Hindawi
Security and Communication Networks
Volume 2017, Article ID 2508693, 11 pages
https://doi.org/10.1155/2017/2508693

WILEY | Hindawi

Research Article

Revocable Key-Aggregate Cryptosystem for Data Sharing in Cloud

Qingqing Gan, Xiaoming Wang, and Daini Wu

Department of Computer Science, Jinan University, Guangzhou 510632, China

Correspondence should be addressed to Xiaoming Wang; twxm@jnu.edu.cn

Received 19 October 2016; Revised 7 February 2017; Accepted 20 February 2017; Published 12 March 2017

Academic Editor: Bruce M. Kapron

With the rapid development of network and storage technology, cloud storage has become a new service mode, while data sharing and user revocation are important functions in the cloud storage. Therefore, according to the characteristics of cloud storage, a revocable key-aggregate encryption scheme is put forward based on subset-cover framework. The proposed scheme not only has the key-aggregate characteristics, which greatly simplifies the user's key management, but also can revoke user access permissions, realizing the flexible and effective access control. When user revocation occurs, it allows cloud server to update the ciphertext so that revoked users can not have access to the new ciphertext, while nonrevoked users do not need to update their private keys. In addition, a verification mechanism is provided in the proposed scheme, which can verify the updated ciphertext and ensure that the user revocation is performed correctly. Compared with the existing schemes, this scheme can not only reduce the cost of key management and storage, but also realize user revocation and achieve user's access control efficiently. Finally, the proposed scheme can be proved to be selective chosen-plaintext security in the standard model.

1. Introduction

With the continuous development of cloud computing technology, a new kind of data storage model called cloud storage has attracted great attention. Derived from cloud computing, cloud storage can provide online storage space through the network [1]. With the advantage of low cost, easy utilizing, and high scalability, it can meet the needs of the mass data storage and provide data sharing service, which has become the important area in the data storage technology. After requesting the storage service from cloud service providers, enterprises or individuals store a large amount of data to the cloud server, greatly reducing the burden of the local hardware infrastructure and saving the local storage overhead. What is more, its function of data sharing is regarded as very important for multiuser cloud computing environment. When data owners outsource their data in the server and want to share these data with other users, they can adopt techniques to delegate permission to these users. By this way, the legitimate users can have access to corresponding data from the cloud server so as to achieve the process of data sharing.

However, when cloud storage brings great convenience for users dealing with large-scale data, it also brings new security issues and challenges [2]. Because the cloud server is not completely trusted, enterprises or individuals will lose absolute control over the data outsourced to the cloud data, which brings the worries about data security and privacy protection. So for these data, such as how to use encryption scheme to ensure the cloud security and how to protect the data privacy, realize effective data sharing, and reduce the user key management cost as much as possible, key-aggregate cryptosystem is brought forward at this moment. In such cryptosystem, user's private keys can be aggregated together to be a single key and only using the single key can user decrypt the corresponding multiple encrypted files, which simplifies the user's key management. It also grants different decryption access for different users and can be applied to the data sharing in cloud flexibly. Meanwhile, since user's access changed dynamically and frequently in the cloud environment, how to realize user's access control and revocation become vital problems to be solved. For example, when an employee leaves his company, he will no longer have permission to the company's internal data. So, in order to

meet the dynamic change of user access, it is necessary to consider the problem of user revocation.

Therefore, according to the characteristics of cloud storage, the research and establishment of an efficient and secure revocable key-aggregate encryption scheme is very necessary and urgent, which has important theoretical significance and application value.

1.1. Contribution. In order to solve the key management problems and realize dynamic access control during data sharing more effectively, this paper has been focused on the study of revocable key-aggregate cryptosystem in cloud. Its main contribution shows the following:

(1) According to the characteristics of the key-aggregate cryptosystem and the needs for user revocation, this paper first makes formal definition about the revocable key-aggregate cryptosystem.

(2) Combining the subset-cover framework, this paper puts forward an efficient revocable key-aggregate encryption scheme based on multilinear maps, realizing the user's access control and revocation. Our construction not only has the characteristics of key aggregation, which simplifies the user's key management effectively, but also can delegate different users with different decryption permission and achieve revocation of user access rights, realizing the flexible access control effectively.

(3) Compared with the existing schemes, this paper analyzes the related performance for the proposed scheme. It indicates that our scheme not only keeps the users' secret key and the ciphertext in constant-size, but also reduces the length of system parameters to $O(\log N)$, where N is the maximum number of files in the system, thus saving the cost of storage and transmission efficiently. By updating ciphertext via the cloud servers, the proposed scheme realizes the user permissions revocation while legitimate users do not need to update their private keys. What is more, it provides a verification mechanism to ensure user revocation executed correctly.

(4) Lastly, security analysis shows that the proposed scheme is proved to be selective chosen-plaintext security based on Generalized DHDHE assumption in the standard model. In addition, we discuss a solution to extend our basic scheme to solve the rapid growing number of files in the cloud environment.

1.2. Related Works. In recent years, it has become a crucial problem to realize secure and effective data sharing, as well as reducing the key management costs in the cloud environment. How to reduce the number of keys that users have to save, thus simplifying the key management problems effectively, has been a hot research topic. In existing research results, they can mainly be divided into four kinds in reducing the cost of the key management: hierarchical key management scheme, key compression scheme based on symmetric encryption, identity-based key compression scheme, and other related solutions.

In cloud storage, the hierarchical key management scheme generally utilizes tree structure, where the key of each nonleaf node can generate keys of its child nodes. And users only need to save the corresponding ancestor nodes, effectively simplifying the key management. This technology was first proposed by Akl and Taylor [7] and later has been applied to the cloud environment with the rise of cloud computing [8, 9]. For example, Ateniese et al. [10] put forward a predefined hierarchical key management scheme based on the logical key tree. However, the main drawback of hierarchical key management scheme was that only under certain conditions can it achieve effective key compression. This was because the node key can only access to the subtree of the node, if authorized files were from different branches, which in turn would increase the number of users' private keys. So its key compression was limited; only when sharing all the documents from the same branch in the tree, it could achieve the effective compression of private key.

In order to solve the issue that it needs to transport a large number of keys in the broadcast encryption scenario, Benaloh et al. [11] proposed a key compression scheme based on symmetric encryption. Its basic method is to split the entire ciphertext space into finite sets and generate a constant-size key corresponding to each of these sets, so as to realize the effect of key compression. Other schemes such as [12, 13] were also symmetrical encryption schemes trying to reduce the key size. Since these schemes were set in the environment of symmetric encryption, which required to share a symmetric key through secure channel, their application scenarios were greatly limited in the cloud environment.

As Shamir [14] proposed the concept of identity-based encryption (IBE) and then Boneh and Franklin [15] put forward the first practical IBE scheme using bilinear pairings, it brought out the research of identity-based key compression scheme. Guo et al. [16] presented a multi-identity single key decryption scheme and proved its security in the random oracle model. In their scheme, when user adopted different identities as the public key in different scenarios, for example, user had more than one email address, it only needed to store a private key to decrypt multiple encrypted messages from different companies, remarkably cutting down the cost of the user key management. Then [17, 18] made improvements on the efficiency and achieved adaptive chosen-ciphertext security in the standard model. But in these schemes, key compression was restricted, which required all the keys from different identity divisions, and the length of ciphertext and public parameters were linearly related to the maximum number of keys that can be aggregated, which increased the overhead of storage and transmission. Sahai and Waters [19] proposed a fuzzy identity-based encryption (FIBE) scheme to take users' biometric information as their identities, so that user's identity was no longer a single one but was made up of several attributes. It allowed a private key to decrypt multiple ciphertexts and was proved to be secure in the standard model. However, this scheme required the ciphertext to be encrypted by identity that met certain conditions, so it could not achieve the flexible key compression.

Other relevant solutions include the attribute-based encryption (ABE) and proxy reencryption (PRE). Waters [20] presented an ABE scheme that its private key was associated with the strategy, and ciphertext was associated with attributes and could decrypt when strategy matched with attributes. In their scheme, however, the length of private key was linearly related to the leaf nodes in the strategy access tree. Li et al. [21] applied ABE to share keys in group users, but the main concern was to resist collusion attacks, rather than key compression. Canetti and Hohenberger [22] put forward PRE scheme using the thought of transformation to turn the original ciphertext into the ciphertext encrypted by the user's public key. However, such technology is essentially aimed at transferring the secure key storage to the cloud proxy server. In addition, a key management scheme based on secret sharing was proposed in [23], but it was suitable for wireless sensor networks.

Recently, Chu et al. [24] first put forward the concept of key-aggregate cryptosystem (KAC) and constructed the first key-aggregate encryption scheme applied to data sharing in the cloud environment flexibly. The scheme was set in public key cryptosystem and it could aggregate users' private key to be a single one, so that users only stored this aggregated key to decrypt multiple files. Most importantly, its aggregation could be achieved without conditions and kept the length of ciphertext in constant-size. However, the length of system parameters in their scheme was linearly related to the maximum number of files, and it did not provide a specific security proof. Soon afterwards, the thought of key-aggregate cryptosystem was adopted in [25–28], such as Dang et al. [27] who applied the key-aggregate cryptosystem in the wireless sensor network and proposed a fine-grained sharing scheme to the encrypted senor data. Sikhar et al. [3] proposed a dynamic key-aggregate encryption scheme to realize the user revocation. But one of its imitations was that once user revocation occurred, all legitimate users needed to update their private keys, which brought expensive overhead of key update.

1.3. Organization. The rest of the paper is organized as follows: Section 2 introduces some related knowledge, including multilinear maps, complexity assumption, and subset-cover framework. In Section 3 we discuss the definition, the security model, and system model of the revocable key-aggregate cryptosystem. Section 4 details our new construction and Section 5 shows the evaluation of our proposed scheme, containing performance analysis and the security analysis. Then in Section 6, we have some discussions and present an extension for our basic scheme. Finally, we conclude this paper and look forward to the future work in Section 7.

2. Preliminaries

In this section we describe some basic primitives and concepts that are used in our scheme.

2.1. Multilinear Maps. Multilinear maps were first put forward by Boneh and Silverberg [29], making the research and application of multilinear maps be more and more widely. Multilinear maps mainly consist of the following two algorithms:

(1) Setup (n): the Setup algorithm outputs an n-linear map, which contains n groups $G = (G_1, G_2, \ldots, G_n)$ with prime order p and generators $g_i \in G_i$.

(2) $e_{i,j}(g, h)$: the map algorithm takes two elements $g \in G_i$ and $h \in G_j$ as input, while $i + j \le n$, and outputs an element in G_{i+j} satisfying $e_{i,j}(g_i^a, g_j^b) = g_{i+j}^{ab}$. We often leave out the subscripts to be written as e. The generalization of e with multiple inputs can be donated as $e(h^{(1)}, h^{(2)}, \ldots, h^{(k)}) = e(h^{(1)}, e(h^{(2)}, \ldots, h^{(k)}))$.

In the asymmetric multilinear maps [30], group is divided by a vector and the map operations make $G_{\mathbf{v}_1} \times G_{\mathbf{v}_2}$ into $G_{\mathbf{v}_1+\mathbf{v}_2}$. The definition shows the following:

(1) Setup ($\mathbf{n}$): the Setup algorithm takes a positive integer vector $\mathbf{n} \in Z^\lambda$ as input and outputs an $\mathbf{n}$-linear map, which contains a set of groups $\{G_{\mathbf{v}}\}$ with prime order p, and generators $g_{\mathbf{v}} \in G_{\mathbf{v}}$, while $\mathbf{v}$ are nonnegative integer vectors meeting $\mathbf{v} \le \mathbf{n}$. Assume $\mathbf{e}_i$ be the vector with 1 at the position i and 0 at else positions. Then $\{G_{\mathbf{e}_i}\}$ are the source groups, $G_{\mathbf{n}}$ is defined as the target group, and the rest of the groups are intermediate group.

(2) $e_{\mathbf{v}_1,\mathbf{v}_2}(g, h)$: the map algorithm inputs two elements $g \in G_{\mathbf{v}_1}$ and $h \in G_{\mathbf{v}_2}$ with $\mathbf{v}_1 + \mathbf{v}_2 \le \mathbf{n}$ and outputs an element of $G_{\mathbf{v}_1+\mathbf{v}_2}$ such that $e_{\mathbf{v}_1,\mathbf{v}_2}(g_{\mathbf{v}_1}^a, g_{\mathbf{v}_2}^b) = g_{\mathbf{v}_1+\mathbf{v}_2}^{ab}$. Similarly, we leave out the subscripts to be written as e and also generalize e with multiple inputs as $e(h^{(1)}, h^{(2)}, \ldots, h^{(k)}) = e(h^{(1)}, e(h^{(2)}, \ldots, h^{(k)}))$.

2.2. Complexity Assumption. We introduce a new complexity assumption named Generalized DHDHE. This new assumption is the variant version of the well-known Decisional n-Hybrid Diffie-Hellman Exponent (DHDHE) proposed by Boneh et al. [30].

Assumption 1 (Generalized Decisional n-Hybrid Diffie-Hellman Exponent, Generalized DHDHE). Let $\text{params}' \leftarrow \text{Setup}'(2\mathbf{n})$. Choose random $\alpha \in Z_p$; set $X_\ell = g_{\mathbf{e}_\ell}^{\alpha^{(2^\ell)}}$ for $\ell = 0, 1, \ldots, n-1$; and set $X_n = g_{\mathbf{e}_\mathbf{n}}^{\alpha^{(2^n+1)}}$ for $\ell = n$. Randomly select $t_1 \in Z_p$, $t_2 \in Z_p$, while $t = t_1 + t_2$, and let $Y_1 = g_{\mathbf{n}}^{t_1}$, $Y_2 = g_{\mathbf{n}}^{t_2}$. Given $\langle \{X_i\}_{i\in\{0,1,\ldots,n\}}, Y_1, Y_2, K\rangle$, the goal is to distinguish $K = g_{2\mathbf{n}}^{t\alpha^{(2^n)}}$ from a random element in $G_{2\mathbf{n}}$.

For a polynomial-time adversary A, its advantages to Generalized DHDHE problem are defined as

$$\begin{aligned}\text{Adv}_A^{\text{Weak DHDHE}} &= \left|\Pr\left[A\left(\{X_i\}_{i\in\{0,1,\ldots,n\}}, Y_1, Y_2, K = g_{2\mathbf{n}}^{t\alpha^{(2^n)}}\right) = 1\right]\right. \\ &\left. - \Pr\left[A\left(\{X_i\}_{i\in\{0,1,\ldots,n\}}, Y_1, Y_2, K = g_{2\mathbf{n}}^{r}\right) = 1\right]\right|. \end{aligned} \quad (1)$$

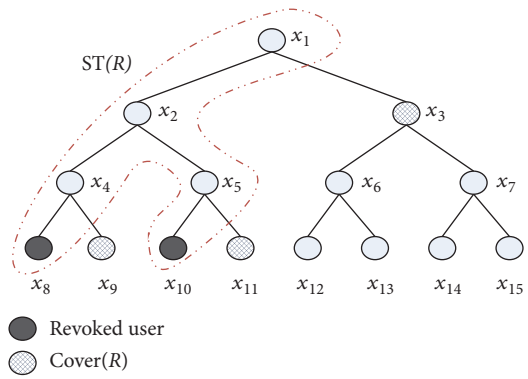

FIGURE 1: Subset-cover framework: Complete Subtree method.

From here we can see that this new assumption is the generalization of DHDHE assumption. Specifically, if we multiply Y_1 and Y_2, Generalized DHDHE assumption can be reduced to DHDHE assumption in [30].

Definition 2. We say the Generalized DHDHE assumption holds if, for any polynomial-time adversary A, A has a negligible advantage in solving the Generalized DHDHE problem.

2.3. Subset-Cover Framework. Naor et al. [4] first proposed the subset-cover framework and applied it to the broadcast encryption scheme, realizing the dynamic authorization of the user. The subset-cover framework includes complete subtree (CS) method and subset difference (SD) method. This paper mainly introduces CS method, shown as follows.

Let T be a full binary tree with depth d. Thus the number of leaf nodes in the tree is $(2^d - 1)$, representing $(2^d - 1)$ users. First, for each user u, we define a path set denoted by path(u), containing all the nodes passing through the root node to leaf node. When given a user revocation set R, let $S_{i_1}, S_{i_2}, \ldots, S_{i_m}$ be the complete subtrees in T rooted at the nodes of outdegree one in Steiner Tree ST(R), and $S_{i_1}, S_{i_2}, \ldots, S_{i_m}$ are not in the ST(R). We said that $S_{i_1}, S_{i_2}, \ldots, S_{i_m}$ cover all the nonrevoked nodes in T, denoted by cover(R). Take the example in Figure 1. Given the full binary tree T with eight leaf nodes, we get the user sets $U = \{x_8, x_9, \ldots, x_{15}\}$. Then the path set for each user can be obtained as path$(x_8) = \{x_1, x_2, x_4, x_8\}$, path$(x_{12}) = \{x_1, x_3, x_6, x_{12}\}$, and so on. Suppose the user revocation set $R = \{x_8, x_{10}\}$; then ST(R) is shown in the dotted box in Figure 1, so that cover(R) $= \{x_3, x_9, x_{11}\}$ including all the nonrevoked users.

When constructing the scheme based on the subset-cover framework, the path set is embedded in private key, while the cover set is related to the ciphertext. If and only if path(u) $\cap$ cover(R) $\neq \phi$, the user u can take the next step to the decryption. In the CS method as shown in Figure 1, only legitimate users, such as x_9, x_{12}, meet the conditions. For revoked user u, since path(u) $\cap$ cover(R) $= \phi$, then he is unable to complete the decryption, as x_8 in Figure 1.

3. Revocable Key-Aggregate Cryptosystem

Since the delegated users in cloud have the feature of dynamic change, revocable key-aggregate cryptosystem is essential for consummating the user revocation function in KAC.

3.1. Definition. Revocable key-aggregate cryptosystem (RKAC) is an extension of KAC such that a user can be revoked if his credential is expired. A revocable key-aggregate encryption scheme consists of seven polynomial-time algorithms as Setup, KeyGen, Encrypt, Extract, Update, Decrypt, and Verify, which are defined as follows:

(1) Setup($1^\lambda, n$): the Setup algorithm takes as input a security parameter 1^λ and the maximum number of files n. It outputs public parameters params.

(2) KeyGen(params): the key generation algorithm takes as input public parameters params. It generates a public key PK and a master secret key msk.

(3) Encrypt(PK, i, m, params): the encryption algorithm takes as input public key PK, an index i denoting the file, a message m, and public parameters params. It outputs a ciphertext C.

(4) Extract(msk, uid, S, params): the Extract algorithm takes as input the master secret key msk and a set S of indices corresponding to different files, user identity uid, and public parameters params. It outputs users' private key SK.

(5) Update(PK, R, C, params): the update algorithm takes as input the public key PK, the user revocation set R, a ciphertext C, and public parameters params. It outputs an updated ciphertext C'.

(6) Decrypt(C, SK, S, i, R, params): the decryption algorithm takes as input a ciphertext C, user private key SK, the set S, an index i denoting the ciphertext C, the user revocation set R, and public parameters params. If $(i \in S) \wedge (\text{uid} \notin R)$, it outputs the result m or else outputs $\perp$.

(7) Verify(C, C', PK, params): the Verify algorithm takes as input a ciphertext C, an updated ciphertext C', public key PK, and public parameters params. If the cloud server has executed the revocation honestly and updated the ciphertext correctly, it outputs 1 or else outputs 0.

3.2. Security Model. For RKAC, we present its security model through the game between a challenger Chal and a polynomial-time adversary A. The selective security property of RKAC under indistinguishable chosen-ciphertext attack (IND-CCA) is defined as follows.

Init. A initially submits a challenge file index i^* and a revoked identity set R^*.

Setup. Chal generates public parameters params and (PK, msk) by running Setup($1^\lambda, n$) and KeyGen(params). It keeps msk secretly to itself and gives params and PK to A.

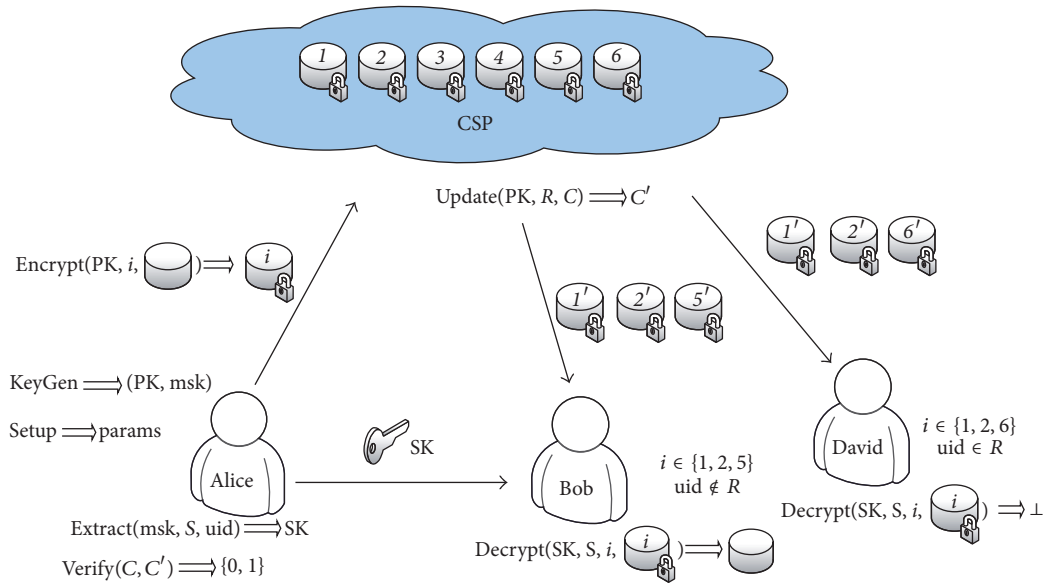

Figure 2: The model for RKAC.

Phase 1. A adaptively requests a series of queries. These queries are processed as follows:

(i) Step 1 (extraction query): for any file index set $S(i^* \notin S)$ and identity uid(uid $\in R^*$), Chal invokes the Extract algorithm Extract(msk, uid, S, params) and sends the generated private key $\mathrm{SK} = (K_S, K_{\mathrm{uid}})$ to A.

(ii) Step 2 (decryption query): for any ciphertext C_i, file index set $S(i^* \notin S)$, and identity uid(uid $\in R^*$), the challenger Chal executes the decryption algorithm Decrypt(C_i, SK, S, i, R, params) and sends the obtained plaintext to A.

Challenge. Once the adversary A decides to end Phase 1, it submits two challenge messages $m_0, m_1 \in M$ with equal length. Chal flips a random coin $b \in \{0, 1\}$ and sets $C = \mathrm{Encrypt}(\mathrm{PK}, i^*, m_b, \mathrm{params})$, $C^* = \mathrm{Update}(C, R^*)$ and then gives the challenge ciphertext C^* to A.

Phase 2. A continues to request a series of adaptive queries, but with the restrictions that it cannot perform the decryption query to C^*. The challenger Chal adopts the same method as in Phase 1 to answer the queries.

Guess. Finally, A outputs a guess $b' \in \{0, 1\}$ and wins the game if $b' = b$.

The acquired advantage of the adversary A for the RKAC scheme is defined as $\mathrm{Adv}_A^{\mathrm{RKAC}} = |\Pr[b' = b] - 1/2|$.

Definition 3. If, for any polynomial-time t and adversary A through q queries in the above game, its advantage for RKAC scheme $\mathrm{Adv}_A^{\mathrm{RKAC}} \leq \varepsilon$, one said this RKAC scheme is selective (t, q, ε)-IND-CCA security.

Definition 4. If, for any polynomial-time t and adversary A through q queries in the above game without the decryption query, its advantage for RKAC scheme $\mathrm{Adv}_A^{\mathrm{RKAC}} \leq \varepsilon$, one said this RKAC scheme is selective (t, q, ε)-IND-CPA security.

3.3. System Model. Applying the RKAC in a cloud environment, the model is shown in Figure 2. It consists of three entities: cloud service provider (CSP), the data owner (DO), and user.

When the data owner Alice wants to share multiple files $m_1, m_2, \ldots, m_n$with others through the cloud server utilizing revocable key-aggregate encryption scheme, Alice first runs Setup algorithm to get the system parameters params. Then Alice executes KeyGen(params) to get a random public/master secret key-pair (PK, msk) and kept msk secretly. After that, Alice and anyone who cooperated with Alice can run the encryption algorithm Encrypt(PK, i, R, m, params) and upload the encrypted files to the cloud server. Once Alice hopes to share several of these files to user Bob, Alice will run the algorithm Extract(msk, uid, S, params) to generate a private key SK for Bob according to authorized files' indices and the user's identity. Since SK is a fixed size, it is easy for Alice to pass SK to Bob through safe channel with small communication cost. Whenever Alice wants to revoke users, Alice will send the user revocation list R to CSP. Then CSP calls the algorithm Update(PK, R, C, params) to update the corresponding ciphertext. If and only if Bob has not been revoked, Bob downloads the updated ciphertext from the cloud server and runs the algorithm Decrypt(C, SK, S, i, R, params) with the use of the private key to obtain plaintext. And if the user has been revoked, such as David in Figure 2, he will not be able to decrypt the updated ciphertext, thus withdrawing David's permission to the files. Finally, by invoking the algorithm Verify(C, C', params), Alice can achieve the verification of the updated ciphertext, to ensure that the user revocation is effectively implemented.

4. Main Construction

Our main construction of the revocable key-aggregate encryption scheme is based on multilinear maps and realizes data sharing and user revocation in cloud storage securely and efficiently.

4.1. Basic Idea. In KAC, the aggregation of file indices is embedded in the user's private key so that authorized users store the aggregate key to realize the access to multiple files. However, the access of user in system is changed dynamically, requiring KAC to support user revocation. Therefore, in order to construct a revocable key-aggregate encryption scheme, two mainly challenges are remained to be solved. One is how to construct an efficient scheme with key-aggregate function, the other is how to realize revoking users securely while not affecting the legitimate users' access to files.

For the first challenge, we are inspired by Boneh et al.'s broadcast encryption [30]. Based on this scheme, we try to construct a key-aggregate scheme to keep the users' secret key and the ciphertext in constant-size. With the multilinear maps, it can reduces the length of system parameters to $O(\log N)$, thus saving the cost of storage and transmission efficiently.

For the second challenge, our inspiration comes from Shi et al. [6] revocable key-policy ABE scheme. The scheme not only realizes the direct user revocation, but also achieves the function of ciphertext delegation by a third-party server. What is more, it provides a verification mechanism to ensure the correctness of the ciphertext delegation, which has been of great significance. However, in their scheme, the user private key is related to the access structure and path set in subset-cover framework. Besides that, Shi et al. [6] scheme is only proved to be secure under the random oracle model. So we try to combine Naor et al. [4] subset-cover framework with our scheme for user revocation. In addition, we make improvement of the complete subtree method of subset-cover framework in [4] to aggregate the path set for each user as private key, so as to realize the user's key aggregation and simplify the key management effectively.

Therefore, this paper proposes a revocable key-aggregate encryption scheme and proves its security in the standard model. The main thought of the scheme lies in constructing the ciphertext and the private key. The ciphertext of the new scheme includes not only the file index, but also the user revocation set, realizing the user revocable directly. At the same time, the private key is correspondingly divided into two parts. One is the aggregation of the file index set, and the other is the aggregation of the path set for each user, so as to realize the user's key aggregation effectively. Through the above method, only the legitimate users have access to the appropriate file, realizing the file access control function in the system effectively. This new scheme achieves the ciphertext updating through the cloud servers to save the computational overhead of data owner; when the user revocation occurs, nonrevoked user does not need to update his private key, greatly reducing the key update expensive cost and the burden of key delegate authority; because the cloud server is not completely trusted, we consider to provide a verification mechanism for the scheme, so that the data owner can validate the updated ciphertext to make sure the user revocation is carried out correctly.

4.2. Scheme Design. Let Setup$'$ be the Setup algorithm for a multilinear map, where outputs group with order p, respectively. Let T be a full binary tree with depth d ($1 \le d \le n$), where the leaf stands for user. Number all the nodes in T from one to $(2^d - 1)$; then our scheme consists of the following algorithms:

(1) Setup$(1^\lambda, n)$: take as input the length n of index. Let $\{0,1\}^n \setminus \{0^n\}$ be the index space. Therefore the maximum number of files in the system is $N = 2^n - 1$. Let $\mathbf{n}$ be the all-ones vector with length $(n+1)$. Run Setup$'(2\mathbf{n})$ to obtain the public parameters params$'$ for a multilinear map of target group $G_{2\mathbf{n}}$. Select a random $\alpha \in Z_p$, and set $X_\ell = g_{\mathbf{e}_\ell}^{\alpha^{(2^\ell)}}$ for $\ell = 0, 1, \ldots, n-1$, and set $X_n = g_{\mathbf{e}_\mathbf{n}}^{\alpha^{(2^n+1)}}$ for $\ell = n$. Lastly, let $W = g_{2\mathbf{n}}^{2\alpha^{(2^n)}}$. It outputs the public parameters params $= \langle \text{params}', \{X_i\}_{i\in\{0,1,\ldots,n\}}, W\rangle$.

(2) KeyGen(params): choose a random $\beta \in Z_p$, $\gamma \in Z_p$ and compute $\mu = g_\mathbf{n}{}^\beta$, $\nu = g_\mathbf{n}{}^\gamma$, output PK $= (\mu, \nu)$, msk $= (\beta, \gamma)$.

(3) Encrypt(PK, i, m, params): for a message m and an index $i \in \{1, 2, \ldots, 2^n - 1\}$, randomly pick $t_1 \in Z_p$ and compute $K = W^{t_1} = g_{2\mathbf{n}}^{t_1\alpha^{(2^n)}}$. The ciphertext is created as

$$C = \langle c_1, c_2, c_3\rangle = \left\langle g_\mathbf{n}^{t_1}, \left(\nu g_\mathbf{n}^{\alpha^i}\right)^{t_1}, m \cdot W^{t_1}\right\rangle. \tag{2}$$

(4) Extract(msk, uid, S, params): given the user identity uid $\in \{0,1\}^d$, make use of the CS method in the full binary tree T to get the user's path path(uid) $= \{y_{u_0}, \ldots, y_{u_d}\}$, such that $y_{u_0} =$ root and $y_{u_d} =$ uid. Compute $P_{\text{uid}} = \prod_{y\in\text{path(uid)}} Z_{2^n-y}$; then the path aggregate key $K_{\text{uid}} = P_{\text{uid}}{}^\beta$. For the set $S \subseteq \{1, 2, \ldots, 2^n - 1\}$, the index aggregate key is computed as $K_S = \prod_{j\in S} Z_{2^n-j}^\gamma$. Since S does not include 0, $Z_{2^n-j} = g_\mathbf{n}{}^{\alpha^{2^n-j}}$ can always be retrieved from params. The user's private key is set to SK $= (K_S, K_{\text{uid}})$.

(5) Update(PK, R, C, params): for user revocation set R, compute cover(R) according to the CS method in subset-cover framework. For $x \in \text{cover}(R)$, compute $P_x = g_\mathbf{n}^{\alpha^x}$. Choose a random $t_2 \in Z_p$; then get $c_3' = c_3 \cdot W^{t_2} = m \cdot W^{t_1+t_2}$. Suppose that $t = t_1 + t_2$; then we have $c_3' = m \cdot W^t$. Compute $c_4 = g_\mathbf{n}^{t_2}$, $c_5 = \{(\mu P_x)^{t_2}\}_{x\in\text{cover}(R)}$. Finally get the updated ciphertext as follows:

$$\begin{aligned} C' &= \langle c_1, c_2, c_3', c_4, c_5\rangle \\ &= \left\langle g_\mathbf{n}^{t_1}, \left(\nu g_\mathbf{n}^{\alpha^i}\right)^{t_1}, m \cdot W^t, g_\mathbf{n}^{t_2}, \left\{(\mu P_x)^{t_2}\right\}_{x\in\text{cover}(R)}\right\rangle. \end{aligned} \tag{3}$$

(6) Decrypt(C, SK, S, i, R, params): when user receives the ciphertext C with the index i, if either the index $i \notin S$ or the user's identity uid $\in R$, then return $\perp$. Otherwise, for $x = \text{path(uid)} \cap \text{cover}(R)$, decryption can be done as follows:

$$m = c_3 \cdot \frac{e\left(K_S \cdot \prod_{j\in S,\, j\neq i} Z_{2^n-j+i}, c_1\right)}{e\left(\prod_{j\in S} Z_{2^n-j}, c_2\right)} \cdot \frac{e\left(K_{\text{uid}} \cdot \prod_{y\in \text{path(uid)}, y\neq x} Z_{2^n-y+x}, c_4\right)}{e\left(P_{\text{uid}}, c_5\right)}. \tag{4}$$

(7) Verify(C, C', PK, params): to verify whether the cloud server has executed the revocation correctly and honestly, the equation $e(\mu P_x, c_4) \stackrel{?}{=} e(c_5, g_{\mathbf{n}})$can be used and it returns 0 or 1. For data owner, in order to verify whether the updated ciphertext c_3' is right or not, he can use the equation $e(c_3'/c_3, g_{\mathbf{n}}) \stackrel{?}{=} e(W, c_4)$. If returning 1, it means right or else means wrong.

For correctness, we can see that

$$\begin{aligned}
&c_3 \cdot \frac{e\left(K_S \cdot \prod_{j\in S, j\neq i} Z_{2^n-j+i}, c_1\right)}{e\left(\prod_{j\in S} Z_{2^n-j}, c_2\right)} \cdot \frac{e\left(K_{\text{uid}} \cdot \prod_{y\in \text{path(uid)}, y\neq x} Z_{2^n-y+x}, c_4\right)}{e\left(P_{\text{uid}}, c_5\right)} = c_3 \\
&\cdot \frac{e\left(\prod_{j\in S} Z_{2^n-j}^{\gamma} \cdot \prod_{j\in S, j\neq i} Z_{2^n-j+i}, g_{\mathbf{n}}^{t_1}\right)}{e\left(\prod_{j\in S} Z_{2^n-j}, \left(\nu g_{\mathbf{n}}^{\alpha^i}\right)^{t_1}\right)} \cdot \frac{e\left(P_{\text{uid}}{}^{\beta} \cdot \prod_{y\in \text{path(uid)}, y\neq x} Z_{2^n-y+x}, g_{\mathbf{n}}^{t_2}\right)}{e\left(P_{\text{uid}}, \left(\mu g_{\mathbf{n}}^{\alpha^x}\right)^{t_2}\right)} = c_3 \\
&\cdot \frac{e\left(\prod_{j\in S} Z_{2^n-j}^{\gamma}, g_{\mathbf{n}}^{t_1}\right)}{e\left(\prod_{j\in S} Z_{2^n-j}, \nu^{t_1}\right)} \cdot \frac{e\left(\prod_{j\in S, j\neq i} Z_{2^n-j+i}, g_{\mathbf{n}}^{t_1}\right)}{e\left(\prod_{j\in S} Z_{2^n-j}, \left(g_{\mathbf{n}}^{\alpha^i}\right)^{t_1}\right)} \\
&\cdot \frac{e\left(P_{\text{uid}}{}^{\beta}, g_{\mathbf{n}}^{t_2}\right)}{e\left(P_{\text{uid}}, \mu^{t_2}\right)} \cdot \frac{e\left(\prod_{y\in \text{path(uid)}, y\neq x} Z_{2^n-y+x}, g_{\mathbf{n}}^{t_2}\right)}{e\left(\prod_{y\in \text{path(uid)}} Z_{2^n-j}, \left(g_{\mathbf{n}}^{\alpha^x}\right)^{t_2}\right)} \\
&= c_3 \cdot \frac{e\left(\prod_{j\in S, j\neq i} Z_{2^n-j+i}, g_{\mathbf{n}}^{t_1}\right)}{e\left(\prod_{j\in S} Z_{2^n-j}, \left(g_{\mathbf{n}}^{\alpha^i}\right)^{t_1}\right)} \cdot \frac{e\left(\prod_{y\in \text{path(uid)}, y\neq x} Z_{2^n-y+x}, g_{\mathbf{n}}^{t_2}\right)}{e\left(\prod_{y\in \text{path(uid)}} Z_{2^n-j}, \left(g_{\mathbf{n}}^{\alpha^x}\right)^{t_2}\right)} = m \cdot W^t \\
&\cdot \frac{e\left(g_{\mathbf{n}}^{\sum_{j\in S, j\neq i} \alpha^{(2^n-j+i)}}, g_{\mathbf{n}}^{t_1}\right)}{e\left(g_{\mathbf{n}}^{\sum_{j\in S} \alpha^{(2^n-j)}}, \left(g_{\mathbf{n}}^{\alpha^i}\right)^{t_1}\right)} \cdot \frac{e\left(g_{\mathbf{n}}^{\sum_{y\in \text{path(uid)}, y\neq x} \alpha^{(2^n-y+x)}}, g_{\mathbf{n}}^{t_2}\right)}{e\left(g_{\mathbf{n}}^{\sum_{y\in \text{path(uid)}} \alpha^{(2^n-y)}}, \left(g_{\mathbf{n}}^{\alpha^x}\right)^{t_2}\right)} = \frac{m \cdot W^t}{\left(g_{2\mathbf{n}}^{\alpha^{2^n}}\right)^{t_1+t_2}} \\
&= \frac{m \cdot W^t}{\left(g_{2\mathbf{n}}^{\alpha^{2^n}}\right)^{t}} = m.
\end{aligned} \tag{5}$$

5. Evaluation

In this section, we evaluate the proposed scheme in two aspects, performance analysis and security analysis.

5.1. Performance Analysis. Performance analysis mainly includes the cost of computation, storage, and communication by comparison with several related schemes. In computation, since our scheme is based on asymmetric multilinear maps, $W = g_{2\mathbf{n}}^{\alpha^{(2^n)}}$ in the ciphertext is system parameter, and the value of $g_{\mathbf{n}}$, $g_{\mathbf{n}}^{\alpha^i}$ and P_x can be calculated in advance. Therefore, multilinear mapping operation in the process of encryption does not exist, which reduces the computational cost greatly. Decryption cost is linearly related to user's authorized file index set and path set in the complete subtree. In terms of storage and communication cost, this paper will compare the new scheme with [3–6], including the length of system public parameters, the length of private key, length of ciphertext, revocation manners and costs, and whether it is able to verify the correctness of revocation, as shown in Table 1.

Note that the length of ciphertext refers to the length of original ciphertext when no user has been revoked, and the revocation cost refers to the computational cost when the user revocation occurs. N stands for the maximum number of encrypted files in the system, r denotes the number of revocation users, U represents the number of legitimate users, ℓ is number for leaf node in the access tree corresponding to the user's private key, and s is on behalf of the number of attributes.

As can be seen from Table 1, the proposed scheme not only keeps the length for user's private key as $O(1)$, but also keeps the length of the ciphertext as $O(1)$, which is as well as [3, 5] and better than [4, 6]. But the length of system parameters in [3, 5] is $O(N)$, while that in the proposed scheme is $O(\log N)$. Revocation manners contain direct and indirect revocation. Direct revocation refers that the revocation list is directly embedded in the ciphertext, so that revoked users cannot decrypt any more, such as our scheme and [4, 6]; indirect revocation refers that the authorized agency or data owner distributes the updated keys for the nonrevoked users so as to realize the user revocation, such as [3, 5]. As for the revocation cost, because the indirect revocation needs to distribute updated keys to all legitimate users, computational cost for revocation is $O(U)$, while the revocation cost in our scheme and [4, 6] is mainly focused on the ciphertext update as $O(r \log U)$. In addition, our scheme and [6] also provide verification mechanism, allowing the

Table 1: Comparison with related schemes.

Scheme	System parameter	Private key	Ciphertext	Direct revocation	Revocation cost	Verifiability
[3]	$O(N)$	$O(1)$	$O(1)$	×	$O(U)$	×
[4]	$O(N)$	$O(\log N)$	$O(1)$	√	$O(r \log U)$	×
[5]	$O(N)$	$O(1)$	$O(1)$	×	$O(U)$	×
[6]	$O(N)$	$O(\ell)$	$O(s)$	√	$O(r \log U)$	√
Our scheme	$O(\log N)$	$O(1)$	$O(1)$	√	$O(r \log U)$	√

data owner and any trusted third-party auditor to verify the updated ciphertext, so as to ensure effective implementation of revocation, which is better than [3–5]. Above all, the proposed scheme is superior to [3–6], with less cost of storage and communication, and has ciphertext verifiability function.

5.2. Security Analysis. Our scheme is based on Generalized DHDHE assumption and is proved to be adaptive IND-CPA security under the standard model. First we analyze Generalized DHDHE assumption. Let $X_\ell = g_{\mathbf{e}_\ell}^{\alpha^{(2^\ell)}}$ for $\ell = 0, 1, \ldots, n-1$, so as $j \in [0, 2^n - 1]$, $Z_j = g_{\mathbf{n}}^{\alpha^j}$ can be directly calculated. And given $X_n = g_{\mathbf{e}_n}^{\alpha^{(2^{n+1})}}$, when $j \in [2^n + 1, 2^{n+1}]$, Z_j can be computed out. However, from X_i, Y_1, and Y_2, it is difficult to compute $K = g_{2\mathbf{n}}^{t\alpha^{(2^n)}}$. The reason is that only the random t is related to Y_1 and Y_2. In order to obtain K, we first need to multiply Y_1 and Y_2, and let the multiplication results do the match operation with $g_{\mathbf{n}}^{\alpha^{(2^n)}}$. In other words, it is necessary to calculate $g_{\mathbf{n}}^{\alpha^{(2^n)}}$ from X_ℓ. Since $\mathbf{n}$ is a $(n+1)$-dimensional vector composed of 1, any X_ℓ cannot match with itself, which means that we can only compute $g_{\mathbf{n}}^{\alpha^{(2^n)}}$ in the form of $e(\mathrm{X}_0^{s_0}, \mathrm{X}_1^{s_1}, \ldots, \mathrm{X}_n^{s_n})$ for $s_\ell \in \{0, 1\}$ and $\mathrm{X}_\ell^0 = g_{\mathbf{e}_\ell}$. Notice that the index of the given $X_n = g_{\mathbf{e}_n}^{\alpha^{(2^{n+1})}}$ is greater than $\alpha^{(2^n)}$, so to calculate $g_{\mathbf{n}}^{\alpha^{(2^n)}}$, it should meet $s_n = 0$ in $e(\mathrm{X}_0^{s_0}, \mathrm{X}_1^{s_1}, \ldots, \mathrm{X}_n^{s_n})$. So, $g_{\mathbf{n}}^{\prod_{\ell \in L} \alpha^{(2^\ell)}}$ can be obtained for $\mathrm{L} \subseteq [0, n-1]$. However, for all the subsets of $\mathrm{L} \subseteq [0, n-1]$, there are $\sum_{\ell \in L} 2^\ell < 2^n$. Therefore it is unable to calculate $g_{\mathbf{n}}^{\alpha^{(2^n)}}$; it also means that assumption is difficult.

By the following theorem, we prove the security of the proposed scheme.

Theorem 5. *If the Generalized DHDHE problem is hard to solve, then the proposed revocable key-aggregate encryption scheme is selective IND-CPA security.*

Proof. Assume there exists a polynomial-time adversary A who can break the selective IND-CPA security of the revocable key-aggregate encryption scheme; then a challenger Chal can use the adversary's ability to construct an algorithm B to solve the Generalized DHDHE problem. It is contradictory to our assumption that Generalized DHDHE problem is difficult to solve, thus proving the proposed scheme is selective IND-CPA security.

Suppose, in an asymmetric multilinear maps group system, B is given an instance of the Generalized DHDHE problem $(\text{params}', \{X_i\}_{i \in \{0,1,\ldots,n\}}, Y_1, Y_2, K)$ as follows:

(1) $\text{params}' \leftarrow \text{Setup}'(2\mathbf{n})$, where n is the all-ones vector of length $(n+1)$.

(2) For $\alpha \in_R Z_p$, let $X_\ell = g_{\mathbf{e}_\ell}^{\alpha^{(2^\ell)}}$ for $\ell = 0, 1, \ldots, n-1$; and let $X_n = g_{\mathbf{e}_n}^{\alpha^{(2^{n+1})}}$ for $\ell = n$.

(3) For $t_1 \in_R Z_p$, $t_2 \in_R Z_p$, $t = t_1 + t_2$, let $Y_1 = g_{\mathbf{n}}^{t_1}$, $Y_2 = g_{\mathbf{n}}^{t_2}$.

(4) $K = g_{2\mathbf{n}}^{t\alpha^{(2^n)}}$ or K is a random group element in $G_{2\mathbf{n}}$.

Algorithm B decides whether $K = g_{2\mathbf{n}}^{t\alpha^{(2^n)}}$; if it holds, it outputs 1 or else outputs 0. Algorithm B proceeds the following game with the adversary A. □

Init. Algorithm B initials a full binary tree T of depth d $(1 \le d \le n)$, and all the node in T is numbered from 1 to $(2^d - 1)$. A submits an index i^*, R^* that A will challenge.

Setup. Algorithm B performs the following operations:

(i) Step 1: it chooses a random $r \in_R Z_p$ and sets $v = g_{\mathbf{n}}{}^r / Z_{i^*}$, of which Z_{i^*} can be calculated by X_ℓ. Therefore, $\gamma = r - \alpha^{i^*}$. Since r is randomly selected in Z_p, it is independent with α. Then, according to the principle of subset-cover framework, cover(R^*) can be obtained from R^*. For any $x_k^* \in \text{cover}(R^*)$, it chooses a random $\varphi_k \in Z_p$ and sets $\mu = g_{\mathbf{n}}{}^{\varphi_k} / Z_{x_k^*}$, of which $\mathrm{Z}_{x_k^*}$ can be calculated by X_ℓ. Therefore, $\beta = \varphi_k - \alpha^{x_k^*}$. As φ_k is randomly selected in Z_p, it is independent with α. The public key is set as $\mathrm{PK} = (\mu, v)$; note that algorithm B does not know the master secret key(β, γ).

(ii) Step 2: it computes $W' = e(g_{\mathbf{e}_0}, g_{\mathbf{e}_1}, \ldots, g_{\mathbf{e}_{n-2}}, X_{n-1}, g_{\mathbf{e}_n})$; then $W = e(W', W') = g_{2\mathbf{n}}^{\alpha^{2^n}}$.

(iii) Step 3: it sends the public parameters $(\{X_i\}_{i \in \{0,1,\ldots,n\}}, W)$ and the public key PK to the adversary A.

Phase 1. A is allowed to query for private keys in this stage. For set S in condition that $i^* \notin S$, B computes the index aggregate key $K_S = \prod_{j \in S} \mathrm{Z}_{2^n - j}^r / \prod_{j \in S} \mathrm{Z}_{2^n - j + i^*}$. For user identity uid $\in R^*$, it satisfies the condition that path(uid) $\cap$ cover$(R^*) = \phi$. So for $x_k^* \in \text{cover}(R^*)$, it is bound to meet $x_k^* \notin \text{path(uid)}$. From the full binary tree T, user path is denoted as path(uid) =

$\{y_{u_0}, \ldots, y_{u_d}\}$, such that y_{u_0} = root and y_{u_d} = uid. B computes $P_{\text{uid}} = \prod_{y \in \text{path(uid)}} Z_{2^n - y}$; then the path aggregate key $K_{\text{uid}} = P_{\text{uid}}^{\varphi_k} / \prod_{y \in \text{path(uid)}} Z_{2^n - y + x_k^*}$, and SK = (K_S, K_{uid}) will be sent to A as the answer to query. Notice that

$$\begin{aligned} K_S &= g_{\mathbf{n}}^{r \sum_{j \in S} \alpha^{2^n - j} - \sum_{j \in S} \alpha^{2^n - j + i^*}} \\ &= g_{\mathbf{n}}^{\gamma \sum_{j \in S} \alpha^{2^n - j} + \alpha^{i^*} \sum_{j \in S} \alpha^{2^n - j} - \sum_{j \in S} \alpha^{2^n - j + i^*}} = g_{\mathbf{n}}^{\gamma \sum_{j \in S} \alpha^{2^n - j}} \\ &= \prod_{j \in S} Z_{2^n - j}^{\gamma}, \\ K_{\text{uid}} &= g_{\mathbf{n}}^{\varphi_k \sum_{y \in \text{path(uid)}} \alpha^{2^n - y} - \sum_{y \in \text{path(uid)}} \alpha^{2^n - y + x_k^*}} \\ &= g_{\mathbf{n}}^{\beta \sum_{y \in \text{path(uid)}} \alpha^{2^n - y} + \alpha^{x_k^*} \sum_{y \in \text{path(uid)}} \alpha^{2^n - y} - \sum_{y \in \text{path(uid)}} \alpha^{2^n - y + x_k^*}} \\ &= g_{\mathbf{n}}^{\beta \sum_{y \in \text{path(uid)}} \alpha^{2^n - y}} = \prod_{y \in \text{path(uid)}} Z_{2^n - y}^{\beta} = P_{\text{uid}}^{\beta}. \end{aligned} \tag{6}$$

Challenge. When A ends Phase 1, it will submit two equal-length messages $m_0, m_1 \in M$ to algorithm B. B works as follows:

(i) Step 1: it flips a random coin and marks the result as b for $b \in \{0, 1\}$ and sets $c_3^* = K \cdot m_b$.

(ii) Step 2: it sets $c_1^* = Y_1 = g_{\mathbf{n}}^{t_1}$, computes $c_2^* = {Y_1}^r$, and observes that $c_2^* = {Y_1}^r = g_{\mathbf{n}}^{t_1 r} = g_{\mathbf{n}}^{t_1(\gamma + \alpha^{i^*})}$.

(iii) Step 3: it sets $c_4^* = Y_2 = g_{\mathbf{n}}^{t_2}$, computes $c_5^* = \{Y^{\varphi_k}\}_{x_k^* \in \text{cover}(R^*)}$, and notices that

$$\begin{aligned} c_5^* &= \{{Y_2}^{\varphi_k}\}_{x_k^* \in \text{cover}(R^*)} = \left\{g_{\mathbf{n}}^{t_2 \varphi_k}\right\}_{x_k^* \in \text{cover}(R^*)} \\ &= \left\{g_{\mathbf{n}}^{t_2(\beta + \alpha^{x_k^*})}\right\}_{x_k^* \in \text{cover}(R^*)}. \end{aligned} \tag{7}$$

(iv) Step 4: it sets the challenge ciphertext $C^* = \langle c_1^*, c_2^*, c_3^*, c_4^*, c_5^* \rangle$ and sends C^* to A.

Phase 2. Similarly to Phase 1, A follows the constraints of the game and continues to query about private keys; algorithm B adopts the same strategy as in Phase 1 to answer the series of queries.

Guess. A outputs a bit b'. If $b' = b$, algorithm B outputs 1, which means $K = g_{2\mathbf{n}}^{t\alpha^{(2^n)}}$. Otherwise, B outputs 0 meaning that K is a random group element of $G_{2\mathbf{n}}$.

Probability analysis: when $K = g_{2\mathbf{n}}^{t\alpha^{(2^n)}}$, the challenge ciphertext $C^* = \langle c_1^*, c_2^*, c_3^*, c_4^*, c_5^* \rangle$ is encrypted of the message m_b. Otherwise, $C^* = \langle c_1^*, c_2^*, c_3^*, c_4^*, c_5^* \rangle$ is encrypted by a random element from the group $G_{2\mathbf{n}}$. In such case, the advantage of the adversary A (i.e., the probability of $b' = b$) is equal to 1/2. Above all, the obtained advantage of A for the proposed scheme is $\text{Adv}_A^{\text{RKAC}} = |\Pr[b' = b] - 1/2| = |(1/2 \pm \varepsilon) - 1/2| = \varepsilon$; namely, the advantage of A to break the scheme is negligible, which indicates the revocable key-aggregate encryption scheme with selective IND-CPA security.

6. Extension

As is known to us, the number of files may extremely be large and grow rapidly in cloud scenario. If the number of files exceeds N, which is the maximum number of files setting in the system, the whole system should be reestablished in our basic scheme. So how to reduce such burden is an important issue. Inspired by the thought of public key extension in the scheme [24], we propose an extended scheme to solve the problem. We attempt to label every file with a two-level index $\{i, j\}$ $(1 \le i \le h, 1 \le j \le (2^n - 1))$. When the number of files is more than N, we increase i by one and run the Extend algorithm to generate a new key-pair, adding to the original key-pair (PK, msk). That is to say, the number of files is increased by N once we obtain a new key-pair. Thus we can extend our basic scheme using this technique. The details of how to extend our basic scheme is shown as below.

The Setup, KeyGen, Update, and Verify algorithm are the same as the basic scheme.

Extend($\{\mu\}_h, \{\beta\}_h$): choose a random $\gamma_{h+1} \in Z_p$ and compute $v_{h+1} = g_{\mathbf{n}}^{\gamma_{h+1}}$ and output $\{v\}_{h+1} = \{v_1, v_2, \ldots, v_{h+1}\}$ as a part of PK and $\{\gamma\}_{h+1} = \{\gamma_1, \gamma_2, \ldots, \gamma_{h+1}\}$ as a part of msk.

Encrypt(PK, (a, b), m, params): for a message m and an index $\{a, b\}$ $(1 \le a \le h, 1 \le b \le (2^n - 1))$, randomly pick $t_1 \in Z_p$ and compute $c_2 = (v_a g_{\mathbf{n}}^{\alpha^b})^{t_1}$ and c_1, c_3 are computed by the same way as the basic scheme.

Extract(msk, uid, S_h, params): the path aggregate key K_{uid} remains the same as the basic scheme. For the set $S_h = \{\{i, j\}\}$ $(1 \le i \le h, 1 \le j \le (2^n - 1))$, the index aggregate key is computed as $K_{S_h} = \{\prod_{\{1,j\} \in S_h} Z_{2^n - j}^{\gamma_1}, \prod_{\{2,j\} \in S_h} Z_{2^n - j}^{\gamma_2}, \ldots, \prod_{\{h,j\} \in S_h} Z_{2^n - j}^{\gamma_h}\}$, denoted as $K_{S_h} = \{k_1, k_2, \ldots, k_h\}$.

Decrypt(C, SK, S_h, $\{a, b\}$, R, params): if either the index $\{a, b\} \notin S_h$ or the user's identity uid $\in R$, then return $\perp$. Otherwise, for $x = \text{path(uid)} \cap \text{cover}(R)$, decryption can be done as follows:

$$\begin{aligned} m = c_3 \cdot \frac{e\left(k_a \cdot \prod_{\{a,j\} \in S_h, j \ne b} Z_{2^n - j + b}, c_1\right)}{e\left(\prod_{\{a,j\} \in S_h} Z_{2^n - j}, c_2\right)} \\ \cdot \frac{e\left(K_{\text{uid}} \cdot \prod_{y \in \text{path(uid)}, y \ne x} Z_{2^n - y + x}, c_4\right)}{e\left(P_{\text{uid}}, c_5\right)}. \end{aligned} \tag{8}$$

The correctness of this equation can be verified after computation and therefore is omitted. The security of this extended scheme can be proved as the similar method as the basic scheme, so we do not explain it in detail here.

7. Conclusion and Future Work

In the cloud storage environment, in order to protect the security and privacy of users' data and to simplify key management in the process of data sharing more effectively, key-aggregate cryptosystem has been put forward. It is realized under the public key cryptosystem and can aggregate the user's private keys into a single one, greatly reducing the user's key management cost. At the same time, the aggregation can be achieved without constraints, realizing the flexible data sharing in cloud environment. This paper mainly studies

the revocable key-aggregate cryptosystem and proposes a revocable key-aggregate encryption scheme combined with the subset-cover framework in cloud environment, realizing the key aggregation and user access control effectively. By updating ciphertext via the cloud servers, the proposed scheme realizes the user permissions revocation while legitimate users do not need to update their private keys. What is more, it provides a verification mechanism to ensure user revocation is executed correctly. Performance analysis shows that, compared with the existing schemes, the proposed scheme reduces the cost of storage and transmission and realizes the user access control effectively. Security analysis shows that the proposed scheme proved to be selective CPA security based on Generalized DHDHE assumption in the standard model. Besides, an extended scheme is proposed to adapt for the cloud scenario, where the number of files is extremely large and growing rapidly.

This paper also has limitations that it only considers to construct a CPA security scheme. Since there are a lot of solutions to transfer a scheme from CPA security to CCA security [31], how to construct an efficient CCA secure key-aggregate encryption scheme will be a concern. And the total number of users is predefined in our revocable scheme, which is not conducive to flexible extension of the system. Therefore, how to design a key-aggregate encryption scheme united the revocation and extensibility will be the future work. In addition, trying to use the theory to solve some security problems in the practical application environment, such as how to apply the idea of revocable key-aggregate cryptosystem in the privacy-preserving of data aggregation and realize the data integrity verification, will be one of the future research directions.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

This work was partially supported by National Natural Science Foundation of China under Grants 61272415, 61070164; Natural Science Foundation of Guangdong Province, China, under Grant S2012010008767; Science and Technology Planning Project of Guangdong Province, China, under Grant 2013B010401015. This work was also supported by the Zhuhai Top Discipline-Information Security.

References

[1] J. Wu, L. Ping, X. Ge, W. Ya, and J. Fu, "Cloud storage as the infrastructure of Cloud Computing," in *Proceedings of the International Conference on Intelligent Computing and Cognitive Informatics (ICICCI '10)*, pp. 380–383, June 2010.

[2] K. Dahbur, B. Mohammad, and A. B. Tarakji, "Security issues in cloud computing: a survey of risks, threats and vulnerabilities," *Cloud Computing Advancements in Design Implementation & Technologies*, vol. 1, no. 3, pp. 1–11, 2011.

[3] P. Sikhar, S. Yash, and M. Debdeep, "Dynamic key-aggregate cryptosystem on ellipticcurves for online data sharing," *IACR Cryptology ePrint Archive*, vol. 2015, pp. 923–942, 2015.

[4] D. Naor, M. Naor, and J. Lotspiech, "Revocation and tracing schemes for stateless receivers," in *Advances in Cryptology—CRYPTO 2001. CRYPTO 2001*, J. Kilian, Ed., vol. 2139 of *Lecture Notes in Computer Science*, pp. 41–62, Springer, Berlin, Germany, 2001.

[5] S. Park, K. Lee, and D. H. Lee, "New constructions of revocable identity-based encryption from multilinear maps," *IEEE Transactions on Information Forensics & Security*, vol. 10, no. 8, pp. 1564–1577, 2015.

[6] Y. Shi, Q. Zheng, J. Liu, and Z. Han, "Directly revocable key-policy attribute-based encryption with verifiable ciphertext delegation," *Information Sciences*, vol. 295, pp. 221–231, 2015.

[7] S. G. Akl and P. D. Taylor, "Cryptographic solution to a problem of access control in a hierarchy," *ACM Transactions on Computer Systems (TOCS)*, vol. 1, no. 3, pp. 239–248, 1983.

[8] M. J. Atallah, M. Blanton, N. Fazio, and K. B. Frikken, "Dynamic and efficient key management for access hierarchies," *ACM Transactions on Information and System Security*, vol. 12, no. 3, pp. 1–43, 2009.

[9] A. R. Pais and S. Joshi, "A new probabilistic rekeying method for secure multicast groups," *International Journal of Information Security*, vol. 9, no. 4, pp. 275–286, 2010.

[10] G. Ateniese, A. De Santis, A. L. Ferrara, and B. Masucci, "Provably-secure time-bound hierarchical key assignment schemes," *Journal of Cryptology*, vol. 25, no. 2, pp. 243–270, 2012.

[11] J. Benaloh, M. Chase, E. Horvitz, and K. Lauter, "Patient controlled encryption: ensuring privacy of electronic medical records," in *Proceedings of the ACM Workshop on Cloud Computing Security (CCSW '09), Co-located with the 16th ACM Computer and Communications Security Conference (CCS '09)*, pp. 103–114, Chicago, Ill, USA, November 2009.

[12] J. Benaloh, "Key compression and its application to digital fingerprinting," Tech. Rep., Microsoft Research, 2009.

[13] B. Alomair and R. Poovendran, "Information theoretically secure encryption with almost free authentication," *Journal of Universal Computer Science*, vol. 15, no. 15, pp. 2937–2956, 2009.

[14] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Advances in Cryptology*, pp. 47–53, Springer, Berlin, Germany, 1984.

[15] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Proceedings of the Annual International Cryptology Conference (CRYPTO '01)*, vol. 44, no. 3 of *Lecture Notes in Computer Science (LNCS)*, pp. 389–392, Santa Barbara, Calif, USA, 2001.

[16] F. Guo, Y. Mu, and Z. Chen, "Identity-based encryption: how to decrypt multiple Ciphertexts using a single decryption key," in *Proceedings of the Pairing-based Cryptography (Pairing '07)*, pp. 392–406, Springer, Berlin, Germany, 2007.

[17] F. Guo, Y. Mu, Z. Chen, and L. Xu, "Multi-identity single-key decryption without random oracles," in *Information security and cryptology*, vol. 4990 of *Lecture Notes in Comput. Sci.*, pp. 384–398, Springer, Berlin, Germany, 2008.

[18] Y. Ming, Y. Wang, and L. Pang, "Provably secure multi-identity single-key decryption scheme in the standard model," *Computer Science*, vol. 37, no. 3, pp. 73–75, 2010.

[19] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Advances in Cryptology—EUROCRYPT 2005*, pp. 457–473, Springer, Berlin, Germany, 2005.

[20] B. Waters, "Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization," in *Public Key Cryptography—PKC 2011. PKC 2011*, D. Catalano, N. Fazio,

R. Gennaro, and A. Nicolosi, Eds., vol. 6571 of *Lecture Notes in Computer Science*, pp. 53–70, Springer, Berlin, Germany, 2011.

[21] M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou, "Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 1, pp. 131–143, 2013.

[22] R. Canetti and S. Hohenberger, "Chosen-ciphertext secure proxy re-encryption," in *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS '07)*, pp. 185–194, Alexandria, Va, USA, November 2007.

[23] C. Wu, S. Li, and Y. Zhang, "Key management scheme based on secret sharing for wireless sensor networks," *International Journal of Information and Communication Technology*, vol. 7, no. 2-3, pp. 126–140, 2015.

[24] C.-K. Chu, S. S. M. Chow, W.-G. Tzeng, J. Zhou, and R. H. Deng, "Key-aggregate cryptosystem for scalable data sharing in cloud storage," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 2, pp. 468–477, 2014.

[25] K. Kate and S. D. Potdukhe, "Data sharing in cloud storage with key-aggregate cryptosystem," *International Journal of Engineering Research and General Science*, vol. 2, no. 6, pp. 882–886, 2014.

[26] B. Cui, Z. Liu, and L. Wang, "Key-aggregate searchable encryption (KASE) for group data sharing via cloud storage," *Institute of Electrical and Electronics Engineers. Transactions on Computers*, vol. 65, no. 8, pp. 2374–2385, 2016.

[27] H. Dang, Y. L. Chong, F. Brun et al., "Fine-grained sharing of encrypted sensor data over cloud storage with key aggregation," *IACR Cryptology ePrint Archive*, pp. 739–750, 2015.

[28] Q. Gan and X. Wang, "An efficient key-aggregate encryption scheme under cloud environment," *Computer Engineering*, vol. 42, no. 2, pp. 33–44, 2016.

[29] D. Boneh and A. Silverberg, "Applications of multilinear forms to cryptography," in *Contemporary Mathematics*, vol. 324, pp. 71–90, 2003.

[30] D. Boneh, B. Waters, and M. Zhandry, "Low overhead broadcast encryption from multilinear maps," in *Advances in Cryptology—CRYPTO 2014*, vol. 8616 of *Lecture Notes in Comput. Sci.*, pp. 206–223, Springer, Heidelberg, Germany, 2014.

[31] R. Canetti, S. Halevi, and J. Katz, "Chosen-ciphertext security from identity-based encryption," in *Advances in cryptology—EUROCRYPT 2004*, vol. 3027 of *Lecture Notes in Comput. Sci.*, pp. 207–222, Springer, Berlin, Germany, 2004.