Hindawi
Security and Communication Networks
Volume 2021, Article ID 9174630, 15 pages
https://doi.org/10.1155/2021/9174630

Research Article

Multi-Authority Criteria-Based Encryption Scheme for IoT

Jianguo Sun, Yang Yang, Zechao Liu, and Yuqing Qiao

College of Computer Science and Technology, Harbin Engineering University, Harbin 150001, China

Correspondence should be addressed to Zechao Liu; liuzechao@hrbeu.edu.cn

Received 6 April 2021; Accepted 7 July 2021; Published 17 July 2021

Academic Editor: Qi Jiang

Currently, the Internet of Things (IoT) provides individuals with real-time data processing and efficient data transmission services, relying on extensive edge infrastructures. However, those infrastructures may disclose sensitive information of consumers without authorization, which makes data access control to be widely researched. Ciphertext-policy attribute-based encryption (CP-ABE) is regarded as an effective cryptography tool for providing users with a fine-grained access policy. In prior ABE schemes, the attribute universe is only managed by a single trusted central authority (CA), which leads to a reduction in security and efficiency. In addition, all attributes are considered equally important in the access policy. Consequently, the access policy cannot be expressed flexibly. In this paper, we propose two schemes with a new form of encryption named multi-authority criteria-based encryption (CE) scheme. In this context, the schemes express each criterion as a polynomial and have a weight on it. Unlike ABE schemes, the decryption will succeed if and only if a user satisfies the access policy and the weight exceeds the threshold. The proposed schemes are proved to be secure under the decisional bilinear Diffie–Hellman exponent assumption (q-BDHE) in the standard model. Finally, we provide an implementation of our works, and the simulation results indicate that our schemes are highly efficient.

1. Introduction

As an emerging concept, the Internet of Things (IoT) offers great convenience to our daily lives since it provides individuals with ultra-fast data transmission and quality storing services by edge infrastructure. Many well-known IT enterprises such as Google, Microsoft, and Amazon have deployed edge computing platforms to integrate edge infrastructure and various devices, so that individuals can benefit in many fields [1]. Unfortunately, due to the complexity of architecture, there are inevitably some security risks in IoT, especially that some unsupervised edge infrastructures may quietly capture users' sensitive information or be compromised by malicious users, which poses a severe threat to individuals [2, 3]. For example, edge devices may reveal sensitive data such as health records and personal finances to the public. Therefore, data security in IoT has become a significant concern for many enterprises or individuals.

To alleviate this situation, Yeh et al. [4] proposed an access control framework for IoT with the property of attribute revocation. Qiu et al. [5] constructed an authentication and key agreement (AKA) protocol for lightweight devices in IoT. The protocol was proved to be secure in the random oracle model and enjoyed desirable computing efficiency. Wang et al. [6] conducted a detailed analysis of the vulnerability for IoT devices and offered targeted countermeasures depending on the types of attacks. However, traditional public-key techniques only support one-to-one encryption, i.e., messages encrypted by public keys can only be decrypted by their corresponding private keys. This means that there needs to be sufficient storage space to store the ciphertext in practical applications, whereas edge devices generally have limited storage capacity.

Attribute-based encryption (ABE) is an effective encryption tool that provides fine-grained and one-to-many access control for outsourcing data in IoT [7]. According to different encryption mechanisms, ABE can be divided into ciphertext-policy ABE (CP-ABE) and key-policy ABE (KP-ABE). In CP-ABE, the data owner can construct an access policy and embed it into the ciphertext, and the user's attribute set is embedded in the secret key. On the contrary,

the private keys in KP-ABE are associated with the access policy, and the ciphertext is labeled with attributes. A user can successfully recover messages if and only if his/her attributes satisfy the access policy. Many excellent ABE schemes for access control in IoT have been proposed [8–11]. However, most of them have two problems. On the one hand, only a single attribute authority (AA) manages the whole attribute set and generates the secret keys. If a large number of users request private keys, the server will be at risk of crashing. Furthermore, once the attribute authority is compromised, any user with unauthorized attributes will be able to decrypt the ciphertext. Therefore, ABE schemes supporting multiple authorities should be considered, i.e., the attribute universe should be managed by multiple attribute authorities. In this way, even if an authority compromises or collapses, a user can still obtain the secret key from other authorities. On the other hand, all attributes in the access policy of the previous schemes are regarded at the same level, which ignores the scenario that some attributes may be more important than the others. More precisely, in an IoT-based medical system, it is desirable to grant doctors higher weights than the nurses.

In order to distinguish the importance among attributes, some weighted ABE schemes [12–14] have been proposed. Liu et al. [12] proposed a weighted CP-ABE scheme. However, in the scheme, the attribute universe is managed by a single central authority. Wang et al. [13] constructed a multi-authority weighted ABE scheme in cloud computing. In the scheme, CA is still required in the key generation phase, which reduces the security of the scheme. Yan et al. [14] introduced a weighted attribute-based encryption scheme. However, the weight corresponding to each attribute is specified by a central authority, while in the actual scenario of encryption, the data owner should be allowed to decide the weight of each attribute in the access policy. To address the above problems, Phuong et al. [15] first proposed criteria-based encryption (CE) scheme, which supports the weighting of each criterion in the access policy. To be precise, each criterion is expressed as a polynomial, each root of which corresponds to a case satisfying the polynomial-associated criterion. The access policy consists of a series of weighted criteria containing at least one case. For this, the main difference between ABE and CE is that each criterion contains multiple satisfying cases and has a reasonable weight specified by the encryptor. An instance of intuition is provided as follows. Suppose that in a smart medical system, the government needs to monitor the health of community members. Since medical data involve sensitive information of individuals and are not available to others, the receivers need to meet certain restrictions to make access possible ((the receiver must be an authorized chief physician, weighted 5, and marked as a criterion P_1) AND (the receiver has more than 5 years of work experience, weighted 2, and marked as a criterion P_2) OR (the receiver is a community manager employed by the government, weighted 1, and marked as a criterion P_3) OR (the receiver is a community member holding a legal device, weighted 6, and marked as a criterion P_4)). And in order to access the data, the cumulative weight of the receiver must be more than 5. Bob is a community manager hired by the government and has 6 years of work experience related to medical treatment. He cannot obtain approval for not reaching the cumulative weight threshold as required. Alice is a chief physician who has served the community for seven years. She satisfies both the access policy and the threshold, so she can be authorized. As shown in Figure 1, the criterion P_3 corresponds to two cases (roots): the receiver is a community manager and appointed by the government. But unfortunately, the issue of generating keys by only a single authority is still unsolved in their scheme.

In this paper, we propose two types of multi-authority criteria-based encryption schemes, named MA-CE-Verify Root and MA-CE-Root Equality, respectively, which aim to solve the problems we mentioned above. Specifically, we denote each criterion as a polynomial. One can assign a weight for each criterion freely according to demands. In addition, the corresponding cases of satisfying the criteria are represented as the roots of polynomials. In the first scheme, at least a case (or root) of each criterion specified in the access policy should be held by the decryptor, and the cumulative weight needs to exceed the threshold as well for successful decryption, while in the second scheme, only if the decryptor satisfies all the cases (or all roots) for each criterion and the cumulative weight exceeds the threshold, he/she can decrypt correctly. Moreover, in our schemes, multiple authorities manage the global criterion universe and perform key generation, which solves the bottleneck of performance and improves the security of the system.

1.1. Our Contributions. In this work, our main contributions can be summarized as follows:

(1) We propose two types of multi-authority criteria-based encryption schemes, which support the weighting of each criterion. In our schemes, multiple AAs jointly manage the criterion universe using the (t, n)-threshold sharing technology. Furthermore, data owners can freely set the weight of each criterion as required. Thus, flexible access control is provided by our schemes.

(2) The security proof shows that our schemes achieve indistinguishability under chosen-plaintext attack (IND-CPA) under the decisional bilinear Diffie–Hellman exponent assumption (q-BDHE).

(3) We implement the proposed schemes and provide theoretical analysis. The results show that our constructions have desirable performance in practical situations.

1.2. Related Work. Goyal et al. [16] proposed attribute-based encryption (ABE) that provides one-to-many encryption. In their works, ABE is divided into two forms: ciphertext-policy ABE (CP-ABE) and key-policy ABE (KP-ABE). Sahai et al. [17] realized a revocable ABE (RABE) scheme, in which the outsourcing server updates the encrypted data to revoke the user's decryption permission. On the downside, the complexity of bilinear-pairing operations makes it difficult to

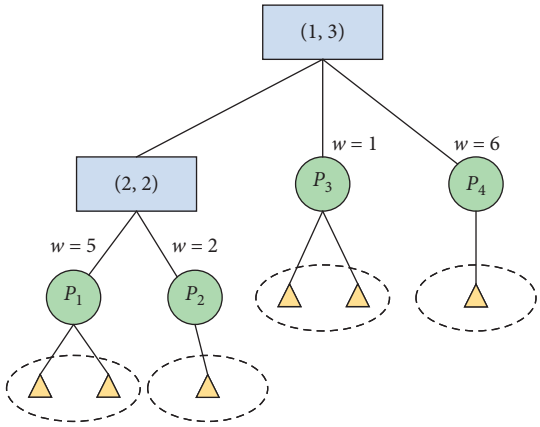

FIGURE 1: The example of access policy.

directly apply this scheme to IoT. Agrawal et al. [18] proposed two versatile ABE architectures with short ciphertext and key. One limitation is that the scheme does not consider that different attributes in the access policy are at different levels of importance, i.e., the attributes do not carry reasonable weights. Waters [19] and Agrawal et al. [20] proposed ABE schemes that support arbitrary length input and provide a general ABE structure. In these schemes, the management of attribute universe and key generation are only implemented by a single attribute authority. Once the authority is corrupted, the adversary can directly generate the key of any user with legal status to decrypt the message [21]. ABE schemes with multiple authorities have been proposed to solve this issue. Lewko et al. [22] constructed an ABE scheme in which any party can become an attribute authority. Moreover, the scheme can resist collision attacks. However, the construction based on composite order group seriously affects the execution efficiency of the scheme. In [23–26], the schemes are provided for different practical application scenarios. Unfortunately, these schemes are limited by some security issues or computational complexity. In this context, there are obstacles to directly applying them in IoT scenarios. Sandor et al. [27] presented an efficient decentralized multi-authority ABE scheme that can significantly solve the key escrow problem for mobile devices. Generally, decentralizing ABE solves the problem of accessing encrypted data when the attributes of users come from multiple authorities, in which each authority is only in charge of issuing attributes and keys in its domain. However, in the schemes, an adversary can still compromise the server of AA to obtain some information that he should not have. The issue can be solved by using (t, n)-threshold sharing in our works. The adversary cannot get any information related to the key unless the number of corrupted authorities is greater than t.

1.3. Organization. In Section 2, we present the notation and preliminaries. In Section 3, we provide three components. The system model and some requirements of the schemes are described in Section 3.1. We define the framework of the schemes in Section 3.2, while the security model is given in Section 3.3. In Section 4, we illustrate how to construct our two schemes. We give the security proof of our schemes in Section 5. The performance analysis of proposed schemes is represented in Section 6. At the end of our work, the conclusions and extensions are put forward in Section 7.

2. Preliminaries

We now introduce some notations and preliminaries.

2.1. Notation. For a positive integer n, $[1, n] = \{1, 2, \ldots, n\}$. For vector $\overrightarrow{u}$ and $\overrightarrow{v}$, let $\langle \overrightarrow{u}, \overrightarrow{v} \rangle$ be the inner product of two vectors. We use $a \in_R S$ to denote a random element a drawn from set S uniformly. For a matrix M, its i-th row is denoted by M_i, and its (i, j)-element is $M_{i,j}$. We use the symbol $C \vDash \mathbb{A}$ to denote the criterion set C satisfies the access structure $\mathbb{A}$. Note that the (monotonic) access structure used in this work is similar to that in literature [8], so the concrete concept is not repeated here. For any set S, $\text{len}(S)$ denotes the number of its elements.

2.2. Bilinear Maps. Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of order p, where p is a large prime number and $\mathbb{G}$ is generated by g. Let e: $\mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$ be an admissible bilinear map, if it satisfies the following properties:

(1) Bilinearity: for any $g, h \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, $e(g^a, h^b) = e(g, h)^{ab}$.

(2) Nondegeneracy: for any $g \in \mathbb{G}$, $e(g, g) \neq 1$.

(3) Computability: for any $g, h \in \mathbb{G}$, there is an efficient algorithm to calculate $e(g, h)$.

2.3. (t, n)-Threshold Secret Sharing. Suppose that several participants intend to share a secret with each other, while they do not hope that any one of them can obtain the secret independently, due to the privacy requirement of the secret. Secret sharing is a technique proposed to be used in the scenario above. In the secret-sharing scheme, each party can obtain a share of the secret, which is actually a part of information about the secret, and the whole secret can be reconstructed only by the cooperation of participants, which means that any party cannot know what the secret is individually. There have been many various secret-sharing schemes suitable for different situations proposed, and the (t, n)-threshold sharing is one of the most widely applicable and basic schemes among them. It was first proposed by Shamir [28] and then improved into many practical schemes, such as [21, 29]. In this work, we adopt the definition in [21].

We take the set $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_n\}$ as n members of the system. The identity of each member x_i $(i \in [1, n])$ is taken from the finite field $\text{GF}(p)$. Let the positive integer t $(t \leq n)$ denote a threshold. Additionally, let S_i represent the subsecret of each member, such that $S = \sum_{i=1}^{n} S_i$. The (t, n)-threshold secret sharing can be described as follows.

Share. Each member constructs the polynomial $q_i(x)$ of degree $t - 1$, such that $S_i = q_i(0)$. For $j = 1$ to n, each

member calculates subshare $\eta_{ij} = q_i(x_j)$ and assigns (x_j, η_{ij}) to member $\mathscr{P}_j$.

Reconstruction. Suppose that there is a function $Q(x)$, such that $Q(x) = \sum_{j=1}^{n} q_j(x)$. Each member calculates the share $\eta_i = \sum_{j=1}^{n} \eta_{ji} = \sum_{j=1}^{n} Q(x_i)$. The shares of any t members are sufficient to reconstruct the function $Q(x)$ according to the Lagrange interpolating formula. The master secret S can be constructed by $S = Q(0)$.

2.4. Linear Secret-Sharing Schemes. We make use of Linear Secret-Sharing Schemes (LSSSs) in [22]. A secret-sharing scheme Π defined on a set of parties $\mathscr{P}$ is linear over $\mathbb{Z}_p$ if

(1) The shares for each party constitute a vector over $\mathbb{Z}_p$.

(2) The matrix M with ℓ rows and n columns is called the share-generating matrix. And the function ρ maps M_i to a party $\rho(i)$, where $i \in [1, n]$. When it comes to the column vector $\overrightarrow{v} = (s, r_2, r_3, \ldots, r_n) \in \mathbb{Z}_p^n$, where $s \in \mathbb{Z}_p$ is the secret to be shared and $r_2, r_3, \ldots, r_n \in_R \mathbb{Z}_p$ are randomly chosen, then $M\overrightarrow{v}$ is the vector composed of ℓ shares of the secret in accordance with Π. The share $(M\overrightarrow{v})_i$ belongs to party $\rho(i)$.

Linear reconstruction is defined as follows: suppose that Π is an LSSS of the access structure $\mathbb{A}$. Let $S \in \mathbb{A}$ be any authorized set, and define $I \subset \{1, 2, \ldots, \ell\}$ as $I = \{i: \rho(i) \in S\}$. Then, there exists a set of constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ that satisfy the proposition; if $\{\lambda_i\}$ are valid shares of any secret s according to Π, then $s = \sum_{i \in I} \omega_i \lambda_i$.

Definition 1 (Decisional Bilinear Diffie–Hellman Exponent Assumption (q-BDHE)). Let $\mathbb{G}$ be a group of prime order p and g_i be short for g^{a^i}. Given $a, s \in \mathbb{Z}_p$ and $h = g^s$, the decision q-BDHE problem [30] can be defined as follows: the adversary is given a vector

$$\overrightarrow{y} = \left(g, g^s, g_1, \ldots, g_q, g_{q+2}, \ldots, g_{2q}\right), \tag{1}$$

and it is hard to distinguish $e(g_{q+1}, h) \in \mathbb{G}_T$ from a random element in $\mathbb{G}_T$. There is an algorithm $\mathscr{B}$ that outputs $b \in \{0, 1\}$ with advantage ε in solving decisional q-BDHE in $\mathbb{G}$ if

$$\left|\Pr\left[\mathscr{B}\left(\overrightarrow{y}, T = e(g, g)^{a^{q+1}s}\right)\right] - \Pr\left[\mathscr{B}(\overrightarrow{y}, T = R)\right] = 0\right| \geq \varepsilon. \tag{2}$$

The decisional q-BDHE assumption holds if there is no polynomial-time algorithm that can solve the (decision) q-BDHE problem with non-negligible advantage.

Mathematically, the Vieta's theorem is used to express the relationship between the root of a polynomial and its coefficients. In our schemes, it is a building block for computing the elements of the ciphertext/secret key.

Definition 2 (Vieta's theorem) (see [15]). Let $P_i = (a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0)$ represent a polynomial of degree n, and its coefficients are expressed as the vector

$$\overrightarrow{u} = (a_n, a_{n-1}, \ldots, a_1, a_0). \tag{3}$$

For any $\overrightarrow{x}$, we represent as follows:

$$\overrightarrow{x} = \left(\underbrace{x \cdot x \ldots x}_{n}, \underbrace{x \ldots x}_{n-1}, \ldots, x, 1\right), \tag{4}$$

where element x is a root of P_i, if the inner product $\langle \overrightarrow{u}, \overrightarrow{x} \rangle = 0$. Suppose $\{x_i\}_{i \in [1,n]}$ are the roots of P_i; then, we have

$$\begin{cases} x_1 + x_2 + \cdots + x_n = -\left(\dfrac{a_{n-1}}{a_n}\right), \\ \left(x_1 x_2 + x_1 x_3 + \cdots + x_n x_{n-1}\right) = \dfrac{a_{n-2}}{a_n}, \\ x_1 x_2 \cdots x_n = (-1)^n \dfrac{a_0}{a_n}. \end{cases} \tag{5}$$

3. Multi-Authority Criteria-Based Encryption

3.1. System Model and Requirements. In this section, we define the notion of the system model and illustrate some requirements in our multi-authority criteria-based encryption schemes. As shown in Figure 2 [31], the system consists of a global central authority (CA), multiple criterion authorities (AAs), the edge infrastructures, data owners (DO), and data consumers (user). Here, we give the formal definition of them as follows.

(1) The central authority (CA) in the whole system is considered to be completely trusted and in charge of system establishment and initialization, including the generation of system parameters and the master public key. When a user (or AA) requests registration, CA verifies the legitimacy of his identity and assigns a unique gid for the user and an aid for the AA, respectively. Besides, CA determines the threshold t in threshold sharing among attribute authorities, which is necessary for the process of secret key generation. In contrast, we note that CA is not responsible for any other issues in the system except for what has been described above. In other words, CA does not participate in the threshold sharing among AAs and key generation, which is the core of decentralization.

(2) A criterion authority (AA) mainly generates the component of the user secret key associated with the criteria in its domain and plays a role in system establishment as well. What's worthy of mention is that, compared with common multi-authority CP-ABE, in our proposed system, all AAs manage the entire criterion universe together. We use the technique of threshold sharing among AAs so that each AA shares a piece of secret key calling its private key, which can ensure that a malicious user cannot get any information unless the number of corrupted authorities exceeds t. After that, CA accepts public

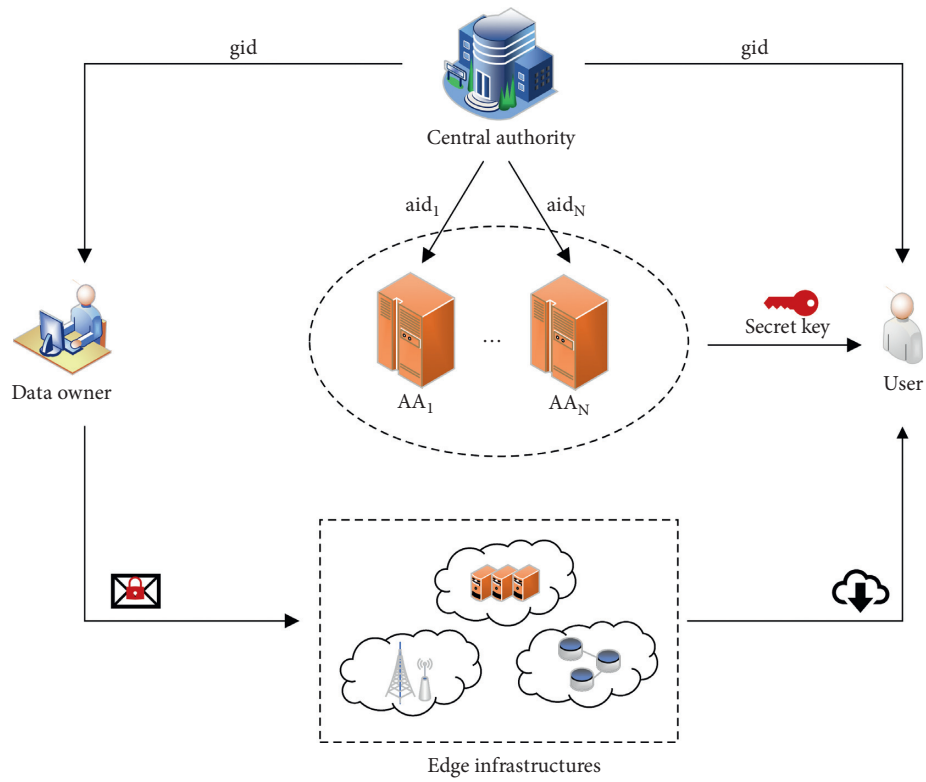

Figure 2: The framework of system model.

keys from all AAs to generate the system public key. Finally, when a user requests for his/her user's secret key, each AA only distributes its corresponding share of user secret key. Namely, there is no need for an AA to communicate with any other AA during the period of encryption and key generation.

(3) A data owner (DO) encrypts the data. He/she specifies the access policy over criteria, the weight of each criterion, and the cumulative weight threshold that a user needs to satisfy. Concretely, DO runs the encryption algorithm and generates a ciphertext associated with all these requirements above and then uploads the ciphertext to edge infrastructure.

(4) The user obtains a global identity gid issued by CA and AAs. Besides, any user in the system can download the encrypted data but can get access to the plaintext only when he/she satisfies both the access policy and weight requirement that the data owner specifies.

(5) Each edge infrastructure is an entity that provides storage and computing services for DO. It accepts encrypted data sent by DO. Then, the data can be obtained by any registered user in the system.

For precision and unambiguity, some default definitions and requirements in our proposed schemes are provided here. In the system model, we suppose that CA is unconditionally credible and cannot be compromised. On the other hand, a user can download whichever encrypted data he wants but can recover the corresponding plaintext if and only if he/she satisfies both the access policy and the cumulative weight threshold. Moreover, since the weights reflect the difference in importance among criteria when formulating an access policy, the ideal situation is that the user criteria that satisfy the policy contain more relatively significant (higher weight) criteria rather than a simple patchwork of low-weight criteria. Therefore, we consider that data owners are all sufficiently rigorous to design access policies, endue weight on each criterion, and set the thresholds over criteria. Furthermore, there are at least two authorities in the system.

3.2. Syntax of Scheme. The syntax of the multi-authority criteria-based encryption scheme consists of the following PPT algorithms:

(1) GlobalSetup$(1^\lambda) \longrightarrow pp$: the algorithm is performed by CA. It takes as input security parameter λ. It consists of three steps. CA first performs the group generation algorithm $\mathcal{G}(1^\lambda)$ to obtain $\mathrm{GP} = (\mathbb{G}, \mathbb{G}_T, e, g, p)$ and defines criterion universe U with size n. Then, it chooses $\varphi_i \in_R \mathbb{Z}_p$ to label each polynomial P_i. Eventually, CA receives registration requests from users and AAs and records the number of AAs as n_θ. It outputs public parameter $pp = (\mathrm{GP}, n, t, \{\varphi_i\}_{i\in[1,n]}, n_\theta)$.

(2) AASetup$(pp) \longrightarrow (pk_\theta, sk_\theta)$: the algorithm is performed by CA. For each authority AA_θ, it first chooses α_θ at random, such that $\alpha = \sum_{\theta=1}^{n_\theta} \alpha_\theta$. Note that the value of α is secret to any AA_θ. Then, all the

authorities run (t, n)-threshold secret sharing according to α_θ. Each authority AA_θ outputs pk_θ and keeps sk_θ as secret.

(3) CASetup$(pp, \{pk_\theta\}_{\theta\in[1,n_\theta]}, d) \longrightarrow$ (PK, MSK): the algorithm is performed by CA and AA. It receives public parameter pp, public keys $\{pk_\theta\}_{\theta\in[1,n_\theta]}$ from all the AAs, and degree d of polynomials. It outputs public key PK and implicitly keeps values (α, a) for secret.

(4) Encrypt$(\text{PK}, m, (\mathbb{A}, \rho), \vec{w}, \tau) \longrightarrow$ CT: the algorithm is performed by DO. It takes in public parameter pp, the public key PK, a message m, an access structure $(\mathbb{A}, \rho)$, a weight vector $\vec{w}$, and weight threshold τ. It outputs a ciphertext CT.

(5) KeyGen$(pp, \text{PK}, \text{gid}, C_{\text{gid}}) \longrightarrow \text{SK}_{\text{gid}}$: the algorithm is performed by the user with identity gid. It takes in pp, public key PK, the global identity gid of a user, and the set of cases C_{gid} belonging to the user. It outputs a SK_{gid}.

(6) Decrypt$(pp, \text{SK}_{\text{gid}}, \text{CT}) \longrightarrow (m/\perp)$: it takes in the public parameter pp, the secret key SK_{gid}, and the ciphertext CT. The algorithm outputs either a message m or the distinctive symbol $\perp$.

For the correctness of our schemes, we require that for the CT$\leftarrow$Encrypt$(\text{PK}, m, (\mathbb{A}, \rho), \vec{w}, \tau)$ and the $\text{SK}_{\text{gid}}\leftarrow$KeyGen$(pp, \text{PK}, \text{gid}, C_{\text{gid}})$, one can execute Decrypt$(pp, \text{SK}_{\text{gid}}, \text{CT})$ algorithm to obtain the correct message m with overwhelming probability.

3.3. Security Model. Here, the IND-CPA security [16] for proposed scheme is defined in the following game which has a challenger $\mathscr{C}$ and an adversary $\mathscr{A}$.

Init. $\mathscr{C}$ performs the algorithm GlobalSetup, AA Setup, and CA Setup and then sends the pp and PK to $\mathscr{A}$.

Phase 1. $\mathscr{A}$ repeatedly performs private key associated with sets of case $\mathbb{C}$.

Challenge. $\mathscr{A}$ specifies two messages $m_0, m_1 \in \mathbb{G}_T$, a challenge access structure $\mathbb{A}^*$, a vector $\vec{w}^*$, and a weight threshold τ^* to $\mathscr{C}$. The default condition is that $\mathbb{C}$ cannot satisfy the access structure $\mathbb{A}^*$. Then, $\mathscr{C}$ randomly picks an element $b \in \{0, 1\}$ and executes Encrypt algorithm to generate $m_b \in \mathbb{G}_T$ under $\mathbb{A}^*$. Finally, $\mathscr{A}$ obtains the ciphertext CT^* from $\mathscr{C}$.

Phase 2. $\mathscr{A}$ can repeatedly make the same queries as Phase 1, except that $\mathbb{C}$ cannot satisfy $\mathbb{A}^*$.

Guess. The adversary outputs a guess b' of b.

The advantage of the adversary $\mathscr{A}$ in this game is defined as $\Pr[b' = b] - 1/2$.

Definition 3. The proposed multi-authority criteria-based encryption scheme is secure if all polynomial-time adversaries have at most a negligible advantage in the above game.

4. Construction

In this section, we first provide an overview of the proposed schemes and then give the detailed constructions of the two schemes.

4.1. Overview. What we first consider is how to find a form to express the criteria. In our schemes, the criterion is related to a polynomial, and each root of the polynomial corresponds to a case that satisfies the criteria. The first scheme requires that the user satisfies at least one case of the criterion, while in the second, there is a stricter restriction that the user must satisfy all cases of the criteria. In this context, our scheme improves the flexibility of access policy in practical application. Specifically, recall the access policy described in Figure 1. DO specifies an access policy $\mathbb{A} = (P_1)\text{AND}\ (P_2)\text{OR}\ (P_3)\text{OR}\ (P_4)$, and the cumulative weight threshold is set to $\tau = 6$. The observation is that the criterion set with cumulative weight exceeding τ can be expressed as $T = \{(4), (1, 2), (1, 3), (1, 4), (2, 4), (3, 4), (1, 2, 3), (1, 2, 4), (1, 3, 4), (2, 3, 4), (1, 2, 3, 4)\}$. Clearly, Alice is a chief physician who has served the community for seven years. The case set and criterion set can be described as $C_{\text{Alice}} = \{1, 2\}; S_{\text{Alice}} = \{(1), (2), (1, 2)\}$. She can successfully decrypt the data due to the fact that set $C_{\text{Alice}} \vDash \mathbb{A}$ (i.e., $S_{C_{\text{Alice}}} = \{(1, 2)\}$ and $W_{\text{Alice}} = T \cap S_{C_{\text{Alice}}} = \{(1, 2)\}$. Bob is a community manager hired by the government and has 6 years of work experience related to medical treatment. He cannot decrypt the message successfully, since $W_{\text{Bob}} = T \cap S_{C_{\text{Bob}}} = \varnothing$.

From the practical perspective, the first scheme is suitable for edge computing platforms, while the second is suitable for users' private edge devices because those devices are more vulnerable to attacks by adversaries. Moreover, we introduce the multi-authority mechanism to solve the security problem caused by all attributes being managed by one authority. In this work, the criterion universe is jointly managed by n_θ AAs. The restriction is that there is no collusion between AAs. Specifically, CA cannot interact with users except for generating global unique identities for them. The user can reconstruct the secret key, which has the term of $e(g, g)^\alpha$, after interacting with t different AAs. This way, we make it impossible for each AA to generate a valid key individually. Meanwhile, data owners can assign a reasonable weight for each criterion and the cumulative weight according to their requirements, which makes the scheme suitable for real application scenarios.

4.2. MA-CE-Verify Root Scheme. Here, we provide our first multi-authority criteria-based encryption scheme that requires the user to have at least one root of a polynomial (or criterion).

(1) GlobalSetup$(1^\lambda) \longrightarrow pp$: CA first runs $\mathscr{G}(1^\lambda)$ to obtain $\text{GP} = (\mathbb{G}, \mathbb{G}_T, e, g, p)$, where g is a generator of $\mathbb{G}$ and $\mathbb{G}$ and $\mathbb{G}_T$ are two multiplicative cyclic groups with the same order p, such that $\mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$. Then, CA defines the criterion universe U with size n and chooses $\varphi_i \in_R \mathbb{Z}_p$ to label each

polynomial P_i. Moreover, CA receives registration requests from AAs and users, records the number of AAs as n_θ, and generates the global unique identity aid, gid $\in \mathbb{Z}_p$ for AA and user, respectively. At last, CA defines threshold t according to the value n_θ. It outputs public parameter $pp = (\mathrm{GP}, n, t, \{\varphi_i\}_{i\in[1,n]}, n_\theta)$.

(2) $\mathrm{AA\,Setup}(pp) \longrightarrow (pk_\theta, sk_\theta)$: firstly, each authority $\mathrm{AA}_\theta(\theta \in [1, n_\theta])$ chooses the secret $\alpha_\theta \in_R \mathbb{Z}_p$, such that master secret $\alpha = \sum_{\theta=1}^{n_\theta} \alpha_\theta$. Then, AA_θ randomly sets a polynomial $q_\theta(x)$ of degree $t-1$ which satisfies $\alpha_\theta = q_\theta(0)$. Other $\mathrm{AA}_\xi(\xi = 1, 2, \ldots, \theta-1, \theta+1, \ldots, n_\theta)$ obtains the value $s_{\theta\xi} = q_\theta(\mathrm{aid}_\xi)$ calculated by AA_θ. Meanwhile, AA_θ calculates $s_{\theta\theta} = q_\theta(\mathrm{aid}_\theta)$ for itself. Finally, AA_θ calculates its secret key $sk_\theta = \sum_{\xi=1}^{n_\theta} s_{\xi\theta}$ and public key $pk_\theta = e(g,g)^{sk_\theta}$.

(3) $\mathrm{CA\,Setup}(pp, \{pk_\theta\}_{\theta\in[1,n_\theta]}, d) \longrightarrow (\mathrm{PK}, \mathrm{MSK})$: CA randomly picks t of n_θ AAs' public keys. Additionally, it picks $h_1, \ldots, h_n \in_R \mathbb{G}$ and calculates as

$$\begin{aligned} e(g,g)^\alpha &= \prod_{\theta=1}^{t} pk_\theta^{\prod_{\xi=1,\xi\neq\theta}^{t} \mathrm{aid}_\xi/(\mathrm{aid}_\xi - \mathrm{aid}_\theta)} \\ &= e(g,g) \sum_{\theta=1}^{t} sk_\theta \prod_{\xi=1,\xi\neq\theta}^{t} \mathrm{aid}_\xi/(\mathrm{aid}_\xi - \mathrm{aid}_\theta). \end{aligned} \tag{6}$$

Then, CA selects $a \in_R \mathbb{Z}_p$ and calculates g^a. For criterion universe U, it picks a set of d-degree polynomials $\{P_i\}_{i\in[1,n]}$ with coefficients $(a_{i,d}, a_{i,d-1}, \ldots, a_{i,0})$ and labels P_i with φ_i. In this case, the set of polynomials can be described as

$$\begin{cases} \overrightarrow{P}_1 = \left(a_{1,d}, a_{1,d-1}, \ldots, a_{1,0}, \dfrac{1}{\varphi_1}\right), \\ \overrightarrow{P}_1' = \left(a_{1,d}, a_{1,d-1}, \ldots, a_{1,0}, 1\right), \\ \cdots, \\ \overrightarrow{P}_n = \left(a_{n,d}, a_{n,d-1}, \ldots, a_{n,0}, \dfrac{1}{\varphi_n}\right), \\ \overrightarrow{P}_n' = \left(a_{n,d}, a_{n,d-1}, \ldots, a_{n,0}, 1\right). \end{cases} \tag{7}$$

For $i = 1$ to n, CA computes $g^{\varphi_i \overrightarrow{P}_i}$ and $g^{\overrightarrow{P'}_i}$. It outputs

$$\mathrm{PK} = \left(g, g^a, e(g,g)^\alpha, \left\{g^{\varphi_i \overrightarrow{P}_i}, g^{\overrightarrow{P'}_i}, h_i\right\}_{i\in[1,n]}\right), \tag{8}$$

and keeps the values (α, a) for secret.

(4) $\mathrm{Encrypt}(\mathrm{PK}, m, (\mathbb{A}, \rho), \overrightarrow{w}, \tau) \longrightarrow \mathrm{CT}$: in this phase, the encryption algorithm sets the access policy $(\mathbb{A}, \rho)$, where the size of the matrix $\mathbb{A}$ is $\ell \times n$, and the function ρ maps $\mathbb{A}_i$ to a criterion. Then, it specifies the weight vector $\overrightarrow{w} = (w_1, \ldots, w_n)$, where the element w_i represents the weight of each criterion. Also, it takes $y_2, y_3, \ldots, y_n \in_R \mathbb{Z}_p$ to construct vector $\overrightarrow{v} = (s, y_2, y_3, \ldots, y_n) \in \mathbb{Z}_p^n$, where the first element $s \in_R \mathbb{Z}_p$ is the secret value to be shared. For $i = 1$ to ℓ, it computes $\lambda_i = \overrightarrow{v} \cdot M_i$. After completing the above processes, it computes the set $T = \left\{(k_1^i, k_2^i, \ldots, k_{\mu_i}^i)\right\}$ according to weight threshold τ, where μ_i indicates the length of i-th subset and $k_j^i \in \{1, 2, \ldots, n\}$ denotes index in U. Finally, the algorithm calculates

$$\begin{aligned} &C_0 = m \cdot e(g,g)^{\alpha s}, \\ &C_0' = g^s, \\ &\cdot \left\{C_i = g^{a\lambda_i} \cdot g^{-\varphi_{\rho(i)} \overrightarrow{P}_{\rho(i)} s}, C_i' = g^{a\lambda_i} \cdot g^{-\overrightarrow{P'}_{\rho(i)} s}\right\}_{i\in[1,\ell]}, \\ &\cdot \left\{\widehat{C}_i = \prod_{j=1}^{\mu_i} h_{k_j^i}^s\right\}_{i\in[1,\mathrm{len}(T)]}. \end{aligned} \tag{9}$$

It outputs ciphertext as $\mathrm{CT} = (C_0, C_0', \{C_i, C_i'\}_{i\in[1,\ell]}, \{\widehat{C}_i\}_{i\in[1,\mathrm{len}(T)]}, T)$.

(5) $\mathrm{KeyGen}(pp, \mathrm{PK}, \mathrm{gid}, C_{\mathrm{gid}}) \longrightarrow \mathrm{SK}_{\mathrm{gid}}$: the key generation algorithm is implemented by the user interacting with t AAs according to the requirements. The restriction is that AA_θ cannot communicate with each other.

Let z_{φ_x} be a root of the polynomial at x. For each root z_{φ_x} that belongs to user, AA creates the vector $\overrightarrow{z}_{\varphi_x} = (z_{\varphi_x}^d, z_{\varphi_x}^{d-1}, \ldots, z_{\varphi_x}, 1)$. We use $C_{\mathrm{gid}} \subseteq \{C_x\}_{x\in[1,n]}$ to denote a set of cases, which belong to the user with gid. Let $P = \left\{P_1, P_2, \ldots, P_{\mathrm{len}(C_{\mathrm{gid}})}\right\}$ denote the set of criteria requested by the user and $S = \{(P_1), (P_2), \ldots (P_{\mathrm{len}(C_{\mathrm{gid}})}), (P_1, P_2), \ldots, (P_1, P_2, \ldots, P_{\mathrm{len}(C_{\mathrm{gid}})})\} = \{k_1^l, \ldots, k_{v_l}^l\}$ be all combinations of entities in set P, where v_l denotes the length of subset and $k_j^l \in [1, n]$ denotes index in U. AA_θ picks $\delta_\theta \in_R \mathbb{Z}_p$ and calculates as

$$\begin{aligned} &L_\theta = g^{\delta_\theta}, \\ &\forall C_{\theta,x} \in C_{\mathrm{gid}} K_{\theta,x} = \left(g^{\varphi_x}\right)^{\overrightarrow{Z}_{\varphi_x} \cdot \delta_\theta}, \\ &K_{\theta,x}' = g^{\overrightarrow{P}_x' \delta_\theta}, \\ &\left\{\widehat{K}_{\theta,i} = g^{sk_\theta} g^{a\delta_\theta} \prod_{j=1}^{v_l} h_{k_j^l}^{\delta_\theta}\right\}_{l\in[1,\mathrm{len}(S)]}. \end{aligned} \tag{10}$$

After interacting with t AAs, the user constructs the secret key as

$$L = \prod_{\theta=1}^{t} L_\theta^{\prod_{\xi=1,\xi\neq\theta}^{t} \mathrm{aid}_\xi/\mathrm{aid}_\xi-\mathrm{aid}_\theta} = g^{\sum_{\theta=1}^{t}\delta_\theta\cdot\prod_{\xi=1,\xi\neq\theta}^{t}\mathrm{aid}_\xi/\mathrm{aid}_\xi-\mathrm{aid}_\theta}. \tag{11}$$

For all $C_{\theta,x} \in C_{\mathrm{gid}}$, we have

$$\begin{aligned}
K_x &= \prod_{\theta=1}^{t}\left(\left(K_{\theta,x}\right)^{\overrightarrow{Z}_{\varphi_x}\cdot\delta_\theta}\right)^{\prod_{\xi=1,\xi\neq\theta}^{t}\mathrm{aid}_\xi/\mathrm{aid}_\xi-\mathrm{aid}_\theta} = g^{\varphi_x}\overrightarrow{Z}_{\varphi_x}{}^{\sum_{\theta=1}^{t}\delta_\theta\prod_{\xi=1,\xi\neq\theta}^{t}\mathrm{aid}_\xi/\mathrm{aid}_\xi-\mathrm{aid}_\theta},\\
K_x' &= \prod_{\theta=1}^{t}\left(g^{\overrightarrow{P}_x'\delta_\theta}\right)^{\prod_{\xi=1,\xi\neq\theta}^{t}\mathrm{aid}_\xi/\mathrm{aid}_\xi-\mathrm{aid}_\theta} = g^{\overrightarrow{P}_x'\sum_{\theta=1}^{t}\delta_\theta\prod_{\xi=1,\xi\neq\theta}^{t}\mathrm{aid}_\xi/\mathrm{aid}_\xi-\mathrm{aid}_\theta},\\
\widehat{K}_l &= \prod_{\theta=1}^{t}\left(\widehat{K}_{\theta,l}\right)^{\prod_{\xi=1,\xi\neq\theta}^{t}\mathrm{aid}_\xi/\mathrm{aid}_\xi-\mathrm{aid}_\theta} = \prod_{\theta=1}^{t}\left(g^{sk_\theta}g^{a\delta_\theta}\prod_{j=1}^{v_l}h_{k_j'}^{\delta_\theta}\right)^{\prod_{\xi=1,\xi\neq\theta}^{t}\mathrm{aid}_\xi/\mathrm{aid}_\xi-\mathrm{aid}_\theta}.
\end{aligned} \tag{12}$$

For simplicity, we make $u = \sum_{\theta=1}^{t}\delta_\theta\cdot\prod_{\xi=1,\xi\neq\theta}^{t}\mathrm{aid}_\xi/\mathrm{aid}_\xi-\mathrm{aid}_\theta$. For this, the secret key of the user can be represented as

$$S, L = g^u, \forall C_x \in C_{\mathrm{gid}} K_x = g^{\varphi_x\cdot}\overrightarrow{Z}_{\varphi_x u}, K_x' = g^{\overrightarrow{P}_x' u},\cdot\left\{\widehat{K}_l = g^\alpha\cdot g^{au}\cdot\prod_{j=1}^{v_l}h_{k_j'}^{u}\right\}_{l\in[1,\mathrm{len}(S)]}. \tag{13}$$

(6) $\mathrm{Decrypt}(pp, \mathrm{SK}_{\mathrm{gid}}, \mathrm{CT}) \longrightarrow (m/\perp)$: the decryption algorithm can successfully be invoked by the user with a valid identity. Namely, the user can download encrypted data from the edge infrastructures, and they can decrypt data successfully if their case set C_{gid} satisfies access policy and the requirement of cumulative weight.

Suppose that the ciphertext CT is encrypted under the access policy $(\mathbb{A}, \rho)$. We recall the definition of LSSS. Let $I \subset \{1, 2, \ldots, \ell\}$ represent a case such that $\rho(i) \in C_{\mathrm{gid}}$. To decrypt the ciphertext, the user with $\mathrm{SK}_{\mathrm{gid}}$ computes $\{\omega_i \in \mathbb{Z}_p\}_{i\in I}$; if λ_i is valid share corresponding access policy $(\mathbb{A}, \rho)$, then the secret $s = \sum_{i\in I}\omega_i\lambda_i$ can be calculated. To summarize, the decryption process is as follows:

$$\begin{aligned}
&\prod_{i\in I}\left(e(C_i', L)\cdot\frac{e\left(C_i, K_{\rho(i)}\right)\cdot e\left(C_0', K_{\rho(i)}{}'\right)}{e\left(C_i', K_{\rho(i)}\right)}\right)^{\omega_i}\\
&= \prod_{i\in I}\left(e\left(g^{a\lambda_i}\cdot g^{-\overrightarrow{P}_{\rho(i)}'s}, g^u\right)\cdot\frac{e\left(g^{a\lambda_i}\cdot g^{-\varphi_{\rho(i)}}\overrightarrow{P}_{\rho(i)}s, g^{\varphi_{\rho(i)}\cdot}\overrightarrow{Z}_{\rho(i)}u\right)\cdot e\left(g^s, g^{\overrightarrow{P}_{\rho(i)}'u}\right)}{e\left(g^{a\lambda_i}\cdot g^{-\overrightarrow{P}_{\rho(i)}'s}, g^{\varphi_{\rho(i)}\cdot}\overrightarrow{Z}_{\varphi_{\rho(i)}u}\right)}\right)^{\omega_i}\\
&= \prod_{i\in I}\left(e\left(g^{a\lambda_i}, g^u\right)\cdot\frac{\left(g^{-\varphi_{\rho(i)}}\overrightarrow{P}_{\rho(i)}s, g^{\varphi_{\rho(i)}\cdot}\overrightarrow{Z}_{\rho(i)}u\right)}{e\left(g^{-\overrightarrow{P}_{\rho(i)}'\cdot s}, g^{\varphi_{\rho(i)}\cdot}\overrightarrow{Z}_{\varphi_{\rho(i)}}u\right)}\right)^{\omega_i}\\
&= e(g, g)^{asu}.
\end{aligned} \tag{14}$$

Define set $W = \{T \cap S_{C \vDash \mathbb{A}}\}$. For each $w \in W$, let w_T and w_S denote the index w in set T and S, respectively. Then, compute

$$\begin{aligned} J &= \prod_{w \in W} \left(\frac{e\left(C_0', \widehat{K}_{w_S}\right)}{e\left(\widehat{C}_{w_T}, L\right)} \right)^{1/W} \\ &= \prod_{w \in W} \left(\frac{e\left(g^s, g^\alpha \cdot g^{au} \prod_{j=1}^{\mathrm{len}(w)} h_{w_j}^u\right)}{e\left(\prod_{j=1}^{\mathrm{len}(w)} h_{w_j}^s, g^u\right)} \right)^{1/W} \\ &= \prod_{w \in W} e\left(g^s, g^\alpha \cdot g^{au}\right)^{1/W} \\ &= e(g,g)^{\alpha s} \cdot e(g,g)^{asu}. \end{aligned} \tag{15}$$

The user can recover the plaintext m from the following equation:

$$m = C_0 \cdot \frac{e(g,g)^{asu}}{J}. \tag{16}$$

4.3. MA-CE-Root Equality Scheme. Here, we provide our second scheme, which needs all the roots (or cases) of each polynomial (or criterion) to be held by the user.

(1) Global Setup $(1^\lambda) \longrightarrow pp$: this algorithm is similar to scheme MA-CE-Verify Root. CA runs $\mathcal{G}(1^\lambda)$ to obtain GP $= (\mathbb{G}, \mathbb{G}_T, e, g, p)$ and defines the criterion universe U with size n. CA also generates unique identity for AAs and users, respectively. Then, it chooses a threshold t and picks $\{\varphi_i\}_{i \in [1,n]} \in_R \mathbb{Z}_p$. Note that φ_i is not used to label the polynomial P_i. It outputs public parameter $pp = (\mathrm{GP}, n, t, \{\varphi_i\}_{i \in [1,n]}, n_\theta)$.

(2) AA Setup $(pp) \longrightarrow (pk_\theta, sk_\theta)$: the algorithm is similar to the AA Setup in the first scheme. For each authority AA_θ, it inputs the public parameter pp and returns a pair of keys (pk_θ, sk_θ), where sk_θ is kept secret for other AAs.

(3) CA Setup $(pp, \{pk_\theta\}_{\theta \in [1,n_\theta]}, d) \longrightarrow (\mathrm{PK}, \mathrm{MSK})$: CA randomly chooses t public keys from n_θ AAs. In addition, it picks $h_1, \ldots, h_n \in_R \mathbb{G}$ and calculates

$$e(g,g)^\alpha = \prod_{\theta=1}^{t} pk_\theta \prod_{\xi=1, \xi \neq \theta}^{t} \mathrm{aid}_\xi / \mathrm{aid}_\xi - \mathrm{aid}_\theta. \tag{17}$$

Then, CA randomly picks $a \in \mathbb{Z}_p$ and computes g^a. For $i = 1$ to n, it picks a set of d-degree polynomials $\{P_i\}_{i \in [1,n]}$, which can be described as

$$\begin{cases} \overrightarrow{P}_1 = \left(1, \dfrac{a_{1,d-1}}{a_{1,d}}, \ldots, \dfrac{a_{1,0}}{a_{1,d}}\right), \\ \overrightarrow{P}_2 = \left(1, \dfrac{a_{2,d-1}}{a_{2,d}}, \ldots, \dfrac{a_{2,0}}{a_{2,d}}\right), \\ \overrightarrow{P}_n = \left(1, \dfrac{a_{n,d-1}}{a_{n,d}}, \ldots, \dfrac{a_{n,0}}{a_{n,d}}\right). \end{cases} \tag{18}$$

It outputs $\mathrm{PK} = \left(g, g^a, e(g,g)^\alpha, \left\{g^{\varphi_i \overrightarrow{P}_i}, h_i\right\}_{i \in [1,n]}\right)$ and keeps the values (α, a) for secret.

(4) Encrypt $(\mathrm{PK}, m, (\mathbb{A}, \rho), \overrightarrow{w}, \tau) \longrightarrow \mathrm{CT}$: the encryption algorithm sets the access policy $(\mathbb{A}, \rho)$, the size of the matrix is $\ell \times n$, and the function ρ maps $\mathbb{A}_i$ to a criterion. Then, it specifies the weight vector $\overrightarrow{w} = (w_1, \ldots, w_n)$, where the element w_i represents the weight of each criterion. Moreover, it constructs the vector $\overrightarrow{v} = (s, r_2, r_3, \ldots, r_n) \in \mathbb{Z}_p^n$. For $i = 1$ to ℓ, it computes $\lambda_i = \overrightarrow{v} \cdot M_i$ and set $T = \left\{(k_1^i, k_2^i, \ldots, k_{\mu_i}^i)\right\}$ according to the weight threshold τ, where μ_i indicates the length of i-th subset and $k_j^i \in \{1, 2, \ldots, n\}$ denotes index in U. Finally, the algorithm computes as

$$\begin{aligned} C_0 &= m \cdot e(g,g)^{\alpha s}, \\ C_0' &= g^s \\ &\quad \cdot \left\{ C_i = g^{a\lambda_i} \cdot g^{-\varphi_{\rho(i)} \overrightarrow{P}_{\rho(i)} s} \right\}_{i \in [1,\ell]}, \\ &\quad \cdot \left\{ \widehat{C}_i = \prod_{j=1}^{\mu_i} h_{k_j^i}^s \right\}_{i \in [1, \mathrm{len}[T]]}. \end{aligned} \tag{19}$$

It outputs ciphertext as $\mathrm{CT} = (C_0, C_0', \{C_i\}_{i \in [1,\ell]}, \{\widehat{C}_i\}_{i \in [1, \mathrm{len}(T)]}, T)$.

(5) Key Gen $(pp, \mathrm{PK}, \mathrm{gid}, C_{\mathrm{gid}}) \longrightarrow \mathrm{SK}_{\mathrm{gid}}$: the user with gid interacts with any t AAs to obtain the key according to requirements. It takes the set $\mathrm{Roots}_x = \{x_1, x_2, \ldots, x_d\}$ to represent all the roots of the polynomial at x. According to Vieta's theorem, AA uses Roots_x to construct the following vector:

$$\overrightarrow{z}_{\varphi_x} = \left(1, z_{x_{d-1}}, z_{x_{d-2}}, \ldots, z_{x_0}\right). \tag{20}$$

Let $C_{\mathrm{gid}} \subseteq \{C_x\}_{x \in [1,n]}$ represent the cases belonging to the user with identity gid, set $P = \{P_1, P_2, \ldots, P_{\mathrm{len}(C_{\mathrm{gid}})}\}$ denote the set of criteria requested by the user, and set $S = \left\{(P_1), (P_2), \ldots, (P_{\mathrm{len}(C_{\mathrm{gid}})}), (P_1, P_2), \ldots, (P_1, P_2, \ldots, P_{\mathrm{len}(C_{\mathrm{gid}})})\right\} = \left\{k_1^l, \ldots, k_{v_l}^l\right\}$ be all combinations of entities in set P, where v_l denotes the

length of l-th subset and $k_j^l \in [1,n]$ denotes index in U. AA_θ picks $\delta_\theta \in_R \mathbb{Z}_p$ and calculates as

$$L_\theta = g^{\delta_\theta},$$
$$\forall C_{\theta,x} \in C_{\mathrm{gid}}\, K_{\theta,x} = \left(g^{\varphi_x}\right)^{\overrightarrow{Z}_{\varphi_x}\cdot\delta_\theta} \cdot \left\{\widehat{K}_{\theta,l} = g^{sk_\theta} g^{a\delta_\theta} \prod_{j=1}^{v_l} h_{k_j^l}^{\delta_\theta}\right\}_{l\in[1,\mathrm{len}(S)]}. \tag{21}$$

After interacting with t AAs, the user constructs the secret key $\mathrm{SK}_{\mathrm{gid}}$ as follows:

$$L = g^u,$$
$$\forall C_x \in C_{\mathrm{gid}} K_x = \left(g^{\varphi_x}\right)^{\overrightarrow{Z}_{\varphi_x}\cdot u} \cdot \left\{\widehat{K}_l = g^\alpha g^{au} \prod_{j=1}^{v_l} h_{k_j^l}^u\right\}_{l\in[1,\mathrm{len}(S)]}. \tag{22}$$

(6) $\mathrm{Decrypt}(pp, \mathrm{SK}_{\mathrm{gid}}, \mathrm{CT}) \longrightarrow (m/\perp)$: to recover the encrypted data under access policy $(\mathbb{A},\rho)$, the user first calculates constants $\{\omega_i \in \mathbb{Z}_p\}_{i\in I}$ and then computes

$$\begin{aligned}
&\prod_{i\in I}\left(e\left(C_i, L\right)\cdot e\left(C_0', K_{\rho(i)}\right)\right)^{\omega_i} \\
&= \prod_{i\in I}\left(e\left(g^{a\lambda_i}\cdot g^{-\varphi_{\rho(i)}\overrightarrow{P}_{\rho(i)}s}, g^u\right)\cdot e\left(g^s, \left(g^{\varphi_x}\right)^{\overrightarrow{Z}_{\varphi_x}\cdot u}\right)\right)^{\omega_i} \\
&= \prod_{i\in I} e\left(g^{a\lambda_i}, g^u\right)^{\omega_i} \\
&= e(g,g)^{asu}.
\end{aligned} \tag{23}$$

For $w \in W = \{T \cap S_{C\models\mathbb{A}}\}$, the symbols w_T and w_S denote the index w in set T and S, respectively; then, compute

$$\begin{aligned}
J &= \prod_{w\in W}\left(\frac{e\left(C_0', \widehat{K}_{w_S}\right)}{e\left(\widehat{C}_{w_T}, L\right)}\right)^{1/|W|} \\
&= e\left(g^s, g^\alpha\cdot g^{au}\right)^{|W|/|W|} \\
&= e(g,g)^{\alpha s}\cdot e(g,g)^{asu}.
\end{aligned} \tag{24}$$

The user can recover the plaintext $m = C_0 \cdot e(g,g)^{asu}/J$.

5. Security Proof

To prove the security of our constructions, the theorem in [8] is introduced as shown below.

Theorem 1. *If the decisional q-BDHE assumption holds, then any polynomial-time adversary cannot selectively break the MA-CE-Verify Root scheme with a challenge matrix of size $\ell^* \times n^*$, where $n^* \le q$.*

Here, we briefly overview the proof technique under the decisional q-BDHE assumption. Suppose that there exists an adversary A with a nonnegligible advantage ε can selectively break the proposed scheme. $\mathscr{A}$ is allowed to select a matrix with the size of at most $q \times q$. Here, the restriction is that the key queried from the challenger cannot decrypt the message. Then, we construct a PPT simulator B, which solves the q-BDHE assumption.

Init. $\mathscr{B}$ first receipts the q-BDHE challenge $\overrightarrow{y} = (g, g^s, g_1, \ldots, g_q, g_{q+2}, \ldots, g_{2q})$ and $\widetilde{T}$. Then, $\mathscr{A}$ sends the challenge structure $(\mathbb{A}^*, \rho^*)$, a weight vector $\overrightarrow{w}^* = \{w_1, w_2, \ldots, w_m\}_{m\in[1,n]}$, and a weight threshold τ^* to $\mathscr{B}$.

Setup. In this phase, $\mathscr{B}$ picks $\alpha\prime \in_R \mathbb{Z}_p$ and implicitly takes $\alpha = \alpha' + a^{q+1}$ by making $e(g,g)^\alpha = e(g^a, g^{a^q})\cdot e(g,g)^{\alpha'}$. Then, $\mathscr{B}$ randomly generates φ_x for polynomial P_x. The symbol Φ represents the collection of indexes i, such that $\rho^*(i) = P_x$. Next, $\mathscr{B}$ takes

$$\begin{aligned}
\psi_x &= g^{\varphi_x\overrightarrow{P}_x}\prod_{i\in\Phi} g^{a\mathbb{A}_{i,1}^*}\cdot g^{a^2\mathbb{A}_{i,2}^*}\cdots g^{a^{n^*}\cdot\mathbb{A}_{i,n^*}^*}, \\
\beta_x &= g^{\overrightarrow{P}'_x}\prod_{i\in\Phi} g^{a\mathbb{A}_{i,1}^*}\cdot g^{a^2\mathbb{A}_{i,2}^*}\cdots g^{a^{n^*}\cdot\mathbb{A}_{i,n^*}^*}, \\
\gamma_x &= g^{\varphi_x}\prod_{i\in\Phi} g^{a\mathbb{A}_{i,1}^*}\cdot g^{a^2\mathbb{A}_{i,2}^*}\cdots g^{a^{n^*}\cdot\mathbb{A}_{i,n^*}^*}.
\end{aligned} \tag{25}$$

We note that $\psi_x = g^{\varphi_x\overrightarrow{P}_x}$, $\beta_x = g^{\overrightarrow{P}'_x}$ and $\gamma_x = g^{\varphi_x}$ if $\Phi = \varnothing$. Finally, $\mathscr{B}$ chooses $\eta_1, \eta_2, \ldots, \eta_n \in_R \mathbb{Z}_p$ and computes

$$h_j = \begin{cases} g^{\eta_j/\sigma}, & \text{if } P_x \text{ has combination } \le \tau^*, \\ g^{\eta_j}, & \text{otherwise,} \end{cases} \tag{26}$$

which implicitly takes $\sigma = w_1 + w_2 + \cdots + w_m$.

Phase 1. $\mathscr{B}$ replies private key queries for $C = \{z_{\varphi_x}\}$, where C cannot satisfy $(\mathbb{A}^*, \rho^*)$. For each z_{φ_x}, $\mathscr{B}$ first creates vector $\overrightarrow{z}_{\varphi_x} = \{z_{\varphi_x}^d, z_{\varphi_x}^{d-1}, \ldots, z_{\varphi_x}, 1\}$ and chooses $r \in_R \mathbb{Z}_p$. Then, according to the definition of LSSS, $\mathscr{B}$ calculates a vector $\overrightarrow{\omega} = (\omega_1, \omega_2, \ldots, \omega_{n^*}) \in \mathbb{Z}_p^{n^*}$ such that $\omega_1 = -1$. For all i such that $\rho^*(i) \in S$, we have that the inner product $\langle\overrightarrow{\omega}, \mathbb{A}_i^*\rangle = 0$. Finally, $\mathscr{B}$ implicitly defines u as

$$u = r + \omega_1 a^q + \omega_2 a^{q-1} + \cdots + \omega_{n^*} a^{(q-n^*+1)}. \tag{27}$$

Therefore, the value L can be denoted as

$$L = g^r\prod_{i=1}^{n^*}\left(g^{a-i+1}\right)^{\omega_i} = g^u. \tag{28}$$

We now consider $z_{\varphi_x} \in S$ for the case that there is no i such that $\rho^*(i)$ has a root equal to z_{φ_x}. $\mathscr{B}$ can simply take $K_x = L^{\varphi_x\overrightarrow{z}_{\varphi_x}}$. Otherwise, it calculates as

$$K_x = \left(g^{\varphi_x} \prod_{i \in \Phi} g^{a\mathbb{A}^*_{i,1}} \cdot g^{a^2 \mathbb{A}^*_{i,2}} \cdots g^{a^{n^*} \mathbb{A}^*_{i,n^*}} \right)^{\overrightarrow{z}_{\varphi_x} u}. \tag{29}$$

Note that by defining u, K_x has the form of $\mathbb{A}^*_{i,j} a^j w_j a^{q-j+1}$ in the exponent for some j. However, we have that $\langle \overrightarrow{w}, \mathbb{A}^*_i \rangle = 0$, and the term of $g^{a^{q+1}}$ can be cancelled. Consequently, K_x can be expressed as

$$\begin{aligned} K_x &= \left(g^{\varphi_x} \prod_{i \in \Phi} g^{a\mathbb{A}^*_{i,1}} \cdot g^{a^2 \mathbb{A}^*_{i,2}} \cdots g^{a^{n^*} \mathbb{A}^*_{i,n^*}} \right)^{\overrightarrow{z}_{\varphi_x} u} \\ &= L^{\varphi_x \overrightarrow{z}_{\varphi_x}} \prod_{i \in \Phi} \prod_{j=1}^{n^*} g^{a^j \cdot r} \cdot \left(\prod_{k=1, k \neq j}^{n^*} g^{a^{q+j-k+1}} \right)^{\mathbb{A}^*_{i,j}}. \end{aligned} \tag{30}$$

We now consider simulating the value of $\widehat{K}_l$. Let $P = \left\{ P_1, P_2, \ldots, P_{\mathrm{len}(C_{\mathrm{gid}})} \right\}$ be the set of criteria corresponding to the criterion universe U and $S = \left\{ k^l_1, \ldots, k^l_{v_l} \right\} (l \in [1, 2^{\mathrm{len}(C_{\mathrm{gid}})}])$ be all combinations of entities in set P. For $l = 1$ to $2^{\mathrm{len}(C_{\mathrm{gid}})}$, we have

$$\begin{aligned} \widehat{K}_l &= g^{\alpha} g^{au} \prod_{j=1}^{v_l} g^{\eta_{k_j} u} \\ &= g^{\alpha'} \cdot g^{au} \cdot g^{ar} \cdot g^{a^{q+1} w_1} \cdot g^{a^q w_2} \cdots g^{a^{q-n^*+2} w_{n^*}} \cdot L^{\sum_{j=1}^{v_l} \eta_{k_j}} \\ &= g^{\alpha'} \cdot g^{ar} \cdot g^{a^q w_2} \cdots g^{a^{q-n^*+2} w_{n^*}} \cdot L^{\sum_{j=1}^{v_l} \eta_{k_j}} \\ &= g^{\alpha'} \cdot g^{ar} \prod_{l=2}^{n^*} g^{a^{q-l+2} w_l} \cdot L^{\sum_{j=1}^{v_l} \eta_{k_j}}. \end{aligned} \tag{31}$$

Otherwise, we have

$$\begin{aligned} \widehat{K}_l &= g^{\alpha} g^{au} \prod_{j=1}^{v_l} g^{\eta_{k_j} t/\sigma} \\ &= g^{\alpha'} \cdot g^{a^{q+1}} \cdot g^{ar} \cdot g^{a^{q+1} w_1} \cdot g^{a^q w_2} \cdots g^{a^{q-n^*+2} w_{n^*}} L^{\sum_{j=1}^{v_l} \eta_{k_j}/\sigma} \\ &= g^{\alpha'} \cdot g^{ar} \cdot \prod_{l=2}^{n^*} \left(g^{a^{q-l+2}} \right)^{w_l} L^{\sum_{j=1}^{v_l} \eta_{k_j}/\sigma}. \end{aligned} \tag{32}$$

Challenge. We show how to build challenge ciphertext. $\mathscr{A}$ submits two messages m_0 and m_1 to $\mathscr{B}$. The simulator $\mathscr{B}$ selects $b \in \{0, 1\}$ at random and constructs $C_0 = m_b \cdot \widetilde{T} \cdot e(g^{\alpha'}, h)$, $C'_0 = h$. Then, it picks $y'_2, y'_3, \ldots, y'_{n^*} \in_R \mathbb{Z}_p$ and secret s using the vector

$$\overrightarrow{v} = \left(s, sa + y'_2, sa^2 + y'_3, \ldots, sa^{n-1} + y'_{n^*} \right) \in \mathbb{Z}^{n^*}_p. \tag{33}$$

Finally, $\mathscr{B}$ chooses threshold value τ^* and performs Encrypt algorithm to construct C_i, C'_i, and $\widehat{C}_i$ as follows:

$$\begin{aligned} C_i &= g^{a \overrightarrow{v} \mathbb{A}^*_i} \cdot g^{-\varphi_i \overrightarrow{P}_i s} \\ &= g^{sa\mathbb{A}^*_{i,1}} \cdot g^{\left(sa^2 + ay'_2\right)\mathbb{A}^*_{i,2}} \cdots g^{\left(sa^{n^*} + ay'_2\right)\mathbb{A}^*_{i,n^*}} \cdot \left(g^s\right)^{-\varphi_i \overrightarrow{P}_i} \cdot \left(g^{a\mathbb{A}^*_{i,1}} \cdot g^{a^2 \mathbb{A}^*_{i,2}} \cdots g^{a^{n^*}} \right)^{-s} \\ &= \left(g^s\right)^{-\varphi_i \overrightarrow{P}_i} \left(\prod_{j=1}^{n^*} \left(g^a\right)^{\mathbb{A}^*_{i,j} y'_j} \right), \\ C'_i &= g^{a \overrightarrow{v} \mathbb{A}^*_i} \cdot g^{-\overrightarrow{P'}_i s} \\ &= g^{sa\mathbb{A}^*_{i,1}} \cdot g^{\left(sa^2 + ay'_2\right)\mathbb{A}^*_{i,2}} \cdots g^{\left(sa^{n^*} + ay'_2\right)\mathbb{A}^*_{i,n^*}} \cdot \left(g^s\right)^{-\overrightarrow{P'}_i} \cdot \left(g^{a\mathbb{A}^*_{i,1}} \cdot g^{a^2 \mathbb{A}^*_{i,2}} \cdots g^{a^{n^*}} \right)^{-s} \\ &= \left(g^s\right)^{-\overrightarrow{P'}_i} \left(\prod_{j=1}^{n^*} \left(g^a\right)^{\mathbb{A}^*_{i,j} y'_j}, \left\{ \widehat{C}_i = \prod_{j=1}^{\mu_i} \mathscr{B}^s_{k^i_j} \right\}_{i \in [1, \mathrm{len}(T)]} \right). \end{aligned} \tag{34}$$

Phase 2. $\mathscr{A}$ can adaptively make queries the same as Phase 1 with the restriction that none of those cases satisfy the access structure corresponding to the Challenge phase.

Guess. The adversary $\mathscr{A}$ eventually outputs a guess bit $b' \in \{0, 1\}$ of b. If $\mathscr{A}$ correctly guesses $b' = b$, then $\mathscr{B}$ returns 0 to guess that $\widetilde{T} = e(g, g)^{a^{q+1} s}$; otherwise, it outputs 1 to

demonstrate that it considers $\widetilde{T}$ is a random element obtained from group $\mathbb{G}_T$. When $\widetilde{T}$ is a tuple, the simulator $\mathscr{B}$ performs a perfect simulation. In this case, we have that

$$\Pr\left[\mathscr{B}\left(\overrightarrow{y}, \widetilde{T} = e(g,g)^{a^{q+1}s}\right) = 0\right] = \frac{1}{2} + A\ dv_{\mathscr{A}}. \tag{35}$$

When $\widetilde{T}$ is a random element in $\mathbb{G}_T$, $\mathscr{B}$ simulates a completely random challenge ciphertext for adversary $\mathscr{A}$, and we have

$$\Pr[\mathscr{B}(\overrightarrow{y}, \widetilde{T} = R) = 0] = \frac{1}{2}. \tag{36}$$

Consequently, $\mathscr{B}$ can play the decisional q-BDHE game with non-negligible advantage.

Theorem 2. *If the decisional q-BDHE assumption holds, then no polynomial-time adversary can selectively break our MA-CE-Root Equality scheme with a challenge matrix of size $\ell^* \times n^*$, where $n^* \leq q$.*

The proof of this theorem is similar to Theorem 1 (here we omit the proof process).

6. Performance Analysis

We now provide theoretical analysis and implementation evaluation of the two schemes in this section.

6.1. Theoretical Analysis. There is the comparison of the four schemes, including [12–14] and our two schemes, in terms of storage overhead and computation cost. Let $\mathbb{P}$ indicate a pairing operation. $\mathbb{E}$ and $\mathbb{E}_T$ denote an exponential operation of group $\mathbb{G}$ and $\mathbb{G}_T$, respectively. $|g|$ and $|g_T|$ represent the size of elements in group $\mathbb{G}$ and $\mathbb{G}_T$, respectively. In our schemes, N represents the size of the criterion universe, while it represents attribute universe in [12–14]. n_ℓ and n_u denote the number of criteria (or attributes) in the access matrix and the number of criteria that are satisfied by the user, respectively. Let n_a denote the number of attributes managed by attribute authority. l is the number of all criterion sets with cumulative weight greater than τ. l_w is the size of the criterion set that satisfies the access policy and cumulative weight.

We first compare the storage overhead of the four schemes, as shown in Table 1. In terms of ciphertext size, our schemes are better than [12–14], since they require storing a large amount of leaf nodes information of the access tree. It can be observed that [13] is superior to our schemes in terms of key size and public key size. The reason is that the public key in our schemes needs to contain information corresponding to the criterion. All weights are specified by the trusted authority TA in [13]. Different from [13], the Key Gen phase of our schemes requires enumerating the criterion set that exceeds the weight. The performance of our schemes in terms of key size is comparable to that of [14]. However, the scheme in [14] cannot support multiple authorities, and the weight of each attribute is specified by TA. This inevitably limits the ability of the scheme in practical scenarios.

Table 2 shows the computation cost of these schemes in Key Gen, Encrypt, and Decrypt phases. In the Key Gen phase, the scheme in [13] performs better than other schemes, because the calculation of all the criterion sets takes up the main computation cost in our schemes and the scheme of [12]. In Encrypt and Decrypt phases, our schemes cost less time than [13] in practical application, since the computation cost of the latter is occupied by a large number of exponential and pairing operations. Moreover, it can be seen that the performance of [14] is similar to our first scheme and slightly inferior to the second scheme. The advantage of our schemes is that users can flexibly choose the weights in the access policy according to different application scenarios.

6.2. Implementation and Evaluation. We implement the proposed schemes in Charm [32] using Python 3.6.5. The programs adopt the Pairing-Based Cryptography (PBC) library version-0.5.14. We pick the symmetric curve with a 512 bit base field, and it provides 160 bit group order. All our programs were executed on VMware @ Workstation Pro 15.5.5 with a dual core Intel (R) Core (TM) i7-7700HQ CPU @2.8 GHz and 2.0 GB RAM running Ubuntu 18.04. All experimental results are taken from the average value of the program executed 20 times.

Figure 3 shows the value of key generation time with threshold t ($t = 1, 2, \ldots, 10$). We set the number of AAs to 10 in the system. As known from the figure, with the increase of threshold t, the time consumed for key generation is fixed basically, due to the fact that the user requests keys from t AAs in the meantime, while the time consumption of each AA for calculating subshare of a key is almost the same. Moreover, the value of t is generally within 10 in actual application scenarios. In summary, it can be considered that the time consumption is hardly affected by the threshold t in the KeyGen phase.

Figure 4 shows the time consumption of Key Gen, Encrypt, and Decrypt algorithms as the number of user attributes increases in the proposed schemes. We take the number n_θ of AAs as 10 and the threshold τ as 6. The performance of scheme-2 is slightly better than scheme-1 because the former has shorter ciphertext and key, which reduces exponential and pairing operations. We observe that the time consumption of each stage shows a nonlinear increasing trend. What mainly affects computational efficiency are summarized as follows. The first aspect is that the encryption algorithm needs to calculate all cases T that exceed the cumulative threshold τ. Another reason is that calculating the criteria set S that belong to the user dominates the execution time of the key generation algorithm (see Section 4 (Key Gen)). In addition, it takes a

Table 1: Comparison of storage overhead.

Schemes	PK size	Ciphertext size	Key size
LMX13 [12]	$(2N+2)\|g\|+\|g_T\|$	$(4n_\ell+4)\|g\|+\|g_T\|$	$7n_u\|g\|$
WZZ14 [13]	$(N+2)\|g\|+\|g_T\|$	$(n_\ell n_a+1)\|g\|+\|g_T\|$	$(n_u+1)\|g\|$
YYZ20 [14]	$(N+2)\|g\|+\|g_T\|$	$(3n_\ell+2)\|g\|+\|g_T\|$	$(2n_u+4)$
Our scheme-1	$(3N+2)\|g\|+\|g_T\|$	$(2n_\ell+l)\|g\|+\|g_T\|$	$(2n_u+l+1)\|g\|$
Our scheme-2	$(2N+2)\|g\|+\|g_T\|$	$(n_\ell+l)\|g\|+\|g_T\|$	$(n_u+l+1)\|g\|$

Table 2: Comparison of computation cost.

Schemes	KeyGen	Encrypt	Decrypt
LMX13 [12]	$7n_u\mathbb{E}$	$\mathbb{E}_T+(4n_\ell+4)\mathbb{E}+\mathbb{P}$	$(5n_u+1)\mathbb{P}$
WZZ14 [13]	$(n_u+1)\mathbb{E}$	$\mathbb{E}_T+(n_\ell n_a+1)\mathbb{E}+\mathbb{P}$	$(n_u l_w+1)\mathbb{P}$
YYZ [14]	$(2n_u+4)\mathbb{E}$	$\mathbb{E}_T+(3n_\ell+2)\mathbb{E}+\mathbb{P}$	$(3n_u+1)\mathbb{P}$
Our scheme-1	$(2n_u+l+1)\mathbb{E}$	$\mathbb{E}_T+(4n_\ell+l+1)\mathbb{E}+\mathbb{P}$	$(4n_u+2l_w)\mathbb{P}$
Our scheme-2	$(n_u+l+1)\mathbb{E}$	$\mathbb{E}_T+(2n_\ell+l+1)\mathbb{E}+\mathbb{P}$	$(2n_u+2l_w)\mathbb{P}$

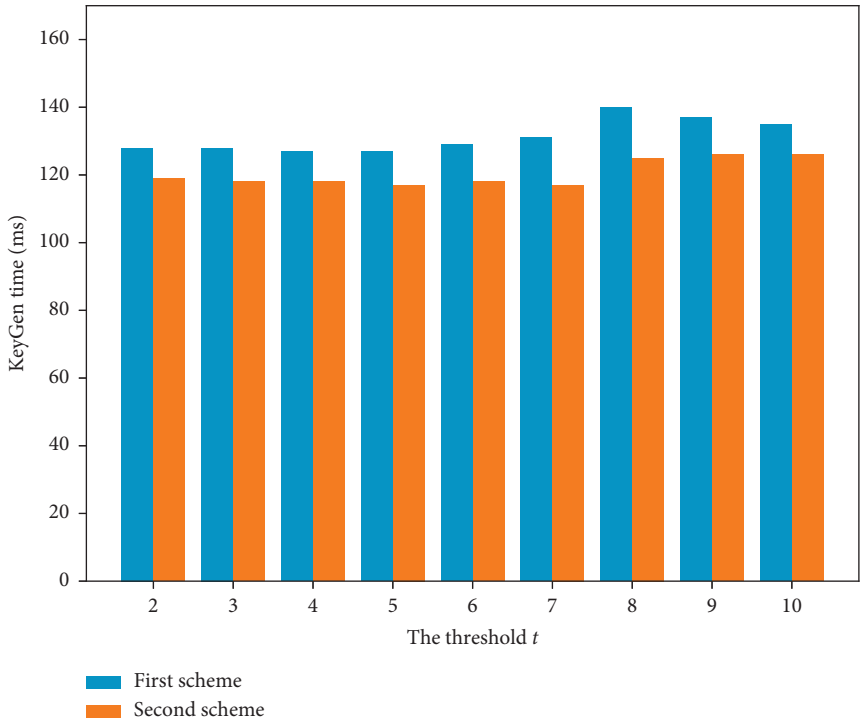

Figure 3: The value of key generation time with threshold t.

relatively long time to evaluate the intersection of set T and S in the decryption phase. Nevertheless, our schemes enjoy tolerable computational efficiency for the following reasons. Clearly, the time consumption does not exceed 130 ms in all phases. To be precise, when a user owns 30 attributes, the time consumption of the first scheme is 123 ms, while that of the second scheme is 120 ms. Therefore, the efficiency of our proposed schemes is acceptable in practical scenarios. Furthermore, we remark that in the IoT scenario, the relatively intensive computation can be offloaded to some outsourced equipment, and the rest of the operations remain on the receiver.

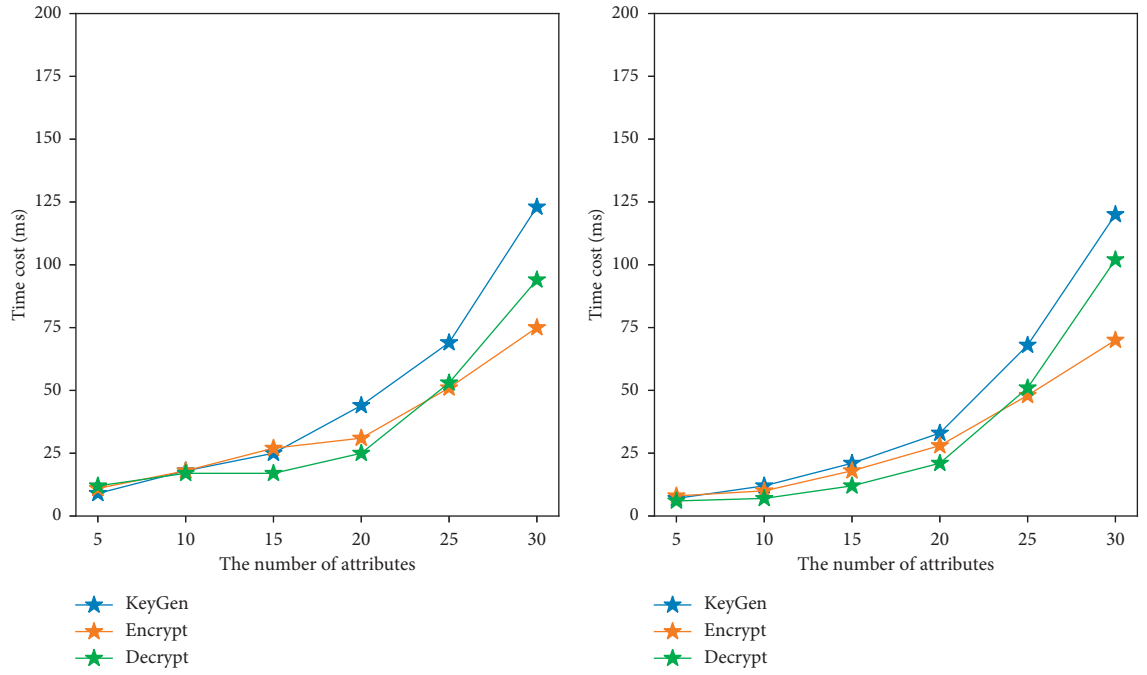

Figure 4: Time cost of the two proposed schemes.

7. Conclusion

In this paper, we propose two multi-authority criteria-based encryption schemes that support data access control in IoT and are proved to be secure in the standard model. Specifically, they solve the problem of security bottleneck and server overload caused by involving only a single authority in the phase of key generation. Moreover, each criterion carries a weight specified by the encryptor, which allows the access policy to be expressed more flexibly. The theoretical analysis and simulation evaluation demonstrate that our schemes can conform to the actual application scenarios. The remaining problem is that the time consumption of each phase in the schemes increases nonlinearly, which limits the size of the criterion universe. In future work, we are committed to constructing more lightweight frameworks.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This study was supported by the Fundamental Research Funds for the Central Universities (no. 3072020CFJ0601) and CSIC 722 Innovation Fund (no. KCJJ2019-12).

References

[1] C. Wang, D. Wang, G. Xu, and D. He, *Efficient Privacy-Preserving User Authentication Scheme with Forward Secrecy for Industry 4.0*, Science China Information Sciences, Beijing, China.

[2] R. Roman, J. Lopez, and M. Mambo, "Mobile edge computing, fog et al.: a survey and analysis of security threats and challenges," *Future Generation Computer Systems*, vol. 78, pp. 680–698, 2018.

[3] D. Wang, X. Zhang, Z. Zhang, and P. Wang, "Understanding security failures of multi-factor authentication schemes for multi-server environments," *Computers & Security*, vol. 88, Article ID 101619, 2020.

[4] L.-Y. Yeh, P.-Y. Chiang, Y.-L. Tsai, and J.-L. Huang, "Cloud-based fine-grained health information access control framework for lightweight IoT devices with dynamic auditing and attribute revocation," *IEEE Transactions on Cloud Computing*, vol. 6, no. 2, pp. 532–544, 2018.

[5] S. Qiu, D. Wang, G. Xu, and S. Kumari, "Practical and provably secure three-factor authentication protocol based on extended chaotic-maps for mobile lightweight devices," *IEEE Transactions on Dependable and Secure Computing*, 2020.

[6] C. Wang, D. Wang, Y. Tu, G. Xu, and H. Wang, "Understanding node capture attacks in user authentication schemes for wireless sensor networks," *IEEE Transactions on Dependable and Secure Computing*, 2020.

[7] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proceedings of the EUROCRYPT*, pp. 457–473, Aarhus, Denmark, May 2005.

[8] B. Waters, "Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization," in *Proceedings of 14th International Conference on Practice and Theory in Public Key Cryptography (PKC'11)*, pp. 53–70, Taormina, Italy, March 2011.

[9] N. Attrapadung, "Dual system encryption via doubly selective security: framework, fully secure functional encryption for regular languages, and more," in *Proceedings of the EUROCRYPT*, pp. 457–473, Aarhus, Denmark, May 2014.

[10] Q. Li, J. Ma, R. Li, J. Xiong, and X. Liu, "Provably secure unbounded multi-authority ciphertext-policy attribute-based encryption," *Security and Communication Networks*, vol. 8, no. 18, pp. 4098–4109, 2015.

[11] J. Li, S. Wang, and Y. Li, "An efficient attribute-based encryption scheme with policy update and file update in cloud computing," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 12, pp. 6500–6509, 2019.

[12] X. Liu, J. Ma, J. Xiong et al., "Ciphertext-policy weighted attribute-based encryption for fine-grained access control," in *Proceedings of the International Conference on Intelligent Networking and Collaborative Systems*, pp. 51–57, IEEE, Xi'an, China, May 2013.

[13] Y. Wang, D. Zhang, and H. Zhong, "Multi-authority based weighted attribute encryption scheme in cloud computing," in *Proceedings of the International Conference on Natural Computation*, pp. 1033–1038, IEEE, Xiamen, China, August 2014.

[14] X. Yan, X. Yuan, Q. Zhang, and Y. Tang, "Traceable and weighted attribute-based encryption scheme in the cloud environment," *IEEE Access*, vol. 8, pp. 38285–38295, 2020.

[15] T. V. X. Phuong, G. Yang, and W. Susilo, "Criteria-based encryption," *The Computer Journal*, vol. 61, no. 4, pp. 512–525, 2018.

[16] V. Goyal, O. Pandey, A. Sahai et al., "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the ACM Conference on Computer and Communications Security*, pp. 89–98, Alexandria, VA, USA, October 2006.

[17] A. Sahai, H. Seyalioglu, and B. Waters, "Dynamic credentials and ciphertext delegation for attribute-based encryption," in *Proceedings of the CRYPTO*, pp. 199–217, Santa Barbara, CA, USA, August 2012.

[18] S. Agrawal and M. Chase, "Fast attribute-based message encryption," in *Proceedings of the ACM Conference on Computer and Communications Security*, pp. 665–682, Dallas, TX, USA, October 2017.

[19] B. Waters, "Functional encryption for regular languages," in *Proceedings of the CRYPTO*, pp. 218–235, Santa Barbara, CA, USA, August 2012.

[20] S. Agrawal, M. Maitra, and S. Yamada, "Attribute based encryption for deterministic finite automata from DLIN," in *Proceedings of the Theory of Cryptography Conference*, pp. 91–117, Nuremberg, Germany, November 2019.

[21] W. Li, K. Xue, Y. Xue, and J. Hong, "A robust and verifiable threshold multi-authority access control system in public cloud storage," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 5, pp. 1484–1496, 2015.

[22] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Proceedings of the EUROCRYPT*, pp. 568–588, Tallinn, Estonia, May 2011.

[23] J. Wei, W. Liu, and X. Hu, "Secure and efficient attribute-based access control for multi-authority cloud storage," *IEEE Systems Journal*, vol. 12, no. 2, pp. 1731–1742, 2016.

[24] Q. Li, J. Ma, R. Li, X. Liu, J. Xiong, and D. Chen, "Secure, efficient and revocable multi-authority access control system in cloud storage," *Computers & Security*, vol. 59, pp. 45–59, 2016.

[25] J. Li, X. Chen, S. S. M. Chow, Q. Huang, D. S. Wong, and Z. Liu, "Multi-authority fine-grained access control with accountability and its application in cloud," *Journal of Network and Computer Applications*, vol. 112, pp. 89–96, 2018.

[26] Q. M. Malluhi, A. Shikfa, V. D. Tran, and V. C. Trinh, "Decentralized ciphertext-policy attribute-based encryption schemes for lightweight devices," *Computer Communications*, vol. 145, pp. 113–125, 2019.

[27] V. K. A Sandor, Y. Lin, X. Li, F. Lin, and S. Zhang, "Efficient decentralized multi-authority attribute-based encryption for mobile cloud data storage," *Journal of Network and Computer Applications*, vol. 129, pp. 25–36, 2019.

[28] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612-613, 1979.

[29] T. P. Pedersen, "A threshold cryptosystem without a trusted party," in *Proceedings of the EUROCRYPT*, pp. 522–526, Brighton, UK, April 1991.

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

[31] https://support.microsoft.com/zh-cn/visio.

[32] J. A. Akinyele, C. Garman, I. Miers et al., "Charm: a framework for rapidly prototyping cryptosystems," *Journal of Cryptographic Engineering*, vol. 3, no. 2, pp. 111–128, 2013.