Hindawi
Security and Communication Networks
Volume 2021, Article ID 8872699, 22 pages
https://doi.org/10.1155/2021/8872699

Research Article

Practical Multiauthority Attribute-Based Access Control for Edge-Cloud-Aided Internet of Things

Kaiqing Huang,[1,2] **Xueli Wang**,[2] **and Zhiqiang Lin**[3]

[1]*Department of Basic Courses, Dongguan Polytechnic, Dongguan, 523808, China*
[2]*School of Mathematical Sciences, South China Normal University, Guangzhou, 510631, China*
[3]*School of Mathematics and Information Science, Guangzhou University, Guangzhou, 510006, China*

Correspondence should be addressed to Kaiqing Huang; kqhuang@m.scnu.edu.cn

Received 28 June 2020; Revised 11 November 2020; Accepted 29 November 2020; Published 2 February 2021

Academic Editor: Salvatore D'Antonio

With the assistance of edge computing which reduces the heavy burden of the cloud center server by using the network edge servers, the Internet of Things (IoTs) architectures enable low latency for real-time devices and applications. However, there still exist security challenges on data access control for the IoT. Multiauthority attribute-based encryption (MA-ABE) is a promising technique to achieve access control over encrypted data in cross-domain applications. Based on the characteristics and technical requirements of the IoT, we propose an efficient fine-grained revocable large universe multiauthority access control scheme. In the proposed scheme, the most expensive encryption operations have been executed in the user's initialization phase by adding a reusable ciphertext pool besides splitting the encryption algorithm to online encryption and offline encryption. Massive decryption operations are outsourced to the near-edge servers for reducing the computation overhead of decryption. An efficient revocation mechanism is designed to change users' access privileges dynamically. Moreover, the scheme supports ciphertext verification. Only valid ciphertext can be stored and transmitted, which saves system resources. With the help of the chameleon hash function, the proposed scheme is proven CCA2-secure under the q-DPBDHE2 assumption. The performance analysis results indicate that the proposed scheme is efficient and suitable in edge computing for the IoT.

1. Introduction

With the development of the 5G network and the Internet of Things (IoTs) technology, more and more objects are connected to the network via information sensing devices which generate a huge amount of data [1, 2]. Cisco predicted that the total amount of data created (and not necessarily stored) by any device will reach 847 ZB per year by 2021 [3].

In traditional cloud computing modes, these massive data are all sent to the cloud for storing and processing, which consumes a large amount of network bandwidth and computing resources. Moreover, massive access requests from users may lead to service interruption and generate traffic overload or even network delay, etc. As a result, the edge computing model is proposed to offload computing tasks to edge servers [4–9].

In an edge computing model, the network edge servers near the data source will preliminarily handle the data and tasks and serves the adjacent devices in time. Then, it selectively sends the important data to the core network (cloud). Consequently, the edge servers reduce the load of cloud centers and ensure efficient network operation for improved services.

However, edge computing is facing some new challenging issues [4–9]. On the one hand, the users' data are mostly privacy-sensitive. A good way to ensure data security and privacy protection is that the data are encrypted before being outsourced and only allow authorized users to access it. On the other hand, users frequently enter and leave the network, and their access privileges change dynamically, which makes the access control of users' data complex and dynamic. Unfortunately, it is difficult for traditional access

control technology to deal with the above problems in such a complex edge computing environment. Therefore, an efficient lightweight data encryption mechanism with fine-grained access control is indispensable for the IoT.

Attribute-based encryption (ABE) is a promising technique (cryptosystem) for protecting data confidentiality and dynamic fine-grained authorized access control through the attribute management, with which the data owners can encrypt data files according to the attribute of the target recipients without knowing their exact identities.

In 2005, Sahai and Waters introduced fuzzy identity-based encryption (FIBE) which can be applied to enable encryption using error-tolerance biometric inputs as identities. Moreover, FIBE is extended into attribute-based encryption (ABE) by defining the biometric identity as a set of attributes [10]. In 2006, Goyal et al. distinguished ABE into key-policy attribute-based encryption (KP-ABE) and ciphertext-policy attribute-based encryption (CP-ABE) [11]. In KP-ABE, secret keys are associated with access policies, and ciphertexts are associated with attributes, while in CP-ABE, the ciphertexts are associated with access policies, and secret keys are associated with attributes. Decryption is enabled if and only if the user's attribute set satisfies the ciphertext access structure. Goyal et al. gave a secure construction of KP-ABE [11], and Bethencourt et al. presented a secure CP-ABE construction against collusion attacks in the generic group model [12]. In 2008, Goyal et al. presented the first construction of a ciphertext-policy attribute-based encryption scheme with a security proof based on a number-theoretic assumption [13]. In 2011, Waters presented the first CP-ABE construction which is proven selectively secure under the decisional parallel bilinear Diffie-Hellman exponent (PBDHE) assumption in the standard model. In 2013, Rouselakis and Waters proposed two large universe attribute-based encryption constructions (CP-ABE and KP-ABE) on prime-order bilinear groups. Both schemes are selectively secure in the standard model under two "q-type" assumptions [14]. However, all these schemes are single-authority ABE.

1.1. Issues and Related Work. Attribute-based access control in the IoT should efficiently address some practical issues in the following details.

Firstly, the massive data produced in the IoT are used in large-scale cross-domain applications. Multipart collaborative (MPC) is one of the motivating scenarios, where data owners want to share their data across different domains and organizations. To illustrate, in a multiexpert cooperative diagnosis system, a patient Alice may need to share her medical data generated by medical wearing equipment with different experts in different affiliations. She defines the access policy as "surgeon@hospital A" or "surgeon@hospital B and medical researcher@research center C," where the attributes "surgeon@hospital A," "surgeon@hospital B," and "medical researcher@research center C" are issued by "hospital A," "hospital B," and "research center C," respectively. In this scenario, three authorities, "hospital A," "hospital B," and "research center C," are needed. Meanwhile, in ABE schemes, if the system attribute universe is "small," the attributes are fixed and enumerated at system setup, which is not practical in the IoT. Conversely, in "large universe" construction, any string can be used as an attribute, and the attributes are not necessarily enumerated at system setup, which is more flexible and practical for the IoT.

To deal with the scenario that the users' attributes are issued by different authorities in the same system, Chase proposed multiauthority attribute-based encryption, which is the first construction to support multiple attribute authorities in 2007 [15]. In 2009, Chase and Chow proposed a solution that removes the trusted central authority and protects the users' privacy by preventing the authorities from pooling their information on particular users [16]. In 2011, Lewko and Waters proposed the first fully secure multi-authority ciphertext-policy attribute-based encryption system which does not require any central authority and collision resistance. They proved their system secure using the dual system encryption methodology in the random oracle model [17]. In 2011, Liu et al. proposed a new multiauthority CP-ABE system which is adaptively secure in the standard model with adaptive authority corruption and can support the large attribute universe [18]. In 2015, Rouselakis and Waters proposed an efficient large-universe multiauthority ciphertext-policy attribute-based encryption system, which uses the significantly faster prime-order bilinear groups rather than composite order groups [19].

Secondly, in the IoT, data users may change their subscription or leave the system, which results in losing part of the access privileges or all the access privileges. To protect the interests of the data owners, an efficient and secure revocation mechanism must be designed. In the above example, an expert Bower with the attribute set "surgeon@hospital B, medical researcher@research center C" departures from research center C, then he loses the privilege to access Alice's encrypted data. To avoid Alice's privacy disclosure, the authority "research center C" should revoke Bower's attribute "medical researcher@research center C." Similarly, when Bower departures from the multiexpert cooperative diagnosis system, Bower should be revoked from the system. When the attribute "medical researcher@research center C" is dropped by the authority "research center C," it also should be revoked.

To solve the security problems brought by the dynamic change in users' access privilege, researchers have come up with revocable ABE. In 2008, Boldyreva et al. firstly proposed an ABE scheme with indirect user revocation. The sender encrypts with the present time slot regarded as an attribute. The authority who possesses the current revocation list, periodically announces the key update material so that only unrevoked users can update their key and decrypt ciphertexts [20]. In 2009, Attrapadung and Imai proposed an ABE system with direct user revocation. It allows senders to specify the revocation list directly when encrypting [21]. The authority does not need to update the secret key concerning the revoked attribute for each nonrevoked user. But it should publish a revocation list and the data users are required to synchronize this revocation list all the time, which makes it

not suitable for large-scale systems. It still faces a backward security problem that the revoked users can access the old data they were authorized to access before being revoked. In 2009, Attrapadung and Imai presented the first hybrid revocable ABE scheme that allows senders to select on-the-fly whether to use either direct or indirect revocation mode when encrypting [22]. In 2011, based on the subset-cover revocation framework [23], Hur et al. proposed an access control mechanism using CP-ABE to enforce access control policies with efficient attribute and user revocation capability [24]. In 2012, Sahai et al. introduced the concept of revocable storage and proposed the revocable ABE by updating keys and CSP by updating ciphertext without accessing any secret information, and the updated ciphertexts are no longer decryptable by revoked users, but the schemes are inefficient [25]. In 2013, Yang et al. constructed a new multiauthority CP-ABE scheme and designed an attribute revocation method that can achieve both forward security and backward security, which is secure under weaker security assumptions [26]. However, the scheme only supports the small attribute universe. In 2017, Li et al. provided a CP-ABE scheme with efficient user revocation by introducing the concept of user group [27].

Thirdly, in ABE, bilinear pairing operation is an expensive computation cost, and its number increases along with the number of attributes involved in the access structure. The drawback of high computation overhead in the ABE schemes with multifunction becomes more serious for resource-constrained users such as mobile devices, sensors, and medical wearing equipment [28]. To save energy for the resource-constrained users and ensure the rapid generation and acquisition of the data, some technologies should be used to reduce the computational overhead both in the encryption phase and the decryption phase.

For eliminating the overhead of decryption for users, Matthew Green et al. [29] gave the methods for efficiently and securely outsourcing decryption which offloads most decryption operations to a third-part server provider and returns a partial decryption ciphertext. The user performs only a little time of exponential operation to recover the plaintext. In 2013, Lai et al. firstly constructed a concrete ABE scheme with verifiable outsourced decryption which allows for verifiability of the partial decryption ciphertext [30]. Li et al. proposed a new secure outsourced ABE system, which supports both secure outsourced key-issuing and decryption with checkability of the outsourced computation results in an efficient way [31]. There are also some works to make verifiable outsourced decryption more efficiently and securely in ABE [32–34].

Meanwhile, for saving online computation, Hohenberger and Waters proposed the first online/offline ABE [35] which splits the encryption into two phases: the data owner does as much precomputation as possible to encrypt a message or create a secret key in the offline phase and rapidly assembles a ciphertext or key in the online phase. It is a promising technique for resource-limited devices. In 2018, Li et al. proposed a new scheme that eliminates a majority of the encryption computation task by adding system public parameters [36].

Last but not the least, it may be that the data owner device makes an error to generate the wrong ciphertext, or the malicious user generates many wrong ciphertexts to attack the system. These wrong ciphertext not only waste system resources but also waste user resources. For example, it wastes data user resources to decrypt the error ciphertext which returns an error result even though his attributes satisfy the access structure. A ciphertext verification algorithm should be used to eliminate invalid ciphertexts as much as possible. In 2014, Liu et al. proposed a KP-ABE scheme with the public ciphertext test, but the verify equations grow linearly with the number of the attributes involved in the access structure, which requires high computation overhead [37].

1.2. Motivation and Contributions. Recently, there have been many researchers who proposed the multifunctional MA-ABE system trying to meet the above needs, but no one solving all the issues simultaneously. In 2017, De and Ruj proposed a decentralized attribute-based encryption scheme with online/offline encryption, outsourced decryption, and user revocation, but their scheme only supports the small universe [38]. In 2018, Liu et al. designed an MA-ABE scheme that simultaneously supports the large attribute universe, outsourcing decryption, and attribute revocation, but their scheme is statically secure, without online/offline encryption and ciphertext publicly verifiable [39]. In 2019, Li et al. constructed a CCA2-secure publicly verifiable revocable large-universe multiauthority attribute-based encryption, but their scheme does not support online/offline encryption and outsourcing decryption [40]. In their scheme, the verify equations grow linearly with the number of the attributes involved in the access structure, which require high computation overhead. Moreover, if the user's some attributes involved in the ciphertext are revoked, she/he will fail to verify the validity of the ciphertext even though her/his rest attributes still have the access privilege to the ciphertext.

As we revisited the existing ABE schemes, no one simultaneously solves all the practical issues. Most of them only concentrate on solving one specific issue. A few schemes try to solve some issues but cause new problems. Therefore, we designed an efficient, CCA2-secure, flexible, fine-grained access control scheme for the IoT using edge computing, which can solve the above issues efficiently. The proposed scheme meets the need of the large-scale multi-domain collaboration in the IoT by using multiauthority attribute-based encryption with the large attribute universe. More contributions are as follows:

(1) By adding a reusable ciphertext pool, the most expensive encryption operations have been executed in the user's initialization phase, which minimizes the offline/online encryption computation. At the same time, the near-edge servers can perform the major decryption with the help of transformed keys given by the data user. As a result, the computation overhead of encryption and decryption is substantially reduced both on the data owner and user sides,

so the scheme is suitable for the resource-constrained users in the IoT.

(2) The proposed scheme designs a fine-grained revocation mechanism to revoke an attribute from the user, by which the user or the attribute can be revoked from the system. So, it can change the user's access privilege timely and is fit for the dynamic IoT.

(3) The scheme supports efficient ciphertext verification, and only valid ciphertext can be stored and transmitted, which saves the system resources. For reducing the computation overhead, we combine all the verify equations into one equation like a "hash function" which reduces expensive pairing operation. Moreover, the user can verify the validity of the ciphertext even though some attributes involved in the ciphertext are revoked, only if the rest of her/his attributes still have the access privilege to the ciphertext.

(4) With the help of the chameleon hash function, the simulator successfully prepares the challenge ciphertext with the dummy attribute in the security proof. Consequently, the proposed scheme is proven CCA2-secure under the q-DPBDHE2 assumption, which is more secure than the previous scheme. The performance analysis results indicate that the proposed scheme is efficient and suitable in edge computing for the IoT.

2. Preliminaries

2.1. Notations. In order to facilitate the understanding, we explain some notations used throughout this article in Table 1.

2.2. Bilinear Pairings and Complexity Assumption

Definition 1 (bilinear pairings). Let $\mathbb{G}$ and $\mathbb{G}_T$ be the cyclic multiplicative groups with prime order p. The identities of $\mathbb{G}$ and $\mathbb{G}_T$ are denoted as $1_{\mathbb{G}}$ and $1_{\mathbb{G}_T}$, respectively. We say a map e: $\mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$ is a bilinear pairing if it satisfies the following properties:

(1) Bilinear.
$\forall g_1, g_2 \in \mathbb{G}, \forall a, b \in \mathbb{Z}_p^*, e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$.

(2) Nondegenerate. $\exists g_1, g_2 \in \mathbb{G}, s.t.\ e(g_1, g_2) \neq 1_{\mathbb{G}_T}$.

(3) Computable. There is an efficient algorithm to compute $e(g_1, g_2), \forall g_1, g_2 \in \mathbb{G}$.

Definition 2 (q-DPBDHE2 problem [41]). Let $\mathbb{G}$ and $\mathbb{G}_T$ be the bilinear groups with prime order p and g be a generator of $\mathbb{G}$. e: $\mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$ is a bilinear map defined on $\mathbb{G}$. Pick $s, a, b_1, b_2, \ldots, b_q \xleftarrow{R} \mathbb{Z}_p$ and $R \xleftarrow{R} \mathbb{G}_T$. Given

$$D = \mathbb{G}, p, e, g, g^s, g^{a^i}, g^{a^i b_j}, g^{\left(s/b_j\right)}, \quad i \in [2q], j \in [q], i \neq q+1, g^{\left(sa^i b_j\right)/b_{j'}}, \quad i \in [q+1], j \in [q], j' \in [q], j \neq j', \tag{1}$$

and the algorithm is asked to distinguish $(D, e(g,g)^{sa^{q+1}})$ from (D, R).

Definition 3 (q-DPBDHE2 assumption [41]). The q-DPBDHE2 assumption holds in $\mathbb{G}$ if no probabilistic polynomial time algorithm has probability at least $(1/2) + \varepsilon$ in solving the q-DPBDHE2 problem in $\mathbb{G}$ for nonnegligible ε.

2.3. Access Structures and Linear Secret-Sharing Schemes

Definition 4 (access structure [42]). Let $P = \{P_1, P_2, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^P$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$ then $C \in \mathbb{A}$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of nonempty subsets of P, i.e., $A \in 2^P \backslash \{\varnothing\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets.

In the attribute-based encryption scheme, the parties are replaced by the attributes. An access structure $\mathbb{A}$ will contain some authorized sets of attributes. Similarly, we restrict our attention to monotone access structures. From now on, by an access structure, we mean a monotone access structure.

Definition 5 (linear secret-sharing schemes (LSSS)) [42]). Let p be a prime and U the attribute universe. A secret-sharing scheme π with domain of secrets $\mathbb{Z}_p$ realizing access structures on U is linear over $\mathbb{Z}_p$ if the following holds:

(1) The shares of a secret $s \in \mathbb{Z}_p$ for each attribute form a vector over $\mathbb{Z}_p$.

(2) For each access structure $\mathbb{A}$ on U, there exists a matrix $M \in \mathbb{Z}_p^{l \times n}$, called the share-generating matrix, and a function ρ that labels the rows of M with attributes from U, i.e., ρ: $[l] \longrightarrow U$, which satisfies the following: during the generation of the shares, we consider the column vector $\overrightarrow{v} = (s, r_2, \ldots, r_n)^\top$, where $r_2, \ldots, r_n \xleftarrow{\$}$. Then, the vector of l shares of the secret s according to π is equal to $M\overrightarrow{v} \in \mathbb{Z}_p^{l \times 1}$. The share $(M\overrightarrow{v})_j$ where $j \in [l]$ "belongs" to attribute $\rho(j)$. We will be referring to the pair (M, ρ) as the policy of the access structure $\mathbb{A}$.

2.4. Chameleon Hash Functions. A chameleon hash function consists of three polynomial time algorithms as follows [43]:

KeyGen $(1^\lambda) \longrightarrow (\mathrm{SK}_{\mathrm{ch}}, \mathrm{PK}_{\mathrm{ch}})$: it takes the security parameter λ as input and outputs a secret key $\mathrm{SK}_{\mathrm{ch}}$ and the corresponding public key $\mathrm{PK}_{\mathrm{ch}}$.

Table 1: Entities.

Symbol	Description
$[n]$	Integer set $\{1, 2, \ldots, n\}$
$[i_R j]$	Set $\{i, i+1, i+2, \ldots, j\}$ containing consecutive integers
$s \leftarrow S$	Variable s is chosen uniformly at random from the set S
$\mathrm{negl}(n)$	Negligible function in n
PPT	Probabilistic polynomial time
$\mathbb{Z}_p^*$	Set $\mathbb{Z}_p - \{0\}$
$\mathbb{Z}_p^{m \times n}$	Set of matrices of size $m \times n$ with elements in $\mathbb{Z}_p$
AID/UID	Attribute authority global identity/User global identity
U/U_{AA}	System attribute/authority universe
MK	System master key
GP	Global public parameter
H/F	Function maps each UID/attribute to an element of $\mathbb{G}$
T	Function maps each attribute to the unique attribute authority who controls it
$\mathrm{SK}_{\mathrm{AID}}/\mathrm{PK}_{\mathrm{AID}}$	Secret key/public key for authority AID
$\mathrm{SK}_{\mathrm{UID},a}/P_{\mathrm{UID},a}$	Private key/path key of attribute a for user UID
$\mathrm{CPool}_{\mathrm{UID}}$	Immediate ciphertext pool of user UID
$\mathrm{CT}_{\mathrm{off}}/\mathrm{CT}_{\mathrm{on}}$	Offline/online ciphertext
CT	Original ciphertext generated by the data owner
$\mathrm{CT}_{\mathrm{CSP}}$	Ciphertext generated after ESP re-encrypts CT by using the latest attribute key
S_{UID}	Attribute set of user UID
AT_a	Binary state tree of attribute a
AK_a	Attribute key of attribute a
RL_a	Revocation list of attribute a
UpAK_a	Update key of attribute a
$\mathrm{TK}_{\mathrm{UID}}$	Transformation key for user UID
$\mathrm{RK}_{\mathrm{UID},a}$	Retrieving key of attribute a for user UID
$\mathrm{CT}_{\mathrm{out}}$	Partial decrypted ciphertext generated by ESP

$H_{\mathrm{ch}}(\mathrm{PK}_{\mathrm{ch}}, m, r_{\mathrm{ch}}) \longrightarrow v$: it takes in the public key $\mathrm{PK}_{\mathrm{ch}}$, a message m, and an auxiliary parameter r_{ch} and then outputs the hashed value v.

$\mathrm{TCollision}_{\mathrm{ch}}(\mathrm{SK}_{\mathrm{ch}}, m, r_{\mathrm{ch}}, m') \longrightarrow r_{\mathrm{ch}}'$: it takes as inputs the secret key $\mathrm{SK}_{\mathrm{ch}}$, a message m with its auxiliary parameter r_{ch}, and another $m' = m$ and outputs another auxiliary parameter r_{ch}' such that $v' = H_{\mathrm{ch}}(\mathrm{PK}_{\mathrm{ch}}, m', r_{\mathrm{ch}}') = H_{\mathrm{ch}}(\mathrm{PK}_{\mathrm{ch}}, m, r_{\mathrm{ch}}) = v$.

A secure chameleon hash function satisfies the requirements of collision resistance and uniformity. All messages m induce the same probability distribution on $H_{\mathrm{ch}}(\mathrm{PK}_{\mathrm{ch}}, m, r_{\mathrm{ch}})$ for r_{ch} chosen uniformly at random. There is no efficient algorithm that takes as input the Chameleon hash public key $\mathrm{PK}_{\mathrm{ch}}$ to find two pairs (m, r_{ch}), (m', r_{ch}') where $m \neq m'$ such that $H_{\mathrm{ch}}(\mathrm{PK}_{\mathrm{ch}}, m', r_{\mathrm{ch}}') = H_{\mathrm{ch}}(\mathrm{PK}_{\mathrm{ch}}, m, r_{\mathrm{ch}})$ except with negligible probability. Only the one who knows the chameleon hash secret key can find collision for every given input efficiently.

2.5. Subset-Cover Revocation Framework. Naor et al. proposed the subset-cover revocation framework [23] and described two algorithms: the complete subtree method and the subset difference method. We can realize the revocation in ABE based on the complete subtree method.

As shown in Figure 1, firstly, it establishes a binary state tree BT. Each user is assigned as a leaf node u. Let v_i denote a node and v_{il} (v_{ir}) denote the left (right) child of v_i. $P(v_i)$ records all the nodes which are on the path from v_i to the root. If a user v_i is revoked, v_i will be added into the revocation list RL. $\mathrm{MinSubCover}(\mathrm{BT}, \mathrm{RL})$ is an algorithm to generate the minimum subset cover whose descendant nodes cover all the unrevoked users (Algorithm 1).

As shown in Figure 1, there are eight users $u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8$. If the user u_2 is revoked, N_9 would be added into $\mathrm{RL} = \{N_9\}$, and the $\mathrm{MinSubCover}(\mathrm{BT}, \mathrm{RL})$ outputs $\{N_3, N_5, N_8\}$.

3. System Model and Security Model

3.1. System Model. As shown in Figure 2, our efficient CCA2-secure flexible fine-grained access control for the IoT by using edge computing consists of the following entities:

Central authority (CA): CA is a trusted entity that initializes the system by setting up the global public parameters. Moreover, it is responsible for the registration of users and authorized attribute authorities. CA assigns each user a unique identity UID, and each attribute authority is a unique identity AID.

Attribute authority (AA): AA is a trusted entity that manages the users' attributes. Each AA is an independent attribute authority and manages a disjoint attribute set, respectively, which means that each

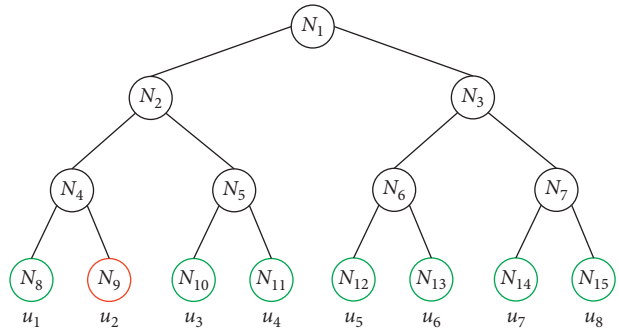

Figure 1: Binary state tree BT.

(1) $X = \varnothing, Y = \varnothing$
(2) **for** $\forall v_i \in$ RL**do**
(3) add $P(v_i)$ to X
(4) **end for**
(5) **for** $\forall x \in X$**do**
(6) **if** $x_l \notin X$**then**
(7) add x_l to Y
(8) **end if**
(9) **if** $x_r \notin X$**then**
(10) add x_r to Y
(11) **end if**
(12) **end for**
(13) **if** $Y = \varnothing$**then**
(14) add root to Y
(15) **end if**
(16) **return** Y

Algorithm 1: MinSubCover (BT, RL).

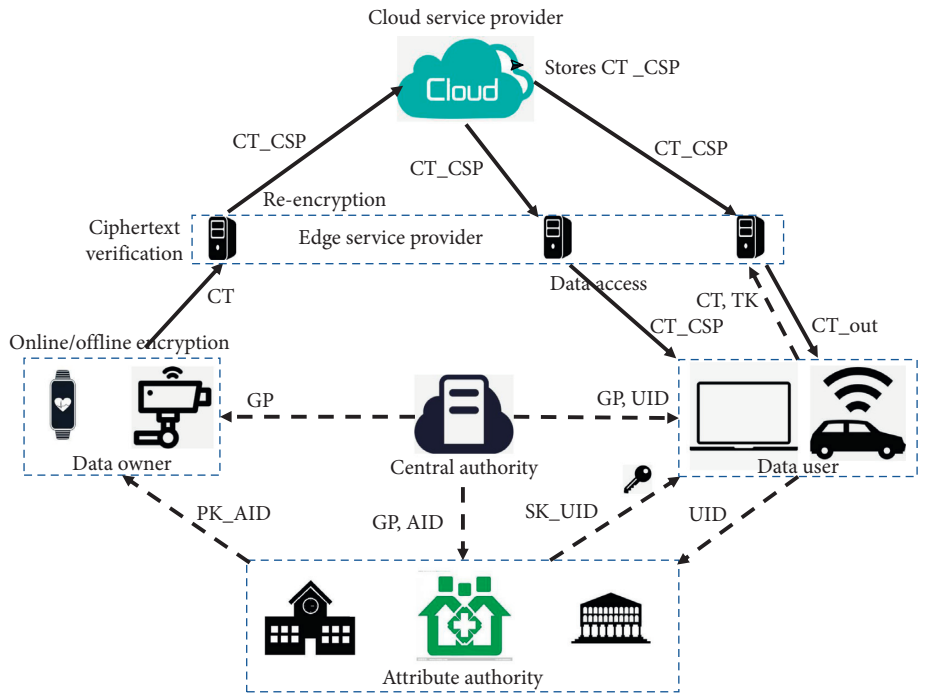

Figure 2: System model.

attribute is associated with a single AA. Each AA issues its public key, authenticates users' attribute sets, and generates the corresponding private keys for them. It also can revoke and update users' attributes by using attribute keys.

Cloud service provider (CSP): CSP is semitrusted (honest-but-curious). It will honestly and correctly execute the tasks but be curious about the data messages which it receives. CSP will not conclude with the malicious users. It stores the valid ciphertexts which are sent from ESP.

Edge service provider (ESP): ESP is semitrusted (honest-but-curious) and will not conclude with the malicious users. It has both computing capability and large storage. It is geographically close to the users (data owners and data users). ESP receives the ciphertext from users and re-encrypts the valid ciphertext by using attribute keys after verifying the validity of ciphertext and then sends the legal data to CSP for permanent storage. It is also in charge of re-encrypting and updating the ciphertext when a revocation happens. Furthermore, it provides partial decryption of ciphertext for some resource-limited users if they require it.

Data owner (DO): to ensure data confidentiality and achieve flexible access to data, DO has two methods to encrypt data before uploading data to ESP. One is DO directly encrypting the data file under an access policy about who can get access to the data. The other is DO firstly encrypting the data file using a symmetric encryption algorithm which is a lightweight encryption method and then encrypting the symmetric key under the access policy.

Data user (DU): each DU is assigned a global user identity UID by CA. DU obtains a set of attributes privileges and corresponding decryption private keys from authorities. DU can freely get any encrypted data from ESP and decrypt the ciphertext if and only if his/her attribute set satisfies the access policy. Moreover, some resource-limited users can outsource decryption to ESP.

3.2. Framework. This system mainly contains the following polynomial time algorithms:

GlobalSetup$(1^\lambda) \longrightarrow$ (MK, GP): the global set up algorithm is run by CA to set up the system. It takes the system security parameter λ as input and outputs the system master key MK and the global public parameters GP.

AASetup$(\mathrm{GP}, \mathrm{AID}) \longrightarrow (\mathrm{PK}_{\mathrm{AID}}, \mathrm{SK}_{\mathrm{AID}})$: the attribute authority set up algorithm is run by attribute authorities. It takes the authority's identity AID as input and sets up the authority's public key and secret key. The authority keeps the secret key $\mathrm{SK}_{\mathrm{AID}}$ and publishes the public key $\mathrm{PK}_{\mathrm{AID}}$.

UKeyGen$(\mathrm{UID}, \mathrm{AID}, a \in U) \longrightarrow \mathrm{SK}_{\mathrm{UID},a}$: the user key generation algorithm is run by the authority AID. It takes the data user's identity UID, his attribute a, and the authority's secret key as input and outputs the private key $\mathrm{SK}_{\mathrm{UID},a} = < K_{T(a),a,1}, K_{T(a),a,2}, P_{\mathrm{UID},a} >$ for the user UID, where the path key $P_{\mathrm{UID},a}$ is generated by the attribute key generation algorithm.

AttrKeyGen$(\mathrm{UID}, \mathrm{AID}, a \in U) \longrightarrow P_{\mathrm{UID},a}$: the attribute key generation algorithm is run by the authority AID. Firstly, if the binary state tree AT_a has not been set up, it establishes an AT_a and initializes a revocation list $\mathrm{RL}_a = \varnothing$, then chooses a random number $\mathrm{AK}_a \in Z_p^*$ as the attribute key. Each user with the attribute a is assigned as a leaf node, and it increases the height with users increasing. For each node N_i in the tree, it randomly picks $\mathrm{NK}_{a,i} \in Z_p^*$. Path$(\mathrm{UID}, a)$ denotes the path of the user UID from its leaf node to the root node, then $P_{\mathrm{UID},a} = \{\mathrm{NK}_{a,i}\}_{i \in \mathrm{Path}(\mathrm{UID},a)}$ is the path key for the user UID which is the key used to recover AK_a. Finally, it sends $P_{\mathrm{UID},a}$ to the user and then publishes $\{\mathrm{AK}_a \cdot \mathrm{NK}_{a,i}\}_{i \in \mathrm{MinSubCover}(\mathrm{AT}_a, \mathrm{RL}_a)}$ and shares $(\mathrm{AT}_a, \mathrm{AK}_a)$ with ESP and CSP.

CPoolGen$(\mathrm{UID}, \mathrm{GP}, \{\mathrm{PK}_{\mathrm{AID}}\}_{\mathrm{AID} \in U_{AA}}) \longrightarrow \mathrm{CPool}_{\mathrm{UID}}$: the ciphertext pool generation algorithm is run by data owners. It takes the data owner's identity UID, the global public parameters GP, the authorities' public keys $\{\mathrm{PK}_{\mathrm{AID}}\}_{\mathrm{AID} \in U_{AA}}$ as input and chooses an integer N to determine the size of the attributes will be associated with any ciphertext and then outputs the immediate ciphertext pool $\mathrm{CPool}_{\mathrm{UID}}$. $\mathrm{CPool}_{\mathrm{UID}}$ is saved in the encryption algorithm and can be reused. Moreover, it can be updated by the user UID whenever a new attribute authority joins the system or the data owner increases the size of attributes for improving the expressiveness.

Enoff$(\mathrm{GP}) \longrightarrow$ CToff: the offline encryption algorithm is run by data owners. It takes the global public parameters GP as input and outputs the offline ciphertext CToff.

Enon$(m, (M, \rho), \mathrm{GP}, \mathrm{CPool}_{\mathrm{UID}}, \mathrm{CToff}) \longrightarrow$ CT: the online encryption algorithm is run by data owners. It takes a plaintext message m, an access structure (M, ρ), the global public parameters GP, the immediate ciphertext pool $\mathrm{CPool}_{\mathrm{UID}}$, and an offline ciphertext CToff as input and outputs the ciphertext CT.

PublicTest$(\mathrm{GP}, \{\mathrm{PK}_{\mathrm{AID}}\}_{\mathrm{AID} \in \delta(I)}, \mathrm{CT}, I) \longrightarrow$ True/False: the public test algorithm can be run by any role in the system if she/he feels she/he needs it. It takes the global public parameters GP, part of ciphertext which is to be tested, the authorities' public keys $\{\mathrm{PK}_{\mathrm{AID}}\}_{\mathrm{AID} \in \delta(I)}$ as input, and outputs True if the ciphertext is valid.

CReEnc$(\mathrm{CT}, \{\mathrm{AK}_{\rho(i)}\}) \longrightarrow \mathrm{CT}_{\mathrm{CSP}}$: the algorithm is run by ESP. It firstly verifies the validity of the ciphertext by the algorithm PublicTest $(\mathrm{GP}, \{\mathrm{PK}_{\mathrm{AID}}\}, \mathrm{CT}, I)$, where $I = [l]$. If the public test algorithm outputs True, it re-encrypts the ciphertext CT by using the latest attribute keys $\{\mathrm{AK}_{\rho(i)}\}$ and outputs the new ciphertext $\mathrm{CT}_{\mathrm{CSP}}$. At last, ESP sends

the ciphertext CT_{CSP} to CSP and drops the original ciphertext CT.

Decrypt $(CT_{CSP}, UID, GP, \{AK_{\rho(i)}\}, \{SK_{UID,a}\}_{a \in S_{UID}}) \longrightarrow m$: the decryption algorithm is run by the data user UID with a set of attributes S_{UID} which wants to decrypt the ciphertext CT_{CSP}. If $S_{UID} \not\models (M, \rho)$, the algorithm outputs $\perp$. Otherwise, it outputs the message m.

The data user can choose to outsource decryption if he owns limited resource or for saving resource. This feature is implemented in the following three algorithms:

TKGen $(CT_{CSP}, UID, GP, \{AK_{\rho(i)}\}, \{SK_{UID,a}\}) \longrightarrow (\{TK_{UID,a}\}, CT)$: this algorithm is run by data users. It chooses a random numbers as the retrieving key RK_{UID}, takes the unrevoked attributes private key $\{AK_{\rho(i)}\}$, $\{SK_{UID,a}\}$, and CT_{CSP} as input, then outputs the transformation key $\{TK_{UID,a}\}$ and the primitive ciphertext CT.

PartialDec $(\{TK_{UID,a}\}_{\forall a \in I}, CT, GP, \{AK_{\rho(i)}\}) \longrightarrow CT_{out}$: this algorithm is run by ESP. It takes $\{TK_{UID,a}\}$ and CT as input and then outputs the partial decrypted ciphertext CT_{out} to the user.

FullDec $(CT_{out}, RK_{UID}) \longrightarrow m$: this algorithm is run by data users. It works out the massage m by using CT_{out} and RK_{UID}.

Revoke $(UID, a, RL_a) \longrightarrow UpAK_a$: this algorithm is run by attribute authorities. It takes the attribute a and the user UID as input then publishes $\{AK'_a \cdot NK_{a,i}\}_{i \in MinSubCover(AT_a, RL_a)}$ and outputs the update key $UpAK_a$ for ESP and CSP to update the ciphertext and the attribute key, where AK'_a is the new attribute key.

If $T(a)$ wants to revoke the attribute a from the system, it only sets all the users who own a into RL_a and run the above operation.

If the system wants to revoke the user UID from the system, it asks all the involved attribute authorities to revoke all the involved attributes from the user UID, by running the above operation.

3.3. Security Model. In this section, our security model is similar to that in [40] which is named indistinguishability against selective authority and access policy and statically chosen-ciphertext attacks (IND-sAA-sCCA2). At the beginning of the security game, the adversary should claim the access policy which it will challenge, and it should also claim the corrupt authorities. The challenge message can be encrypted by some attributes from some of these corrupt authorities but should at least one attribute from an honest authority, which means that the ciphertext still cannot be attacked successfully if only part of the encrypted attributes is from corrupted authorities.

The security game played between adversary $\mathcal{A}$ and challenger $\mathcal{C}$ is as follows:

Init: adversary $\mathcal{A}$ selects a challenge access policy (A^*, δ^*) and a set of corrupt authorities C_{AA} and sends it to the challenger $\mathcal{C}$.

Global setup: the challenger $\mathcal{C}$ runs the GlobalSetup (1^λ) algorithm to get a system master key MK and the global public parameters GP. It keeps MK and sends GP to $\mathcal{A}$.

Phase 1: the adversary $\mathcal{A}$ issues a polynomially bounded number of queries statically:

Authority's public key queries: $\mathcal{A}$ submits a set of the noncorrupt authorities, and $\mathcal{C}$ replies to $\mathcal{A}$ with the corresponding public keys. $\mathcal{A}$ can create the public keys of the corrupt authorities by itself.

User's secret key queries: $\mathcal{A}$ submits some users identity and a set of his all attributes S^*, $\mathcal{C}$ gives $\mathcal{A}$ the secret keys. If S^* satisfies (A^*, δ^*), $\mathcal{C}$ revokes one of these attributes from all users.

TransformKey queries: $\mathcal{A}$ submits some users with a set of his all attributes S^*, where there is no sense querying the transformation key for the same user who has been queried the secret key. Then, $\mathcal{C}$ gives $\mathcal{A}$ the transformation keys.

CReEncryption queries: $\mathcal{A}$ submits some ciphertext, then $\mathcal{C}$ re-encrypts the ciphertext with the latest attribute keys and return them to $\mathcal{A}$.

Decryption queries: $\mathcal{A}$ submits a ciphertext and $\mathcal{C}$ returns the message if the ciphertext is legitimate.

Revocation queries: $\mathcal{A}$ queries to revoke some users with a set of attributes. $\mathcal{C}$ runs the algorithm Revoke $(UID_i, u, RL_u) \longrightarrow UpAK_u$ to get the update keys and then re-encryptes the ciphertext with the latest update keys.

Challenge: $\mathcal{A}$ submits to $\mathcal{C}$ two equal-length messages m_0 and m_1. $\mathcal{C}$ firstly randomly flips coin $b \in \{0, 1\}$ and encrypts m_b with (A^*, δ^*). Then, it re-encryptes the ciphertext by the latest attribute keys and sends to $\mathcal{A}$.

Phase 2: the same as Phase 1, except that $\mathcal{A}$ cannot make the secret key query for the selected access policy or make decryption query for the challenge ciphertext.

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

Definition 6. The proposed scheme is indistinguishable against selective authority and access policy and statically chosen-ciphertext attacks if no probabilistic polynomial time adversary can break the above security game with a nonnegligible advantage.

4. Concrete Scheme

In this section, we present the concrete construction of our efficient CCA2-secure flexible fine-grained access control scheme for the IoT using edge computing based on prime-order bilinear groups which perform more efficiently than composite-order bilinear groups as follows:

GlobalSetup$(1^\lambda) \longrightarrow$ (MK, GP): this algorithm is run by CA. It takes the system security parameter λ as input. It chooses two suitable multiplicative cyclic groups $\mathbb{G}$ and $\mathbb{G}_T$ with large prime-order $p \in \Theta\{2^\lambda\}$. Let g be a generator of $\mathbb{G}$ and defines a bilinear map e: $\mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$ on $\mathbb{G}$. The attribute universe is $U = \mathbb{Z}_p$. U_{AA} denotes the set of all attribute authorities. Additionally, it chooses three functions H, F, and T. H maps user identities to elements of $\mathbb{G}$. F maps attributes to elements of $\mathbb{G}$. T maps each attribute to the unique attribute authority who controls it. It picks a secure chameleon hash function H_{ch}: $\{0,1\}^* \longrightarrow U$ and runs the KeyGen$_{\text{ch}}(1^\lambda)$ algorithm to obtain a chameleon hash key pair $(\text{SK}_{\text{ch}}, \text{PK}_{\text{ch}})$. Finally, it picks two random numbers $\alpha_0, \beta_0 \in \mathbb{Z}_p^*$ and computes $e(g,g)^{\alpha_0}$ and g^{β_0}. This algorithm outputs the system master key $\text{MK} = <\alpha_0, \beta_0, \text{SK}_{\text{ch}}>$ and the global public parameters $\text{GP} = <p, g, \mathbb{G}, \mathbb{G}_T, e, e(g,g)^{\alpha_0}, g^{\beta_0}, U, U_{AA}, \text{PK}_{\text{ch}}, H, F, T, H_{\text{ch}}>$.

AASetup(GP, AID) $\longrightarrow$ $(\text{PK}_{\text{AID}}, \text{SK}_{\text{AID}})$: this algorithm is run by attribute authorities. For each authority $\text{AID} \in U_{AA}$, it chooses two $\alpha_{\text{AID}}, \beta_{\text{AID}} \in \mathbb{Z}_p^*$ as its secret key SK_{AID} and publishes the public key:

$$\text{PK}_{\text{AID}} = <e(g,g)^{\alpha_{\text{AID}}}, g^{\beta_{\text{AID}}}>. \tag{2}$$

UKeyGen(UID, AID, $a \in U$) $\longrightarrow$ $\text{SK}_{\text{UID},a}$: since the attribute a is managed by the authority $T(a)$, then this algorithm is run by the authority $T(a)$. Firstly, it chooses a random number $t_a \in \mathbb{Z}_p^*$. Then, it computes

$$\begin{aligned} K_{T(a),a,1} &= g^{\alpha_{T(a)}} H(\text{UID})^{\beta_{T(a)}} F(a)^{t_a}, \\ K_{T(a),a,2} &= g^{t_a}. \end{aligned} \tag{3}$$

The path key $P_{\text{UID},a}$ is generated by the attribute key generation algorithm.

AttrKeyGen(UID, AID, $a \in U$) $\longrightarrow$ $P_{\text{UID},a}$: since the attribute a is managed by the authority $T(a)$, this algorithm is run by the authority $T(a)$. Firstly, if the binary state tree AT_a has not been set up, it establishes an AT_a, initializes a revocation list $\text{RL}_a = \varnothing$, and then chooses a random number $\text{AK}_a \in \mathbb{Z}_p^*$ as the attribute key. Each user with the attribute a is assigned as a leaf node, and it increases the height with users increasing. For each node N_i in the tree, it randomly picks $\text{NK}_{a,i} \in \mathbb{Z}_p^*$. Path(UID, a) denotes the path of the user UID from its leaf node to the root node, and then, $P_{\text{UID},a} = \{\text{NK}_{a,i}\}_{i \in \text{Path}(\text{UID},a)}$ is the path key for the user UID which is used to recover AK_a. Finally, it sends $P_{\text{UID},a}$ to the user, publishes $\{\text{AK}_a \cdot \text{NK}_{a,i}\}_{i \in \text{MinSubCover}(\text{AT}_a, \text{RL}_a)}$, and then shares $(\text{AT}_a, \text{AK}_a)$ with CSP and ESP.

Finally, the private key of the attribute a for the data user UID is

$$\text{SK}_{\text{UID},a} = <K_{T(a),a,1}, K_{T(a),a,2}, P_{\text{UID},a}>. \tag{4}$$

CPoolGen(UID, GP, $\{\text{PK}_{\text{AID}}\}_{\text{AID} \in U_{AA}}$) $\longrightarrow$ $\text{CPool}_{\text{UID}}$: this algorithm is run by data owners. The data owner UID chooses an integer N to determine the size of the attributes will be associated with any ciphertext. $\forall j \in U_{AA}$, $\forall i \in [N]$, the algorithm picks randomly numbers $r_i, x_i, y_i, \in Z_p^*$, and computes

$$C_{1,i}' = e(g,g)^{x_i} e(g,g)^{\alpha_j r_i}, C_{2,i}' = g^{-r_i}, C_{3,i}' = g^{\beta_j r_i} g^{y_i}. \tag{5}$$

Set

$$\begin{aligned} \text{IT}_{ji} &= \left(r_i, x_i, y_i, C_{1,i}', C_{2,i}', C_{3,i}'\right), \\ \text{IT}_j &= \left\{\text{IT}_{ji}\right\}_{i \in [N]}, \end{aligned} \tag{6}$$

then the immediate ciphertext pool for the data owner UID is

$$\text{CPool}_{\text{UID}} = \left\{\text{IT}_j\right\}_{j \in U_{AA}}. \tag{7}$$

Remark 1. $\text{CPool}_{\text{UID}}$ is generated when the data owner initializes her/his client. It is saved on the data owner's side and can be reused. Moreover, it can be updated by the data owner UID whenever a new attribute authority joins the system or the data owner increases the size of attributes for improving the expressiveness.

Enoff(GP) $\longrightarrow$ CToff: this algorithm is run by data owners. It picks randomly numbers $r_0, s, \in Z_p^*$, and computes

$$\begin{aligned} K &= e(g,g)^s, \\ C_{1,0}' &= e(g,g)^s e(g,g)^{\alpha_0 r_0}, \\ C_{2,0}' &= g^{-r_0}, C_{3,0}' = g^{\beta_0 r_0}. \end{aligned} \tag{8}$$

Then, the offline ciphertext is

$$\text{CToff} = \left\{r_0, s, K, C_{1,0}', C_{2,0}', C_{3,0}'\right\}. \tag{9}$$

Remark 2. The offline ciphertext is used only one time. After it is used, the algorithm Enoff(GP) $\longrightarrow$ CToff in the client

generates a new offline ciphertext for the next plaintext message.

$\mathrm{Enon}(m, (M,\rho), \mathrm{GP}, \mathrm{CPool}_{\mathrm{UID}}, \mathrm{CToff}, \{\mathrm{PK}_{\mathrm{AID}}\}) \longrightarrow$ CT: this algorithm is run by data owners. It takes a plaintext message m, an access structure (M,ρ), and a set of authority public keys $\{\mathrm{PK}_{\mathrm{AID}}\}$ as input, where $M \in \mathbb{Z}_p^{l\times n}$ and ρ is a map from each row $\overrightarrow{M}_i$ of M to an attribute $\rho(i) \in U$. Let δ be a function maps each row $\overrightarrow{M}_i$ to the authority who manages attribute $\rho(i)$, i.e., $\delta(i) = T(\rho(i))$. For encryption, the algorithm randomly picks numbers $v_2, \ldots, v_n, w_2, \ldots, w_n \in Z_p^*$. Let $\overrightarrow{v} = (s, v_2, \ldots, v_n)^\top$ and $\overrightarrow{w} = (0, w_2, \ldots, w_n)^\top$. For $i = 1, \ldots, l$, it computes $\lambda_i = \overrightarrow{M}_i \overrightarrow{v}$, $w_i = \overrightarrow{M}_i \overrightarrow{w}$, and $w_0 = -\sum_{i=0}^{n} w_i$. Then, the ciphertext is computed as follows:

$$\begin{aligned} C_0 &= mK, \\ C_{1,0} &= C_{1,0}' = e(g,g)^s e(g,g)^{\alpha_0 r_0}, \\ C_{2,0} &= C_{2,0}' = g^{-r_0}, \\ C_{3,0} &= C_{3,0}' g^{w_0} = g^{\beta_0 r_0} g^{w_0}. \end{aligned} \tag{10}$$

For $i = 1, \ldots, l$, it randomly chooses immediate ciphertexts without repetition in $\mathrm{IT}_{\delta(i)}$ and an auxiliary parameter $r_{\mathrm{ch}} \xleftarrow{R} \mathbb{Z}_p$, then computes

$$\begin{aligned} &C_{1,i} = C_{1,i}' = e(g,g)^{x_i} e(g,g)^{\alpha_{\delta(i)} r_i}, \\ &C_{2,i} = C_{2,i}' = g^{-r_i}, C_{3,i} = C_{3,i}' = g^{\beta_{\delta(i)} r_i} g^{y_i}, \\ &C_{4,i} = F(\rho(i))^{r_i}, C_{5,i} = \lambda_i - x_i, C_{6,i} = w_i - y_i, \\ &V = H_{\mathrm{ch}}\left(\mathrm{PK}_{\mathrm{ch}}, \mathrm{PK}_{\mathrm{ch}} || C_0 || C_{1,0} || C_{1,1} || C_{5,1} || \cdots || C_{1,l} || C_{5,l}, r_{\mathrm{ch}}\right), \\ &C_{4,0} = F(V)^{r_0}. \end{aligned} \tag{11}$$

At last, the ciphertext is

$$\begin{aligned} \mathrm{CT} = \Big\{ &(M,\rho), C_0, C_{1,0}, C_{2,0}, C_{3,0}, C_{4,0}, \\ &\left\{C_{1,i}, C_{2,i}, C_{3,i}, C_{4,i}, C_{5,i}, C_{6,i}\right\}_{i=1}^{l}, r_{\mathrm{ch}} \Big\}. \end{aligned} \tag{12}$$

$\mathrm{PublicTest}(\mathrm{GP}, \{\mathrm{PK}_{\mathrm{AID}}\}, \mathrm{CT}, I) \longrightarrow$ True/False: the public test algorithm can be run by any role in the system, if she/he feels she/he needs it. Suppose the components which to be tested is

$$\left\{C_{1,i}, C_{2,i}, C_{3,i}, C_{4,i}, C_{5,i}, C_{6,i}\right\}_{i\in I}, \quad \text{where } I \subset [l]. \tag{13}$$

It firstly computes

$$V' = H_{\mathrm{ch}}\left(\mathrm{PK}_{\mathrm{ch}}, \mathrm{PK}_{\mathrm{ch}} || C_0 || C_{1,0} || C_{1,1} || C_{5,1} || \cdots || C_{1,l} || C_{5,l}, r_{\mathrm{ch}}\right), \tag{14}$$

and then

$$\begin{aligned} &e\left(g, C_{3,0} C_{4,0} \prod_{i\in I} C_{4,i} \prod_{i\in[l]} C_{3,i} g^{C_{6,i}}\right) e\left(C_{2,0}, F(V') g^{\beta_0}\right) \\ &\cdot \prod_{i\in I} e\left(C_{2,i}, F(\rho(i)) g^{\beta_{\delta(i)}}\right) \prod_{i\in[l]\setminus I} e\left(C_{2,i}, g^{\beta_{\delta(i)}}\right) = 1. \end{aligned} \tag{15}$$

If the above equation holds, it means that the ciphertext is legitimate. To be specific, it is encrypted exactly by the attribute set $\{\rho(i)\}_{i\in I}$, and the legitimate private key set with respect to the attribute set $\{\rho(i)\}_{i\in I}$ can decrypt out the correct plaintext message when $\{\rho(i)\}_{i\in I}$ is an authorized set of (M,ρ).

$\mathrm{CReEnc}(\mathrm{CT}, \{\mathrm{AK}_{\rho(i)}\}) \longrightarrow \mathrm{CT}_{\mathrm{CSP}}$: this algorithm is run by ESP. After receiving the data owner's encrypted data, it first verifies the validity of the ciphertext by using the algorithm $\mathrm{PublicTest}(\mathrm{GP}, \{\mathrm{PK}_{\mathrm{AID}}\}, \mathrm{CT}, I)$, where $I = [l]$. If the public-test algorithm outputs True, it re-encrypts the ciphertext by using the latest attribute keys as follows:

$$\begin{aligned} \mathrm{CT}_{\mathrm{CSP}} = \Big\{ &(M,\rho), C_0, C_{1,0}, C_{2,0}, C_{3,0}, C_{4,0}, \\ &\left\{C_{1,i}, C_{2,i}, C_{3,i}, \left(C_{4,i}\right)^{\mathrm{AK}_{\rho(i)}}, C_{5,i}, C_{6,i}\right\}_{i=1}^{l}, r_{\mathrm{ch}} \Big\}. \end{aligned} \tag{16}$$

Finally, ESP sends the ciphertext $\mathrm{CT}_{\mathrm{CSP}}$ to CSP.

$\mathrm{Decrypt}(\mathrm{CT}_{\mathrm{CSP}}, \mathrm{UID}, \mathrm{GP}, \{\mathrm{AK}_{\rho(i)}\}, \{\mathrm{SK}_{\mathrm{UID},a}\}) \longrightarrow m$: this algorithm is run by data users. Suppose a user UID with a set of attributes S_{UID} wants to decrypt the ciphertext $\mathrm{CT}_{\mathrm{CSP}}$. If $S_{\mathrm{UID}} \not\models (M,\rho)$, this algorithm outputs $\perp$. Otherwise, it exist a subset $\{\rho(i): i \in I \subset [l]\}$ of S_{UID} satisfy the access policy (M,ρ). Then, it calculates constants $\{c_i: i \in I\}$ such that $\sum_{i\in I} c_i \overrightarrow{M}_i = (1, 0, \ldots, 0)$. For each attribute $\rho(i) \in \{\rho(i): i \in I\}$, it recovers $\mathrm{AK}_{\delta(i)}$ by using the path keys and computes the original ciphertext $\mathrm{CT} = \mathrm{CT}_{\mathrm{CSP}}$ except that $(C_{4,i}) = (C_{4,i})^{\mathrm{AK}_{\rho(i)} \mathrm{AK}_{\rho(i)}^{-1}}$ for all $i \in I$. Then, it can verify the validity of the ciphertext by the algorithm $\mathrm{PublicTest}(\mathrm{GP}, \{\mathrm{PK}_{\mathrm{AID}}\}, \mathrm{CT}, I)$. If the public-test algorithm outputs True, it computes

$$\begin{aligned}
&\forall i \in I, \quad C_{1,i} e(g,g)^{C_{5,i}} e(K_{\delta(i),\rho(i),1}, C_{2,i}) \cdot e(H(\mathrm{UID}), C_{3,i} g^{C_{6,i}}) e(K_{\delta(i),\rho(i),2}, C_{4,i}) \\
&= e(g,g)^{x_i} e(g,g)^{\alpha_{\delta(i)} r_i} e(g,g)^{\lambda_i - x_i} \cdot e(g^{\alpha_{\delta(i)}} H(\mathrm{UID})^{\beta_{\delta(i)}} F(\rho(i))^{t_{\rho(i)}}, g^{-r_i}) \\
&\quad \cdot e(H(\mathrm{UID}), g^{\beta_{\delta(i)} r_i} g^{y_i} g^{w_i - y_i}) e(g^{t_{\rho(i)}}, F(\rho(i))^{r_i}) \\
&= e(g,g)^{\lambda_i} e(H(\mathrm{UID}),g)^{w_i}, \prod_{i \in I} \left(e(g,g)^{\lambda_i} e(H(\mathrm{UID}), g)^{w_i} \right)^{c_i} = e(g,g)^{s},
\end{aligned} \tag{17}$$

$$\frac{C_0}{e(g,g)^{s}} = \frac{m e(g,g)^{s}}{e(g,g)^{s}} = m.$$

Outsourcing decryption: the data user can choose to outsource decryption if he owns limited resource or for saving resource. This feature is implemented in three algorithms:

$\mathrm{TKGen}(\mathrm{CT}_{\mathrm{CSP}}, \mathrm{UID}, \mathrm{GP}, \{\mathrm{AK}_{\rho(i)}\}, \{\mathrm{SK}_{\mathrm{UID},a}\}) \longrightarrow (\{\mathrm{TK}_{\mathrm{UID},a}\}, \mathrm{CT})$: this algorithm is run by data users. Assuming that the subset $J = \{\rho(i): i \in [l]\}$ of S_{UID} is unrevoked. Firstly, it chooses random numbers $z \in Z_p^*$ as the retrieving key, i.e., $\mathrm{RK}_{\mathrm{UID}} = z$. Then, set the transformation key $\mathrm{TK}_{\mathrm{UID},a} = < K_{T(a),a,1}^{(1/z)}, K_{T(a),a,2}^{(1/z)} >$, for every $a \in J$. Secondly, it computes the ciphertext $\mathrm{CT} = \mathrm{CT}_{\mathrm{CSP}}$ except that $(C_{4,i}) = (C_{4,i})^{\mathrm{AK}_{\rho(i)} \mathrm{AK}_{\rho(i)}^{-1}}$ for all $i \in J$. At last, it keeps the retrieving key $\mathrm{RK}_{\mathrm{UID}} = z$ and then sends CT and $\{\mathrm{TK}_{\mathrm{UID},a}\}_{a \in J}$ to ESP.

$\mathrm{PartialDec}(\{\mathrm{TK}_{\mathrm{UID},a}\}_{\forall a \in J}, \mathrm{CT}, \mathrm{GP}, \{\mathrm{AK}_{\rho(i)}\}) \longrightarrow \mathrm{CT}_{\mathrm{out}}$: this algorithm is run by ESP. Assuming that the subset $\{\rho(i): i \in I \subset [l]\}$ of S_{UID} satisfies the access policy (M, ρ). It calculates constants $\{c_i: i \in I\}$ such that $\sum_{i \in I} c_i \overrightarrow{M}_i = (1, 0, \ldots, 0)$, and then, it computes

$$\begin{aligned}
\mathrm{CT}_1 &= \prod_{i \in I} \left(C_{1,i} e(g,g)^{C_{5,i}} e(H(\mathrm{UID}), C_{3,i} g^{C_{6,i}}) \right)^{-c_i} \\
&= \prod_{i \in I} \left(e(g,g)^{x_i} e(g,g)^{\alpha_{\delta(i)} r_i} e(g,g)^{\lambda_i - x_i} e(H(\mathrm{UID}), g^{\beta_{\delta(i)} r_i} g^{y_i} g^{w_i - y_i}) \right)^{-c_i} \\
&= \prod_{i \in I} \left(e(g,g)^{\lambda_i} e(g,g)^{\alpha_{\delta(i)} r_i} e(H(\mathrm{UID}), g^{\beta_{\delta(i)} r_i} g^{w_i}) \right)^{-c_i}, \\
\mathrm{CT}_2 &= \prod_{i \in I} \left(e(K_{\delta(i),\rho(i),1}^{(1/z)}, C_{2,i}) e(K_{\delta(i),\rho(i),2}^{(1/z)}, C_{4,i}) \right)^{-c_i} \\
&= \prod_{i \in I} \left(e(g^{(\alpha_{\delta(i)}/z)} H(\mathrm{UID})^{(\beta_{\delta(i)}/z)} F(\rho(i))^{(t_{\rho(i)}/z)}, g^{-r_i}) e(g^{(t_{\rho(i)}/z)}, F(\rho(i))^{r_i}) \right)^{-c_i} \\
&= \prod_{i \in I} e\left(g^{\alpha_{\delta(i)}} H(\mathrm{UID})^{\beta_{\delta(i)}}, g^{-r_i} \right)^{-(c_i/z)}.
\end{aligned} \tag{18}$$

Finally, it sets $\mathrm{CT}_{\mathrm{out}} = (\mathrm{CT}_1, \mathrm{CT}_2)$ and sends it to the data user.

$\mathrm{FullDec}(\mathrm{CT}_{\mathrm{out}}, \mathrm{RK}_{\mathrm{UID}}) \longrightarrow m$: this algorithm is run by data users. It computes

$$C_0 \mathrm{CT}_1 \mathrm{CT}_2^{z} = m e(g,g)^{s} e(g,g)^{-s} = m. \tag{19}$$

$\mathrm{Revoke}(\mathrm{UID}, a, \mathrm{RL}_a) \longrightarrow \mathrm{UpAK}_a$: if the authority $T(a)$ wants to revoke the attribute a from the data user UID, it adds UID into the revocation list RL_a of the attribute a and chooses a new attribute key $\mathrm{AK}_a' \xleftarrow{R} Z_p^*$. Then, it sends the update keys

$$\mathrm{UpAK}_a = \mathrm{AK}_a' \mathrm{AK}_a^{-1}, \tag{20}$$

to ESP and CSP and publishes

$$\{\mathrm{AK}_a' \cdot \mathrm{NK}_{a,i}\}_{i \in \mathrm{MinSubCover}(\mathrm{AT}_a, \mathrm{RL}_a)}. \tag{21}$$

ESP or CSP updates the ciphertext related to a using UpAK_a by calculating

$$\left(C_{4,i}^{\mathrm{AK}_{\rho(i)}}\right)^{\mathrm{UpAK}_{\rho(i)}} = \left(C_{4,i}\right)^{\mathrm{AK}_{\rho(i)}'}, \tag{22}$$

and updates the attribute key by calculating

$$\mathrm{AK}_a' = \mathrm{UpAK}_a \cdot \mathrm{AK}_a. \tag{23}$$

If the authority $T(a)$ wants to revoke the attribute a from the system, it only sets all the data users {UID} who own a into RL_a and runs the above operation.

If the system wants to revoke the data user UID from the system, it asks all the involved attribute authorities to revoke all the involved attributes from the user UID, by running the above operation.

5. Security Analysis

In this section, we prove the proposed scheme is IND-sAA-sCCA2 secure.

Lemma 1 ("zero-out" Lemma [19]). *Let $(M^* \in \mathbb{Z}_p^{l\times n^*}, \rho)$ be a linear secret sharing schemes of an access policy and $C \in [l]$ be a nonauthorized set of rows. Let $c \in \mathbb{N}$ be the dimension of the subspace spanned by the rows of $\mathbb{C}$. Then, the distribution of the shares $\{\lambda_x^*\}_{x\in[l]}$ sharing the secret $s \in \mathbb{Z}_p$ generated with the matrix M^* is the same as the distribution of the share $\{\lambda_x\}_{x\in[l]}$ sharing the same secret s generated with some matrix M, where $M_{x,j} = 0$ for all $(x, j) \in C \times [n^* - c]$.*

The main security theorem is shown as follows.

Theorem 1. *The proposed concrete scheme is IND-sAA-sCCA2 secure in the random oracle model if the $(q+1)$ – DPBDHE2 assumption holds, the chameleon hash function is secure, and the size of challenge matrix is at most $q \times q$.*

Proof. Suppose that there exists a PPT adversary $\mathscr{A}$ who can break the proposed scheme with nonnegligible advantage ε, then we can build a simulator $\mathscr{B}$ which can break the $(q+1)$ – DPBDHE2 assumption with the tuple (D, Y) as input and interact with $\mathscr{A}$ as follows:

Init: initially, $\mathscr{A}$ gives $\mathscr{B}$ a set of corrupt authorities C_θ and a challenge access policy (M^*, ρ), where M^* is a matrix whose size $l \times n$ is at most $q \times q$ and ρ: $[l] \longrightarrow \mathbb{Z}_p$ which implies that its attribute set is $S^* = \{\rho(i)\}_{i\in[l]} \triangleq \rho[l]$. Let δ be a function map in each row $\overrightarrow{M_i}$ to the authority who manages attribute $\rho(i)$, i.e., $\delta(i) = T(\rho(i))$, and denote $\{\delta(i)\}_{i\in[l]}$ by $\delta[l]$.

Global setup: $\mathscr{B}$ gets (D, Y) from $(q+1)$ – DPBDHE2 challenger and substitutes M^* with the matrix M according to "zero-out" lemma where denotes $n - c$ by n' and restricts that, for each row $M_{x\in C}$, there is only one "1" in one of the last c positions and "0" in others. It picks randomly a challenge message $m^* \xleftarrow{R} \mathbb{G}_T$ and sets $C_0 = m^* \cdot Y$. It selects a secure chameleon hash function H_{ch}: $\{0,1\}^* \longrightarrow U$ and then runs the algorithm $\mathrm{KeyGen}_{\mathrm{ch}}(1^\lambda)$ to get the key pair $(\mathrm{SK}_{\mathrm{ch}}, \mathrm{PK}_{\mathrm{ch}})$. It randomly chooses an auxiliary parameter $r_{\mathrm{ch}}^* \xleftarrow{R} \mathbb{Z}_p$ and computes $V^* = H_{\mathrm{ch}}(\mathrm{PK}_{\mathrm{ch}}, C_0^*, r_{\mathrm{ch}}^*)$. Then, V^* is regarded as a challenge on-the-fly dummy attribute authorized by θ_0 controlled by $\mathscr{B}$. It calculates a matrix $M_{l+1} \xleftarrow{R} \mathbb{Z}_p^{1\times n}$ representing the access policy of V^*; M_{l+1} is regarded as the $l+1$ row of M, and then, M becomes a $(l+1) \times n$ matrix.

$\mathscr{B}$ randomly selects $\alpha_0', \beta_0' \xleftarrow{R} \mathbb{Z}_p$ and sets

$$\alpha_0 = \alpha_0' + b_{l+1}a^{q+2}M_{l+1,1}, \beta_0 = \beta_0' + \sum_{j=2}^{n'} b_{l+1}a^{q+3-j}M_{l+1,j}. \tag{24}$$

Then, it calculates

$$e(g,g)^{\alpha_0} = e(g,g)^{\alpha_0'} e\left(g^{b_{l+1}a}, g^{a^{q+1}}\right)^{M_{l+1,j}}, g^{\beta_0} = g^{\beta_0'} \cdot \prod_{j=2}^{n'} \left(g^{b_{l+1}a^{q+3-j}}\right)^{M_{l+1,j}}. \tag{25}$$

At last, $\mathscr{B}$ sends

$$\mathrm{GP} = \langle p, g, \mathbb{G}, e, e(g,g)^{\alpha_0}, g^{\beta_0}, U, U_\theta, \mathrm{PK}_{\mathrm{ch}}, T \rangle, \tag{26}$$

to $\mathscr{A}$, where the random oracles H and F are programmed by the simulator.

Phase 1: the adversary $\mathscr{A}$ issues a polynomially bounded number of queries statically:

(i) Authority's public key queries: $\mathscr{A}$ submits a set of the noncorrupt authorities $N_\theta \subseteq U_\theta$ and $N_\theta \cap C_\theta = \varnothing$ since $\mathscr{A}$ can create the public keys of the corrupt authorities by himself. For each $\theta \in N_\theta$, $\mathscr{A}$ considers two cases:

(a) $\theta \notin \delta[l]$: $\mathscr{B}$ randomly selects $\alpha_\theta, \beta_\theta \xleftarrow{R} \mathbb{Z}_p$ and outputs the public key $\langle e(g,g)^{\alpha_\theta}, g^{\beta_\theta} \rangle$.

(b) $\theta \in \delta[l] \backslash C_\theta$: let $X = \{x | \delta t(x) n = q\theta\}$, $\mathscr{B}$ randomly selects $\alpha_\theta', \beta_\theta' \xleftarrow{R} \mathbb{Z}_p$ and sets

$$\alpha_\theta = \alpha_\theta' + \sum_{x\in X\cup\{l+1\}} b_x a^{q+2} M_{x,1}, \beta_\theta = \beta_\theta' + \sum_{x\in X\cup\{l+1\}} \sum_{j=2}^{n'} b_x a^{q+3-j} M_{x,j}. \tag{27}$$

Then, it calculates

$$e(g,g)^{\alpha_\theta} = e(g,g)^{\alpha_{\theta\prime}} \prod_{x\in X\cup\{l+1\}} e\left(g^{b_x a}, g^{a^{q+1}}\right)^{M_{x,1}}, \quad g^{\beta_\theta} = g^{\beta_{\theta\prime}} \prod_{x\in X\cup\{l+1\}} \prod_{j=2}^{n'} \left(g^{b_x a^{q+3-j}}\right)^{M_{x,j}}. \tag{28}$$

(ii) H-oracle queries: $\mathscr{A}$ queries to oracle H for identity UID_i with attribute set S_i. There are two cases:

(a) $\rho[l] \cap S_i = \varnothing$: $\mathscr{B}$ randomly selects $h_i' \overset{R}{\leftarrow} \mathbb{Z}_p$ and computes

$$H(\mathrm{UID}_i) = g^{h_i'} g^{a}, \ldots, g^{a^{n'-1}} = g^{h_i'} \prod_{k=2}^{n'} g^{a^{k-1}}. \tag{29}$$

(b) $\rho[l] \cap S_i \neq \varnothing$: for some rows $X' = \{x|\rho t(x)n \in q\rho h[l] \cap xS_i\}$ and $X_C = \{x|\delta t(x)n \in q\delta h[l] \cap xC_\theta\}$, $\mathscr{B}$ can find a vector $\overrightarrow{d}_i \in \mathbb{Z}_p^{n\times 1}$ with $d_{i,1} = 1$ such that $\overrightarrow{M}_x^{*} \overrightarrow{d}_i = 0$ and $\overrightarrow{M}_x \overrightarrow{d}_i = 0$ for all $x \in X' \cup X_C$ and then $\mathscr{B}$ randomly selects $h_i' \overset{R}{\leftarrow} \mathbb{Z}_p$ and computes

$$\begin{aligned} H(\mathrm{UID}_i) &= g^{h_i'} (g^{a})^{d_{i,2}}, \ldots, \left(g^{a^{n'-1}}\right)^{d_{i,n'}} \\ &= g^{h_i'} \prod_{k=2}^{n'} \left(g^{a^{k-1}}\right)^{d_{i,k}}. \end{aligned} \tag{30}$$

(iii) F-oracle queries: $\mathscr{A}$ queries to oracle F for the attribute u whose authority is $T(u)$. There are two cases:

(a) $T(u) \notin \delta[l]$ or $T(u) \in C_\theta$: $\mathscr{B}$ randomly selects $F(u) \overset{R}{\leftarrow} \mathbb{G}$ if it has not been stored.

(b) $T(u) \in \delta[l]$: let $X'' = \{x|\delta t(x)n = qTh(u)\} - \{x|\rho t(x)n = qu\}$ and $\mathscr{B}$ randomly selects $f_u \overset{R}{\leftarrow} \mathbb{Z}_p$ and outputs

$$F(u) = g^{f_u} \prod_{x\in X'' \cup \{l+1\}} \prod_{j\in[n']} \left(g^{b_x a^{q+2-j}}\right)^{M_{x,j}}. \tag{31}$$

(iv) User's secret keys queries: $\mathscr{A}$ submits the user identity UID_i and a set of his all attributes S_i, and $\mathscr{B}$ gives $\mathscr{A}$ the secret keys for every $u \in S_i$. Firstly, for each $u \in S_i$, $\mathscr{B}$ runs the algorithm $\mathrm{AttrKeyGen}(\mathrm{UID}, T(u), u) \longrightarrow P_{\mathrm{UID},u}$ to generate a binary tree AT_u with an attribute key AK_u and the path key $P_{\mathrm{UID},u}$. Then, it considers the following three cases:

(a) $T(u) \notin \delta[l]$: $\mathscr{B}$ randomly selects $r, \alpha_{T(u)}, \beta_{T(u)} \overset{R}{\leftarrow} \mathbb{G}$ if it has not been stored. Then, it outputs

$$\begin{aligned} &K_{T(u),u,1} = g^{\alpha_{T(u)}} H(\mathrm{UID}_i)^{\beta_{T(u)}} F(u)^{r}, \\ &K_{T(u),u,2} = g^{r}, P_{\mathrm{UID},u}. \end{aligned} \tag{32}$$

(b) $T(u) \in \delta[l]$ and $S_i \cap \rho[l] = \varnothing$: according to the authority public key queries and H-oracle and F-oracle phases,

$$\begin{aligned} &\alpha_{T(u)} = \alpha'_{T(u)} + \sum_{x\in X\cup\{l+1\}} b_x a^{q+2} M_{x,1}, \beta_{T(u)} = \beta'_{T(u)} \\ &\qquad + \sum_{x\in X\cup\{l+1\}} \sum_{j=2}^{n'} b_x a^{q+3-j} M_{x,j}, \\ &H(\mathrm{UID}_i) = g^{h_i'} \prod_{k=2}^{n'} g^{a^{k-1}}, \\ &F(u) = g^{f_u} \prod_{x\in X\cup\{l+1\}} \prod_{j\in[n']} \left(g^{b_x a^{q+2-j}}\right)^{M_{x,j}}, \end{aligned} \tag{33}$$

where $X = \{x|\delta(x) = T(u)\}$.

$\mathscr{B}$ randomly selects $t \overset{R}{\leftarrow} \mathbb{Z}_p$ sets $r = t - \sum_{k\in[n']} a^k$ and computes

$$\begin{aligned} K_{T(u),u,1} &= g^{\alpha_{T(u)}} H(\mathrm{UID}_i)^{\beta_{T(u)}} F(u)^{r} \\ &= g^{\alpha'_{T(u)}} g^{\sum_{x\in X\cup\{l+1\}} b_x a^{q+2} M_{x,1}} \left(g^{h_i'} \prod_{k=2}^{n'} g^{a^{k-1}}\right)^{\beta_{T(u)}-\beta'_{T(u)}} H(\mathrm{UID}_i)^{\beta'_{T(u)}} g^{f_u r} \prod_{x\in X\cup\{l+1\}} \prod_{j\in[n']} \left(g^{b_x a^{q+2-j}}\right)^{M_{x,j} r} \\ &= g^{\alpha'_{T(u)}} g^{\sum_{x\in X\cup\{l+1\}} b_x a^{q+2} M_{x,1}} H(\mathrm{UID}_i)^{\beta'_{T(u)}} g^{(\beta_{T(u)}-\beta'_{T(u)})h_i'} g^{\sum_{k=2} n' a^{k-1}} \sum_{x\in X\cup\{l+1\}} \prod_{j=2}^{n'} b_x a^{q+3-j} M_{x,j} \end{aligned}$$

$$
\begin{aligned}
& g^{f_u r} \prod_{x \in X \cup \{l+1\}} \prod_{j \in [n']} \left(g^{b_x a^{q+2-j}}\right)^{M_{x,j} t} g^{\Sigma_{x \in X \cup \{l+1\}} \Sigma_{j \in [n']} b_x a^{q+2-j} M_{x,j} \left(-\Sigma_{k \in [n']} a^k\right)} \\
& = g^{\alpha_{T(u)}'} H\left(\mathrm{UID}_i\right)^{\beta_{T(u)}'} g^{\left(\beta_{T(u)} - \beta_{T(u)}'\right) h_i'} g^{f_u r} \prod_{x \in X \cup \{l+1\}} \prod_{j \in [n']} \left(g^{b_x a^{q+2-j}}\right)^{M_{x,j} t} g^{\Sigma_{x \in X \cup \{l+1\}} b_x a^{q+2} M_{x,1}} \\
& g^{\Sigma_{x \in X \cup \{l+1\}} \prod_{j=2}^{n'} \prod_{k=2}^{n'} b_x a^{q+2-j+k} M_{x,j}} g^{-\Sigma_{x \in X \cup \{l+1\}} \Sigma_{j \in [n']} \Sigma_{k \in [n']} b_x a^{q+2-j+k} M_{x,j}} \\
& = g^{\alpha_{T(u)}'} H\left(\mathrm{UID}_i\right)^{\beta_{T(u)}'} g^{\left(\beta_{T(u)} - \beta_{T(u)}'\right) h_i'} g^{f_u r} \prod_{x \in X \cup \{l+1\}} \prod_{j \in [n']} \left(g^{b_x a^{q+2-j}}\right)^{M_{x,j} t} g^{-\Sigma_{x \in X \cup \{l+1\}} \prod_{j=2}^{n'} b_x a^{q+3-j} M_{x,j}} \\
& \quad \cdot g^{-\Sigma_{x \in X \cup \{l+1\}} \prod_{j=2}^{n'} b_x a^{q+1+j} M_{x,1}} \\
& = g^{\alpha_{T(u)}'} H\left(UID_i\right)^{\beta_{T(u)}'} g^{\left(\beta_{T(u)} - \beta_{T(u)}'\right) h_i'} g^{f_u r} \prod_{x \in X \cup \{l+1\}} \prod_{j \in [n]} \left(g^{b_x a^{q+2-j}}\right)^{M_{x,j} t} \prod_{x \in X \cup \{l+1\}} \prod_{j=2}^{n'} g^{b_x a^{q+3-j}\left(-M_{x,j}\right)} \\
& \quad \cdot \prod_{x \in X \cup \{l+1\}} \prod_{j=2}^{n'} g^{b_x a^{q+1+j}\left(-M_{x,1}\right)} \\
& K_{T(u),u,2} = g^r = g^t \prod_{j \in [n']} \left(g^{a^j}\right)^{-1}.
\end{aligned}
\tag{34}
$$

(c) $T(u) \in \delta[l]$ and $S_i \cap \rho[l] \neq \varnothing$: according to the authority public key queries and H-oracle and F-oracle phases,

$$
\begin{aligned}
& \alpha_{T(u)} = \alpha_{T(u)}' + \sum_{x \in X \cup \{l+1\}} b_x a^{q+2} M_{x,1}, \beta_{T(u)} = \beta_{T(u)}' + \sum_{x \in X \cup \{l+1\}} \sum_{j=2}^{n'} b_x a^{q+3-j} M_{x,j}, \\
& H\left(\mathrm{UID}_i\right) = g^{h_i'} \prod_{k=2}^{n'} g^{a^{k-1} d_{i,k}}, \\
& F(u) = g^{f_u} \prod_{x \in X'' \cup \{l+1\}} \prod_{j \in [n']} \left(g^{b_x a^{q+2-j}}\right)^{M_{x,j}},
\end{aligned}
\tag{35}
$$

where $X = \{x | \delta t(x) n = qTh(u)\}$ and $X'' = X \backslash \{x | \rho t(x) n = qu\}$.

Hence, $\forall x \in X \backslash X'' = \{x | \rho t(x) n = qu\}$, $d_{i,1} = 0$, and $\overrightarrow{M}_x \overrightarrow{d}_i = 0$. $\mathcal{B}$ randomly selects $t \xleftarrow{R} \mathbb{Z}_p$, sets $r = t - \sum_{k \in [n']} a^k d_{i,k}$, and computes

$$
\begin{aligned}
K_{T(u),u,1} &= g^{\alpha_{T(u)}} H(\mathrm{UID}_i)^{\beta_{T(u)}} F(u)^r \\
&= g^{\alpha_{T(u)'}} g^{\sum_{x\in X\cup\{l+1\}} b_x a^{q+2} M_{x,1}} \left(g^{h'_i} \prod_{k=2}^{n'} g^{a^{k-1} d_{i,k}} \right)^{\beta_{T(u)}-\beta_{T(u)'}} H(\mathrm{UID}_i)^{\beta_{T(u)'}} g^{f_u r} \prod_{x\in X''\cup\{l+1\}} \prod_{j\in[n']} \left(g^{b_x a^{q+2-j}} \right)^{M_{x,j} r} \\
&= g^{\alpha_{T(u)'}} g^{\sum_{x\in X\cup\{l+1\}} b_x a^{q+2} M_{x,1}} H(\mathrm{UID}_i)^{\beta_{T(u)'}} g^{(\beta_{T(u)}-\beta_{T(u)'}) h'_i} g^{\sum_{k=2}^{n'} a^{k-1} d_{i,k} \sum_{x\in X\cup\{l+1\}} \sum_{j=2}^{n'} b_x a^{q+3-j} M_{x,j}} \\
&\quad \cdot g^{f_u r} \prod_{x\in X''\cup\{l+1\}} \prod_{j\in[n']} \left(g^{b_x a^{q+2-j}} \right)^{M_{x,j} t} g^{\sum_{x\in X''\cup\{l+1\}} \sum_{j\in[n']} b_x a^{q+2-j} M_{x,j} \left(-\sum_{k\in[n']} a^k d_{i,k} \right)} \\
&= g^{\alpha_{T(u)'}} H(\mathrm{UID}_i)^{\beta_{T(u)'}} g^{(\beta_{T(u)}-\beta_{T(u)'}) h'_i} g^{f_u r} \prod_{x\in X''\cup\{l+1\}} \prod_{j\in[n']} \left(g^{b_x a^{q+2-j}} \right)^{M_{x,j} t} g^{\sum_{x\in X\cup\{l+1\}} b_x a^{q+2} M_{x,1}} g^{\sum_{x\in X\cup\{l+1\}} \sum_{j=2}^{n} \sum_{k=2}^{n} b_x a^{q+2-j+k} M_{x,j} d_{i,k}} \\
&\cdot g^{-\sum_{x\in X''\cup\{l+1\}} \sum_{j\in[n']} \sum_{k\in[n']} b_x a^{q+2-j+k} M_{x,j} d_{i,k}} \\
&= g^{\alpha_{T(u)'}} H(\mathrm{UID}_i)^{\beta_{T(u)'}} g^{(\beta_{T(u)}-\beta_{T(u)'}) h'_i} g^{f_u r} \prod_{x\in X''\cup\{l+1\}} \prod_{j\in[n']} \left(g^{b_x a^{q+2-j}} \right)^{M_{x,j} t} g^{x\in(X\backslash X'')\cup\{l+1\}} \sum b_x a^{q+2} M_{x,1} g^{\sum_{x\in(X\backslash X'')\cup\{l+1\}} \sum_{j=2}^{n'} \sum_{k=2}^{n'} b_x a^{q+2-j+k} M_{x,j} d_{i,k}} \\
&\cdot g^{\sum_{x\in X''\cup\{l+1\}} b_x a^{q+2} M_{x,1}} g^{\sum_{x\in X''\cup\{l+1\}} \sum_{j=2}^{n'} \sum_{k=2}^{n'} b_x a^{q+2-j+k} M_{x,j} d_{i,k}} g^{-\sum_{x\in X''\cup\{l+1\}} \sum_{j\in[n']} \sum_{k\in[n']} b_x a^{q+2-j+k} M_{x,j} d_{i,k}} \\
&= g^{\alpha_{T(u)'}} H(\mathrm{UID}_i)^{\beta_{T(u)'}} g^{(\beta_{T(u)}-\beta_{T(u)'}) h'_i} g^{f_u r} \prod_{x\in X''\cup\{l+1\}} \prod_{j\in[n']} \left(g^{b_x a^{q+2-j}} \right)^{M_{x,j} t} g^{\sum_{x\in(X\backslash X'')\cup\{l+1\}} b_x a^{q+2} M_{x,1}} g^{\sum_{x\in(X\backslash X'')\cup\{l+1\}} \sum_{j=2}^{n'} \sum_{k=2}^{n'} b_x a^{q+2-j+k} M_{x,j} d_{i,k}} \\
&\cdot g^{\sum_{x\in X''\cup\{l+1\}} b_x a^{q+2} M_{x,1}} g^{-\sum_{x\in X''\cup\{l+1\}} \sum_{k=2}^{n'} b_x a^{q+1+k} M_{x,1} d_{i,k}} g^{-\sum_{x\in X''\cup\{l+1\}} \sum_{j\in[n']} b_x a^{q+2-j} M_{x,j} d_{i,1}} \\
&= g^{\alpha_{T(u)'}} H(\mathrm{UID}_i)^{\beta_{T(u)'}} g^{(\beta_{T(u)}-\beta_{T(u)'}) h'_i} g^{f_u r} \prod_{x\in X''\cup\{l+1\}} \prod_{j\in[n']} \left(g^{b_x a^{q+2-j}} \right)^{M_{x,j} t} g^{\sum_{x\in(X\backslash X'')\cup\{l+1\}} b_x a^{q+2} M_{x,1}} g^{\sum_{x\in(X\backslash X'')\cup\{l+1\}} \sum_{j=2}^{n'} \sum_{k=2}^{n'} b_x a^{q+2-j+k} M_{x,j} d_{i,k}} \\
&\cdot g^{-\sum_{x\in X''\cup\{l+1\}} \sum_{k=2}^{n'} b_x a^{q+1+k} M_{x,1} d_{i,k}} g^{-\sum_{x\in X''\cup\{l+1\}} \sum_{j=2}^{n'} b_x a^{q+3-j} M_{x,j} d_{i,1}} \\
&= g^{\alpha_{T(u)'}} H(\mathrm{UID}_i)^{\beta_{T(u)'}} g^{(\beta_{T(u)}-\beta_{T(u)'}) h'_i} g^{f_u r} \prod_{x\in X''\cup\{l+1\}} \prod_{j\in[n']} \left(g^{b_x a^{q+2-j}} \right)^{M_{x,j} t} g^{\sum_{x\in(X\backslash X'')\cup\{l+1\}} b_x a^{q+2} M_{x,1} d_{i,1}} g^{\sum_{x\in(X\backslash X'')\cup\{l+1\}} \sum_{j=2}^{n'} b_x a^{q+2} M_{x,j} d_{i,j}} \\
&\cdot g^{\sum_{x\in(X\backslash X'')\cup\{l+1\}} \sum_{j,k=2,j\neq k}^{n',n'} b_x a^{q+2-j+k} M_{x,j} d_{i,k}} g^{-\sum_{x\in X''\cup\{l+1\}} \sum_{k=2}^{n'} b_x a^{q+1+k} M_{x,1} d_{i,k}} g^{-\sum_{x\in X''\cup\{l+1\}} \sum_{j=2}^{n'} b_x a^{q+3-j} M_{x,j} d_{i,1}} \\
&= g^{\alpha_{T(u)'}} H(\mathrm{UID}_i)^{\beta_{T(u)'}} g^{(\beta_{T(u)}-\beta_{T(u)'}) h'_i} g^{f_u r} \prod_{x\in X''\cup\{l+1\}} \prod_{j\in[n']} \left(g^{b_x a^{q+2-j}} \right)^{M_{x,j} t} \\
&\quad g^{x\in(X\backslash X'')\cup\{l+1\}} \sum \sum_{j,k=2,j\neq k}^{n',n'} b_x a^{q+2-j+k} M_{x,j} d_{i,k} g^{-\sum_{x\in X''\cup\{l+1\}} \sum_{k=2}^{n'} b_x a^{q+1+k} M_{x,1} d_{i,k}} g^{-\sum_{x\in X''\cup\{l+1\}} \sum_{j=2}^{n'} b_x a^{q+3-j} M_{x,j} d_{i,1}} \\
&= g^{\alpha_{T(u)'}} H(\mathrm{UID}_i)^{\beta_{T(u)'}} g^{(\beta_{T(u)}-\beta_{T(u)'}) h'_i} g^{f_u r} \prod_{x\in X''\cup\{l+1\}} \prod_{j\in[n']} \left(g^{b_x a^{q+2-j}} \right)^{M_{x,j} t} \prod x \in (X\backslash X'')\cup\{l+1\} \\
&\prod_{j,k=2,j\neq k}^{n',n'} g^{b_x a^{q+2-j+k} M_{x,j} d_{i,k}} \prod_{x\in X''\cup\{l+1\}} \prod_{k=2}^{n'} g^{b_x a^{q+1+k} (-M_{x,1} d_{i,k})} \prod_{x\in X''\cup\{l+1\}} \prod_{j=2}^{n'} g^{b_x a^{q+3-j} (-M_{x,j} d_{i,1})} \\
K_{T(u),u,2} &= g^r = g^t \prod_{k\in[n']} \left(g^{a^k} \right)^{-1}.
\end{aligned}
\tag{36}
$$

At last, the secret key for u is

$$
\mathrm{SK}_{\mathrm{UID}_i,u} = < K_{T(u),u,1}, K_{T(u),u,2}, P_{\mathrm{UID}_i,u} > . \tag{37}
$$

If S_i satisfies (M^*, ρ), $\mathscr{B}$ revokes one of these attributes from all users.

(d) TransformKey queries: $\mathscr{A}$ submits a user identity UID_i and a set of his all attributes S_i. For every $u \in S_i$, $\mathscr{B}$ generates the secret keys $K_{T(u),u,1}$ and $K_{T(u),u,2}$ for u as shown in the phase of user's secret key queries. Next, $\mathscr{B}$ chooses a random number $\mathscr{B}$ and then computes the transformation keys $\mathrm{TK}_{\mathrm{UID},u} = < K_{T(u),u,1}^{(1/z)}, K_{T(u),u,2}^{(1/z)} >$. At last, $\mathscr{B}$ gives $\mathscr{A}$ the transformation keys $\mathrm{TK}_{\mathrm{UID},u}$ for every $u \in S_i$.

(e) CReEncryption queries: $\mathscr{A}$ submits some ciphertext and then $\mathscr{B}$ re-encrypts the ciphertext with the latest attribute keys and returns them to $\mathscr{A}$.

(v) Decryption queries: $\mathscr{A}$ submits a ciphertext $\mathrm{CT}_{\mathrm{CSP}}$, and $\mathscr{B}$ firstly transforms $\mathrm{CT}_{\mathrm{CSP}}$ to the original ciphertext CT by the latest $\{\mathrm{AK}_{\mathrm{AID}}\}$. Then, it judges the ciphertext is valid or not by the algorithm

PublicTest(GP, $\{PK_{AID}\}$, CT). At last, $\mathcal{B}$ decrypts the ciphertext with the corresponding secret keys and returns the message if the ciphertext is legitimate.

(vi) Revocation queries: $\mathcal{A}$ queries to revoke a user UID_i with a set of attribute S_i. For every $u \in S_i$, $\mathcal{B}$ runs the algorithm $\text{Revoke}(UID_i, u, RL_u) \longrightarrow UpAK_u$ to get the update keys $UpAK_u$, re-encrypts the ciphertext with the latest update keys, and then publishes $\{AK_u' \cdot NK_{u,i}\}_{i \in \text{MinSubCover}(AT_u, RL_u)}$ to $\mathcal{A}$.

Challenge: $\mathcal{A}$ submits to $\mathcal{B}$ two equal-length messages m_0 and m_1. $\mathcal{C}$ firstly randomly flips coin $b \in \{0,1\}$ and computes $C_0 = m_b Y$, where Y may be $e(g,g)^{sa^{q+2}}$ or $R \xleftarrow{R} \mathbb{G}_T$. $\mathcal{B}$ sets $\overrightarrow{v} = (sa^{q+2}, 0, \ldots, 0) \in \mathbb{Z}_p^{1\times n}$ and $\overrightarrow{w} = (0, sa^{q+1}, \ldots, sa^{q+n'-1}, 0, \ldots, 0) \in \mathbb{Z}_p^{1\times n}$. Then, it produces the cipertexts by considering the following two cases:

$x^* \in [l]$ and $\delta(x^*) \in C_\theta$: $\forall i \in [n'], M_{x^*,i} = 0$ implies that $\lambda_{x^*} = \overrightarrow{M}_{x^*}\overrightarrow{v} = 0$ and $w_{x^*} = \overrightarrow{M}_{x^*}\overrightarrow{w} = 0$. $\mathcal{B}$ randomly selects $z_{x^*}, y_{x^*}, t_{x^*} \xleftarrow{R} \mathbb{Z}_p$ and calculates

$$\begin{aligned} &C_{1,x^*} = e(g,g)^{\lambda_{x^*}} e(g,g)^{\alpha_{\delta(x^*)} t_{x^*}} e(g,g)^{z_{x^*}} = e(g,g)^{\alpha_{\delta(x^*)} t_{x^*}} e(g,g)^{z_{x^*}+\lambda_{x^*}}, C_{2,x^*} = g^{-t_{x^*}}, \\ &C_{3,x^*} = g^{\beta_{\delta(x^*)} t_{x^*}} g^{w_{x^*}} g^{y_{x^*}}, C_{4,x^*} = F(\rho(x^*))^{t_{x^*}}, C_{5,x^*} = -z_{x^*}, C_{6,x^*} = -y_{x^*}. \end{aligned} \tag{38}$$

$x^* \in [l]$ and $\delta(x^*) \notin C_\theta$: $\lambda_{x^*} = sa^{q+2} M_{x^*,1}$ and $w_{x^*} = \sum_{j=2}^{n'} sa^{q+3-j} M_{x^*,j}$. $\mathcal{B}$ sets $t_{x^*} = -s/b_{x^*}$ and $t_0 = -s/b_{l+1}$. Then, it randomly selects $z_{x^*}, y_{x^*} \xleftarrow{R} \mathbb{Z}_p$ and computes

$$\begin{aligned} C_{1,x^*} &= e(g,g)^{\lambda_{x^*}} e(g,g)^{\alpha_{\delta(x^*)} t_{x^*}} e(g,g)^{z_{x^*}} = e(g,g)^{sa^{q+2} M_{x^*,1}} e(g,g)^{-\sum_{x \in X \cup \{l+1\}} (sb_x a^{q+2} M_{x^*,1})/b_x} e(g,g)^{z_{x^*}} \\ &= e(g,g)^{-\sum_{x \in X \cup \{l+1\} \setminus \{x^*\}} (sb_x a^{q+2} M_{x^*,1})/b_x} e(g,g)^{z_{x^*}}, \\ C_{2,x^*} &= g^{-t_{x^*}} = g^{s/b_{x^*}}, \\ C_{3,x^*} &= g^{\beta_{\delta(x^*)} t_{x^*}} g^{w_{x^*}} g^{y_{x^*}} = g^{-\sum_{x \in X \cup \{l+1\}} \sum_{j=2}^{n'} (sb_x a^{q+3-j} M_{x^*,j})/b_{x^*}} g^{\sum_{j=2}^{n'} sa^{q+3-j} M_{x^*,j}} g^{y_{x^*}} \\ &= g^{-\sum_{x \in X \cup \{l+1\} \setminus \{x^*\}} \sum_{j=2}^{n'} (sb_x a^{q+3-j} M_{x^*,j})/b_{x^*}} g^{y_{x^*}} = \prod_{x \in X \cup \{l+1\} \setminus \{x^*\}} \prod_{j=2}^{n'} \left(g^{sb_x a^{q+3-j}/b_{x^*}} \right)^{-M_{x^*,j}} g^{y_{x^*}}, \\ C_{4,x^*} &= F(\rho(x^*))^{t_{x^*}} = \prod_{x \in X'' \cup \{l+1\}} \prod_{j \in [n']} \left(g^{sb_x a^{q+2-j}/b_{x^*}} \right)^{-M_{x^*,j}}, \\ C_{5,x^*} &= -z_{x^*}, C_{6,x^*} = -y_{x^*}, \\ C_{1,0} &= Y e(g,g)^{\alpha_0 t_0}, C_{2,0} = g^{-t_0} = g^{s/b_{l+1}}, \\ C_{3,0} &= g^{\beta_0 t_0} g^{w_0} = g^{\beta_0 t_0} g^{-\sum_{x^* \in [l]} w_{x^*}}, C_{4,0} = F(V^*)^{t_0}, \\ \text{CT} &= \left\{ (M^*, \rho), C_0, C_{1,0}, C_{2,0}, C_{3,0}, C_{4,0}, \left\{ C_{1,x^*}, C_{2,x^*}, C_{3,x^*}, C_{4,x^*}, C_{5,x^*}, C_{6,x^*} \right\}_{x^* \in [l]} \right\}. \end{aligned} \tag{39}$$

Since $\{t_{x^*}\}_{x^*=0}^{l}$ are not properly distributed, $\mathcal{B}$ randomly selects $s', t_{x^*}', t_0' \xleftarrow{R} \mathbb{Z}_p$ and random vectors $\overrightarrow{v}' = (s', v_2, \ldots, v_n)$, $\overrightarrow{w}' = (0, w_2, \ldots, w_n) \in \mathbb{Z}_p^{1\times n}$. Then, it computes the re-randomized ciphertext CT*, $\forall x^* \in [l]$:

Table 2: Comparison of properties with previous works.

Schemes	Multiauthority	Public test	Revocation	Online/offline encryption	Outsource decryption	Security
RW15 [19]	√	×	×	×	×	Statically secure
LJWY18 [39]	√	×	√	×	√	Statically secure
LLWG19 [40]	√	√	√	×	×	IND-sAA-sCCA2 secure
Ours	√	√	√	√	√	IND-sAA-sCCA2 secure

$$
\begin{aligned}
&C_0^* = C_0 e(g,g)^{s'}, C_{1,0}^* = C_{1,0} e(g,g)^{s'+\alpha_0 t_{0'}}, C_{2,0}^* = C_{2,0} g^{-t_{0'}},\\
&C_{3,0}^* = C_{3,0} g^{\beta_0 t_{0'}} g^{\beta_0 t_0} g^{-\sum_{x^* \in [l]} w_{x^*}'},\\
&C_{4,0}^* = C_{4,0} F\left(V^*\right)^{t_{0'}}, C_{1,x^*}^* = C_{1,x^*} e(g,g)^{\overrightarrow{M}_{x^*} \overrightarrow{v}' + \alpha_{\rho x^*} t_{x^*}},\\
&C_{2,x^*}^* = C_{2,x^*} g^{-t_{x^*}'}, C_{3,x^*}^* = C_{3,x^*} g^{\beta_{\rho x^*} t_{x^*}' + \overrightarrow{M}_{x^*} w_{x^*}'},\\
&C_{4,x^*}^* = C_{4,x^*} F\left(\rho\left(x^*\right)\right)^{t_{x^*}'},\\
&C_{5,x^*}^* = C_{5,x^*} = -z_{x^*}, C_{6,x^*}^* = C_{6,x^*} = -y_{x^*}.
\end{aligned}
\tag{40}
$$

$\mathcal{B}$ runs $\mathrm{TCollision}_{\mathrm{ch}}(\mathrm{SK}_{\mathrm{ch}}, C_0^*, r_{\mathrm{ch}}^*, \mathrm{PK}_{\mathrm{ch}} || C_0^* || C_{1,0}^* || C_{1,1}^* || C_{5,1}^* || \cdots || C_{1,l}^* || C_{5,l}^*)$ to get the r_b^* such that $V^* = H_{\mathrm{ch}}(\mathrm{PK}_{\mathrm{ch}}, \mathrm{PK}_{\mathrm{ch}} || C_0^* || C_{1,0}^* || C_{1,1}^* || C_{5,1}^* || \cdots || C_{1,l}^* || C_{5,l}^*, r_b^*)$. At last, it re-encryptes the ciphertext CT^* by the latest attribute keys and sends to $\mathcal{A}$.

Phase 2: the same as Phase 1, except that $\mathcal{A}$ cannot make the secret key query for the selected access policy or make decryption query for the challenge ciphertext.

Guess: $\mathcal{A}$ outputs a guess bit $b' \in \{0,1\}$. If $b' = b$, then $\mathcal{B}$ outputs that $Y = e(g,g)^{sa^{q+2}}$. Otherwise, $\mathcal{B}$ outputs that $Y = R$.

If $Y = e(g,g)^{sa^{q+2}}$, then $\mathcal{B}$ plays the proper security game with $\mathcal{A}$, since CT^* is a valid ciphertext for the message m_b. Denote the advantage of $\mathcal{A}$ wins in this case by ε. Otherwise, $Y = R$, CT^* is a ciphertext of a random message, so the advantage of $\mathcal{A}$ is 0. Therefore, $\mathcal{A}$ has advantage $\mathrm{Adv}_{\mathcal{A}}(\lambda) = \varepsilon$ in breaking the concrete scheme, and then, $\mathcal{B}$ can break the $(q+1)$-DPBDHE2 assumption with advantage $\mathrm{Adv}_{\mathcal{A}}(\lambda) = \varepsilon$. □

6. Performance Analysis

In this section, we analyze and compare the proposed scheme with others in terms of characteristics and efficiency both in the theoretical method and in the experimental method.

6.1. Theoretical Analysis. We compare the proposed scheme with several related MA-CP-ABE schemes in the terms of the feature, the computation, and the storage, respectively, in Tables 2–4. Table 5 summarizes notations.

In Table 2, we compare the properties of our scheme with other previous schemes. All these MA-CP-ABE access control schemes are constructed on the prime-order bilinear group and support the large universe. We can observe that our scheme is superior to other existing previous relevant schemes. The schemes in [19, 39] cannot filter illegal or invalid ciphertext because they do not support the ciphertext public test. The schemes in [19, 40] deal with the scenario that dynamically change the users' access privilege since they support users' attribute revocation. They also cannot support outsourcing decryption to reduce the computation overhead on the users' side. In the other schemes, the data owners will face a heavy computation load when encrypting the data. Only our scheme provides the online/offline encryption algorithm. Moreover, ours is CCA2-secure which is more secure than the other statically secure schemes.

In Table 3, we compare the computational complexity of our scheme with the previous schemes. As for the computation efficiency of the encryption, the proposed scheme is much more efficient than the other schemes since the expensive encryption operations have been executed in the user's initialization phase by adding the reusable ciphertext pool and split to online encryption and offline encryption. In comparison with the scheme in [39], the computation complexity of final decryption (the last step to recover plaintext for the user) is substantially reduced in our scheme. Moreover, our scheme is more efficient than the scheme in [40] when publicly verifying the ciphertext.

In Table 4, we compare the storage overhead of these above schemes on every entity. The main storage overhead of each AA comes from the master secret key is the same

Table 3: Comparison of computation cost.

Schemes	Offline encryption	Online encryption	Public ciphertext test	User decryption	User decryption (outsource decryption model)
RW15 [19]	×	$lF + (2l+1)E_T + 4lE + (l+1)M_T + lM$	×	$3\text{IP} + \text{IE}_T + 4\text{IM}_T + H$	×
LJWY18 [39]	×	$lF + (2l+1)E_T + 4lE + (l+1)M_T + lM$	×	$3\text{IP} + \text{IE}_T + \text{IE} + 4\text{IM}_T + H$	$(I+1)E_T + 3\text{IE} + M_T$
LLWG19 [40]	×	$(l+1)F + (2l+2)E_T + (4l+4)E + (l+2)M_T + (l+1)M$	$(4l+1)P + (2l+3)M_T + H_{\text{ch}}$	$3\text{IP} + \text{IE}_T + \text{IE} + 4\text{IM}_T + H$	×
Ours	$2E_T + 2E + M_T$	$lF + lE + M_T + M + H_{\text{ch}}$	$(l+2)P + lE + (l+2)M_T + (2I+l+4)M + H_{\text{ch}}$	$3\text{IP} + 2\text{IE}_T + 2\text{IE} + 5\text{IM}_T + H$	$E_T + 2\text{IE} + 2M_T$

Table 4: Comparison of storage overhead.

Schemes	AA's MK	AA's PK	User's private key	CTpool	Offline ciphertext	Online ciphertext
RW15 [19]	$2l_{Z_p}$	$l_{G_T} + l_G$	$2\|S\|l_G$	×	×	$(l+1)l_{G_T} + 3ll_G$
LJWY18 [39]	$2l_{Z_p}$	$l_{G_T} + l_G$	$2\|S\|l_G + \|S\|P_a l_{Z_p}$	×	×	$(l+1)l_{G_T} + 3ll_G$
LLWG19 [40]	$2l_{Z_p}$	$l_{G_T} + l_G$	$2\|S\|l_G + \|S\|P_a l_{Z_p}$	×	×	$(l+2)l_{G_T} + (3l+3)l_G + l_{Z_p}$
Ours	$2l_{Z_p}$	$l_{G_T} + l_G$	$2\|S\|l_G + \|S\|P_a l_{Z_p}$	$Nl_{G_T} + 2Nl_G + 3Nl_{Z_p}$	$2l_{G_T} + 2l_G + 2l_{Z_p}$	$(l+2)l_{G_T} + (3l+3)l_G + 3l_{Z_p}$

Table 5: Notations in performance analysis.

Notation	Description
M/M_T	One multiplication operation in the group $\mathbb{G}/\mathbb{G}_T$
E/E_T	One exponentiation operation on $\mathbb{G}/\mathbb{G}_T$
P	One pairing operation
H	One hash operation
H_{ch}	One chameleon hash operation
I	Number of attributes required for the decryption
N_A	Number of attributes authorities
l	Complexity of the access structure
S	Set of the attributes in user's private key
N	Size of ciphertext pool
P_a	Size of the path key for attribute a
l_{Z_p}	Size of an element in $\mathbb{Z}_p$
l_G	Size of an element in $\mathbb{G}$
l_{G_T}	Size of an element in $\mathbb{G}_T$

constant in every scheme. The data owner's storage overhead comes from the global public parameters and authority public keys. In our scheme, the data owner needs to shore the reusable ciphertext pool additionally. The data user's storage overhead mainly comes from the attribute-related secret keys. In these schemes [39, 40], and our scheme, the data user needs to store the path key since they using the same revocation method.

6.2. Experimental Analysis. To evaluate the performance of our proposed scheme, we implemented and compared the computation cost of the RW15 scheme [19], the LLWG19 scheme [40], and ours in Charm [44] from the Stanford Pairing-Based Crypto library [45]. The program was running on a supersingular symmetric elliptic curve group ("SS512"), over 512-bit base field size. The curve group has a 160-bit order. All the experiments were conducted on a virtual machine platform: Vmware@Workstation 15 Pro 15.5.2 build-15785246, equipped with a 2.30 GHz Intel Cored CPU with 2 GB RAM running 64-bit Linux Ubuntu 18.04.4. We run the schemes for 100 rounds and calculate the average execution time they took, where the number of attributes is ranging from 1 to 50. Finally, the graphs are shown in Figure 3.

As shown in Figure 3, the encryption cost, the decryption cost, and the public ciphertext test cost grow linearly with the number of attributes in the access structure. From Figure 3(a), in the proposed scheme, the encryption is split into online encryption and offline encryption. The online encryption time grows with the number of attributes since the online encryption has to compute the hash operation and exponentiation operation related to attributes. However, the online encryption time can be nearly haft reduced. In Figure 3(b), the predecryption denotes the data user recovers the original ciphertext by the attribute keys when she/he chooses outsourcing decryption, so its time grows with the number of attributes. Moreover, the final decryption only requires a constant amount of computation, so its time is nearly constant. Taken together, the total decryption time in our proposed scheme is nearly a quarter of it in the RW15 scheme. Focusing on the public ciphertext test cost, Figure 3(c) shows that the public ciphertext test time in the proposed scheme is less than that in the LLWG19 scheme. Therefore, the proposed scheme is very efficient in terms of the encryption cost, the decryption cost, and the public ciphertext test cost.

In conclusion, our proposal is more flexible and supports more functions than other previous schemes, which is more suitable for the IoT.

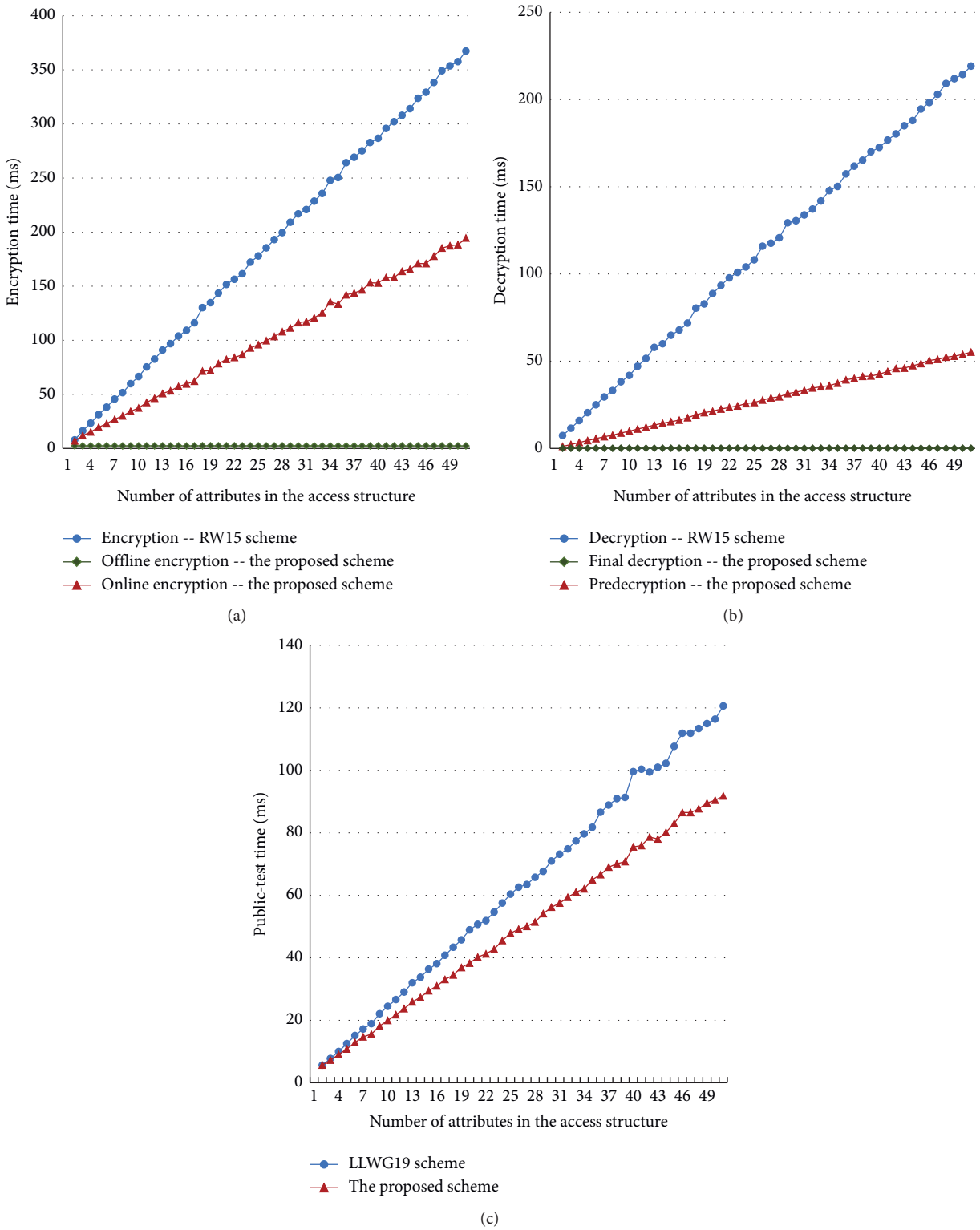

Figure 3: Experimental performance comparisons: (a) encryption cost; (b) decryption cost; (c) public-test cost.

7. Conclusion

In this paper, we design an efficient, CCA2-secure, flexible fine-grained access control scheme to solve the practical issues efficiently for the IoT using edge computing. The proposed scheme meets the need of the large-scale multi-domain collaboration in the IoT by using multiauthority attribute-based encryption with the large attribute universe.

By adding a reusable ciphertext pool, the most expensive encryption operations have been executed in the user's initialization phase which minimizes the online encryption computation. At the same time, the near-edge servers can perform the major decryption for the data user. As a result, the computation overhead of encryption and decryption is substantially reduced both on the data owner and user sides. A fine-grained revocation mechanism is designed to revoke an attribute from the user, by which the user or the attribute can be revoked from the system. So it can change user's access privileges timely and be suitable for the dynamic IoT. The scheme supports efficient ciphertext verification. Only valid ciphertext can be stored and transmitted, which saves the system resources. Moreover, the user can verify the validity of the ciphertext even though some attributes involved in the ciphertext are revoked, only if the rest of his attributes still has the access privilege to decrypt the ciphertext. The proposed scheme is proven CCA2-secure under the q-DPBDHE2 assumption, which is more secure than the previous scheme. The performance analysis results show that the proposed scheme is highly efficient and suitable in edge computing for the IoT.

Data Availability

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

Conflicts of Interest

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

Acknowledgments

This research was funded by the Program for Young Innovative Talents in Higher Education of Guangdong, China (Grant no. 2019GKQNCX004), Scientific Research Foundation of Dongguan Polytechnic (Grant no. 2019a21), and National Natural Science Foundation of China (Grant no. 61902164).

References

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): a vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[2] S. Madakam, R. Ramaswamy, and S. Tripathi, "Internet of things (IoT): a literature review," *Journal of Computer and Communications*, vol. 3, no. 5, pp. 164–173, 2015.

[3] Cisco, "Cisco Global Cloud Index: Forecast and Methodology, 2016–2021," vol. 1, 2018 White paper, Cisco Public, San Francisco, CA, USA, .

[4] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing (MCC'12)*, pp. 13–16, Helsinki, Finland, August 2012.

[5] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog computing: a platform for Internet of Things and analytics," *Big Data and Internet of Things: A Roadmap for Smart Environments*, Springer, Cham, Switzerland, 2014.

[6] S. Yi, C. Li, and Q. Li, "A survey of fog computing: concepts, applications and issues," in *Proceedings of the 2015 Workshop on Mobile Big Data (Mobidata '15)*, pp. 37–42, Hangzhou, China, June 2015.

[7] S. N. Shirazi, A. Gouglidis, A. Farshad, and D. Hutchison, "The extended cloud: review and analysis of mobile edge computing and fog from a security and resilience perspective," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2586–2595, 2017.

[8] E. Ahmed and M. H. Rehmani, "Mobile edge computing: opportunities, solutions, and challenges," *Future Generation Computer Systems*, vol. 70, pp. 59–63, 2017.

[9] W. Yu, F. Liang, X. He et al., "A survey on the edge computing for the Internet of Things," *IEEE Access*, vol. 6, pp. 6900–6919, 2017.

[10] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *EUROCRYPT 2005: Advances in Cryptology-EUROCRYPT*, pp. 457–473, Springer, Cham, Switzerland, 2005.

[11] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS '06)*, pp. 89–98, Alexandria, VA, USA, , 2006.

[12] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proceedings of the 2007 IEEE Symposium on Security & Privacy (SP'07)*, pp. 321–334, Oakland, CA, USA, May 2007.

[13] V. Goyal, A. Jain, O. Pandey, and A. Sahai, "Bounded ciphertext policy attribute based encryption," in *Proceedings of the ICALP: International Colloquium on Automata, Languages, and Programming*, pp. 579–591, Reykjavik, Iceland, July 2008.

[14] Y. Rouselakis and B. Waters, "Practical constructions and new proof methods for large universe attribute-based encryption," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security (CCS'13)*, pp. 463–474, Berlin, Germany, November 2013.

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

[16] M. Chase and S. S. Chow, "Improving privacy and security in multi-authority attribute-based encryption," in *Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS'09)*, pp. 121–130, Chicago, IL, USA, 2009.

[17] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Proceedings of the Advances in Cryptology (EUROCRYPT 2011)*, pp. 568–588, RI, USA, March 2011.

[18] Z. Liu, Z. Cao, Q. Huang, D. S. Wong, and T. H. Yuen, "Fully secure multi-authority ciphertext-policy attribute-based encryption without random oracles," in *Proceedings of the European Symposium on Research in Computer Security (ESORICS 2011)*, pp. 278–297, Leuven, Belgium, September 2011.

[19] 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 (FC 2015)*, pp. 315–332, San Juan, Puerto Ric, 2015.

[20] A. Boldyreva, V. Goyal, and V. Kumar, "Identity-based encryption with efficient revocation," in *Proceedings of the 15th*

ACM Conference on Computer and Communications Security (CCS '08), pp. 417–426, Alexandria, VA, USA, October 2008.

[21] N. Attrapadung and H. Imai, "Conjunctive broadcast and attribute-based encryption," in *Proceedings of the International Conference on Pairing-Based Cryptography (Pairing 2009)*, pp. 248–265, Palo Alto, CA, USA, 2009.

[22] N. Attrapadung and H. Imai, "Attribute-based encryption supporting direct/indirect revocation modes," in *Proceedings of the IMA International Conference on Cryptography and Coding (IMACC 2009)*, pp. 278–300, Cirencester, UK, December 2009.

[23] D. Naor, M. Naor, and J. Lotspiech, "Revocation and tracing schemes for stateless receivers," in *Proceedings of the Advances in Cryptology (CRYPTO 2001)*, pp. 41–62, Santa Barbara, CA, USA, August 2001.

[24] J. Hur and D. K. Noh, "Attribute-based access control with efficient revocation in data outsourcing systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 7, pp. 1214–1221, 2010.

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

[26] K. Yang, X. Jia, K. Ren, B. Zhang, and R. Xie, "DAC-MACS: effective data access control for multiauthority cloud storage systems," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 11, pp. 1790–1801, 2013.

[27] J. Li, W. Yao, Y. Zhang, H. Qian, and J. Han, "Flexible and fine-grained attribute-based data storage in cloud computing," *IEEE Transactions on Services Computing*, vol. 10, no. 5, pp. 785–796, 2017.

[28] G. Premsankar, M. Di Francesco, and T. Taleb, "Edge computing for the Internet of Things: a case study," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1275–1284, 2018.

[29] M. Green, S. Hohenberger, and B. Waters, "Outsourcing the decryption of abe ciphertexts," in *Proceedings of the USENIX Security Symposium*, San Francisco, CA, USA, August 2011.

[30] J. Lai, R. H. Deng, C. Guan, and J. Weng, "Attribute-based encryption with verifiable outsourced decryption," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 8, pp. 1343–1354, 2013.

[31] J. Li, X. Huang, J. Li, X. Chen, and Y. Xiang, "Securely outsourcing attribute-based encryption with checkability," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 8, pp. 2201–2210, 2013.

[32] S. Lin, R. Zhang, H. Ma, and M. Wang, "Revisiting attribute-based encryption with verifiable outsourced decryption," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 10, pp. 2119–2130, 2015.

[33] X. Mao, J. Lai, Q. Mei, K. Chen, and J. Weng, "Generic and efficient constructions of attribute-based encryption with verifiable outsourced decryption," *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 5, pp. 533–546, 2015.

[34] B. Qin, R. H. Deng, S. Liu, and S. Ma, "Attribute-based encryption with efficient verifiable outsourced decryption," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 7, pp. 1384–1393, 2015.

[35] S. Hohenberger and B. Waters, "Online/offline attribute-based encryption," in *Proceedings of the International Workshop on Public Key Cryptography (PKC2014)*, pp. 293–310, Buenos Aires, Argentina, March 2014.

[36] J. Li, Y. Zhang, X. Chen, and Y. Xiang, "Secure attribute-based data sharing for resource-limited users in cloud computing," *Computers & Security*, vol. 72, pp. 1–12, 2018.

[37] W. Liu, J. Liu, Q. Wu, B. Qin, and Y. Zhou, "Practical direct chosen ciphertext secure key-policy attribute-based encryption with public ciphertext test," in *Proceedings of the ESORICS 2014: European Symposium on Research in Computer Security*, pp. 91–108, Wroclaw, Poland, September 2014.

[38] S. J. De and S. Ruj, "Efficient decentralized attribute based access control for mobile clouds," *IEEE Transactions on Cloud Computing*, vol. 8, no. 1, pp. 124–137, 2017.

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

[40] D. Li, J. Liu, Q. Wu, and Z. Guan, "Efficient CCA2 secure flexible and publicly-verifiable fine-grained access control in fog computing," *IEEE Access*, vol. 7, pp. 11688–11697, 2019.

[41] B. Waters, "Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization," in *Proceedings of the International Workshop on Public Key Cryptography (PKC 2011)*, pp. 53–70, Taormina, Italy, March 2011.

[42] A. Beimel, *Secure Schemes for Secret Sharing and Key Distribution*, Technion-Israel Institute of Technology, Faculty of Computer Science, Haifa, Israel, 1996.

[43] H. Krawczyk and T. Rabin, "Chameleon hashing and signatures," in *Proceedings of the Network and Distributed System Security Symposium (NDSS 2000)*, vol. 8, no. 1, pp. 124–137, 2000.

[44] 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.

[45] B. Lynn, *The Pairing-Based Cryptography Library*, https://crypto.stanford.edu/pbc/, 2020.