

Received March 10, 2021, accepted March 29, 2021, date of publication April 5, 2021, date of current version April 13, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3070907

Secure Efficient Revocable Large Universe Multi-Authority Attribute-Based Encryption for Cloud-Aided IoT

KAIQING HUANG 

Department of Basic Courses, Dongguan Polytechnic, Dongguan 523808, China
School of Mathematical Sciences, South China Normal University, Guangzhou 510631, China
e-mail: kqhuang@m.scnu.edu.cn

This work was supported in part by the Project for Young Innovative Talents in 2019 of Education Department of Guangdong Province under Grant 2019GKQNCX004, in part by the Scientific Research Foundation of Dongguan Polytechnic under Grant 2019a21, and in part by the Key Scientific Research Platforms and Projects of Colleges and Universities in 2020 of Education Department of Guangdong Province under Grant 2020ZDZX3109.

ABSTRACT With the help of cloud computing, the ubiquitous and diversified Internet of things (IoT) has greatly improved human society. Revocable multi-authority attribute-based encryption (MA-ABE) is considered a promising technique to solve the security challenges on data access control in the dynamic IoT since it can achieve dynamic access control over the encrypted data. However, on the one hand, the existing revocable large universe MA-ABE suffers the collusion attack launched by revoked users and non-revoked users. On the other hand, the user collusion avoidance revocable MA-ABE schemes do not support large attributes (or users) universe, i.e. the flexible number of attributes (or users). In this article, the author proposes an efficient revocable large universe MA-ABE based on prime order bilinear groups. The proposed scheme supports user-attribute revocation, i.e., the revoked user only loses one or more attributes, and she/he can access the data so long as her/his remaining attributes satisfy the access policy. It is static security in the random oracle model under the q -DPBDHE2 assumption. Moreover, it is secure against the collusion attack launched by revoked users and non-revoked users. Meanwhile, it meets the requirements of forward and backward security. The limited-resource users can choose outsourcing decryption to save resources. The performance analysis results indicate that it is suitable for large-scale cross-domain collaboration in the dynamic cloud-aided IoT.


INDEX TERMS Cloud-aided IoT, access control, attribute-based encryption, outsource decryption, user-attribute revocation, collusion attack.

I. INTRODUCTION

The Internet of Things (IoT), connects many objects across different areas by ubiquitous enabling device technologies such as near field communication (NFC) devices, RFID tags and readers, embedded sensors, and actuator nodes [1]. It collects and shares information across different platforms, and brings intelligent service with the help of technologies such as cloud computing and artificial intelligence. Since the users' data is mostly privacy-sensitive. It faces severe challenges regarding security, privacy, and access control.

Encryption is a good way to protect data security. However, it's difficult for traditional technology to achieve access

control on the encrypted data. Luckily, in 2005, Sahai and Waters proposed a new cryptographic primitive: attribute-based encryption (ABE) [2]. Specifically, ABE is distinguished into key-policy ABE (KP-ABE) and ciphertext-policy ABE (CP-ABE) [3]. In KP-ABE, secret keys are associated with access policies, and the message is encrypted with attributes, while in CP-ABE the message is encrypted with an access policy, and secret keys are generated with attributes. A user can decrypt the ciphertext if and only if his attributes (i.e. access privileges) satisfy the ciphertext access structure. Goyal *et al.* give the first construction of KP-ABE in 2006 [3]. Unfortunately, these schemes do not address resistance to collusion attacks. In 2007, Bethencourt *et al.* presented a secure CP-ABE construction against collusion attacks [4].

The associate editor coordinating the review of this manuscript and approving it for publication was Mansoor Ahmed .

Single-authority ABE has the bottlenecks in practical application. Specifically, in the IoT, the data are shared across different domains and organizations, which implies that the users' attributes are issued by different authorities in the same system. To deal with this scenario, Chase proposed the first multi-authority ABE (MA-ABE), which supports multiple attribute authorities in 2007 [5]. In 2011, Lewko and Waters proposed the first decentralized MA-CP-ABE system by using the dual system encryption methodology based on the composite order group [6]. Different from the general MA-ABE system in which some secrets in the ciphertext or in the secret key are embedded in the global public parameters. No secret is embedded in the global public parameters in the decentralized MA-ABE system and the entity who sets up the system does not need to distribute or keep any secret key.

Meanwhile, the dynamic capacity expansion of attributes in the system is desirable for MA-ABE used in the dynamic IoT. So 'large universe' construction is more practical than 'small universe' construction. In small universe ABE, the attributes are fixed and enumerated at the system setup. After that, the requirement of change or add even only one attribute will result in rebuilding the system and possibly re-encrypting all the data. Conversely, in large universe ABE, 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. In 2015, Rouselakis and Waters constructed a large-universe MA-CP-ABE scheme in prime order bilinear groups, which is more efficient than composite order groups [7]. Now, large universe MA-ABE is considered as a promising technique to achieve fine-grained access control on the encrypted data in large-scale cross-domain applications for the IoT.

A lot of research work has been done by many scholars devoted to KP-ABE, CP-ABE, and MA-ABE in different aspects such as hidden policy, accountability, outsourced decryption, and revocation mechanism [8], [9]. For example, Li *et al.* constructed a white-box traceable CP-ABE scheme with hiding ciphertext-policy, which guarantees the privacy of the users, and accountability to address the user key abuse recently [10]. Liao *et al.* also put forward a new and secure approach to reduce the total overhead of the cloud server when many users satisfying an access policy require the outsourced decryptions for the same ciphertext besides decreasing the decryption computation cost for users [11].

Since users' access privilege will be changed dynamically in the dynamic IoT, revocation is a significant problem that needs to be properly addressed [8], [9].

There are three kinds of revocation. In attribute revocation schemes, an attribute is removed from the system, and all the users who have the attribute will lose this attribute. Moreover, the system cannot revoke one or some users. In user revocation schemes, a user is removed from the system and loses all her/his attributes. Moreover, the system cannot revoke one or some attributes. However, in user-attribute revocation schemes, the revoked user only loses one or more attributes, and she/he can access the data so long as her/his remaining

attributes satisfy the access policy. Moreover, the system can achieve attribute revocation and user revocation by using user-attribute revocation. Therefore, user-attribute revocation is granular revocation, which is more practical in the IoT.

There are two security requirements for the user-attribute revocation: forward security and backward security. Forward security means that the user who is revoked one or some attributes cannot access the data that was previously accessible for them with the revoked attributes if her/his remaining attributes do not satisfy the access policy. Backward security means that the user who is revoked one or some attributes should be prevented from accessing the subsequent data by using the revoked attributes.

In this study, the author will try to achieve user-attribute revocation in the large universe MA-ABE for the IoT under the premise of meeting the above requirements.

A. RELATED WORK

There was much work devoted to revocable single-authority ABE.

In 2008, Boldyreva *et al.* proposed the first revocable KP-ABE construction from [3]. It sets that one child of the root node of the decryption key is a time slot (regarded as an attribute), the other one is the root node of the access tree. The sender encrypts with the present time slot. The authority revokes users indirectly by periodically announces the key update material so that only non-revoked users can update their key and decrypt ciphertexts [12].

In 2009, Attrapadung and Imai proposed the first directly revocable ABE scheme. The sender encrypts messages with the revocation list directly and only non-revoked users can decrypt ciphertexts [13]. Non-revoked user does not need to update the secret key periodically. But it should publish a user list in the public parameters, and does not support the flexible number of users, which makes it not suitable for the large-scale dynamic IoT.

At the same time, they 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 [14]. In indirect revocation mode, the authority uses the subset-cover revocation framework [15] to manage users and broadcasts the key update material for non-revoked users after one user is revoked. However, it does not support attribute revocation. Moreover, it faces a forward security problem that the revoked users can access the old data they were authorized to access before being revoked, and a collusion attack that the revoked users can decrypt the data cooperating with the non-revoked users after being revoked, which implies that it also faces the backward security problem.

In 2011, based on the subset-cover revocation framework [15], Hur *et al.* proposed an access control mechanism using CP-ABE to enforce access control policies with efficient user-attribute revocation capability [16]. However, Li *et al.* pointed out it does not resist collusion attacks launched by revoked users and non-revoked users [17], which implies that it faces the forward and backward

security problems, and give a concrete instance to clarify the attack [18]. Moreover, Li *et al.* provided a CP-ABE scheme with user revocation by introducing the concept of the user group. It resists collusion attack but does not support attribute revocation [17]. Meanwhile, Li, Yao, and Han presented a user collusion avoidance CP-ABE scheme with attribute revocation by exploiting the concept of an attribute group, but it does not support the large attribute universe [18].

In 2016, Cui *et al.* proposed a server-aided revocable ABE scheme, in which almost all workloads of data users incurred by user revocation are delegated to an untrusted server, and the data user only needs to store a key of constant size [19]. In 2017, Wei *et al.* proposed a forward and backward secure CP-ABE scheme such that a revealed user secret key is useless for decrypting any ciphertexts [20]. However, Cui's and Wei's schemes only support user revocation, do not support attribute revocation. Moreover, the number of users is fixed at the system setup, and the dynamic capacity expansion of users is not supported.

There were also some generic constructions of Revocable ABE. In 2017, Yamada *et al.* proposed two generic constructions of RABE from ABE [21]. Chen *et al.* proposed two generic user revocation systems for ABE in 2018, and a generic attribute revocation system for ABE in 2019.

There was also some work devoted to revocable MA-ABE. Yang *et al.* constructed new MA CP-ABE schemes with user-attribute revocation, i.e., it supports revoke an attribute from a user [22], [23]. The schemes do not use the subset-cover revocation framework to manage users so that support the dynamic capacity expansion of users, and more efficient than the revocable single-authority ABE. However, they only support the small attribute universe. Moreover, they suffer the collusion attack launched by revoked users and non-revoked users, so that they do not meet the forward and backward security [24]. In [24], Wu *et al.* proposed a new MA CP-ABE scheme with user-attribute revocation against collusion attack. However, it also only supports the small attribute universe.

In 2016, Wei *et al.* proposed a large universe revocable MA-ABE scheme. However, it only supports user revocation, the number of users is fixed at the system setup, and the dynamic capacity expansion of users is not supported. What is more, the revocation times are fixed at the system setup, and the public parameter increases with the revocation times. So, it does not suit the large-scale dynamic IoT.

In 2018, Liu *et al.* proposed a large universe MA-ABE scheme with user-attribute revocation [25]. In 2019, Li *et al.* proposed a CCA2-secure large-universe MA-ABE with user-attribute revocation [26]. In 2021, Huang *et al.* proposed a CCA2-secure large universe multiauthority access control scheme that supports user-attribute revocation, online/offline encryption, ciphertext verification, and outsourcing decryption [27]. However, all these schemes achieve revocation based on the subset-cover revocation framework so that do not support the large user-universe. Moreover, they do not resist the collusion attack launched by revoked users and

non-revoked users, so that they do not meet the forward and backward security.

B. MOTIVATION AND CONTRIBUTIONS

Motivated by the aforementioned practical issues in the dynamic IoT, the author revisited the existing revocable ABE schemes and find that some issues make them unsuitable to the dynamic IoT. Firstly, some schemes only support small attributes (or users) universe: the attributes (or users) are fixed and enumerated at system setup, which implies that the flexible number of attributes (or users) is not supported. Secondly, some schemes do not resist the collusion attack launched by revoked users and non-revoked users, i.e., revoked users can collude with non-revoked to decrypt the ciphertext with her/his revoked attributes, which implies that the revocation is failed. Collusion attack is also the common reason why the scheme cannot meet the requirements of the forward and backward security. Thirdly, some schemes do not meet the requirements of the forward and backward security essentially. Fourthly, some schemes are inefficient: the system is constructed in the composite order bilinear groups or there is heavy computation overhead when revocation happens. As far as the authors' knowledge, there is not a secure efficient revocable large universe MA-ABE solving all the issues simultaneously.

In this article, the author designed an efficient revocable large universe MA-ABE based on prime order bilinear groups by improving the previous schemes [27]. As the same as the previous schemes [27], the proposed scheme supports user-attribute revocation, i.e. the revoked user only loses one or more attributes, and she/he can access the data so long as her/his remaining attributes satisfy the access policy. And the limited-resource user can choose to outsource decryption for saving resources. Compared with the previous schemes [27], the contributions are as follows:

(1) By removing the subset-cover revocation framework as the same as the schemes in [22], [23], the proposed MA-ABE supports not only the large attribute-universe but also the large user-universe which implies the flexible number of users. It is suitable for large-scale multi-domain collaboration in the dynamic IoT.

(2) By improving the ciphertext and the data users' secret key, the proposed scheme is secure against the collusion attack launched by revoked users and non-revoked users. Meanwhile, It meets the requirements of forward and backward security.

(3) The performance analysis results indicate that the proposed scheme is more efficient and suitable for the IoT.

C. PAPER ORGANIZATION

The rest of this paper is organized as follows. Some related preliminaries are reviewed in Section II. The system model and security model are presented in Section III. The concrete construction is proposed in Section IV. The security analysis of the proposed scheme is given in Section V. The performance analysis and experimental results are shown

in Section VI. Finally, the author concludes the paper in Section VII.

II. PRELIMINARIES

A. NOTATIONS

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

TABLE 1. Entities.

Symbol	Description
$[n]$	The integer set $\{1, 2, \dots, n\}$.
$[i, j]$	The set containing consecutive integers from i to j .
$s \xleftarrow{R} S$	The variable s is chosen uniformly at random from the set S .
PPT	Probabilistic Polynomial Time.
\mathbb{Z}_p^*	The set $\mathbb{Z}_p - \{0\}$.
$\mathbb{Z}_p^{m \times n}$	The 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}	The system attribute/authority universe.
GP	The system global public parameter.
H/F	The function maps each uid /attribute to a element of \mathbb{G} .
T	The function maps each attribute to the unique attribute authority who controls it.
ASK_{aid}/APK_{aid}	The secret key/public key for authority aid .
$USK_{uid,a}$	The private key of attribute a for user uid .
CT	The original ciphertext generated by the data owner.
CT_{CSP}	The ciphertext generated after CSP re-encrypts CT by using the latest attribute key.
S_{uid}	The attribute set of user uid .
UL_a	The user list of attribute a .
$UpAK_a$	The update key of attribute a .
$TK_{uid,a}$	The transformation key of attribute a for user uid .
RK_{uid}	The retrieving key for user uid .
CT_{out}	The partial decrypted ciphertext generated by CSP.

B. 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 $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a bilinear pairing if it satisfy 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) *Non-degenerate.* $\exists g_1, g_2 \in \mathbb{G}, s.t. e(g_1, g_2) \neq 1_{\mathbb{G}_T}$.

(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 [28]): Let \mathbb{G} and \mathbb{G}_T be the bilinear groups with prime order p , and g be a generator of \mathbb{G} . $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a bilinear map defined on \mathbb{G} . Pick $s, a, b_1, b_2, \dots, b_q \xleftarrow{R} \mathbb{Z}_p$ and $R \xleftarrow{R} \mathbb{G}_T$. Given

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

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

Definition 3: (q -DPBDHE2 Assumption) [28] The q -DPBDHE2 Assumption holds in \mathbb{G} if no probabilistic polynomial time algorithm has probability at least $\frac{1}{2} + \epsilon$ in solving the q -DPBDHE2 problem in \mathbb{G} for non-negligible ϵ .

C. ACCESS STRUCTURES AND LINEAR SECRET-SHARING SCHEMES

Definition 4 (Access Structure [29]): Let $P = \{P_1, P_2, \dots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^P$ is monotone if $\forall B, C : \text{if } B \in \mathbb{A} \text{ and } B \subseteq C \text{ then } C \in \mathbb{A}$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) \mathbb{A} of non-empty subsets of P , i.e., $A \in 2^P \setminus \{\emptyset\}$. The sets in \mathbb{A} are called the authorized sets, and the sets not in \mathbb{A} are called the unauthorized sets.

In attribute-based encryption scheme, the parties are replaced by the attributes. An access structure \mathbb{A} will contain some authorized sets of attributes. Similarly, an access structure we mean a monotone access structure in this study.

Definition 5 (Linear Secret-Sharing Schemes(LSSS) [29]): 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

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., $\rho : [l] \rightarrow U$, which satisfy the following: During the generation of the shares, we consider the column vector $\vec{v} = (s, r_2, \dots, r_n)^T$, where $r_2, \dots, r_n \xleftarrow{\$} \mathbb{Z}_p$. Then the vector of l shares of the secret s according to π is equal to $M\vec{v} \in \mathbb{Z}_p^{l \times 1}$. The share $(M\vec{v})_j$ 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} .

III. SYSTEM MODEL AND SECURITY MODEL

A. SYSTEM MODEL

As shown in Figure 1, the proposed secure efficient revocable large universe multi-authority attribute-based encryption scheme for cloud aided IoT consists of the following entities. As the same as the other decentralized attribute-based encryption schemes [6], [7], the system needs to be created during a trusted setup and publish the correct parameters. Otherwise, the system cannot be working properly. The entity that sets up the system is called Central Authority (CA) in the proposed scheme.

1) CENTRAL AUTHORITY (CA)

As the same as the other multi-authority attribute-based encryption schemes [5], [22], [23], [25]–[27], assuming that CA is a trusted entity and will not compromise or collude with malicious attribute authorities or users. It initializes the

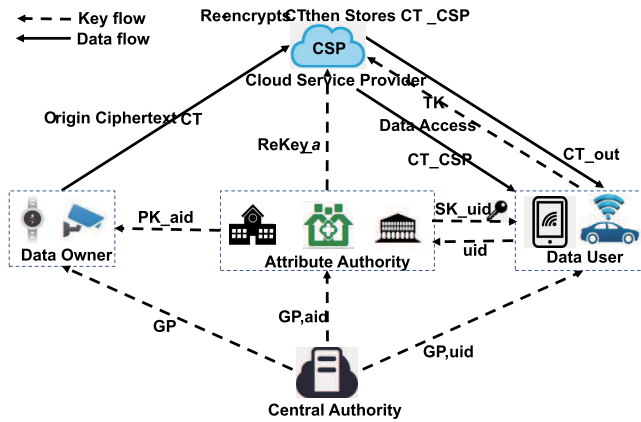


FIGURE 1. System model.

system and publishes the correct global public parameters which can be got by anyone in the system. As the same as the other decentralized MA-ABE schemes [6], [7], [25], secrets in the ciphertext and in the secret key are not embedded in the global public parameters. CA in the proposed scheme is not involved in any attribute management and any secret key generation and does not need to keep any secret key. Besides, it is responsible for the registration of users and authorized attribute authorities. CA assigns each user a unique identity *uid*, and each attribute authority a unique identity *aid*. For instance, CA can be the Social Security Administration, an independent agency of the United States government. Each user will be issued a Social Security Number (SSN) as its unique identity [22].

2) ATTRIBUTE AUTHORITY (AA)

AA is a trusted entity but can be corrupted by the adversary. Each AA is an independent attribute authority and manages a disjoint attribute set respectively which means that each attribute is associated with a single AA. Each AA issues its public key, authenticates users' attributes, and generates the corresponding private keys for them. It also can revoke and update users' attributes by using attribute keys.

3) CLOUD SERVICE PROVIDER (CSP)

CSP is semi-trusted (honest-but-curious). It will honestly and correctly execute the tasks but be curious about the data messages which it receives. As the same in [17], [23], [25]–[27], CSP will not collude with the malicious users. It has both computing capability and large storage. It re-encrypts the origin ciphertext from users by using the latest re-encrypt keys and stores the re-encrypted ciphertext. It is also in charge of re-encrypting the ciphertext when a revocation happens. Moreover, it provides partial decryption of ciphertext for the resource-limited users if they require it.

4) DATA OWNER (DO)

To ensure data confidentiality and achieve flexible access to data, data owners encrypt the data file using a symmetric encryption algorithm which is a lightweight encryption

method, then encrypt the symmetric key under the access policy.

5) DATA USER (DU)

Each data user is assigned a global user identity *uid* by CA. The data user obtains a set of attributes privileges and corresponding decryption private keys from authorities. The data user can freely get any encrypted data from CSP, and decrypt the ciphertext if and only if her/his attribute set satisfies the access policy. Moreover, the resource-limited users can outsource decryption to CSP.

B. FRAMEWORK

This proposed scheme mainly contains ten probabilistic polynomial time algorithms:

SystemSetup(1^λ) \rightarrow (*GP*). The system setup algorithm is run by CA. It takes the system security parameter λ as input and outputs the global public parameters *GP*.

AASetup(*GP, aid*) \rightarrow (*APK_{aid}, ASK_{aid}*). The attribute authority setup algorithm is run by each attribute authority. It takes the global public parameters *GP* and the authority's identity *aid* as input, and outputs the authority's public key *APK_{aid}* and secret key *ASK_{aid}*. The authority keeps the secret key and publishes the public key.

Encrypt(*m, (M, ρ), GP, {APK_{aid}}*) \rightarrow *CT*. The encryption algorithm is run by data owners. It takes a plaintext message *m*, an access structure (*M, ρ*), the global public parameters *GP*, and the involved authorities' public keys {*APK_{aid}*} as input and outputs the ciphertext *CT*.

CREncrypt(*CT, {ReKey_a}*) \rightarrow *CT_{CSP}*. This algorithm is run by CSP. On receiving the ciphertext, CSP re-encrypts it by using the latest re-encrypt keys {*ReKey_a*} corresponding to the attributes set {*a*} in the ciphertext. And outputs the re-encrypted ciphertext *CT_{CSP}*.

If there is a new attribute in the ciphertext, CSP will ask the re-encrypt key to its attribute authority *T(a)*. On receiving the require of the re-encrypt key for a new attribute *a*, the attribute authority *T(a)* initializes a list of users $UL_a = \emptyset$, then chooses a random number $v_a \in \mathbb{Z}_p^*$ as the attribute key, and sets the re-encrypt key $ReKey_a = v_a$. At last, it sends the re-encrypt key *ReKey_a* to CSP via secure channel.

USKeyGen(*uid, aid, a* \in *U*) \rightarrow *USK_{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 *USK_{uid,a}* for the user *uid*.

Decrypt(*CT_{CSP}, uid, GP, {USK_{uid,a}}_{*a* \in *S_{uid}*}) \rightarrow *m*. The decryption algorithm is run by the data user. It takes the ciphertext *CT*, the user's identity *uid*, the global public parameters *GP*, and the private keys {*USK_{uid,a}*}_{*a* \in *S_{uid}*} as input. If the set of user's attributes $S_{uid} \models (M, \rho)$, it outputs the message *m*. Otherwise, it outputs \perp .*

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

TKGen($\{USK_{uid,a}\}$) \rightarrow ($RK_{uid}, \{TK_{uid,a}\}$). This algorithm is run by data users uid . It chooses a random numbers as the retrieving key RK_{uid} , and takes the private keys $\{USK_{uid,a}\}$ as input, then outputs the transformation keys $\{TK_{uid,a}\}$.

PartDec($\{TK_{uid,a}\}, CT_{CSP}, GP$) \rightarrow CT_{out} . This algorithm is run by CSP. It takes the transformation keys $\{TK_{uid,a}\}$ and the ciphertext CT as input, then outputs the partial decrypted ciphertext CT_{out} .

FinalDec(CT_{out}, RK_{uid}) \rightarrow m . This algorithm is run by data users uid . It works out the message m by using the partial decrypted ciphertext CT_{out} and the retrieving key RK_{uid} .

Revoke(uid, UL_a) \rightarrow $UpAK_a$. This algorithm is run by the attribute authority who manages the attributes a . It takes the attribute a , the user uid as input, then publishes the new re-encrypt key $ReKey'_a$ for CSP to update the involved data. Meanwhile, it sends the update key $UpAK_{a,uid'}$ for the non-revoked users uid' update the involved private keys.

If an attribute authority wants to revoke the attribute a from the system, it only deletes all the users who own a from UL_a , and updates all the involved data.

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

C. SECURITY MODEL

In this section, the security model is similar to [7] which is named statically security. In the security game, the adversary should 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 honest authority, which means that the ciphertext still cannot be attacked successfully if only part of the encrypted attributes are from corrupted authorities.

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

System setup. The challenger \mathcal{C} runs the **SystemSetup**(1^λ) algorithm to get the global public parameters GP . It sends GP to \mathcal{A} .

Adversary's queries. The adversary \mathcal{A} issues a polynomially bounded number of queries statically:

- **Authority's public key queries.** \mathcal{A} submits a set of the non-corrupt authorities $N_{AA} \subseteq U_{AA}$, and a set of the corrupt authorities $C_{AA} \subseteq U_{AA}$, where $N_{AA} \cap C_{AA} = \emptyset$.
- **User's secret key queries.** \mathcal{A} submits a sequence $\{(uid_i, S_i)\}_{i \in I}$, where $S_i \subseteq U$ and $T(S_i) \cap C_{AA} = \emptyset$. A pair (uid_i, S_i) denotes that the adversary requests the secret keys for the attributes set S_i of the user uid_i .
- **TransformKey queries.** \mathcal{A} submits a sequence $\{(uid_j, S_j)\}_{j \in J}$, where $J \cap I = \emptyset$, $S_j \subseteq U$ and $T(S_j) \cap C_{AA} = \emptyset$. A pair (uid_j, S_j) denotes that the adversary requests the transformation keys for the attributes set S_j of the user uid_j .

- **Revocation queries.** \mathcal{A} submits a sequence $\{(uid_k, S_k)\}_{k \in K}$, where $S_k \subseteq U$ and $T(S_k) \cap C_{AA} = \emptyset$. A pair (uid_k, S_k) denotes that the adversary asks to revoke the attributes set S_k from the user uid_k .

- **Encryption queries.** \mathcal{A} submits a challenge access structure (M, ρ) , and two equal-length messages m_0, m_1 . It requires that the attributes set $\bigcup_{i \in I \cup J} S_i \cup \{a \in aid\}_{aid \in C_{AA}}$ does not satisfy (M, ρ) .

Challenger's replies. The \mathcal{C} randomly chooses a bit $b \in \{0, 1\}$ and replies to the queries as follows:

- **Authority's public key replies.** For each non-corrupt authority $aid \in N_{AA}$, \mathcal{C} runs the algorithm **AASetup**(GP, aid) \rightarrow (APK_{aid}, ASK_{aid}) to get the authority's key pair, then replies to \mathcal{A} with the corresponding public keys. \mathcal{A} can create the public keys of the corrupt authorities by himself.
- **User's secret key replies.** \mathcal{C} runs the algorithm **USKeyGen**($uid, aid, a \in U$) \rightarrow $USK_{uid,a}$ to get the users' secret keys, then sends them to \mathcal{A} .
- **TransformKey replies.** Firstly, \mathcal{C} runs the algorithm **USKeyGen**($uid, aid, a \in U$) \rightarrow $USK_{uid,a}$ to get the users' secret keys, then runs the algorithm **TKGen**($\{USK_{uid,a}\}$) \rightarrow ($\{TK_{uid,a}\}$) to get the transformation keys and sends them to \mathcal{A} .
- **Revocation replies.** For each pair (uid_k, S_k) , each attribute $a \in S_k$, \mathcal{C} runs the algorithm **Revoke**(uid, UL_a) \rightarrow $UpAK_a$ to get the update keys and sends them to \mathcal{A} , then updates all the involved data with the last re-encrypt keys $\{ReKey_a\}$.
- **Encryption replies.** \mathcal{C} runs the algorithm **Encrypt**($m, (M, \rho), GP, \{APK_{aid}\}$) \rightarrow CT to encrypt m_b with (M, ρ) , then runs the algorithm **CREncrypt**($CT_{CSP}, \{ReKey_a\}$) \rightarrow CT_{CSP} to re-encrypt CT with the last re-encrypt keys $\{ReKey_a\}$. At last, it sends CT_{CSP} to \mathcal{A} .

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

Definition 6: The proposed scheme is static security in the random oracle model if no probabilistic polynomial time adversary can break the above security game with a non-negligible advantage.

IV. CONCRETE SCHEME

In this section, the author presents the concrete construction of revocable MA-ABE for the IoT based on prime-order bilinear groups as follows.

SystemSetup(1^λ) \rightarrow (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 $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ on \mathbb{G} . The attribute universe is $U = \mathbb{Z}_p$. U_{AA} denotes the set of all attribute authorities. Additionally, it chooses two hash functions H and F maps user identities and attributes to elements of \mathbb{G} respectively. The function T maps each attribute

to the unique attribute authority who controls it. Finally, the algorithm outputs the global public parameters $GP = \langle p, g, \mathbb{G}, \mathbb{G}_T, e, U, U_{AA}, H, F, T \rangle$.

AASetup(GP, aid) \rightarrow (APK_{aid}, ASK_{aid}). This algorithm is run by attribute authorities. For each authority $aid \in U_{AA}$, it choose $\alpha_{aid}, \beta_{aid} \in \mathbb{Z}_p^*$ as its secret key ASK_{aid} , and publishes the public key

$$APK_{aid} = \langle e(g, g)^{\alpha_{aid}}, g^{\beta_{aid}} \rangle.$$

Encrypt($m, (M, \rho), GP, \{APK_{aid}\}$) $\rightarrow CT$. 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 $\{APK_{aid}\}$ as input, where $M \in \mathbb{Z}_p^{l \times n}$ and ρ is a map from each row \vec{M}_i of M to an attribute $\rho(i) \in U$. Let δ be a function maps each row \vec{M}_i to the authority who manages attribute $\rho(i)$. i.e., $\delta(i) = T(\rho(i))$. For encryption, the algorithm randomly picks numbers $v_2, \dots, v_n, w_2, \dots, w_n \in \mathbb{Z}_p^*$. Let $\vec{v} = (s, v_2, \dots, v_n)^T$ and $\vec{w} = (0, w_2, \dots, w_n)^T$. For $i = 1, \dots, l$, it computes $\lambda_i = \vec{M}_i \vec{v}$ and $w_i = \vec{M}_i \vec{w}$. The algorithm picks randomly numbers $r_i \in \mathbb{Z}_p^*$, and computes:

$$C_0 = me(g, g)^s, \quad C_{1,i} = e(g, g)^{\lambda_i} e(g, g)^{\alpha_{\delta(i)} r_i}, \\ C_{2,i} = g^{-r_i}, \quad C_{3,i} = g^{\beta_{\delta(i)} r_i} g^{w_i}, \quad C_{4,i} = F(\rho(i))^{r_i}.$$

At last, the origin ciphertext is

$$CT = \{(M, \rho), C_0, \{C_{1,i}, C_{2,i}, C_{3,i}, C_{4,i}\}_{i=1}^l\}.$$

Finally, the ciphertext is sent to CSP.

CReencrypt($CT, \{ReKey_a\}$) $\rightarrow CT_{CSP}$. This algorithm is run by CSP. On receiving the ciphertext, CSP re-encrypts it by using the latest re-encrypt keys $\{ReKey_a\}$ corresponding to the attributes set $\{a\}$ in the ciphertext. If there is a new attribute in the ciphertext, CSP will ask the re-encrypt key to its attribute authority $T(a)$. On receiving the require of the re-encrypt key for a new attribute a , the attribute authority $T(a)$ initializes a list of users $UL_a = \emptyset$, then chooses a random number $v_a \in \mathbb{Z}_p^*$ as the attribute key, and sets the re-encrypt key $ReKey_a = v_a$. At last, it sends the re-encrypt key $ReKey_a$ to CSP via secure channel.

The algorithm takes the origin ciphertext CT and the latest re-encrypt keys $\{ReKey_a\}$ as input, and computes:

$$C'_{3,i} = C_{3,i} C_{2,i}^{-ReKey_{\rho(i)}} = g^{\beta_{\delta(i)} r_i} g^{v_{\rho(i)} r_i} g^{w_i}.$$

At last, the re-encrypted ciphertext is

$$CT_{CSP} = \{(M, \rho), C_0, \{C_{1,i}, C_{2,i}, C'_{3,i}, C_{4,i}\}_{i=1}^l\}.$$

USKeyGen($uid, T(a), a \in U$) $\rightarrow USK_{uid,a}$. Since the attribute a is managed by the authority $T(a)$, then this algorithm is run by the authority $T(a)$. To generate the user's private key, it adds the data user uid into the list of users UL_a , and chooses a random numbers $t_a \in \mathbb{Z}_p^*$. Then it computes

$$K_{uid,a,1} = g^{\alpha_{T(a)} H(uid)^{\beta_{T(a)} + v_a} F(a)^{t_a}},$$

and

$$K_{uid,a,2} = g^{t_a}.$$

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

$$USK_{uid,a} = \langle K_{uid,a,1}, K_{uid,a,2} \rangle.$$

Decrypt($CT_{CSP}, uid, GP, \{USK_{uid,a}\}$) $\rightarrow m$. This algorithm is run by data users. Suppose a user uid with a set of attributes S_{uid} wants to decrypt the ciphertext CT_{CSP} . If $S_{uid} \not\models (M, \rho)$, 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 $\sum_{i \in I} c_i \vec{M}_i = (1, 0, \dots, 0)$. Then it computes: for all $i \in I$,

$$D_i = C_{1,i} e(K_{uid,\rho(i),1}, C_{2,i}) \cdot e(H(uid), C'_{3,i}) e(K_{uid,\rho(i),2}, C_{4,i}),$$

$$\prod_{i \in I} D_i^{c_i} = e(g, g)^s, \\ \frac{C_0}{e(g, g)^s} = \frac{me(g, g)^s}{e(g, g)^s} = m.$$

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.

TKGen($\{USK_{uid,a}\}$) $\rightarrow (RK_{uid}, \{TK_{uid,a}\})$. 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 a random numbers $z \in \mathbb{Z}_p^*$ as the retrieving key, i.e., $RK_{uid} = z$. Then set the transformation key $TK_{uid,a} = \langle K_{uid,a,1}^z, K_{uid,a,2}^z \rangle$, for every $a \in J$.

At last, it keeps the retrieving key $RK_{uid} = z$, then sends $\{TK_{uid,a}\}_{a \in J}$ to CSP.

PartDec($\{TK_{uid,a}\}_{a \in J}, CT_{CSP}, GP$) $\rightarrow CT_{out}$. This algorithm is run by CSP. 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 $\sum_{i \in I} c_i \vec{M}_i = (1, 0, \dots, 0)$, then it computes:

$$CT_1 = \prod_{i \in I} (C_{1,i} e(H(uid), C'_{3,i}))^{c_i},$$

and

$$CT_2 = \prod_{i \in I} (e(K_{uid,\rho(i),1}^z, C_{2,i}) e(K_{uid,\rho(i),2}^z, C_{4,i}))^{c_i}.$$

Finally, it sets $CT_{out} = (CT_1, CT_2)$ and sends it to the data user.

FinalDec(CT_{out}, RK_{uid}) $\rightarrow m$. This algorithm is run by data users. It computes:

$$\frac{C_0}{CT_1 CT_2^{\frac{1}{z}}} = \frac{me(g, g)^s}{e(g, g)^s} = m.$$

Revoke(uid, a) $\rightarrow UpAK_a, UpAK_{a,uid}$. If an attribute a is revoked from the data user uid , the authority $T(a)$ deletes uid from the list UL_a , and chooses a new attribute key $v'_a \xleftarrow{R} \mathbb{Z}_p^*$. Then calculates the new re-encrypt key: $ReKey'_a = v'_a$, and the update key $UpAK_{a,uid'} = H(uid')^{v'_a - v_a}$, for the non-revoked user uid' .

After receiving the update keys, the non-revoked user uid' updates the private keys by calculating:

$$\begin{aligned} K'_{uid',a,1} &= K_{uid',a,1} UpAK_{a,uid'} \\ &= g^{\alpha T(a)} H(uid')^{\beta T(a)} H(uid')^{v_a} F(a)^{t_a}. \end{aligned}$$

CSP updates the involved data by running the algorithm **CRencrypt**($CT_{CSP}, \{ReKey_a - ReKey'_a\}$) $\rightarrow CT_{CSP}$.

$$\begin{aligned} C'_{3,i} &= C'_{3,i}(C_{2,i})^{ReKey_a - ReKey'_a} = C'_{3,i}(C_{2,i})^{v_a - v'_a} \\ &= g^{\beta_{\delta(i)} r_i} g^{v_{\rho(i)} r_i} g^{w_i} g^{(v_a - v'_a) r_i} = g^{\beta_{\delta(i)} r_i} g^{v'_{\rho(i)} r_i} g^{w_i}, \end{aligned}$$

where $a = \rho(i)$.

If an attribute authority wants to revoke the attribute a from the system, it only deletes all the users who own a from UL_a , and updates all the involved data.

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

Correctness.

$$\begin{aligned} D_i &= C_{1,i} e(K_{uid,\rho(i),1}, C_{2,i}) \cdot e(H(uid), C'_{3,i}) e(K_{uid,\rho(i),2}, C_{4,i}) \\ &= e(g, g)^{\lambda_i} e(g, g)^{\alpha_{\delta(i)} r_i} \\ &\quad \cdot e(g^{\alpha_{\delta(i)} H(uid)}^{\beta_{\delta(i)} + v_{\rho(i)}} F(\rho(i))^{t_{\rho(i)}}, g^{-r_i}) \\ &\quad \cdot e(H(uid), g^{\beta_{\delta(i)} r_i} g^{v_{\rho(i)} r_i} g^{w_i}) e(g^{t_{\rho(i)}}, F(\rho(i))^{r_i}) \\ &= e(g, g)^{\lambda_i} e(H(uid), g)^{w_i}, \end{aligned}$$

If $S_{uid} \models (M, \rho)$, it exist a subset $\{\rho(i) : i \in I \subset [l]\}$ of S_{uid} satisfy the access policy (M, ρ) . Then it can calculate constants $\{c_i : i \in I\}$ such that $\sum_{i \in I} \bar{c}_i M_i = (1, 0, \dots, 0)$. Then, it has

$$\sum_{i \in I} \lambda_i c_i = \sum_{i \in I} \bar{v} \bar{M}_i c_i = \bar{v}(1, 0, \dots, 0) = s,$$

and

$$\sum_{i \in I} w_i c_i = \sum_{i \in I} \bar{w} \bar{M}_i c_i = \bar{w}(1, 0, \dots, 0) = 0.$$

Therefore,

$$\begin{aligned} \prod_{i \in I} D_i^{c_i} &= \prod_{i \in I} (e(g, g)^{\lambda_i} e(H(uid), g)^{w_i})^{c_i} \\ &= \prod_{i \in I} e(g, g)^{\lambda_i c_i} e(H(uid), g)^{w_i c_i} \\ &= e(g, g)^{\sum_{i \in I} \lambda_i c_i} e(H(uid), g)^{\sum_{i \in I} w_i c_i} \\ &= e(g, g)^s, \end{aligned}$$

$$\text{Hence, it has } \frac{C_0}{e(g, g)^s} = \frac{me(g, g)^s}{e(g, g)^s} = m.$$

In the outsourcing decryption mode,

$$\begin{aligned} CT_1 &= \prod_{i \in I} (C_{1,i} e(H(uid), C'_{3,i}))^{c_i} \\ &= \prod_{i \in I} (e(g, g)^{\lambda_i} e(g, g)^{\alpha_{\delta(i)} r_i} e(H(uid), g^{\beta_{\delta(i)} r_i} g^{v_{\rho(i)} r_i} g^{w_i}))^{c_i} \end{aligned}$$

$$\begin{aligned} &= \prod_{i \in I} (e(g, g)^{\lambda_i c_i} \prod_{i \in I} e(H(uid), g)^{w_i c_i} \\ &\quad \cdot \prod_e (g, g)^{\alpha_{\delta(i)} r_i c_i} e(H(uid), g)^{(\beta_{\delta(i)} + v_{\rho(i)}) r_i c_i}) \\ &= e(g, g)^s \prod_{i \in I} e(g, g)^{\alpha_{\delta(i)} r_i c_i} e(H(uid), g)^{(\beta_{\delta(i)} + v_{\rho(i)}) r_i c_i} \end{aligned}$$

$$\begin{aligned} CT_2 &= \prod_{i \in I} (e(K_{uid,\rho(i),1}^z, C_{2,i}) e(K_{uid,\rho(i),2}^z, C_{4,i}))^{c_i} \\ &= \prod_{i \in I} (e(g^{\alpha_{\delta(i)} z} H(uid)^{\beta_{\delta(i)} z + v_{\rho(i)} z} F(\rho(i))^{t_{\rho(i)} z}, g^{-r_i}) \\ &\quad \cdot e(g^{t_{\rho(i)} z}, F(\rho(i))^{r_i}))^{c_i} \\ &= \prod_{i \in I} e(g, g)^{-\alpha_{\delta(i)} r_i c_i z} e(H(uid), g)^{-(\beta_{\delta(i)} + v_{\rho(i)}) r_i c_i z}. \end{aligned}$$

Hence,

$$\frac{C_0}{CT_1 CT_2^{\frac{1}{z}}} = \frac{me(g, g)^s}{e(g, g)^s} = m.$$

V. SECURITY ANALYSIS

In this section, the formal security analysis is given to prove the proposed scheme is static security, collusion resistance, and meets the requirements of the forward and backward security.

Theorem 1 proves that the proposed scheme is statically secure as the RW15 scheme [7].

Theorem 1: The proposed concrete scheme is statically secure in the random oracle model under the $q - DPBDHE2$ assumption.

Proof: Suppose that there exists a PPT adversary \mathcal{A} who can break the proposed scheme with non-negligible advantage ε , then a simulator \mathcal{B} can be built to break the RW15 scheme [7] with the same advantage. Denote the challenger of the RW15 scheme by \mathcal{C} .

System setup: \mathcal{B} gets the global parameters $GP = \langle p, g, \mathbb{G}, g, e, U, U_{AA}, T, F, H \rangle$ from \mathcal{C} , then passes them to the adversary \mathcal{A} .

Adversary's queries: The adversary \mathcal{B} issues a polynomially bounded number of queries statically:

- **Authority's public key queries:** \mathcal{A} submits a set of the non-corrupt authorities $N_{AA} \subseteq U_{AA}$, and a set of the corrupt authorities $C_{AA} \subseteq U_{AA}$, where $N_{AA} \cap C_{AA} = \emptyset$, and \mathcal{A} creates the public keys of the corrupt authorities by himself.
- **User's secret key queries:** \mathcal{A} submits a sequence $\{(uid_i, S_i)\}_{i \in I}$, where $S_i \subseteq U$ and $T(S_i) \cap C_{AA} = \emptyset$. A pair (uid_i, S_i) denotes that the adversary requests the secret keys for the attributes set S_i of the user uid_i .
- **TransformKey queries:** \mathcal{A} submits a sequence $\{(uid_j, S_j)\}_{j \in J}$, where $J \cap I = \emptyset$, $S_j \subseteq U$ and $T(S_j) \cap C_{AA} = \emptyset$. A pair (uid_j, S_j) denotes that the adversary requests the transformation keys for the attributes set S_j of the user uid_j .
- **Revocation queries:** \mathcal{A} submits a sequence $\{(uid_k, S_k)\}_{k \in K}$, where $S_k \subseteq U$ and $T(S_k) \cap C_{AA} = \emptyset$. A pair (uid_k, S_k)

denotes that the adversary asks to revoke the attributes set S_k from the user uid_k .

- **Encryption queries:** \mathcal{A} submits a challenge access structure (M, ρ) , and two equal-length messages m_0, m_1 . It requires that the attributes set $\bigcup_{i \in I \cup J} S_i \cup \{a \in aid\}_{aid \in C_{AA}}$ does not satisfy (M, ρ) .

Challenger's replies: After receiving the adversary's queries, the simulator \mathcal{B} sends $C_{AA}, N_{AA}, \{(uid_i, S_i)\}_{i \in I \cup J}, (M, \rho), m_0, m_1$ to \mathcal{C} for request the corresponding public keys, secret keys, ciphertext. \mathcal{C} returns the public keys

$$\{APK_{aid} = \langle e(g, g)^{\alpha_{aid}}, g^{\beta_{aid}} \rangle\}_{aid \in N_{AA}},$$

the secret keys

$$\{USK_{uid_i, S_i} = \langle K_{uid_i, a, 1}, K_{uid_i, a, 2} \rangle_{a \in S_i}\}_{i \in I \cup J},$$

i.e.,

$$\{\langle g^{\alpha_{T(a)}} H(uid_i)^{\beta_{T(a)}} F(a)^{t_a}, g^{t_a} \rangle_{a \in S_i}\}_{i \in I \cup J},$$

and the challenge ciphertext CT as follows:

$$C_0 = m_b e(g, g)^s, \quad C_{1,i} = e(g, g)^{\lambda_i} e(g, g)^{\alpha_{\delta(i)} r_i}, \\ C_{2,i} = g^{-r_i}, \quad C_{3,i} = g^{\beta_{\delta(i)} r_i} g^{w_i}, \quad C_{4,i} = F(\rho(i))^{r_i},$$

where $i = 1, \dots, l$. Then \mathcal{B} replies the queries as follows:

- **Authority's public key replies:** For each authority $aid \in N_{AA}$, each attribute $a \in aid$, \mathcal{B} randomly chooses $v_a \in \mathbb{Z}_p^*$, and sets the public key as

$$APK'_{aid} = \langle e(g, g)^{\alpha_{aid}}, g^{\beta_{aid}} \rangle.$$

- **User's secret key replies:** For each pair (uid_i, S_i) , each attribute $a \in S_i$, \mathcal{B} computes

$$K'_{uid_i, a, 1} = K_{uid_i, a, 1} H(uid_i)^{v_a} \\ = g^{\alpha_{T(a)}} H(uid_i)^{\beta_{T(a)}} F(a)^{t_a} H(uid_i)^{v_a}, \\ K'_{uid_i, a, 2} = K_{uid_i, a, 2} = g^{t_a}.$$

At last, \mathcal{B} sets the secret key as

$$USK'_{uid_i, S_i} = \langle K'_{uid_i, a, 1}, K'_{uid_i, a, 2} \rangle_{a \in S_i}.$$

- **TransformKey replies:** For each pair (uid_j, S_j) , each attribute $a \in S_j$, \mathcal{B} chooses a random value $z_j \in \mathbb{Z}_p^*$, then computes

$$K'^{z_j}_{uid_j, a, 1} = K'^{z_j}_{uid_j, a, 1} H(uid_j)^{z_j v_a} \\ = g^{\alpha_{T(a)} z_j} H(uid_j)^{\beta_{T(a)} z_j} F(a)^{t_a z_j} H(uid_j)^{v_a z_j}, \\ K'^{z_j}_{uid_j, a, 2} = K'^{z_j}_{uid_j, a, 2} = g^{z_j t_a}.$$

At last, \mathcal{B} sets the transformation key as

$$TK'_{uid_j, S_j} = \langle K'^{z_j}_{uid_j, a, 1}, K'^{z_j}_{uid_j, a, 2} \rangle_{a \in S_j}.$$

- **Revocation replies:** For each pair (uid_k, S_k) , each attribute $a \in S_k$, \mathcal{B} deletes uid_k from the list UL_a , and chooses a new attribute key $v'_a \xleftarrow{R} \mathbb{Z}_p^*$. Then calculates the re-encrypt key: $ReKey'_a = v'_a$, and $UpAK_{a, uid'} =$

$H(uid')^{v'_a - v_a}$, where uid' is the non-revoked users. Then updates the involved data by running the algorithm $CReencrypt(CT_{CSP}, \{ReKey_a - ReKey'_a\}) \rightarrow CT_{CSP}$.

- **Encryption replies:** For $i = 1, \dots, l$, \mathcal{B} computes

$$C'_{3,i} = C_{3,i} C_{2,i}^{-v_{\rho(i)}} = g^{\beta_{\delta(i)} r_i} g^{w_i} g^{v_{\rho(i)} r_i},$$

then sets the challenge ciphertext as

$$CT' = (C_0, \{C_{1,i}, C_{2,i}, C'_{3,i}, C_{4,i}\}_{i=1}^l).$$

Guess: \mathcal{A} outputs a guess bit $b' \in \{0, 1\}$, then \mathcal{B} sends b' to \mathcal{C} .

If \mathcal{A} has advantage $Adv_{\mathcal{A}}(\lambda) = \epsilon$ in breaking the concrete scheme, then \mathcal{B} can break the RW15 scheme [7] with the same advantage $Adv_{\mathcal{A}}(\lambda) = \epsilon$. As shown in [7], the RW15 scheme is statically secure in the random oracle model under the $q - DPBDHE2$ assumption, so is the proposed scheme. ■

Theorem 2 proves that the proposed scheme is secure against the collusion attack launched by revoked users and non-revoked users.

Theorem 2: In the proposed scheme, the revoked user has no chance to update its secret key even if she/he colludes with the non-revoked users and corrupts with the attribute authorities which do not manages the revoked attribute.

Proof: In the proposed scheme, when an attribute a is revoked from the data user uid , each update key $UpAK_{a, uid'} = H(uid')^{v'_a - v_a}$ for the non-revoked user $uid' \neq uid$ is associated with the hash value of her/his unique identity which prevents the revoked user from updating her/his secret keys with the other users' update keys.

Meanwhile, it is hard for the non-revoked user to calculate $v'_a - v_a$ or v'_a by solving the discrete logarithm problem, which prevents her/him from working out her/his update key even if she/he colludes with the non-revoked users and corrupts with the attribute authorities which do not manages the revoked attribute. ■

TABLE 2. The 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
n	the number of the users in the system
N	the number of the attributes in the system
N_a	the number of attributes managed by the authority AA_a
S_a	the number of the attributes in user's private key
S_A	the number of authorities involved in user's private key
P_a	the size of the path key for attribute a
l	the complexity of the access structure
N_A	the number of authorities involved in the access structure
I	the number of attributes required for the decryption
I_A	the number of the involved authorities required for the decryption
l_Z	the size of an element in \mathbb{Z}_p
l_G	the size of an element in \mathbb{G}
l_{G_T}	the size of an element in \mathbb{G}_T

TABLE 3. Comparison of properties with previous works.

Schemes	Multi-authority	Large attribute-universe	User-attribute revocation	Resisting collusion attack	Outsource decryption	Forward security	Backward security	Security
YJ14[23]	✓	×	✓	×	×	✓	×	Selective-security
RW15[7]	✓	✓	×	✓	×	–	–	Static-security
LJWY18[25]	✓	✓	✓	×	✓	✓	×	Static-security
Mine	✓	✓	✓	✓	✓	✓	✓	Static-security

TABLE 4. Comparison of computation cost.

Schemes	Encryption (DO)	Re-encryption (CSP)	Decryption (DU)	Outsourcing decryption model			
				TK-generation	Pre-decryption (DU)	Decryption (CSP)	Final-decryption (DU)
YJ14[23]	$E_T + (5l + 2)E + N_A M_T + lM$	0	$(2I_A + 4I)P + IE_T + IE + (5I + 2I_A)M_T$	×	×	×	×
RW15[7]	$lH + (2l + 1)E_T + 4lE + (l + 1)M_T + lM$	0	$3IP + IE_T + 4IM_T + H$	×	×	×	×
LJWY18[25]	$lH + (2l + 1)E_T + 4lE + (l + 1)M_T + lM$	lE	$3IP + IE_T + IE + 4IM_T + H$	$2IE + M$	$I(E_T + E) + M$	$3IP + IE_T + (4I - 1)M_T + H$	$E_T + M_T$
Mine	$lH + (2l + 1)E_T + 4lE + (l + 1)M_T + lM$	$lE + lM$	$3IP + IE_T + 4IM_T + H$	$2IE$	0	$3IP + 2IE_T + (4I - 2)M_T + H$	$E_T + 2M_T$

TABLE 5. Comparison of computation cost for user-attribute revocation^[1].

Schemes	User updates USK ^[2]	CSP updates ciphertext ^[3]
YJ14[23]	M	$2E + M$
RW15[7]	–	–
LJWY18[25]	E	$lH + (2l + 1)E_T + 4lE + (l + 1)M_T + lM$
Mine	M	$E + M$

^[1]One attribute is revoked from one user.
^[2]One user updates his/her secret keys.
^[3]CSP updates one involved ciphertext.

Theorem 3 proves that the proposed scheme meets the requirements of the forward and backward security.

Theorem 3: The proposed scheme meets the requirements of the forward and backward security in the context of attribute revocation.

Proof: When an attribute a is revoked from the data user uid , all the involved ciphertext whether the previous ciphertext or the newly ciphertext have been re-encrypted by the latest re-encrypt key $ReKey'_a = v'_{\rho(i)}$: $C'_{3,i} = g^{\beta_{s(i)}r_i} g^{v'_{\rho(i)}r_i} g^{w_i}$, where $a = \rho(i)$. It is hard for the revoked user to stretch the re-encrypted ciphertext back to the previous version ciphertext she/he can properly decrypt.

Meanwhile, the newly joined user who has the attribute a can decrypt any a -corresponding ciphertext. ■

VI. PERFORMANCE ANALYSIS

In this section, the author compares the proposed scheme with several related MA-CP-ABE schemes in the terms of the feature. Then the author analyzes and compares their efficiency both in the theoretical method and in the experimental method. Table 2 summarizes notations.

A. THEORETICAL ANALYSIS

Firstly, the author compares the properties of the proposed scheme with other previous schemes in Table 3. These MA-CP-ABE schemes all are constructed on the prime order bilinear group. The systems in [7] are created during a trusted setup and publish the global public parameters. My system is created by a trusted entity (called CA) and publishes the global public parameters (GP) as the same as the systems in the other schemes in [23], [25]. All the global public parameters are summarized in Table 6. Different from the YJ14 scheme [23] in which CA is responsible for the distribution of global secret keys for each legal user. As the same as the schemes in [7], [25], CA in the proposed scheme is not involved in any attribute management and any secret key generation. Moreover, no secret is embedded in the global public parameters and CA does not need to keep any secret key. Different from the YJ14 scheme [23] is selective security. the proposed scheme is static security as the same as the schemes in [7], [25]. It can be observed that the proposed scheme is superior to other existing relevant schemes. The attributes universe in the scheme YJ14 [23] is small, which means the scheme YJ14 does not support the flexible number of attributes. The scheme in [7] does not support revocation to change the users' access privilege dynamically, and outsourcing decryption to reduce the computation overhead on the users' side. The scheme in [23] and [25] support user-attribute revocation, but both schemes do not resist the collusion attack launched by revoked users and non-revoked users so that they do not meet the forward and backward security. The proposed scheme provides the large attribute-universe and user-attribute revocation. Only it resists the collusion attack and meets the forward and backward security.

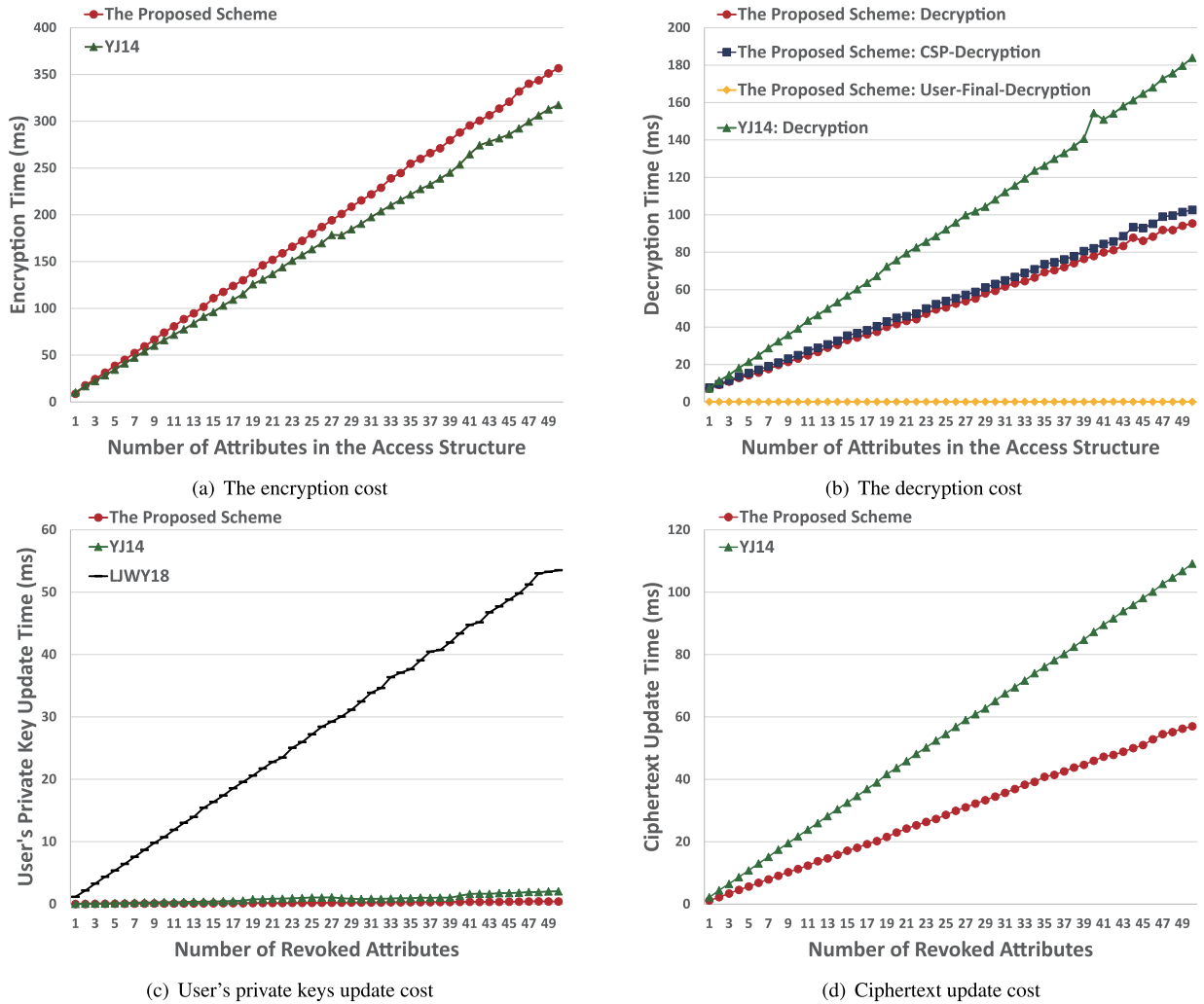


FIGURE 2. Experimental performance comparisons.

TABLE 6. Comparison of storage overhead.

Schemes	GP	AA's ASK	AA's APK	User's USK	Ciphertext
YJ14[23]	$3l_G + l_{G_T} + l_Z$	$(N_a + 3 + n)l_Z + nl_G$	$l_{G_T} + 2(N_a + 1)l_G$	$(2S_A + S_a + 1)l_G + l_Z$	$l_{G_T} + (4l + 2)l_G$
RW15[7]	$l_G + l_{G_T} + l_Z$	$2l_Z$	$l_{G_T} + l_G$	$2S_a l_G$	$(l + 1)l_{G_T} + 3ll_G$
LJWY18[25]	$l_G + l_{G_T} + l_Z$	$(2 + N_a)l_Z$	$l_{G_T} + l_G$	$2S_a l_G + S_a P_a l_Z$	$(l + 1)l_{G_T} + 3ll_G$
Mine	$l_G + l_{G_T} + l_Z$	$(2 + N_a)l_Z$	$l_{G_T} + l_G$	$2S_a l_G$	$(l + 1)l_{G_T} + 3ll_G$

Secondly, the author compares the computational complexity of the proposed scheme with the previous schemes except the LW11 scheme [6] in Table 4 and Table 5. In terms of the computation efficiency of the encryption, the proposed scheme needs more hash operations than the scheme in [23] as the same as the schemes in [7], [25]. As for the computation efficiency of the decryption, the proposed scheme needs the same operations as it in the RW15 scheme [7] which needs fewer pairing operations, exponentiation operations, and multiplication operations than the scheme in [23], and fewer exponentiation operations the scheme in [25]. So the

proposed scheme is more efficient than the other revocable MA-ABE scheme [23], [25]. Especially compare the out-source decryption model with the scheme in [25], the data users do not need pre-decryption the proposed scheme. So the computation complexity of user decryption (the total computation includes pre-decryption and final-decryption by the user) is substantially reduced in the proposed scheme. Moreover, as shown in Table 5, if an attribute is revoked from a user, the proposed scheme needs less exponentiation operation than the scheme in [23], and much less various operation than the scheme in [25] when updating one involved

ciphertext. The involved users need only one multiplication operation as the same as the LJWY18 scheme [25], which is more efficient than the scheme in [23] when updating the secret keys. Therefore, the computation complexity of revocation is more efficient than the other revocable MA-ABE schemes. These are shown more intuitive and clear in Figure 2.

Finally, the author compares the storage overhead of these schemes on every entity in Table 6. As the same as the schemes in [7], [25], the global public parameters in the proposed scheme are two elements less than them in the YJ14 scheme [23]. The attribute authority's secret key and public key in the proposed scheme has the same storage overhead as them in the LJWY18 scheme [25] which is more less than them in the YJ14 scheme [23]. The main storage overhead of each attribute authority comes from its master secret key, public key, and global public parameters. The data owner's storage overhead comes from authorities' public keys and global public parameters. So, the attribute authority's and the data owner's storage overhead in the proposed scheme is not more than the other revocable MA-ABE schemes. The data user's storage overhead mainly comes from the attribute-related secret keys and global public parameters which means that the data user's storage overhead in the proposed scheme is more less than the other revocable MA-ABE schemes.

B. EXPERIMENTAL ANALYSIS

We implemented the YJ14 scheme [23], the LJWY18 scheme [25], and ours in Charm [30] from the Stanford Pairing-Based Crypto library [31]. In the program, the author uses a super-singular symmetric elliptic curve group ("SS512") whose order and base field size is 160-bit order and 512-bit respectively. All the programs were conducted on a virtual machine platform: VMware@Workstation 15 Pro 15.5.2 build-15785246, equipped with a 2.30Ghz Intel Cored CPU with 2GB RAM running 64-bit Linux Ubuntu 18.04.4. The number of attributes is ranging from 1 to 50. All the experiment results are the mean of 100 trials. Finally, the author drew the graphs to compare the computation cost of these schemes in figure 2.

As shown in Figure 2, the encryption cost, the decryption cost, the user's private keys update cost, and ciphertext update cost grow linearly with the number of attributes. From Figure 2(a), the encryption time of the proposed scheme is a little more than it in the YJ14 scheme [23]. In Figure 2(b), it is easy to find that the proposed scheme incurs less decryption time than the YJ14 scheme [23]. Moreover, in the outsource decryption model, the final decryption only requires a constant amount of computation, so its time is nearly constant and much less than it in the YJ14 scheme [23]. Focusing on the cost when revocation happens, Figure 2(c) shows that the user's private keys update time in the proposed scheme is much less than it in the LJWY18 scheme [25]. The ciphertext update time is reduced almost by 50 percent compare with the YJ14 scheme [23] as shown in Figure 2(d).

Therefore, the proposed scheme is very efficient in terms of the decryption cost, and the revocation cost, and the encryption cost may be improved for more efficiency in future work.

VII. CONCLUSION

In this article, the author proposes an efficient revocable large universe multi-authority attribute-based encryption based on prime order bilinear groups. The proposed scheme supports user-attribute revocation. It is proven static security in the random oracle model under the q-DPBDHE2 assumption, and secure against the collusion attack launched by revoked users and non-revoked users. Meanwhile, It meets the requirements of forward and backward security. The outsourcing decryption module is optional for limited-resource users to save resources. The performance analysis results indicate that it is suitable for large-scale cross-domain collaboration in the dynamic cloud-aided IoT. However, the author considers improving the encryption algorithm for more efficiency in future work. The author also consider the security without the random oracle model and supporting more functions such as hiding policy, accountability, and variable outsourced decryption.

ACKNOWLEDGMENT

The author would like to thank the anonymous reviewers for their comments which helped to improve the article.

REFERENCES

- [1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, Sep. 2013.
- [2] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2005, pp. 457–473.
- [3] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proc. 13th ACM Conf. Comput. Commun. Secur.*, 2006, pp. 89–98.
- [4] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2007, pp. 321–334.
- [5] M. Chase, "Multi-authority attribute based encryption," in *Proc. Theory Cryptogr. Conf.* Berlin, Germany: Springer, 2007, pp. 515–534.
- [6] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2011, pp. 568–588.
- [7] Y. Rouselakis and B. Waters, "Efficient statically-secure large-universe multi-authority attribute-based encryption," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Berlin, Germany: Springer, 2015, pp. 315–332.
- [8] J.-S. Su, D. Cao, X.-F. Wang, Y.-P. Sun, and Q.-L. Hu, "Attribute-based encryption schemes," *J. Softw.*, vol. 22, no. 6, pp. 1299–1315, Jun. 2011.
- [9] P. P. Kumar, P. S. Kumar, and P. J. A. Alphonse, "Attribute based encryption in cloud computing: A survey, gap analysis, and future directions," *J. Netw. Comput. Appl.*, vol. 108, pp. 37–52, Apr. 2018.
- [10] J. Li, Y. Zhang, J. Ning, X. Huang, G. S. Poh, and D. Wang, "Attribute based encryption with privacy protection and accountability for CloudIoT," *IEEE Trans. Cloud Comput.*, early access, Feb. 19, 2020, doi: 10.1109/TCC.2020.2975184.
- [11] Y. Liao, G. Zhang, and H. Chen, "Cost-efficient outsourced decryption of attribute-based encryption schemes for both users and cloud server in green cloud computing," *IEEE Access*, vol. 8, pp. 20862–20869, 2020.
- [12] A. Boldyreva, V. Goyal, and V. Kumar, "Identity-based encryption with efficient revocation," in *Proc. 15th ACM Conf. Comput. Commun. Secur.*, 2008, pp. 417–426.

- [13] N. Attrapadung and H. Imai, "Conjunctive broadcast and attribute-based encryption," in *Proc. Int. Conf. Pairing-Based Cryptogr.* Berlin, Germany: Springer, 2009, pp. 248–265.
- [14] N. Attrapadung and H. Imai, "Attribute-based encryption supporting direct/indirect revocation modes," in *Proc. IMA Int. Conf. Cryptogr. Coding.* Berlin, Germany: Springer, 2009, pp. 278–300.
- [15] D. Naor, M. Naor, and J. Lotspiech, "Revocation and tracing schemes for stateless receivers," in *Proc. Annu. Int. Cryptol. Conf.* Berlin, Germany: Springer, 2001, pp. 41–62.
- [16] J. Hur and D. K. Noh, "Attribute-based access control with efficient revocation in data outsourcing systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 7, pp. 1214–1221, Jul. 2011.
- [17] J. Li, W. Yao, Y. Zhang, H. Qian, and J. Han, "Flexible and fine-grained attribute-based data storage in cloud computing," *IEEE Trans. Serv. Comput.*, vol. 10, no. 5, pp. 785–796, Sep. 2017.
- [18] J. Li, W. Yao, J. Han, Y. Zhang, and J. Shen, "User collusion avoidance CP-ABE with efficient attribute revocation for cloud storage," *IEEE Syst. J.*, vol. 12, no. 2, pp. 1767–1777, Jun. 2018.
- [19] H. Cui, R. H. Deng, Y. Li, and B. Qin, "Server-aided revocable attribute-based encryption," in *Proc. Eur. Symp. Res. Comput. Secur.* Berlin, Germany: Springer, 2016, pp. 570–587.
- [20] J. Wei, X. Hu, W. Liu, and Q. Zhang, "Forward and backward secure fuzzy encryption for data sharing in cloud computing," *Soft Comput.*, vol. 23, no. 2, pp. 497–506, Jan. 2019.
- [21] K. Yamada, N. Attrapadung, K. Emura, G. Hanaoka, and K. Tanaka, "Generic constructions for fully secure revocable attribute-based encryption," in *Proc. Eur. Symp. Res. Comput. Secur.*, 2017, pp. 532–551.
- [22] K. Yang, X. Jia, K. Ren, B. Zhang, and R. Xie, "DAC-MACS: Effective data access control for multiauthority cloud storage systems," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 11, pp. 1790–1801, Nov. 2013.
- [23] K. Yang and X. Jia, "Expressive, efficient, and revocable data access control for multi-authority cloud storage," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 7, pp. 1735–1744, Jul. 2014.
- [24] X. Wu, R. Jiang, and B. Bhargava, "On the security of data access control for multiauthority cloud storage systems," *IEEE Trans. Serv. Comput.*, vol. 10, no. 2, pp. 258–272, Mar. 2017.
- [25] Z. Liu, Z. L. Jiang, X. Wang, and S. M. Yiu, "Practical attribute-based encryption: Outsourcing decryption, attribute revocation and policy updating," *J. Netw. Comput. Appl.*, vol. 108, pp. 112–123, Apr. 2018.
- [26] 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.
- [27] K. Huang, X. Wang, and Z. Lin, "Practical multiauthority attribute-based access control for edge-cloud-aided Internet of Things," *Secur. Commun. Netw.*, vol. 2021, Art. no. 8872699, Feb. 2021.
- [28] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Proc. Int. Workshop Public Key Cryptogr.* Berlin, Germany: Springer, 2011, pp. 53–70.
- [29] A. Beimel, "Secure schemes for secret sharing and key distribution," M.S. thesis, Fac. Comput. Sci., Technion-Israel Inst. Technol., Haifa, Israel, 1996.
- [30] J. A. Akinyele, C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green, and A. D. Rubin, "Charm: A framework for rapidly prototyping cryptosystems," *J. Cryptograph. Eng.*, vol. 3, no. 2, pp. 111–128, Jun. 2013.
- [31] B. Lynn. *The Pairing-Based Cryptography Library*. Accessed: 2020. [Online]. Available: <https://crypto.stanford.edu/pbc/>



KAIQING HUANG received the B.E. and M.S. degrees in mathematics from South China Normal University, in 2007 and 2010, respectively, where he is currently pursuing the Ph.D. degree in applied mathematics. Since 2013, he has been with Dongguan Polytechnic, China, as a Lecturer. His research interests include applied cryptography and data security.

• • •