Fully Secure and Efficient Data Sharing with Attribute Revocation for Multi-Owner Cloud Storage

K.Kokilavani, K.S. Saravanan

M.Phil Full Time Research Scholar, Dept. of Computer Science
Assistant Professor, Dept. of Computer Sc. and Application
Vivekananda College of Arts and Sciences for Women, Namakkal, TamilNadu, India

Abstract
Now a days, a lot of users are storing their data’s in cloud, because it provides storage flexibility. But the main problem in cloud is data security. Data access control is an effective way to ensure the data security in the cloud. Due to data outsourcing and untrusted cloud servers, the data access control becomes a challenging issue in cloud storage systems. Ciphertext-Policy Attribute based Encryption (CP-ABE) is regarded as one of the most suitable technologies for data access control in cloud storage, because it gives data owners more direct control on access policies. However, it is difficult to directly apply existing CP-ABE schemes to data access control for cloud storage systems because of the attribute revocation problem. In this paper, we design an expressive, efficient and revocable data access control scheme for multi-authority cloud storage systems, where there are multiple authorities co-exist and each authority is able to issue attributes independently. Specifically, we propose a revocable multi-authority CP-ABE scheme, and apply it as the underlying techniques to design the data access control scheme. Our attribute revocation method can efficiently achieve both forward security and backward security. The analysis and simulation results show that our proposed data access control scheme is secure in the random oracle model and is more efficient than previous works.

Keywords
Access control, multi-authority, CP-ABE, attribute revocation, cloud storage

I. Introduction
All Data access control is an efficient way to ensure the data security in the cloud. Cloud storage services allows data owner to outsource their data to the cloud. Attribute-based encryption (ABE) is a new concept of encryption algorithms that allow the encryptor to set a policy describing who should be able to read the data. In an attribute-based encryption system, private keys distributed by an authority are associated with sets of attributes and ciphertexts are associated with formulas over attributes. A user should be able to decrypt a ciphertext if and only if their private key attributes satisfy the formula. In traditional public-key cryptography, a message is encrypted for a specific receiver using the receiver’s public-key. Identity-based cryptography and in particular identity-based encryption (IBE) changed the conventional understanding of public-key cryptography by allowing the public-key to be an arbitrary string, e.g., the email address of the receiver. ABE goes one step further and defines the identity not atomic but as a set of attributes, e.g. roles, and messages can be encrypted with respect to subsets of attributes (key-policy ABE - KP-ABE) or policies defined over a set of attributes (ciphertext-policy ABE - CP-ABE).

In ciphertext-policy attribute-based encryption (CP-ABE) a user’s private-key is associated with a set of attributes and a ciphertext specifies an access policy over a defined universe of attributes with in the system. A user will be able to decrypt a ciphertext if and only if his attributes satisfy the policy of the respective ciphertext. Cipher text-Policy Attribute-based Encryption (CP-ABE) is considered as one of the most suitable scheme for data access control in cloud storage. This scheme provides data owners more direct control on access policies. However, CP-ABE schemes to data access control for cloud storage systems are difficult because of the attribute revocation problem. So this paper produces survey on efficient and revocable data access control scheme for multi-authority cloud storage systems, where there are multiple authorities cooperate and each authority is able to issue attributes independently.

CP-ABE thus allows to realize implicit authorization, i.e., authorization is included into the encrypted data and only people who satisfy the associated policy can decrypt data. Another nice feature is that users can obtain their private keys after data has been encrypted with respect to policies. So data can be encrypted without knowledge of the actual set of users that will be able to decrypt, but only specifying the policy which allows decrypting. Any future users that will be given a key with respect to attributes such that the policy can be satisfied will then be able to decrypt the data.

II. System Model and Security Model
System Model
We consider data access control system in multi-authority cloud storage, as described in Fig. 1. 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. Every AA is an independent attribute authority that is responsible for creating and revoking user’s attributes according to its role or identity in its domain. In our 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 semantic of its attributes. Each AA is responsible for generating...
apublic attribute key for each attribute it manages and a secret key for each user reflecting his/her attributes.

Each user has a global identity in the system. A user may be entitled to 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. The server does not rely on the cloud server for data access control. But, the access control happens inside the cryptography. It 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 numbers of content keys and thus obtain different granularities of information from the same data.

**Security model**

In multi-authority cloud storage systems, we 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.

**III. CP-ABE**

One of the most suitable technologies for data access control in cloud storage systems is Cipher text-Policy Attribute-based Encryption (CP-ABE). It provides the data owner to direct control on access policies. The Authority in this scheme is responsible for key distribution and attribute management. The authority may be the university Administration office, Staff maintenance (Human resource-HR) department in a company, etc. The data owner in CP-ABE scheme defines the access policies and encrypts data depending on the policies.

**A. CP-ABE Types**

In CP-ABE scheme for every user will be issued a secret key reflecting its attributes. A user can decrypt the data only when its attributes satisfy the access policies.

There are two types of CP-ABE systems:

- Single-authority CP-ABE
- Multi-authority CP-ABE

In single-authority CP-ABE method, where all the attributes are managed by only one single authority. In a Multi-authority CP-ABE scheme where attributes are from different attribute authorities. This method is more suitable for data access control of cloud storage systems. Data users contain attributes should be issued by multiple authorities and data owners. Data users may also share the data using access policy defined over attributes from different authorities.

In our scheme, the data owner does not required to trust the server. Because, the key is based on attribute and maintained by the attribute authority. We designed new revocation method for multi-authority CP-ABE. Then, we apply them to design a fully secure and efficient data sharing for multi-authority scheme. The important advantages of this work can be summarized as follows,

- We proposed third party auditor (TPA) which used for auditing the data.
- We develop a new revocation method for user attribute revocation.

**B. CP-ABE Algorithm**

ACP-ABE scheme have four algorithms: Setup, Encrypt, KeyGen, and Decrypt.

1. **Setup (λ; U)**

The setup algorithm takes input as security parameter and attribute universe description. It outputs the global public parameters PK and a global master key MK.

2. **Encrypt (PK; M; A)**

The encryption algorithm takes as input the public parameters PK of attributes, a message M, and an access structure A over the attributes. The algorithm will encrypt M and produce a ciphertext (CT) that only a user having a set of attributes that satisfies the access structure will be able to decrypt the message. We will assume that the ciphertext implicitly contains A.

3. **Key Generation (MK; S)**

The key generation algorithm takes as input the global master key MK and a set of attributes S that clarify the key. It outputs a private key SK.

4. **Decrypt (PK; CT; SK)**

The decryption algorithm takes as input the public parameters PK, a ciphertext (CT), which contains an access policy A, and a private key SK, which is a private key for a set S of attributes. If the set S of attributes satisfies the access structure A then the algorithm will decrypt the ciphertext and return a message M.

**IV. Frame Work**

The data access control for Multi-Authority cloud storage system consists of following methods.
1) System Initialization

- **CA Setup** (\(\lambda\)): \((G\text{MK}, G\text{PP}, (G\text{PK}'\text{id}), (G\text{SK}id; G\text{SK}'id), \text{Certificate}\text{id})\).
- The CA setup algorithm is run by the CA. It takes no input other than the implicit security parameter \(\lambda\). It generates the global master key \(G\text{MK}\) of the system and the global public parameters \(G\text{PP}\). For each user \(id\), it generates the user’s global public keys \((G\text{PK}'id, G\text{PK}id)\), the user’s global secret keys \((G\text{SK}id, G\text{SK}'id)\) and a certificate \text{Certificate}\text{id} of the user.

- **AA Setup** \((\text{UA}\text{id})\):(SK\text{Aaid}, PK\text{Aaid}, \{VK\text{Aaid}, PK\text{Aaid} \} \text{xaid},\text{UA}\text{id})

The attribute authority setup algorithm is run by each attribute authority. It takes the attribute universe \(U\text{Aaid}\) managed by the \(\text{AA}\text{id}\) as input. It outputs a secret and public key pair \((\text{SK}\text{Aaid}, \text{PK}\text{Aaid})\) of the \text{Aaid} and a set of version keys and public attribute keys \(\{VK\text{Aaid}, PK\text{Aaid}\} \text{xaid},\text{UA}\text{id}\) for all the attributes managed by the \text{AA}\text{id}.

2) Attribute Authority’s key generation and management

- **Secret Key Distribution**

  A randomized algorithm takes as input the authority’s secret key \(SK\), a user’s \(u\)’s UID, and a set of attributes \(Aku\) in the authority AAK’s domain. (We will assume that the user’s claim of these attributes has been verified before this algorithm is run, \(A_{u} = \{A_{ku} , k = 1, \ldots , n\}\)). Output a secret key \(Du\) for the user \(u\).

- **Access issue id Distribution**

  The collected attributes from all attribute authorities (\(A\text{a}\)) will be sent to the users for the encryption purpose.

3) Data Encryption

The data owner runs the encryption algorithm to encrypt the content keys. By using symmetric encryption method the data is encrypted with content keys. A randomized algorithm takes as input a set of public key of attributes involved in encryption, a message \(M\), the global public parameters \(G\text{PP}\) and outputs the ciphertext \(C\).

4) Data Decryption

The users first run the decryption algorithm and use them to decrypt data’s from the ciphertext \(C\). It takes input the \(V\text{Ciphertext} C\), it have access policy with itself for verifying the access rules of the users. If the access policy is satisfied with \(V\) the users attribute, the decryption algorithm will decrypt the \(V\text{Ciphertext} C\).

5) Attribute revocation

The attribute revocation has been solved by assigning new version key \(VK\) for non-revoked attribute. It takes as input the secret key of Attribute authority, revoked attribute id and current version key. Its outputs as new version key and new attribute key.

V. Our Data Access Control Scheme

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

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, 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.

We propose a new revocable multi-authority CP-ABE protocol based on the single-authority CP-ABE proposed by Lewko and Waters in. That is we extend it to multi-authority scenario and make it revocable. We apply the techniques in Chase’s multi-authority CP-ABE protocol to tie together the secret keys generated by different authorities for the same user and prevent the collusion attack. Specifically, we 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 id to each user and a global authority identity id to each authority attribute in the system. Because the id is globally unique in the system, secret keys issued by different AAs for the same user can be tied together for decryption. Also, because each AA is associated with an id, every attribute is distinguishable even though some AAs may issue the same attribute.

To deal with the security issue in, instead of using the system unique public key (generated by the unique master key) to encrypt data, our scheme requires all attribute authorities to generate their own public keys and use them to encrypt data together with the global public parameters. This prevents the certificate authority in our scheme from decrypting the ciphertexts.

To solve the attribute revocation problem, we assign an 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 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, we 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 their previous secret keys.

VI. Security Analysis

We prove that our data access control is secure under the security model we defined, which can be summarized as in the following theorems.

Theorem 1. When the decisional \(q\)-parallel BDHE assumption holds, no polynomial time adversary can selectively break our system with a challenge matrix of size \(1\times n\), where \(n \geq q\).

Proof. The proof is given in the supplemental file available online.

Theorem 2. Our scheme can achieve both Forward Security and Backward Security.
Backward Security: 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. Moreover, suppose the revoked user can corrupt some other AAs (not the AA corresponding to the revoked at-tributes), the item $H_{\text{Auxid}}(\text{aid}_{\text{aid}})$ in the secret key can prevent users from updating their secret keys with update keys of other users, since _aid is only known by the AA_{aid} and kept secret to all the users. This guarantees the backward security. 

Forward Security: After each attribute revocation operation, the version of the revoked attribute will be updated. When new users join the system, their secret keys are as-associated with attributes with the latest version. However, previously published cipher texts are encrypted under at-tributes with old version. The cipher text update algorithm in our protocol can update previously published cipher-texts into the latest attribute version, such that newly joined users can still decrypt previously published cipher texts, if their attributes can satisfy access policies associated with cipher texts. This guarantees the forward security.

Theorem 3. Our access control scheme can resist the collusion attack, even when some AAs are corrupted by the adversary. Each other, although some AAs may issue the same attributes. Moreover, the secret key is also associated with the user’s globally unique identity uid. Thus, users cannot collude together to gain illegal access by combining their attributes together.

However, when some AAs is corrupted by the adversary, the collusion resistance becomes more complicated. Specifically, the adversary may launch Attribute Forge Attack, defined as follows. Suppose a user uid0 possesses an attribute “xaid0” from AA_{aid0}, while the adversary does not hold the attribute “xaid0” from AA_{aid0}. The adversary attempts to forge (”clone”) the attribute “xaid0” from the user uid0’s secret key by colluding with some other AAs. In our scheme, the item $g_{\text{Uid}} \text{uid};\text{aid}_{\text{aid}}$ in the secret key construction helps to resist this attack. When the adversary corrupts any AAs, he/she can get all the global secret key GSK{uid} for all the users in the system (because each AA has full knowledge on one of the user’s global secret keys GSK{uid}). Suppose all the $K_{\text{Xaid}}.\text{aid}$ in the secret key is constructed without this item. The adversary can success- fully forge the attribute “xaid0” as Privacy-Preserving Guarantee: 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_{aid}, the CA cannot decrypt any ciphertexts in the system. Our scheme can also prevent the server from getting the content of the cloud data by using the proxy-encryption method.

VII. Performance Analysis

In this section, we analyze the performance of our scheme by comparing with the Ruj’s DACC scheme and our previous scheme in the conference version, in terms of storage overhead, communication cost and computation efficiency. We conduct the comparison under the same security level. Let $p$ be the element size in the $G;$ GT ; $Z_p$. Suppose there are nA authorities in the system and each attribute authority AA_{aid} manages naid attributes. Let nU and nO be the total number of users and owners in the system respectively. For a user uid, let nuid;aidk $\frac{1}{j}$ |Suid;aidk j denote the number of attributes that the user uid obtained from AA_{aidk}. Let t be the total number of attributes in the ciphertext.

VIII. Feature Work

Ciphertext-Policy Attribute-based Encryption (CP-ABE), is regarded as one of the most suitable technologies for data access control in cloud storage systems, because it gives the data owner more direct control on access policies. In CP-ABE scheme, there is an authority that is responsible for attribute management and key distribution. The authority can be the registration office in a university, the human resource department in a company, etc. The data owner defines the access policies and encrypts data according to the policies. Each user will be issued a secret key reflecting its attributes. A user can decrypt the data only when its attributes satisfy the access policies. In this paper, we first propose 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.

1. We modify 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.
2. We greatly improve the efficiency of the attribute revocation method.

IX. Conclusion

In this paper, we proposed a revocable multi-authority CPABE scheme that can support efficient attribute revocation. Then, we constructed an effective data access control scheme for multi-authority cloud storage systems. We also proved that our scheme was provable secure in the random oracle model. The revocable multi-authority CPABE is a promising technique, which can be applied in any remote storage systems and online social networks etc.

Ciphertext-Policy Attribute-Based Encryption (CP-ABE) 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 where all attributes are managed by a single authority, and multi-authority CP-ABE, 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.

References


