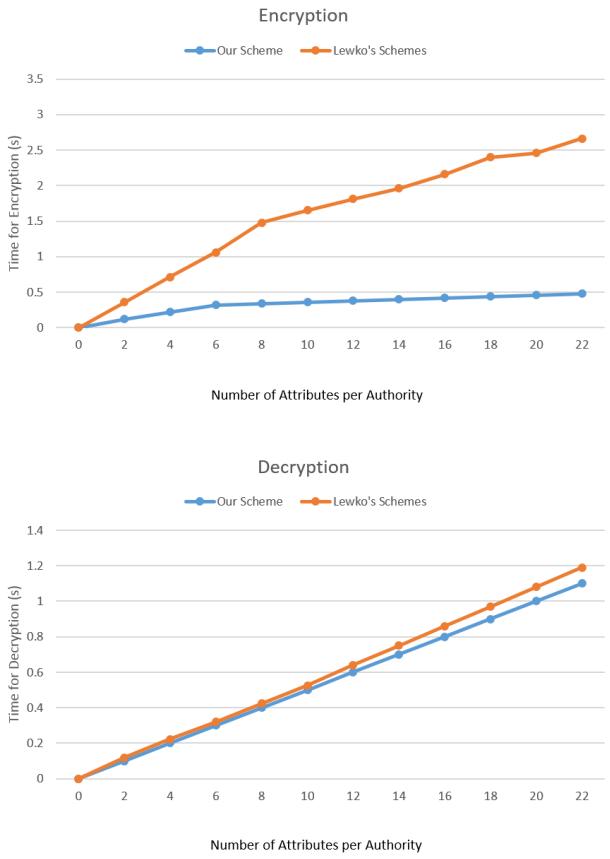

Figure 6: Comparison of Time Consumption with Different Number of Attributes per Authority

5. Conclusion and Future Work

This paper presenting a novel decentralized CP-ABE access control scheme for cloud Storage systems, which is both efficient and secure. This Work makes an effort towards comparing some mechanisms and effective tools that can be implemented for the Ciphertext-Policy Attribute-based Encryption. In this work, we have identified a new privacy challenge during data accessing in the cloud computing to achieve privacy-preserving access authority sharing. In this paper proposing a revocable multi-authority CP-ABE scheme that can support efficient attribute revocation. Then constructing an effective data access control scheme for multi-authority cloud storage systems. Authentication is established to guarantee data confidentiality and data integrity. The revocable multi-authority CP-ABE is a promising technique, which can be applied in any remote storage systems and online social networks etc. Our scheme does not require any central authority and coordination among multiple authorities, thus eliminating the burden of heavy communication and the delay of collaborative computation.

References

[1] Kan Yang and Xiaohua "Expressive, Efficient, and Revocable Data Access Control for Multi-Authority Cloud Storage," IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 7, july 2014.

[2] J. Bethencourt, A. Sahai, and B. Waters, „„Ciphertext-Policy Attribute-Based Encryption, ‟‟ in Proc. IEEE Symp. Security and privacy (S&P‟07), 2007, pp. 321-334.
[3] Waters, „„Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization, ‟‟ in Proc. 4th Int‟l Conf. Practice and Theory in Public Key Cryptography (PKC‟11), 2011, pp. 53-70.
[4] A. Tassanaviboon and G. Gong, "OAuth and ABE based authorization in semi-trusted cloud computing, " in Proc. 2nd Int. Workshop Data Intensive Comput. Clouds, 2011, pp. 41–50.
[5] P. Mell and T. Grance, „„The NIST Definition of Cloud Computing, ‟‟ National Institute of Standards and Technology, Gaithersburg, MD, USA, Tech. Rep., 2009.
[6] Eric Zavattoni, Luis J. Dominguez Perez et al. "Software implementation of an Attribute-Based Encryption scheme" submitted to IEE journal on June 2, 2014
[7] K. Ren, C. Wang, and Q. Wang, "Security challenges for the public cloud, " IEEE Internet Comput., vol. 16, no. 1, pp. 69–73, Jan./Feb. 2012.
[8] P. Samarati, and S. D. C. di Vimercati, "Data protection in outsourcing scenarios: Issues and directions, " in Proc. 5th ACM Symp.Inf., Comput. Commun. Security, 2010, pp. 1–14.
[9] I. Agudo, "Cryptography goes to the cloud, " in Proc. Workshop Secure Trust Comput., Data Manage. Appl., 2011, pp. 190–197.
[10] J. Xu, E.-C. Chang, and J. Zhou, "Weak leakage-resilient clientside deduplication of encrypted data in cloud storage, " in Proc. 8th ACM SIGSAC Symp. Inf., Comput. Commun. Security, 2013, pp. 195–206.
[11] C. Wang, S. S. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for secure cloud storage, " IEEE Trans. Comput., vol. 62, no. 2, pp. 362–375, Feb. 2013.
[12] Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn, H. Hu, and S. S. Yau, "Dynamic audit services for outsourced storages in clouds, " IEEE Trans. Serv. Comput., vol. 6, no. 2, pp. 227–238, Apr.–Jun. 2013.
[13] S. Ruj, M. Stojmenovic, and A. Nayak, "Decentralized access control with anonymous authentication of data stored in clouds, " IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 2, pp. 384–394, Feb. 2014.
[14] K. Yang and X. Jia, „„Attribute-Based Access Control for Multi-Authority Systems in Cloud Storage, ‟‟ in Proc. 32th IEEE Int‟l Conf. Distributed Computing Systems (ICDCS‟12), 2012, pp. 1-10.
[15] C. C. Erway, A. Kp, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession, " in Proc. 16th ACM Conf. Comput. Commun. Security, 2009, pp. 213–222.
[16] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data, " in Proc. 13th ACM Conf. Comput. Commun. Security, 2006, pp. 89–98.
[17] A. Sahai and B. Waters, "Fuzzy identity-based encryption, " in Proc. 24th Annu. Int. Conf. Theory Appl. Cryptographic Techn.,2005, vol. 3494, pp. 457–473.
[18] A.B. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, „„Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption, ‟‟ in Proc. Advances in Cryptology-EUROCRYPT‟10, 2010, pp. 62-91.
[19] D. Boneh and X. Boyen, "Efficient selective-ID secure identity based encryption without random oracles, " in Proc. Adv. Cryptology Int. Conf. Theory Appl. Cryptographic Techn., 2004, vol. 3027, pp. 223–238.
[20] W. Wang, Z. Li, R. Owens, and B. Bhargava, "Secure and efficient access to outsourced data, " in Proc. ACM Workshop Cloud Comput. Security, 2009, pp. 55–65.
[21] S. Yu, "Data sharing on untrusted storage with attribute-based encryption, " Ph.D. dissertation, Dept. Elect. Comput. Eng., Worcester Polytechnic Inst., Worcester, MA, USA, Jul. 2010.
[22] S. Yu, C. Wang, K. Ren, and W. Lou, "Attribute based data sharing with attribute revocation, " in Proc. 5th ACM Symp. Inf., Comput. Commun. Security, 2010, pp. 261–270.
[23] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing, " in Proc. 29th Conf. Inf. Commun., 2010, pp. 534–542.
[24] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing, " IEEE Trans. Parallel Distrib. Syst., vol. 22, no. 5, pp. 847–859, May 2011.
[25] H. Shacham, and B. Waters, "Compact proofs of retrievability, " in Proc. 14th Int. Conf. Theory Appl. Cryptology Inf. Security: Adv. Cryptol., 2008, pp. 90–107.

Author Profile

Shintomon Mathew received the B.Tech degree in Information Technology from University Of Calicut in 2011 and currently pursuing final year M. Tech degree in Computer Science and Engineering with specialization in Cyber Security from KMP College of Engineering, Perumbavoor.

George T. Vadakkumcherilreceived degree of M.Tech Computer Science & Engineering from Karunya University, Coimbatore in 2014 and B.Tech in Information Technology from Mahatma Gandhi University, Kerala in 2011. He is working as Assistant Professor in Computer Science & Engineering Department under Mahatma Gandhi University, Kerala.

T.Justin Jose received his B.E and M.Tech degrees in Computer Science and Engineering from ManonmaniumSundaranar University, Tirunelveli in the year 1998 and 2004 respectively. He is currently working as a Professor in the Computer Science and Engineering Department of KMP College of Engineering, Kothamangalam, Kerala, India. He is currently working in the area of image processing and Genetic Algorithms. He is a reviewer of few Indian and International journals. He has authored and co-authored about 15 technical journal papers and conferences.