Multi-Level Attribute-Based Encryption Access Control Scheme for Big Data

Zhao Li*, *Shuiyuan* Huan

Chongqing University of Posts and Telecommunications, Chongqing Mobile Internet Application Engineering Technology Research Center, 400065 Chongqing City, China

Abstract. There are many security threats such as data's confidentiality and privacy protection in the new application scenario of big data processing, and for the problems such as coarse granularity and low sharing capability existing in the current research on big data access control, a new model to support fine-grained access control and flexible attribute change is proposed. Based on CP-ABE method, a multi-level attribute-based encryption scheme is designed to solve fine-grained access control problem. And to solve the problem of attribute revocation, the technique of re-encryption and version number tag is integrated into the scheme. The analysis shows that the proposed scheme can meet the security requirement of access control in big data processing environment, and has an advantage in computational overhead compared with the previous schemes.

1 Introduction

Nowadays, the society is in a big data era of large-scale production, sharing and application of data, big data analysis and processing technologies are widely used in all walks of life. However, when people use big data technology to create value, they are also plagued by the security problems that they bring. Compared with ordinary data, big data faces more serious security threats in data collection, storage, sharing and utilization, therefore, the need for big data security and privacy protection is becoming more and more urgent.

In the context of big data applications, access control technology, as one of the important means to ensure data security sharing, will still play an important role -- it can guarantee that resources can only be operated legally by a legitimate user to prevent unauthorized access to information, based on a predetermined access control strategy. However, how to implement secure access control of big data, that is, to ensure data's confidentiality and achieve conditional data sharing, the traditional solutions need to be improved and innovated to meet the new security requirements under new application scenarios.

At present, a typical enterprise big data application scenario is as follows: diversified data are gathered from multiple service channels to the enterprise big data processing platform for processing and analysis, the analysis results will be used by enterprise subordinate organizations or business departments according to their business needs. Traditional access control methods applied to this application scenario will encounter many problems: 1) Coarse granularity. There are many participants in the new application scenario, it may cause a problem of over-authorization or under-authorization due to non-provisioned permissions; 2) The same resource may be shared among various internal business departments and the third-party organizations, which may lead to repeated processing of data. So, with the wide application of the mobile Internet and the Internet of Things, the demand for cross-enterprise and cross-industry big data applications will rapidly increase in the future, and the problem of secure big data access control will inevitably become more prominent.

Attribute-based encryption (ABE) scheme enables secure, fine-grained access control, which provides a way to achieve access control for big data. However, it still needs to be comprehensively considered in the following aspects: 1) Because of its complexity, ABE is not suitable for encryption of large-scale data, it should be improved to adapt to the data size and the growth rate in big data application scenarios; 2) Big data applications are quite complex for data collection, storage, sharing, and utilization, these steps are quite time-consuming, so the repeated processing of data should be avoided in big data access control scheme; 3) In the big data application scenario, it is generally necessary to set up a large set of attributes to achieve fine-grained access control, so how to realize flexible and efficient attribute revocation in big data access control scheme is an important issue.

This paper presents a multi-level access control scheme based on ciphertext-policy attribute-based encryption (CP-ABE) scheme, which can realize secure and flexible access control in big data environment.

2 Related work

* Corresponding author: li27z@qq.com

Conventional access control models, such as role-based access control (RBAC), risk-based access control, cannot provide fine-grained and flexible access control when applied to new computing environment. In comparison, attribute-based encryption can provide more secure and efficient access control and are widely discussed in the new computing environment. Goyal et al. [1] first proposed key-policy attributed-based encryption (KP-ABE) to serve a more general and richer encrypted access control, but it lacks the authority control over the policy for enforcement. In 2007, Bethencourt et al. [2] proposed ciphertext-policy attributed-based encryption (CP-ABE) scheme to address this limitation, attributes are used to describe a user's credentials and a party encrypting data determines a policy for who can decrypt.

In CP-ABE scheme, the private key is only determined by user's attributes, user privacy can be effectively protected in the process of encryption and decryption, and the paper [3] proposed an access control scheme for big data environment using these security features. Fine-grained access control is also an important demand for big data access control scheme, Somechart Fugkeaw and Hiroyuki Sato [4] proposed an access control model combining role-based access control, symmetric encryption and ciphertext attribute-based encryption to support fine-grained access control for big data environment.

And for the complexity of the attribute-based encryption model, it will cause huge system cost to encrypt large-scale data, so the access control scheme in big data environment should also concentrate on how to improve efficiency. Yang K and Jia X [5] designed an efficient multi-authority CP-ABE scheme that does not require a global authority and can support any Linear Secret Sharing Scheme(LSSS) structure, and the scheme is proved to be scalable and efficient in its application scenario. Encryption and decryption outsourcing can also improve the efficiency of access control schemes, the paper [6] proposed an ABE access control scheme that supports multiple authority and revocation, and the decryption operation is outsourced for higher efficiency.

In the new computing environment, the large-scale feature of users and attributes also brings about the attribute revocation problem. Existing permissions management schemes are generally divided into two categories: direct management and indirect management. Direct management is generally maintained by manual methods, which does not apply to numerous users and the huge number of attributes in the new computing environment. Indirect management is relatively more flexible and efficient, for example, Yu et al. [7] introduced a proxy re-encryption mechanism and added version information to the attribute revocation scheme to implement flexible attribute changes. The paper presented an attribute-based secure data sharing scheme with efficient revocation combined CP-ABE, symmetric encryption and homomorphic encryption techniques, it can achieve immediate attribute revocation and solve the key escrow problem in current attribute based data sharing schemes.

The above analysis shows that, existing access control schemes are mostly researched in cloud storage environment, they are not suitable for big data environment, and few proposed big data access control schemes take attribute revocation problem into consideration. Access control solutions for big data are now very demanding and, it is necessary to study a secure and flexible access control scheme in the context of big data applications to achieve a relative balance between attribute change granularity and computing resource consumption.

3 Background

3.1 Access structure

Let $\{P_1, P_2, \ldots, P_n\}$ be a set of parties. A collection $A \subseteq 2^{\{P_1, P_2, \ldots, P_n\}}$ is monotone if $\forall B, C$: if $B \in A$ and $B \subseteq C$ then $C \in A$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) A of non-empty subsets of $\{P_1, P_2, \ldots, P_n\}$, i.e., $A \subseteq 2^{\{P_1, P_2, \ldots, P_n\}} \setminus \{\emptyset\}$. The sets in A are called the authorized sets, and the sets not in A are called the unauthorized sets.

In our context, the role of the parties is taken by the attributes. Thus, the access structure A will contain the authorized sets of attributes.

3.2 Bilinear maps

Our scheme is based on some facts about groups with efficiently computable bilinear maps.

Let G_1 and G_2 be two multiplicative cyclic groups of prime order p and e be a bilinear map, $e: G_1 \times G_1 \rightarrow G_2$. Let g be a generator of G_1.

The bilinear map e has the following properties:

1) Bilinearity: for all $u, v \in G_1$ and $a, b \in Z_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$;

2) Non-degeneracy: $e(g, g) \neq 1$;

3) Computability: There is an efficient algorithm to compute e(u, v) for $\forall u, v \in G_1$.

3.3 CP-ABE scheme

An CP-ABE scheme consists of four algorithms:

1) Setup: This algorithm completes the work of the system initialization, generates a public key PK used for encryption and a master secret key MK held by the central authority for generating user secret keys.

2) Encrypt(PK, M, A): This algorithm takes the public key PK, an access structure A and a message M as input. It returns a ciphertext CT such that a secret key generated from user attribute set can be used to decrypt CT if and only if the attributes satisfy A.

3) KeyGen(MK, γ): This algorithm takes MK and a user attribute set γ as input and generates a secret key SK associated with γ.

4) Decrypt(PK, CT, SK): This algorithm takes PK, CT and SK associated with γ as input and returns the message M if the attribute set γ satisfies the ciphertext policy A specified for CT.

4 The proposed scheme

In this section, we present our scheme that achieves fine-grained access control and flexible access revocation in big data environment.

4.1 System model

The system model is as shown in Fig.1, it consists of Data Owner, Cloud Service Provider, Center Authority, Department User and User.

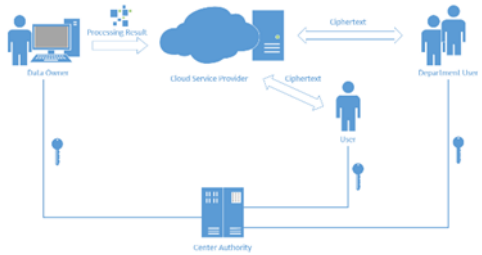

Fig. 1. Big Data Access Control Architecture..

1) Data Owner (DO): DO is the owner of the original data, and the original data will be processed according to the business needs. In our scheme, DO will encrypt the processing result with its own access policy, and store the ciphertext in the cloud storage server.

2) Cloud Service Provider (CSP): CSP provides users with data storage services, and has powerful computing and storage capabilities. This paper assumes that CSP is not completely trusted, but it will not take the initiative to reveal the stored information, and there will be the risk of privacy leaks caused by malicious attacks.

3) Center Authority (CA): CA will generate the public parameters of the system initialization, the master key and re-encryption key, and generate the encryption, decryption key according to the attributes and the access structures.

4) Department User (DU): DU can not only get the resources on CSP, but also can take advantage of the resources already acquired to carry out the next level of access control. DU requests CSP for the ciphertext of the processing result and applies to CA for the decryption key, it does not immediately decrypt the ciphertext, but uses the decryption key to carry out the next level of access control. DU will encrypt the key by its own access policy, then store the new ciphertext with specific organizational structure on CSP.

5) User (U): User can get the ciphertext of the new resource (the key of the processing result ciphertext) on CSP, and when the attribute set of U satisfies the access policy of the resource, U can get the decryption key, then U can decrypt to get the plaintext of the processing result by the key.

4.2 Our construction

In this section, we describe the construction of our scheme.

4.2.1 System initialization

This step outputs the system public parameter PK, the system master key MK and the Re-Encryption key set RK by calling the algorithm Setup.

Setup → PK, MK, RK:

$$PK = G_1, g, h = g^{\beta}, e(g, g)^{\alpha} \quad (1)$$

$$MK = (\beta, g^{\alpha}) \quad (2)$$

$$RK = \{rk_1, rk_2, \ldots, rk_n\} \quad (3)$$

The Setup algorithm will choose a bilinear group G1 of prime order p with generator g. Next it will choose two random exponents $\alpha, \beta \in Z_p$. And it will also choose a number of random exponents rk_i (i is the version number, $1 \le i \le n) \in Z_p$ to make up the Re-Encryption key set like scheme [8], they will be used in attribute revocation.

4.2.2 Big data processing result encryption

This step will encrypt the big data processing result by calling the algorithm CP-ABE Encrypt.

Encrypt(PK, M, T) → CT_M:

$$CT_M = (T, \widetilde{C} = M \cdot e(g, g)^{\alpha \cdot s}, C = h^s,$$

$$\forall y \in Y: C_y = g^{q_y(0)}, C_y' = H(att(y))^{q_y(0)}) \quad (4)$$

Let T be a tree that represent an access structure, and let Y be the set of leaf nodes in T. The algorithm chooses a polynomial q_x for each node x (including the leaves) in the tree T, and for the root node R, the algorithm chooses a random $s \in Z_p$ and sets $q_R(0) = s$. The function att(x) is defined only if x is a leaf node and denotes the attribute associated with the leaf node x in the tree, and H(i) represents the hash function of attribute i.

4.2.3 DU key generation

This step will generate the private key for every DU request.

KeyGen(MK, γ) → SK_{DU}:

$$SK_{DU} = (D = g^{(\alpha+r)/\beta}, \forall j \in \gamma: D_j = g^r \cdot H(j)^{r_j}, D_j' = g^{r_j}) \quad (5)$$

The key generation algorithm will take as input a set of attributes γ and output a key that identifies with that set. The algorithm first chooses a random $r \in Z_p$, and then random $r_j \in Z_p$ for each attribute $j \in \gamma$.

4.2.4 Re-Encryption key generation

To ensure the uniqueness of every DU request to CT_M, and ensure that it is not affected by policy change or attribute change, we need to re-encrypt the obtained ciphertext and the private key. This step will generate a re-encryption key $rk_{DU} \in Z_p$, and it will be used in the

following two steps to re-encrypt the ciphertext and the private key.

4.2.5 Re-Encrypt the ciphertext

ReEncrypt(CT_M, rk_{DU}) → CT'_M:

$$CT'_M = (T, \tilde{C} = M \cdot e(g,g)^{\alpha \cdot s}, C = h^s, \forall y \in Y: C_y = g^{q_y(0)}, C'_y = (H(att(y))^{q_y(0)})^{rk_{DU}}) \quad (6)$$

4.2.6 Re-Encrypt the key

ReKey(SK_{DU}, rk_{DU}) → SK'_{DU}:

$$SK'_{DU} = (D = g^{(\alpha+r)/\beta}, \forall j \in \gamma: D_j = g^r \cdot H(j)^{r_j}, D'_j = (g^{r_j})^{1/rk_{DU}}) \quad (7)$$

4.2.7 DU private key encryption

In this step, DU will encrypt SK'_D, that is, the private key of DU after re-encryption, to be our new access control resource. The encrypt method is almost the same as step 4.2.2, but we still need to put the re-encryption key with version number generated in Setup algorithm in our construction, the initial value of the version number is 1.

Encrypt(PK, M', T') → CT_{DU}:

$$CT_{DU} = (T', \tilde{C} = M' \cdot e(g,g)^{\alpha \cdot t}, C = h^t, \forall y \in Y': C_y = g^{q_y(0)}, C'_y = (H(att(y))^{q_y(0)})^{rk_{y\text{-}Ver}}) \quad (8)$$

In this algorithm, the message M' is SK'_D, T' is the tree access structure set by DU, Y' is the set of leaf nodes in T'. And for the root node R, the algorithm chooses a random $t \in Z_p$ and sets $q_R(0) = t$. Then send the tuple {Ver, CT_{DU}, CT'_M} to CSP, CSP stores the new resource.

4.2.8 U key generation

This step is similar to *4.2.3*, execute the KeyGen algorithm to generate U's private key. And the re-encryption key associated with the version number also needs to be embedded in the algorithm construction.

KeyGen(MK, I) → SK':

$$SK_U = (D = g^{(\alpha+u+rk_{Ver})/\beta}, \forall i \in I: D_i = g^{u+rk_{i\text{-}Ver}} \cdot H(i)^{r_i \cdot rk_{i\text{-}Ver}}, D'_i = g^{r_i}) \quad (9)$$

In this algorithm, I is a set of user attributes, it chooses a random $u \in Z_p$, and then random $r_i \in Z_p$ for each attribute $i \in I$.

4.2.9 U access

When U requests the resource on CSP, CSP will check whether the version number is equal, if true, U will receive the tuple {Ver, CT_{DU}, CT'_M} from CSP, then U will call the algorithm CP-ABE Decrypt to decrypt CT_{DU} and CT'_M.

Decrypt(PK, CT_{DU}, SK_U) → M':

Compute DecryptNode(CT_{DU}, SK_U, x) = e(g, g)$^{(u+rk_{Ver}) \cdot q_x(0)}$, we can finally get the result is e(g, g)$^{(u+rk_{Ver}) \cdot t}$, and we set A = DecryptNode(CT_{DU}, SK_U, x). The algorithm now decrypts by computing $\tilde{C}$/(e(C, D)/A) = M'. Then we continue to decrypt CT'_M.

Decrypt(PK, CT_M, $M' = SK'_D$) → M:

Similarly, compute DecryptNode(CT_M, SK'_D, x) = e(g, g)$^{\alpha \cdot s}$ and set A = DecryptNode(CT_M, SK'_D, x), then compute $\tilde{C}$/(e(C, D)/A), we can get the processing result M.

4.3 Attribute revocation

When a User's attribute is revoked and does not satisfy the access structure, we need to re-encrypt our resource to prevent User from accessing the resource with the acquired private key. For convenient re-encryption, the version number of attribute set has been embedded in the ciphertext and the User's private key at the time of encryption and key generation, and the re-encryption key is associated with the version number.

The re-encryption operation includes re-encrypt the ciphertext and re-encrypt the private key, and we define two algorithms as follows:

CT-ReEncrypt(RK, CT)

Let $C'_{i\text{-}k}$ be the ciphertext of attribute i with version number k, and it will be $C'_{i\text{-}(k+1)}$ after update, the computing method is as follows:

$$C'_{i\text{-}(k+1)} = (C'_{i\text{-}k})^{rk_{k+1}/rk_k} = H(i)^{q_x(0) \cdot rk_k \cdot rk_{k+1}/rk_k} = H(i)^{q_x(0) \cdot rk_{k+1}}$$

So, the re-encryption key $RK_{k \to k+1} = rk_{k+1}/rk_k$.

SK-ReEncrypt(RK, SK)

Input the attribute i that needs to be updated (the version is k) and the re-encryption key, we can get the new secret key with version number (k+1):

$$D_{i\text{-}(k+1)} = D_{i\text{-}k} \cdot H(i)^{r_i \cdot (rk_{k+1} - rk_k)} \cdot g^{u+rk_{k+1}-rk_k} = g^{u+rk_k} \cdot H(i)^{r_i \cdot rk_k} \cdot H(i)^{r_i \cdot (rk_{k+1}-rk_k)} \cdot g^{u+rk_{k+1}-rk_k} = g^{u+rk_{k+1}} \cdot H(i)^{q_x(0) \cdot rk_{k+1}}$$

After the above re-encryption operation, the information in both the ciphertext and the key is changed, so that the key held by the revoked User can no longer satisfy the attribute structure in the ciphertext, and the correct result cannot be obtained when decrypting.

5 Discussion

5.1 Security analysis

5.1.1 Security proof

The paper [2] has proved the security of CP-ABE scheme and our scheme implements re-encryption based on CP-ABE, so the security has not reduced.

Proof. As for ciphertext, let $q_y(0)^{'} = q_y(0)\ rk_{Ver}$, because $q_y(0)$ and rk_{Ver} are all random, so $q_y(0)^{'}$ is also a random number, so that we can get $C_y^{''} = C_y^{'} = H(att(y))^{q_y(0)^{'}}$, and $CT_{DU} = (T^{'}, \tilde{C}, C, C_y, C_y^{''})$. Similarly, $CT_M^{'}$, $SK_{DU}^{'}$ and SK_U are all random, thus, the security of the scheme is not less than the CP-ABE scheme.

5.1.2 Preventing collusion

In order to decrypt the ciphertext, the collusion attackers must recover $e(g, g)^{(u+rk_{Ver})t}$ and $e(g, g)^{\alpha \cdot s}$. To do this, the attackers must provide enough attribute keys to satisfy the access tree $T^{'}$ and T, but attribute keys that satisfy each attribute in $T^{'}$ and T come from different keys. Due to the different random numbers used in encryption, it cannot satisfy the polynomial interpolation when decrypting, and cannot recover the plaintext. Therefore, the combination of key components from different keys cannot decrypt other files, so can withstand collusion attacks of attackers.

5.2 Efficiency

We analyze and compare the computation complexity of the proposed scheme with the other big data access control schemes based on CP-ABE in Table.1.

Table 1. Scheme complexity comparison.

Scheme	Encryption	Decryption
Scheme[3]	O(|U|)	O(|U|)
Scheme[4]	O(|N|)	O(|N|)
Our scheme	O(|S|)+O(|I|)	O(|S|)+O(|I|)

|U| denotes the number of attributes in the user attribute set in scheme [3], |N| denotes the number of attributes in the user attribute set in scheme [4], and |S| and |I| denote attribute number in attribute set S of access tree T and attribute set I of access tree $T^{'}$. The amount of computation overhead is proportional to the number of attributes.

Firstly, as we can see from the table, the efficiency is related to the size of the attribute set, and |S| + |I| is approximately equal to |U| and |N|, our scheme maintains high performance and it avoids the repetitive execution of big data processing. Secondly, the proposed scheme achieves a more fine-grained access control, it designs two levels of access control and grants privileges to DU and then to U. And it deals with the attribute revocation in the case of a huge set of attributes well, so our scheme is relatively a thorough and applicable big data access control scheme.

6 Conclusion

In this paper, aiming at solving the problem that fine-grained access control and flexible attribute change are difficult to implement in the new application scenarios of big data, a multi-level access control scheme based on CP-ABE is proposed. Security analysis shows that the proposed scheme can not only achieve fine-grained access control, but also support resisting the collusive attack. As for efficiency, the scheme can achieve more efficient access control in big data scenarios, and can avoid the repetitive execution of big data processing. The next work will make an analysis of the changes in the attributes of DU and the further improvement of the efficiency in the scheme.

References

1. Goyal V, Pandey O, Sahai A, et al. Attribute-based encryption for fine-grained access control of encrypted data. 89-98 (2006)
2. Bethencourt J, Sahai A, Waters B. Ciphertext-Policy Attribute-Based Encryption. 321-334 (2007)
3. Yuan Q, Ma C, Lin J. Fine-Grained Access Control for Big Data Based on CP-ABE in Cloud Computing. **503,** 344-352 (2015)
4. Fugkeaw S, Sato H. Privacy-preserving access control model for big data cloud. 1-6 (2016)
5. Yang K, Jia X. Attributed-based access control for multi-authority systems in cloud storage. 536-545 (2012)
6. Ying-Hui Z, Dong Z, Jin L and Hui L. Attribute Directly-revocable Attribute-based Encryption with Constant Ciphertext Length. **5,** 465–480 (2014)
7. Yu S, Wang C, Ren K, et al. Achieving secure, scalable, and fine-grained data access control in cloud computing. 534-542 (2010)
8. Guan Zhi-tao, Yang Ting-ting, Xu Rui-zhi, Wang Zhu-xiao. Multi-authority attribute-based encryption access control model for cloud storage. **36**, 116–126 (2015)