Computer Science and Information Systems 16(3):797–813 https://doi.org/10.2298/CSIS180912027T

Hierarchical Authority Based Weighted Attribute Encryption Scheme

Qiuting Tian, Dezhi Han, and Yanmei Jiang

College of Information Engineering, Shanghai Maritime University
Shanghai, China
{tqt2018, jym2097}@163.com, dzhan@shmtu.edu.cn

Abstract. With the development of cloud storage technology, data storage security has become increasingly serious. Aiming at the problem that existing attribute-based encryption schemes do not consider hierarchical authorities and the weight of attribute. A hierarchical authority based weighted attribute encryption scheme is proposed. This scheme will introduce hierarchical authorities and the weight of attribute into the encryption scheme, so that the authorities have a hierarchical relationship and different attributes have different importance. At the same time, the introduction of the concept of weight makes this scheme more flexible in the cloud storage environment and enables fine-grained access control. In addition, this scheme implements an online/offline encryption mechanism to improve the security of stored data. Security proof and performance analysis show that the scheme is safe and effective, and it can resist collusion attacks by many malicious users and authorization centers. It is more suitable for cloud storage environments than other schemes.

Keywords: cloud storage, attribute-based encryption, weighted attribute, access control, hierarchical authority.

1. Introduction

With the large-scale deployment of cloud storage systems, a large amount of sensitive data is outsourced to cloud storage servers [19,7]. However, the outsourced storage mode of data also brings some security problems. For example, the cloud server can expose the privacy data of the user or the enterprise without the authorization of the user for certain benefits or curiosity. As one of the solutions, the attribute-based encryption (ABE) [20] mechanism emerged as the times require, and has become a hot topic in recent years. Most of the early ABE solutions were for a single authority, and the attribute private keys of all users were distributed by a single authority, which increased the burden on the authority. And in the current cloud environment [13], many scenarios involve massive users and massive data, such as personal health record sharing systems. If us still use a single authority in this scenario, it will cause a system bottleneck. Once the attacker breaks the server, it will bring inevitable losses to the user, and the early encryption method is not applicable to the current distributed computing environment. If it distributes the user's decryption key through multiple authorities, the resulting loss may be less. In practical applications, multi-authority can process user attributes quickly and efficiently, and manage the user's private key. To solve the problems caused by a single authority, Chase [5]

first proposed the ABE scheme of the multi-authority, but the scheme only supports the basic ABE algorithm and lacks the flexibility of access. In [6], in view of the hidden dangers of the existence of central authority (CA), an improved CA-free multi-authority ABE scheme is proposed. Yang et al. [24] proposed an effective data access control for multi-authority cloud storage systems, which outsources the decrypted main calculations to the cloud server. Gorasia et al. [12] proposed a multi-authority ABE scheme for fast decryption algorithm, but did not give corresponding security proof.

Since the disadvantage of the traditional ABE mechanism is that the attribute structure is not layered, all attributes have the same security level, and it supports only a single assignment of attributes. Such an attribute structure makes it difficult to have flexible, fine-grained access. Although there are many multi-authority based on attribute access control, most of these solutions do not consider the weight of attributes, that is to say, attributes are equal. But in practical applications, the attribute with weight has practical significance.

Aiming at the above problems, this paper designs a flexible and efficient hierarchical authority based weighted attribute encryption scheme. Our research contributions are summarized as follows:

1) This method uses a layered certification authority to distribute private keys for users. On the one hand, it avoids problems such as server load operation caused by a single certification authority; on the other hand, it facilitates management by trusted authorities. It can achieve more efficient hierarchical management between authorities.

2) The method introduces attribute weights into the encryption scheme, so that the highest weights of the attribute private keys distributed by the authorities of different levels to the users are different, thereby achieving fine-grained access control. By splitting the attributes into different sets, each level of authority selects the appropriate subset based on the weights and distributes them to the lower authority or user. This scheme is suitable for large authorities, which is convenient for managing users and authorities at each level, and also improves system security.

3) This method divides the encryption process into two phases, online and offline. The offline phase pre-processes complex calculations, while the online phase requires only a small amount of simple calculations. The online and offline double-layer encryption greatly improves the security of the entire encryption phase, and is suitable for cloud computing users with limited computing power.

The remaining of the paper is organized as follows. In Section 2, we introduce the research results of the other researchers, related to our research. Some preliminaries are given in Section 3. In Section 4, we propose the scheme formalization and security model. Next, the corresponding specific algorithm is presented. Section 6 presents the security analysis. Section 7 gives the performance analysis of our scheme. In the final section, the conclusion is given.

2. Related Work

The distribution of a large number of users in real life may be in a large hierarchical authority, and there may be a relationship between authorities when distributing private keys to users. At this time, it is necessary to consider how to achieve hierarchical access between authorities. Ref. [22] considered the hierarchy of the authority and proposed a

hierarchical ABE scheme, but in this scheme, there is no hierarchical difference in system attributes. Ref. [21] proposed a layered attribute set encryption scheme, which can achieve scalability through a layered structure. Ref. [27] began to study the attribute set and subset, and realized the stratification between attributes. The only drawback of these two schemes is that the connection between the authorities has not yet been given. If there is no intersection between the user's attribute sets, the application in the real scene is restricted. Subsequently, Ref. [17] proposed a hierarchical multi-authority and attribute-based encryption scheme, it employs character attribute subsets to achieve flexible fine-grained access control. In the system, trusted authority manages hierarchical attribute authorities, but must ensure that the trusted authority is absolutely trustworthy and that it is limited by the size of the attribute set. Ref. [1] gave the encryption scheme of the hierarchical authority, and the scheme does not consider the problem of attribute weights. In practical applications, each attribute has a different role and status, and the attribute has a practical value. In this scenario, the status of each attribute in the system is equal. Therefore, the concept of weights introduced into attributes is more suitable for the needs of practical systems.

Subsequently, Ref. [15] proposed a ciphertext-policy weighted ABE scheme, the attributes have different weights according to their importance. Ref. [23] proposed a multi-authority based weighted attribute encryption scheme. The attribute authorities assign different weights to attributes according to their importance. Ref. [9] proposed a weighted multi-authority attribute encryption based on ciphertext policy, which implements the encryption mechanism without a trusted center. However, in this scheme, the user's attribute private key and system key are calculated by the attribute authorization center, which causes a load of a single authorization center to be too large, and may eventually cause the attribute authority to crash. Although these schemes introduce the concept of weights, they do not consider the hierarchical relationship between the authorities. Ref. [8] proposed a multi-authority and hierarchical weighted attribute-based encryption scheme, which uses different levels of authorities to distribute different weighted attribute private keys. Thereby, finer-grained access control can be achieved, but the computational overhead of the scheme is large.

3. Preliminaries

3.1. Bilinear Pairing Map [4]

Let G_1 and G_2 be two multiplicative cyclic groups with the prime order P, and set the generator of group G_1 to be g. There exists a bilinear map e: $G_1 \times G_1 \rightarrow G_2$ that satisfies the following properties:

(1) Bilinearity: for all $u, v \in G_1$, $a, b \in Z_p$, there is $e(u^a, v^b) = e(u, v)^{ab}$.

(2) Non-degeneracy: for any $g \in G$, there is $e(g, g) \neq 1$.

(3) Computability: for all $u, v \in G_1$, there is an effective algorithm for calculating $e(u, v) \in G_2$.

3.2. Access Structure [8]

Let $\{p_1, p_2, ..., p_n\}$ denote a set of parties, set $P = 2^{\{p_1, p_2, ..., p_n\}}$. Access structure A is a non-empty subset of $\{p_1, p_2, ..., p_n\}$, i.e. $A \subseteq P \backslash \{\phi\}$. If the access structure A is

monotonic, then for $\forall B, C$, if $B \in A$ and $B \subseteq C$, then $C \in A$. The sets in A are the authorized sets, and the sets outside A are unauthorized sets.

3.3. Weighted Threshold Access Structure [2]

Let U be the set of all attributes, weight function is ω: $U \to N$, threshold is $T \in N$, define $\omega(A) = \sum_{u \in A} \omega(u)$ and $\Gamma = \{A \subset U : \omega(A) \geq T\}$. Then Γ is called as a weighted threshold access structure on U.

Each leaf node corresponds to the weight of one attribute, and the root node corresponds to the threshold. The size of the weight must be expressed as a positive integer. As is shown in Fig. 1, an example of a weighted threshold access structure is given. Suppose someone has three attributes: career, age and country. It distributes the corresponding weights of the three attributes in the leaf node. When the threshold value t is less than or equal to the sum of the weights of the three parts in the private key, it can decrypt the ciphertext. If user A's attribute set is $\{\mathrm{Director}, 55, \mathrm{China}\}$, User B's attribute set is $\{\mathrm{Manager}, 38, \mathrm{Australia}\}$. The system assigns weight values $\{7, 1, 1\}$ and $\{9, 4, 1\}$ to the attribute sets of users A and B, respectively. If the system threshold is set to 11, then the sum of the user's attribute weights must be greater than or equal to 11. All attribute weight values of user A add up to 9, which is less than the threshold value and cannot be decrypted. The total weight of the user B adds up to 14, which is greater than the threshold value, and it can decrypt the ciphertext to get the plaintext.

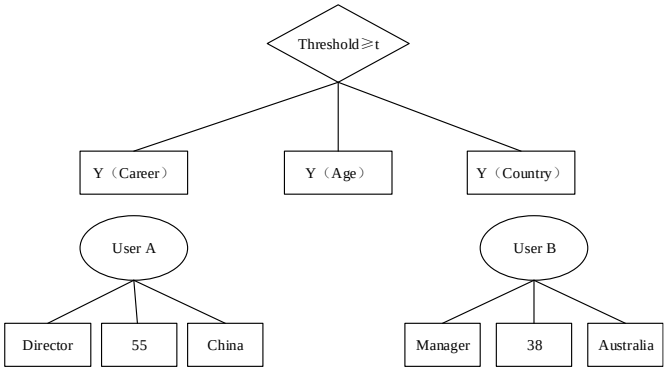

Fig. 1. Weighted threshold access structure

3.4. Attribute Set Split [16]

Split all the attributes in the system. For any attribute λ_i in the attribute set $\Gamma = \{\lambda_1, \lambda_2, ..., \lambda_n\}$, the maximum weight is computed as:

$$\omega_i = weight(\lambda_i) \tag{1}$$

According to the calculation result ω_i, the attribute λ_i is divided into $(\lambda_i, 1), ..., (\lambda_i, \omega_i)$. Assuming that the smallest split unit is 1 and the weights are only integers, the set formed is the split set Γ^* of weighted attributes.

4. Scheme Formalization and Security Model

This paper proposes a hierarchical authority based weighted attribute encryption scheme, which supports layering between authorities, and different attributes have different importance. Different levels of authority can be used to distribute the difference in attribute weights, enabling more fine-grained access control. In addition, because of the complexity of the access policy and the number of attributes in the existing ABE scheme, the overhead of encryption and key generation becomes large. This scheme introduces online/offline encryption technology, which divides the encryption process into an offline phase and an online phase [26]. The offline phase pre-processes complex calculations by the data owner during idle periods. In this way, the online phase requires only a small amount of simple calculations to generate ciphertext. This scheme is mainly composed of the following five parts: Cloud Service Provider (CSP), Trust Authority (TA), Attribute Authority (AA), Data owner (DO) and User.

CSP mainly provides cloud data storage service, stores data uploaded by DO; TA is responsible for generating and distributing system parameters, and manages upper-layer AA; upper-layer AA authorizes the lower-layer AA so that the lower-layer AA has the right to in charge some of the user's attributes. DO encrypts its own data and uploads it to the CSP, which is stored by the CSP; the user sends a file request to the CSP according to his/her needs, downloads the file from the cloud and decrypts it with the user's secret key. The structure of the system is shown in Fig. 2.

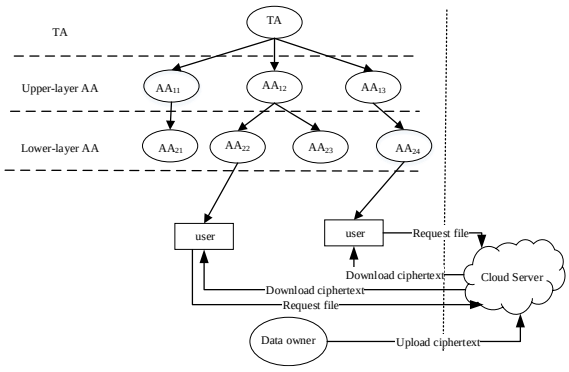

Fig. 2. System architecture of hierarchical authority

In this scheme, CSP is honest and curious. That is, CSP will process cloud data strictly according to the algorithms and protocols in the scheme, but, it will also snoop the owner's data as much as possible. TA is a trusted central authority, we believe TA is trustworthy, and lower-layer AA is semi-trustworthy. The user's attributes are primarily managed by

all AAs in the chain of authorization centers they are in. Suppose the user's attribute set is $A_u = \{A_1, A_2, ..., A_n\}$, and A_u is automatically divided into K disjoint subsets according to the hierarchy.

The divided subset is managed by K AAs on the AA chain and satisfies $\sum_{k=1}^{K} A_u^k = A_u$. As shown in Fig. 2 above, the user belongs to AA_{24}, assuming that all attribute sets owned by the user are: $\{Name : \text{Java}, \text{ID} : 687921, \{\text{Age} : 18, \text{Sex} : \text{Female}, \{\text{Job} : \text{Manager}, \text{Tel} : 1255766\}\}\}$.

Then TA manages attribute $\{Name : \text{Java}, \text{ID} : 687921\}$; A_{13} manages attribute $\{\text{Age} : 18, \text{Sex} : \text{Female}\}$; and A_{24} manages attribute $\{\text{Job} : \text{Manager}, \text{Tel} : 1255766\}$.

4.1. Concepts of the Scheme Formalization

This scheme is mainly composed of the following basic algorithms. The specific structure is as follows:

(1) TA $\text{Setup}(\gamma, \text{S}) \rightarrow (\text{PK}, \text{MK})$: Takes input as the security parameter γ and all attribute sets S in the system, and the system outputs public key PK and the master key MK.

(2) AA $\text{Setup}(A, \lambda, \lambda_i, \lambda_{i,j}) \rightarrow (\text{MK}_{k+1}, \text{M}_j)$: Operates the initial algorithm with every AA by inputting the arbitrary parameter λ, λ_i, $\lambda_{i,j}$ and the attribute set A of the upper-layer authority; the master key MK_{k+1} and the public key M_j of the lower-layer authority are output.

(3) $\text{KeyGen}(\text{MK}, \text{M}_j, \text{MK}_{k+1}, \text{u}) \rightarrow (PK_u, \text{SK}_u, \text{SK})$: The user applies for a private key to the AA. Takes input as the master key MK, M_j, user identifiers u and MK_{k+1} in the system, and the system outputs the user's system public key PK_u, system private key SK_u and user's private key SK.

(4) $\text{Encrypt}^{\text{Offline}}(PK, \text{M}_j, \Lambda) \rightarrow IT$: The data owner invokes the offline encryption algorithm according to the system public key PK, the authority public key M_j, and the access structure Λ to obtain the middle ciphertext IT.

(5) $\text{Encrypt}^{\text{Online}}(IT, \text{m}, S) \rightarrow CT$: The data owner invokes the online encryption algorithm according to the middle ciphertext IT, plaintext m and attribute set S, and obtains the complete ciphertext CT uploads it to the CSP.

(6) $\text{Decrypt}(\text{MK}_{k+1}, SK, \text{CT}) \rightarrow m$: The user accesses the CSP and downloads the file to the local, and then calls the decryption algorithm to decrypt the ciphertext CT according to the private key MK_{k+1} of the attribute authority and the user private key SK. If the attribute related to the user satisfies the access policy embedded in the ciphertext CT, the decryption obtains the plaintext m; otherwise the decryption fails.

4.2. Security Model

Theorem 1: If any attacker's advantage $Adv_{Ad}(k)$ in the security game is negligible, then the scheme is called CPA (Chosen-Plaintext Attack) [3] security. The following is the construction process of choosing the plaintext security model in the security game plan. Let the attacker be Ad and the challenger be C. The specific game process is as follows:

(1) Initial: Attacker Ad arbitrarily selects an access policy Λ^*, challenged by challenger C.

(2) Setup: TA first executes the initialization algorithm and obtains the system public key PK. At the same time, each attribute authority also executes an initialization algorithm to obtain the public key M_j. The public key of the system and the public key of the attribute authority are sent to the attacker Ad by the challenger C.

(3) Phase 1: Attacker Ad sends a challenged attribute split set $A_1^*, ..., A_q^*$ requesting the construction of a private key, which includes the subset of attributes being queried. However, arbitrary A_i^* $(1 \leq i \leq q)$ and access tree structure Λ^* are inconsistent. Challenger C sends the obtained private key A_j, A_j' $(1 \leq j \leq q)$ to attacker Ad after performing the key generation algorithm.

(4) Challenge: Attacker Ad selects two equal length plaintext M_0 and M_1, and sends them to challenger C. Challenger C randomly selects $b \in \{0, 1\}$ and sends the ciphertext CT to Challenger C by encrypting the plaintext.

(5) Phase 2: Phase 1 is repeated.

(6) Guess: Attacker Ad gives a guess value $b^* \in \{0, 1\}$. If $b^* = b$, attacker Ad wins the final game. The advantage of the attacker in this game is:

$$Adv_{Ad}(k) = \left| \Pr[b^* = b] - \frac{1}{2} \right| \tag{2}$$

If the advantage $Adv_{\mathrm{Ad}}(k)$ of attacker Ad is negligible in polynomial time, it indicates that this scheme satisfies CPA security conditions.

5. Hierarchical Authority Based Weighted Attribute Encryption Scheme

(1) TA Setup: TA randomly selects g as a generator at the initialization stage. A bilinear group G_0, whose order is prime p, and satisfies the bilinear map e: $G_0 \times G_0 \to G_1$. All identifiers that represent the AA and user identity are uniformly distributed by the TA. AA is used to verify whether the user is legitimate and the user's identity is highly secure. $AID(AA) = \{P(AA), index(AA), P(AA) \in Z_p\}$ represents the random serial number of TA, and $index(AA) \in Z_p$ represents the random serial number that the TA distributes for AA.

The first $|\Gamma^*|$ elements in Z_p are randomly selected, i.e. $1, 2, ..., |\Gamma^*|$ $(\bmod p)$. Set the recursion depth of the user key structure to 2 for the first time. Then randomly select l_1, l_2, set $L_1 = g^{l_1}, L_2 = g^{l_2}$, $D_1 = g^{\frac{1}{l_1}}, D_2 = g^{\frac{1}{l_2}}$. Finally, choose $\alpha, y, \{\beta_1, \beta_2\} \in Z_p, h \in G_0$ randomly from Z_p, output the system public key as $PK = \{G_0, G_T, g, L_1, L_2, D_1, D_2, e(g, g)^\alpha\}$ and the system master key $MK = \{l_1, l_2, g^\alpha\}$.

(2) AA Setup: After the TA is initialized, a global identifier AID is generated for each AA to indicate the uniqueness of the identity. When the number of AA is N, the authority of each layer distributes keys for users according to the weighted attributes. It is assumed that the attribute set of the kth layer authority is $A_k = \{a_1, ..., a_{n_k}\}$, and the corresponding attribute split set is A_k'. Let $|A'_k| = n_k'$, take the first n_k' elements in Z_p, i.e. $1, ..., n_k' (\mathrm{mod} p)$. Then, randomly select $r_1, ..., r'_{n_k} \in Z_p, 1 \leq k \leq N$, the public key of the AA is:

$$M_j = (e(g, g)^c, g^{r_j})(1 \leq j \leq n_k') \tag{3}$$

1) *Upper-layer AA Authorization*: The system randomly selects parameter c and attribute set $A = \{A_0, A_1, ..., A_n\}$. The first layer attribute is A_0, and $A_i = \{(\lambda_i, 1), ..., (\lambda_i, \omega_i)\}(1 \leq i \leq n)$ is the attribute split set. When TA initializes AA, A is represented by randomly selected $\lambda \in Z_p$, $A_i \in A$ is represented by $\lambda_i \in Z_p$, and $a_{i,j}$ is represented by $\lambda_{i,j} \in Z_p$. The master key of the upper-layer AA is:

$$\begin{aligned} &MK_0 = \{A, g^{\alpha}, K = g^{\frac{\lambda-\alpha}{\beta_1}}, K_{i,j} = g^{\lambda_i} \cdot H(a_{i,j})^{\lambda_{i,j}}, \\ &K'_{i,j} = g^{\frac{1}{\lambda_{i,j}}}(0 \leq i \leq n, 0 \leq j \leq n), D_i = g^{\frac{\lambda-\lambda_i}{\beta_2}}(1 \leq i \leq n)\} \end{aligned} \tag{4}$$

D_i can be converted during the match of attributes to decrypt. When converting, $\frac{D_i}{D'_i}$ can go from λ'_i to λ_i.

2) *Lower-layer AA Authorization*: The upper-layer authority authorizes the authority to enter the system after verifying the identity of the lower-layer authority. Let A denote the attribute set of the upper-layer AA and A' denote the attribute set of the lower-layer AA to satisfy $A' \subset A$. The AA_{k+1}'s master key is distributed through the AA_k. As above, AA_k randomly selects $\lambda' \in Z_p$ for A' and $\lambda'_i \in Z_p$ for $A'_i \in A'$, $\lambda'_{i,j} \in Z_p, a'_{i,j} \in A'_i, 0 \leq i \leq n, 1 \leq j \leq n$. Then AA_{k+1}'s master key is:

$$\begin{aligned} &MK_{k+1} = \{A', g^{\alpha}, K' = K \cdot g^{\frac{\lambda'(\lambda-\alpha)}{\beta_1}}, K'_{i,j} = K_{i,j} \cdot g^{\lambda'_i} \cdot H(a'_{i,j})^{\lambda'_{i,j}}, \\ &K''_{i,j} = K'_{i,j} \cdot g^{\lambda'_{i,j}}(0 \leq i \leq n, 1 \leq j \leq n), D'_i = D_i \cdot g^{\frac{\lambda'+\lambda'_i}{\beta_2}}\} \end{aligned} \tag{5}$$

Among them, K, $K_{i,j}$, D_i are the corresponding items in AA_k.

(3) Key Generation: The user first applies for the private key to the lower layer AA. If the lower layer AA cannot generate the required weight attribute private key component for the user, then the request is handed over to the upper layer AA. If the upper layer AA is still unable to authorize, the user continues to go up. The application is handed over until a level of authority is able to satisfy the user's request.

1) *User System Private Key Generation*: The TA generates a system private key for a user who temporarily joins the system and sends a certificate. After the MK and the user's identity identifier u are entered into the system, TA randomly selects $z_u \in Z_p$. Calculate the public key of the system according to equation (6) and output:

$$PK_u = g^{Z_u} \tag{6}$$

Calculate the user's system private key:

$$SK_u = g^{\alpha} h^{y Z_u} \tag{7}$$

2) *User Private Key Generation*: TA assigns each AA a unique identifier AID and sends a verification code to AA to verify the validity of its certificate. AA first verifies the validity of the user's uid and then accepts the user's request for a private key. When the user has S attribute sets, the split set of weighted attributes is S'. The TA transmits the user system private key SK_u to each AA. Each AA will obtain the certificate $Certificate\,(u)$ submitted by the user and will also receive the verification code sent by the TA. The validity of the user certificate is verified by the verification code. After the verification is passed, it is determined whether the user's attribute exists in the authority, and after the existence is confirmed, the user private key is distributed to the user.

If the kth layer authority of the system directly operates on the user, then the K TAs will jointly manage the user attributes dispersed at each layer. The user attribute split set on the $k \leq K$ layer AA is denoted by $A^k = \{{A_1}^k, {A_2}^k, ..., {A_n}^k\}$. The kth layer authority randomly generates an identifier AID, which is denoted by c. $\tilde{\lambda}^k \in Z_p$, $\tilde{\lambda}_i^k \in Z_p$, $\tilde{\lambda}_{i,j}^k \in Z_p$, $r' \in AID$ is arbitrarily selected by AA, output the user's private key:

$$SK = (g^c, A^k, D = g^{\frac{\tilde{\lambda}^k - r'}{\beta_{1,k}}}, P = g^{\alpha} g^{u/\lambda} g^{\lambda' r'}, \mathrm{P}' = g^{r'}, D_j = g^{\tilde{\lambda}_i^k/\lambda'} \cdot H(a_{i,j}{}^k)^{\tilde{\lambda}_{i,j}^k},$$
$$D'_j = g^{\lambda \cdot \tilde{\lambda}_{i,j}^k} (0 \leq i \leq n, 0 \leq j \leq n)) \tag{8}$$

(4) Encrypt: The encryption algorithm of this scheme divides the encryption process into two parts: online/offline. In this algorithm, user attributes are classified and each attribute has a public key. The intermediate ciphertext is obtained by encryption during the offline phase; after the user attribute set and plaintext are known in the online phase, it quickly generates these intermediate ciphertexts into complete ciphertexts by a simple calculation. The data owner encrypts his data through the encryption algorithm and then uploads the generated ciphertext to the CSP storage. The main encryption algorithms are as follows:

1) *Offline Encryption*: The system public key PK, the attribute authority public key M_j, and the access tree Λ are input into the system, and the node x threshold is represented by k_x. Each node x in the tree corresponds to a polynomial q_x. Starting from the root node R, the relation $d_x = k_x - 1$ is satisfied. If x is a leaf node, $d_x = 0$ is satisfied. Select random number $s \in Z_p$, let $q_R(0) = s$, the other node x set value: $q_x(0) = q_{parent}(index(x))$. Optionally a polynomial q_x whose order is d_x. Y denotes the set of all leaf nodes, Y' denotes the split set of weighted attributes, $attr\,(x)$ denotes $x \in Y$ attributes, and the middle ciphertext is calculated: $IT = (C_0, C_1, C_2, M, \Lambda)$.

$$\begin{cases} M = \prod\limits_{c \in IA} e(g,g)^{\alpha s}, C_0 = g^s \\ C_1 = g^{\lambda' q_x(0)}, C_2 = H(attr(x))^{q_x(0)} \end{cases} \tag{9}$$

2) *Online Encryption*: Input middle ciphertext IT, plaintext message m and user's attribute set $A_k = \{a_1, ..., a_{n_k}\}$ into the system. Calculated according to equation (10), where I_A is the smallest attribute set of the user and the complete ciphertext is output: $CT = (IT, C)$.

$$C = m \prod_{c \in I_A, x_{n_i} \in A} \left(e(g,g)^{\lambda_n}\right)^{x_{n_i}} = m \prod_{c \in I_A, x_{n_i} \in A} e(g,g)^{\lambda_n {x_{n_i}}} = mM \tag{10}$$

(5) Decrypt: The user downloads the file to be accessed from the CSP and then decrypts it with the obtained user private key. The premise of ciphertext decryption is that the identifier AID of the TA in the ciphertext and the private key component of the user are the same, and when the attribute possessed by the user satisfies the structure of the access tree Λ, i.e. $\sum\limits_{p \in \{S' \cap Y'\}} weight(p) \geq t$. Selecting t elements from the set $K = \{S' \cap Y'\}$ decrypts the ciphertext to get the plaintext. otherwise it returns $\perp$. Call $\mathrm{Decrypt}(\mathrm{MK}_{k+1}, SK, \mathrm{CT}) \rightarrow m$, for any node x in the access tree, after executing the recursive algorithm, get a set S_x. When the user's attribute set matches the access structure of the ciphertext in the authority of this layer, an arbitrary value $i \in S$ is selected,

and the recursive function is executed from the root node. Otherwise, the null value is returned. For any node x: If $i \in A_{c\in I_A}$, set $\mathrm{i} = \mathrm{attr(x)}$, calculate:

$$\begin{aligned}\frac{e(C_1(i),D_i)}{e(C_2(i),D'_i)} &= \frac{e(g^{\lambda' q_x(0)},g^{\tilde{\lambda}_i^k/\lambda'}\cdot H(a_{i,j}^k)^{\tilde{\lambda}_{i,j}^k})}{e(H(i)^{q_x(0)},g^{\lambda'\cdot\tilde{\lambda}_{i,j}^k})} = \frac{e(g^{\lambda' q_x(0)},g^{t\tilde{\lambda}_{i,j}^k})\cdot e(g^{\lambda' q_x(0)},g^{\tilde{\lambda}_i^k/\lambda'})}{e(g^{tq_x(0)},g^{\lambda'\cdot\tilde{\lambda}_{i,j}^k})} \\ &= e(g,g)^{\tilde{\lambda}_i^k q_x(0)}\end{aligned} \tag{11}$$

If $i \notin A_{c\in I_A}$, this algorithm has no solution and returns $\perp$. If and only if the user's attribute set $\sum_{c\in I_A} A$ matches the access tree Λ, the equation (12) is calculated by the Lagrange mean value theorem, then the equation (13) is calculated, and the equation (14) is finally calculated, as follows:

$$\begin{cases} e(g,g)^{\tilde{\lambda}_i^k q_x(0)} = e(g,g)^{\tilde{\lambda}_i^k s} \\ e(g,g)^{I_A\tilde{\lambda}_i^k q_x(0)} = e(g,g)^{I_A\tilde{\lambda}_i^k s} \end{cases} \tag{12}$$

$$\begin{aligned}\prod_{c\in I_A}\frac{e(C_0,P)}{e(C_1,\mathrm{P}')} &= \prod_{c\in I_A}\frac{e(g^s,g^\alpha g^{u/\lambda}g^{\lambda' r'})}{e(g^{\lambda' s},g^{r'})} = \prod_{c\in I_A}\frac{e(g^s,g^{\lambda' r'})e(g^s,g^\alpha g^{u/\lambda})}{e(g^{\lambda' s},g^{r'})} \\ &= e(g,g)^{I_A us/\lambda}\prod_{c\in I_A} e(g,g)^{\alpha s}\end{aligned} \tag{13}$$

$$\frac{e(g,g)^{I_A us/\lambda}\prod_{c\in I_A} e(g,g)^{\alpha s}}{e(g,g)^{I_A us/\lambda}} = \prod_{c\in I_A} e(g,g)^{\alpha s} = M \tag{14}$$

Output plaintext: $m = \frac{C}{M}$.

Finally, the user who meets the ciphertext access policy can decrypt the ciphertext CT and get the plaintext m to be accessed; if it is not satisfied, it can not be decrypted, and the file to be accessed cannot be obtained.

6. Security Analysis

6.1. Security Proof

The proof of security of this scheme is based on the difficult problem of DBDH (Decisional Bilinear Diffie-Hellman) [10]. If the attacker's advantage $Adv_{Ad}(k)$ is negligible, it indicates that the solution is to satisfy CPA security. The theorem of CPA security is given below, which is proved by proof by contradiction.

Theorem 2: If the advantage of solving DBDH problem on (G_0, G_1) is negligible, then the DBDH assumption is established on (G_0, G_1), i.e. the scheme is CPA safe under the standard model.

Proof: Use proof by contradiction to prove. Assuming an attacker's advantage $Adv_{Ad}(k)$ is not negligible in polynomial time, the attacker wins. Use the following games to further illustrate:

(1) Initial: Attacker Ad randomly selects an access policy Λ^*.

(2) Setup: Simulator K runs an "Initial" algorithm to obtain the system's public key $PK = \{G_0, G_T, g, L_1, L_2, D_1, D_2, e(g,g)^\alpha\}$ and the system's master key $MK = \{l_1, l_2,$

$g^{\alpha}\}$. Challenger C saves the master key MK itself, and the system public key is sent to attacker Ad.

(3) Phase 1: Attacker Ad sends a private key access request for the partitioned attribute set ${A_1}^*, {A_2}^*, ..., {A_q}^*$. Note that ${A_i}^*(1 < i < q)$ is completely mismatched with the structure Λ^* of the access tree. Perform private key generation algorithm:

$$\begin{aligned} &SK = (g^c, A^k, D = g^{\frac{\tilde{\lambda}^k - r'}{\beta_{1,k}}}, P = g^{\alpha} g^{u/\lambda} g^{\lambda' r'}, \mathrm{P}' = g^{r'}, D_j = g^{\tilde{\lambda}_i^k/\lambda'} \cdot H(a_{i.j}^k)^{\tilde{\lambda}_{i,j}^k}, \\ &{D'}_j = g^{\lambda \cdot \tilde{\lambda}_{i,j}^k} (0 \le i \le n, 0 \le j \le n)) \end{aligned} \tag{15}$$

Send the private key to attacker Ad.

(4) Challenge: Attacker Ad sends two equal length plaintext M_0, M_1 and challenged access structure Λ^* to challenger C. Attacker Ad did not find Challenger C's key during the query phase and the simulator randomly selected a bit attribute $\eta \in \{0, 1\}$. First, calculate C_0, C_1,C_2 and M. Under access structure Λ^*, encrypt the plaintext M_η offline to obtain the middle ciphertext: $IT = (C_0, C_1, C_2, M, \Lambda^*)$. Calculate:

$$\hat{C} = M_\eta \prod_{c \in I_A, x_{n_i} \in A} \left(e(g,g)^{\lambda_n}\right)^{x_{n_i}} = M_\eta \prod_{c \in I_A, x_{n_i} \in A} e(g,g)^{\lambda_n x_{n_i}} = M_\eta M \tag{16}$$

When $b = 0$, define equation (17)

$$Z = e(g,g)^{abc} \tag{17}$$

Let $bc = \theta$, $\hat{C}$ be ciphertext, so:

$$\hat{C} = M\eta \cdot e(g,g)^{a\theta} \tag{18}$$

Otherwise, when $b = 1$, there is:

$$\begin{cases} Z = e(g,g)^z \\ \hat{C} = M\eta \cdot Z = M\eta \cdot e(g,g)^z \end{cases} \tag{19}$$

Since z is randomly selected, $\hat{C}$ obtained at this time is a random element, and $\hat{C}$ does not include information related to M_η.

(5) Phase 2: Attacker Ad repeatedly sends a private key request for user attribute set $A_1^*, A_2^*, ..., A_q^*$, i.e. $A_1^*, A_2^*, ..., A_q^*$ does not meet the access structure Λ^* in the ciphertext.

(6) Guess: Attacker Ad outputs a conjecture on η. If $\eta' = \eta$, the attacker guesses correctly and the simulator outputs $b' = 0$, indicating that the tuple received by the simulator is the DBDH tuple $(g, g^a, g^b, g^c, e(g,g)^{abc})$. Otherwise, the simulator outputs $b' = 1$, which is a five-tuple $(g, g^a, g^b, g^c, e(g,g)^z)$. In the above game, if $b = 1$, the attacker is not receiving any information related to the ciphertext M_η, i.e.

$$\Pr[\eta' \ne \eta \,|b = 1\,] = \frac{1}{2} \tag{20}$$

Because, if $\eta' \ne \eta$, the simulator guesses $b' = 1$, then

$$\Pr[b' = b \,|b = 1\,] = \frac{1}{2} \tag{21}$$

When $b = 0$, the attacker gets a ciphertext M_η of a scheme. By definition, the attacker has a non-negligible advantage ε to break this scheme, so

$$\Pr[\eta' = \eta \,|b = 0] = \varepsilon + \frac{1}{2} \tag{22}$$

Because, if $\eta' = \eta$, the simulator will guess $b' = 0$, so there is

$$\Pr[b' = b \,|b = 0] = \varepsilon + \frac{1}{2} \tag{23}$$

So in the above game, the simulator correctly guesses the advantage of $b' = b$ is:

$$\begin{aligned} &Adv_C = \Pr[b' = b] - \tfrac{1}{2} \\ &= \tfrac{1}{2} \Pr[b' = b \,|b = 1] + \tfrac{1}{2} \Pr[b' = b \,|b = 0] - \tfrac{1}{2} \\ &= \tfrac{1}{2} \cdot \tfrac{1}{2} + \tfrac{1}{2} \cdot (\tfrac{1}{2} + \varepsilon) - \tfrac{1}{2} \\ &= \tfrac{\varepsilon}{2} \end{aligned} \tag{24}$$

Based on the definition and the above security game, in any polynomial-time algorithm, the advantage of attacker Ad winning the game is negligible, so the scheme is CPA safe.

6.2. Security Analysis

(1) Collusion Resistance: When each user is registered, the TA will distribute a unique uid for it. Generate the user's system private key by randomizing parameters:

$$SK_u = g^\alpha h^{yz_u} \tag{25}$$

The TA distributes random AID to each layer of AA. The TA first splits the user's attribute set according to the size of the weight value, and the attribute split set is sent by the TA to the AA of each layer. Before AA distributes the attribute private key for users, it first verifies the validity of each user's AID. If the attribute of the user belongs to the layer AA, the user private key $SK = (g^c, A^k, D = g^{\frac{\tilde{\lambda}^k - r'}{\beta_{1,k}}}, P = g^\alpha g^{u/\lambda} g^{\lambda' r'}, \mathrm{P}' = g^{r'}, D_j = g^{\tilde{\lambda}_i^k/\lambda'} \cdot H(a_{i.j}^k)^{\tilde{\lambda}_{i,j}^k}, D_j' = g^{\lambda \cdot \tilde{\lambda}_{i,j}^k} (0 \le i \le n, 0 \le j \le n))$ is distributed for it. Each user's private key is also formed by the serial number and randomization parameters, so that at the time of decryption, each user's private key is embedded with a different random value [11]. During encryption, the encryption exponent s, the random numbers λ, λ_n and the weighted split set parameter x are all embedded in different ciphertext segments. If multiple users collusion, even if $H(attr(x))^{q_x(0)}$ is recovered, the random numbers of different users cannot recover g^s and $g^{\lambda' q_x(0)}$. Therefore, the middle ciphertext IT cannot be recovered, and the final ciphertext cannot be recovered. In the end, the collusion user could not decrypt the ciphertext and thus well achieved the collusion resistance attack.

(2) Hierarchical Authority Security: When the user decrypts, the TA needs to generate the user's system private key, and the AA of this layer generates the AA private key to jointly constitute the user's private key [25]. If the TA is compromised, the user's system private key may be exposed, but it cannot be decrypted with the system private key. If any AA is compromised, the attribute key may be exposed and the ciphertext cannot be

decrypted accurately. In summary, multi-authority can collectively manage keys, reducing the heavy workload of a single authority, and improve the security of the system and reduce the overall risk.

(3) Access Policy Flexibility: It allows the intersection between user attributes of this scheme and the expression of access policy is also more flexible. The attribute sets of multiple users can also be queried. The entire process is based on the encryption of data by the data owner.

(4) System Scalability: This scheme extends from TA to multi-authority to jointly manage user's keys. This has reduced the workload of the previous single authority and improved the work efficiency. At the same time, the security of the entire system has also been improved.

7. Performance Analysis

7.1. Performance Comparison

The above security proofs show that this scheme is CPA safe. This scheme and existing schemes [6], [23] and [14] are compared and analyzed in terms of function and performance. It shows the results in Table 1.

In Table 1, suppose n is the number of attributes in the access policy. The pairing operation, exponentiation operation of G_1, exponentiation operation of G_2 are denoted by T_{p}, T_{e1}, T_{e2} respectively.

Table 1. Performance Comparison of Related Schemes

Schemes	Data encryption	Data decryption	Multi-authority	Weighted attribute	Hierarchical
[6]	$2nT_{e1}+T_{e2}+T_p$	$3T_{e2}+T_p$	√	×	×
[14]	$(2n+1)T_{e1}+T_{e2}+T_p$	T_{p}	√	×	√
[23]	$nT_{e1}+T_{e2}+T_p$	$2T_{e2}$	√	√	×
Our scheme	nT_{e2}	T_{e2}	√	√	√

From Table 1, we can see that in [6], the use of multi-authority to reduce the pressure on the central authority to manage and distribute private keys is considered. However, in this scheme, the problem of authority hierarchy and weighted attribute is not considered, and more fine-grained access control policies cannot be implemented. Although Ref. [14] introduces a hierarchical approach to solve the hierarchical relationship between multi-authority. However, in this scheme, the attribute sets managed by different levels of authority are equally important, and the status of each attribute in the system is equal. Ref. [23] distinguishes the importance of different attributes by introducing weighted attributes, but the scheme does not consider the hierarchical constraints among the various authorities. This makes it possible for some unreliable authorities to distribute the private key corresponding to the attribute with a relatively large weight, which brings certain security risks to the encryption system. The scheme proposed in this paper, it adopts the hierarchy to solve the hierarchical relationship among multi-authority, and weighted attributes are introduced to distinguish the importance of different attributes, making the actual system more secure and flexible.

From Table 1, it can be seen that the encryption cost in [14] is large. Since the encryption phase of the scheme proposed in this paper is divided into two online and offline processes, most of the tedious calculation processes are placed in the offline stage, which greatly reduces the computational cost of the encryption stage. From the table, it can be seen that the encryption cost of the scheme proposed in this paper is much smaller than the other three schemes. In terms of decryption cost, the cost of [6] has experienced three exponential operations and one pairing operation. Ref. [14] runs a partial decryption algorithm, which requires a pairing operation to recover the plaintext. However, in [23], users need to perform two exponential operations when decrypting. When the scheme proposed in this paper is decrypted, only one exponential operation needs to be performed. Overall, the scheme proposed in this paper is relatively excellent, and it has improved in terms of function and performance, and is more suitable for cloud environments.

7.2. Simulation Evaluation

In order to show the performance of this scheme more intuitively, this paper selects scheme [1] and scheme [8] to carry out simulation experiments, and evaluates the effectiveness of the scheme through experiments. The implementation is on a Linux system with CPU i5-6500 at 3.20GHZ and the memory is 8GB. The implementation adopts a 224-bit elliptic curve group from Pairing-Based Cryptography Library [18] and based on the CP-ABE package. It shows the experimental results in Fig. 3 and Fig. 4.

The complexity of the access policy affects computational and communication overhead. We generate the form of the access strategy $(A_1, A_2, ..., A_n)$ to simulate the worst environment, and $A_{i,i\in(1,n)}$ is an attribute. We set distinct access policies in this form and the number of user's attributes vary from 10 to 50. It must compare all the attributes of user that can know whether his/her attributes satisfy the access policy or not. The time is given in milliseconds.

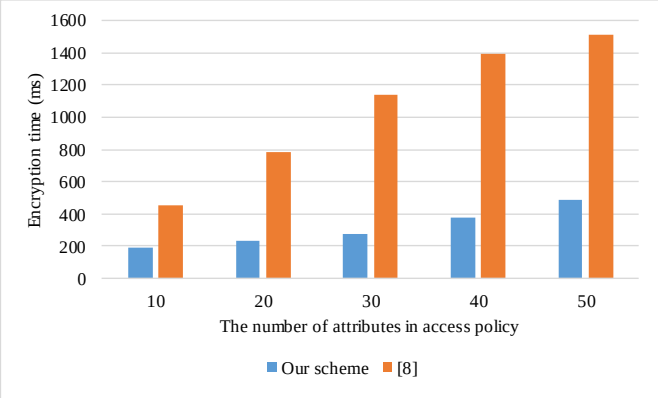

Fig. 3. Comparison of encryption time

Fig. 3 shows the relationship between the encryption time and the number of attributes. The encryption time of the two schemes increases as the number of attributes in-

creases. As can be seen from the figure, the encryption time cost of the proposed scheme is obviously less than that of scheme [8]. This is because the encryption phase of the scheme proposed in this paper is divided into two phases: online and offline. Most of the encryption calculations are completed in the offline phase.

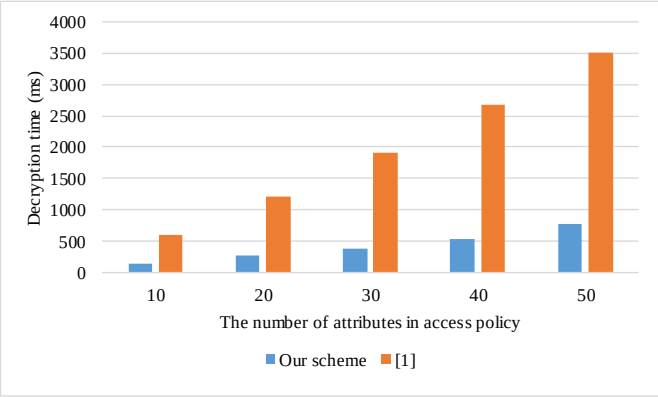

Fig. 4. Comparison of decryption time

As shown in Fig. 4, the comparison of the decryption time consumption for the scheme proposed in this paper and scheme [1]. As can be seen from the figure, the decryption time of the scheme proposed in this paper is less than that of scheme [1], and with the increase of the number of attributes, the advantages of the scheme proposed in this paper become more obvious. This is because in the process of decryption, the amount of calculation of scheme [1] is large, while the scheme proposed in this paper only needs to perform one exponent operation and the efficiency is high.

8. Conclusions

For the problem of cloud storage security, this paper proposes a hierarchical authority based weighted attribute encryption scheme. Through the introduction of the mechanisms of hierarchical authorities and the weight of attribute. There is a hierarchical relationship among the authorities, different attributes have different importance, a more fine-grained access control is achieved, and the security of the system is enhanced. By performing online and offline encryption mechanisms, it places most of the tedious calculation processes on the offline stage, greatly reducing the amount of computation in the encryption stage and reducing the computational cost. Compared with other schemes, the decryption cost is also reduced, which improves the operating efficiency of the system. Through security analysis and comparison, it is shown that the scheme proposed in this paper is safe and efficient under the cloud storage environment.

Acknowledgment. This work has been supported by the National Natural Science Foundation of China (No. 61672338 and No. 61873160).

References

1. Ai, Q., Dan, T., Wang, Z., Jianwei, L.: Hierarchical authority attribute-based encryption. Journal of Wuhan University (Natural Science Edition) 60(5), 441–446 (2014)
2. Beimel, A., Tassa, T., Weinreb, E.: Characterizing ideal weighted threshold secret sharing. In: Theory of Cryptography Conference. pp. 600–619. Springer (2005)
3. Bergen, H.A., Hogan, J.M.: A chosen plaintext attack on an adaptive arithmetic coding compression algorithm. Computers & Security 12(2), 157–167 (1993)
4. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Annual international cryptology conference. pp. 213–229. Springer (2001)
5. Chase, M.: Multi-authority attribute based encryption. In: Theory of Cryptography Conference. pp. 515–534. Springer (2007)
6. Chase, M., Chow, S.S.: Improving privacy and security in multi-authority attribute-based encryption. In: Proceedings of the 16th ACM conference on Computer and communications security. pp. 121–130. ACM (2009)
7. Chattopadhyay, A.K., Nag, A., Majumder, K.: Secure data outsourcing on cloud using secret sharing scheme. IJ Network Security 19(6), 912–921 (2017)
8. Chen, D., Liu, S., Li, Q.: Multi-authority and hierarchical weighted attribute-based encryption scheme. Journal of Nanjing University of Posts and Telecommunications 36(5), 18–23 (2016)
9. Chen, W., Zhang, Y., Zheng, D.: Weighted multi-authority attribute encryption based on ciphertext policy. Journal of GuiLin University of Electronic Technology 35(5), 386–390 (2015)
10. Cheung, L., Newport, C.: Provably secure ciphertext policy abe. In: Proceedings of the 14th ACM conference on Computer and communications security. pp. 456–465. ACM (2007)
11. Cui, M., Han, D., Wang, J.: An efficient and safe road condition monitoring authentication scheme based on fog computing. IEEE Internet of Things Journal (2019)
12. Gorasia, N., Srikanth, R., Doshi, N., Rupareliya, J.: Improving security in multi authority attribute based encryption with fast decryption. Procedia Computer Science 79, 632–639 (2016)
13. Han, D., Bi, K., Xie, B., Huang, L., Wang, R.: An anomaly detection on the application-layer-based qos in the cloud storage system. Comput. Sci. Inf. Syst. 13(2), 659–676 (2016)
14. Huang, Q., Yang, Y., Shen, M.: Secure and efficient data collaboration with hierarchical attribute-based encryption in cloud computing. Future Generation Computer Systems 72, 239–249 (2017)
15. Liu, X., Ma, J., Xiong, J., Li, Q., Ma, J.: Ciphertext-policy weighted attribute based encryption for fine-grained access control. In: 2013 5th International Conference On Intelligent Networking And Collaborative Systems. pp. 51–57. IEEE (2013)
16. Liu, X., Ma, J., Xiong, J., Li, Q., Zhang, T.: Ciphertext-policy weighted attribute based encryption scheme. Journal of Xi'an Jiaotong University 47(8), 44–48 (2013)
17. Luo, E., Liu, Q., Wang, G.: Hierarchical multi-authority and attribute-based encryption friend discovery scheme in mobile social networks. IEEE Communications Letters 20(9), 1772–1775 (2016)
18. Lynn, B., et al.: Pbc library. Online: http://crypto. stanford. edu/pbc 59, 76–99 (2006)
19. Nag, A., Choudhary, S., Dawn, S., Basu, S.: Secure data outsourcing in the cloud using multi-secret sharing scheme (msss). In: Proceedings of the First International Conference on Intelligent Computing and Communication. pp. 337–343. Springer (2017)
20. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 457–473. Springer (2005)
21. Wan, Z., Liu, J., Deng, R.H.: Hasbe: A hierarchical attribute-based solution for flexible and scalable access control in cloud computing. IEEE transactions on information forensics and security 7(2), 743–754 (2012)
22. Wang, G., Liu, Q., Wu, J.: Hierarchical attribute-based encryption for fine-grained access control in cloud storage services. In: Proceedings of the 17th ACM conference on Computer and communications security. pp. 735–737. ACM (2010)

23. Wang, Y., Zhang, D., Zhong, H.: Multi-authority based weighted attribute encryption scheme in cloud computing. In: 2014 10th International Conference on Natural Computation (ICNC). pp. 1033–1038. IEEE (2014)
24. Yang, K., Jia, X., Ren, K., Zhang, B., Xie, R.: Dac-macs: Effective data access control for multiauthority cloud storage systems. IEEE Transactions on Information Forensics and Security 8(11), 1790–1801 (2013)
25. Yang, L., Deng, Y., Yang, L.T., Lin, R.: Reducing the cooling power of data centers by intelligently assigning tasks. IEEE Internet of Things Journal 5(3), 1667–1678 (2018)
26. Zhou, Y., Deng, Y., Xie, J., Yang, L.T.: Epas: A sampling based similarity identification algorithm for the cloud. IEEE Transactions on Cloud Computing 6(3), 720–733 (2018)
27. Zou, X.: A hierarchical attribute-based encryption scheme. Wuhan University Journal of Natural Sciences 18(3), 259–264 (2013)

Qiuting Tian is currently a Ph.D. at Shanghai Maritime University. Her research interests include cloud computing, mobile networking security and cloud security.

Dezhi Han received the Ph.D. degree from the Huazhong University of Science and Technology. He is currently a Professor of computer science and engineering with Shanghai Maritime University. His research interests include cloud computing, mobile networking, wireless communication, and cloud security.

Yanmei Jiang is currently a master student of Shanghai Maritime University. Her research interests include cloud computing and mobile networking security.

Received: September 12, 2018; Accepted: August 8, 2019.