Hindawi
Security and Communication Networks
Volume 2020, Article ID 6661243, 14 pages
https://doi.org/10.1155/2020/6661243

Research Article

A Traceable and Revocable Multiauthority Attribute-Based Encryption Scheme with Fast Access

Kai Zhang,[1] **Yanping Li**,[1] **Yun Song**,[2] **Laifeng Lu**,[1] **Tao Zhang**,[3] **and Qi Jiang**[4,5]

[1]*School of Mathematics and Information Science, Shaanxi Normal University, Xi'an 710119, China*
[2]*School of Computer Science, Shaanxi Normal University, Xi'an 710119, China*
[3]*School of Computer Science and Technology, Xidian University, Xi'an, Shaanxi 710071, China*
[4]*Network Communication Research Centre, Peng Cheng Laboratory, Shenzhen 518055, Guangdong, China*
[5]*Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China*

Correspondence should be addressed to Yun Song; songyun09@snnu.edu.cn

Received 26 October 2020; Revised 23 November 2020; Accepted 2 December 2020; Published 22 December 2020

Academic Editor: Athanasios V. Vasilakos

Multiauthority ciphertext-policy attribute-based encryption (MA-CP-ABE) is a promising technique for secure data sharing in cloud storage. As multiple users with same attributes have same decryption privilege in MA-CP-ABE, the identity of the decryption key owner cannot be accurately traced by the exposed decryption key. This will lead to the key abuse problem, for example, the malicious users may sell their decryption keys to others. In this paper, we first present a traceable MA-CP-ABE scheme supporting fast access and malicious users' accountability. Then, we prove that the proposed scheme is adaptively secure under the symmetric external Diffie–Hellman assumption and fully traceable under the q-Strong Diffie–Hellman assumption. Finally, we design a traceable and revocable MA-CP-ABE system for secure and efficient cloud storage from the proposed scheme. When a malicious user leaks his decryption key, our proposed system can not only confirm his identity but also revoke his decryption privilege. Extensive efficiency analysis results indicate that our system requires only constant number of pairing operations for ciphertext data access.

1. Introduction

In recent years, the rise of the Internet of things [1] promotes the application and development of sensor technology [2–4]. As an important sensing paradigm, mobile crowdsensing [5] has been widely used in various industries due to its large coverage area and low deployment cost characteristics. One of the most significant services for mobile crowdsensing is cloud storage [6], which supports large-scale data sharing. In cloud storage, the individuals or organizations often need to share the sensitive data with the users whose attributes satisfy a specific policy. For example, a patient wants to share his medical data with nurses and doctors in neurosurgery, but he does not know the identities of the nurses and doctors. Security is a very important issue [7, 8] in the Internet, and a potential solution for achieving data security is to encrypt the sensitive data before sharing it by the cloud. Unfortunately, the traditional public key encryption [9] requires the data owner to know the receiver's exact identity, so it is not suitable for the above scenario.

To address this issue, ciphertext-policy attribute-based encryption (CP-ABE) [10, 11] was introduced as an expansion of the traditional public key encryption. In CP-ABE, the user's secret key is associated with his attributes, and the ciphertext is associated with an access policy, which is defined in the form of Boolean formula over a set of attributes; the user can decrypt the ciphertext only when his attributes satisfy the access policy. By using CP-ABE in the above example, the patient can encrypt the medical data with the access policy ("Doctor" AND "Neurosurgery") OR ("Nurse" AND "Neurosurgery") and upload the ciphertext to the cloud; then, only nurses and doctors in neurosurgery can access the medical data.

In the typical CP-ABE system, a single central authority should manage all attributes and generate all users' decryption keys. However, many scenarios require multiple authorities to manage different attribute domains. For instance, a patient wants to share his medical document with the users with the attribute "Doctor" that is issued by a hospital and attribute "Researcher" that is issued by a medical research institute. To solve this problem, Chase [12] introduced the multiauthority attribute-based encryption (MA-ABE), in which different authorities manage different attribute sets and each authority issues secret keys only for the attributes it manages. However, before the MA-ABE being applied in practice, there exist the following issues that need to be solved.

The standard MA-ABE suffers the decryption key abuse problem. In multiauthority ciphertext-policy attribute-based encryption (MA-CP-ABE), the decryption privilege is only based on the user's attributes and the ciphertext does not contain the user's identity information. Hence, a ciphertext can be decrypted by multiple users with same attributes. For example, Alice and Bob have the attributes {"Researcher," "Neurosurgery"}; then, both of them can decrypt the ciphertext associated with the access policy ("Doctor" AND "Neurosurgery") OR ("Researcher" AND "Neurosurgery"). In the MA-CP-ABE system, if a malicious user who has same attributes with others sells his decryption key on the Internet, how to identify the malicious user?

Another major issue in MA-ABE is malicious user revocation. In the MA-CP-ABE system, the decryption keys may be compromised and the corresponding malicious users should be removed from the system. Hence, the user revocation mechanism should be designed for the MA-CP-ABE system. The user revocation mechanism was divided into direct revocation and indirect revocation. In direct revocation, the data owner encrypts the data by a specified revocation list, and the revoked users who in this list cannot decrypt the corresponding ciphertext. Unfortunately, the direct revocation mechanism requires each data owner to keep a revocation user identity list and breaks the user anonymity in the ABE system. In indirect revocation, the authorities help the nonrevoked users to update their decryption keys periodically, so the revoked users cannot decrypt the new ciphertexts. In this paper, we focus on the indirect user revocation issue in the MA-CP-ABE system.

One efficiency drawback for MA-ABE is the significant cost of data access. In the MA-CP-ABE system, the number of resource-consuming pairing operations required to decrypt a ciphertext grows linearly with the number of attributes used for decryption, which makes the data access too expensive. This drawback hinders the large-scale application of the MA-CP-ABE system in lightweight devices. For example, consider a medical cloud system based on MA-CP-ABE, the patients encrypt the data and upload the ciphertexts in cloud, and the doctor may need to real-time access the medical data by a smartphone. Due to the expensive access cost, the traditional MA-CP-ABE system is obviously unsuitable in this scenario.

1.1. Our Contributions.

Seeking to address the above issues, we first give the formal definition and security model for traceable MA-CP-ABE (T-MA-CP-ABE) scheme and propose a concrete construction of T-MA-CP-ABE on prime order bilinear groups. Then, we prove the construction is adaptively secure under the symmetric external Diffie–Hellman assumption and fully traceable under the q-Strong Diffie–Hellman assumption in the random oracle model. Based on the T-MA-CP-ABE construction, we further present a traceable and revocable MA-CP-ABE (TR-MA-CP-ABE) system for secure cloud storage. To the best of our knowledge, this is the first practical MA-ABE system that simultaneously supports traceability, revocation, and fast access. The major features of our TR-MA-CP-ABE system are outlined as follows:

(1) Multiauthority. There exists a central authority (CA) and multiple attribute authorities in our TR-MA-CP-ABE system. Each attribute authority (AA) is responsible for generating the user secret keys for the attributes under its control, and CA is responsible for tracing and revoking the malicious users. Unlike prior MA-ABE schemes, neither CA nor AA can independently generate user decryption keys in our system, even for just one attribute. In addition, the access policies can be expressed as any monotone access structures, which make our system more practical.

(2) Traceability. Our TR-MA-CP-ABE system supports white-box traceability (traceability can be divided into white-box traceability and black-box traceability. White-box traceability can catch the malicious user who leaks his decryption keys to others, while black-box traceability can catch the malicious user who leaks a decryption black-box). In our system, CA generates tracing information and user secret keys for the identity. If a malicious user leaks his decryption key to others, then CA can trace the malicious user identity from the corresponding decryption key. By adopting a full signature technique, our system does not require any identity table for tracing, which significantly reduces the storage overhead for CA.

(3) Revocation. Our TR-MA-CP-ABE system supports indirect user revocation. If a malicious user was caught by CA, then CA adds his identity into a revocation list, and AAs only periodically update the attribute-based secret keys for the users whose identities do not belong to the revocation list. Hence, the malicious users cannot obtain the new decryption keys and access the new ciphertext data created in the current time period.

(4) Fast access. In our TR-MA-CP-ABE system, the number of pairings for decrypt a ciphertext is only 6, rather than increases linearly with the number of attributes used during decryption. Furthermore, our decryption operation is run on prime order bilinear groups, which makes access speed significantly faster. The efficiency comparison shows that the data access in our system is more efficient than that in other related works.

Table 1 compares the specific features of our TR-MA-CP-ABE system with the existing ABE schemes [13–16] that achieve multiauthority and traceability simultaneously.

1.2. Related Works. Chase [12] introduced the notion of MA-ABE and gave the first concrete construction of MA-ABE. As CA is assumed to be able to decrypt every ciphertext in [12], Chase and Chow [17] proposed a MA-ABE scheme without any CA, which was limited to expressing a strict "AND" policy over a predetermined set of authorities. Later, Lewko and Waters [18] presented an adaptively secure MA-ABE scheme where a policy could be expressed as any monotonic Boolean formula. Based on [18], Cui and Deng [19] presented a revocable MA-ABE that achieves attribute revocation. Zhang et al. [20] presented a shorter MA-ABE where a ciphertext can be decrypted with a constant number of pairing operations. Wang et al. [21] constructed a MA-ABE scheme from the LWE assumption. More recently, Xiong et al. [22] presented a revocable MA-ABE with outsourced decryption. However, these schemes did not consider the trace problem.

Hinek et al. [23] proposed the first traceable CP-ABE, but their scheme only supports "AND gates with wildcard." To improve the expression ability, Liu et al. [24] presented the first traceable CP-ABE that supports monotonic access structures. Later, Wang et al [25] presented a traceable CP-ABE that can catch the malicious user who leaks a black-box decryption equipment. Ning et al. [26] presented a traceable and revocable CP-ABE that supports both accountable authority and public auditing. Liu and Wong [27] proposed a traceable and revocable CP-ABE for large universe. Xu [28] constructed a traceable CP-ABE with short decryption key. Recently, Han et al. [29] presented a traceable and revocable CP-ABE with hidden policy. Unfortunately, the above schemes can only apply to the single-authority setting.

To address the key abuse problem in MA-ABE, Li et al. [13] presented a traceable MA-CP-ABE with limited access policy and security. Later, Zhou et al. [14] proposed a revocable and traceable MA-CP-ABE that achieves high expressiveness and full security. However, there exists multiple CAs in their scheme, and each CA needs to maintain a tracing identity table. Yu et al. [15] constructed a traceable MA-CP-ABE without any identity table and proved it is adaptively secure in composite order groups. Recently, Zhang et al. [16] presented a more efficient traceable MA-CP-ABE in prime order groups. Unfortunately, their scheme only achieves statically secure and does not support user revocation. In addition, the common efficiency drawback of these schemes is that the number of pairing operations required to decrypt a ciphertext increases linearly with the number of attributes satisfying the access policy, which presents significant challenges for the users who access data by mobile devices.

1.3. Organization. Section 2 introduces the relevant preliminaries, which includes the access structure, bilinear group, and complexity assumptions. Section 3 gives the system architecture, algorithm definition, and security model of TR-MA-CP-ABE. Section 4 presents the detailed constructions and formal security analysis of T-MA-CP-ABE scheme. Section 5 designs a TR-MA-CP-ABE system and compares its efficiency with other related works. Section 6 concludes the whole paper.

Table 1: Features comparison with other related works.

	AS[2]	MAS[3]	ZST[4]	POG	FA[5]	R
[13]	×	×	✓	✓	×	×
[14]	✓	✓	×	×	×	✓
[15]	✓	✓	✓	×	×	×
[16]	×	✓	✓	✓	×	×
This paper	✓	✓	✓	✓	✓	✓

AS represents adaptively secure, MAS represents supporting any monotone access structures, ZST represents zero storage cost for tracing, POG represents constructed in prime order groups, FA represents constant pairing operations for data access, and R represents revocation. [2]The scheme [13] achieves selectively secure and the scheme [16] only achieves statically secure. [3]In [13], their scheme only supports "AND gates with wildcard." [4]In [14], the number of identity tables for tracing is equal to the number of central authorities in the scheme. [5]In [13–16], the number of pairing operations for decryption grows linearly with the number of attributes used for decryption.

2. Preliminaries

2.1. Notations. For convenience, we define some notations that will be used in this paper. For a finite set S, we denote by $s \leftarrow_R S$, the fact that s is chosen uniformly at random from S. Let $\mathbb{Z}_p$ be a set $\{0, 1, 2, \ldots, p-1\}$, where p is a prime. Let $\mathbb{Z}_p^n$ and $\mathbb{Z}_p^{l\times n}$ denote the set of all n-dimensional vectors and $l \times n$ matrices (l rows and n columns) in $\mathbb{Z}_p$, respectively. We denote a matrix by a bold letter. For a matrix $A \in \mathbb{Z}_p^{l\times n}$, let $A^\top$ be the transposition of A, and $a_{ij} \in \mathbb{Z}_p$ be the $(i, j)^{\text{th}}$ (the i^{th} row and j^{th} column) element of A. For group G, $g \in G$, and matrix $A \in \mathbb{Z}_p^{l\times n}$, we use g^A to denote the $l \times n$ matrix, in which its $(i, j)^{\text{th}}$ element is $g^{a_{ij}}$. For matrix $B \in \mathbb{Z}_p^{n\times m}$, we denote $(g^{\mathbf{A}})^{\mathbf{B}} = g^{\mathbf{AB}} \in G^{l\times m}$. For $\overrightarrow{v} = (v_1, v_2, \ldots, v_n) \in \mathbb{Z}_p^n$, we denote $g^{\overrightarrow{v}} = (g^{v_1}, g^{v_2}, \ldots, g^{v_n}) \in G^n$. For two vectors $\overrightarrow{v} = (v_1, v_2, \ldots, v_n)$ and $\overrightarrow{\omega} = (\omega_1, \omega_2, \ldots, \omega_n) \in \mathbb{Z}_p^n$, we denote the inner product of $\overrightarrow{v}$ and $\overrightarrow{\omega}$ by $\langle \overrightarrow{v}, t\overrightarrow{\omega} \rangle = \sum_{i=1}^{i=n} v_i\omega_i$. We can also denote the above inner product notation for row and column vectors as follows.

$$\langle \overrightarrow{v}, t\overrightarrow{\omega} \rangle = \begin{cases} \overrightarrow{v}\,\overrightarrow{\omega}^\top, & \text{if } \overrightarrow{v} \text{ and } \overrightarrow{\omega} \text{ are both row vectors,} \\ \overrightarrow{v}\,\overrightarrow{\omega}, & \text{if } \overrightarrow{v} \text{ is a row vector and } \overrightarrow{\omega} \text{ is a column vector,} \\ \overrightarrow{v}^\top\overrightarrow{\omega}^\top, & \text{if } \overrightarrow{v} \text{ is a column vector and } \overrightarrow{\omega} \text{ is a row vector,} \\ \overrightarrow{v}^\top\overrightarrow{\omega}, & \text{if } \overrightarrow{v} \text{ and } \overrightarrow{\omega} \text{ are both column vectors.} \end{cases} \tag{1}$$

Note that $\langle \overrightarrow{v}, t\overrightarrow{\omega} \rangle = \langle \overrightarrow{v}, t\overrightarrow{\omega} \rangle^\top$ and $\overrightarrow{v}^\top\overrightarrow{\omega}^\top = (\overrightarrow{v}^\top\overrightarrow{\omega}^\top)^\top = \overrightarrow{\omega}\,\overrightarrow{v}$.

2.2. Access Structures

Definition 1 (Access structure [30]). Let U be the attributes universe. An access structure $\mathbb{A}$ is a collection of nonempty subsets of U, i.e., $\mathbb{A} \subseteq 2^U \setminus \{\emptyset\}$. If for $\forall B, C$, we have $B \in \mathbb{A}, B \subseteq C \Rightarrow C \in \mathbb{A}$; then, we say $\mathbb{A}$ is monotone. The sets in

$\mathbb{A}$ are called authorized sets, while the sets not in $\mathbb{A}$ are called unauthorized sets.

Definition 2 (Linear secret-sharing schemes (LSSS) [30]). A secret-sharing scheme Π over the attributes universe U is called linear over $\mathbb{Z}_p$ if

(1) The shares for each attribute form a vector over $\mathbb{Z}_p$

(2) There exists a matrix $A \in \mathbb{Z}_p^{l\times n}$ and function ρ: $\{1, 2, \ldots, l\} \longrightarrow U$ satisfy the following: let the column vector $\overrightarrow{v} = (s, r_2, \ldots, r_n) \in \mathbb{Z}_p^{1\times n}$, where $s \in \mathbb{Z}_p$ is the secret to be shared, and $r_2, \ldots, r_n \leftarrow_R \mathbb{Z}_p$; then, $A\overrightarrow{v}^{\top}$ is equal to the vector of l shares of the secret s according to Π. The share $(A\overrightarrow{v}^{\top})_i$ belongs to attribute $\rho(i)$.

Let Π be an LSSS for the access structure $\mathbb{A}$ and (A, ρ) be the access policy for $\mathbb{A}$. According to [30], LSSS enjoys the linear reconstruction as follows. Let $S \in \mathbb{A}$ be an authorized set, and let $I = \{i\colon \rho(i) \in S\} \subset \{1, 2, \ldots, l\}$. Then, there exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i\in I}$ such that $\sum_{i\in I} c_i \overrightarrow{A_i} = (1, 0, \ldots, 0) \in \mathbb{Z}_p^{1\times n}$, where $\overrightarrow{A_i}$ is the row i of matrix A.

2.3. Bilinear Groups and Assumptions. Let $\mathcal{G}$ be an asymmetric bilinear group generator that takes as input a security parameter λ and outputs a tuple $\mathbb{G} = (p, G_1, G_2, G_T, g_1, g_2, e)$, where G_1, G_2, and G_T are the cyclic groups of prime order p, g_1 (respectively, g_2) is a generator of G_1 (respectively, G_2), and e: $G_1 \times G_2 \longrightarrow G_T$ is an efficiently computable bilinear map such that

(1) Bilinear: $\forall g \in G_1, h \in G_2, a, b \in \mathbb{Z}_p, e(g^a, h^b) = e(g, h)^{ab}$

(2) Nondegenerate: $e(g_1, g_2) \neq 1$

For $g \in G_1, h \in G_2, \overrightarrow{v}, \overrightarrow{\omega} \in \mathbb{Z}_p^n$, we denote $e(g^{\overrightarrow{v}}, h^{\overrightarrow{\omega}}) = \prod_{i=1}^{n} e(g^{v_i}, h^{\omega_i}) = e(g, h)^{\langle \overrightarrow{v}, t\overrightarrow{\omega}\rangle} \in G_T$.

Definition 3 (SXDH, Symmetric External Diffie–Hellman assumption [31]). The adversary $\mathcal{A}$'s advantage in SXDH assumption is defined as

$$\mathrm{Adv}_{\mathcal{A}}^{\mathrm{SXDH}}(\lambda) = \left|\Pr\left[\mathcal{A}\left(D, T_{0,i}\right) = 0\right] - \Pr\left[\mathcal{A}\left(D, T_{1,i}\right) = 0\right]\right|, \tag{2}$$

where $\mathbb{G} = (p, G_1, G_2, G_T, g_1, g_2, e) \leftarrow \mathcal{G}(\lambda)$, $a, b, c \leftarrow_R \mathbb{Z}_p$, $i \in \{1, 2\}$, $D = (\mathbb{G}, g_i^a, g_i^b)$, $T_{0,i} = g_i^{ab}$, and $T_{1,i} = g_i^{ab+c}$. We say the SXDH assumption holds if for all polynomial time algorithm adversaries $\mathcal{A}$ and both $i \in \{1, 2\}$, $\mathrm{Adv}_{\mathcal{A}}^{\mathrm{SXDH}}(\lambda)$ is negligible in λ.

Definition 4 (q-SDH, q-Strong Diffie–Hellman assumption [32]). The adversary $\mathcal{A}$'s advantage in q-SDH assumption is defined as

$$\mathrm{Adv}_{\mathcal{A}}^{q\text{-SDH}}(\lambda) = \Pr\left[\mathcal{A}\left(g_1, g_1^x, g_2, g_2^x, g_2^{x^2}, \ldots, g_2^{x^q}\right) = \left(d, g_2^{(1/x+d)}\right)\right], \tag{3}$$

where $\mathbb{G} = (p, G_1, G_2, G_T, g_1, g_2, e) \leftarrow \mathcal{G}(\lambda)$, $x \leftarrow_R \mathbb{Z}_p$, and $d \in \mathbb{Z}_p \backslash \{-x\}$. We say the q-SDH assumption holds if for all polynomial time algorithm adversaries $\mathcal{A}$, $\mathrm{Adv}_{\mathcal{A}}^{q\text{-SDH}}(\lambda)$ is negligible in λ.

Note that compared with the q-SDH assumption in [32], g_1 and g_2 have exchanged places here. However, this will not affect the security of full signature scheme [32], that is, strong existential unforgeability under an adaptive chosen message attack based on q-SDH assumption because we will also exchange the places of g_1 and g_2 in the full signature scheme. The modified full signature scheme (BB scheme) is briefly described as follows:

(i) Setup (λ). Run $\mathcal{G}(\lambda)$ to obtain $(p, G_1, G_2, G_T, g_1, g_2, e)$. Pick $a, b \longleftarrow_R \mathbb{Z}_p$, set the public key $PK = (p, G_1, G_2, G_T, g_1, g_2, e, u = g_1^a, \quad v = g_1^b, z = e(g_1, g_2))$, and secret key $SK = (g_2, a, b)$.

(ii) Sign (SK, M). Given a message M and SK, pick $r \leftarrow_R \mathbb{Z}_p \backslash \{(-a + M)/b$, compute $\sigma = g_2^{(1/a+M+\mathrm{br})}$, and set the signature as (σ, r)

(iii) Verify $(\mathrm{PK}, M, \sigma, r)$. If $e(ug_1^M v^r, \sigma) = z$, it outputs 1 meaning that the signature (σ, r) is valid. Otherwise, it outputs 0 meaning that the signature (σ, r) is invalid.

3. Problem Formulation

In this section, we first describe the system architecture of our TR-MA-CP-ABE. Then, we give the formal algorithm definition and security model for T-MA-CP-ABE and TR-MA-CP-ABE scheme.

3.1. System Architecture. As shown in Figure 1, our TR-MA-CP-ABE system comprises the following entities: a cloud sever (CS), a central authority (CA), multiple attribute authorities (AAs), data owners (DOs), and data users (DUs). The role of each party is described as follows:

(i) CS: CS is responsible for storing the ciphertexts and processing the ciphertext upload and download requests

(ii) CA: CA is not only responsible for generating the identity keys for data users but also for tracing and revoking the malicious users

(iii) AA: each AA generates the attribute keys for data users and updates the attribute keys for nonrevoked users

(iv) DO: each DO encrypts his own data and outsources the corresponding ciphertext to CS

(v) DU: each DU downloads the ciphertext from CS and accesses the corresponding data by his decryption key

More specifically, CA generates its own public/secret key pair, publishes the CA public key, and uses the CA secret key to generate the identity keys for all DUs. Each AA generates its own public/secret key pair, publishes the AA public key, and generates the user keys corresponding to the attributes

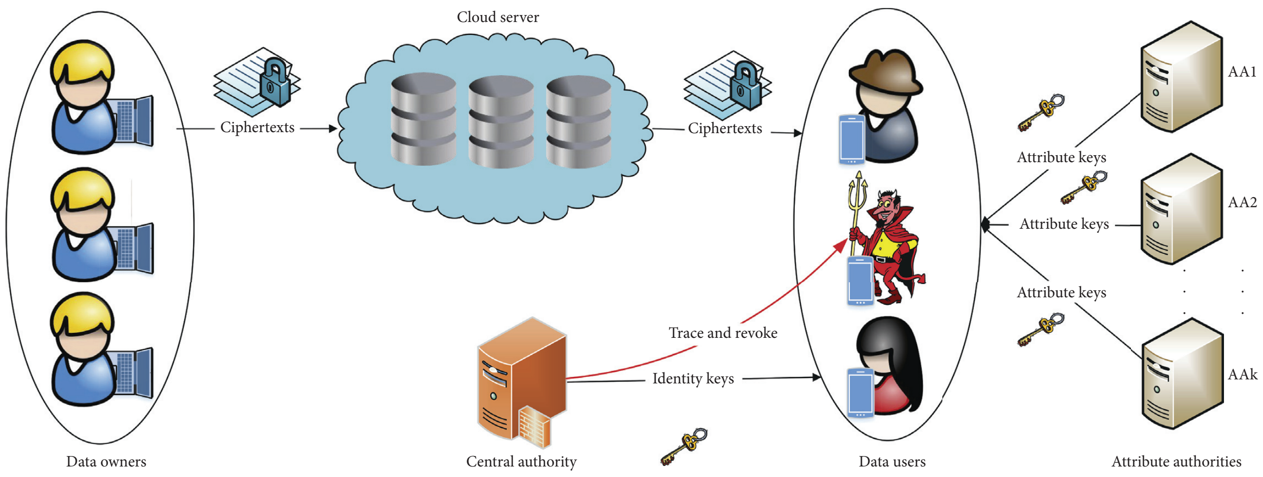

FIGURE 1: System architecture of TR-MA-CP-ABE.

that are managed by it. Then, DU uses the identity key and attribute keys to generate his own decryption key. Next, DO encrypts the data by the public keys and an access policy and uploads the ciphertext to CS. Finally, the nonrevoked uses can decrypt the ciphertext when their attributes satisfy the access policy, and other users cannot access the data. In our system, when a malicious user sells his decryption key, CA first identifies him by a tracing algorithm and then revokes him by adding his identity to a revocation list. Since AA will not update the attribute keys for the users whose identities are in the revocation list, the malicious users cannot update their decryption keys and access new ciphertext data.

In our system, DOs are fully trusted entities who honestly execute the encryption algorithm. CS, CA, and AAs are both honest but curious, who correctly execute the algorithms in the system, but try to learn any sensitive information about the data. Our system does not allow CS to modify or delete the stored ciphertext, but allows several corrupt AAs to make an attack on the unauthorized ciphertext whose policy cannot be satisfied by the corrupt attributes. Note that the decryption key is generated by the combination of identity key and attribute keys, so neither CA nor AA can independently construct the complete decryption key in our system. DUs are untrusted entities that may not only try to access the unauthorized data but also sell their decryption keys on the Internet. To formally describe the above system and attacks, Section 3.2 defines the TR-MA-CP-ABE algorithms, and Section 3.3 presents an adaptive security model against the adversary who try to learn any information about the unauthorized data and a traceable security model against the malicious data user who leaks his decryption key.

3.2. Algorithm Definition. A T-MA-CP-ABE scheme consists of eight algorithms:

(i) Global Setup (λ). On input a security parameter λ, it outputs the global parameters GP for the system

(ii) CA Setup (GP). CA runs this algorithm with the global parameters GP as input, and outputs its public/secret key pair $(\mathrm{CPK}, \mathrm{CSK})$

(iii) AA Setup (GP, S_j). Each attribute authority AA_j runs this algorithm with the global parameters GP and its attributes set S_j as input and outputs its public/secret key pair $(\mathrm{APK}_j, \mathrm{ASK}_j)$

(iv) CA KeyGen $(\mathrm{GID}, \mathrm{CSK}, \mathrm{GP})$. On input an identity GID, the CA secret key CSK and the global parameters GP, the CA key generation algorithm outputs the user's CA key $\mathrm{CSK}_{\mathrm{GID}}$

(v) AA KeyGen $(\mathrm{GID}, \mathrm{GP}, S, \mathrm{CSK}_{\mathrm{GID}}, \{\mathrm{ASK}_j\})$. On input an identity GID, the global parameters GP, a set of attributes S, a user's CA key $\mathrm{CSK}_{\mathrm{GID}}$, and the set of AA secret keys $\{\mathrm{ASK}_j\}$ for the relevant AAs, the AA key generation algorithm outputs the user's decryption key $\mathrm{SK}_{S,\mathrm{GID}}$

(vi) Encrypt $(\mathrm{GP}, \mathrm{CPK}, \{\mathrm{APK}_j\}, M, (A, \rho))$. On input the global parameters GP, the CA public key CPK, the set of AA public keys $\{\mathrm{APK}_j\}$ for the relevant AAs, a message M, and an access policy (A, ρ), the encryption algorithm outputs a ciphertext CT

(vii) Decrypt $(\mathrm{GP}, \mathrm{SK}_{S,\mathrm{GID}}, \mathrm{CT})$. On input the global parameters GP, a decryption key $\mathrm{SK}_{S,\mathrm{GID}}$ for an attributes set S, and a ciphertext CT for an access policy (A, ρ), the decryption algorithm returns either the message M when the attributes set S satisfies the access policy (A, ρ) or the error symbol $\perp$ meaning that decryption fails

(viii) Trace $(\mathrm{GP}, \mathrm{CPK}, \{\mathrm{APK}_j\}, \mathrm{SK}_{S,\mathrm{GID}})$. On input the global parameters GP, the CA public key CPK, the AA public keys $\{\mathrm{APK}_j\}$, and a decryption key $\mathrm{SK}_{S,\mathrm{GID}}$, the tracing algorithm returns either an identity GID when $\mathrm{SK}_{S,\mathrm{GID}}$ passes the key sanity check, or the symbol $\top$ meaning that $\mathrm{SK}_{S,\mathrm{GID}}$ does not need to be traced. The key sanity check is a deterministic algorithm to determine whether $\mathrm{SK}_{S,\mathrm{GID}}$ needs to be traced

Our TR-MA-CP-ABE scheme is almost the same with the T-MA-CP-ABE scheme, except for modifying CA Setup by adding a revocation list, Encrypt and Decrypt by adding a time period, and replacing AA KeyGen by AA KeyGen and KeyUpdate and Trace by Trace and Revoke. The above modified and replaced algorithms in TR-MA-CP-ABE scheme are described as follows.

(ix) CA Setup (GP). CA runs the CA setup algorithm with the global parameters GP as input to generate its public/secret key pair $(\mathrm{CPK}, \mathrm{CSK})$. In addition, CA initializes an empty revocation identity list RL.

(x) AA KeyGen and KeyUpdate $(\mathrm{GID}, \mathrm{GP}, S, \mathrm{CSK}_{\mathrm{GID}}, T, \mathrm{RL}, \{\mathrm{ASK}_j\})$. It takes as input an identity GID, the global parameters GP, a set of attributes S, a time period T, a user's CA key $\mathrm{CSK}_{\mathrm{GID}}$, a revocation list RL, and the set of AA secret keys $\{\mathrm{ASK}_j\}$ for the relevant AAs. If $\mathrm{GID} \in \mathrm{RL}$, it outputs $\perp$. Otherwise, it outputs the user's decryption key $\mathrm{SK}_{S,\mathrm{GID}}$.

(xi) Encrypt $(\mathrm{GP}, \mathrm{CPK}, \{\mathrm{APK}_j\}, M, (A, \rho), T')$. It takes as input the global parameters GP, the CA public key CPK, the set of AA public keys $\{\mathrm{APK}_j\}$ for the relevant authorities, an access policy (A, ρ), a message M, and a time period T'. It outputs a ciphertext CT.

(xii) Decrypt $(\mathrm{GP}, \mathrm{SK}_{S,\mathrm{GID}}, \mathrm{CT})$. It takes as input the global parameters GP, a decryption key $\mathrm{SK}_{S,\mathrm{GID}}$ for an attributes set S for a time period T, and a ciphertext CT for an access policy (A, ρ) for a time period T'. If $T = T'$ and S satisfies (A, ρ), it outputs the message M. Otherwise, it outputs the error symbol $\perp$.

(xiii) Trace and Revoke $(\mathrm{GP}, \mathrm{CPK}, \{\mathrm{APK}_j\}, \mathrm{SK}_{S,\mathrm{GID}})$. It takes as input the global parameters GP, the CA public key CPK, the AA public keys $\{\mathrm{APK}_j\}$, and a decryption key $\mathrm{SK}_{S,\mathrm{GID}}$. If $\mathrm{SK}_{S,\mathrm{GID}}$ passes the key sanity check, it returns an identity GID and add it to the revocation list RL. Otherwise, it returns the symbol $\top$.

3.3. Security Model. We now describe the adaptive security model for T-MA-CP-ABE scheme. In our security model, an AA can manage multiple attributes, while each attribute can only be controlled by one AA. Let V be the attribute authority universe and U be the attribute universe. The adaptive security game between a challenger and an adversary is defined as follows.

(i) Setup. The challenger runs the global setup and CA setup algorithms and then gives GP and CPK to the adversary. The adversary specifies a set of corrupt AAs $V' \subseteq V$. For noncorrupt AAs in $V - V'$, the challenger runs the AA setup algorithm and provides the AA public keys to the adversary.

(ii) Phase 1. The adversary can repeatedly make two types of key queries as follows

(1) CA key query. The adversary sends a user's identity GID to the challenger. The challenger returns the corresponding private key $\mathrm{CSK}_{\mathrm{GID}}$ to the adversary.

(2) AA key query. The adversary sends a pair (S, GID) to the challenger, where GID is an identity, and S is a set of attributes belonging to noncorrupt AAs. The challenger returns the corresponding decryption key $\mathrm{SK}_{S,\mathrm{GID}}$ to the adversary. Note that the user's AA private key is part of his decryption key in our scheme, so the challenger gives the user's AA private key to the adversary in this query.

(iii) Challenge. The adversary submits two messages M_0, M_1 and an access policy (A, ρ), where (A, ρ) satisfies the following constraint. Let $S_{V'}$ denote the attributes controlled by corrupt AAs, and S_{GID} denotes the attributes in which the adversary has queried for identity GID. For each GID, we require that $S_{V'} \cup S_{GI\,D}$ does not satisfy (A, ρ). The challenger chooses a random coin $\beta \in \{0, 1\}$ and returns ciphertext $CT^* = \mathbf{Encrypt}(\mathrm{GP}, \mathrm{CPK}, \{\mathrm{APK}_j\}, M_\beta, (A, \rho))$ to the adversary.

(iv) Phase 2. The adversary can make the key queries as Phase 1, with the restriction of (A, ρ) as described above

(v) Guess. The adversary submits a guess $\beta' \in \{0, 1\}$ and wins if $\beta = \beta'$. The advantage of an adversary in this game is defined as $\Pr[\beta = \beta'] - (1/2)$.

Definition 5. A T-MA-CP-ABE scheme is adaptively (or fully) secure if for any probabilistic polynomial time adversary, its advantage is negligible in λ.

A T-MA-CP-ABE scheme is called selectively secure if the adversary submits the access policy (A, ρ) before the Setup phase. A T-MA-CP-ABE scheme is called statically secure if the adversary submits all queries immediately after seeing the global parameters. Our construction will be proved to satisfy adaptively secure without the above restrictions.

Traceability of the T-MA-CP-ABE is described by a game as follows:

(i) Setup. The challenger runs the global setup, CA setup, and AA setup algorithms and then gives GP, CPK, and $\{\mathrm{APK}_j\}$ to the adversary

(ii) Key query. The adversary makes the following queries

(1) CA key query. The adversary sends $\{\mathrm{GID}_i\}_{i=1}^{m}$ to the challenger, where GID_i is an identity. The challenger returns the corresponding private keys $\{\mathrm{SK}_{\mathrm{GID}_i}\}_{i=1}^{m}$.

(2) AA key query. The adversary sends $\{(S_i, \mathrm{GID}_i)\}_{i=1}^{m}$ to the challenger, where S_i is an attributes set. The challenger returns the corresponding decryption keys $\{\mathrm{SK}_{S_i,\mathrm{GID}_i}\}_{i=1}^{m}$.

(iii) Key forgery. The adversary submits a decryption key SK^* and wins if $[\mathbf{Trace}(\mathrm{GP}, \mathrm{CPK}, \{\mathrm{APK}_j\}, \mathrm{SK}^*) \notin \{\top, \mathrm{GID}_1, \ldots, \mathrm{GID}_m\}]$.

The advantage of an adversary in this game is defined as

$$\Pr\left[\textbf{Trace}\left(\mathrm{GP}, \mathrm{CPK}, \{\mathrm{APK_j}\}, \mathrm{SK}^*\right) \notin \{\top, \mathrm{GID}_1, \ldots, \mathrm{GID}_m\}\right]. \tag{4}$$

Definition 6. A T-MA-CP-ABE scheme is fully traceable if for any probabilistic polynomial time adversary, its advantage is negligible in λ.

In our TR-MA-CP-ABE scheme, the AA key generation algorithm is same with the AA key update algorithm. Hence, the security model of our TR-MA-CP-ABE scheme is same with that of our T-MA-CP-ABE scheme.

4. Our T-MA-CP-ABE Scheme

In this section, we present a T-MA-CP-ABE scheme in an asymmetric bilinear group and prove it is adaptively secure and fully traceable in the random oracle model.

4.1. Construction. Inspired by [18, 20], we adopt a hash function H to map user identities to the elements in group G_2. Unlike with [18, 20], we use a CA to personalize the identity key for each user and the AAs to generate the corresponding attribute keys, so our construction can achieve multiple authorities and the AAs cannot get the user decryption key. Furthermore, we employ a full signature scheme [32] to realize traceability. More specifically, the CA injects the signature of the user identity into the user identity key and traces the user by his decryption key. We now present our T-MA-CP-ABE construction based on [18, 20], in which each attribute authority AA_j manages an attributes set S_j.

(i) Global Setup (λ). The algorithm first runs $\mathcal{G}(\lambda)$ to obtain $(p, G_1, G_2, G_T, g_1, g_2, e)$. G_1, G_2, and G_T are the cyclic groups of prime order p, g_1 is a generator of G_1, g_2 is a generator of G_2, and e: $G_1 \times G_2 \longrightarrow G_T$ is a bilinear map. It then samples $B = (b_{ij})_{3\times 3} \leftarrow \mathrm{GL}_3(\mathbb{Z}_p)$ and sets $B^* = (B^{-1})^\top$ and $\overrightarrow{b_1} = (b_{11}, b_{21}, b_{23})^\top$. It chooses a hash function H: $\mathbb{Z}_p^* \longrightarrow G_2^3$ and publishes $\mathrm{GP} = (p, G_1^3, G_2^3, G_T, g_1, g_2, e, g_1^{\overrightarrow{b_1}}, H)$ as the global parameters.

(ii) CA Setup (GP). CA picks $a, b \leftarrow_R \mathbb{Z}_p$ and computes $\mathrm{cpk}_1 = (g_1^{\overrightarrow{b_1}})^a = g_1^{\mathrm{a}\overrightarrow{\mathrm{b}_1}}$, $\mathrm{cpk}_2 = (g_1^{\overrightarrow{b_1}})^b = g_1^{\mathrm{b}\overrightarrow{\mathrm{b}_1}}$. Then, CA publishes the public key $\mathrm{CPK} = (\mathrm{cpk}_1, \mathrm{cpk}_2)$ and sets $\mathrm{CSK} = (a, b)$ as its secret key.

(iii) AA Setup (GP, S_j). For each attribute $i \in S_j$, AA_j picks $\overrightarrow{k_i} \leftarrow_R \mathbb{Z}_p^3, Y_i \leftarrow_R \mathbb{Z}_p^{3\times 3}$ and computes $\mathrm{apk}_{i,1} = (g_1^{\overrightarrow{b_1}^\top Y_i})^\top = g_1^{Y_i^\top \overrightarrow{b_1}}$, and $\mathrm{apk}_{i,2} = e(g_1^{\overrightarrow{b_1}}, g_2^{\overrightarrow{k_i}}) = e(g_1, g_2)^{\overrightarrow{b_1}^\top \overrightarrow{k_i}^\top} = e(g_1, g_2)^{\overrightarrow{k_i}\,\overrightarrow{b_1}}$. Then, AA_j publishes the public key $\mathrm{APK}_j = \left\{\mathrm{apk}_{i,1}, \mathrm{apk}_{i,2}\right\}_{i\in S_j}$ and sets $\mathrm{ASK}_j = \left\{\overrightarrow{k_i}, Y_i^\top\right\}_{i\in S_j}$ as its secret key.

(iv) CA KeyGen $(\mathrm{GID}, \mathrm{CSK}, \mathrm{GP})$. For a user's identity $\mathrm{GID} \in \mathbb{Z}_p$, CA picks $r_{\mathrm{GID}} \leftarrow_R \mathbb{Z}_p \setminus \{-(a + \mathrm{GID}/b)\}$ and sets $K_{1,GID} = \mathrm{GID}, K_{2,\mathrm{GID}} = r_{\mathrm{GID}}, K_{3,\mathrm{GID}} = g_2^{(1/a+\mathrm{GID}+\mathrm{br}_{\mathrm{GID}})}$, and $K_{4,\mathrm{GID}} = H(\mathrm{GID})^{(1/a+\mathrm{GID}+\mathrm{br}_{\mathrm{GID}})}$. Then, CA sends the private key $\mathrm{CSK}_{\mathrm{GID}} = (K_{1,\mathrm{GID}}, K_{2,\mathrm{GID}}, K_{3,\mathrm{GID}}, K_{4,\mathrm{GID}})$ to the user whose identity is GID.

(v) AA KeyGen $(\mathrm{GID}, \mathrm{GP}, S, \mathrm{CSK}_{\mathrm{GID}}, \{\mathrm{ASK_j}\})$. A user submits his identity GID and attributes set S and $(K_{3,\mathrm{GID}}, K_{4,\mathrm{GID}})$ to the relevant authorities $\left\{\mathrm{AA}_j\right\}$. For each attribute $i \in S \cap S_j$, AA_j computes and sends the private key $\mathrm{SK}_{i,\mathrm{GID}} = K_{3,\mathrm{GID}}^{\overrightarrow{k_i}} K_{4,\mathrm{GID}}^{Y_i^\top}$ to the corresponding user. When the user receives $\left\{\mathrm{SK}_{i,\mathrm{GID}}\right\}_{i\in S}$, he sets $\mathrm{SK}_{S,\mathrm{GID}} = (K_{1,\mathrm{GID}}, K_{2,\mathrm{GID}}, \left\{\mathrm{SK}_{i,\mathrm{GID}}\right\}_{i\in S})$ as his decryption key.

(vi) Encrypt $(\mathrm{GP}, \mathrm{CPK}, \{\mathrm{APK_j}\}, M, (A, \rho))$. On input a message M and an access policy (A, ρ). A is a $l \times n$ matrix, and ρ maps its rows to attributes. It first picks $U_2, U_3, \ldots, U_n \leftarrow_R \mathbb{Z}_p^{3\times 3}, \overrightarrow{0} \leftarrow \mathbb{Z}_p^3, \overrightarrow{v} = (s_0, v_2, v_3, \ldots, v_n) \leftarrow_R \mathbb{Z}_p^n, s \leftarrow_R \mathbb{Z}_p$. For each $x \in \{1, 2, \ldots l\}$, it computes $\lambda_x = \langle \overrightarrow{A_x}, \overrightarrow{v} \rangle$, , where $\overrightarrow{A_x}$ is the row x of A. The ciphertext CT is computed as

$$\begin{aligned}
C &= \mathrm{M}e(g_1, g_2)^{s_0}, \\
C_0 &= g_1^{s\overrightarrow{b_1}}, \\
C_1 &= g_1^{s\mathrm{a}\overrightarrow{\mathrm{b}_1}}, \\
C_2 &= g_1^{s\mathrm{b}\overrightarrow{\mathrm{b}_1}}, \\
C_{1,x} &= e(g_1, g_2)^{\lambda_x} e(g_1, g_2)^{s\overrightarrow{k_{\rho(x)}}\,\overrightarrow{b_1}}, \\
C_{2,x} &= g_1^{\left(\overrightarrow{0}^\top, sU_2^\top\overrightarrow{\mathrm{b}_1}, \ldots, sU_l^\top\overrightarrow{\mathrm{b}_1}\right)\overrightarrow{A_x}^\top} g_1^{sY_{\rho(x)}^\top\overrightarrow{\mathrm{b}_1}}.
\end{aligned} \tag{5}$$

Decrypt $(\mathrm{GP}, \mathrm{SK}_{S,\mathrm{GID}}, \mathrm{CT})$. On input a ciphertext CT for a policy (A, ρ) and a decryption key $\mathrm{SK}_{S,\mathrm{GID}}$ for an attributes set S. Let $I = \{x\colon \rho(x) \in S\}$. If S does not satisfy (A, ρ), it outputs $\perp$. Otherwise, it chooses constants $\left\{c_x \in \mathbb{Z}_p\right\}_{x\in I}$ such that $\sum_{x\in I} c_x \overrightarrow{A_x} = (1, 0, \ldots, 0)$ and computes

$$\frac{\prod_{x\in I} C_{1,x}^{c_x} e\left(\prod_{x\in I} C_{2,x}^{c_x}, H\left(K_{1,\mathrm{GID}}\right)\right)}{e\left(C_0^{K_{1,\mathrm{GID}}} C_1 C_2^{K_{2,\mathrm{GID}}}, \prod_{x\in I} SK_{\rho(x),\mathrm{GID}}^{c_x}\right)} = e(g_1, g_2)^{s_0}. \tag{6}$$

Finally, the message can be recovered as $M = (C/e(g_1, g_2)^{s_0})$

(vii) Trace $(\mathrm{GP}, \mathrm{CPK}, \{\mathrm{APK_j}\}, \mathrm{SK}_{S,\mathrm{GID}})$. If the decryption key $\mathrm{SK}_{S,\mathrm{GID}}$ is not in the form of $\mathrm{SK}_{S,\mathrm{GID}} = (K_{1,\mathrm{GID}}, K_{2,\mathrm{GID}}, \left\{\mathrm{SK}_{i,\mathrm{GID}}\right\}_{i\in S})$, it outputs $\top$. Otherwise, it runs a key sanity check on $\mathrm{SK}_{S,\mathrm{GID}}$ as follows: $K_{1,\mathrm{GID}} \in \mathbb{Z}_p, K_{2,\mathrm{GID}} \in \mathbb{Z}_p$, $\exists i \in S$, s.t. $\mathrm{SK}_{i,\mathrm{GID}} \in G_2^3$ and

$$e\left(g_1^{\vec{b}_1 K_{1,\mathrm{GID}}}\mathrm{cpk}_1\mathrm{cpk}_2^{K_{2,\mathrm{GID}}}, \mathrm{SK}_{i,\mathrm{GID}}\right) = e\left(\mathrm{apk}_{i,1}, H\left(K_{1,\mathrm{GID}}\right)\right)\mathrm{apk}_{i,2}. \quad (7)$$

If $\mathrm{SK}_{S,\mathrm{GID}}$ passes the above check, it outputs the identity $K_{1,\mathrm{GID}}$. Otherwise, it outputs $\top$.

(viii) Correctness

If the attributes set S satisfies the policy (A,ρ), we have that $\sum_{x\in I} c_x \vec{A_x} = (1,0,\ldots,0)$. Then,

$$\begin{aligned}&\sum_{x\in I} c_x\lambda_x = \sum_{x\in I} c_x \vec{A_x}\vec{v}^{\top} = s_0,\\ &\sum_{x\in I} c_x\left(\vec{0}^{\top}, sU_2^{\top}\vec{b}_1,\ldots,sU_l^{\top}\vec{b}_1\right)\vec{A_x}^{\top} = \left(\vec{0}^{\top}, sU_2^{\top}\vec{b}_1,\ldots,sU_l^{\top}\vec{b}_1\right)\\ &(1,0,\ldots,0)^{\top} = \vec{0}^{\top}.\end{aligned} \quad (8)$$

Therefore,

$$\begin{aligned}\prod_{x\in I} C_{1,x}^{c_x} &= e(g_1,g_2)^{\sum_{x\in I} c_x\lambda_x} e(g_1,g_2)^{\sum_{x\in I} c_x \overrightarrow{sk_{\rho(x)}}\vec{b}_1}\\ &= (g_1,g_2)^{s_0} e(g_1,g_2)^{\sum_{x\in I} c_x \overrightarrow{sk_{\rho(x)}}\vec{b}_1}.\end{aligned} \quad (9)$$

Note that $H(K_{1,\mathrm{GID}}) = H(\mathrm{GID}) \in G_2^3$, so there exists an unknown vector $\overrightarrow{t_{\mathrm{GID}}} \in \mathbb{Z}_p^3$ such that $H(\mathrm{GID}) = g_2^{\overrightarrow{t_{\mathrm{GID}}}}$. Then, we have

$$\begin{aligned}e\left(\prod_{x\in I} C_{2,x}^{c_x}, H\left(K_{1,\mathrm{GID}}\right)\right) &= e\left(\prod_{x\in I} g_1^{\left(\vec{0}^{\top}, sU_2^{\top}\vec{b}_1,\ldots,sU_l^{\top}\vec{b}_1\right)\vec{A_x}^{\top} + sY_{\rho(x)}^{\top}\vec{b}_1}, g_2^{\overrightarrow{t_{\mathrm{GID}}}}\right)\\ &= e\left(g_1^{\sum_{x\in I} c_x sY_{\rho(x)}^{\top}\vec{b}_1}, g_2^{\overrightarrow{t_{\mathrm{GID}}}}\right) = e(g_1,g_2)^{\overrightarrow{t_{\mathrm{GID}}}\sum_{x\in I} c_x sY_{\rho(x)}^{\top}\vec{b}_1},\\ e\left(C_0^{K_{1,\mathrm{GID}}}C_1C_2^{K_{2,\mathrm{GID}}}, \prod_{x\in I} SK_{\rho(x),\mathrm{GID}}^{c_x}\right) &= e\left(g_1^{s(a+\mathrm{GID}+\mathrm{br}_{\mathrm{GID}})\vec{b}_1}, \prod_{x\in I} g_2^{\left(c_x\overrightarrow{k_{\rho(x)}}/(a+\mathrm{GID}+\mathrm{br}_{\mathrm{GID}})\right)} H(\mathrm{GID})^{\left(c_x Y_{\rho(x)}^{\top}/(a+\mathrm{GID}+\mathrm{br}_{\mathrm{GID}})\right)}\right)\\ &= e\left(g_1^{s\vec{b}_1}, g_2^{\sum_{x\in I} c_x\overrightarrow{k_{\rho(x)}}} g_2^{\overrightarrow{t_{GID}}\sum_{x\in I} c_x Y_{\rho(x)}^{\top}}\right)\\ &= e(g_1,g_2)^{\sum_{x\in I} c_x \overrightarrow{sk_{\rho(x)}}\vec{b}_1} e(g_1,g_2)^{\overrightarrow{t_{\mathrm{GID}}}\sum_{x\in I} c_x sY_{\rho(x)}^{\top}\vec{b}_1}.\end{aligned} \quad (10)$$

Hence, $\prod_{x\in I} C_{1,x}^{c_x}\left(e\left(\prod_{x\in I} C_{2,x}^{c_x}, H(K_{1,\mathrm{GID}})\right)/e\left(C_0^{K_{1,\mathrm{GID}}}C_1 C_2^{K_{2,\mathrm{GID}}}, \prod_{x\in I}\mathrm{SK}_{\rho(x),}\ \mathrm{GID}^{c_x}\right)\right) = e(g_1,g_2)^{s_0}$.

4.2. Security Analysis. In this section, we first prove that our T-MA-CP-ABE scheme is adaptively secure based on the SXDH assumption by a reduction to the underlying scheme in [20]. More specifically, we assume an adversary $\mathscr{A}$ breaks our T-MA-CP-ABE scheme in the random oracle model with advantage ε; then, we build a simulator $\mathscr{B}$ that breaks the scheme [20] in the random oracle model with advantage ε. Then, we prove our T-MA-CP-ABE scheme is fully traceable based on the q-SDH assumption by a reduction to a signature scheme [32]. More specifically, we assume an adversary $\mathscr{A}$ breaks our T-MA-CP-ABE scheme in the traceability game; then, we build a simulator $\mathscr{B}$ that breaks the signature scheme [32] under an adaptive chosen message attack.

4.2.1. Adaptive Security. Note that there are two typos (that make encryption and decryption algorithms cannot be completely executed) in the scheme [20] that should be corrected: $C_{1,x}$ should be corrected as $C_{1,x} = e(g_1,g_2)^{\lambda_x} e(g_1,g_2)^{\overrightarrow{k_{\rho(x)}}\vec{b}_1\vec{s}}$, and $\mathrm{SK}_{\mathrm{GID},i}$ should be corrected as $\mathrm{SK}_{\mathrm{GID},i} = g_2^{\vec{k}_i} H(\mathrm{GID})^{Y_i^{\top}}$. We denote the scheme [20] with $k=1$ as ZCGM1 scheme, which has been proved adaptively secure in [20].

Lemma 1 (see [20]). *If the SXDH assumption holds, then the ZCGM1 scheme is adaptively secure in the random oracle model.*

Lemma 2. *Assuming that the ZCGM1 scheme [20] is adaptively secure, then our T-MA-CP-ABE scheme is adaptively secure.*

Proof. Let $\mathscr{C}$ denote the challenger corresponding to $\mathscr{B}$ in the adaptive security game of ZCGM1 scheme.

(i) Setup. When $\mathscr{B}$ receives the global parameters GP from $\mathscr{C}$, it picks $a, b \leftarrow_R \mathbb{Z}_p$ and computes $\mathrm{cpk}_1 = (g_1^{\overrightarrow{b_1}})^a = g_1^{a\overrightarrow{b_1}}$, and $\mathrm{cpk}_2 = (g_1^{\overrightarrow{b_1}})^b = g_1^{b\overrightarrow{b_1}}$. Then, $\mathscr{B}$ stores (a, b) and sends GP and CPK = $(\mathrm{cpk}_1, \mathrm{cpk}_2)$ to $\mathscr{A}$. Next, $\mathscr{A}$ submits a corrupt AAs set $V' \subseteq V$ to $\mathscr{B}$, and $\mathscr{B}$ submits $V - V'$ to $\mathscr{C}$ to request the AA public keys for noncorrupt AAs. When $\mathscr{B}$ obtains AA public keys $\{\mathrm{APK}_j\}_{j\in V-V'}$ from $\mathscr{C}$, it sends $\{\mathrm{APK}_j\}_{j\in V-V'}$ to $\mathscr{A}$.

(ii) Phase 1. $\mathscr{B}$ initializes an empty table Q and answers the CA key and AA key queries as follows:

(1) CA key query. When $\mathscr{A}$ submits an identity GID to $\mathscr{B}$ to request the corresponding CA key, $\mathscr{B}$ first searches the entry $(\mathrm{GID}, r_{\mathrm{GID}}, K_{3,\mathrm{GID}}, K_{4,\mathrm{GID}})$ in table Q. If such entry exists, $\mathscr{B}$ returns $(\mathrm{GID}, r_{\mathrm{GID}}, K_{3,\mathrm{GID}}, K_{4,\mathrm{GID}})$ to $\mathscr{A}$. Otherwise, $\mathscr{B}$ picks $r_{\mathrm{GID}} \leftarrow_R \mathbb{Z}_p \setminus \{-(a + \mathrm{GID}/b)\}$ and computes $K_{3,\mathrm{GID}} = g_2^{(1/a+\mathrm{GID}+br_{\mathrm{GID}})}$, $K_{4,\mathrm{GID}} = H(\mathrm{GID})^{(1/a+\mathrm{GID}+br_{\mathrm{GID}})}$. Then, $\mathscr{B}$ sends $\mathrm{CSK}_{\mathrm{GID}} = (\mathrm{GID}, r_{\mathrm{GID}}, K_{3,\mathrm{GID}}, K_{4,\mathrm{GID}})$ to $\mathscr{A}$ and stores it in Q.

(2) AA key query. When $\mathscr{A}$ submits a pair (S, GID) to $\mathscr{B}$ to request the corresponding decryption key, $\mathscr{B}$ first searches the entry $(\mathrm{GID}, r_{\mathrm{GID}}, K_{3,\mathrm{GID}}, K_{4,\mathrm{GID}})$ in table Q. If such entry exists, $\mathscr{B}$ can obtain r_{GID} from table Q. Otherwise, $\mathscr{B}$ picks $r_{\mathrm{GID}} \leftarrow_R \mathbb{Z}_p \setminus \{-(a + \mathrm{GID}/b)\}$, computes $K_{3,\mathrm{GID}} = g_2^{(1/a+\mathrm{GID}+br_{\mathrm{GID}})}$, $K_{4,\mathrm{GID}} = H(\mathrm{GID})^{(1/a+\mathrm{GID}+br_{\mathrm{GID}})}$, and stores $(\mathrm{GID}, r_{\mathrm{GID}}, K_{3,\mathrm{GID}}, K_{4,\mathrm{GID}})$ in table Q. Then, $\mathscr{B}$ calls the ZCGM1 AA key generation oracle on (S, GID) to obtain the private key $\{\mathrm{SK}_{\mathrm{GID},i}\}_{i\in S}$. For each $i \in S$, $\mathscr{B}$ computes $\mathrm{SK}_{i,\mathrm{GID}} = \mathrm{SK}_{\mathrm{GID},i}^{(1/a+\mathrm{GID}+br_{\mathrm{GID}})}$. Finally, $\mathscr{B}$ sets the corresponding decryption key as $\mathrm{SK}_{S,\mathrm{GID}} = (\mathrm{GID}, r_{\mathrm{GID}}, \{\mathrm{SK}_{i,\mathrm{GID}}\}_{i\in S})$ and sends it to $\mathscr{A}$.

(iii) Challenge. The adversary $\mathscr{A}$ submits two messages M_0, M_1 and an access policy (A, ρ), where (A, ρ) satisfies the following constraint. Let $S_{V'}$ denote the attributes controlled by corrupt AAs, and S_{GID} denotes the attributes in which the adversary has queried for identity GID. For each GID, we require that $S_{V'} \cup S_{\mathrm{GID}}$ does not satisfy (A, ρ). The challenger $\mathscr{B}$ sends M_0, M_1, and (A, ρ) to $\mathscr{C}$ to obtain the ZCGM1 challenge ciphertext $\mathrm{CT} = (C, C_0, \{C_{1,x}, C_{2,x}\}_{x\in\{1,2,\ldots,n\}})$. Then, $\mathscr{B}$ computes $C_1 = C_0^a = g_1^{sa\overrightarrow{b_1}}$, $C_2 = C_0^b = g_1^{sb\overrightarrow{b_1}}$ and sends $\mathrm{CT}^* = (C, C_0, C_1, C_2, \{C_{1,x}, C_{2,x}\}_{x\in\{1,2,\ldots,n\}})$ to $\mathscr{A}$.

(iv) Phase 2. The adversary makes the key queries as Phase 1, but with the restriction of (A, ρ) as described above. $\mathscr{B}$ responds the queries in the same way as Phase 1.

(v) Guess. When $\mathscr{A}$ outputs a guess β', then $\mathscr{B}$ outputs β'.

Since $\mathscr{B}$ perfectly simulates the ZCGM1 security game for $\mathscr{A}$, the advantage of $\mathscr{B}$ breaks the ZCGM1 scheme equals to the advantage of $\mathscr{A}$ breaks our scheme. □

Theorem 1. *If the SXDH assumption holds, then our T-MA-CP-ABE scheme is adaptively secure.*

Proof. This proof follows directly from Lemmas 1 and 2. □

4.2.2. *Traceability.*

Now, we prove our T-MA-CP-ABE scheme is fully traceable by a reduction to BB scheme [32], which is strongly existentially unforgeable.

Lemma 3 (See [32]). *If the q-SDH assumption holds, then the BB scheme is strongly existentially unforgeable under an adaptive chosen message attack.*

Lemma 4. *Assuming that the BB scheme [32] is strongly existentially unforgeable under an adaptive chosen message attack, then our T-MA-CP-ABE scheme is fully traceable in the random oracle model.*

Proof. Let $(p, G_1, G_2, G_T, g_1, g_2, e)$ be a prime order bilinear group, and $\{p, g_1, g_2, g_1^a, g_1^b\}$ be the public key of BB scheme. Let S_j be the attributes set managed by attribute authority AA_j, and $\mathscr{C}$ be the challenger corresponding to $\mathscr{B}$ in the BB security game.

(i) Setup. When $\mathscr{B}$ receives public key $\{p, g_1, g_2, g_1^a, g_1^b\}$ from $\mathscr{C}$, it first samples $B = (b_{ij})_{3\times 3} \leftarrow \mathrm{GL}_3(\mathbb{Z}_p)$, sets $B^* = (B^{-1})^\top$, $\overrightarrow{b_1} = (b_{11}, b_{21}, b_{23})^\top$, and $\mathrm{GP} = (p, G_1^3, G_2^3, G_T, g_1, g_2, e, g_1^{\overrightarrow{b_1}})$. Then, $\mathscr{B}$ computes $\mathrm{cpk}_1 = (g_1^{\overrightarrow{b_1}})^a = g_1^{a\overrightarrow{b_1}}$, $\mathrm{cpk}_2 = (g_1^{\overrightarrow{b_1}})^b = g_1^{b\overrightarrow{b_1}}$, and sets $CPK = (\mathrm{cpk}_1, \mathrm{cpk}_2)$. For each attribute $i \in S_j$, $\mathscr{B}$ picks $\overrightarrow{k_i} \leftarrow_R \mathbb{Z}_p^3, \mathbf{Y}_i \leftarrow_R \mathbb{Z}_p^{3\times 3}$, computes $\mathrm{apk}_{i,1} = g_1^{\mathbf{Y}_i^\top \overrightarrow{b_1}}$ and $\mathrm{apk}_{i,2} = e(g_1^{\overrightarrow{b_1}}, g_2^{\overrightarrow{k_i}})$, and sets $\mathrm{APK}_j = \{\mathrm{apk}_{i,1}, \mathrm{apk}_{i,2}\}_{i\in S_j}$. Finally, $\mathscr{B}$ sends global parameters GP, CA public key CPK, and AA public keys $\{\mathrm{APK}_j\}$ to $\mathscr{A}$. $\mathscr{B}$ stores $(\overrightarrow{b_1}, t\{\overrightarrow{k_i}, t\mathbf{Y}_i\})$ and controls the random oracle H.

(ii) Key query. In this phase, $\mathscr{A}$ queries the CA keys corresponding to $\{\mathrm{GID}_i\}_{i=1}^m$ and AA keys corresponding to $\{(S_i, \mathrm{GID}_i)\}_{i=1}^m$. $\mathscr{B}$ initializes two empty tables Q_1 and Q_2 and answers $\mathscr{A}$'s queries as follows:

(1) Random oracle hash query. When $\mathscr{A}$ submits an identity GID to $\mathscr{B}$ to request the corresponding random oracle hash value $H(\mathrm{GID})$, $\mathscr{B}$ first searches the entry $(\mathrm{GID}, \overrightarrow{t_{\mathrm{GID}}}, g_2^{\overrightarrow{t_{\mathrm{GID}}}})$ in Q_1. If such entry exists, $\mathscr{B}$ returns $g_2^{\overrightarrow{t_{\mathrm{GID}}}}$. Otherwise, $\mathscr{B}$ picks $\overrightarrow{t_{\mathrm{GID}}} \leftarrow_R \mathbb{Z}_p^3$, sends $g_2^{\overrightarrow{t_{\mathrm{GID}}}}$ to $\mathscr{A}$, and stores $(\mathrm{GID}, \overrightarrow{t_{\mathrm{GID}}}, g_2^{\overrightarrow{t_{\mathrm{GID}}}})$ in Q_1.

(2) CA key query. When $\mathscr{A}$ submits an identity GID_i to $\mathscr{B}$ to request the corresponding CA key $\mathrm{SK}_{\mathrm{GID}_i}$, $\mathscr{B}$ first searches the entry $\mathrm{SK}_{\mathrm{GID}_i} = (\mathrm{GID}_i, r_{\mathrm{GID}_i}, K_{3,\mathrm{GID}_i}, K_{4,\mathrm{GID}_i})$ in Q_2. If such entry exists, $\mathscr{B}$ returns $\mathrm{SK}_{\mathrm{GID}_i}$ to $\mathscr{A}$. Otherwise, $\mathscr{B}$ submits GID_i to $\mathscr{C}$ to request the corresponding signature. When $\mathscr{B}$ receives signature $(r_{\mathrm{GID}_i}, \sigma_{\mathrm{GID}_i} = g_2^{(1/a+\mathrm{GID}_i+br_{\mathrm{GID}_i})})$ from $\mathscr{C}$, $\mathscr{B}$ searches the entry $(\mathrm{GID}_i, \overrightarrow{t_{\mathrm{GID}_i}}, g_2^{\overrightarrow{t_{\mathrm{GID}_i}}})$ in Q_1. If no such entry exists, $\mathscr{B}$ picks $\overrightarrow{t_{\mathrm{GID}_i}} \leftarrow_R \mathbb{Z}_p^3$ and stores $(\mathrm{GID}_i, \overrightarrow{t_{\mathrm{GID}_i}}, g_2^{\overrightarrow{t_{\mathrm{GID}_i}}})$ in Q_1. Next, $\mathscr{B}$ obtains $\overrightarrow{t_{\mathrm{GID}_i}}$ from table Q_1 and sets $\mathrm{SK}_{\mathrm{GID}_i} = (\mathrm{GID}_i, r_{\mathrm{GID}_i}, K_{3,\mathrm{GID}_i} = \sigma_{\mathrm{GID}_i}, K_{4,\mathrm{GID}_i} = \sigma_{\mathrm{GID}_i}^{\overrightarrow{t_{\mathrm{GID}_i}}})$ as the corresponding CA private key. Finally, $\mathscr{B}$ sends $\mathrm{SK}_{\mathrm{GID}_i}$ to $\mathscr{A}$ and stores it in Q_2.

(3) AA key query. When $\mathscr{A}$ submits a pair (S_i, GID_i) to $\mathscr{B}$ to request the corresponding decryption key $\mathrm{SK}_{S_i,\mathrm{GID}_i}$, $\mathscr{B}$ first searches CA key $\mathrm{SK}_{\mathrm{GID}_i} = (\mathrm{GID}_i, r_{\mathrm{GID}_i}, K_{3,\mathrm{GID}_i}, K_{4,\mathrm{GID}_i})$ in Q_2. If no such entry exists, $\mathscr{B}$ generates CA key SK_{GID_i} as in (2) and stores it in Q_2. For each $l \in S_i$, $\mathscr{B}$ computes $\mathrm{SK}_{l,\mathrm{GID}_i} = K_{3,\mathrm{GID}_i}^{\overrightarrow{k_l}} K_{4,\mathrm{GID}_i}^{\mathbf{Y}_l^\top}$. Finally, $\mathscr{B}$ sends the corresponding decryption key $\mathrm{SK}_{S_i,\mathrm{GID}_i} = (\mathrm{GID}_i, r_{\mathrm{GID}_i}, \{\mathrm{SK}_{l,\mathrm{GID}_i}\}_{l\in S_i})$ to $\mathscr{A}$.

(iii) Key forgery. $\mathscr{A}$ returns a decryption key SK^* to $\mathscr{B}$. If $\mathscr{A}$ wins this game, then **Trace** $(\mathrm{GP}, \mathrm{CPK}, \{\mathrm{APK_j}\}, \mathrm{SK}, \mathrm{SK}^*) \notin \{\top, \mathrm{GID}_1, \ldots, \mathrm{GID}_m\}$. Therefore, the decryption key $\mathrm{SK}^* = (K_{1,\mathrm{GID}}, K_{2,\mathrm{GID}}, \{\mathrm{SK}_{i,\mathrm{GID}}\}_{i\in S})$ passes the key sanity check, and $K_{1,\mathrm{GID}} \notin \{\mathrm{GID}_1, \ldots, \mathrm{GID}_m\}$. Hence, $K_{1,\mathrm{GID}} \in \mathbb{Z}_p, K_{2,\mathrm{GID}} \in \mathbb{Z}_p$, $\exists i \in S$, *s.t.* $\mathrm{SK}_{i,\mathrm{GID}} \in G_2^3$, and

$$e\left(g_1^{\overrightarrow{b_1}K_{1,\mathrm{GID}}}\mathrm{cpk}_1\mathrm{cpk}_2^{K_{2,\mathrm{GID}}}, \mathrm{SK}_{i,\mathrm{GID}}\right) = e\left(\mathrm{apk}_{i,1}, H\left(K_{1,\mathrm{GID}}\right)\right)\mathrm{apk}_{i,2}. \tag{11}$$

$\mathscr{B}$ queries the random oracle hash $H(K_{1,\mathrm{GID}})$ as in (1) and gets the record $(K_{1,\mathrm{GID}}, \overrightarrow{t_{K_{1,\mathrm{GID}}}}, g_2^{\overrightarrow{t_{K_{1,\mathrm{GID}}}}})$ from Q1. Then,

$$\begin{aligned} e\left(g_1^{(K_{1,\mathrm{GID}}+a+bK_{2,\mathrm{GID}})\overrightarrow{b_1}}, \mathrm{SK}_{i,\mathrm{GID}}\right) &= e\left(g_1^{\mathbf{Y}_i^\top\overrightarrow{b_1}}, g_2^{\overrightarrow{t_{K_{1,\mathrm{GID}}}}}\right)e\left(g_1^{\overrightarrow{b_1}}, g_2^{\overrightarrow{k_i}}\right) \\ &= e\left(g_1^{\overrightarrow{b_1}}, g_2^{\overrightarrow{k_i}+\overrightarrow{t_{K_{1,\mathrm{GID}}}}\mathbf{Y}_i^\top}\right). \end{aligned} \tag{12}$$

Hence, we have $\mathrm{SK}_{i,\mathrm{GID}} = g_2^{(\overrightarrow{k_i}+\overrightarrow{t_{K_{1,\mathrm{GID}}}}\mathbf{Y}_i^\top/K_{1,\mathrm{GID}}+a+bK_{2,\mathrm{GID}})}$. As $\mathscr{B}$ knows the pairs $(\overrightarrow{t_{K_{1,\mathrm{GID}}}}, t\overrightarrow{k_i}n, q\mathbf{Y}_i^\top)$, it can compute $\overrightarrow{k_i} + \overrightarrow{t_{K_{1,\mathrm{GID}}}}\mathbf{Y}_i^\top = (a_1, a_2, a_3) \in \mathbb{Z}_p^3$. Then, $\mathscr{B}$ obtains $\mathrm{SK}_{i,\mathrm{GID},1} = g_2^{(a_1/K_{1,\mathrm{GID}}+a+bK_{2,\mathrm{GID}})}$ from $\mathrm{SK}_{i,\mathrm{GID}}$ and sets $\sigma = \mathrm{SK}_{i,\mathrm{GID},1}^{(1/a_1)} = g_2^{(1/K_{1,\mathrm{GID}}+a+bK_{2,\mathrm{GID}})}$.

Since $e(g_1^a g_1^{K_{1,\mathrm{GID}}}(g_1^b)^{K_{2,\mathrm{GID}}}, \sigma) = z$, $\mathscr{B}$ can output a valid signature $(\sigma, K_{2,\mathrm{GID}})$ on message $K_{1,\mathrm{GID}}$ in the BB security game. Note that $K_{1,\mathrm{GID}} \notin \{\mathrm{GID}_1, \ldots, \mathrm{GID}_m\}$, so $\mathscr{B}$ has never queried a signature on $K_{1,\mathrm{GID}}$, and then, $\mathscr{B}$ wins the BB security game. Hence, if $\mathscr{A}$ breaks our T-MA-CP-ABE scheme in the traceability game with advantage ε, then $\mathscr{B}$ breaks the BB scheme with advantage ε. □

Theorem 2. *If the q-SDH assumption holds, then our T-MA-CP-ABE scheme is fully traceable in the random oracle model.*

Proof. This proof follows directly from Lemmas 3 and 4. □

5. Our TR-MA-CP-ABE System

Based on our T-MA-CP-ABE scheme, we design a TR-MA-CP-ABE system for secure and flexible data access control in cloud storage. In our system, each data owner can share his data with multiple data users whose attributes satisfy the specific access policy. The malicious users who leak their decryption keys on the Internet will be caught and revoked by CA. Furthermore, we give an efficiency comparison that shows our system accesses the data significantly faster than other related works.

5.1. Concrete System. Inspired by [9, 19], we extend our construction to realize malicious user revocation by adopting a hash function F: $\{0,1\}^* \longrightarrow \mathbb{Z}_p$ and a revocation identity list RL. In our TR-MA-CP-ABE system, CA adds the malicious user's GID to the revocation identity list RL in the user tracing and revocation phase, and AAs help the nonrevoked users to update their decryption keys in the key update phase. More specifically, we first use the hash function F to map the time period T and user attribute i in $\mathbb{Z}_p$, and then add T into the system by embedding the element $F(T\|i)$ into the decryption key and ciphertext. The malicious users' time elements will not be updated by AAs, so they cannot update their decryption keys and decrypt the new ciphertexts encrypted in new time period.

Let TMABE = (TMABE : GlobalSetup; TMABE : CASetup; TMABE : AASetup; TMABE : CAKeyGen; TMABE : AAKeyGen; TMABE : Encrypt; TMABE : Decrypt; TMABE : Trace.) be the T-MA-CP-ABE scheme in Section 4.1. Below, we give the details of our TR-MA-CP-ABE system.

5.1.1. System Initialization. In this phase, CA generates the system parameters, revocation list, and its public and secret keys. Each AA_j creates a secret key for itself and a corresponding public key for public usage.

CA first runs the algorithm TMABE : GlobalSetup (λ) to obtain $(p, G_1^3, G_2^3, G_T, g_1, g_2, e, g_1^{\vec{b}_1}, H)$ and chooses a hash function F: $\{0,1\}^* \longrightarrow \mathbb{Z}_p$. Then, CA runs the algorithm TMABE : CASetup (GP) to obtain its own public key $CPK = (\mathrm{cpk}_1, \mathrm{cpk}_2)$ and secret key $\mathrm{CSK} = (a, b)$. Next, CA publishes the system parameters $\mathrm{GP} = (p, G_1^3, G_2^3, G_T, g_1, g_2, e, g_1^{\vec{b}_1}, H, F)$ and its public key CPK. Finally, CA initializes an empty revocation identity list RL.

AA_j first runs the algorithm TMABE : AASetup (GP, S_j) to generate its own public key APK_j and secret key ASK_j. Then, AA_j keeps ASK_j secret and publishes APK_j to others.

5.1.2. User Registration. When a data user wants to join the system, he should register himself to the CA and relevant AAs. In this phase, CA issues the identity keys, and AAs issue the attribute keys to the registered users. From the identity and attribute keys, the registered users can crate their decrypt keys, which can be used for decrypting the policy-matching ciphertext.

First, the data user with identity GID makes a registration request to CA. CA runs the algorithm TMABE : CAKeyGen (GID, CSK, GP) to obtain $\mathrm{CSK}_{\mathrm{GID}} = (K_{1,\mathrm{GID}}, K_{2,\mathrm{GID}}, K_{3,\mathrm{GID}}, K_{4,\mathrm{GID}})$, sets $\mathrm{CSK}_{\mathrm{GID}}$ as the user identity key, and sends it to the data user.

Next, the data user submits his identity GID and attributes set S and $(K_{3,\mathrm{GID}}, K_{4,\mathrm{GID}})$ to the relevant authorities $\{\mathrm{AA}_j\}$. For each attribute $i \in S \cap S_j$, AA_j sets the corresponding user attribute key $\mathrm{SK}_{i,\mathrm{GID}} = K_{3,\mathrm{GID}}^{F(T\|i)\vec{k}_i} K_{4,\mathrm{GID}}^{\mathbf{Y}_i^\top}$ and sends it to the data user, where T is a time period.

Finally, the user sets $\mathrm{SK}_{S,\mathrm{GID}} = (K_{1,\mathrm{GID}}, K_{2,\mathrm{GID}}, \{\mathrm{SK}_{i,\mathrm{GID}}\}_{i\in S})$ as his decryption key.

5.1.3. Data Outsource. In this phase, each data owner encrypts his data with a specific access policy and then outsources the ciphertext data in the cloud. When a data owner wants to share data F with the specific data users, he should generate a ciphertext data that is composed of the body and header as follows.

First, the data owner picks a symmetric session key $K \in G_T$, uses it to encrypt the data F under a symmetric encryption algorithm (such as AES), and sets the resulting ciphertext CT_F as the ciphertext body.

Then, the data owner encrypts the session key $K \in G_T$ under an access policy (A, ρ) and a time period T' as follows. He picks $U_2, U_3, \ldots, U_n \leftarrow_R \mathbb{Z}_p^{3\times 3}$, $\vec{0} \leftarrow \mathbb{Z}_p^3$, $\vec{v} = (s_0, v_2, v_3, \ldots, v_n) \leftarrow_R \mathbb{Z}_p^n$, $s \leftarrow \mathbb{Z}_p$. For each $x \in \{1, 2, \ldots, l\}$, he sets $\lambda_x = \langle \vec{A}_x, \vec{v} \rangle$, and computes

$$\begin{aligned}
C &= Ke(g_1, g_2)^{s_0},\\
C_0 &= g_1^{s\vec{b}_1},\\
C_1 &= g_1^{sa\vec{b}_1},\\
C_2 &= g_1^{sb\vec{b}_1},\\
C_{1,x} &= e(g_1, g_2)^{\lambda_x} e(g_1, g_2)^{sF(T'\|\rho(x))\overrightarrow{k_{\rho(x)}}\vec{b}_1},\\
C_{2,x} &= g_1^{\left(\vec{0}^\top, s\mathbf{U}_2^\top \vec{\mathbf{b}}_1, \ldots, s\mathbf{U}_l^\top \vec{\mathbf{b}}_1\right)\vec{A}_x^\top} g_1^{sY_{\rho(x)}^\top \vec{\mathbf{b}}_1}.
\end{aligned} \tag{13}$$

Finally, the data owner sets ciphertext head $CT_K = ((A, \rho), T', C, C_0, C_1, C_2, \{C_{1,x}, C_{2,x}\}_{1,2,\ldots l})$ and outsources the ciphertext data $(\mathrm{CT}_K, \mathrm{CT}_F)$ to the cloud server.

5.1.4. Data Access. In our system, each data user can download any ciphertext data from the cloud server, but can only access the limited plain data by decryption of the corresponding ciphertext data successfully. In this phase, the data user has a decryption key $\mathrm{SK}_{S,\mathrm{GID}}$ for an attributes set S for a time period T and tries to access the data in the cloud.

The data user first queries an interested ciphertext data and gets the ciphertext $(\mathrm{CT}_K, \mathrm{CT}_F)$ from the cloud server. Then, the data user checks whether he has the access permission or not. If S does not satisfy (A, ρ) or $T \neq T'$, then he outputs $\perp$ meaning that he cannot access this data. Otherwise, the data user runs the algorithm TMABE : Decrypt $(\mathrm{GP}, \mathrm{SK}_{S,\mathrm{GID}}, \mathrm{CT}_K)$ and gets the session key K. Finally, the data user decrypts the ciphertext body CT_F by the key K and recovers the plain data F.

5.1.5. Key Update. In this phase, AAs help the nonrevoked data users update their decryption keys in a new time period T''. When the data user wants to update his decryption key, he submits his identity GID and attributes set S and $(K_{3,\mathrm{GID}}, K_{4,\mathrm{GID}})$ to the relevant authorities $\{\mathrm{AA}_j\}$. AA_j first checks whether the data user has been revoked or not. If $\mathrm{GID} \in \mathrm{RL}$, AA_j outputs $\perp$ meaning that the revoked data user cannot update his decryption key. Otherwise, for each attribute $i \in S \cap S_j$, AA_j computes the user attribute key $\mathrm{SK}_{i,\mathrm{GID}} = K_{3,\mathrm{GID}}^{F(T''\|i)\vec{k}_i} K_{4,\mathrm{GID}}^{\mathbf{Y}_i^\top}$. After that, AA_j sends the update

TABLE 2: Efficiency comparison with other related works.

	PK	DK	CT	PID	GO
[13]	$3\|U\| + \|V\| + 3\rho$	$3\|S\| + 3\rho + 1$	$2\|U\| + 4\rho + 2$	$3\|S\| + 3\rho$	Prime
[14]	$\|U\| + D(\|V\| + 3)$	$\|S\| + D(\|V\| + 5)$	$2l + D + 2$	$4\|I\| + D + 2$	Composite
[15]	$2\|V\|$	$2\|S\| + 1$	$4l + 1$	$3\|I\|$	Composite
[16]	$4\|V\|$	$4\|S\| + 1$	$6l + 1$	$3\|I\|$	Prime
This paper	$4\|U\| + 6$	$3\|S\| + 2$	$4l + 10$	6	Prime

attribute key $\{SK_{i,GID}\}_{i\in S\cap S_j}$ to the data user. Finally, the data user sets the decryption key as $SK_{S,GID} = (K_{1,GID}, K_{2,GID}, \{SK_{i,GID}\}_{i\in S})$, which can be used for decrypt the ciphertext data in the new time period T''.

5.1.6. User Tracing and Revocation. In this phase, CA traces the malicious users who leak their decryption key to others and revokes their access permissions in the system. When CA finds a decryption key $SK_{S,GID}$ is sold on the Internet, it first runs the algorithm TMABE : Trace $(GP, CPK, \{APK_j\}, SK_{S,GID})$. If the algorithm outputs $\top$, CA outputs $\top$ meaning that $SK_{S,GID}$ does not need to be traced. If the algorithm outputs an identity GID, then CA sets GID as the identity of the malicious user who leaks his decryption key $SK_{S,GID}$. Finally, CA adds GID into the revocation list RL and updates RL for public usage.

Alike with [9, 19], we view hash function F as a random oracle, and our TR-MA-CP-ABE scheme has the same security conclusion with our T-MA-CP-ABE scheme. The correctness and security proofs are almost the same with that in Section 4.

5.2. Efficiency Comparison. In this section, we give an efficiency comparison between our TR-MA-CP-ABE system with other T-MA-CP-ABE schemes, all of which support multiauthority and traceability. In Table 2, PK represents the public key (including the CA and AA public keys) size, DK represents the decryption key size, CT represents the ciphertext size, PID represents the number of pairing operations in decryption, and GO represents the group order. Let $|U|$ be the number of attributes in the system, $|V|$ the number of AAs in the system, D the number of CAs in the system, ρ the bit length of the user identity, $|S|$ the number of attributes in the decryption key, l the number of rows of the matrix in the access policy, and $|I|$ ($|I| \le |S|, l$) the number of attributes used for decryption.

As shown in Table 2, the number of resource-consuming pairing operations required to decrypt a ciphertext in [13–16] increases with the number of attributes used for decryption. While our TR-MA-CP-ABE system only needs to compute 6 pairings for decryption. Since an element in prime order groups is 12 times shorter than that in composite order groups [33], the storage overhead of our system is significantly smaller than that of the schemes [14, 15] which are constructed in composite order groups. Compared with our TR-MA-CP-ABE system, the schemes [13, 16] in prime order groups achieve smaller public key

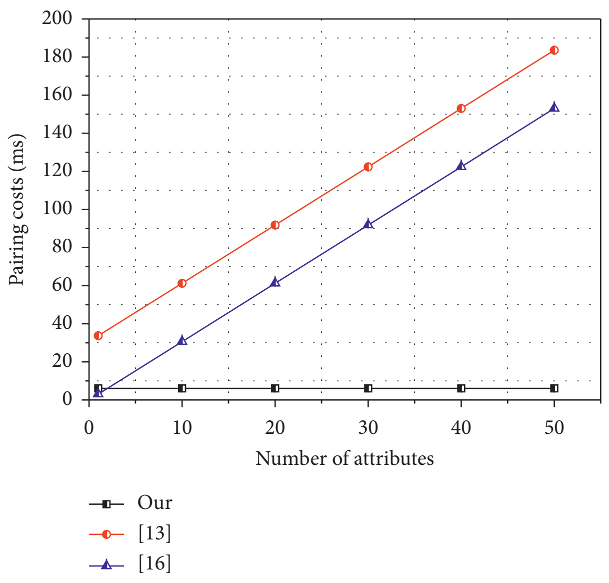

FIGURE 2: Pairing costs for decryption.

size, but they neither achieve adaptively secure nor support user revocation.

We evaluate our pairing operations in Python language using the PBC library [34] with type A curve. The experiment is performed on a Macbook laptop with a 2.8 GHz Intel Core i7 processor and 16 GB memory. Figure 2 illustrates the pairing costs for decryption in our TR-MA-CP-ABE system and other two T-MA-CP-ABE schemes [13, 16] in prime order groups. We set $|S| = |I|$ and $\rho = 10$ and increase the value of $|I|$ from 1 to 50. It is easy to see that pairing costs for decryption in our system is a constant time and significantly shorter than that grows linearly with the number of attributes in [13, 16]. Note that a pairing operation in prime order groups is about 100 times faster than that in composite (3 primes) order groups [35], which makes the data access speed in our system is significantly faster than that in schemes [14, 15] constructed in composite order groups.

6. Conclusion and Future Work

In this work, we presented a traceable and revocable MA-CP-ABE system in the prime order groups. Specifically, the proposed system has the following advantages: (1) the ciphertext cannot be decrypted by any individual authority, and the ciphertext policy can be any monotone access structures; (2) CA can not only catch the malicious user by his decryption key but also revoke the corresponding decryption privilege; and (3) the system achieves adaptively secure and fast access.

As far as we know, our TR-MA-CP-ABE system is the first MA-CP-ABE system that supports traceability, revocation, and fast access simultaneously. However, our system only supports white-box traceability: the decryption key leaked by the malicious user is assumed to pass the key sanity check. Hence, our system is not suitable for black-box traceability scenario: the malicious user can construct a decryption black-box by his decryption key and unknown decryption algorithm and leak a decryption black-box instead of his decryption key. We leave it as our future work to obtain a black-box traceable and revocable MA-CP-ABE system with fast access.

Data Availability

No data were used to support the findings of this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (Grant nos. 61802243, 11801345, and 62072352), the Natural Science Foundation of Shaanxi Province (Grant nos. 2019JQ-273, 2020JM-288, 2019JQ-472, 2020CGXNG-002, and 2019ZDLGY13-03-01), the Key Research and Development Program in Industry Field of Shaanxi Province (Grant no. 2019GY-013), the China Postdoctoral Science Foundation (Grant no. 2018M633456), the project "The Verification Platform of Multi-tier Coverage Communication Network for Oceans" (Grant no. LZC0020), Guangxi Key Laboratory of Trusted Software (Grant no. KX202035), and the Fundamental Research Funds for the Central Universities (Grant no. GK201903011).

References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: a survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] K. Huang, Q. Zhang, C. Zhou, N. Xiong, and Y. Qin, "An efficient intrusion detection approach for visual sensor networks based on traffic pattern learning," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 10, pp. 2704–2713, 2017.

[3] Y. Yang, N. Xiong, N. Y. Chong et al., "A decentralized and adaptive flocking algorithm for autonomous mobile robots," in *Proceedings of the 3rd International Conference on Grid and Pervasive Computing*, pp. 262–268, IEEE, Kunming, China, May 2008.

[4] W. Wu, N. Xiong, and C. Wu, "Improved clustering algorithm based on energy consumption in wireless sensor networks," *IET Networks*, vol. 6, no. 3, pp. 47–53, 2017.

[5] R. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges," *IEEE Communications Magazine*, vol. 49, no. 11, pp. 32–39, 2011.

[6] S. Kamara and K. Lauter, "Cryptographic cloud storage," in *Proceedings of the International Conference on Financial Cryptography and Data Security*, pp. 136–149, Springer, Tenerife, Spain, January 2010.

[7] A. Shahzad, M. Lee, Y.-K. Lee et al., "Real time MODBUS transmissions and cryptography security designs and enhancements of protocol sensitive information," *Symmetry*, vol. 7, no. 3, pp. 1176–1210, 2015.

[8] Q. Zhang, C. Zhou, N. Xiong et al., "Multimodel-based incident prediction and risk assessment in dynamic cybersecurity protection for industrial control systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 10, pp. 1429–1444, 2015.

[9] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Proceedings of the Annual International Cryptology Conference*, pp. 213–229, Springer, Santa Barbara, CA, USA, August 2001.

[10] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proceedings of the 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 457–473, Aarhus, Denmark, May 2005.

[11] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute based encryption," in *Proceedings of the 28th IEEE Symposium on Security and Privacy*, pp. 321–334, Berkeley, CA, USA, May 2007.

[12] M. Chase, "Multi-authority attribute based encryption," in *Proceedings of the 4th Theory of Cryptography Conference*, pp. 515–534, Amsterdam, The Netherlands, February 2007.

[13] J. Li, Q. Huang, X. Chen et al., "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, May 2011.

[14] J. Zhou, Z. Cao, X. Dong et al., "TR-MA-CP-ABE: 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*, pp. 2398–2406, Hong Kong, China, May 2015.

[15] G. Yu, X. Ma, Z. Cao et al., "Accountable multi-authority ciphertext-policy attribute-based encryption without key escrow and key abuse," in *Proceedings of the International Symposium on Cyberspace Safety and Security*, pp. 337–351, Guangzhou, China, December 2017.

[16] K. Zhang, H. Li, J. Ma et al., "Efficient large-universe multi-authority ciphertext-policy attribute-based encryption with white-box traceability," *Science China Information Sciences*, vol. 61, no. 3, p. 13, Article ID 032102, 2018.

[17] M. Chase and S. S. M. Chow, "Improving privacy and security in multi-authority attribute-based encryption," in *Proceedings of the 16th ACM Conference on Computer and Communications Security*, pp. 121–130, Chicago, IL, USA, November 2009.

[18] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Proceedings of the 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 568–588, Tallinn, Estonia, May 2011.

[19] H. Cui and R. H. Deng, "Revocable and decentralized attribute-based encryption," *The Computer Journal*, vol. 59, no. 8, pp. 1220–1235, 2016.

[20] J. Zhang, J. Chen, A. Ge et al., "Shorter decentralized attribute-based encryption via extended dual system groups," *Security and Communication Networks*, vol. 201719 pages, Article ID 7323158, 2017.

[21] Z. Wang, X. Fan, and F. H. Liu, "Fe for inner products and its application to decentralized abe," in *Proceedings of the IACR*

International Workshop on Public Key Cryptography, pp. 97–127, Edinburgh, UK, May 2019.

[22] S. Xiong, Q. Ni, L. Wang, and Q. Wang, "SEM-ACSIT: secure and efficient multiauthority access control for IoT cloud storage," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2914–2927, 2020.

[23] M. J. Hinek, S. Jiang, R. Safavi-Naini et al., "Attribute-based encryption with key cloning protection," IACR Cryptology ePrint Archive, 2008, https://eprint.iacr.org/2008/478Report 2008/478.

[24] Z. Liu, Z. Cao, and D. S. Wong, "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, 2013.

[25] Z. Liu, Z. Cao, and D. S. Wong, "Traceable CP-ABE: How to trace decryption devices found in the wild," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 1, pp. 55–68, 2015.

[26] J. Ning, Z. Cao, X. Dong et al., "Cryptcloud+: secure and expressive data access control for cloud storage," *IEEE Transactions on Services Computing*, In press.

[27] 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.

[28] S. Xu, J. Yuan, G. Xu et al., "Efficient ciphertext-policy attribute-based encryption with blackbox traceability," *Information Sciences*, vol. 538, pp. 19–38, 2020.

[29] D. Han, N. Pan, and K. C. Li, "A traceable and revocable ciphertext-policy attribute-based encryption scheme based on privacy protection," *IEEE Transactions on Dependable and Secure Computing*, In press.

[30] A. Beimel, *Secure schemes for secret sharing and key distribution*, Dissertation for Ph.D. Degree, Technion-Israel Institute of technology, Haifa, Israel, 1996.

[31] L. Ballard, M. Green, B. de Medeiros et al., "Correlation-resistant storage via keyword-searchable encryption," IACR Cryptology ePrint Archive, 2005, https://eprint.iacr.org/2005/417Report 2005/417.

[32] D. Boneh and X. Boyen, "Short signatures without random oracles and the SDH assumption in bilinear groups," *Journal of Cryptology*, vol. 21, no. 2, pp. 149–177, 2008.

[33] A. Guillevic, "Comparing the pairing efficiency over compositeorder and prime-order elliptic curves," in *Proceedings of the 11th International Conference Applied Cryptography and Network Security*, pp. 357–372, Berlin, Heidelberg, Germany, June 2013.

[34] B. Lynn, "The stanford pairing based crypto library," http://crypto.stanford.edu/pbc.

[35] 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, San Juan, PR, USA, January 2015.