Hindawi
Security and Communication Networks
Volume 2021, Article ID 6661450, 13 pages
https://doi.org/10.1155/2021/6661450

Research Article

Multiauthority Attribute-Based Encryption with Traceable and Dynamic Policy Updating

Jie Ling,[1] **Junwei Chen**,[1] **Jiahui Chen**,[1] **and Wensheng Gan**[2]

[1]*School of Computer, Guangdong University of Technology, Guangzhou 510006, China*
[2]*College of Cyber Security, Jinan University, Guangzhou 510632, China*

Correspondence should be addressed to Jiahui Chen; csjhchen@gmail.com

Received 9 December 2020; Revised 15 January 2021; Accepted 10 February 2021; Published 26 February 2021

Academic Editor: Prosanta Gope

Ciphertext policy attribute-based encryption (CP-ABE) is an encryption mechanism that can provide fine-grained access control and adequate cloud storage security for Internet of Things (IoTs). In this field, the original CP-ABE scheme usually has only a single trusted authority, which will become a bottleneck in IoTs. In addition, different users may illegally share their private keys to obtain improper benefits. Besides, the data owners also require the flexibility to change their access policy. In this paper, we construct a multiauthority CP-ABE scheme on prime order groups over a large attribute universe. Our scheme can support white-box traceability along with policy updates to solve the abovementioned three problems and, thus, can fix the potential requirements of IoTs. More precisely, the proposed scheme supports multiple authority, white box traceability, large attribute domains, access policy updates, and high expressiveness. We prove that our designed scheme is static secure and traceable secure based on the state-of-the-art security models. Moreover, by theoretical comparison, our scheme has better performance than other schemes. Finally, extensive experimental comparisons show that our proposed algorithm can be better than the baseline algorithms.

1. Introduction

With the help of cloud computing technology, Internet of Things (IoTs) [1] can bridge physical devices and virtual objects, which has become a promising networking scenario in the cyber world. In IoTs, more and more companies and individuals store data in the cloud, requiring the cloud servers to provide data access services. However, cloud servers are generally considered to be untrustworthy for the reason that the data of IoTs often contain sensitive information. In order to protect the privacy of these data, one of the traditional technologies is to encrypt the data, and data owners need to be online at all times to distribute their secret keys. Although these technologies achieve access control, the management of these keys will become a bottleneck when more and more users joined the system. In addition, for each type of data, it is necessary to maintain one or more copies of the ciphertext for different users with different keys, which will cause a waste of storage overhead in an IoTs system [2]. To this end, Sahai et al. [3] firstly proposed attribute-based encryption. The concept of attribute-based encryption (ABE) is a one-to-many encryption mechanism that can provide fine-grained access control and data security. Goyal et al. [4] further proposed the key policy ABE (KP-ABE) and ciphertext policy ABE (CP-ABE). Then, Bethencourt et al. [5] studied the CP-ABE scheme with a complete description, showing that CP-ABE allows data owners to define access strategies under the user's attributes. Once the user encrypts specific data, other users can decrypt them if and only if their attributes meet the access policy. Thanks to these characteristics, the CP-ABE scheme is considered a more suitable encryption mechanism for cloud storage access control than KP-ABE.

However, the original CP-ABE scheme only has a single, trusted authority dealing with the user's key distribution and attribute management, which will become a bottleneck in the cloud, especially in an IoTs system. Liu et al. [6] proposed a scheme under a different hierarchy of attributes with the

name of ciphertext-policy hierarchical attribute-based encryption. Deng et al. [7] elaborate on ABE and propose a new versatile cryptosystem referred to as ciphertext-policy hierarchical ABE. Wang et al. [8], based on the access structure layered model, proposed a novel access control scheme about file hierarchy by using ABE to solve the problem. Liu et al. [9] propose a novel T-CP-ABE system that gives high policies expressiveness in any monotone access structures and add traceability. Liang et al. [10] propose a CP-ABPRE to deal with the security problem by using the dual system encryption technology with the selective proof technique. But, the schemes mentioned above are all single attribute authorization (AA) ABE schemes. It is completely borne in the cloud environment, which not only brings a serious burden to the authorization center but also requires the authorization center to be completely trusted. Single-attribute authority cannot meet the development needs of practical applications because different attributes in different fields in many application scenarios are caused by different environments. For example, there is a situation that the data owner wants to share data with the researchers in the research institutes and the managers in the government departments. In this case, the attributes of researchers are determined by the research institutes. At the same time, the "government attributes" are managed by the government department. The abovementioned ABE schemes are not suitable for this situation where the attributes need to be managed by multiple agencies.

On the other side, in some CP-ABE schemes, it is easy to discover their attributes in the private key. There may be another situation that some malicious users illegally share their private keys to obtain economic benefits. Thus, the features of the CP-ABE scheme that can track leaked secret keys are particularly important. Therefore, we also need a traceability mechanism to track these malicious users. For example, attackers can access critical vulnerabilities in a wide variety of IoTs applications and devices to perform their malicious activities. This requires the design of effective security mechanisms in an IoTs-related application.

Except for the traceability, the policy update of the CP-ABE system also needs to be considered for supplying more functions. For instance, when addressing security, trust, and privacy in IoTs, the data owner may need to alter the access policy stored on the cloud. In that case, the traditional solution is to let the data owner find the cloud storage server's relevant ciphertext and decrypt it, then encrypt the ciphertext using a new access strategy, and upload the newly encrypted ciphertext back to the cloud server. It, thus, brings much computational burden to the system. Therefore, the policy update is another important characteristic of the actual system.

To sum up, there are three major challenges in CP-ABE that we need to solve as follows:

(1) How to solve the bottleneck of single authority authorization in cloud storage applications, especially in an IoTs system?

(2) How to prevent some malicious users from illegally sharing their private keys?

(3) How to propose an algorithm that makes the data owner's access control more flexible in IoTs-enabled applications?

1.1. Our Contribution. This paper addresses the abovementioned challenges by proposing a scheme named T-DPU-MCP-ABE (Traceable and Dynamic Policy Updating Multiauthority Attribute-based Encryption). More precisely, we propose a T-DPU-MCP-ABE based on the prime order bilinear group, and we prove its static security and resistance to traceable attacks under two related security models. Our security assumption utilizes the q-type hypothesis [11] and is based on the LRSW hypothesis [12]. As far as we know, we are the first one to support the properties of large attribute domain, policy update, white box traceability, multiauthorization, and high expressiveness and still have good performance. Especially, the features are described in detail as follows:

(1) Large attribute domain: the size of public parameters is affected by the number of authorized institutions and will not increase linearly with the number of attributes. There is no need to determine the system attribute domain when the system is established.

(2) Policy update: data owners may often need to modify the ciphertext access policy according to various requirements. Policy updates provide flexibility and allow data owners to adjust their encrypted data access policies to achieve fine-grained control.

(3) White box Traceability: it can track malicious users who illegally share private keys. Through white box tracking that does not need to maintain a user list, the efficiency of the solution is improved, and no additional storage overhead is consumed.

(4) Multiple authorized authorities: multiple authorized authorities undertake the key distribution work and, thus, reduce the workload and solve the problem of incomplete trustworthiness of the single authority.

(5) High expressiveness: supports flexible access control and supports any monotonous access structure access strategy.

1.2. Organization. The rest of this paper is arranged as follows. In Section 3, we introduce the necessary background knowledge. In Section 4, we give the formal definition and security model of auditable ABE. In Section 5, we give the main constructions and security analysis. In Section 6, we provide a performance and experiment evaluation. Finally, Section 7 presents a brief conclusion and future work.

2. Related Work

Melissa [13] proposed a ciphertext strategy-based multiagency authorization attribute-based encryption (MCP-ABE) scheme. The scheme has a central authority with the ability to decrypt each ciphertext, which reduces the security of decryption key storage. Lewko et al. [14] proposed a

multiagency authorization scheme that supports arbitrary access structures based on the groups in composite order, resulting in a low efficiency. In order to improve the efficiency of the scheme, Yannis et al. [15] proposed a CP-ABE scheme based on prime order groups and made it support large attribute domains. Then, Yannis et al. [11] proposed a multiagency authorization CP-ABE scheme based on prime order groups and also support large attribute domains. In this scheme, the authors used the linear secret-sharing scheme (LSSS) to improve expression ability. However, none of the abovementioned studies support traceability.

The traceability in ABE is divided into white-box traceable and black-box traceable [16]. In this field, Ning et al. [17] proposed a white-box traceable method that enables large attribute domains and high expressive capability. Their white-box traceable scheme is based on a single authorization center. To improve this, Li et al. [18] proposed a CP-ABE scheme with multiauthorization centers. However, this scheme only supports the access strategy of the AND gate, which limits in low expressive capability. Then, Zhou et al. [19] proposed a multiagency authorization CP-ABE scheme with white-box traceable that supports high expressive capability on medical cloud systems. However, their scheme does not support large attribute domains, and each authorization center has to maintain an identification table, which increases the storage overhead for tracking.

In the study of policy update, Ying et al. [20] proposed the first CP-ABE scheme that supports the modification of any form of fine-grained access control policy, and it is proved to be adaptive and secure under the standard model, but the system's communication overhead and storage overhead are high. After that, Liu et al. [21] proposed an ABE scheme that supports outsourcing decryption, attribute revocation, and policy update. This scheme is more flexible and practical in practice, but its privacy-protection capabilities are slightly lacking. Recently, Jing et al. [22] proposed a CP-ABE scheme that supports access policy update and rapid expansion of attributes but did not consider the application scenarios of multiauthorization agencies.

3. Background

3.1. Access Structure. We define U as a set of attributes, an access structure $\mathbb{A}$ is a collection of nonempty subsets of U, that is, $\mathbb{A} \in 2^U/\{\phi\}$, and the collection contained in $\mathbb{A}$ is called an authorization set. If the user has an authorized attribute set, the user can perform decryption, but not vice versa.

For all B and C, $B \in A$, and $B \subseteq C$, if $C \in A$, we say that the access structure $\mathbb{A}$ is monotonous. We restrict to a monotone access structure in this paper.

3.2. Prime-Order Bilinear Groups. Let p be a big prime and $\mathbb{G}$ and $\mathbb{G}_T$ be cyclic groups with prime order p; we say that e: $G \times G \longrightarrow G_T$ is a computable bilinear map if it has the following properties:

(1) Bilinear, i.e., $(e(P^a, Q^b) = e(P, Q)^{ab})$ for all $P, Q \in \mathbb{G}, a, b \in \mathbb{Z}_q$

(2) Nondegeneracy, i.e., there exists $P, Q \in \mathbb{G}$ such that $e(P, Q) \neq 1$, namely, the map does not send all pairs in $\mathbb{G} \times \mathbb{G}$ to the identity in $\mathbb{G}_T$

(3) Computability, i.e., there is an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in \mathbb{G}$

3.3. Linear Secret-Sharing Schemes. Let U be the set of attributes, as shown in [23]; $\prod$ is a linear secret-sharing scheme (LSSS) on U if it has the following properties:

(1) For each attribute form of a vector over $\mathbb{Z}_p$, there is a secret share $s \in \mathbb{Z}_p$.

(2) The matrix for $\prod$ is called a share-generating matrix meaning a matrix M with l rows and n columns for each access structure $\mathbb{A}$ on S. For $i = 1, \ldots, l$, we define a function ρ labels row i of M with attribute $\rho(i)$. We consider the column vector $\overrightarrow{v} = (s, r_2, \ldots, r_n)$, where $s \in \mathbb{Z}_p$ is the secret to be shared and $r_2, \ldots, r_n \in \mathbb{Z}_p$ are randomly chosen. Then, $M\overrightarrow{v} \in \mathbb{Z}_p^{l\times 1}$ is the vector of l shares of the secret s according to $\prod$.

For the LSSS scheme, it enjoys the linear reconstruction property. More precisely, let $\prod$ be an LSSS for the access structure $\mathbb{A}$, $S^* \in \mathbb{A}$ be an authorized set, and let $I \subset \{1, 2, \ldots, l\}$ be defined as $I = \{i \in [l] \wedge \rho(i) \in S^*\}$. Then, for constants $\{\omega_i \in \mathbb{Z}_p\}_{i\in I}$ such that, for any valid shares $\{\lambda_i = (M\overrightarrow{v})_i\}_{i\in I}$ of a secret s according to $\prod$, we have $\sum_{i\in I}\omega_i\lambda_i = s$.

3.4. Problem Assumption. Decisional q-parallel bilinear Diffie–Hellman exponent *(*q-PBDHE*)* assumption: the decisional q-parallel bilinear Diffie–Hellman exponent (decisional q – PBDHE) problem [11] is saying that, given the tuple (G, p, e, g, g^s), it satisfies

$$\forall i \in \{1, \ldots, 2q\}, j \in \{1, \ldots, q\}, i \neq q+1: \left(g^{a^i}, g^{b_j a^i}\right), \quad (1)$$

$$\forall i \in \{1, \ldots, q\}: \left(g^{s/b_i}\right), \quad (2)$$

$$\forall i \in \{1, \ldots, q+1\}, j, j' \in \{1, \ldots, q\}, j \neq j': \left(g^{\left(sa^i b_j/b_{j'}\right)}\right), \quad (3)$$

if we can distinguish $Z = e(g, g)^{a^{q+1}s}$ from a random value in G_T.

Formally speaking, if $|\Pr[\mathscr{A}(\overrightarrow{y}, Z = e\ (g, g)^{a^{q+1}s}) = 0] - \Pr[\mathscr{A}(\overrightarrow{y}, R) = 0]| \geq \varepsilon$, we say that an algorithm $\mathscr{A}$ has advantage ε in solving the abovementioned decisional q – PBDHE problem. Then, if all probabilistic polynomial time (PPT) algorithms have, at most, a negligible advantage in solving the decisional q – PBDHE problem, we say that the decisional q – PBDHE assumption holds.

LRSW assumption [12]: let G be the cyclic group of order p, g be a generator of G, and two random values $x, y \in \mathbb{Z}_p$ satisfy $X = g^x$ and $Y = g^y$. Let $\mathcal{O}_{X,Y}(\cdot)$ be the random oracle, which inputs $m \in \mathbb{Z}_p$ and outputs a triplet

$A = (a, a^y, a^{x+mxy})$, where $a \in G$. If there is no probability polynomial time algorithm that can generate m, a, b, c satisfying $m \notin Q, Q \in \mathcal{O}_{X,Y}(\cdot), m \in \mathbb{Z}_p, m \neq 0, a \in G, b = a^x, c = a^{x+mxy}$ with probability at the least ε, then the LRSW assumption in group G is said to be true.

4. Definition and Security Model

4.1. System Model. We show the framework of our system in Figure 1. There are six main entities, namely, cloud storage provider, attribute authorities (AAs), data owners, data users, system party, and trusted party. The system party will invoke the system setup algorithm and generate the public parameters (PP). The PP is then firstly distributed to the attribute authorities, data owners, data users, and the trusted party. Then, the AAs invoke the authority setup process to generate public keys (PKs) and send their public keys to the data owners, data users, and the trusted party. Also, if the data users possess valid credentials, AAs will assign the attributes to them according to their request. The data owner generates ciphertext (CT) for the message he wants to encrypt and uploads to the cloud storage provider. Once the data owner wishes to alter the access policy over the existing CT, he/she sends a policy update key to the cloud storage provider. Then, in the cloud storage, the ciphertext will be updated accordingly. Subsequently, if the users' attributes satisfy the access policy of the CT, they can use the components of secret key to generate their secret key SK and perform decryption operation. Finally, the trusted party invokes the tracing algorithm if there is dispute or suspicion and reports the suspected user's ID (gid) to the AAs.

4.2. Definition. Our proposed cryptosystem according to the abovedescription consists of the following eight algorithms:

Setup $(\lambda) \longrightarrow$ (PP): on input of a security parameter λ, the algorithm (run by the system) outputs the global PPs.

AuthoritySetup $(\text{aid}, \text{PP}) \longrightarrow (\text{SK}_{\text{aid}}, \text{PK}_{\text{aid}})$: we assume each authority is recognized by an identifier aid. On input of the global PPs and aid, the algorithm outputs the public key PK_{aid} and the cloud secret key SK_{aid}.

KeyGen $(\text{gid}, S, \{\text{SK}_{\text{aid}}\}, \text{PP}) \longrightarrow \text{SK}_{S,\text{gid}}$: on input of the user identity (gid), a set of user's attributes S, and the corresponding authority's secret keys SK_{aid} and PP, the algorithm outputs the private key $\text{SK}_{S,\text{gid}}$ for user matching his/her attribute set S.

Encrypt $(\text{msg}, (M, \rho), \text{PK}_{\text{aid}}, \text{PP}) \longrightarrow$ (CT): this algorithm is run by a data owner who wants to share the data in the cloud. The algorithm inputs the message (msg) concerning an access policy (M, ρ), a set of respective public keys PK_{aid} and PP, and outputs the ciphertext CT.

Decrypt $(\text{CT}, \text{SK}_{S,\text{gid}}, \text{PP}) \longrightarrow$ msg: this algorithm is run by a data user. On input of the global PPs, a ciphertext CT and a private key $\text{SK}_{S,\text{gid}}$ matching an attribute set S and the algorithm outputs the message msg if decryption is possible.

PolicyUpdateKeyGen $(\text{PP}, \text{PK}_{\text{aid}}, \text{SharesInfo}(\text{msg}), (M, \rho), (M', \rho')) \longrightarrow \text{UK}_{\text{msg}}$: this algorithm is run by a data owner. On input of the global PPs, a set of public keys PK_{aid}, the encryption information $\text{SharesInfo}(\text{msg})$, the old access policy (M, ρ), and new access policy (M', ρ'), the algorithm outputs the policy update key UK_{msg}.

CTUpte $(\text{CT}, \text{UK}_{\text{msg}}) \longrightarrow \text{CT}'$: this algorithm is run by the cloud storage provider. On input of the ciphertext CT and updated key UK_{msg}, the algorithm outputs an updated ciphertext CT'.

Trace $(\text{SK}_{S,\text{gid}}, \{\text{PK}_{\text{aid}}\}, \text{PP}) \longrightarrow$ gid or $\perp$: this algorithm is run by the trusted party. On input of the decryption key $\text{SK}_{S,\text{gid}}$ and the public keys $\{\text{PK}_{\text{aid}}\}$ for corresponding authorities and PPs, the algorithm outputs an authority gid.

4.3. Security Model. We focus on two types of adversaries as follows:

(1) We consider the malicious data users as the *static adversary*. For static adversaries [11], we request that no unauthorized user can decrypt encrypted data stored in the cloud. In addition, we request that the collusion of a group of unauthorized malicious users is still unable to obtain unauthorized decryption privileges, which means our scheme needs to have collusion resistance.

(2) We consider the "honest but curious" cloud provider as the *traceable adversary*. We assume that the traceable adversary [24] will follow the protocol's specification but will collect as much information as possible, i.e., secret/private keys. The traceable adversary is not allowed to obtain more secret information than it already has. In addition, it cannot identify "who has accessed the encrypted data" and "who has requested the decryption service." Also, it cannot link a valid decryption request to a previous decryption request.

Then, we have the following two security models.

4.3.1. Model 1: Security for Static Adversary. The security model for static adversary is based on the static security model [11]. To define the security of our scheme (satisfying the abovementioned requirements), we design the following security games:

Init. The adversary $\mathcal{A}$ selects a set of corrupted authorization agencies, records it as $C_{\text{aid}} \subseteq U_{\text{aid}}$, and keeps it unchanged throughout the game. The normal authorized agencies are recorded as $N_{\text{aid}} \subseteq U_{\text{aid}}$ with $N_{\text{aid}} \subseteq U_{\text{aid}} = \varnothing$; $\mathcal{A}$ knows the secret key of each corrupted organization $\{\text{SK}_{\text{aid}}\}_{\text{aid} \in C_{\text{aid}}}$.

Setup. The challenger $\mathscr{C}$ runs the system Setup of the solution in this article and sends the global PP to the opponent.

Query. $\mathscr{A}$ requests $\{(\text{gid}_j, S_j)\}_{j\in[m]}$ as the relevant private key, where $S_j \subseteq U$ is the attribute set of the user with identity gid_j. All users' identities are unique, and for arbitrary $i \in S$, there holds $T(i) \notin C_{\text{aid}}$. Then, the adversary sends two messages msg_0 and msg_1 with the same length and a set of challenges $\{(M_i, \rho_i), \ldots, (M_p, \rho_p)\}$. For each challenge, the access policy must satisfy the nonauthorization set. Finally, the ciphertext policy is requested to update any two access policies of the query challenge message and among them.

Challenge. The challenger $\mathscr{C}$ randomly selects and responds to the adversary according to the RW scheme [11], including a set of public keys of the normal authority, a satisfied user private key, and a set of verification ciphertexts used to challenge the adversary. We use the simulator to convert the adversary's query into a form that the challenger can recognize as a RW scheme and also convert the challenger's response to the adversary.

Guess. $\mathscr{A}$ outputs a guess $b' = \{0, 1\}$ for b.

As can be seen in this game, the advantage of $\mathscr{A}$ is defined as $\text{Adv} = |\Pr[b' = b] - 1/2|$.

According to [11], we have the following definition.

Definition 1. The T-DPU-MCP-ABE scheme is static secure if all PPT adversaries have at most a negligible advantage in the abovementioned game.

4.3.2. Model 2: Security for Traceable Adversary. The security game for traceable adversary is similar to the game of the static one except the *Setup*, *Query*, and *Forgery* (identical to *Guess*) as follows:

Setup. $\mathscr{C}$ runs $\text{Setup}(\lambda)$ and $\text{AuthoritySetup}(\text{aid}, \text{PP})$ and sends the PP and the authority public key PK_{aid} to $\mathscr{A}$.

Query. $\mathscr{A}$ requests $\{(\text{gid}_j, S_j)\}_{j\in[m]}$ as the relevant private key, where $S_j \subseteq U$ is the attribute set of the user with identity gid_j. Then, $\mathscr{C}$ runs *KeyGen* $(\text{gid}_j, S_j, \text{SK}_{\text{aid}}, \text{PP})$ and sends $\{\text{SK}_{S_j,\text{gid}_j}\}_{j\in[m]}$ to $\mathscr{A}$.

Forgery. $\mathscr{A}$ outputs a forgery secret key SK^*, if $\text{Trace}(\text{SK}_{\text{S,gid}}, \{\text{PK}_{\text{aid}}\}, \text{PP}) \notin \Delta$, and $\text{gid} \notin \{\text{gid}_1, \ldots, \text{gid}_m\}$.

According to [24], we have the following definition.

Definition 2. The T-DPU-MCP-ABE scheme is traceable secure if all PPT adversaries have at most a negligible advantage $|\Pr[\text{Trace}(\text{SK}^*, \{\text{PK}_{\text{aid}}\}, \text{PP}) \notin \{\Delta, \text{gid}_1, \ldots, \text{gid}_m\}]|$ in the abovementioned game.

5. Traceable and Dynamic Policy Updating Multiauthority Attribute-Based Encryption

Here, we present our attribute-based key encryption scheme. Our scheme is constructed on the bilinear group G with a large prime order p and utilizes the LSSS access strategy together with two random oracle hash functions H_1 and H_2. We realize the traceability by adopting the CL (Camenisch–Lysyanskaya) signature scheme [25]. Our scheme has two domains, namely, the attribute domain U and the authority domain U_{aid}. There is a corresponding authorized authority aid releasing an effective attribute set to the users for each attribute.

Then, our scheme is specifically constructed as follows.

5.1. Our Construction

Setup $(\lambda) \longrightarrow$ (PP): this algorithm takes as input the security parameter λ and gets $D = (G, G_T, p, e)$, where p is the prime order and G_T, e is the bilinear mapping $e: G \times G \longrightarrow G_T$. It sets the attribute universe be $\mathscr{U} = \mathbb{Z}_p$. It then chooses random $g \in G$ and three cryptographic hash functions H_1, H_2, and T, where $H_1, H_2: \{0,1\}^* \longrightarrow \mathbb{G}$ are used to hash the identity and the attribute of a user into an element of G, respectively. Also, $T: \{0,1\}^* \longrightarrow \mathbb{Z}_q^*$ is used to hash the attribute i into the corresponding aid. Finally, this algorithm sets the global public parameters $\text{PP} = (G, G_T, p, e, g, H_1, H_2, T)$ as output.

AuthoritySetup $(\text{aid}, \text{PP}) \longrightarrow (\text{SK}_{\text{aid}}, \text{PK}_{\text{aid}})$: the algorithm chooses three random $\alpha_{\text{aid}}, \beta_{\text{aid}}, \gamma_{\text{aid}} \in \mathbb{Z}_p$. Together with the inputs aid and PP, it then publishes the public key $\text{PK}_{\text{aid}} = \{e(g,g)^{\alpha_{\text{aid}}}, g^{\beta_{\text{aid}}}, g^{\gamma_{\text{aid}}}\}$ of the AU and sets the secret key as $\text{SK}_{\text{aid}} = \{\alpha_{\text{aid}}, \beta_{\text{aid}}, \gamma_{\text{aid}}\}$.

KeyGen $(\text{gid}, S, \{\text{SK}_{\text{aid}}\}, \text{PP}) \longrightarrow \text{SK}_{\text{S,gid}}$: the algorithm chooses random $t \in \mathbb{Z}_p$, $u \in G, u \notin H_1(\text{gid})$ and computes

$$K_{1,i,\text{gid}} = g^{\alpha_{\text{aid}}} \cdot H_1(\text{gid})^{\beta_{\text{aid}}} \cdot H_2(i)^t \cdot u^{\beta_{\text{aid}}(\text{gid}+\gamma_{\text{aid}})} K_{2,i,\text{gid}} = u^{\gamma_{\text{aid}}} K_{3,i,\text{gid}} = u K_{4,i,\text{gid}} = g^t K_{5,\text{gid}} = \text{gid}. \tag{4}$$

It outputs the secret key $\text{SK}_{\text{S,gid}} = \{\{K_{1,i,\text{gid}}, K_{2,i,\text{gid}}, K_{3,i,\text{gid}}, K_{4,i,\text{gid}}\}_{i\in S}, K_{5,\text{gid}}\}$.

Encrypt $(\text{msg}, (M, \rho), \text{PK}_{\text{aid}}\text{PP}) \longrightarrow (\text{CT})$: on input of the message (msg), the PPs and an access policy (M, ρ) (where M is an $l \times n$ matrix), the public key of the agency PK_{aid}, and the public parameters PP, the algorithm firstly chooses a random $s \in \mathbb{Z}_p$. Then, it chooses random $x_2, \ldots, x_n \in \mathbb{Z}_p$, sets two vectors $\mathbf{v} = (s, x_1, x_2, \ldots, x_n)$ and $v = (0, v_2, \ldots, v_n)$, and computes the vectors of shares of s and 0 as $\lambda_x = \mathbf{M}_x \mathbf{v}^T$ and $\omega_x = \mathbf{M}_x v^T$, respectively (where T denotes the transpose of the matrix).

Finally, it chooses random $r_x \in \mathbb{Z}_p$ and computes

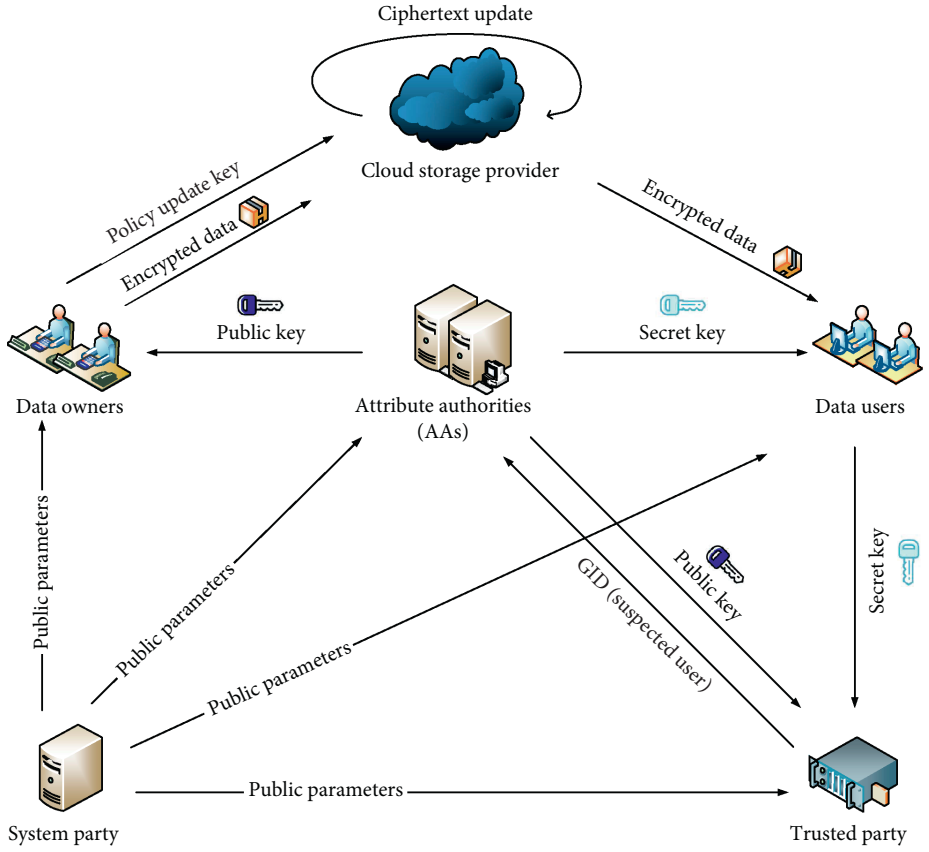

Figure 1: Framework of the system model.

$$\begin{aligned} C_0 &= \mathrm{msg} \cdot e(g,g)^{s}, \\ C_{1,x} &= e(g,g)^{\lambda_x + \alpha_{\delta(x)} r_x}, \\ C_{2,x} &= g^{\omega_x}, \\ C_{3,x} &= g^{\beta_{\delta(x)} r_x}, \\ C_{4,x} &= H_2(\rho(x))^{r_x}, \\ C_{5,x} &= g^{-r_x}. \end{aligned} \tag{5}$$

The ciphertext CT is set as $\mathrm{CT} = \{C_0, \{C_{1,x}, C_{2,x}, C_{3,x}, C_{4,x}, C_{5,x}\}_{x\in\{1,\ldots,l\}}\}$.

Decrypt $(\mathrm{CT}, \mathrm{SK}_{S,\mathrm{gid}}, \mathrm{PP}) \longrightarrow$ msg: on input of $\mathrm{CT} = \left\{C_0, \left\{C_{1,x}, C_{2,x}, C_{3,x}, C_{4,x}, C_{5,x}\right\}_{x\in\{1,\ldots,l\}}\right\}$, S, $\mathrm{SK}_{S,\mathrm{gid}}$, and PP, the algorithm sets the identification set as $I \subseteq \{1,\ldots,l\}$. For all $x \in I$ and $\{x\colon \rho(x) \in S\}$, the algorithm computes

$$\begin{aligned} D_x = {} & C_{1,x} \cdot e\left(H_1\left(K_{5,\mathrm{gid}}\right), C_{2,x} \cdot C_{3,x}\right) \\ & \cdot e\left(K_{2,\rho(x),\mathrm{gid}} \cdot K_{3,\rho(x),\mathrm{gid}}^{K_{5,\mathrm{gid}}}, C_{3,x}\right) \cdot e\left(C_{4,x}, K_{4,\rho(x),\mathrm{gid}}\right) \\ & \cdot e\left(K_{1,\rho(x),\mathrm{gid}}, C_{5,x}\right), \end{aligned} \tag{6}$$

where $\{c_x\}_{x\in I}$ and $\sum_{x\in I} c_x \mathbf{M}_x = (1, 0, \ldots, 0)$.

Finally, the message is recovered by computing

$$\mathrm{msg} = \frac{C_0}{\prod_{x\in I} D_x^{c_x}}. \tag{7}$$

PolicyUpteKeyGen $(\mathrm{PP}, \mathrm{PK}_{\mathrm{aid}}, \mathrm{Shares}, (M,\rho), (M',\rho')) \longrightarrow \mathrm{UK}_{\mathrm{msg}}$: M is a generator matrix of $1,\ldots,n$, and Shares represents the information of the two random vectors $\mathbf{v}$ and v contained in the encryption algorithm. We define the function $\delta(i) = T(\rho(i))_{i\in[I]}$ and $\delta'(j) = T'(\rho(j))_{j\in[I]}$.

First, the new access strategy and the old access strategy are used as input through the strategy comparison method in the literature [26] to generate three subset record rows indexes $I_{1,M}, I_{2,M}, I_{3,M}$. Then, it picks two random vectors $v' = (s, v_2', \ldots, v_n')$ and $v' = (0, v_2', \ldots, v_n')$ and then calculates $\lambda_j' = \mathbf{M}_j' \mathbf{v}'^{T}$ and $\omega_j' = \mathbf{M}_j' v'^{T}$ with $j \in \{1,\ldots,l'\}$.

When the row index satisfies $(j,i) \in I_{1,M'}$ (marked as module 1), the algorithm generates the update key as

$$\left\{\mathrm{UK}_{j,i,\mathrm{msg}}\right\}_1 = \left\{\mathrm{UK}_{1,j,i,\mathrm{msg}} = g^{\lambda_{j\prime} - \lambda_i}, \mathrm{UK}_{2,j,i,\mathrm{msg}} = g^{\omega_{j\prime} - \omega_i}\right\}. \tag{8}$$

When the row index satisfies $(j,i) \in I_{2,M'}$ (marked as module 2), the algorithm randomly picks $a_j \in \mathbb{Z}_p$ and calculates the update key as

$$\left\{\mathrm{UK}_{j,i,\mathrm{msg}}\right\}_2 = \left\{a_j, \mathrm{UK}_{1,j,i,\mathrm{msg}} = g_i^{\lambda_j' - a_j\lambda}, \mathrm{UK}_{2,j,i,\mathrm{msg}} = g^{\omega_{j'} - a_j\omega_i}\right\}. \tag{9}$$

When the row index satisfies $(j,i) \in I_{3,M'}$ (marked as Module 3), the algorithm randomly picks $r_j' \in \mathbb{Z}_p$ and generates the update key as

$$\left\{\mathrm{UK}_{j,i,\mathrm{msg}}\right\}_3 = \left\{\begin{array}{c} \mathrm{UK}_{1,j,i,\mathrm{msg}} = g^{\lambda_j + \alpha_{\delta'(j)} r_{j'}}, \\ \mathrm{UK}_{2,j,i,\mathrm{msg}} = g^{\omega_{j'}}, \\ \mathrm{UK}_{3,j,i,\mathrm{msg}} = g^{\beta_{\delta'(j)} r_{j'}}, \\ \mathrm{UK}_{4,j,i,\mathrm{msg}} = H_2(\rho'(j))^{r_{j'}}, \\ \mathrm{UK}_{5,j,i,\mathrm{msg}} = g^{r_{j'}} \end{array}\right\}. \tag{10}$$

Finally, the data owner sends the updated key $\mathrm{UK}_{\mathrm{msg}}$ to the cloud storage service provider with $\mathrm{UK}_{\mathrm{msg}} = \left\{\left\{\mathrm{UK}_{j,i,\mathrm{msg}}\right\}_1, \left\{\mathrm{UK}_{j,i,\mathrm{msg}}\right\}_2, \left\{\mathrm{UK}_{j,i,\mathrm{msg}}\right\}_3\right\}$.

CTUpdate$(\mathrm{CT}, \mathrm{UK}_{\mathrm{msg}}) \longrightarrow \mathrm{CT}'$: after the cloud storage service provider receives the update key, it updates the ciphertext CT to CT′. By doing so, the cloud storage service provider cannot obtain relevant information during the re-encryption process of the ciphertext. The specific updates are as follows:

When the row index belongs to module 1, the update parameter is

$$\begin{aligned} C_{1,j}' &= C_{1,i} \cdot e\left(g, \mathrm{UK}_{1,j,i,\mathrm{msg}}\right) = e(g,g)^{\lambda_{j'} + \alpha_{\delta'(j)} r_{j'}}, \\ C_{2,j}' &= C_{2,i} \cdot \mathrm{UK}_{2,j,i,\mathrm{msg}} = g^{\omega_{j'}}, \\ C_{3,j}' &= C_{3,i} = g^{\beta_{\delta'(j)} r_{j'}}, \\ C_{4,j}' &= C_{4,i} = H_2(\rho'(j))^{r_{j'}}, \\ C_{5,j}' &= C_{5,i} = g^{-r_{j'}}, \\ r_j' &= r_i, \delta'(j) = \delta(i) = H_2(\rho'(j)) = H_2(\rho(i)). \end{aligned} \tag{11}$$

When the row index belongs to module 2, the update parameter is

$$\begin{aligned} C_{1,j}' &= \left(C_{1,i}\right)^{a_j} \cdot e\left(g, \mathrm{UK}_{1,j,i,\mathrm{msg}}\right) = e(g,g)^{\lambda_{j'} + \alpha_{\delta'(j)} r_{j'}}, \\ C_{2,j}' &= \left(C_{2,i}\right)^{a_j} \cdot \mathrm{UK}_{2,j,i,\mathrm{msg}} = g^{\omega_{j'}}, \\ C_{3,j}' &= \left(C_{3,i}\right)^{a_j} = g^{\beta_{\delta(i)} r_i a_j} = g^{\beta_{\delta'(j)} r_{j'}}, \\ C_{4,j}' &= \left(C_{4,i}\right)^{a_j} = H_2(\rho(i))^{r_i a_j} = H_2(\rho'(j))^{r_{j'}}, \\ C_{5,j}' &= \left(C_{5,i}\right)^{a_j} = g^{-r_i a_j} = g^{-r_{j'}}, \\ r_j' &= r_i a_j, \delta'(j) = \delta(i). \end{aligned} \tag{12}$$

When the row index belongs to module 3, the update parameter is

$$\begin{aligned} C_{1,j}' &= e\left(g, \mathrm{UK}_{1,j,i,\mathrm{msg}}\right) = e(g,g)^{\lambda_{j'} + \alpha_{\delta'(j)} r_{j'}}, \\ C_{2,j}' &= \mathrm{UK}_{2,j,i,\mathrm{msg}} = g^{\omega_{j'}}, \\ C_{3,j}' &= \mathrm{UK}_{3,j,i,\mathrm{msg}} = g^{\beta_{\delta'(j)} r_{j'}}, \\ C_{4,j}' &= \mathrm{UK}_{4,j,i,\mathrm{msg}} = H_2(\rho'(j)) r_j', \\ C_{5,j}' &= \mathrm{UK}_{5,j,i,\mathrm{msg}} = g^{-r_{j'}}, \\ r_j' &= r_i a_j, \delta'(j) = \delta(i). \end{aligned} \tag{13}$$

Finally, the updated ciphertext CT' is $\mathrm{CT}' = \left\{C_0, \left\{C_{1,j}', C_{2,j}', C_{3,j}', C_{4,j}', C_{5,j}'\right\}_{j \in \{1,\ldots,l'\}}\right\}$.

Trace$(\mathrm{SK}_{S,\mathrm{gid}}, \{\mathrm{PK}_{\mathrm{aid}}\}, \mathrm{PP}) \longrightarrow$ gid or $\perp$: the algorithm inputs the decryption key $\mathrm{SK}_{S,\mathrm{gid}}$ and the public key $\{\mathrm{PK}_{\mathrm{aid}}\}$ associated with the global public parameter PP. If the decryption key $\mathrm{SK}_{S,\mathrm{gid}}$ is not in the form $\mathrm{SK}_{S,\mathrm{gid}} = \left\{\left\{K_{1,i,\mathrm{gid}}, K_{2,i,\mathrm{gid}}, K_{3,i,\mathrm{gid}}, K_{4,i,\mathrm{gid}}\right\}_{i \in S}, K_{5,\mathrm{gid}}\right\}$ or cannot pass the key integrity check, the algorithm will output a special symbol to indicate that there is no need to trace $\mathrm{SK}_{S,\mathrm{gid}}$. The key integrity check of this scheme is as follows:

$$\begin{aligned} &K_{1,i,\mathrm{gid}}, K_{2,i,\mathrm{gid}}, K_{3,i,\mathrm{gid}}, K_{4,i,\mathrm{gid}} \in G, K_{5,\mathrm{gid}} \in \mathbb{Z}_p^*, \\ &e\left(K_{2,i,\mathrm{gid}}, g\right) = e\left(K_{3,i,\mathrm{gid}}, g^{\gamma_{\mathrm{aid}}}\right), \\ &e\left(K_{1,i,\mathrm{gid}}, g\right) = e(g,g)^{\alpha_{\mathrm{aid}}} \cdot e\left(H\left(K_{5,\mathrm{gid}}\right), g^{\beta_{\mathrm{aid}}}\right) \cdot e\left(F(i), K_{4,i,\mathrm{gid}}\right) \cdot e\left(K_{2,\rho(x),\mathrm{gid}} \cdot K_{3,\rho(x),\mathrm{gid}}^{K_{5,\mathrm{gid}}}, g^{\beta_{\mathrm{aid}}}\right). \end{aligned} \tag{14}$$

If there is an attribute $i \in S$ that satisfies equations (14), it is considered that the key $\mathrm{SK}_{S,\mathrm{gid}}$ passes the integrity check, and the identity gid is output as the trace identity.

5.2. Correctness. The correctness of our scheme can be obtained from the following equations. It is known that

$$\begin{aligned} D_x = {} & C_{1,x} \cdot e\left(H_1\left(K_{5,\mathrm{gid}}\right), C_{2,x} \cdot C_{3,x}\right) \\ & \cdot e\left(K_{2,\rho(x),\mathrm{gid}} \cdot K_{3,\rho(x),\mathrm{gid}}^{K_{5,\mathrm{gid}}}, C_{3,x}\right) \cdot e\left(C_{4,x}, K_{4,\rho(x),\mathrm{gid}}\right) \\ & \cdot e\left(K_{1,\rho(x),\mathrm{gid}}, C_{5,x}\right). \end{aligned} \tag{15}$$

According to the corresponding values of CT and $\mathrm{SK}_{S,\mathrm{gid}}$, we can obtain

$$
\begin{aligned}
D_x &= e(g,g)^{\lambda_x+\alpha_{\delta(x)}r_x}\cdot e\left(H_1(\mathrm{gid}), g^{\omega_x}\cdot g^{\beta_{\delta(x)}r_x}\right)\\
&\quad\cdot e\left(u^{\gamma_{\delta(x)}}\cdot u^{\mathrm{gid}}, g^{\beta_{\delta(x)}r_x}\right)\\
&\quad\cdot e\left(H_2(\rho(x))^{r_x}, g^{t}\right)\cdot e\left(g^{\alpha_{\delta(x)}}\cdot H_1(\mathrm{gid})^{\beta_{\delta(x)}}\right.\\
&\quad\left.\cdot H_2(i)^{t}\cdot u^{\beta_{\delta(x)}(\mathrm{gid}+\gamma_{\delta(x)})}, g^{-r_x}\right),\\
&= e(g,g)^{\lambda_x+\alpha_{\delta(x)}r_x}\cdot e\left(H_1(\mathrm{gid}),g\right)^{\omega_x+\beta_{\delta(x)}r_x}\\
&\quad\cdot e(u,g)^{(\gamma_{\delta(x)}+\mathrm{gid})\beta_{\delta(x)}r_x}\cdot e\left(H_2(\rho(x)),g\right)^{r_x t}\\
&\quad\cdot e(g,g)^{-\alpha_{\delta(x)}r_x}\cdot e\left(H_1(\mathrm{gid}),g\right)^{-\beta_{\delta(x)}r_x}\\
&\quad\cdot e\left(H_2(\rho(x)),g\right)^{-r_x t}\cdot e(u,g)^{-(\gamma_{\delta(x)}+\mathrm{gid})\beta_{\delta(x)}r_x},\\
&= e(g,g)^{\lambda_x}\cdot e\left(H_1(\mathrm{gid}),g\right)^{\omega_x}.
\end{aligned}
\tag{16}
$$

Then, for $\{c_x\}_{x\in I}$ and $\sum_{x\in I}c_x\mathbf{M}_x = (1,0,\ldots,0)$, we have

$$
\begin{aligned}
\sum_{x\in I}\lambda_x c_x &= \sum_{x\in I}\mathbf{M}_x\mathbf{v}^T c_x = (1,0,\ldots,0)\cdot\mathbf{v}^T = s,\\
\sum_{x\in I}\omega_x c_x &= \sum_{x\in I}\mathbf{M}_x\mathbf{v}^T c_x = (1,0,\ldots,0)\cdot\mathbf{v}^T = 0.
\end{aligned}
\tag{17}
$$

Hence, we have

$$
\begin{aligned}
\prod_{x\in I}D_x^{c_x} &= \prod_{x\in I}\left(e(g,g)^{\lambda_x}\cdot e\left(H_1(\mathrm{gid}),g\right)^{\omega_x}\right)^{c_x},\\
&= e(g,g)^{\sum_{x\in I}\lambda_x c_x}\cdot e(g,g)^{\sum_{x\in I}\omega_x c_x},\\
&= e(g,g)^{s}.
\end{aligned}
\tag{18}
$$

This proves that the message can be correctly restored to

$$
\mathrm{msg} = \frac{C_0}{\prod_{x\in I}D_x^{c_x}}. \tag{19}
$$

5.3. Security Analysis

Theorem 1. *Assume the CP-ABE system in [11] is statically secure; then, the T-DPU-MCP-ABE system is static secure with respect to Definition 1.*

Proof. For simplicity, we use Σ_{RW}, Σ_{tdpum} to denote the CP-ABE system in [11] and our T-DPU-MCP-ABE system, respectively. We suppose there exists a static polynomial time attacker $\mathscr{A}$ that breaks Σ_{RW} with a nonnegligible advantage in selectively with a challenge LSSS access policy (M^*,ρ^*), where M^* is an $l\times n$ matrix. We will build a PPT algorithm $\mathscr{B}$ that breaks Σ_{tdpum} with a nonnegligible advantage.

Init: $\mathscr{B}$ gets a challenge LSSS access policy (M^*,ρ^*) from $\mathscr{A}$ and transmits the received (M^*,ρ^*) to the Σ_e challenger $\mathscr{C}$.

Setup: $\mathscr{C}$ generates the common parameter $\mathrm{PP} = (G, G_T, p, e, g, H_1, H_2, T)$ and sends it to $\mathscr{A}$.

Query: $\mathscr{B}$ initializes an integer counter $j = 0$ and an empty table T. Then, $\mathscr{A}$ makes the following queries:

Receiving $\mathscr{A}$'s decryption key query with an attribute does not satisfy (M^*,ρ^*), $\mathscr{B}$ sets the attribute as S_j and $j = j+1$, then sends them to the Σ_{tdpum} challenger, and obtains a secret key $\mathrm{SK}'_{S,\mathrm{gid}} = (\{K'_{1,\tau,\mathrm{gid}}, K'_{2,\tau,\mathrm{gid}}, K'_{3,\tau,\mathrm{gid}}, K'_{4,\tau,\mathrm{gid}}\}_{\tau\in[|S|]}, K'_{5,\mathrm{gid}})$. $\mathscr{A}$ chooses a corrupted AA $C_{\mathrm{aid}}\in U_{\mathrm{aid}}$ and generates the corresponding public key $\mathrm{PK}'_{\mathrm{aid}} = (e(g,g)^{\mathrm{aid}}, g^{\beta_{\mathrm{aid}}})$ in S_{RW}. Also, for each $\mathrm{aid}\in C_{\mathrm{aid}}$, $\mathscr{A}$ randomly chooses $\gamma_{\mathrm{aid}}\in\mathbb{Z}_p^*$ and generates the system public key $\mathrm{PK}_{\mathrm{aid}} = (e(g,g)^{\mathrm{aid}}, g^{\beta_{\mathrm{aid}}}, g^{\gamma_{\mathrm{aid}}})$. Then, $\mathscr{A}$ responses for the normal AA N_{aid}, the corrupted AA C_{aid} by interacting with $\mathscr{B}$ as follows. $\mathscr{A}$ requires $\{(\mathrm{gid}_j, S_j)\}_{j\in[m]}$, where $S_j\subset U$ is the corresponding attribute set of user gid_j. All users' gid_j are unique and for arbitrary $i\in S$, we have $T(i)\notin C_{\mathrm{aid}}$. Then, $\mathscr{A}$ fixes a coin $b\in\{0,1\}$, which is used to generates message msg_0 or msg_1 with the same length. $\mathscr{A}$ chooses a set of challenge $\{(M_1,\rho),\ldots,(M_q,\rho_q)\}$. Finally, $\mathscr{A}$ sends all the chosen parameters to $\mathscr{B}$.

Challenge: $\mathscr{A}$ chooses two same length messages (m_0,m_1) and sends to $\mathscr{B}$. Then, $\mathscr{B}$ submits (m_0,m_1) to the Σ_{tdpum} challenger, obtains a challenge common public key $PK'_{\mathrm{aid}} = (e(g,g)^{\mathrm{aid}}, g^{\beta_{\mathrm{aid}}})$, and generates a ciphertext $\mathrm{ct}^* = (C_0^*\{C_{1,x}^*, C_{2,x}^*, C_{3,x}^*, C_{4,x}^*, C_{5,x}^*\}_{x\in\{1,\ldots,l\}})$. $\mathscr{B}$ chooses a random bit $b_{\mathscr{B}}\in\{0,1\}$, computes $\mathrm{key}_{b_{\mathscr{B}}} = C^*/m_{b_{\mathscr{B}}}$, and sends the new ciphertext $\mathrm{ct}^* = (C_0^*\{C_{1,x}^*, C_{2,x}^*, C_{3,x}^*, C_{4,x}^*, C_{5,x}^*\}_{x\in\{1,\ldots,l\}})$ to $\mathscr{A}$.

Guess: finally, after receiving the abovementioned responses, $\mathscr{A}$ outputs a guess $b_{\mathscr{A}}\in\{0,1\}$. If $b_{\mathscr{A}} = 1$, it means that $\mathscr{A}$ guesses that $\mathrm{key}_{b_{\mathscr{B}}}$ is a random key, and $\mathscr{B}$ outputs $1-b_{\mathscr{B}}$. If $b_{\mathscr{A}} = 0$, meaning that $\mathscr{A}$ guesses that $\mathrm{key}_{b_{\mathscr{B}}}$ is the key from $\mathrm{ct}^*_{\mathrm{new}}$, $\mathscr{B}$ outputs $b_{\mathscr{B}}$.

Since the real system is the same as the distributions of the challenge ciphertext, if $\mathscr{A}$ breaks the security of S_{RW} with a nonnegligible advantage, then the simulator $\mathscr{B}$ can selectively break S_{tdpum} with the same advantage. □

Theorem 2. *Assume the CL signature scheme in [25] is against existing forgery, and the T-DPU-MCP-ABE system in Section 5.1 is traceable secure with respect to Definition 2.*

Proof. The security proof of the T-DPU-MCP-ABE system with respect to Definition 2 (i.e., for traceable adversary) is identical to the abovementioned proof except that the adversary runs the **Forgery** phase instead of the **Guess** phase. Here, we suppose there exists a PPT attacker $\mathscr{A}$ that selectively breaks the CL scheme with a nonnegligible advantage. We can build a PPT simulator algorithm $\mathscr{B}$ that selectively breaks Σ_{tdpum} with a nonnegligible advantage. It is proved that the CL scheme is secure against existential forgery under adaptive chosen message attack with LRSW assumption.

Setup: the CL scheme challenger $\mathscr{C}$ delivers each authority's public keys $\{G, G_T, p, g, g^{\beta_{\mathrm{aid}}}, g^{\gamma_{\mathrm{aid}}}\}$ to the simulator algorithm $\mathscr{B}$. $\mathscr{B}$ chooses random values

$\alpha_{\text{aid}} \in \mathbb{Z}_p^*$ for each authority, runs $\text{Setup}(\lambda)$ and $\text{AuthoritySetup}(\text{aid}, \text{PP})$ to generate the public key $\text{PK}_{\text{aid}} = \{e(g,g)^{\alpha_{\text{aid}}}, g^{\beta_{\text{aid}}}, g^{\gamma_{\text{aid}}}\}$, and sends the public parameter PP and the authority public key PK_{aid} to $\mathscr{A}$. The two hash functions H_1 and H_2 of our scheme are managed by simulator $\mathscr{B}$.

Query. $\mathscr{A}$ requests $\{(\text{gid}_j, S_j)\}_{j\in[m]}$ as the relevant private key, where $S_j \subseteq U$ means the attribute set of the user gid_j. Before $\mathscr{A}$ forges the key, to maintain hash functions H_1 and H_2, $\mathscr{B}$ will set two empty tables, T_1 and T_2, respectively, and update them according to the query of $\mathscr{A}$. When the gid queried by $\mathscr{A}$ does not exist in the table of T_1 and T_2, $\mathscr{B}$ will select a random element $t_{gid} \in \mathbb{Z}_p^*$ and a random element $t_i \in \mathbb{Z}_p^*$ and then record $(t_{\text{gid}}, g^{t_{\text{gid}}})$ and (t_i, g^{t_i}) with T_1 and T_2, respectively. At the same time, simulator $\mathscr{B}$ will return the hash value of H_1 or H_2 according to opponent the query of $\mathscr{A}$. For each $i \in S_j$, if the attribute authority $\text{aid} = T(i)$, then B will submit $(\text{gid}_j, \text{aid})$ to Challenger C according to the query of $\mathscr{A}$ so as to obtain the signature $(u, u^{\gamma_{\text{aid}}}, u^{\beta_{\text{aid}}\cdot((\gamma_{\text{aid}}/\text{gid}_j)+1)})$ in the CL scheme. Then, $\mathscr{B}$ takes the random value $t \in \mathbb{Z}_p^*$ and runs $\text{KeyGen}(\text{gid}_j, S_j, \text{SK}_{\text{aid}}, \text{PP})$ as well as sends $\left\{\text{SK}_{S_j,\text{gid}_j}\right\}_{j\in[m]}$ to $\mathscr{A}$. In this step, $\mathscr{B}$ should computes the following:

$$\begin{aligned}
K_{1,i,\text{gid}} &= g^{\alpha_{\text{aid}}} \cdot H_1\left(\text{gid}_j\right)^{\beta_{\text{aid}}} \cdot H_2(i)^t \cdot u^{\beta_{\text{aid}}(\text{gid}+\gamma_{\text{aid}})},\\
K_{2,i,\text{gid}} &= u^{\gamma_{\text{aid}}},\\
K_{3,i,\text{gid}} &= u,\\
K_{4,i,\text{gid}} &= g^t,\\
K_{5,\text{gid}} &= \text{gid}.
\end{aligned} \tag{20}$$

Then, the final calculation is $\text{SK}_{S_j,\text{gid}_j}$ as $\left\{\left\{K_{1,i,\text{gid}_j}, K_{2,i,\text{gid}_j}, K_{3,i,\text{gid}_j}, K_{4,i,\text{gid}_j}\right\}_{i\in S_j}, K_{5,\text{gid}_j}\right\}$.

Forgery. in this step, $\mathscr{A}$ already queries from simulator $\mathscr{B}$ the value of $H_1(\text{gid})$ and $H_2(i)$ and obtains $H_1(\text{gid})$ as $g^{t_{K_{5,\text{gid}}}}$ and $H_2(i)$ as g^{t_i}. $\mathscr{A}$ assumes the unknown $K_{3,i,\text{gid}} = g^{t_3}$ and $K_{4,i,\text{gid}} = g^{t_4}$. Through formula (14) in Section 5.1, we could get that $K_{2,i,\text{gid}} = (K_{3,i,\text{gid}}) = g^{t_3\gamma_{\text{aid}}}$. Also through formula (14) in Section 5.1, we could get that $K_{1,i,\text{gid}} = g^{\alpha_{\text{aid}}+t_{K_{5,\text{gid}}}\beta_{\text{aid}}} \cdot (K_{4,i,\text{gid}})^{t_i} \cdot (K_{3,i,\text{gid}})^{\beta_{\text{aid}}(K_{5,\text{gid}}+\gamma_{\text{aid}})}$.

Then, $\mathscr{B}$ calculates a legal signature σ according to the CL scheme, and the calculation process is as follows:

$$\begin{aligned}
\sigma_1 &= \frac{K_{1,i,\text{gid}}}{g^{\alpha_{\text{aid}}+t_{K_{5,\text{gid}}}\beta_{\text{aid}}} \cdot \left(K_{4,i,\text{gid}}\right)^{t_i}},\\
&= \left(K_{3,i,\text{gid}}\right)^{\beta_{\text{aid}}\left(K_{5,\text{gid}}+\gamma_{\text{aid}}\right)}.
\end{aligned} \tag{21}$$

Then, $\mathscr{A}$ picks a gid as a message and gives $\sigma = (K_{3,i,\text{gid}}, K_{2,i,\text{gid}}, (\sigma_1/K_{3,i,\text{gid}}^{\text{gid}}))$ as the signature of the message gid according to the CL scheme.

Finally, $\mathscr{A}$ outputs a forgery secret key SK^*, if $\text{Trace}(\text{SK}_{S,\text{gid}}, \{\text{PK}_{\text{aid}}\}, \text{PP}) \notin \Delta$ and $\text{gid} \notin \{\text{gid}_1, \ldots, \text{gid}_m\}$. As $\text{gid} \notin \{\text{gid}_1, \ldots, \text{gid}_m\}$, we know that the signature of message gid is not invoked by $\mathscr{B}$ yet. Thus, the simulator $\mathscr{B}$ breaks the CL scheme with the same advantage.

Since in the abovementioned game the whole system has the decryption keys, the distributions of the public parameters, and challenge ciphertext, if $\mathscr{A}$ breaks the security of the CL scheme, then the simulator $\mathscr{B}$ can selectively break S_{tdpum} with the same advantage. Hence, if the LRSW assumption holds true, the proposed cryptosystem is against forgery, meaning that our scheme is traceable secure for the adversary. □

5.4. Proof of Collusion Prevention. In our scheme, we use the unique gid and construct the hash function value corresponding to gid to resist collusion attack, which has been proved to be feasible by Allison and Waters [14]. In the process of decryption, the data user needs to calculate $D_x = e(g,g)_x^{\lambda} \cdot e(H_1(\text{gid}), g)^{\omega_x}$. For a single user with the access policy satisfaction attribute set, since ω_x are the shares of secret value 0, $e(H_1(\text{gid}), g)^{\omega_x}$ can be eliminated, where $e(H_1(\text{gid}), g)^{\omega_x} = 1$. In case of collusion attack, two or more users will have different gid; thus, the value of $H_1(\text{gid})$ will also be different; $e(H_1(\text{gid}), g)^{\omega_x}$ with a secret value of 0 cannot be constructed, and thus, it cannot be eliminated. Therefore, two or more users cannot share their attribute key values to generate collusion attacks, which means this scheme is resistant to collusion attack.

6. Performance Evaluations

6.1. Theoretical Analysis. We first theoretically make a comparison of our scheme with others. The comparison of feature and performance of our work and related works is given in Tables 1 and 2 .

It can be seen from Table 1 that the YB scheme [11] does not realize the traceability, nor does it have the function of dynamic access policy update; although the JZXL scheme [27] has both traceability and large attribute domains, it is constructed based on composite orders and is a single authorization which will become a bottleneck. Since the QLZH scheme [28] and the YLLT scheme [29] are based on tree access structure, they do not have the functions of large attribute domain, dynamic access strategy update, and traceability. The YLMH scheme [30] can realize the dynamic access strategy update but does not support traceability; while the ZLML scheme [31] does not have the function of dynamic access policy update. Compared with the abovementioned related schemes, our scheme not only supports traceability, large attribute domain, and dynamic access policy update at the same time under multiple authorization agencies but also is based on the prime order bilinear group structure, which is more efficient.

Let G and G_T be the size of elements in G and an exponentiation in G_T, respectively. Let e be a pairing and exp be the maximum amounts of time to compute an exponentiation in G. Let A be the number of ciphertext attributes, $|S|$ be the size of the attribute set of a private key, and l be the

Table 1: Characteristics comparison of ABE schemes.

	YB [11]	JZXL [27]	QLZH [28]	YLLT [29]	YLMH [30]	ZLML [31]	Ours
Order groups	Prime	Composite	Prime	Prime	Prime	Prime	Prime
Large universe	√	√	×	×	×	√	√
Policy updating	×	×	×	×	√	×	√
Traceable	×	√	×	×	×	√	√
Access structure	LSSS	LSSS	TREE	TREE	LSSS	LSSS	LSSS
Multiauthority	√	×	√	√	√	√	√

Table 2: Performance comparison of multiauthority ABE schemes (https://github.com/monzxcv/ABE).

	YB [11]	SPB [32]	YLMH [30]	ZLML [31]	Ours
AA's public key	$G + G_T$	$3G + G_T$	$(n_i + 1)G + G_T$	$3G + G_T$	$2G + G_T$
User's private key	$2\|S\|G$	$4\|S\|G + G_T$	$2\|S\|G + G_T$	$4\|S\|G$	$4\|S\|G + G_T$
Ciphertext	$3lG + (l + 1)G_T$	$4lG + (l + 1)G_T$	$(2l + 1)G + G_T$	$5lG + (l + 1)G_T$	$4lG + (l + 1)G_T$
Encryption cost	$3l\exp + (l + 1)e$	$4l\exp + (l + 1)e$	$(2l + 1)\exp + e$	$5l\exp + (l + 1)e$	$4l\exp + (l + 1)e$
Decryption cost	$3\|I\|$	$4\|I\|$	$2\|S\| + 2\|I\|$	$3\|I\|$	$4\|I\|$
Security assumption	q-type	q-type	q-PBDHE	q-type	q-type

output size of a function. Let I be the number of rows of the matrix when decrypting.

In Table 2, we show the communication cost and the computing cost comparison. Compared with other solutions, our scheme is relatively better in the process of adding multiple functions. On the one hand, for the communication cost, we can draw the following conclusions: Firstly, our scheme has the advantages in the length of the private key that our scheme supports big attribute universe. More precisely, the public key of our scheme does not increase linearly with the size of the attribute domain in an attribute authority, while that of the YLMH scheme will, and the storage occupied by our public key is smaller than that of the SPB scheme [32] and the ZLML scheme. Secondly, although the user's private keys in the YB scheme and the YLMH scheme are relatively small, none of these schemes support traceability. In order to enhance the security of the system, the scheme in this paper supports the traceability function, and the user's private key does not increase too much. Furthermore, compared to the YLMH scheme and the ZLML scheme, the length of the ciphertext in our scheme is optimized, which is only linearly related to the number of rows from the generator matrix. On the other hand, for the calculation cost, our scheme supports an access strategy update algorithm, while the YB scheme and the YLMH scheme do not support this function. Finally, for the decryption cost, our scheme is much smaller than that of the YLMH scheme. The decryption cost in our scheme is only related to the number of attribute organizations where the attributes belong. Although the decryption cost in our scheme is slightly higher than that of the YB scheme and the ZLML scheme, the YB scheme does not support traceability and the ZLML scheme does not support access policy update.

6.2. Experimental Analysis. In this section, we conduct a simulation experiment to evaluate the comparison of our scheme and the baseline algorithms (the simulation code is available in (https://github.com/monzxcv/ABE)). We select the scheme in [11] (YB scheme) and the scheme in [30] (YLMH scheme) as our baseline algorithms and run the experiments in five aspects: system initialization, key generation, data encryption, user decryption, and access strategy re-encryption. All the experiments are run on a 64-bit operating system of the Ubuntu 14.04 platform with a core 1.8 GHz processor and 4 GB RAM. We used Charm version 0.50 and Python version 3.7 as our program languages. We first convert the YB scheme, YLMH scheme, and our scheme into asymmetric bilinear mapping and use the famous supersingular symmetric elliptic curve group ("SS512"). Then, in the process of encryption and decryption, the YB scheme, YLMH scheme, and our scheme are only related to the number of access policy attributes. Therefore, in this experiment, we change the number of user attributes and calculate the time of system initialization and user key generation under the same condition to get our first comparison. In addition, we change the access policy and calculate the time of the user encryption and decryption to get another comparison. Finally, the time consumed for updating ciphertext under the same condition is calculated. The experimental attributes are constructed with $A_N, N \in [1, \ldots, 50]$. The strategy set is selected $(A_1 \wedge A_2 \wedge \cdots \wedge A_N)$. We increase the number of attributes from 5 to 50, and there are ten different access strategies. In order to ensure the accuracy of the conclusion, every experiment is run 15 times.

The system initialization cost and the average time cost of user private key generation are shown in Figures 2 and 3 when the number of attributes varies from 5 to 50. We fix the number of AAs in 8, and we also fix the number of attributes in the access policy in 8. Since both our scheme and YB scheme support large attribute domains, the system initialization process has nothing to do with the number of attributes, as is verified in Figure 2. It can be seen that as the number of attributes increases, the cost of the YLMH scheme increases, and the cost of our scheme still keeps a constant value, so the larger the number of attributes, the more the advantage in our scheme. It can be seen from Figure 3 that the cost of the user private key generation time in all the three schemes increases linearly with the increase of

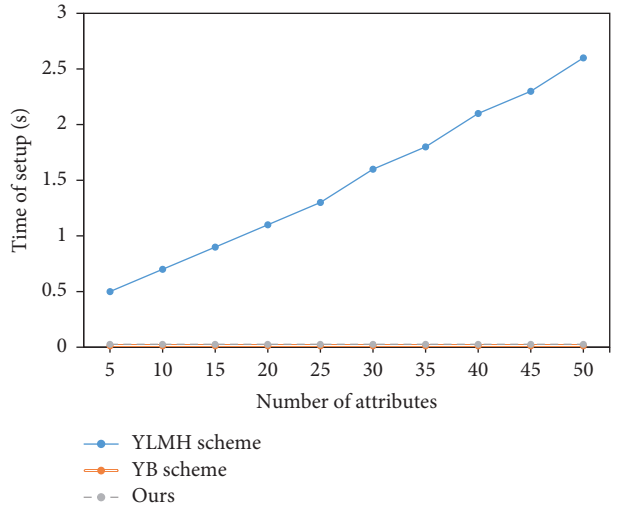

Figure 2: Comparison of the system setup process.

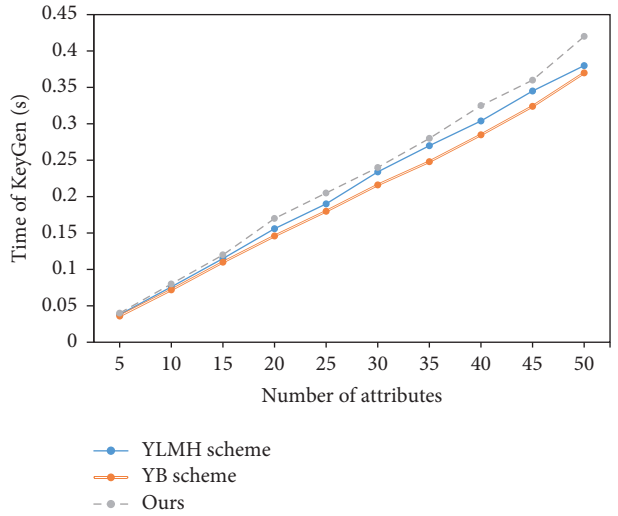

Figure 3: Comparison of the KeyGen process.

attributes. This is because each attribute in the user's private key must be calculated accordingly. Finally, the generation time cost is not much different from that of the YB scheme and the YLMH scheme.

Figure 4 shows the average time cost of the encryption and decryption process when the number of attributes used in the access policy varies from 5 to 50. We fix the number of AAs in 8, and the number of attributes for each user is also fixed in 8. It can be seen from Figure 4 that the average execution time of the key generation and encryption/decryption process of the proposed scheme is equivalent to that of the YB scheme, while our scheme is more practical than the YB scheme, such as supporting traceability and dynamic access policy update. Although the YLMH scheme's encryption cost is the smallest, its decryption cost is the largest among the three schemes and is related to the number of attributes the user has. If the user's attributes increase, the decryption time cost of the YLMH scheme will be higher.

Figure 5 shows the algorithms' average computing time in the YB scheme, YLMH scheme, and our scheme in policy update. Since the YB scheme does not support dynamic strategy updates, we use the traditional update method. There are three modes for updating of dynamic strategy in the YLMH scheme and our scheme. We use mode 3 (which has the highest cost) for comparison. In addition, the number of AAs is fixed in 8, and the number of attributes for each user is also fixed in 8. We vary the number of attributes by 5, 10, and 15. As it can be seen from Figure 5, our scheme and YLMH scheme can dynamically update the strategy. Thus, the time cost is less than that of the YB scheme. Although our scheme costs slightly more than the YLMH scheme, our scheme supports traceability, which is considered to be more practical.

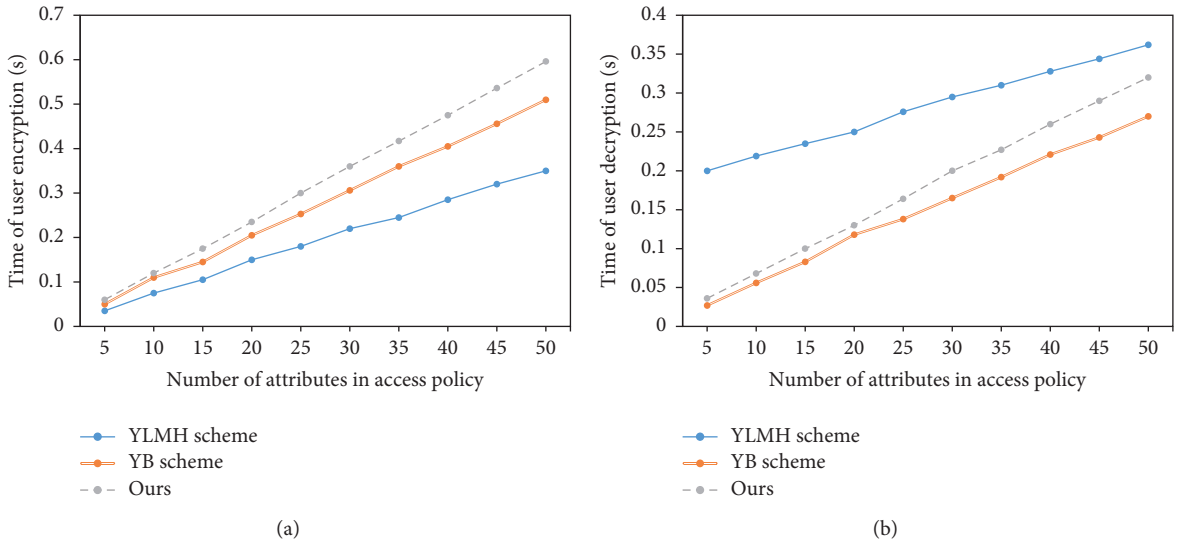

Figure 4: Comparison of encryption algorithms and decryption algorithms.

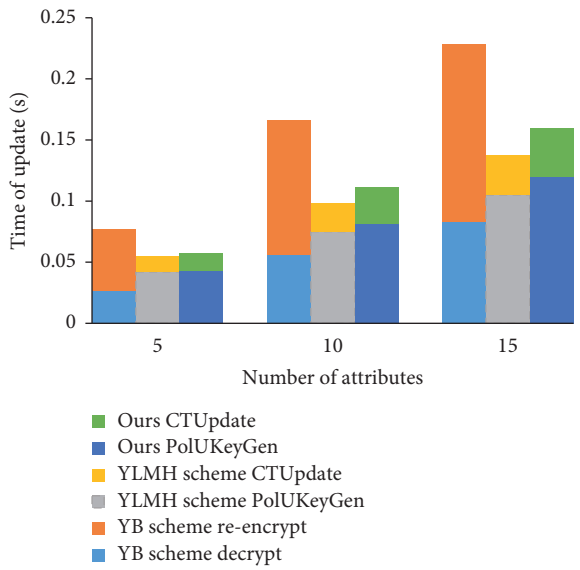

Figure 5: Comparison of policy updating.

7. Conclusions and Future Work

Regarding the three problems in the CP-ABE scheme of multiauthority, traceability, and the flexibility in changing the access policy, we propose a scheme to achieve good solutions. Our scheme supports multiple authorities, white box traceability, large attribute domains, access policy updates, and high expressiveness. Then, we prove that our scheme is static secure and traceable secure based on the state-of-the-art security models. By supporting the traceability, there is no need to maintain the authorized institution's identity table; thus, our solution is more practical. The experimental results indicate that our scheme has efficient performance while enjoying the abovementioned features. In future work, we plan to conduct a study on computational outsourcing and hidden access strategies for CP-ABE.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was partially supported by the Key Areas Research and Development Program of Guangdong Province (Grant no. 2019B010139002), the Project of Guangzhou Science and Technology (Grant no. 202007010004), and National Natural Science Foundation of China (Grant no. 61902079 and 62002136).

References

[1] E. K. Wang, C.-M. Chen, M. M. Hassan, and A. Almogren, "A deep learning based medical image segmentation technique in internet-of-medical-things domain," *Future Generation Computer Systems*, vol. 108, pp. 135–144, 2020.

[2] C.-M. Chen, Y. Huang, K.-H. Wang, S. Kumari, and M.-E. Wu, "A secure authenticated and key exchange scheme for fog computing," *Enterprise Information Systems*, vol. 14, pp. 1–16, 2020.

[3] S. Amit and B. Waters, "Fuzzy identity-based encryption," in *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 457–473, Aarhus, Denmark, May 2005.

[4] V. Goyal, O. Pandey, S. Amit, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM Conference on Computer and Communications Security*, pp. 89–98, Alexandria, VA, USA, November 2006.

[5] J. Bethencourt, S. Amit, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 321–334, Oakland, CL, USA, September 2007.

[6] X. Liu, J. Ma, J. Xiong, and G. Liu, "Ciphertext-policy hierarchical attribute-based encryption for fine-grained access control of encryption data," *IJ Network Security*, vol. 16, no. 6, pp. 437–443, 2014.

[7] H. Deng, Q. Wu, B. Qin et al., "Ciphertext-policy hierarchical attribute-based encryption with short ciphertexts," *Information Sciences*, vol. 275, pp. 370–384, 2014.

[8] S. L. Wang, J. P. Yu, P. Zhang, and P. Wang, "A novel file hierarchy access control scheme using attribute-based encryption," in *Applied Mechanics and Materials*Trans Tech Publ, Stafa-Zurich, Switzerland, 2015.

[9] Z. Liu, Z. Cao, and S. Duncan, "White-box traceable ciphertext-policy attribute-based encryption supporting any monotone access structures," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 1, pp. 76–88, 2012.

[10] K. Liang, W. Susilo, D. S. Wong et al., "A secure and efficient ciphertext-policy attribute-based proxy re-encryption for cloud data sharing," *Future Generation Computer Systems*, vol. 52, pp. 95–108, 2015.

[11] Y. Rouselakis and B. Waters, "Efficient statically-secure large-universe multi-authority attribute-based encryption," in *Proceedings of the International Conference on Financial Cryptography and Data Security*, pp. 315–332, Juan, Puerto Rico, November 2015.

[12] L. Anna, R. L. Rivest, S. Amit, and S. Wolf, "Pseudonym systems," in *International Workshop on Selected Areas in Cryptography,* Springer, Berlin, Germany, 1999.

[13] M. Chase, "Multi-authority attribute based encryption," in *Proceedings of the Theory of Cryptography Conference*, pp. 515–534, Amsterdam, Netherland, February 2007.

[14] L. Allison and B. Waters, "Decentralizing attribute-based encryption," in *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 568–588, Tallinn, Estonia, May 2011.

[15] Y. Rouselakis and B. Waters, "Practical constructions and new proof methods for large universe attribute-based encryption," in *Proceedings of the ACM SIGSAC Conference on Computer & Communications Security*, pp. 463–474, London, UK, November 2013.

[16] Z. Liu and D. S. Wong, "Practical attribute-based encryption: traitor tracing, revocation and large universe," *The Computer Journal*, vol. 59, no. 7, pp. 983–1004, 2016.

[17] J. Ning, X. Dong, Z. Cao, L. Wei, and X. Lin, "White-box traceable ciphertext-policy attribute-based encryption supporting flexible attributes," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 6, pp. 1274–1288, 2015.

[18] L. Jin, Q. Huang, X. Chen, S. M. Sherman, S. Duncan, and D. Xie, "Multi-authority ciphertext-policy attribute-based encryption with accountability," in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, pp. 386–390, Hong Kong, China, March 2011.

[19] J. Zhou, Z. Cao, X. Dong, and X. Lin, "White-box traceable and revocable multi-authority attribute-based encryption and its applications to multi-level privacy-preserving e-healthcare cloud computing systems," in *Proceedings of the IEEE Conference on Computer Communications*, Hong Kong, China, March 2015.

[20] Z. Ying, H. Li, J. Ma, J. Zhang, and J. Cui, "Adaptively secure ciphertext-policy attribute-based encryption with dynamic policy updating," *Science China Information Sciences*, vol. 59, no. 4, Article ID 042701, 2016.

[21] Z. Liu, Z. L. Jiang, X. Wang, and S. M. Yiu, "Practical attribute-based encryption: outsourcing decryption, attribute revocation and policy updating," *Journal of Network and Computer Applications*, vol. 108, pp. 112–123, 2018.

[22] Y. Jiang, W. Susilo, Y. Mu, and F. Guo, "Ciphertext-policy attribute-based encryption supporting access policy update and its extension with preserved attributes," *International Journal of Information Security*, vol. 17, no. 5, pp. 533–548, 2018.

[23] M. V. Dijk, "A linear construction of secret sharing schemes," *Designs, Codes and Cryptography*, vol. 12, no. 2, pp. 161–201, 1997.

[24] J. Ning, Z. Cao, X. Dong, K. Liang, H. Ma, and L. Wei, "Auditable σ-time outsourced attribute-based encryption for access control in cloud computing," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 1, pp. 94–105, 2017.

[25] C. Jan and L. Anna, "Signature schemes and anonymous credentials from bilinear maps," in *Proceedings of the Annual International Cryptology Conference*, pp. 56–72, Santa Barbara, CA, USA, August 2004.

[26] K. Yang, X. Jia, and K. Ren, "Secure and verifiable policy update outsourcing for big data access control in the cloud," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 12, pp. 3461–3470, 2014.

[27] J. Ning, Z. Cao, X. Dong, and L. Wei, "White-box traceable cp-abe for cloud storage service: how to catch people leaking their access credentials effectively," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 5, pp. 883–897, 2016.

[28] H. Qian, J. Li, Y. Zhang, and J. Han, "Privacy-preserving personal health record using multi-authority attribute-based encryption with revocation," *International Journal of Information Security*, vol. 14, no. 6, pp. 487–497, 2015.

[29] X. Yan, Y. Liu, Z. Li, and Y. Tang, "Multi-authority attribute-based encryption scheme with privacy protection," *Journal of Computer Research and Development*, vol. 55, no. 4, p. 846, 2018.

[30] X. Yan, H. Ni, Y. Liu, and D. Han, "Privacy-preserving multi-authority attribute-based encryption with dynamic policy updating in phr," *Computer Science and Information Systems*, vol. 16, no. 3, pp. 831–847, 2019.

[31] K. Zhang, H. Li, J. Ma, and X. Liu, "Efficient large-universe multi-authority ciphertext-policy attribute-based encryption with white-box traceability," *Science China Information Sciences*, vol. 61, no. 3, Article ID 032102, 2018.

[32] K. Sethi, A. Pradhan, and P. Bera, "Practical traceable multi-authority cp-abe with outsourcing decryption and access policy updation," *Journal of Information Security and Applications*, vol. 51, Article ID 102435, 2020.