IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 7, 2020, accepted December 20, 2020, date of publication December 23, 2020,
date of current version January 5, 2021.

Digital Object Identifier 10.1109/ACCESS.2020.3046869

A Fine-Grained and Lightweight Data Access Control Model for Mobile Cloud Computing

SOMCHART FUGKEAW, (Member, IEEE)
Sirindhorn International Institute of Technology, Thammasat University, Bangkok 12000, Thailand

e-mail: somchart@siit.tu.ac.th

This work was supported by the Sirindhorn International Institute of Technology (SIIT) Young Researcher Grant under Contract SIIT2019-YRG-SF02.

ABSTRACT With rapidly increasing adoption of cloud computing and the advancement of today mobile computing, it is inevitable that mobile devices are used to receive and send the data through the mobile cloud platform. This increases the convenience and flexibility of data access over the cloud computing since data users are able to access the shared data anytime, anywhere via mobile devices. However, using mobile devices in accessing shared data in a cloud where the sensitive data is encrypted is not practical because mobile devices have limited computing resources in dealing with heavy cryptographic operations. In this article, we propose a lightweight collaborative ciphertext policy attribute role-based encryption (LW-C-CP-ARBE) scheme to support a fine-grained and lightweight access control for mobile cloud environment. We apply CP-ABE approach as a core cryptographic access control and introduce a new proxy re-encryption (PRE) protocol to reduce data re-encryption and decryption cost for the mobile users. To this end, the overhead in running the cryptographic operation at the end-user device is small. In addition, we develop secure access policy sharing and re-encryption protocol to enable users having write privilege to update the data and request the proxy to perform data re-encryption. Finally, we present the evaluation and experiments to demonstrate the efficiency and practicality of our system.

INDEX TERMS Access control, CP-ABE, role-based, mobile cloud, proxy re-encryption, write privilege.

I. INTRODUCTION

Data outsourcing service in cloud has been getting much more adoption by many enterprises for their data sharing solution. The way to store, retrieve, view, and manipulate the data is prone to be done by various kinds of smart mobile computing terminal such as mobile phone, tablets. Generally, mobile devices have limited computing resources especially the storage and computing power. Hence, they are more suitable for dealing with less-computing tasks while a large volume of data is housed at the cloud service provider (CSP). More recent mobile devices have been developed to have more computing power to do more power-consuming tasks such as update data, query the data over the remote database. Nevertheless, they have limited capability in running sophisticated tasks because its computing resources (e.g. processor, energy, storage) cannot achieve the suitable performance as it should be compared to the PCs or server.

There are a number of mobile cloud applications allowing data owners to upload data, video, movie clips, photo to the cloud in order to share to their users. The users then can access these shared data deliberately via their PCs or smart mobile devices. However, if the content of data is sensitive or the owner only wants to share the data to a certain groups of user, these data must be protected from the CSPs and other unauthorized users. Generally, CSPs can be considered as honest but curious as they will operate the functions and service correctly, but may probe or possibly explore the content stored in the cloud server. Therefore, security and privacy protection of the shared data is paramount of importance. Existing cloud applications generally adopt encryption techniques and access control model to support the security and privacy of the outsourced data.

Cryptographic access control is among the most effective solutions for suitable controls of shared data since it supports both access control and encryption feature. Data users are thus required the authorization with the decryption capability to gain access to the shared data.

The associate editor coordinating the review of this manuscript and approving it for publication was Fan-Hsun Tseng.

Attribute-based encryption (ABE) introduced by Sahai and Waters [8] is a generalization of an identity-based encryption. Its variants called Key-Policy Attribute based Encryption (KP-ABE) and Ciphertext Policy Attribute-based Encryption (CP-ABE) were developed by Goyal *et al.* [11] and Bethencourt *et al.* [1] respectively. They have been widely employed by many works as a core construct of a fine-grained access control for outsourced data. In KP-ABE, a message is encrypted by a set of attributes, users having the attributes that satisfy the access structure associated to the secret key can decrypt the ciphertext. In CP-ABE, the message is encrypted by the access structure defined by the data owner, users having the keys containing a set of attributes that satisfy the access structure can decrypt the ciphertext.

According to the integrated encryption feature within the access control model, CP-ABE is a nice approach and suitable for securing outsourced data, especially when the data owners need to control their own access policy over the encrypted data.

Nevertheless, CP-ABE is generally not designed for a mobile cloud environment as its cryptographic operation (e.g. key generation, encryption and decryption) is based on pairing operation, exponentiation, and multiplication [13] that consume a large amount of computation resources in executing the cryptographic functions. Therefore, the straightforward deployment of CP-ABE for in mobile cloud is not applicable. In addition, most CP-ABE approaches provide read access only as the write access can be only done by the data owner because he/she can invoke the access policy to re-encrypt the data. Sharing access control policy requires data privacy-preserving feature since the access policies should not be viewed by any users and their hidden formats must be selectively called by the authorized encryptors only. Delegating write access to data users is thus difficult to implement in practice.

Regarding the policy sharing for supporting write access, we have found that the scheme proposed in [19] is capable to support write access where the access policies are hidden and shared to the write-access users. Nevertheless, they are not suitable for mobile cloud environment.

To the best of our knowledge, most data access control solutions based on CP-ABE works are technically applicable for non-mobile devices while the existing solutions for mobile cloud are generally not designed for guaranteeing the security when they are implemented in the semi-trusted server. In addition, the complexity of CP-ABE cryptographic operations is still required to be computed at the mobile client end. Accordingly, they are no real lightweight data access solutions for mobile cloud environment.

In this article, we propose a lightweight collaborative ciphertext policy attribute role-based encryption (LW-C-CP-ARBE) to support secure and fine-grained access control in mobile cloud environment. In LW-C-CP-ARBE, a lightweight proxy re-encryption (PRE) method and privacy-aware policy sharing are introduced to enable efficient read and write access control in mobile cloud environment. The contributions of this article are depicted as follows.

1. We propose a flexible, fine-grained, and privacy-aware outsourced data sharing model supporting both read and write access control in mobile cloud environment.
2. Our scheme optimizes file re-encryption and data decryption cost based on our proposed delegated proxy. Specifically, the mobile users deal with only the cost of encryption and decryption of the key, and cost of taking the random value out of the intermediate ciphertext. Hence, the cost is typically small.
3. We develop secure policy sharing and retrieval mechanism that allows all access control policies to be stored in the cloud in the encrypted format and dynamically retrieved for data re-encryption without the disclosure of policy content to mobile users.
4. We conduct the performance evaluation to show that our LW-C-CP-ARBE is efficient in mobile data sharing environment. The results demonstrate that LW-C-CP-ARBE provides better re-encryption performance compared to the traditional CP-ABE approach.

The remainder of the paper is organized as follows. Section 2 discusses related works. Section 3 describes the theoretical background of the CP-ABE model and system definitions. Section 4 presents the system overview and our proposed LW-C-CP-ARBE algorithms. Section 5 discusses the security of our scheme. Section 6 shows the evaluation and experiments. Finally, conclusion and future work are given in Section 7.

II. RELATED WORKS

This section presents related research works of CP-ABE based access control and proxy re-encryption. Existing works employing CP-ABE model usually concentrate on minimizing key distribution, reducing computing cost of interaction between data owner and outsourced data storage, improving scalability, efficient enforcement of a policy, and addressing the revocation problem.

To facilitate the secure access control in a multi-authority cloud environment, multi-authority attribute-based encryption (MA-ABE) solutions have been proposed by several works [3], [5], [6], [10]. Nevertheless, none of these approaches have taken policy privacy and write access enforcement into their consideration.

In [10], a collaborative ciphertext policy attribute role-based encryption (C-CP-ARBE) scheme was introduced to support fine-grained, expressive, and flexible access control in multi-authority cloud system. The proposed cryptographic model provides zero key distribution cost and efficient user revocation. In this system, the data is encrypted by CP-ABE method and its ciphertext is then encrypted with the symmetric key. The AES key is then encrypted with the user's public key and stored in the cloud server. For the revocation, if the user is revoked, only the symmetric key is changed, and the CP-ABE key does not need to be re-generated for all non-revoked users. However, this approach is not suitable for MCC environment, because expensive paring operation needs to be done by the client side.

Proxy re-encryption approach was initially introduced by Mambo and Okamoto [15]. They proposed a technique that uses a concept of delegator to perform re-encryption of the ciphertext sent by the originator. In this scheme, the delegator learns neither the decryption keys nor original plaintext. The proxy re-encryption (PRE) concept has been adopted by many works since the heavy cryptographic operation costs are shared to the proxy. For example, In [16], Liang *et al.* applied PRE in the attribute-based cryptographic setting and proposed a Ciphertext Policy Attribute Proxy Re-encryption (CP-ABPRE) for supporting PRE in a CP-ABE setting in which access policy is used to generate re-encryption key. In [18], Kawai proposed a flexible CP-ABE proxy re-encryption scheme by combining key randomized and encrypted methodology and adaptive CP-ABE. The proposed scheme focuses on reducing the computation cost at client side by outsourcing the re-encryption key generation to cloud server. Importantly, Kawai's approach is the first attempt dealing with the outsourcing concept of re-encryption key generation in PRE setting.

In [7], a lightweight proxy-re-encryption scheme called L-PRE was proposed to support lightweight access control for outsourced data. L-PRE offloads the re-encryption task to be done by the proxy. The size of re-encryption key was designed to be small with the assigned expiry period. Hence, re-encryption key is not required to be changed for every re-encryption. This enables the reduction of re-encryption cost compared to other PRE methods.

Even though PRE has been proven for overheads reduction and cost optimization for data access control in cloud computing, the aforementioned PRE approaches were not applicable for data access through mobile devices. This is because the PRE is mainly responsible for re-encryption function but the data access by means of the data decryption is subject to the user's computing resource. Nevertheless, most of the PRE schemes applied for data access control have not been designed for data access with lightweight decryption through mobile devices.

To date, there are a few works [2], [13], [17] dealing with the secure data access in mobile cloud computing. In [13], the authors applied CP-ABE to encrypt outsourced data and proposed an Attribute Based Data Storage (ABDS) system as a cryptographic group-based access control scheme in mobile cloud environment. Their concept is to outsource the heavy encryption and decryption operations to CSPs. However, this approach is not practical for data sharing as it has no clear method for secure key management for multiple data users. In [17], a system called Mobiflage was introduced to hide encrypted volumes within random data in a devices free storage space. This prevents the data leakage in mobile devices and optimizes the costs of data encryption and storage. However, the scheme does not provide the details and evaluate the cost of file access by the mobile devices.

Recently, Li *et al.* [2] proposed a lightweight data sharing scheme called LDSS in mobile cloud computing. They focused on moving an intensive computation to the proxy server and introduce lazy revocation method. Their scheme is based on CP-ABE model. However, in the proposed scheme, the clients still need to compute partial CP-ABE decryption.

In [20], the authors proposed a fast CP-ABE scheme for mobile healthcare network. In the proposed scheme, three semi-trusted third parties are introduced to perform most of the computation tasks for key generation, encryption, and decryption. This helps significantly reduce the computation cost at client side. However, the proposed scheme supports only read access. Also, there are three local outsourcing servers to be responsible for generating user key and delegate key, and file encryption. This incurs the trust issue and expensive operation and maintenance overheads of the local outsourcing servers.

Role-based encryption (RBE) has been also introduced to support cryptographic-based access control in cloud computing. For example, Zhou *et al.* [27] proposed the first a role-based encryption (RBE) scheme for cloud storage systems. Basically, role-based access control (RBAC) policy is enforced through a public parameter of role and a group public to encrypt the data. To decrypt the data, user decryption key containing the public attributes set and public role parameters are used. However, in this system, data is encrypted by the data owner to the specific role, several copies of the encrypted data are required for authorized users who belong to different roles.

In [28], [29], the authors proposed a RBE technique for a secure data sharing in cloud environment. In [28], they introduced authorized keyword search mechanism with efficient decryption using outsourced decryption concept. Their proposed schemes also support data access control in multi-organizations context. As for the RBAC mechanism, search and revocation in role level are practical. Role hierarchy and inheritance properties enable efficient management of multi-users access control. It also helps reduce the expensive overhead in managing such issues in the user level. Also, they introduced outsourced decryption to reduce decryption processing cost at the client side. However, these works use private cloud to keep user and role secrets. Also, there have not provided the real experiment on the resource-constraint devices.

Recently, there are a few works focusing on minimizing the cryptographic protocols such as lightweight authentication with privacy-preserving access control on resource constraint devices such as mobile, IoT devices [21], [22]. However, the proposed schemes have not focused on the lightweight data sharing in which the mobile clients gain access to encrypted data located in the cloud storage.

For lightweight security schemes related to identity-based system and bilinear pairing operations, there are a few works playing around this issue. Chang *et al.* [23] introduced related-key attack (RKA) security into IBS scheme. The provided the proof that a GG scheme [24] is insecure under simple RKA. They thus modified GG scheme and give more security on master secret key generation. This scheme achieves higher security and suitable for lightweight applications. In [25], the authors presented a homomorphic signature scheme based on RSA crypto primitive. The proposed scheme offers more efficient processing at the intermediate nodes compared to bilinear groups and pairings. The idea is that they applied network coding in the module over the integers rather than over a field in a vector space.

In [26], the authors recently proposed an identity-based provable data possession (PDP) scheme of multi-copy on multiple cloud storage servers. They applied the lightweight homomorphic verifiable tags to provide the integrity check of all copies simultaneously. This helps to reduce communication cost and increase availability of the data copies in multi-cloud environment.

Nevertheless, all security protocols proposed in [23]–[26] have not be implemented in the mobile cloud environment.

Also, to the best of our knowledge, there are no works supporting both read and write access in the mobile cloud computing environment.

In this article, we aim at proposing the comprehensive access control scheme supporting both read and write access for MCC as well as optimizing the cryptographic-related operations at mobile device.

III. BACKGROUND

This section describes the basic concept of CP-ABE model used as a core cryptographic module in our proposed scheme. Then, we give the definition of basic elements constituting the access control policy extended from the original CP-ABE access tree.

A. CIPHERTEXT POLICY ATTRIBUTE-BASED ENCRYPTION (CP-ABE)

Bethencourt *et al.* [1] proposed ciphertext policy attribute-based encryption as another kind of attribute-based encryption (ABE) for access control. Basically, the concept of cryptographic construction for both ABE is based on the bilinear maps. A description of the formal definition of bilinear maps is shown below.

1) BILINEAR MAP

Let G_0 and G_1 be two multiplicative cyclic groups of prime order p and e be a bilinear map e: $G_0 \times G_0 \rightarrow G_1$. Let g be a generator of G$_0$. Let H: $\{0,1\}* \rightarrow G0$ be a hash function that the security model is in random oracle.

The bilinear map e has the following properties:

1. Bilinearity: for all $u, v \in G_1$ and $a, b \in Z_p$

$$e\left(u^a, v^b\right) = e(u, v)^{ab}$$

2. Non-degeneracy: $e(g, g) \neq 1$.

Definition 1: Let a set $\{P_1, P_2, \ldots, P_n\}$ be given. A collection $\mathbb{A} \subset 2^{\{P_1,P_2,\ldots,P_n\}}$ is monotone if $\forall B, C$: *if* $B \in \mathbb{A}$ *and* $B \subset$ C $\longrightarrow$ C$\mathbb{A}$.

An access structure is a monotone collection $\mathbb{A}$ of non-empty subsets of $\{P_1, P_2, \ldots, P_n\}$, i.e. $\mathbb{A} \subset 2^{\{P_1,P_2,\ldots,P_n\}}/\emptyset$.

Definition 2 Access Tree T: Let T be a tree representing an access structure. Each non-leaf node of the tree represents a threshold gate, described by its children and a threshold value. If num_x is the number of children of a node x and k_x is its threshold value, then $0 < k_x \leq num_x$. When $k_x = 1$, the threshold gate is an OR gate and when $k_x = num_x$, it is an AND gate. Each leaf node x of the tree is described by an attribute and a threshold value $k_x = 1$. The *kofn* threshold gate is also allowed in T, in this case $k_x = k$ where k is the threshold value determined in the *kofn* gate.

B. ROLE-BASED ACCESS CONTROL MODEL

RBAC is an access control model that provides the relationships of user‘s role, permission, and objects or resources.

Definition 3: RBAC is a tuple of (*U, R, P, UA, PA, RH*) where:

1. *U, R, P* are given sets that represent the set of users, roles, and permission, respectively.
2. $UA \subseteq U \times R$, a many-to-many user-to-role assignment relations;
3. $PA \subseteq P \times R$, a many-to-many permission-to-role assignment relation;
4. $RH \subseteq R \times R$ is a partial order on R representing the role hierarchy.

In RBAC model, it supports role hierarchy where a role can be structured in hierarchy. The top role is on the top of hierarchy while less-powerful roles lay on the lower level. In our model, a set of attributes is mapped to the specific role in RBAC model. We assumed that a higher role can take over all the permissions of its lower roles. For example, a super doctor can access (decrypt) any files that can be accessed by any types of medical doctors.

C. EXTENSION OF ACCESS TREE

To integrate the RBAC model into CP-ABE to enhance the expressiveness and scalability, we define attribute-role based (A-RBAC) access control model as follows:

Definition 4: A-RBAC is a tuple of (*U, R, P, UA, RH, D, APA, Attr*) where:

1. *U, R, P, UA, RH* are as in the RBAC model,
2. $D \subseteq R \times Attr$ is a many-to-many permission-to-role assignment relation.
3. $APA \subseteq Attr \times P$ is an attribute-to-permission assignment relation.

We define $PA \subseteq P \times R$ as for $r \in R, p \in P$, $PA(r,p)$ iff there exists an attribute a such that $D(r,a)$ and $APA(a,p)$. Hence, A-RBAC is an extension of RBAC.

In cloud computing environment, a user u ($\in$ U) is an entity to whom assigned a specific role r_u ($\in$ R) and she requests to access the resource with a permission p ($\in P$) i.e. read or write. Attributes *Attr* are a set of attributes used to

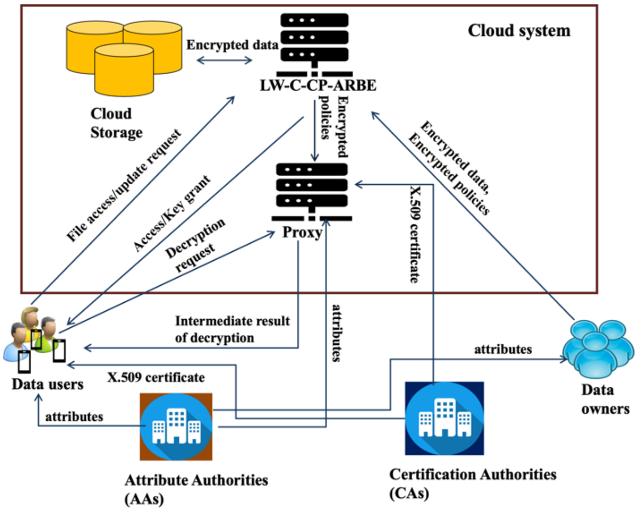


FIGURE 1. Our Proposed LW-C-CP-ARBE Framework.

characterize a role r_u. A set of attributes is issued by attribute authority AA.

In our access control model, the extended access tree is referred to as an access control policy (*ACP*) used to encrypt the data files.

IV. OUR PROPOSED APPROACH

This section presents LW-C-CP-ARBE approach. First, we give the overview of the framework of our proposed scheme. The details of entities and their communications are described. Then, we present LW-C-CP-ARBE algorithm. Finally, we discuss the security analysis of our proposed scheme.

A. SYSTEM OVERVIEW

We propose LW-C-CP-ARBE as the extended version of our previous C-CP-ARBE scheme [10] where the cryptographic construct is based on CP-ABE model [1]. LW-C-CP-ARBE is proposed an access control mechanism supporting lightweight data sharing in the mobile cloud environment. Figure 1 illustrates the system overview of our proposed scheme.

The system model consists of the following entities.

1. Certificate Authorities (CAs) are the trusted parties who issue the public key certificate (X.509 certificate) to all entities including users, AAs, data owners, and a proxy. The key pairs and certificate are used to sign transactions, encrypt secret key, and authenticate the entities in the system.
2. Attribute Authorities (AAs) are the independent parties who issue, revoke, and update users' attributes according to their roles of the particular domain. Each AA is responsible for generating public attribute keys for all attributes belong to the AA and issuing the secret keys to users enrolled in the domain.
3. Data Owners (DOs) are the entities who upload their transaction or processed data into the cloud. They also specify the access control policy to regulate how the users gain access the resource (i.e. data files) and what privilege they have over resources.
4. Data Users (DUs) are entities who requests to access (read or write) the data files from the mobile cloud. Each user is assigned a set of attributes with respect to his/her role by the attribute authority.
5. Cloud Storage is a database housing the data outsourced by the data owners. The storages are maintained by the cloud provider.
6. LW-C-CP-ARBE is an access control system that consists of a collection of cryptographic algorithms for supporting authentication, authorization, key management, and policy management function for outsourced data. The system manages the user decryption keys UDKs) of all users. In our system, UDKs are delivered to the users upon their access.
7. Proxy is a semi-trusted server responsible for data decryption and re-encryption. In our design, proxy server has its own PKI key pair and is installed with X.509 certificate used for authentication with other system modules.

As displayed in Fig. 1, DOs send encrypted data and encrypted policies to the cloud by interfacing with the LW-C-CP-ARBE. In LW-C-CP-ARBE, we extend the CP-ABE scheme and develop more supporting protocols to support both read and write access with the preserved data privacy and secure policy sharing in the mobile cloud environment. The relying parties (DOs, DUs, Proxy) can call the functions from LW-C-CP-ARBE to execute the crypto operations (key generation, encryption, decryption). The system manages the user decryption keys (UDKs) generated from the attributes issued by AAs in the way that all UDKs are delivered to the users upon their access. Hence, DUs do not need to hold their UDKs, they are only responsible to retain the private key issued by the CA. This supports flexibility and scalability of user key management.

In the mobile cloud access control system, we introduce the proxy to perform the heavy crypto operations (decryption, re-encryption) of CP-ABE. When there is a re-encryption request, the proxy will check the privilege of user through the encrypted access control policies sent by the LW-C-CP-ARBE. If the user has the write privilege, he/she can use their mobile devices to update and re-encrypt the data efficiently with the assistance of the dedicated proxy and our LW-C-CP-ARBE. The proxy is issued a key pair and X.509 certificate that are used for authentication when it is communicated with other entities. Hence, the trust of the proxy located in the cloud is assured by the PKI technology.

B. LW-C-CP-ARBE CONSTRUCT

In this section, we present the cryptographic construct of our LW-CP-CP-ARBE. Basically, bilinear map is a major construct in our system setup and user key generation protocol. In our scheme, the proposed cryptographic algorithms are

TABLE 1. Notations used in our model.

Notation	Description
$S_{uid.k}$	Set of all attributes issued to user *uid* and managed by authority *k*.
SK_k	A secret key which belongs to authority *k*.
PK_k	Public key which belongs to authority *k*.
$PK_{x.k}$	Public attribute key issued by authority *k*.
GSK_{uid}	A global secret key of a user *uid*. GSK is a private key issued by the certification authority CA.
$Cert_{uid}$	A public key certificate containing user's public key issued by a CA.
GSKprox	A proxy's global secret key issued by a CA.
Certprox	A proxy's public key certificate issued by a CA.
$(PubK_k, PrivK_k)$	A PKI Key pair consisting of public key and private key issued by a CA. This key pair is issued to the attribute authority *k*.
R_{id}	A role (identified with id) available in the system.
$UDK_{uid.k}$	User Decryption key issued by an authority *k*.
UL	A database that contains a set of user list of each role (UL_{Rid})
GRP	Group role parameter is a list of user lists for all roles.
$EDK_{uid.k}$	*EDK* is an encrypted form of a *UDK* which is encrypted by a user public key.
$ProxK_{pid}$	Proxy decryption key issued by an authority *k*.
SS1	A symmetric key created from the AES algorithm.
SS2	A Session Key used to encrypt random number *R*.
ACP_{Pid}	An extended access tree (referred to as an access control policy) used to encrypt the data files.
ACP^{PE}	An access control policy used to encrypt the ACP_{Pid}. It contains the role attributes of the data owner and proxy allowed to decrypt the ACP_{Pid}.
ACP^{ENCpid}	An encrypted form of ACP_{Pid}.
SCT	A sealed ciphertext which is a ciphertext encrypted with the *SS*.
R1 and *R2*	PseudoRandom value used to encode data *M* and $UDK_{uid.k}$ respectively.

developed based on the extension of the original CP-ABE [1]. The notations used in our model are shown in Table 1.

Our model consists of five major phases including System Setup, Key Generation, Encryption, Decryption, and Re-encryption. Table 1 presents a list of notations used in our model.

1) PHASE 1: SYSTEM SETUP

This phase consists of the following six algorithms run by the AA or data owner.

1. CreateAttributeAuthority(k) → $PK_k, SK_k, PK_{x.k}$. This algorithm takes as input an attribute authority ID(k). The algorithm chooses a bilinear group G_0 of prime order p with generator g. Next it will choose two random $\alpha, \beta \in Z_p$. The public key is computed as:
$$PK_k = \left\{G_0, g, h = g^{\beta}, f = g^{\frac{1}{\beta}}, e(g,g)^{\alpha}\right\}.$$
Note that f is used only for delegation), and the secret key SK_k is (β, g^{α}) . The algorithm also publishes public attribute keys ($PK_{x,k}$) for all attributes issued by the A_k.
2. CreateRole($SK_k, R_{id},$) → UL_{Rid}. The CreateRole algorithm is used to create a role in the system. It takes as inputs attribute authority's secret key (SK_k), Role (R_{id}). It returns user list (UL_{Rid}) of users who qualify to the role R_{id} and stored in the database *UL*.
3. UserRegister(*uid*, $Cert_{uid}$). The UserRegister algorithm is used to bind the users to the specific role. It takes as input userID (*uid*) and user's certificate ($Cert_{uid}$), then it updates UL so that if *uid* has a role R_{id}, then *uid* is contained in UL_{Rid}.
4. CreateGrouproleParameterGRP()→ *GRP*. The Create GroupRole parameter algorithm first collects all user list UL_{Rid} and concatenates them. GRP will be recomputed by either taking a new user *uid* to the R_{id} (add a new user) or remove revoked a user *uid* from R_{id} (revoke a user).
5. EncACP($PK_{aid}, ACP_{PE}, ACP_{Pid}$) → ACP^{ENCpid}. This algorithm is used to encrypt the access control policies. It takes as inputs authority public key PK_k, encryption policy ACP$^{\text{PE}}$ which is a simple CP-ABE policy containing data owner role and proxy role attribute, and access control policy ACP_{Pid}. Then it returns an encrypted ACP^{ENC}.
6. CreateFileProfile($File_{id}$,ACP_{Pid},ACP^{ENCpid}) → *File ProfileTable*. The *FileProfileTable* algorithm is created to maintain the encryption profile of all files shared in the system. Each file is encrypted with the ACP_{Pid} and each corresponding ACP_{Pid} has its encrypted form.

2) PHASE 2: KEY GENERATION

This phase is run by the *AA*. The UserKeyGen algorithm is used to generate the user decryption key (CP-ABE decryption key). The details of the algorithm is detailed as follows.

UserKeyGen($S_{uid,k}$, SK_k, $Cert_{uid}$) →$EDK_{uid,k}$. The KeyGen algorithm consists of two sub-modules:

(1) UDKGen. It takes as input a set of attributes ($S_{uid,k}$) that describes the *uid*'s user decryption key, attribute authority's secret key (SK_k), and $Cert_{uid}$ of user *uid* issued by a CA, then it returns the set of user decryption key (*UDKs*).

For each user *uid*, the *AA*,A_k chooses a random r and $r_j \in \mathbb{Z}_p$, for each attribute $j \in S_{A_k}$. Then the user decryption key ($UDK_{uid.k}$) is computed as:
$$UDK_{j,k} = \Big(D = g^{(\alpha_k + r)/\beta_k, A_i \in S} : \\ D_i = g^r.H(i)^{r_i}, D'_i = g^{r_i}\Big).$$

(2) EDKGen. The algorithm takes public key certificate of users ($Cert_{uid}$) issued by CA to encrypt the $UDK_{id.k}$. The encryption function is expressed as:

$$\mathrm{ENC}_{\mathrm{RSA}}\left(Cert_{uid}, UDK_{uid,k}\right) \equiv EDK_{uid,k}$$

3) PHASE 3: ENCRYPTION

The encryption function is either performed by data owners or users having write privilege. This phase performs two encryption layers and secret seal encryption. It accommodates following algorithms:

ENC(PK_k, *R, M*, ACP_{Pid}, $Cert_{uid}$)→*SCT*. The encryption algorithm performs the following steps as follows:

(1) Encrypt Message M: the algorithm first generates random value R which is combined into M, then M' is obtained. The encryption function is described as follows.

$$\mathrm{GenR}\left(R\left\{rs_1, r\,s_2, \ldots rs_n\right\}\right) \rightarrow R\,1$$
$$\mathrm{Combine}\,(R1, M) = M'$$

Hereafter, the following CP-ABE encryption algorithm takes as inputs authority public key PK_k, access control policy ACP_{Pid}, and data M'. Then it returns a ciphertext CT.

$$M' \mapsto \mathrm{ENC}_{\mathrm{CP-ABE}}\left(PK_k, A\,CP_{Pid}, M'\right) \equiv CT$$

(2) Encrypt *CT*: The algorithm used to encrypt the ciphertext using AES encryption defined as:

$$CT \mapsto \mathrm{ENC}_{\mathrm{AES}}(SS\ 1, CT) \equiv SCT$$

The cyphertext is sealed (encrypted) by the symmetric encryption algorithm referred as secret seal ($SS1$). $SS1$ is calculated from the hash value of *GRP*.

(3) Encrypt *R*: The algorithm takes the session key $SS2$ to encrypt *R1* using AES encryption algorithm defined as:

$$R1 \mapsto \mathrm{ENC}_{\mathrm{AES}}(SS\ 2, R1) \equiv R'$$

Then $SCT||R'$ is sent to be stored at cloud security manager server.

(4) Encrypt *SS1||SS2*: For the final encryption step, *SS1||SS2* is encrypted with $Cert_{uid}$ and the encrypted secret seal (*ESS*) based on RSA encryption as follows:

$$SS \mapsto ENC_{RSA}\left(Cert_{uid}, S\ SI1\|SS2\right) \equiv ESS$$

4) PHASE 4: DECRYPTION

The decryption phase is done by the mobile users and the proxy.

DEC(GSK_{uid},$SCT||R'$, *ESS*, $EDK_{uid.k}$) → M. The algorithm takes as inputs global user secret key GSK_{uid} and a set of encrypted elements including sealed ciphertext *SCT* and encrypted random number R', encrypted secret seal *ESS*, encrypted user decryption key $EDK_{uid.k}$. The decryption process consists of the following steps.

It first requires the mobile user to decrypt *ESS* and $SCT||\,R'$ based on the following RSA decryption algorithm and AES decryption algorithm respectively.

(1) Decrypt *ESS* and R'.

$$SS\ 1, S\ S2 = \mathrm{DEC}_{\mathrm{RSA}}\left(GSK_{uid}, E\ SS\right)$$
$$R1 = \mathrm{DEC}_{\mathrm{AES}}\left(SS\ 2, R'\right)$$

After the above functions are run, *CT, SS1, SS2*, and *R1* are obtained. Then, the user needs to decrypt the $EDK_{uid,k}$ based on the RSA decryption algorithm as follows.

$$\mathrm{DEC}_{\mathrm{RSA}}\left(GSK_{uid}, E\ DK_{uid,k}\right) \equiv UDK_{uid,k}$$

(2) Generate Random number *R2*

The algorithm randomly chooses a set of random seeds *rs* as input and generates secure random number *R2*.

$$\mathrm{Gen}\,R\left(R\left\{rs_1, r\,s_2, \ldots rs_n\right\}\right) \rightarrow R2$$

Then, *R2* is applied to $UDK_{uid,k}$ based on the function below.

$$\mathrm{Combine}\left(R2, U\ DK_{uid,k}\right) = UDK'_{uid,k}$$

In this step, *R2* is assigned with a ticket number(tk_{no}) and it is sent to the LW-C-CP-ARBE system. Before storing *R2* into the system, it is padded with the 8-bit random before the first bit and after the last bit of *R2* string. Then, the user runs RSA encryption algorithm taking the proxy's certificate to encrypt the $UDK'\ _{uid,k}$ and *SS1* to produce encrypted credent key (*ECK*).
$\mathrm{ENC}_{\mathrm{RSA}}(Certprox_{uid}, UDK'\ _{uid,k}, tk_{no}||SS1) \equiv ECK$
Then, *ECK* and *CT* are sent to the proxy server.

(3) Decrypt *CT* The proxy is assumed to perform the functions without the knowledge of each function. The decryption is conducted in order as follows.

3.1 Decrypt *ECK* and *SS1*

The proxy runs the RSA decryption by using its private key to obtain $UDK'\ _{uid,k}$, $\mathrm{tk_{no}}$.,and *SS1*.

$$\mathrm{DEC}_{\mathrm{RSA}}\left(GSKProx, U\ DK'_{uid,k,}, t\ k_{n0.}\|SS1\right)$$
$$\equiv UDK'_{uid,k,}, \mathrm{k_{no}}.\|SS\ 1$$

3.2 Remove *R2* from $UDK'_{uid,k}$

The proxy requests *R2* from the LW-C-CP-ARBE system by giving the ticket no. it received and then the system returns the complete *R2* (with its padding) to the proxy.

3.3 Remove(*R2*, $UDK'_{uid,k}$) = $UDK_{uid,k}$

The function *Remove* is configured to realize the padding term and takes the actual *R2*.

3.4 Decrypt *SCT*

$$CT = \mathrm{DEC}_{\mathrm{AES}}(SS\ 1, S\ CT)$$

3.5 Decrypt *CT*

$$M' = \mathrm{DEC}_{\mathrm{CP-ABE}}\left(CT, U\ DK_{uid,k}\right)$$

If the set of attributes S in $UDK_{uid,k}$ satisfies the *ACP* encrypting the *CT*, the algorithm returns the message M'.
Then M' is sent back to mobile user.

(4) The mobile user runs the following function to remove R from M' and then M is obtained.
The function is defined as:

$$M = Remove(R1, M')$$

5) PHASE 5 RE-ENCRYPTION

This phase is run by the data owner or users who have write access privilege. Once the data is updated, it needs to be re-encrypted and uploaded to the cloud. Here, the proxy first checks the privilege of the user who requests for data re-encryption. The procedure for privilege checking is detailed as follows:

(1) User signs the request for re-encryption ReEncReq<*userid,fileid*> with her private key and sends it to LW-C-CP-ARBEsystems.
(2) LW-C-CP-ARBE checks the authenticity of the users and policy used to re-encrypt the updated file.
(3) LW-C-CP-ARBE calls the proxy to perform re-encryption and sends the ACP^{ENCpid} corresponding to the file to be re-encrypted to the proxy.
(4) A proxy verifies privileges of the user by using its $ProxK_{pid}$ to decrypt the ACP^{ENCpid} and checks the privilege of the user role in the *ACP*. If the user has the write privilege, the user is allowed to run function (1) under the Enc algorithm while function (2) and (3) are done by the proxy. For the re-encryption process, function (4) is not required since *R1, SS1* and *SS2* do not need to be changed.

We offload related CP-ABE decryption and re-encryption which are expensive operations to the proxy while the mobile user executes the lightweight cost of symmetric crypto functions. For mobile users, they only deal with the cost of encryption and decryption of the key, and cost of taking the random value out of the intermediate ciphertext. These costs are typically small; therefore, the solution is practical for data sharing in MCC environment. In addition, our scheme allows data user can update and re-encrypt the data with an optimized re-encryption cost.

We use symmetric encryption as another encryption layer over the ciphertext encrypted by the CP-ABE because we need to minimize the cost of user revocation in the system. In our scheme, if any user is revoked, only the symmetric encryption layer is required to be updated and thus the CP-ABE encryption and the CP-ABE keys in the system are not affected. The change of symmetric key is done by the system and all keys are encrypted by the users' public keys automatically.

V. SECURITY ANALYSIS

This section gives the discussion of security analysis of LW-C-CP-ARBE based on security assumptions and security constructs given in section 3 and section 4.

A. SECURITY MODEL

In this article, the proof of security is a game-based. Since our scheme is based on CP-ABE, the detailed proof of its security can be referred to the original CP-ABE [1], [12].

In the cloud computing environment, we assume that data owners are fully trusted. The users are assumed to be dishonest, i.e., they may collude to access unauthorized data. It is also assumed that the adversary can corrupt authorities only statically, but key queries can be made adaptively. The attack of the security can be done by an adversary requesting a key from the attribute authority.

The security model of our proposed system is defined as follows between an adversary A and a challenger C:

1) SETUP

The simulator generates a key pair and receives a key pair from a CA. For uncorrupted authorities in S_A -S'_A the challenger C runs CreateAttributeAuthority algorithm and gives a public keys *PK* to the adversary A. For corrupted authorities S'_A, the challenger sends both the public keys and secret keys to adversary A.

Phase1: The adversary submits ($S_{uid.k}$, $Cert_{uid}$) to the challenger, where $S_{uid.k}$ is a set of attributes belonging to an uncorrupted authority AA_k. The challenger gives the corresponding user decryption keys *UDK* to the adversary A.

2) CHALLENGE

Adversary A submits two challenge messages m_0 and m_1 to the simulator. The simulator flips a fair binary coin ν, and returns an encryption of m_ν. The ciphertext is computed as follows:

$CT = (ACP, \hat{C} = \mathrm{m}_\nu Z, CT = h^s, \forall y \in Y : C_y = g^{q_y}(0), C'_y = H(att(y))^{q_{y(0)}}$) where γ is a chosen set of attributes. If $\mu = 0$ then $z = e(g,g)^{\alpha s.}$. Therefore, the ciphertext is a valid random encryption of message m_ν.

Otherwise, if $\mu = 1$ then $z = e(g,g)^{z.}$. We then have, $\hat{C} = m_\nu\ e(g,g)^{z.}$. Since z is random, $\hat{C}$ will be a random element of G_1 from the adversaries view and the message contains no information about m_ν.

Phase 2: The simulator performs as it did in Phase 1.

Guess Adversary A submits a guess of ν' *of* ν.

The advantage of A in this game is defined as:

$$ADV_A = \Pr\left[\nu = \nu'\right] - 1/2.$$

□

Definition 5: Our LW-C-CP-ARBE scheme is secure if all polynomial time adversaries have at most a negligible advantage in the above game.

Theorem 1: Suppose there is no polytime adversary who can break the security of CP-ABE with nonneglible advantage; then there is no polytime adversary who can break our system with nonnegligible advantage.

Proof: As we have shown how the adversary A has nonnegligible advantage against our scheme. Similar to A, we show how the adversary B, is built to break the CP-ABE scheme with nonnegligible advantage. The adversary B can play a similar game with the CP-ABE scheme to make private queries during the game to get the private keys in the CP-ABE scheme.

3) INITIALIZATION

The adversary B takes the public key of the authority k, $PK'_k = G_0, g, h = g^{\beta}, f = g^{\frac{1}{\beta}}, e(g, g)^{\alpha}$, and the corresponding secret key (β, g^{α}) . is unknown to the adversary.

Setup. The adversary B gets our public parameters from PK' as $PK_k = G_0, g, h = g^{\beta}, f = g^{\frac{1}{\beta}}, e(g, g)^{\alpha}$, then the public key PK_k is given to the adversary.

Phase 1: B answers private key queries. Suppose the adversary is given a user decryption key query for a set of attributes S where S does not satisfy ACP. Here, B makes a query for obtaining UDK for the same set S twice. Then, B obtains two different $UDKs$ as follows.

$$UDK_{uid,k} = \Big(D = g^{(\alpha_k+r)/\beta_k,} A_i \in S :$$
$$D_i = g^r .H(i)^{r_i}, D'_i = g^{r_i}\Big) .$$
$$UDK'_{uid,k} = \Big(D = g^{(\alpha_k+r')/\beta_k,} A_i \in S :$$
$$D_i = g^{r'} \cdot H(i)^{r_i}, D'_i = g^{r_i}\Big) .$$

where i's are attributes from S, and r, r',r_i, r'_i are random number in Z_p. With $UDK_{uid,k}$ and $UDK'{}_{uid,k}$, B can obtain $\beta g^{r-r'/}$, and chooses random number t_i, $t_{i,j} \in Z_p$. Let $r^* = t_i - r_i$ and $r'' = t_{i,j}$ - r'_i. Then B derives the UDK requested by A as $UDK^* = (D = g^{(\alpha_k+r')/\beta_{k,}}$ $\mathrm{D} = \mathrm{g}^{(\alpha_k+r)/\beta_k}$, $\mathrm{D} = \mathrm{g}^{(\alpha_k+r)/\beta_k}, A_i \in S : D_i = g^{r^*}.H(i)^{r''_i}, D'_i = g^{r''_i}$). Then, the UDK is returned to the adversary A.

4) CHALLENGE

When A decides that Phase 1 is over, it outputs an access policy ACP and two messages m_0 and m_1, which it wishes to be challenged. B gives the two messages to the challenger, and is given the challenge ciphertext CT. Then B computes the challenges ciphertext for A from CT as CT^*. Finally, the challenge ciphertext CT^* is returned to the adversary A.

Phase 2. A makes queries not issued in Phase 1. B responds as in Phase 1.

Guess. Finally, it outputs a guess $v' \in \{1,0\}$, and then B concludes its own game by generating v'. According to the above security model, the advantage of the adversary B is:

$$ADV_A = \left|\Pr\left[v = v'\right] - 1/2\right| = ADV_B$$

Hence, B has nonnegligible advantage against the CP-ABE, which completes the proof of the theorem.

In addition to the Theorem 1 and its proof provided, we also provide the Theorem 2 to substantiate the strength of our cryptographic algorithm.

Theorem 2: An attacker cannot recover a plaintext by using only a decryption key generated from the (corrupted) attribute authority.

Proof: As for the data security encrypted in the CP-ABE method, the CP-ABE key generation, encryption and decryption process are thus secure. The difference between this work and CP-ABE is the way of managing user decryption keys.

In LW-C-CP-ARBE, all UDKs are not broadcasted to the users. Instead, we use public key encryption to encrypt the UDKs and they will be stored in the cloud server. The security of UDKs is thus relied on the private key hold by the user.

In our scheme, *SS* is not used to directly protect the data file, it can be changed when the data owner updates a file or there is a case of user revocation. Even though the proxy knows *SS*, it cannot be used to access the encrypted data.

Without cryptographic random number, the attacker cannot access the plaintext of data outsourced.

B. CONFIDENTIALITY OF ACCESS CONTROL POLICIES

In our scheme, we allow policies to be shared and used by the users having write privilege for data re-encryption. All access polices are encrypted with the traditional CP-ABE [19]. Only the legitimate proxy can invoke the encrypted policy for the decryption process. Therefore, they are secured even they are stored in the cloud server. This reduces the storage and maintenance cost at data owner side. To invoke the policies, the proxy will run the relevant function to use the policies as a part of re-encryption process.

C. TRUST OF THE PROXY

The trust of the proxy is guaranteed by the PKI authentication as the proxy is installed with its key pair and the certificate. The communication between relying parties (LW-C-CP-ARBE, proxy, and users) and the proxy are verified the authenticity based on X.509 certificate. In addition, the decryption package is encrypted with the proxy's public key. Therefore, the cloud cannot pass the ciphertext and collude with other DUs to perform data decryption.

In our scheme, the original plaintext M is combined with the pseudorandom number *R1* before it is encrypted with CP-ABE. Therefore, the proxy itself cannot gain access to the original plaintext even it is given with the decryption key. The intermediate decryption result will be reverted to the mobile user for the final decryption.

VI. PERFORMANCE ANALYSIS

This section presents the performance analysis in three parts including functionality analysis, efficiency analysis, and experimental analysis.

A. FUNCTIONALITY ANALYSIS

Table 2 presents a comparison of the general functionalities of our LW-C-CP-ARBE and related schemes including Li *et al.*'s scheme [2], Wang *et al.*'s scheme [20], and N. H. Sultan *et al.*'s scheme [29]. Generally, our LW-C-CP-ARBE and N.H. Sultan *et al.*'s scheme provide

TABLE 2. Functionality comparison.

	Organization	Private Cloud	Write Access	Outsourced Encryption/ Decryption	Outsourced Access Policy	Re-encryption after Data update
[2]	Single	Not Required	No	Outsourced Encryption and Decryption	No	Done by data owner and proxy
[20]	Single	Not Required	No	Outsourced Encryption and Decryption	No	Done by data owner and proxy
[29]	Multi	Required	No	Outsourced Decryption	No	Done by the data owner
Our s	Multi	Not Required	Yes	Outsourced Encryption and Decryption	Yes	Done by Write access user and the proxy

access control for the collaborative data sharing environment where multiple users from different organizations can access the outsourced data shared by any data owner; while the access control model proposed in [2] and [20] support multiple users from the single organization. Importantly, all schemes provide a lightweight decryption on an end-user side. However, only N.H. Sultan *et al.*'s scheme relies on the private cloud to assist the public cloud during outsourced decryption while other schemes do not require the private cloud. For the effect of data update, Li *et al.*'s scheme and Wang *et al.*'s scheme require full re-encryption by the data owner and the proxy, while N.H. Sultan *et al.*'s scheme requires re-encryption by the data owner, proxy in the private cloud, and public cloud. For our LW-C-CP-ARBE, it allows users having write privilege to do a partial re-encryption and most of encryption processes are done by the proxy. Also, in our scheme, a list of access control policies is encrypted stored in the cloud, while other methods have no this feature. Hence, overall our scheme can support real-world collaborative data sharing with optimized overhead of decryption at the user side and enable data owner to securely and flexibly manage access control policies in the cloud.

B. EFFICIENCY ANALYSIS

For the efficiency analysis, we analyze the computation cost of encryption and decryption of the our LW-C-CP-ARBE and related outsourced cryptographic schemes. We especially focus on the major cost of exponentiation and pairing operation of each scheme that occur at data owner, user, and cloud side. Especially, minimizing the decryption cost at the client side is the core target of most schemes. Table 3 presents the comparison of computation cost of our scheme and Li *et al.*'s scheme [2], Wang *et al.*'s scheme [20], and N. H. Sultan *et al.*'s scheme [29].

In order to simplify the representation of computation cost of each scheme, we define the following notations.

$|A_U|$: The number of attributes contained in the *UDK*.
$|T_A|$: The number of leaf nodes in access control policy.
E_{G0}, E_{G1} : The size of element in G_0 and G_1 groups.
G_0: Exponentiation operation in group G_0
G_1: Exponentiation in group G_1
G_e: Pairing operation in group G_0
G_m: Multiplication operation in group G_0
R_d : Random decryption over the message or ciphertext
n_c: number of attributes associated with the ciphertext
Z_p: The group $\{0,1,\ldots, p\text{-}1\}$ multiplication modulo p

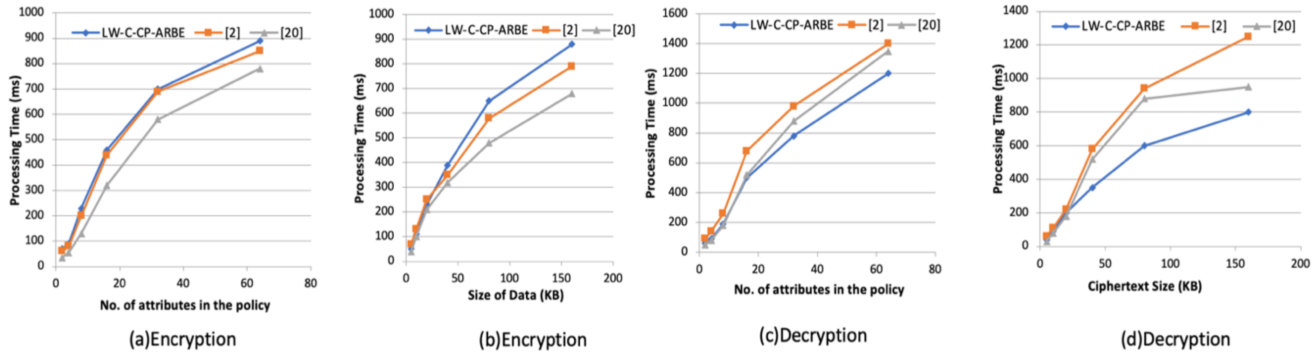
In essence, computation cost is the most important overhead all approaches have focused on. As can be seen from Table 3, Li *et al.*'s scheme applied symmetric encryption to improve the performance of ciphertext encryption and then used CP-ABE to encrypt the symmetric key. This scheme renders more computation for G_0 and G_m on mobile client.

In [20], there were more complexity in outsourcing provider as key generation, encryption, and decryption were offloaded to the provider. Mobile clients need to compute the random component of the intermediate ciphertext obtained from the proxy. The computation cost at the user side comprises three parts: the decryption information cost, the blind key generation cost, and the checking cost. Here, the first two costs are $3G_1$ and G_0,and the checking cost is $2G_e$, blind key generation $(G_1 + G_0)$, and verification cost on G_e. The complexity is propositional to the number of key components that will be computed in each process as well.

In [29], the encryption operation is done by the data owner by using role-based encryption that deals with the number of attributes or roles and the number of exponentiations on G_0 and G_1. For the decryption cost, the user deals with three exponentiation operations, while other associated decryption

TABLE 3. Computation cost comparison.

Scheme	Encryption cost		Decryption cost	
	Owner/User	*Provider or Proxy*	*User*	*Provider or Proxy*
[2]	$3G_0 + G_m$	$2\,\lvert T_A \rvert\, G_0$	$G_0 + G_m$	$(2\,\lvert A_u \rvert +1)\, G_0 + \lvert T_A \rvert\, G_1$
[20]	$4G_0 + 2G_1 + Z_p$	$2\,\lvert T_A \rvert\, G_0$	$G_0 + 3G_1 + 2G_e$	$(2\lvert A_u \rvert+3)G_e + (2\lvert T_A \rvert+1)G_1$
[29]	$(2+n_c)\, G_0 + G_1$	*N/A*	$G_0 + 2G_1$	$3G_0 + 3G_e$
Ours	$2\lvert T_A \rvert\, G_0$	$2\lvert T_A \rvert\, G_0$	r_d	$(2\,\lvert A_u \rvert +1)\, G_0 + \lvert T_A \rvert\, G_1$



(a)Encryption (b)Encryption (c)Decryption (d)Decryption

FIGURE 2. Comparison of Encryption Time and Decryption Time.

operations such as authentication, key search and partial decryption are performed by the public cloud and private cloud.

In our LW-C-CP-ARBE, the data owner initially encrypts the data and outsourced to the cloud. The encryption cost is based on the size of policy or the number of attributes in the policy and the exponentiation on G_0. If there is any re-encryption request due to the data update, the proxy is allowed to fully take the encryption cost. For the decryption, the computation cost of CP-ABE is fully offloaded to the proxy server while the random decryption is only done at the mobile client side. Regarding the communication cost, the proxy directly gets the ciphertext and the cryptographic key components from the LW-C-CP-ARBE system which both they are in the cloud system. The major communication for decryption and re-encryption thus mainly occurs in cloud. Only the intermediate ciphertext is sent out to the mobile user for final decryption. Thus, our LW-C-CP-ARBE renders more competitive cost of decryption than other schemes.

C. EXPERIMENTAL ANALYSIS

For the experimental analysis, we chose [2] and [20] for comparison as they are based on CP-ABE. For the experimental setting, we use Open SSL as a core PKI service to generate key pairs to users and system entities.

The core CP-ABE toolkit and Java Pairing-Based Cryptography [30] are used to implement [2] and [20]. We conduct the test on Intel(R)Xeon(R)-CPU E5620, 2.40GHz with Ubuntu Linux. The mobile device is Samsung Galaxy S7 Qualcomm Snapdragon 820 Octa Core 2.3 GHz, RAM 4 GB with Android 6.0 (Marshmallow).

To measure the encryption time, we use a server to run all the encryption process of these three schemes. We compare the computation efficiency of both encryption and decryption in two criteria: number of attributes in the policy is 5 attributes and size of data/ciphertext is 30 KB. As can be seen from Figure 2a and Fig. 2b, the overhead of encryption operation is proportional to the number of attributes contained in the policy and the size of data to be encrypted. Wang *et al.*'s scheme provides less encryption time compared to ours and Li *et al.*'s scheme. This is because Wang *et al.*'s scheme relies on the only CP-ABE operation while the data encryption mechanism of LW-C-CP-ARBE and Li *et al.*'s scheme is based on two encryption layers including symmetric encryption and CP-ABE. The encryption cost of our proposed scheme provides slightly higher cost than [2] since our approach applies random number to encrypt the message before it is encrypted with the CP-ABE.

For the decryption, we compare the total decryption time taken by the proxy and mobile client. As illustrated in Fig.2c and Fig.2d, our proposed scheme requires less computation time than [2] and [20] when the number of attributes in the policy and ciphertext size is increased. In our scheme, the mobile client only deals with the random decryption; hence our scheme can ensure a constant decryption cost at the user side. In contrast, Li *et al.*'s scheme requires mobile device to run partial CP-ABE decryption to get the symmetric key to decrypt the ciphertext. In Wang *et al.*'s scheme, the user needs to deal with partial exponentiation on the random

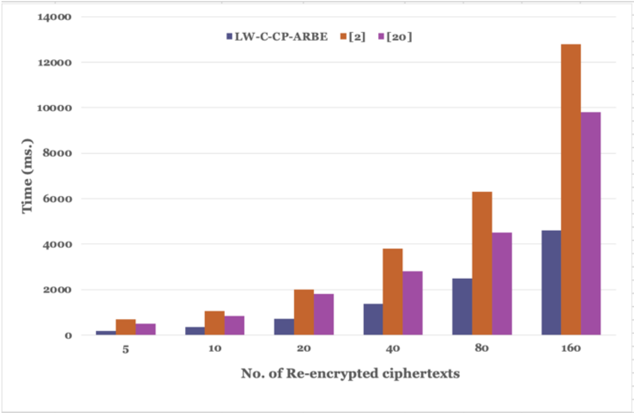


FIGURE 3. Re-encryption cost.

component of the intermediate ciphertext. Accordingly, our scheme renders advantage for the efficiency of decryption at mobile client side as even the number of attributes rises to 64, the decryption time is lower than 1.5 s.

Finally, we also compare the re-encryption cost caused from revocation of our LW-C-CP-ARBE with [2] and [20]. Basically, the revocation cost is evaluated by the total time required to complete the revocation. When the user or attribute is revoked, all ciphertexts can be accessed by the revoked user are required to be re-encrypted. Thus, the re-encryption of ciphertext is the most expensive cost of the revocation. Figure 3 presents the comparison of ciphertext re-encryption cost resulting from user revocation of three schemes. In the test, we varied the number of ciphertexts (re-encryption requests) required to be re-encrypted. Experimentally, the policy that contains 5 attributes is used to re-encrypt files having 30-KB size in average.

As can be seen from Figure 3, our LW-C-CP-ARBE outperforms [2] and [20] for optimizing the cost of revocation. This is because both schemes need to re-encrypt all ciphertexts that has ever been accessed by the revoked users. Hence, the cost of re-encryption is propositional to the number of affected ciphertexts. In our scheme, only the outer layer of symmetric encryption (a secret seal *SS1*) needs to be re-encrypted while the inner layer which is a CP-ABE encryption does not get the effect. When the number of revoked users or attributes grows higher, this advantage becomes more obvious.

To sum up, the experimental results indicate that LW-C-CP-ARBE offers practical solution that provides competitive cost with other outsourced decryption schemes. Especially, the scheme renders acceptable cost of data decryption occurred in mobile device. It also performs better in user or attribute revocation operations.

VII. CONCLUSION AND FUTURE WORKS

In this article, we have presented a LW-C-CP-ARBE supporting write access with lightweight decryption for outsourced data in collaborative mobile cloud computing. We presented the algorithms used to support secure access control together the introduced proxy-based re-encryption to securely and efficiently support secure policy sharing for re-encryption performed by users having write access. Our scheme minimizes the computation overhead from mobile devices to a proxy server, thus the scheme is efficient for implementation in mobile cloud environment. We also discuss the security properties of LW-C-CP-ARBE.

Finally, we provided a performance analysis to demonstrate the feature and efficiency comparison of our scheme and related works. We conducted the experiments to evaluate the encryption, decryption, and re-encryption performance of LW-C-CP-ARBE and related MCC works. The results reveal that our proposed scheme is efficient and practical for real deployment in mobile cloud environment. For future work, we will conduct larger scale of experiments and consider employing multi-proxy in serving the concurrent accesses. Also, we will figure out the way of outsourcing encryption that allows mobile users can encrypt the data and upload to the cloud.

REFERENCES

[1] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. IEEE Symp. Secur. Privacy*, Oakland, CA, USA, May 2007, pp. 321–334.

[2] R. Li, C. Shen, H. He, X. Gu, Z. Xu, and C.-Z. Xu, "A lightweight secure data sharing scheme for mobile cloud computing," *IEEE Trans. Cloud Comput.*, vol. 6, no. 2, pp. 344–357, Apr. 2018.

[3] Z. Wan, J. Liu, and R. H. Deng, "HASBE: A hierarchical attribute-based solution for flexible and scalable access control in cloud computing," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 2, pp. 743–754, Apr. 2012.

[4] T. Nishide, K. Yoneyama, and K. Ohta, "Attribute-based encryption with partially hidden encryptor-specified access structures," in *Applied Cryptography and Network Security* (Lecture Notes in Computer Science), vol. 5037. Berlin, Germany: Springer, 2008, pp. 111–129.

[5] M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou, "Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 1, pp. 131–143, Jan. 2013.

[6] 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.

[7] S. Fugkeaw and H. Sato, "Embedding lightweight proxy re-encryption for efficient attribute revocation in cloud computing," *J. High Perform. Comput. Netw.*, vol. 9, no. 4, pp. 299–309, 2016.

[8] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Eurocrypt*, vol. 3494. Berlin, Germany: Springer-Verlag, 2005, pp. 457–473.

[9] M. Chase and S. S. M. Chow, "Improving privacy and security in multi-authority attribute-based encryption," in *Proc. 16th ACM Conf. Comput. Commun. Secur. - CCS*, 2009, pp. 121–130.

[10] S. Fugkeaw and H. Sato, "An extended CP-ABE based access control model for data outsourced in the cloud," in *Proc. IEEE 39th Annu. Comput. Softw. Appl. Conf.*, Jul. 2015, pp. 73–78.

[11] 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. - CCS*, 2006, pp. 89–98.

[12] L. Cheung and C. Newport, "Provably secure ciphertext policy ABE," in *Proc. 14th ACM Conf. Comput. Commun. Secur. CCS*, 2007, pp. 456–465.

[13] Z. Zhou and D. Huang, "Efficient and secure data storage operations for mobile cloud computing," in *Proc. 8th Int. Conf. Netw. Service Manage. (CNSM) Workshop Syst. Virtualiztion Manage. (SVM)*, Oct. 2012, pp. 37–45.

[14] S. Yu, K. Ren, and W. Lou, "Attribute-based content distribution with hidden policy," in *Proc. 4th Workshop Secure Netw. Protocols*, Oct. 2008, pp. 39–44.

[15] M. Mambo and E. Okamoto, "Proxy cryptosystems: Delegation of the power to decrypt ciphertexts," *IEICE Trans.*, vol.E80-A, no. 1, pp. 54–63, 1997.

[16] K. Liang, J. K. Liu, D. S. Wong, and W. Susilo, "An efficient cloud-based revocable identity-based proxy re-encryption scheme for public clouds data sharing," in *Proc. Eur. Symp. Res. Comput. Secur. (EROSICS)*, Wroclaw, Poland, 2014, pp. 257–272.

[17] Skillen and M. Mannan, "Mobilflage: Deniable storage encryption for mobile devices," *IEEE Trans. Depend. Sec. Comput.*, vol. 11, no. 3, pp. 224–237, Dec. 2014.

[18] Y. Kawai, "Outsourcing the re-encryption key generation: Flexible ciphertext-policy attribute-based proxy re-encryption," in *Proc. Int. Conf. Inf. Secur. Pract. Exper. (ISPEC)*, vol. 9065. Cham, Switzerland: Springer, 2015, pp. 301–315.

[19] S. Fugkeaw and H. Sato, "Enforcing hidden access policy for supporting write access in cloud storage systems," in *Proc. 7th Int. Conf. Cloud Comput. Services Sci.*, 2017, pp. 530–536.

[20] S. Wang, H. Wang, J. Li, H. Wang, J. Chaudhry, M. Alazab, and H. Song, "A fast CP-ABE system for cyber-physical security and privacy in mobile healthcare network," *IEEE Trans. Ind. Appl.*, vol. 56, no. 4, pp. 4467–4477, Jul./Aug. 2020.

[21] S. F. Aghili, H. Mala, M. Shojafar, and P. Peris-Lopez, "LACO: Lightweight three-factor authentication, access control and ownership transfer scheme for E-Health systems in IoT," *Future Gener. Comput. Syst.*, vol. 96, pp. 410–424, Jul. 2019.

[22] S. F. Aghili, H. Mala, P. Kaliyar, and M. Conti, "SecLAP: Secure and lightweight RFID authentication protocol for medical IoT," *Future Gener. Comput. Syst.*, vol. 101, pp. 621–634, Dec. 2019.

[23] J. Chang, H. Wang, F. Wang, A. Zhang, and Y. Ji, "RKA security for identity-based signature scheme," *IEEE Access*, vol. 8, pp. 17833–17841, 2020.

[24] D. Galindo and F. Garcia, "A Schnorr-like lightweight identity-based signature scheme," in *Proc. AFRICACRYPT*, in Lecture Notes in Computer Science, vol. 5580. Berlin, Germany: Springer, 2009, pp. 135–148.

[25] R. Gennaro, J. Katz, H. K. Krawzcyk, and T. Rabin, "Secure network coding over the integers," in *Proc. Int. Workshop Public Key Cryptogr. (PKC)*, 2010, pp. 142–160.

[26] J. Li, H. Yan, and Y. Zhang, "Efficient identity-based provable multi-copy data possession in multi-cloud storage," *IEEE Trans. Cloud Comput.*, early access, Jul. 16, 2019, doi: 10.1109/TCC.2019.2929045.

[27] L. Zhou, V. Varadharajan, and M. Hitchens, "Achieving secure role-based access control on encrypted data in cloud storage," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 12, pp. 1947–1960, Dec. 2013.

[28] N. H. Sultan, M. Laurent, and V. Varadharajan, "Securing Organization's data: A role-based authorized keyword search scheme with efficient decryption," 2020, *arXiv:2004.10952*. [Online]. Available: http://arxiv.org/abs/2004.10952

[29] N. Haque Sultan, V. Varadharajan, L. Zhou, and F. Ahmed Barbhuiya, "A role-based encryption scheme for securing outsourced cloud data in a multi-organization context," 2020, *arXiv:2004.05419*. [Online]. Available: http://arxiv.org/abs/2004.05419

[30] *PBC (Pairing-Based Cryptography) Library*. Accessed: Sep. 30, 2020. [Online]. Available: https://crypto.stanford.edu/pbc/

SOMCHART FUGKEAW (Member, IEEE) received the bachelor's degree in management information systems from Thammasat University, Bangkok, Thailand, the master's degree in computer science from Mahidol University, Thailand, and the Ph.D. degree in electrical engineering and information systems from The University of Tokyo, Japan, in 2017. He is currently a Faculty Member with the Sirindhorn International Institute of Technology, Thammasat University. His research interests include information security, access control, cloud computing security, big data analysis, and high-performance computing.