A fully distributed revocable ciphertext-policy hierarchical attribute-based encryption without pairing

Mohammad Ali[a], Javad Mohajeri[b], Mohammad-Reza Sadeghi[a]

[a] *Department of Mathematics and Computer Science, Amirkabir University of Technology, Tehran, Iran.*
[b] *Electronic Research Institute, Sharif University of Technology, Tehran, Iran.*

Abstract

Several appealing features of cloud computing such as cost-effectiveness and user-friendliness have made many users and enterprises interested to outsource their sensitive data for sharing via cloud. However, it causes many new challenges toward data confidentiality, access control , scalability, and flexibility. Ciphertext-policy Hierarchical attribute-based encryption (CP-HABE) can be a promising solution to the mentioned problems. But, the existing HABE schemes have several limitations in their key delegation and user revocation mechanisms. In this work, to solve these problems, we introduce the concept of *fully distributed revocable* CP-HABE (FDR-CP-HABE) system and propose the first FDR-CP-HABE scheme. The proposed scheme provides a high level of flexibility and scalability in the key delegation and user revocation mechanisms. Moreover, our proposed system is pairing-free and realizes lightweight computing in decryption phase. Indeed, by exploiting the computational operation outsourcing technique, most of the operations have been done by the powerful cloud service provider and very few computations have been leaved to the data user. Also, in our scheme the storage cost on the data user side has been decreased, compared to the other similar works. Moreover, using the hardness assumption of Decisional Bilinear Diffie-Hellman (DBDH) problem, we show that the proposed scheme is adaptively semantically secure in the standard model.

Keywords: `Cloud computing, Hierarchical attribute-based encryption, Access control , Lightweight computation`

1. Introduction

With cloud computing any data owner can store remotely his/her data in a cloud. It provides many facilities. Actually, the data user is not forced to have a high-capacity storage for saving his/her data and to carry a cumbersome device for accessing the data. By using this technology, anyone can access his/her data anytime and anywhere by any device. Also, each user can outsource his/her heavy computation tasks to the cloud, with no need to have a powerful device.

However, the users have to share his/her individual data with cloud. In this case, the shared data may be disclosed by the cloud, or some unauthorized data users obtain the shared data for hostile or profitable purposes. This issue arises the concern about the users privacy and data confidentiality. The most obvious way for addressing the mentioned problems is encrypting the data before sharing it. But using this approach makes a new problem of access control over the encrypted data.

Attribute-based encryption (ABE) [1] is a one-to-many applicable cryptographic primitive which offers data confidentiality and fine-grained access control over the outsourced encrypted data. In such schemes,

Email addresses: `mali71@aut.ac.ir` (Mohammad Ali), `mohajer@sharif.ir` (Javad Mohajeri), `msadeghi@aut.ac.ir` (Mohammad-Reza Sadeghi)

an access structure is defined for controlling the users' access to data outsourced to the cloud. According to the position of the embedded access structure, ABE schemes can be classified into two categories of key-policy ABE (KP-ABE) [2] and ciphertext-policy ABE (CP-ABE) [3]. In a KP-ABE an access structure is imposed on the data user's secret-keys by the key generator authority. Each ciphertext is labeled by the data owner with a set of descriptive attributes. A data user can obtain the corresponding plaintext, only if his/her access structure is satisfied by the attribute set which has been determined by the data owner. While, in a CP-ABE the access structure is embedded in the ciphertext by the data owner, and data users' secret-keys are assigned by the key generator authority according to the set of data users' attributes. Data users whose attributes satisfy the access structure can decrypt the ciphertext.

In traditional CP-ABE schemes, it is assumed that there is a unique key generator authority which generates the attribute secret-keys of data users in the system. But in practice, when a large number of data users make queries for their secret-keys, the key generator authority can not answer them immediately and it causes delay in providing of services. Also, if the key generator authority can not work for a period of time, then key delegation and user revocation services is completely interrupted.

For addressing the mentioned problem, Wang *et al.* [4] proposed the CP-hierarchical ABE (CP-HABE) scheme, by combining a hierarchical identity-based encryption (HIBE) [5] and a CP-ABE scheme. In the proposed scheme there are several key generator authorities, named domain, that each of them administers a set of disjoint attributes. In other words, for each attribute a in a universal attribute set there is a unique key generator authority which administers it. So, any data user for each his/her attribute makes a query to a specified key generator authority for the corresponding attribute secret-key. Also, if an attribute is revoked from some data users, the domain authority should update the attribute secret-keys of the other data users and also generate an update-key for the cloud to re-encrypt the ciphertexts which contains the attribute. In this case, sine each attribute is manged by a unique domain, several problems may be occur in key delegation and user revocation mechanisms. For example, if too many data users has a common attribute, then the corresponding domain has too many client. In this case, either the domain should improve its computation and communication power which it raises the costs, or the key delegation and the user revocation services will be disrupted. Also, as before, if the domain can not work in a time period, then key delegation and user revocation services corresponding to the attribute set managed by the domain will be stopped.

Although, in [6], it is shown that the scheme proposed by Wang *et al.* is fully insecure, their proposed idea had a considerable impact on the several works. In fact, the hierarchal model is used in several CP-ABE schemes [7, 8, 9, 10] to improve the efficiency of their key delegation mechanism. However, in [7, 8] just like the scheme of Wang *et al.* [4], each administers a number of disjoint attributes and therefore they may have the similar problems mentioned in the last paragraph. In [9, 10], the scenario is a bit different. Unlike [4, 7, 8] that any attribute in the universal set is assigned to a unique key generator authority, in [9, 10] any data user is assigned to a unique key generator authority. Indeed, in such the schemes, data users are not allowed to get his/her attribute secret-keys from different domains. Actually, the secret-keys gotten from different domains can not be used together in decryption process.

Although, two schemes [9, 10] seem more flexible than the others in the key management aspect, improving the flexibility and scalability of key delegation and revocation mechanisms can be an interesting problem.

In this paper, we propose a novel fully distributed lightweight access control system. The main contributions of the paper can be summarized as follows:

1. We introduce the concept of *fully distributed revocable* CP-HABE (FDR-CP-HABE) system and propose the first FDR-CP-HABE scheme. Unlike the existing CP-HABE schemes, in our proposed scheme, each domain can manage all of the attributes in the universal attribute set, so for any attribute, the corresponding secret-key can be achieved from any key generator authority. Moreover, the generated attribute secret-keys by different key generator authorities can be used together in the decryption process.

2. We provide a lightweight pairing-free CP-HABE scheme, using the outsourcing technique. Also, we significantly reduce the key storage cost on the user side.

3. We prove our proposed scheme is adaptively semantically secure in the standard model, using the hardness assumption of DBDH problem.

The rest of this paper is organized as follows: Section 2 reviews some related works. In Section 3 we introduce some required preliminaries. The system model, security model, and design goals of our proposed scheme are provided in Section 4. Section 5 presents our proposed CP-HABE in detail. Also the security and performance analyses are given in Section 6 and 7, respectively. In Section 8, we conclude this paper.

2. Related work

The notion of attribute-based encryption (ABE) was introduced for the first time by Sahi and Waters in [1]. The origin of ABE schemes can be derived from the identity-based encryption (IBE) schemes [11, 12, 13]. In such schemes, data owner could share his/her data with several expected data users without knowing their identifier and their public-keys. Actually, the data owner just specifies a set of descriptive attributes and a threshold value d. Each user who has at least d attributes of the specified set can recover the data. After the advent of ABE, two different works [2, 3] divided the primitive class of encryption schemes into two categories of key-policy ABE (KP-ABE) and ciphertext-policy ABE (CP-ABE), respectively. Until then, all the exist ABE schemes were proven in the selective-model. Lewko *et al.* [14] proposed the first adaptive-secure ciphertext-policy and key-policy schemes.

In an ABE scheme, there is just one key authority for generating attribute secret-keys. So, this kind of schemes are not desirable for a large network. For addressing the problem, Wang *et al.* proposed hierarchical ABE (HABE) schemes [4]. After introducing the system, someone uses the hierarchal model for improving the flexibility and scalability of their schemes.

Liu *et al.* [7] by considering the concept of time into the combination of HABE and PRE, proposed a time-based proxy re-encryption (TimePRE) scheme. The main benefit of the scheme is that it is not necessary for the data owner to be online during the user revocation process. In this scheme each user can be identified by a set of attributes and a set of effective time periods that specifies how long the data user is authorized for the attributes.

Li *et al.* [8] proposed a multi-authority attribute-based data access control scheme. In this scheme by outsourcing technique, the decryption overhead for data users has been reduced significantly. Also, an efficient user revocation scheme has been proposed in this work.

In the schemes [4, 7, 8] there are several key generator authorities that each of them administers a number of disjoint attributes. Each data user who has a specific attribute can request to the unique key generator authority and get the corresponding secret-key. Also, for revoking some users, the key generator authority have to update the secret-keys of the unrevoked users and generate an update-key for re-encrypting some encrypted data.

By using an ABE scheme a data owner can control data users' access rights under some attributes which can be organized logically as a single set. It can cause some problems when the data owner does not want to allow the data users who have some specific attributes together to access some data. Bobba *et al.* addressed the problem by introducing attribute set-based encryption (ASBE) schemes [15]. After that Wan *et al.* [10], proposed a CP-HASBE by combining the notions of CP-ASBE and HABE.

Huang *et al.* [9] proposed a data collaboration scheme, by using HABE model in its key delegation mechanism. In this scheme, decryption outsourcing technique is used to reduce the decryption and signing computation overhead. However, the revocation problem has not been considered in the scheme.

It is notable that the term of HABE have been used in two quite different categories of attribute-based scheme. We emphasize that the common term of HABE in these two categories is only a nominal similarity.

In the first category like [4, 8, 9, 10, 16] and ours, in order to lighten the burden of key generator authority, several domain authorities with a hierarchy structure are considered. The main goal in such the systems is to improve the scalability and flexibility of key delegation and user revocation mechanisms. However, In the second one [17, 18, 19, 20] the attributes and files are assumed in a hierarchical relationship with each other and there is just one key generator authority. In CP-HABE schemes [18, 19], the shared data have hierarchical structure. A group of files in these schemes are divided into some hierarchy subgroups and the file in the same hierarchy structure can be encrypted under an integrated access policy structure. The main benefit of designing such schemes is saving encryption time and storage cost. It seems that combining the benefits of the two class of HABE schemes can afford a system with high capabilities. But we will not consider the problem in this work and we will assume that all the attributes are in a same level.

3. Preliminaries

In this section, we introduce some required definitions and assumptions. The notations used in this section can be found in Table 1.

3.1. Bilinear map

Suppose that $(G_1, +)$ and $(G_2, .)$ are two cyclic groups of a prime order q and P_0 is a generator of G_1. A function $\hat{e} : G_1 \times G_1 \to G_2$ is called a bilinear map if it satisfies the following properties:

1. Non-degeneracy: $\hat{e}(P_0, P_0) \neq 1$.
2. Bilinearity: $\hat{e}(aP_1, bP_2) = \hat{e}(bP_1, aP_2) = \hat{e}(P_1, P_2)^{ab}$, for any $a, b \in \mathbb{Z}_q$, and $P_1, P_2 \in G_1$.
3. Computability: There is a polynomial time algorithm which compute $\hat{e}(P_1, P_2)$, for any $P_1, P_2 \in G_1$.

Given two cyclic groups G_1, G_2 of a prime order q, a bilinear map $\hat{e}$, and a random generator $P_0 \in G_1$. Consider three elements aP_0, bP_0, cP_0 of G_1, where a, b and c are three uniform elements of $\mathbb{Z}_q$. The decisional bilinear Diffie-Hellman (DBDH) problem is to distinguish between $\hat{e}(P_0, P_0)^{abc}$ and $\hat{e}(P_0, P_0)^z$, where $z \in \mathbb{Z}_q$ is a uniform element. The hardness assumption of DBDH problem relative to cyclic groups G_1 and G_2 of a prime order q and a bilinear map $\hat{e} : G_1 \times G_1 \to G_2$ is that for any probabilistic polynomial-time (PPT) distinguisher $\mathcal{D}$, there is a negligible function $negl$ such that:

$$\left|\Pr(\mathcal{D}(G_1, G_2, q, \hat{e}, aP_0, bP_0, cP_0, \hat{e}(P_0, P_0)^{abc}) = 1) - \Pr(\mathcal{D}(G_1, G_2, q, \hat{e}, aP_0, bP_0, cP_0, \hat{e}(P_0, P_0)^z) = 1)\right| \leq negl(n), \tag{1}$$

where $P_0 \in G_1$ is a random generator, $a, b, c, z \in \mathbb{Z}_q$ are four uniform elements of $\mathbb{Z}_q$, and n is a security parameter.

3.2. Access tree

Let $\mathbb{U} = \{a_1, \dots, a_m\}$ be a universal set of attributes. An access structure on $\mathbb{U}$ is a nonempty subset $\mathbb{A}$ of $2^{\mathbb{U}}$. Access structure $\mathbb{A}$ is called monotone if for each $A \in \mathbb{A}$ and $B \in 2^{\mathbb{U}}$, $A \cup B \in \mathbb{A}$. For an access structure $\mathbb{A}$, the sets in $\mathbb{A}$ are called authorized sets and the other sets of attributes are unauthorized. Since then, we focus on monotone access structures.

Any monotone access structure $\mathbb{A}$ can be presented by a tree $\mathcal{T}$, where $\mathcal{T}$ is the logical representation of $\mathbb{A}$. We refer to $\mathcal{T}$ as *access tree*.

In this case, any leaf node of the access tree corresponds to an attribute of the access structure. For each node x of the tree, k_x denotes its threshold value. Also, the threshold value of any leaf-node of the tree is equal to 1. For example, the access tree corresponding to access structure

$$\mathbb{A} = \{\{a_1, a_2\}, \{a_1, a_3\}, \{a_2, a_3\}, \{a_3, a_4\}, \{a_1, a_2, a_3\}, \{a_1, a_2, a_4\}, \{a_1, a_3, a_4\}, \{a_2, a_3, a_4\}, \{a_1, a_2, a_3, a_4\}\}$$

Table 1 – Notations

Notation	Description
$\mathcal{D}$	A probabilistic polynomial time distinguisher
DBDH	Decisional bilinear Diffie-Hellman
$\mathbb{U}$	Universal attribute set
a	An attribute
$\mathbb{A}$	Access structure
$\mathcal{T}$	Access tree
k_x	The threshold value of a node x of an access tree
χ	The node set of an access tree
CP-HABE	Ciphertext-policy hierarchical attribute-based encryption
n	The security parameter of the scheme
CA	Central authority
sk_a, K_a, PK_a	Secret-keys and public-key of an attribute a
MSK_1	Master secret-key of the CA
SK_I	Secret-key of a PDA or a KGDA with label I
$params$	Public-parameter of the system
ID_I	Identifier of a domain authority with label I
ID_u	Identifier of a data user u
M, CT	Plaintext and ciphertext
AS	Data user's attribute secret-key set
S	Attribute set
sk'_a, PK'_a	Updated secret-key and public-key of an attribute a
$\widetilde{CT}$	The re-encrypted ciphertext corresponding to CT
$SK_{a,u}$	Secret key of a data user u corresponding to an attribute a
FDR-CP-HABE	Fully distributed revocable ciphertext policy hierarchical attribute-based encryption
PDA	Parent domain authority
KGDA	Key generator domain authority
CSP	Cloud service provider
$SK_{I,a,u}$	Secret-key of a data user u corresponding to attribute a generated by a KGDA with label I
d	Partial decryption key (PDK)
$T^{(i)}$	Decryption token $i = 1, \ldots, 5$
UK_a	Attribute update-key corresponding to an attribute a
CT'	Partially decrypted ciphertext corresponding to CT
$\widetilde{SK}_{I,a,u}$	An updated attribute secret-key corresponding to secret-key $SK_{I,a,u}$
$\mathcal{A}$	A probabilistic polynomial time adversary
F	A pseudorandom function
Π_{mac}	A secure message authentication code scheme
y_a	The leaf-node of an access tree corresponding to an attribute a
q_{y_a}	Polynomial of the leaf node y_a corresponding to an attribute a

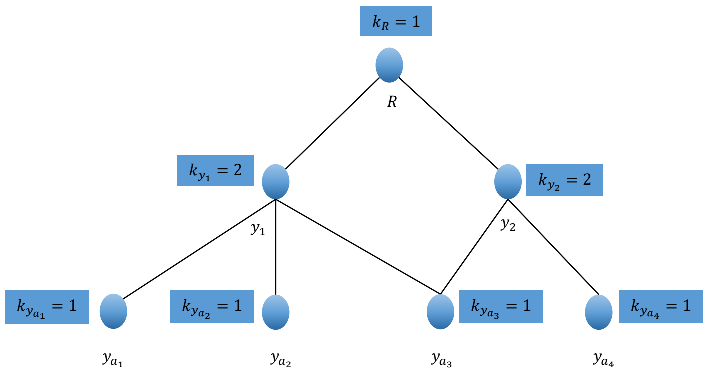

Figure 1: The access tree corresponding to the monotone access structure $\mathbb{A} = \{\{a_1, a_2\}, \{a_1, a_3\}, \{a_2, a_3\}, \{a_3, a_4\}, \{a_1, a_2, a_3\}, \{a_1, a_2, a_4\}, \{a_1, a_3, a_4\}, \{a_2, a_3, a_4\}, \{a_1, a_2, a_3, a_4\}\}$

is shown in Figure 1.

Let χ be the node set of an access tree $\mathcal{T}$ and S be a subset of the universal attribute set $\mathbb{U}$. Consider a function $\widetilde{\mathcal{T}}_S : \chi \to \{0, 1\}$, where for each leaf node x, $\widetilde{\mathcal{T}}_S(x) = 1$ if and only if the corresponding attribute to x belongs to S and for each non-leaf node x, $\widetilde{\mathcal{T}}_S(x) = 1$ if and only if x has at least k_x children like $x_1, \ldots, x_{k_x}$ with $\widetilde{\mathcal{T}}_S(x_i) = 1$, for any $1 \leq i \leq k_x$.

We say that a set of attributes S satisfies the access tree $\mathcal{T}$ if $\widetilde{\mathcal{T}}_S(R) = 1$, where R is the root of $\mathcal{T}$.

3.3. *Revocable CP-HABE*

Any revocable CP-HABE scheme consists of a cloud service provider (CSP) which provides the storage service, a central authority (CA) which responsible for generating public-parameters of the system, several domain authorities that generate the users' secret-keys, some data owners that encrypt and outsource their data to the CSP, and some data users which are labeled by a set of attributes and can access to the outsourced data according to their attributes. When a data user possesses an attribute, he/she can get the corresponding secret-key from a domain authority. Also, in these systems, it is possible that an attribute is revoked from a data user. In this case, in order to prevent the access of the data user, the corresponding parameters of the system to the attribute should be updated and the ciphertexts that are relevant to the attribute should be re-encrypted. Also, the attribute secret-keys of the other data users should be updated according to the new parameters. These schemes consist of the following nine algorithms [4]:

1. **Setup(1^n):** The CA runs this algorithm. The input of this algorithm is a security parameter 1^n. The algorithm outputs public-parameters of the system, *params*, secret key, sk_a, and public-key, PK_a, of each attribute in the universal attribute set, and master secret-key of the CA, MSK_1.

2. **CreateDomain(**$params, SK_i, \{ID_{i+1}\}_{i=1}^{t}$**):** The algorithm is run by the CA or a domain authority to create $t \in \mathbb{N}$ new subdomains. The inputs of the algorithm are the public-parameters of the system, *params*, secret-key of the domain authority running the algorithm, SK_i, (master secret-key of CA, MSK_1, if CA is the runner of this algorithm) and an identifier set of the new sub-domains, $\{ID_{i+1}\}_{i=1}^{t}$. The outputs of the algorithm are secret-keys of the new sub-domains.

3. **CreateUser(**$params, SK_i, ID_u, sk_a$**):** This algorithm can be run by any domain authority. Its inputs are the system public-parameters, *params*, secret-key of the domain authority running the algorithm, SK_i, a data user's identifier, ID_u, and secret-key of an attribute, sk_a. The output of the algorithm is a secret-key of the data user corresponding to the attribute.

4. **Encrypt(**$params, M, \mathcal{T}$**):** This algorithm is run by a data owner. It takes the public-parameters of the system, *params*, a message M and an access tree $\mathcal{T}$ as inputs. It outputs the corresponding ciphertext.

5. **Decrypt($params, CT, AS$):** This algorithm is run by a data user. The inputs of the algorithm are the system public-parameters $params$, a ciphertext CT, and a secret-key set AS of a specific data user u corresponding to an attribute set S. The decryption can be done if and only if S satisfies the access tree of CT.

6. **UpdateAttribute($params, a$):** A domain authority runs this algorithm. It takes the public-parameters of the system, $params$, and an attribute a revoked from a data user. The outputs of the algorithm is the updated secret-key sk'_a and public-key PK'_a of the attribute.

7. **UpdatekeyGen($params, sk_a, sk'_a$):** This algorithm is run by a domain authority. It takes public-parameters of the system, $params$, the secret-key of a revoked attribute, sk_a, and the updated secret-key sk'_a obtained from **UpdateAttribute($params, a$)**. It returns an update-key UK_a.

8. **ReEncrypt($params, a, UK_a, CT$):** The cloud service provider runs this algorithm. It takes the system public-parameters $params$, an attribute a, an update-key UK_a corresponding to the attribute a, and a ciphertext CT which its access tree has a leaf-nod corresponding to the attribute. The output of the algorithm is a re-encrypted ciphertext $\widetilde{CT}$.

9. **UpdateSecretKey($params, a, UK_a, SK_{a,u}, ID_u$):** Any domain authority can run this algorithm. The inputs of the algorithm are the system public-parameters $params$, an attribute a, an update-key UK_a corresponding to the attribute, a data user's attribute secret-key $SK_{a,u}$, and a data user's identifier ID_u. It outputs an updated attribute secret-key of the data user.

4. Definitions, system model and security requirements of the proposed scheme

In this section, we propose the definition of a *fully distributed revocable* CP-HABE (FDR-CP-HABE) scheme for the first time. Then, we present the system model and system definition of our proposed scheme. After that, we give the security model, design goals, and security definitions of an FDR-CP-HABE scheme. Table 1 presents the notations used in this section and our proposed construction in Section 5.

4.1. Definition of a fully distributed revocable CP-HABE (FDR-CP-HABE) scheme

Consider a revocable CP-HABE scheme as Section 3.3. We say that this scheme is *fully distribute* if the following conditions hold:

- For any attribute a in the universal attribute set, a data user u eligible for the attribute can get the corresponding attribute secret-key from any key-generator domain authority in the system.
- The attribute secret-keys generated by different domain authorities can be used together in the decryption process.
- For any attribute a revoked from a data user, generating the correponding update-key, and updating the parameters of the attribute and secret-keys of the other data users can be done by any key-generator domain authority in the system, even if the domain authority updating a secret-key is not the generator of it.

It is clear that an FDR-CP-HABE system provides an ideal flexibility and scalability in key delegation and user revocation mechanisms. To the best of our knowledge, no such the system has been introduced yet. Indeed, as we have mentioned in the introduction, in the existing schemes, either each attribute is managed by a specified domain authority, or each data user is assigned to a specified domain authority. In Section 5, we will propose the first FDR-CP-HABE scheme in detail.

4.2. System model

The architecture of our proposed FDR-CP-HABE scheme is described in Fig.2. Similar to Section 3.3, the following entities participate in our proposed FDR-CP-HABE scheme:

1. Central authority (CA): A trusted third party which is responsible for generating the global public parameters of the system and also providing secret-keys of domain authorities in the top level.
2. Domain authority: It is a trusted third party which, as Fig.2 illustrates, according to its position can be considered as one of the following domain authorities:
   - Parent domain authority (PDA): These Domains at least have one subdomain. Their responsibility is to generate secret-keys of their subdomains.
   - Key generator domian authority (KGDA): These Domains does not have any subdomains. They are responsible for generating data users' attribute secret-keys.
3. Cloud service provider (CSP): A semi-trusted entity which has a high computational power and storage capacity. This party provides storage and computational services for the other entities.
4. Data owner: This client outsources his/her data to the cloud after defining access structure over a set of attributes and encrypting the data under the access structure using the public-parameters of the system.
5. Data user: Each data user is specified by a set of attributes and a unique identifier (ID). Any data user who has an attribute can make a query to an arbitrary KGDA. The KGDA checks whether the data user is eligible for the attribute or not. If so, the corresponding secret-key is generated and is returned to him/her. The data user can access to some outsourced data using the obtained secret-keys.

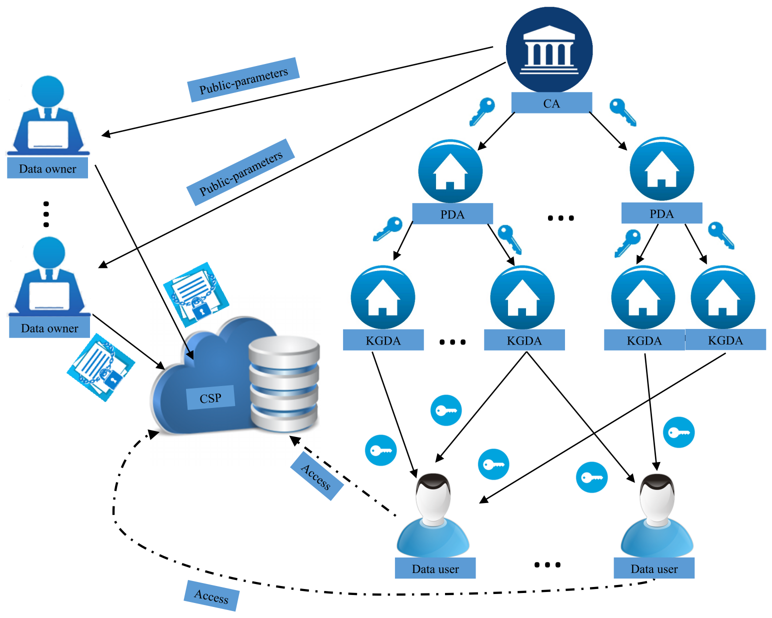

Figure 2: Architecture of FDR-CP-HABE

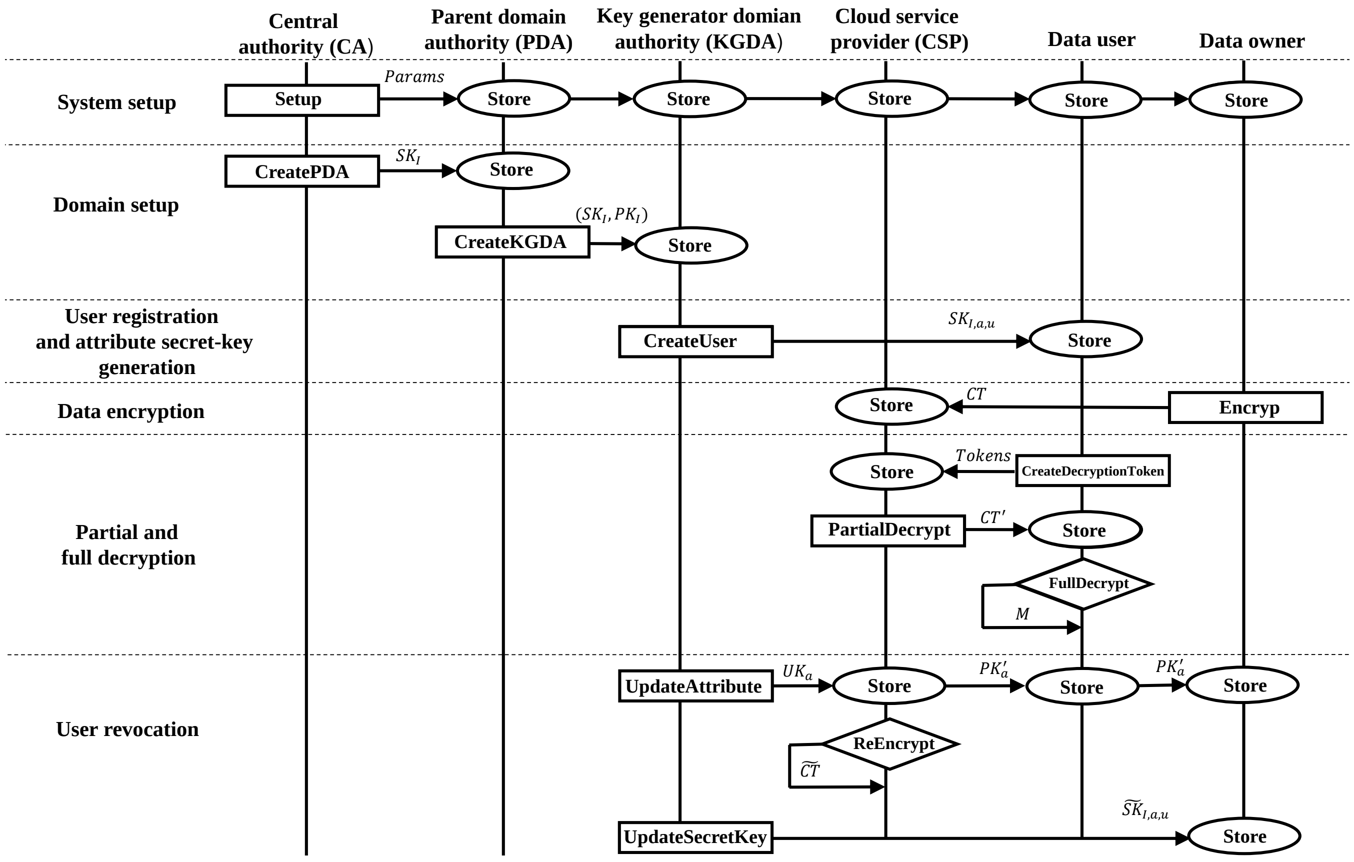

Figure 3: Workflow of our proposed scheme

4.3. *Definition of our proposed FDR-CP-HABE system*

Our proposed FDR-CP-HABE scheme consists of the following eleven algorithms shown in Figure 3 which are classified in six phases: system setup, domain setup, user registration and attribute secret-key generation, data encryption, partial and full decryption, and user revocation.

4.3.1. *System setup*

This phase is managed by the CA by running the following algorithm.

- **Setup**(1^n): The CA takes security parameter 1^n as input and generates the global public-parameters, $params$, secret keys sk_a and K_a for each attribute a in the universal attribute set, and its own master secret-key MSK_1.

4.3.2. *Domain setup*

Domain authorities of the system administer this phase using the following algorithms:

- **CreatePDA**$\Big(Params, SK_{(i_1=1,i_2,\ldots,i_k)}, \{ID_{(i_1=1,i_2,\ldots,i_k,j)}\}_{j=1}^{t}\Big)$: The algorithm is run by the CA or a PDA with identifier $ID_{(i_1=1,i_2,\ldots,i_k)}$. The inputs of the algorithm are global public-parameters, $params$, the secret-key of the domain authority running the algorithm, $SK_{(i_1=1,i_2,\ldots,i_k)}$, and a set consists of $t \in \mathbb{N}$ unique identifiers of the new PDAs, $\{ID_{(i_1=1,i_2\ldots,i_k,j)}\}_{j=1}^{t}$. The algorithm outputs a set consists of t secret-keys of the new PDAs, $\{SK_{(i_1=1,i_2,\ldots,i_k,j)}\}_{j=1}^{t}$.

- **CreateKGDA**$\Big(params, SK_{(i_1=1,\ldots,i_k)}, \{ID_{(i_1=1,\ldots,i_k,j)}\}_{j=1}^{t}\Big)$: The algorithm is run by a PDA. The inputs of this algorithm are system public-parameters *params*, the secret-key of the PDA, $SK_{(i_1=1,\ldots,i_k)}$, and $t \in \mathbb{N}$ identifiers of the new KGDAs, $\{ID_{(i_1,\ldots,i_k,j)}\}_{j=1}^{t}$. The algorithm outputs the secret-key set $\{SK_{(i_1=1,\ldots,i_k,j)}\}_{j=1}^{t}$ of the new KGDAs.

4.3.3. *User registration and attribute secret-key generation*

The attribute secret-keys of data users are delegated in this phase by the following algorithm:

- **CreateUser**$\Big(params, SK_{(i_1=1,i_2,\ldots,i_k)}, ID_u, sk_a\Big)$: This algorithm is conducted by a KGDA takes public-parameters of the system, *params*, secret-key of the KGDA, $SK_{(i_1=1,i_2,\ldots,i_k)}$, a data user's identifier, ID_u, and secret-key of an attribute a, sk_a, as inputs. It outputs the attribute secret-key of data user u, $SK_{(i_1=1,i_2,\ldots,i_k),a,u}$, corresponding to the KGDA and attribute a.

4.3.4. *Data encryption*

In this phase, using the following algorithm a data file is encrypted before outsourcing to the cloud.

- **Encrypt**$(params, M, \mathcal{T})$: Data owner runs this algorithm. The inputs of the algorithm are system public-parameters *params*, a message M, an access tree $\mathcal{T}$. The algorithm outputs the corresponding ciphertext CT.

4.3.5. *Partial and full decryption*

This phase is managed by data users and the CSP using the following three algorithms:

- **CreateDecryptionToken(CDT)** $(params, CT, AS)$: The algorithm is run by a data user. The inputs of the algorithm are system public-parameters *params*, a ciphertext CT, and a set of the data user's attribute secret-keys AS. The algorithm returns a partial decryption key (PDK) d, two decryption token sets $TS^{(1)}$, $TS^{(2)}$, and three tokens $T^{(3)}$, $T^{(4)}$ and $T^{(5)}$ as outputs.

- **PartialDecrypt**$\Big(params, CT, TS^{(1)}, TS^{(2)}, T^{(3)}, T^{(4)}, T^{(5)}\Big)$: The algorithm is run by the CSP . The inputs of the algorithm are system public-parameters *params*, a ciphertext CT, and decryption tokens $TS^{(1)}, TS^{(2)}, T^{(3)}$, $T^{(4)}$ and $T^{(5)}$ obtained from a data user. The output of the algorithm is a partially decrypted ciphertext CT'.

- **FullDecrypt**$(params, CT, d, CT')$: This algorithm is run by a data user. The inputs of the algorithm are system public-parameters *params*, a ciphertext CT, a PDK d obtained from **CDT**$(params, CT, AS)$, and the partially decrypted ciphertext CT'. The algorithm outputs the message corresponding to CT if and only if the attribute set corresponding to AS satisfies the access tree of ciphertext CT.

4.3.6. *User revocation*

This phase can be defined by the following three algorithms:

- **UpdateAttribute**$(params, ID_u, a, sk_a)$: This algorithm is run by KGDAs of the system. The inputs of the algorithm are system public-parameters *params*, a data user identifier ID_u, an attribute a revoked from the data user, and the secret-key of the attribute, sk_a. The algorithm outputs the updated secret-key and public-key of the attribute, sk'_a and PK'_a, and an update-key UK_a for re-encrypting the ciphertexts that are relevant to the revoked attribute and updating secret-keys of the other data users.

- **ReEncrypt**$(params, a, UK_a, CT)$: The algorithm is run by the CSP. The inputs of the algorithm are system public-parameters *params*, an attribute a revoked from a data user, an update key UK_a, and a ciphertext CT that its access tree has a leaf-node corresponding to the attribute a. The output of the algorithm is a re-encrypted cipherthext $\widetilde{CT}$.

- **UpdateSecretKey (USK)**$(params, a, UK_a, SK_{(i_1=1,i_2,\ldots,i_k),a,u}, ID_u)$: The algorithm can be run by any KGDA. The algorithm takes system public-parameters, $params$, an attribute a, an update key UK_a, a data user's attribute secret-key $SK_{(i_1=1,i_2,\ldots,i_k),a,u}$, and the data user identifier ID_u. It returns an updated secret-key $\widetilde{SK}_{(i_1=1,i_2,\ldots,i_k),a,u}$ of the data user.

4.4. Security model

The CA and the other domain authorities are assumed trusted. They never collude with the CSP and users. Similar to [4], [21] and [22] the CSP is assumed semi-trusted. It always execute the given protocols correctly. However, it sometimes try to find further information about the stored data. Also, to get illegal access privileges, it may collude with some data users. All the data users are assumed malicious. They may collude with each other and the CSP to get some unauthorized information about the outsourced data. Also, it is assume that there are some secure communication channels between each domain authority and the other existing parties (domain authorities, users and the CSP). Moreover, the communication channels between users (data users and data owners) and the CSP are assumed insecure.

4.5. Design goals

4.5.1. Fine-grained data access control

In practice, there are several data owners that outsource their sensitive data to the CSP and several data users that are interested to access to the outsourced data. It this case, data owners should be able to determine who has access to their data and who does not have. So, they should be enabled to define a flexible access structure on their data such that the access structure identifies the data users who have the privilege to access the outsourced data.

4.5.2. Data confidentiality

Our proposed scheme must be semantically secure. In other words, unauthorized data users must be unable to learn any partial information about the plaintext from the corresponding ciphertexts.

4.5.3. Collusion resistance

Some unauthorized data users may collude to decrypt some outsourced encrypted data such that non of them can do it alone. Our scheme must be resistant against such collision attacks.

4.5.4. Scalable and flexible key delegation and user revocation mechanisms

Our scheme should provide a high level of flexibility and scalability in key delegation and user revocation mechanisms. In order to achieve this goal, our proposed scheme should be fully distributed as we have presented in Section 4.1.

4.5.5. High performance

The computational and storage cost on the data user side should be low enough such that data users can easily store and access their data anytime and anywhere without having a powerful device.

4.6. Security definition of an FDR-CP-HABE scheme

We define the semantic security for an FDR-CP-HABE scheme in terms of the following chosen-plaintext attack (CPA) indistinguishability experiment.

Consider the following experiment, denoted by $\mathrm{Pub}^{cap}_{\mathcal{A},\Pi}(n)$, where Π is an FDR-CP-HABE scheme, $\mathcal{A}$ is a probabilistic polynomial time (PPT) adversary, and n is an integer-valued as the security parameter. The experiment consists of the following five steps:

The CPA indistinguishability experiment $\mathbf{Pub}^{cap}_{\mathcal{A},\Pi}(n)$:

1. **Setup:** Challenger $\mathcal{C}$, at first, runs the **Setup**, **CreatePDA** and **CreateKGDA** algorithms, and gives the adversary $\mathcal{A}$ the generated system public-parameters *params* and the identifiers of the created domain authorities.
2. **Phase 1:** Adversary $\mathcal{A}$ can adaptively make a polynomial number of queries for secret-keys of some arbitrary data users corresponding to his/her attributes generated by some KGDAs. Once challenger $\mathcal{C}$ receives a query, it runs **CreateUser** algorithm and gives the requested secret-key to the adversary.
3. **Challenge:** Adversary $\mathcal{A}$ declares an access tree $\mathcal{T}$ and two equal length messages M_0 and M_1 such that there is no any data user that the adversary have queried a set of attribute secret-keys of him/her that the corresponding attribute set satisfies the access tree. Challenger $\mathcal{C}$ flips a random coin $b \in \{0,1\}$ and then encrypts M_b under the access tree. The generated ciphertext is returned to the adversary.
4. **Phase 2:** The adversary can adaptively make new secret-key queries, with the same mentioned constraint in Phase 1, and the challenger responds them same as before.
5. **Guess:** The adversary outputs a guess $b' \in \{0,1\}$ of b.
   The output of the experiment is 1, $\text{Pub}^{cap}_{\mathcal{A},\Pi}(n) = 1$, if $b = b'$; otherwise the output is 0, $\text{Pub}^{cap}_{\mathcal{A},\Pi}(n) = 0$. we say the adversary succeeds, if $\text{Pub}^{cap}_{\mathcal{A},\Pi}(n) = 1$.

Definition 1. *An FDR-CP-HABE scheme Π is called CPA-secure if for all probabilistic polynomial-time adversaries $\mathcal{A}$, there is a negligible function negl such that:*

$$Pr(Pub^{cpa}_{\mathcal{A},\Pi}(n) = 1) \leq \frac{1}{2} + negl(n), \tag{2}$$

where n is the security parameter of the scheme, and the probability is taken over the randomness of the experiment, as well as the randomness used by the adversary $\mathcal{A}$.

5. Our construction

In this section, we present our proposed FDR-CP-HABE scheme in detail.

5.1. System setup

The CA produces the system public-parameters and some required keys using the following algorithm. It is assumed that any data user and domain authority has a unique identifier (ID).

Setup: According to the security parameter n, this algorithm at first selects a prime number q and then chooses $x, sk_0 \in \mathbb{Z}_q$ uniformly at random, two groups G_1 and G_2 of order q, a bilinear map $\hat{e} : G_1 \times G_1 \to G_2$ and three random generators $P_0, P_1, P_2 \in G_1$. Then, for each attribute a in the universal attribute set $\mathbb{U}$, two uniform elements $sk_a \in \mathbb{Z}_q$, $K_a \in \{0,1\}^n$, and $PK_a = sk_a P_0$ are considered as secret-keys and public-key of the attribute. After that, it chooses a pseudorandom function $F : \{0,1\}^* \times \{0,1\}^* \to \mathbb{Z}_q$ and a secure message authentication code scheme $\Pi_{mac} = (Gen, Mac, Vrfy)$ [23]. Finally, $params = (q, G_1, G_2, \hat{e}, n, P_0, P_1, P_2, Q_0, Q_1, g_0, g_1, \{PK_a\}_{a\in\mathbb{U}}, F, \Pi_{mac})$ is published as the system public-parameters, where $Q_0 = sk_0 P_0$, $Q_1 = xP_0$, $g_0 = \hat{e}(Q_0, P_1)$, and $g_1 = \hat{e}(Q_1, -Q_0)$. Also $MSK_1 = (sk_0, x, SK_1^1 = sk_0 P_1, SK_1^2 = xQ_0, \{sk_a\}_{a\in\mathbb{U}}, \{K_a\}_{a\in\mathbb{U}})$ will be kept secret as the master secret-key of the CA.

5.2. *Domain setup*

The CA or any PDA may run the **CreatePDA** algorithm to create some new PDAs. Let us label CA with 1 and its N_1 subdomains with $(1,1)$,$(1,2)$,…,$(1,N_1)$, from left to right. Also, the j-th subdomain of a PDA with label $I = (i_1 = 1, i_2, \ldots, i_k)$ is labeled with $I||j = (i_1 = 1, i_2, \ldots, i_k, j)$.

CreatePDA: The CA or a PDA with label $I = (i_1 = 1, i_2, \ldots, i_k)$ and secret-keys SK_I^1, SK_I^2, $\{sk_a\}_{a\in\mathbb{U}}$ and $\{K_a\}_{a\in\mathbb{U}}$ can run this algorithm and create some new PDAs. This algorithm selects a uniform element $sk_I \in \mathbb{Z}_q$ and outputs:

$$SK_{I||j} = (SK_{I||j}^1 = SK_I^1 + sk_I ID_{I||j}, SK_{I||j}^2 = SK_I^2 + sk_I ID_{I||j}, \{sk_a\}_{a\in\mathbb{U}}, \{K_a\}_{a\in\mathbb{U}}), \tag{3}$$

as the secret-key of the j-th created PDA, where $ID_{I||j} \in G_1$ is the identifier of the j-th created PDA.

Any PDA which have not created any subdomains can run the **CreateKGDA** algorithm and create an arbitrary number of KGDA.

CreateKGDA: A PDA with label I and secret-key $SK_I = (SK_I^1, SK_I^2, \{sk_a\}_{a\in U}, \{K_a\}_{a\in\mathbb{U}})$ which wants to create some KGDAs, at first chooses a uniform element $sk_I \in \mathbb{Z}_q$, and generates the secret-key

$$SK_{I||j} = (SK'_{I||j} = SK_I^1 + sk_I ID_{I||j}, \{sk_a\}_{a\in\mathbb{U}}, \{K_a\}_{a\in\mathbb{U}}) \tag{4}$$

and the public-key

$$PK_{I||j} = SK_I^2 + sk_I ID_{I||j} \tag{5}$$

for the j-th created KGDA, where $I||j$ and $ID_{I||j} \in G_1$ are the label and identifier of the j-th created KGDA.

5.3. *User registration and generation of attribute secret-keys*

When a data user u joins to the system, he/she should choose a unique identifier ID_u. When the data user asks a KGDA the corresponding secret-key of an attribute a. The KGDA at first checks whether the data user is eligible for the attribute or not. If so, then the KGDA runs **CreateUser** algorithm and generates the requested secret-key.

CreateUser: Let a data user u with identifier ID_u has made a query to a KGDA with label I and secret-key $(SK'_I = SK_I^1 + sk_I ID_I, \{sk_a\}_{a\in\mathbb{U}}, \{K_a\}_{a\in\mathbb{U}})$ for the secret-key corresponding to an attribute a. The KGDA runs this algorithm and computes the requested secret-key as follows:

$$SK_{I,a,u} = SK'_I + sk_a ID_u. \tag{6}$$

In order to ease notation, for a label $I = (i_1 = 1, i_2, \ldots, i_k)$ of a KGDA and any $l \in \{1, \ldots, k\}$, we assume that $I^l = (i_1, \ldots, i_l)$. Therefore, $I^1 = i_1 = 1$ and $I^k = I$. It is not hard to verify that, by combining Equations (3), (4), (5) and (6), and using the new notation, we have:

$$SK_{I,a,u} = SK_1^1 + \sum_{l=1}^{k-1} sk_{I^l} ID_{I^{l+1}} + sk_a ID_u \tag{7}$$

and

$$PK_I = SK_1^2 + \sum_{l=1}^{k-1} sk_{I^l} ID_{I^{l+1}}, \tag{8}$$

where $ID_{I^{l+1}}$ is the identifier of the domain authority with label I^{l+1}.

5.4. *Data encryption*

Before outsourcing a data file to the CSP, for controlling the access rights, the data owner should define an access structure on his/her data, in terms of an access tree, and encrypt the data under the access tree. This process can be described in detail as follows:

Encryption: For encrypting a data file $M \in G_1$, the data owner at first defines an access tree $\mathcal{T}$ with root R and a threshold value k_x for each node x in the tree. Then, he/she sets $q_R(x) = r + b_1 x, \ldots, b_{k_R-1} x^{k_R-1}$, where r and b_i are uniform elements of $\mathbb{Z}_q$, $i = 1, \ldots, k_R - 1$. In a similar way, a polynomial q_{x_i} of degree $k_{x_i} - 1$ that $q_{x_i}(0) = q_R(i)$ is selected for the i-th child of R. This process is continued until each node of the tree gets one polynomial. As we noted in Section 3.2, for each leaf node y of the tree, the threshold value is equal to one. Therefore, q_y is a constant polynomial. Let y_a be the leaf node of the tree corresponding to the attribute a and $L_{\mathcal{T}}$ be the leaf node set of tree $\mathcal{T}$. The output of the algorithm is:

$$CT = (\mathcal{T}, V = M.g_0^r, V' = g_1^r, C = rP_0, C' = rP_2, \{C_a = q_{y_a}P_0, C'_a = q_{y_a}(PK_a - P_2)\}_{y_a \in L_{\mathcal{T}}}) \tag{9}$$

5.5. *Partial and full decryption*

As we mentioned in Section 4.5.5, one of our main goals in designing our FDR-CP-HABE scheme is to decrease the computational costs on the data user side. In other words, we want to provide a situation in which even data users with low computing resources can easily use the scheme. For this purpose, in our scheme, the CSP does the most amount of the required computations in decryption algorithm and very few computations are done by the data user.

When a data user wants to decrypt a ciphertext, he first runs **CreateDecryptionToken(CDT)** algorithm and generates some decryption tokens for the CSP such that they do not leak any partial information about the corresponding plaintext and the data user's secret-keys. Then, using the tokens, the CSP runs **PartialDecrypt** algorithm and produces a partially decrypted ciphertext for the data user. Finally, using the partially decrypted ciphertext and running the **FullDecrypt** algorithm, the data user obtains the corresponding plaintext. Also, as we will note in Remark 2, a data user with the required computational resources can decrypt the ciphertext without the contribution of the CSP.

CreateDecryptionToken(CDT): Given a ciphertext $CT = (\mathcal{T}, V, V', C, C', \{C_a, C'{}_a\}_{y_a \in L_{\mathcal{T}}})$, a data user with identifier ID_u and an attribute set $S = \{a_i\}_{i=1}^t$ that for any attribute a_i, $i \in \{1, \ldots, t\}$, there is a leaf nod $y_{a_i} \in L_{\mathcal{T}}$. This algorithm outputs a uniform element d as partial decryption key (PDK), two token sets $TS^{(1)} = \left\{T_{a_i}^{(1)} = d.SK_{I_i,a_i,u}\right\}_{i=1}^t$, $TS^{(2)} = \left\{T_{a_i}^{(2)} = d.C_{a_i}\right\}_{i=1}^t$, and three tokens $T^{(3)} = dID_u$, $T^{(4)} = dQ_0$ and $T^{(5)} = V'^d$, where $SK_{I_i,a_i,u}$ is the secret-key of data user u corresponding to the attribute a_i and generated by the KGDA with label $I_i = (1, \ldots, k_i)$.

For a ciphertext $CT = (\mathcal{T}, V, V', C, C', \{C_a, C'_a\}_{y_a \in L_{\mathcal{T}}})$, a node y of $\mathcal{T}$, and a data user u, let

$$B_{y,u} = g_0^{q_y(0)}.g_1^{q_y(0)}.\hat{e}(P_2, ID_u)^{q_y(0)},$$

where q_y is the assigned polynomial to the node y in the encryption phase.

PartialDecrypt: Given a ciphertext CT with access structure $\mathcal{T}$ satisfied by a data user's attribute set $S = \{a_i\}_{i=1}^t$ and the decryption tokens $TS^{(1)} = \left\{T_{a_i}^{(1)}\right\}_{i=1}^t$, $TS^{(2)} = \left\{T_{a_i}^{(2)}\right\}_{i=1}^t$, $T^{(3)}$, $T^{(4)}$ and $T^{(5)}$ obtained from **CDT** algorithm. The algorithm at first computes:

$$
\begin{aligned}
B^d_{y_{a_i},u} &= \frac{\hat{e}(C_{a_i}, T^{(1)}_{a_i})}{\hat{e}(T^{(2)}_{a_i}, PK_{I_i})\hat{e}(C'_{a_i}, T^{(3)}))} \\
&= g_0^{dq_{y_{a_i}}} . g_1^{dq_{y_{a_i}}} . \hat{e}(P_2, T^{(3)})^{q_{y_{a_i}}}, \qquad (10)
\end{aligned}
$$

for any $i \in \{1, \ldots, t\}$. Then, it calculates $B_{R,u}$ in the following way, where R is the root node of the access tree $\mathcal{T}$:

Since the access tree $\mathcal{T}$ is satisfied by S, there are some internal nodes y as the parent of leaf-nodes $y_{a_{i_1}}, \ldots, y_{a_{i_{k_y}}}$ in the access tree, where k_y is the threshold value of node y . Now, if q_y is the assigned polynomial to a node y of the tree in the encryption phase, then by using the interpolation method we have $q_y(x) = \sum_{s=1}^{k_y} l_{i_s}(x) q_y(i_s) = \sum_{s=1}^{k_y} l_{i_s}(x) q_{y_{a_{i_s}}}(0)$, where

$$
l_{i_s}(x) = \frac{(x - i_1) \ldots (x - i_{s-1})(x - i_{s+1}) \ldots (x - i_{k_{y_i}})}{(i_s - i_1) \ldots (i_s - i_{s-1})(i_s - i_{s+1}) \ldots (i_s - i_{k_{y_i}})}
$$

is the Lagrange polynomial, $s = 1, \ldots, k_{y_i}$.

So, the value $B^d_{y,u}$ can be computed as follows:

$$
\begin{aligned}
B^d_{y,u} &= \prod_{s=1}^{k_{y_i}} (B^d_{y_{a_{i_s}},u})^{l_{i_s}(0)} \\
&= g_0^{dq_y(0)} . g_1^{dq_y(0)} . \hat{e}(P_2, T^{(3)})^{q_y(0)}. \qquad (11)
\end{aligned}
$$

After that, in a similar way, the CSP computes $B^d_{y',u}$ for some nodes y' in the next level of the access tree, by using some values $B^d_{y_1,u}, \ldots, B^d_{y_{k_{y'}},u}$ obtained before, where the nodes $y_1, \ldots, y_{k_{y'}}$ are children of the node y'. Continuing this process, the CSP calculates:

$$
\begin{aligned}
B^d_{R,u} &= g_0^{dq_R(0)} . g_1^{dq_R(0)} . \hat{e}(P_2, T^{(3)})^{q_R(0)} \\
&= g_0^{dr} . g_1^{dr} . \hat{e}(P_2, T^{(3)})^r \\
&= g_0^{dr} . T^{(5)} . \hat{e}(C', T^{(3)}) \qquad (12)
\end{aligned}
$$

and the partially decrypted ciphertext

$$
\begin{aligned}
CT' &= \frac{B^d_{R,u}}{T^{(5)} . \hat{e}(C', T^{(3)})} \qquad (13) \\
&= g_0^{dr},
\end{aligned}
$$

is returned as the output.

FullDecrypt: Once the data user receives partially decrypted ciphertext CT', he/she can recover the message as follows:

$$
M = \frac{V}{CT'^{\frac{1}{d}}}. \qquad (14)
$$

Remark 2. *A data user u who has an attribute set S satisfying the access tree of a given ciphertext $CT = (\mathcal{T}, V, V', C, C', \{C_a, C'_a\}_{y_a \in L_{\mathcal{T}}})$ can decrypt CT without getting help from the CSP. In this case, the data user can recover the file as follows:*

- *First, the data users calculates*
$$B_{y_{a_i},u} = \frac{\hat{e}(C_{a_i}, SK_{I^{(i)},a_i,u})}{\hat{e}(C_{a_i}, PK_{I^{(i)}}).\hat{e}(C'_{a_i}, ID_u)} = g_0^{q_{y_{a_i}}}.g_1^{q_{y_{a_i}}}.\hat{e}(P_2, ID_u))^{q_{y_{a_i}}},$$
for each attribute $a_i \in S$.
- *Then, as before, by using the interpolation method, the following expression can be calculated*
$$\begin{aligned} B_{R,u} &= g_0^r.g_1^r.\hat{e}(P_2, ID_u))^r \\ &= g_0^r.g_1^r.\hat{e}(rP_2, ID_u) \\ &= g_0^r.V'.\hat{e}(C', ID_u). \end{aligned}$$
Then, the corresponding message is recovered as follows:
$$M = \frac{V}{\frac{B_{R,u}}{V'.\hat{e}(C', ID_u)}}.$$

Remark 3. *The most impressive benefit of a FDR-CP-HABE can be observed in the decryption process. As we have seen in this section, despite to the other existing works, the secret-keys of a data user gotten from different domain authorities can be used together in the decryption process. This option offers a high level of scalability and flexibility in the key delegation mechanism.*

5.6. User revocation

In real applications, some attributes might be revoked from a data user. To achieve forward secrecy, whenever an attribute is revoked from a data user, it should be made sure that the data user no longer has access to the associated shared data.

For these purposes, KGDAs update secret-key and public-key of the revoked attribute. Also, they update attribute secret-keys of the authorized data users. Furthermore, the related ciphertexts should be re-encrypted by the CSP. The revocation process is described through the following three algorithms.

UpdateAttribute: For an attribute a_0 with secret-keys sk_{a_0} and K_{a_0}, and public-key PK_{a_0}, once a KGDA judges that a data user u_0 no longer is eligible for the attribute, it selects a random integer number s and broadcasts the message $R(a_0, u_0) = (a_0, ID_{u_0}, s, time, \mathbf{Mac}_{K_{a_0}}(ID_{u_0}, s, time))$, where $time$ is a time stamp. Also, it sends an update-key $UK_{a_0} = F_{K_{a_0}}(s) - sk_{a_0}$ to the CSP through an authenticated secure channel, where $F_{K_{a_0}}(s) = F(K_{a_0}, s)$. Any KGDA, once receives the message $R(a_0, u_0)$, using the **Vrfy** algorithm checks the validity of the message. If so, they substitute sk_{a_0} and PK_{a_0} with $sk'_a = F_{K_{a_0}}(s) = F(K_{a_0}, s)$ and $PK'_a = sk'_a P_0$, respectively, and compute $UK_{a_0} = sk'_{a_0} - sk_{a_0}$.

ReEncrypt: For a revoked attribute a_0, once the CSP receives the update-key UK_{a_0} from a KGDA, any ciphertext $CT = (\mathcal{T}, V, V', C, C', \{C_a, C'_a\}_{y_a \in L_{\mathcal{T}}})$ that its access tree $\mathcal{T}$ has a leaf node y_{a_0} corresponding to attribute a_0 is re-encrypted as $\widetilde{CT} = (\mathcal{T}, V, C, C_{a_0}, \widetilde{C}'_{a_0} = C'_{a_0} + UK_{a_0}C_{a_0}, \{C_a, C'_a\}_{y_a \in L_{\mathcal{T}} - \{y_{a_0}\}})$. Then CSP stores $\widetilde{CT}$ and removes the old version of the ciphertext.

UpdateSecretKey(USK): When an authorized data user u makes a query for updating his/her attribute secret-key $SK_{I,a_0,u}$ to a KGDA, the KGDA updates his/her secret-key as $\widetilde{SK}_{I,a_0,u} = SK_{I,a_0,u} + UK_{a_0}ID_u$ and returns the result to him/her.

Remark 4. *In the revoking process one can see another benefit of a FDR-CP-HABE scheme. Same as the key delegation phase, in this case, the parameters of the revoked attributes and also the corresponding secret-keys of the authorized data users can be updated by any KGDA. Indeed, each KGDA can update secret-keys of any data user, even if the KGDA has not generated them.*

5.7. Correctness analysis of our proposed scheme

In this section, we show the correctness of our proposed scheme.

Theorem 5. *The decryption process is correct.*

Proof. Consider a ciphertext $CT = (\mathcal{T}, V, V', C, C', \{C_a, C'_a\}_{y_a \in L_{\mathcal{T}}})$. Let $TS^{(1)}$, $TS^{(2)}$, $T^{(3)}$, $T^{(4)}$, $T^{(5)}$, d, $S = \{a_i\}_{i=1}^{t}$, I_i and $SK_{I_i,a_i,u}$ be as in Section 5.5, $i = 1, \dots t$. In the following, we first prove the correctness of **PartialDecrypt** algorithm. For this purpose, we show the correctness of Equations (10), (11), (12) and (13). We have:

$$\begin{aligned} B^d_{y_{a_i},u} &= \frac{\hat{e}(C_{a_i}, T^{(1)}_{a_i})}{\hat{e}(T^{(2)}_{a_i}, PK_{I_i}).\hat{e}(C'_{a_i}, T^{(3)})} \\ &= \frac{\hat{e}(C_{a_i}, dSK_{I_i,a_i,u})}{\hat{e}(T^{(2)}_{a_i}, PK_{I_i}).\hat{e}(C'_{a_i}, T^{(3)})} \end{aligned} \tag{15}$$

Using Equation (7), we thus have:

$$\begin{aligned} B^d_{y_{a_i},u} &= \frac{\hat{e}(C_{a_i}, d(SK^1_1 + \sum_{l=1}^{k_i-1} sk_{I^l_i} ID_{I^{l+1}_i} + sk_{a_i} ID_u))}{\hat{e}(T^{(2)}_{a_i}, PK_{I_i}).\hat{e}(C'_{a_i}, T^{(3)})} \\ &= \frac{\hat{e}(C_{a_i}, dSK^1_1).\hat{e}(C_{a_i}, d\sum_{l=1}^{k_i-1} sk_{I^l_i} ID_{I^{l+1}_i}).\hat{e}(C_{a_i}, dsk_{a_i} ID_u)}{\hat{e}(T^{(2)}_{a_i}, PK_{I_i}).\hat{e}(C'_{a_i}, T^{(3)})} \\ &= \frac{\hat{e}(C_{a_i}, dSK^1_1).\hat{e}(C_{a_i}, d(\sum_{l=1}^{k_i-1} sk_{I^l_i} ID_{I^{l+1}_i} + xQ_0 - xQ_0)).\hat{e}(C_{a_i}, dsk_{a_i} ID_u)}{\hat{e}(T^{(2)}_{a_i}, PK_{I_i}).\hat{e}(C'_{a_i}, T^{(3)})} \\ &= \frac{\hat{e}(dC_{a_i}, SK^1_1).\hat{e}(dC_{a_i}, (\sum_{l=1}^{k_i-1} sk_{I^l_i} ID_{I^{l+1}_i} + xQ_0 - xQ_0)).\hat{e}(dC_{a_i}, sk_{a_i} ID_u)}{\hat{e}(T^{(2)}_{a_i}, PK_{I_i}).\hat{e}(C'_{a_i}, T^{(3)})} \end{aligned} \tag{16}$$

$$= \frac{\hat{e}(dC_{a_i}, SK^1_1).\hat{e}(T^{(2)}_{a_i}, \sum_{l=1}^{k_i-1} sk_{I^l_i} ID_{I^{l+1}_i} + SK^2_1 - xQ_0).\hat{e}(dq_{y_{a_i}} P_0, sk_{a_i} ID_u)}{\hat{e}(T^{(2)}_{a_i}, PK_{I_i}).\hat{e}(C'_{a_i}, T^{(3)})} \tag{17}$$

Now, using Equation (8), we see that:

$$\begin{aligned}
B^d_{y_{a_i},u} &= \frac{\hat{e}(dq_{y_{a_i}}P_0, sk_0P_1).\hat{e}(T^{(2)}_{a_i}, PK_{I_i} - xQ_0).\hat{e}(dq_{y_{a_i}}P_0, sk_{a_i}ID_u)}{\hat{e}(T^{(2)}_{a_i}, PK_{I_i}).\hat{e}(C'_{a_i}, T^{(3)})} \\
&= \frac{\hat{e}(dq_{y_{a_i}}P_0, sk_0P_1).\hat{e}(T^{(2)}_{a_i}, PK_{I_i}).\hat{e}(dq_{y_{a_i}}P_0, -xQ_0).\hat{e}(dq_{y_{a_i}}P_0, sk_{a_i}ID_u)}{\hat{e}(T^{(2)}_{a_i}, PK_{I_i}).\hat{e}(C'_{a_i}, T^{(3)})} \\
&= \frac{\hat{e}(P_0, sk_0P_1)^{dq_{ya_i}}.\hat{e}(P_0, -xQ_0)^{dq_{ya_i}}.\hat{e}(q_{y_{a_i}}sk_{a_i}P_0, dID_u)}{\hat{e}(C'_{a_i}, T^{(3)})} \\
&= \frac{\hat{e}(sk_0P_0, P_1)^{dq_{ya_i}}.\hat{e}(xP_0, -Q_0)^{dq_{ya_i}}.\hat{e}(q_{y_{a_i}}sk_{a_i}P_0, T^{(3)})}{\hat{e}(q_{y_{a_i}}(PK_{a_i} - P_2), T^{(3)})} \\
&= \frac{\hat{e}(Q_0, P_1)^{dq_{ya_i}}.\hat{e}(Q_1, -Q_0)^{dq_{ya_i}}.\hat{e}(q_{y_{a_i}}PK_{a_i}, T^{(3)})}{\hat{e}(q_{y_{a_i}}(PK_{a_i} - P_2), T^{(3)})} \\
&= \frac{\hat{e}(Q_0, P_1)^{dq_{ya_i}}.\hat{e}(Q_1, -Q_0)^{dq_{ya_i}}.\hat{e}(q_{y_{a_i}}PK_{a_i}, T^{(3)})}{\hat{e}(-q_{y_{a_i}}P_2, T^{(3)}).\hat{e}(q_{y_{a_i}}PK_{a_i}, T^{(3)})} \\
&= \frac{\hat{e}(Q_0, P_1)^{dq_{ya_i}}.\hat{e}(Q_1, -Q_0)^{dq_{ya_i}}}{\hat{e}(P_2, T^{(3)})^{-q_{ya_i}}} \\
&= \hat{e}(Q_0, P_1)^{dq_{ya_i}}.\hat{e}(Q_1, -Q_0)^{dq_{ya_i}}.\hat{e}(P_2, T^{(3)})^{q_{ya_i}} \\
&= g_0^{dq_{ya_i}}.g_1^{dq_{ya_i}}.\hat{e}(P_2, T^{(3)})^{q_{ya_i}}. \qquad (18)
\end{aligned}$$

So, Equation (10) is correct. On the other hand,

$$\begin{aligned}
B^d_{y_i,u} &= \prod_{s=1}^{k_{y_i}} (B_{y_{a_{i_s}},u})^{dl_{i_s}(0)} \\
&= \prod_{s=1}^{k_{y_i}} g_0^{dl_{i_s}(0)q_{ya_i}}.g_1^{dl_{i_s}(0)q_{ya_i}}.\hat{e}(P_2, T^{(3)})^{l_{i_s}(0)q_{ya_i}} \\
&= g_0^{d\sum_{s=1}^{k_{y_i}} l_{i_s}(0)q_{ya_i}}.g_1^{d\sum_{s=1}^{k_{y_i}} l_{i_s}(0)q_{ya_i}}.\hat{e}(P_2, T^{(3)})^{\sum_{s=1}^{k_{y_i}} l_{i_s}(0)q_{ya_i}} \\
&= g_0^{dq_{y_i}(0)}.g_1^{dq_{y_i}(0)}.\hat{e}(P_2, T^{(3)})^{q_{y_i}(0)} \qquad (19)
\end{aligned}$$

which proves the correctness of Equation (11). Also,

$$\begin{aligned}
B^d_{R,u} &= g_0^{dq_R(0)}.g_1^{dq_R(0)}.\hat{e}(P_2, T^{(3)})^{q_R(0)} \\
&= g_0^{dr}.g_1^{dr}.\hat{e}(P_2, T^{(3)})^r \\
&= g_0^{dr}.V'^d.\hat{e}(rP_2, T^{(3)}) \\
&= g_0^{dr}.T^{(5)}.\hat{e}(C', T^{(3)}). \qquad (20)
\end{aligned}$$

Moreover,

$$\begin{aligned} CT' &= \frac{B_{R,u}^{d}}{T^{(5)}.\hat{e}(C', T^{(3)})} \\ &= \frac{g_0^{dr}.T^{(5)}.\hat{e}(C', T^{(3)})}{T^{(5)}.\hat{e}(C', T^{(3)})} \\ &= g_0^{rd}. \end{aligned} \tag{21}$$

Therefore, Equations (12) and (13) are correct.

Now, in order to show the correctness of the **FullDecrypt** algorithm, we should prove Equation (14). Consider the partially decrypted ciphertext CT' and **FullDecrypt** algorithm. We see that:

$$\begin{aligned} \frac{V}{CT'^{\frac{1}{d}}} &= \frac{M.g_0^r}{(g_0^{dr})^{\frac{1}{d}}} \\ &= \frac{M.g_0^r}{g_0^r} \\ &= M. \end{aligned} \tag{22}$$

This proves the theorem. □

Theorem 6. *The re-encryption process is correct.*

Proof. Let a_0, C_{a_0}, C'_{a_0}, $\widetilde{C}'_{a_0}$, UK_{a_0}, sk_{a_0}, sk'_{a_0}, PK_{a_0} and PK'_{a_0} be the same as Section 5.6. To show the correctness of the re-encryption process, we should prove that $\widetilde{C}'_{a_0} = q_{y_{a_0}}(PK'_{a_0} - P_2)$. We have:

$$\begin{aligned} \widetilde{C}'_{a_0} &= C'_{a_0} + UK_{a_0}C_{a_0} \\ &= q_{y_{a_0}}(PK_{a_0} - P_2) + (sk'_{a_0} - sk_{a_0})(q_{y_{a_0}}P_0) \\ &= q_{y_{a_0}}(sk_{a_0}P_0 - P_2) + (sk'_{a_0} - sk_{a_0})(q_{y_{a_0}}P_0) \\ &= q_{y_{a_0}}(sk'_{a_0}P_0 - P_2) \\ &= q_{y_{a_0}}(PK'_{a_0} - P_2). \end{aligned}$$

So, the re-encryption algorithm is correct.

□

6. Security analysis

In this section we show that our proposed scheme fulfills the mentioned security requirements in Section 4. We first prove that the scheme is adaptively semantically secure in the standard model. Then we conclude that the scheme is collusion resistance. After that, we show that it achieves fine-grained access control over the outsourced encrypted data.

6.1. Data confidentiality

Theorem 7. *If the DBDH problem is hard relative to two cyclic groups G_1, G_2 of a prim order q and a bilinear map $e : G_1 \times G_1 \to G_2$, then our construction is adaptively semantically secure in the standard model.*

Proof. Let Π denotes our proposed FDR-CP-HABE scheme. For a security parameter n, let $\mathcal{A}$ be a PPT adversary in the experiment $\mathrm{Pub}^{cap}_{\mathcal{A},\Pi}(n)$ introduced in Section 4.6. We prove that:

$$\Pr(\mathrm{Pub}^{cpa}_{\mathcal{A},\Pi}(n) = 1) \leq \frac{1}{2} + negl(n),$$

for a negligible function $negl$. Consider a PPT distinguisher $\mathcal{D}$ which attempts to solve the DBDH problem. The distinguisher $\mathcal{D}$ has gotten $(q, G_1, G_2, e, P, \alpha P, \beta P, \gamma P, h = \hat{e}(P,P)^z)$, where P is a randome generator of G_1, α, β and γ are three uniform elements of $\mathbb{Z}_q$, z is either $\alpha\beta\gamma$ or is a uniform element of $\mathbb{Z}_q$. $\mathcal{D}$ wants to determine the case of z.

The distinguisher $\mathcal{D}$ runs adversary $\mathcal{A}$ as a subroutine as follows:

1. $\mathcal{D}$ sets

$$P_0 = P \tag{23}$$
$$P_1 = \gamma P \tag{24}$$
$$Q_0 = \alpha P \tag{25}$$
$$Q_1 = \gamma P - t_1 P = (\gamma - t_1)P \tag{26}$$
$$P_2 = t_2 P_0 \tag{27}$$
$$g_0 = \hat{e}(Q_0, P_1) \tag{28}$$
$$g_1 = \hat{e}(Q_1, -Q_0), \tag{29}$$

   where t_1 and t_2 are two uniform elements of $\mathbb{Z}_q$ chosen by $\mathcal{D}$. Then, for each attribute a in the universal attribute set $\mathbb{U}$, it chooses $sk_a \in \mathbb{Z}_q$ uniformly at random, and gives the system public-parameters $params = \big(q, G_1, G_2, e, n, P_0, P_1, Q_0, Q_1, P_2, g_0, g_1, \{PK_a\}_{a \in \mathbb{U}}, F, \Pi_{mac}\big)$ to the adversary $\mathcal{A}$, where F is a a pseudorandom function and Π_{mac} is a secure MAC scheme.

2. Before starting **Phase 1**, the distinguisher $\mathcal{D}$ selects a label set L consists of some elements as $I = (i_1 = 1, i_2, \ldots, i_k)$, where $i_j \in \mathbb{Z}$, $j = 1, \ldots, k$, as the labels of the KGDAs of the system. For any $I \in L$, it chooses $ID_I \in G_1$ as the identifier of the KGDA with label I and gives it to the adversary $\mathcal{A}$. Then, for each $I \in L$, it selects a uniform element $A_I \in G_1$ and gives

$$PK_{I_0} = A_{I_0} - t_1 \alpha P_0 \tag{30}$$

   to adversary $\mathcal{A}$ as the public-key of the KGDA with label I. The value A_I is kept secret for each $I \in L$.

   In **phase 1**, when adversary $\mathcal{A}$ makes a query for a secret-key of a data user u_0 corresponding to an attribute a_0 generated by the KGDA with label I_0, distinguisher $\mathcal{D}$ responds to the query with

$$SK^1_{I_0, a_0, u_0} = A_{I_0} + sk_a ID_u. \tag{31}$$

   Remark 8. *In the construction introduced in Section 5, we had $Q_1 = xP_0$, $Q_0 = sk_0 P_0$, $SK^1_1 = sk_0 P_1$ and $SK^2_1 = xQ_0 = xsk_0 P_0$, where x and sk_0 are two uniform elements of $\mathbb{Z}_q$. So, by considering $Q_1 = (\gamma - t_1)P_0$, $Q_0 = \alpha P_0$ and $P_1 = \gamma P_0$, we get:*

$$x = \gamma - t_1,\ sk_0 = \alpha. \tag{32}$$

   Therefore, we have:

$$SK^1_1 = sk_0 P_1 = \alpha\gamma P_0 \tag{33}$$

   and

$$SK^2_1 = xQ_0 = xsk_0 P_0 = (\gamma - t_1)\alpha P_0 = \gamma\alpha P_0 - t_1 \alpha P_0. \tag{34}$$

Now, by combining Equations (7), (8), (33) and (34), we conclude that SK_{I_0,a_0,u_0} and PK_{I_0} must be in the following forms:

$$SK_{I_0,a_0,u_0} = \alpha\gamma P_0 + \sum_{l=1}^{k} sk_{I_0^l} ID_{I_0^{l+1}} + sk_a ID_u \tag{35}$$

and

$$PK_{I_0} = \alpha\gamma P_0 - t_1\beta P_0 + \sum_{l=1}^{k} sk_{I_0^l} ID_{I_0^{l+1}}. \tag{36}$$

On the other hand, the parameters $sk_{I_0^l}$, $l = 1, \dots, k$ are some arbitrary elements of $\mathbb{Z}_q$ and distinguisher $\mathcal{D}$ can freely select them. So, we can assume that they has been selected in such a way that $\alpha\gamma P_0 + \sum_{l=1}^{k} sk_{I_0^l} ID_{I_0^{l+1}} = A_{I_0}$. Therefore, $PK_{I_0} = A_{I_0} - t_1\alpha P_0$ is a valid public-key of the KGDA with label I_0 and $SK^1_{I_0,a_0,u_0} = A_{I_0} + sk_a ID_u$ is a valid responses to the query.

3. In **Challenge** step, adversary $\mathcal{A}$ outputs two equal length plaintexts M_0 and M_1, and an access tree $\mathcal{T}$ which the adversary $\mathcal{A}$ has not queried the attribute secret-keys of a specific data user corresponding to an attribute set S satisfying $\mathcal{T}$, in **Phase 1**. Once the distinguisher $\mathcal{D}$ receives the plaintexts, chooses $b \in \{0,1\}$ and encrypts m_b as follows:

   Firstly, k_R elements $b_1 \dots b_{k_R} \in \mathbb{Z}_q$ are chosen uniformly at random, where k_R is the threshold value of the access tree's root. Then, it calculates $sk_a(\beta P_0) - t_2(\beta P_0) = \beta(sk_a P_0 - t_2 P_0) = \beta(PK_a - P_2)$ and considers two following polynomial functions:

$$\begin{cases} Q_R : \mathbb{Z}_q \to G_1 \\ Q_R(x) = \beta P_0 + (b_1 P_0)x + \cdots + (b_{k_R-1} P_0)x^{k_R-1} \end{cases} \tag{37}$$

   and

$$\begin{cases} Q'_R : \mathbb{Z}_q \to G_1 \\ Q'_R(x) = \beta(PK_a - P_2) + b_1(PK_a - P_2)x + \cdots + b_{k_R-1}(PK_a - P_2)x^{k_R-1}, \end{cases} \tag{38}$$

   for the root of the tree. Then, in a similar way, two following polynomial

$$\begin{cases} Q_{c_i} : \mathbb{Z}_q \to G_1 \\ Q_{c_i}(x) = Q_R(i) + (b_1^{(i)} P_0)x + \cdots + (b_{k_{c_i}-1}^{(i)} P_0)x^{k_{c_i}-1} \end{cases} \tag{39}$$

   and

$$\begin{cases} Q'_{c_i} : \mathbb{Z}_q \to G_1 \\ Q'_{c_i}(x) = Q'_R(i) + (b_1^{(i)}(PK_a - P_2))x + \cdots + (b_{k_{c_i}-1}^{(i)}(PK_a - P_2))x^{k_{c_i}-1}, \end{cases} \tag{40}$$

   are generated for each i-th child c_i of the root.

   By continuing this process two polynomials Q_{y_i} and Q'_{y_i} with degree $k_{y_i} - 1$ are generated for any node y_i of the tree. As we noted, the threshold value of each leaf-node of an access tree is equal to one. Therefore, the corresponding polynomials to each leaf-node y_a are two constant polynomials $Q_{y_a}, Q'_{y_a} \in G_1$. Finally, the distinguisher $\mathcal{D}$ outputs

$$\begin{aligned} CT_b = (\mathcal{T}, V = M_b.h, V' = h^{-1}.\hat{e}(\alpha P_0, \beta P_0)^{t_1}, C = \beta P_0, C' = t_2(\beta P_0) = \beta(t_2 P_0) = \beta P_2, \\ \left\{C_a = Q_{y_a}, C'_a = Q'_{y_a}\right\}_{y_a \in L_{\mathcal{T}}}), \end{aligned} \tag{41}$$

Remark 9. *As we have mentioned in Section 5, the ciphertext corresponding to the message M_b and access tree $\mathcal{T}$ should be as follows:*

$$CT = (\mathcal{T}, V = M_b.\hat{e}(Q_0, P_1)^r, V' = \hat{e}(Q_1, -Q_0)^r, C = rP_0, C' = rP_2, \{C_a = q_{y_a}P_0, C'_a = q_{y_a}(PK_a - P_2)\}_{y_a \in L_{\mathcal{T}}}), \tag{42}$$

where $r \in \mathbb{Z}_q$ is a uniform element. Comparing Equations (41) and (42), we conclude that $rP_0 = \beta P_0$ and therefore $r = \beta$. Combining (42) with (24), (25), (26), and $r = \beta$, we get :

$$V = M_b.\hat{e}(Q_0, P_1)^r = M_b.\hat{e}(\alpha P_0, \gamma P_0)^\beta = M_b.\hat{e}(P_0, P_0)^{\alpha\beta\gamma} \tag{43}$$

and

$$V' = \hat{e}(Q_1, -Q_0)^r = \hat{e}((\gamma - t_1)P_0, -\alpha P_0)^\beta = \hat{e}(P_0, P_0)^{-\alpha\beta(\gamma - t_1)} = \hat{e}(P_0, P_0)^{-\alpha\beta\gamma}\hat{e}(\alpha P_0, \beta P_0)^{t_1}. \tag{44}$$

4. **Phase 2:** The adversary $\mathcal{A}$ queries more attribute secret-keys with the same constrains in **Challenge** step and $\mathcal{D}$ responds them as **Phase 1**.
5. In **Guess** step, $\mathcal{A}$ outputs $b' \in \{0, 1\}$ and $\mathcal{D}$ checks whether $b' = b$ or not. If so, $\mathcal{D}$ outputs 1. Otherwise, 0 is returned by $\mathcal{D}$.

From Equations (43) and (44) we see that if $h = \hat{e}(P_0, P_0)^{\alpha\beta\gamma}$, then the values assigned to V and V' in (41) are valid. Furthermore, it is clear that the components C, C', $\{C_a = Q_{y_a}, C'_a = Q'_{y_a}\}_{y_a \in L_{\mathcal{T}}}$, of CT_b are also chosen correctly.
Therefore, if $h = \hat{e}(P_0, P_0)^{\alpha\beta\gamma}$, then CT_b is a valid ciphertext of M_b and thus

$$\Pr(\mathcal{D}(P, \alpha P, \beta P, \gamma P, h = \hat{e}(P, P)^{\alpha\beta\gamma}) = 1) = \Pr(\mathrm{Pub}(n)^{cpa}_{\mathcal{A},\Pi} = 1). \tag{45}$$

Moreover, if $h = \hat{e}(P_0, P_0)^z$ for a uniform element $z \in \mathbb{Z}_q$, then V is also a uniform element of G_1 and therefore adversary $\mathcal{A}$ can not obtain any information about M_b. So, in this case, the returned bit b' is equal to b with probability $\frac{1}{2}$. Therefore,

$$\Pr(\mathcal{D}(P, \alpha P, \beta P, \gamma P, h = \hat{e}(P, P)^z) = 1) = \frac{1}{2}. \tag{46}$$

On the other hand, under the hardness assumption of $DBDH$ problem:

$$|\Pr(\mathcal{D}(P, \alpha P, \beta P, \gamma P, h = \hat{e}(P, P)^{\alpha\beta\gamma}) = 1) - \Pr(\mathcal{D}(P, \alpha P, \beta P, \gamma P, h = \hat{e}(P, P)^z) = 1)| \leq negl(n), \tag{47}$$

for a negligible function $negl$. Combining (45), (46) and (47), we see that:

$$\Pr(\mathrm{Pub}(n)^{cpa}_{\mathcal{A},\Pi} = 1) \leq \frac{1}{2} + negl(n). \tag{48}$$

This proves the theorem. □

6.1.1. Collusion resistance

Our scheme is collusion resistance. From Theorem 7, we know that for a given outsourced ciphertext any groups of unauthorized parties can not learn any partial information about the corresponding data. So, it is obvious that, they are unable to decrypt the ciphertext.

6.2. Fine-grained access control:

Our scheme offers fine-grained access control. Indeed, as we have seen in Section 5, our proposed scheme enables the data owner to define a flexible monotone access tree over an attribute set and encrypt his/her data under the access tree. Also, as we have proved in Theorem 7, a data user can access to the data if and only if his/her attribute set satisfies the access structure. Otherwise, the data user can not learn any partial information about the data.

Table 2 – Comparison with some similar schemes

Schemes	Fully distributed	Decryption outsourcing	Semantic seurity proof	Access structure	User revocation
Wang *et al.* [4] (2011)	No	No	In the random oracle model	DNF	Yes
Liu *et al.* [7] (2014)	No	No	-	DNF	Yes
Deng *et al.* [24] (2014)	No	No	In the standard model	LSSS	No
Li *et al.* [8] (2016)	No	Yes	In the standard model	LSSS	Yes
Huang *et al.* [9] (2017)	No	Yes	-	Access tree	No
Our scheme	Yes	Yes	In the standard model	Access tree	Yes

7. Performance analysis

In this section, we first compare the features of our proposed scheme with some current similar schemes in Table 2. Then, we analyze the computational cost of our proposed scheme in Table 4. After that, we compare the storage cost on data users and the CSP in Table 5. Also, the notations used in these comparisons are given in Table 3.

7.1. Comparison

We list some features of our proposed scheme in Table 2 and make a comparison between it and five similar schemes in terms of fully disturbed, decryption outsourcing, semantic security proof, access structure, and user revocation.

As it is presented in this table, our scheme is the only one that has all the properties. None of the other five schemes is fully distributed. Also, it can be seen that just our scheme and the schemes proposed by Li *et al.* [8] and Huang *et al.* [9] offer decryption outsourcing. From the table, the semantic security of our scheme and the schemes proposed by Deng *et al.* [25] and Li *et al.* [8] have been proven in the standard model. Also, the security of the Wang's scheme only has been proven in the random oracle model. However, any formal semantic security proof for the schemes proposed by Liu *et al.* [7] and Huang *et al.* [9] have not been given. The access structures used in schemes of Huang *et al.* [9] and ours are in the form of access tree, in the schemes of wang *et al.* [4] and Liu *et al.* [7] the DNF access structure is used, and in schemes of Deng *et al.* [25] and Li *et al.* [8] the LSSS access structure is used to present users' access policies. Moreover, all the schemes except Deng *et al.* [25] and Huang *et al.* [9] offer user revocation mechanism.

7.2. Computational cost

In the following we compare our work with schemes of Wang *et al.* [4], Liu *et al.* [7], Deng *et al.* [24], Li *et al.* [8], and Huang *et al.* [9] in terms of the computational cost on data users when the encryption and decryption algorithms are run.

As we mentioned in Table 2, three types of access structures DNF, LSSS and access tree are used in these schemes. In the scheme proposed by Wang *et al.* [4], for any given DNF access structure $\mathbb{A} = \bigvee_{i=1}^{N} CC_i = \bigvee_{i=1}^{N} \bigwedge_{j=1}^{n_i} a_{ij}$, attributes included in each conjunctive clauses CC_i are administered by a unique domain authority DM_i and for each domain $DM_{i_{t_i}}$ with identifier $ID_{i_{t_i}}$, there is a unique sequence $(ID_{i_1=1}, ID_{i_2}, \ldots, ID_{i_{t_i}})$ from the domain in the next level, $DM_{i_1=1}$, to $DM_{i_{t_i}}$. In an LSSS access structure the attributes are arranged in a matrix $\widetilde{U} = (U_1, \ldots, U_L)^T$ and an attribute vector of depth k is defined as a vector $\mathbf{u} = (u_1, \ldots, u_k)$,

Table 3 – Notations used in our numerical comparison

Notation	Description
N	The number of conjunctive clauses in a DNF access structure
n_i	The number of attributes in the i-th conjunctive clause of a DNF access structure
n^*	The number of attributes in a conjunctive clause satisfied by attributes of a data user who runs the decryption algorithm
n	Security parameter of the system
t_i	The length of the unique path between the domain authority administering the attributes in the i-th conjunctive clause of a given DNF access structure, $DM_{i_{t_i}}$, and domain authority DM_1
t^*	The length of the unique path between the domain authority administering an attribute set of a data user running the decryption algorithm and domain authority DM_1
$\|L_{\mathcal{T}}\|$	The number of leaf nods in the access tree $\mathcal{T}$
S^*	The attribute vector set which satisfies a given LSSS access structure
$\|\mathbb{F}_E\|$	The number of domain authorities that administrate the attributes in a given LSSS access structure
l	The number of rows of the share-generating matrix in a given LSSS access structure
l_1	The bit length of the elements of G_1
l_2	The bit length of the elements of G_2
k	Depth of attribute vectors
τ_p	The pairing operation
τ_e	The exponentiation operation (calculating g^k, for a given $k \in \mathbb{Z}_q$ and $g \in G_2$)
τ_m	The multiplication operation (calculating rP, for a given $r \in \mathbb{Z}_q$ and $P \in G_1$)
τ_{o_1}	The operation of G_1
τ_{o_2}	The operation of G_2
$\|\mathbb{U}\|$	The number of attributes in the universal attribute set
$\|\mathbf{ND}\|$	The number of domain authorities in the scheme of Wang *et al.*(the number of PSDAs in our proposed scheme.
$\|\mathbf{KGDA}\|$	The number of KGDAs in our proposed system
$\|A_u\|$	The number of attributes of data user u

Table 4 – Comparison of computational cost on data user

Schemes	Data encryption	Data decryption
Wang *et al.* [4] (2011)	$\tau_p + ((\sum_{i=1}^{N} t_i) + N + 2)\tau_m + \tau_{o_1} \sum_{i=1}^{N} n_i$	$(t^* + 1)\tau_p + n^*\tau_{o_1} + (t^* + 1)\tau_{o_2} + 3\tau_m$
Liu *et al.* [7] (2014)	$\tau_p + N\tau_m + \tau_{o_2} + \tau_{o_1} \sum_{i=1}^{N} n_i$	$2\tau_p + n^*\tau_{o_1} + 3\tau_m$
Deng *et al.* [24] (2014)	$(l(4 + k))\tau_m + \tau_e + \tau_{o_2}$	$(3\|S^*\| + 1)\tau_p + \|S^*\|\tau_e$
Li *et al.* [8] (2016)	$\tau_e + l(\|\mathbb{F}_E\| + 1)(\tau_m + \tau_{o_2}) + \tau_{o_1}$	$\tau_e + \tau_{o_2}$
Huang *et al.* [9] (2017)	$\tau_p + (2\|L_{\mathcal{T}}\| + 1)\tau_m + \tau_e$	$\tau_p + \tau_{o_2}$
Our scheme	$\tau_e + (2\|L_{\mathcal{T}}\| + 2)\tau_m + \|L_{\mathcal{T}}\|\tau_{o_1} + \tau_{o_2}$	$\tau_e + \tau_{o_2}$

where $u_i \in U_i$. In this case, an access structure $\mathbb{A}$ is a collection of non-empty subsets contain the attribute vectors of depth k. Also the access structure can be defined by a share-generating matrix $\mathbf{A}$ with l rows and a function ρ which maps each row of the matrix to a unique attribute vector. We refer the reader to [4, 8, 24] for further details.

Table 4 shows the comparison results. In this case, we consider the pairing operation, exponentiation operation, multiplication operation and operations of the groups. It is known that the pairing computation is the most expensive operation [26]. From the table, it can be verified that the schemes proposed by Li *et al.* [8] and us are pairing-free in both of the encryption and decryption phases. But the other schemes have at least one pairing operation in their decryption phase.

In the data encryption phase, the computation costs of the schemes proposed by Wang *et al.* [4] and Liu *et al.* [7] are $\tau_p + \tau_m((\sum_{i=1}^{N} t_i) + N + 2) + \tau_{o_1} \sum_{i=1}^{N} n_i$ and $\tau_p + N\tau_m + \tau_{o_2} + \tau_{o_1} \sum_{i=1}^{N} n_i$, respectively, which both of them have one paring operation, and the number of group operations grows linearly with the number of attributes in the access structure. Also, in the scheme of Wang *et al.* [4], the number of the multiplication operations grows linearly with the number of conjunctive clauses of the access structure and the lengths of the paths from the domain authorities administering the attributes of the conjunctive clauses to the domain authority in the top level. Also, in the scheme of Liu *et al.* [7] it grows linearly with the number of conjunctive clauses of the access structure. In the schemes proposed by Deng *et al.* [24] and Li *et al.* [8], the computational cost of encryption are $(l(4+k))\tau_m + \tau_e + \tau_{o_2}$ and $\tau_e + l(|\mathbb{F}_E| + 1)(\tau_m + \tau_{o_2}) + \tau_{o_1}$, respectively, which in both schemes there is no pairing operation and the number of multiplication operations grows linearly with the number of rows of the share-generating matrix. Also, in the scheme of Huang *et al.* [9], the computational cost of encryption is $\tau_p + (2|L_{\mathcal{T}}| + 1)\tau_m + \tau_e$, which has one paring operation and the number the multiplication operations grows linearly with the number of attributes exiting in the access tree. In our scheme, the encryption computation cost is $\tau_e + (2|L_{\mathcal{T}}| + 2)\tau_m + |L_{\mathcal{T}}|\tau_{o_1} + \tau_{o_2}$ that does not have any pairing operations and similar to the scheme proposed by Huang *et al.* [9], the multiplication operations grows linearly with the number of attributes of the access tree. Also, in our scheme, the number of group operations grows linearly with the number of attributes of the access tree.

In the data decryption phase, the computational cost on the user side in the schemes of Wang *et al.* [4] and Deng *et al.* [24] are $(t^* + 1)\tau_p + n^*\tau_{o_1} + (t^* + 1)\tau_{o_2} + 3\tau_m$ and $(3|S^*| + 1)\tau_p + |S^*|\tau_e$, respectively, which in the first scheme the number of pairing and group operations grow linearly with the length of the path between the domain authority administering the attributes in the conjunctive clause satisfied by the data user's attributes and the domain authority in the top level. In the second one the number of pairing and exponentiation operations grow linearly with the number of attribute vectors in S^*. In the scheme proposed by Liu *et al.* [7], the computational cost of decryption phase is $2\tau_p + n^*\tau_{o_1} + 3\tau_m$ which has two pairing and tree exponentiation operations. Also, the number of group operations grows linearly with the number of attributes in the conjunctive clause which is satisfied by the data user's attributes. In the scheme proposed by Huang *et al.* [9], the decryption computation cost is $\tau_p + \tau_{o_2}$ [9] which is reduced significantly, in comparison with the three mentioned schemes. But the scheme has one pairing operation which is more costly than exponentiation operation. The decryption computation cost in the schemes of Li *et al.* [8] and ours is $\tau_e + \tau_{o_2}$ which does not have any pairing operations.

7.3. Storage cost

Another significant aspect of a data access control system in the cloud storage services is its storage cost. In this section, we analyze the storage cost on data user and cloud in our proposed scheme and compare them with scheme of Wang *et al.* [4], and the scheme of Huang *et al.* [9]. We considered schemes proposed by Deng *et al.* [24], Liu *et al.* [7] and Li *et al.* [8] in the Sections 7.1 and 7.2. But because of many differences in these schemes with ours, we do not consider them in this section.

Table 5 – Storage costs

Schemes	Public parameters	Ciphertext	Key storage
Wang *et al.* [4] (2011)	$(2+\lvert\mathbb{U}\rvert+\lvert\mathbf{ND}\rvert)l_1$	$n+((\sum_{i=1}^{N} t_i)+N+1)l_1$	$\sum_{i=1}^{\lvert A_u\rvert}(t_i+1)l_1$
Huang *et al.* [9] (2017)	$2l_1$	$n+l_2+(2\lvert L_{\mathcal{T}}\rvert+1)l_1$	$2\lvert A_u\rvert l_1+1$
Our scheme	$(\lvert\mathbb{U}\rvert+2\lvert\mathbf{KGDA}\rvert+\lvert\mathbf{ND}\rvert+3)l_1+l_2$	$2l_2+(2\lvert L_{\mathcal{T}}\rvert+2)l_1$	$\lvert A_u\rvert l_1$

The size of public-parameters in the Scheme of Wang *et al.* [4] and ours are $(2+|\mathbb{U}|+|\mathbf{ND}|)l_1$ and $(|\mathbb{U}|+2|\mathbf{KGDA}|+|\mathbf{ND}|+3)l_1+l_2$, respectively. However, in the scheme of Huang *et al.* [9] this size is equal to $2l_1$, which is significantly shorter than two mentioned schemes. The main reason is that Huang *et al.* [9] did not consider the user revocation and therefore many parameters are not needed in their scheme.

The length of each ciphertext in the scheme of Huang *et al.* [9] and ours are $n+l_2+(2|L_{\mathcal{T}}|+1)l_1$ and $2l_2+(2|L_{\mathcal{T}}|+2)l_1$, respectively, which are about the same. In the scheme of Wang *et al.* [4], the size of each ciphertext is equal to $n+((\sum_{i=1}^{N} t_i)+N+1)l_1$ which grows linearly with the number of conjunctive clauses of the access structure and the lengths of the paths from the domain authorities administering the attributes of the conjunctive clauses to the domain authority in the top level .

As we can see in Table 3, in the schemes of Wang *et al.* [4] and Huang *et al.* [9] the secret-key storage costs are $\sum_{i=1}^{|A_u|}(t_i+1)l_1$ and $2|A_u|l_1+1$, respectively. In the first one, the size of secret-keys increases with the length of paths between the domain authority which administers the i-th attribute of the data user and the domain authority in the top level. In the second one it increases linearly with number of the data user's attributes. Also, the secret-key size of a data user in our proposed scheme is $|A_u|l_1$ which is half of the secret-key size of the scheme of Huang *et al.* [9]. So, our scheme greatly reduces the cost of secret-key storage.

8. Conclusion

In this paper, we proposed the concept of a fully distributed revocable ciphertext-policy hierarchical attribute-based encryption system, named FDR-CP-HABE. Then, we proposed the first FDR-CP-HABE scheme. Our proposed scheme offers a high level of flexibility and scalability in key delegation and user revocation mechanisms. Our scheme enables the data owner to define and enforce an access structure on a set of attribute. Also, it supports any monotone access structure in terms of an access tree. We proved the semantic security of our scheme in the standard model based on the harness assumption of the decisional bilinear DiffieHellman (DBDH) problem. We showed that our proposed scheme achieves fine-grained data access control over the outsourced ciphertexts, and it is resistance against the collusion attack of unauthorized data users. Our proposed scheme is pairing-free and offers a lightweight computing in decryption phase, using the outsourcing technique. We analyzed the performance of our proposed scheme and made a comparison between it and some similar existing works. We showed that the performance of our work is acceptable compared with the other similar schemes. By observing the security, efficiency, scalability and flexibility of our proposed scheme, one can conclude that it is appropriate for cloud computing.

References

[1] Sahai, A. and Waters, B., 2005, May. Fuzzy identity-based encryption. In Annual International Conference on the Theory and Applications of Cryptographic Techniques (pp. 457-473). Springer, Berlin, Heidelberg.

[2] Goyal, V., Pandey, O., Sahai, A. and Waters, B., 2006, October. Attribute-based encryption for fine-grained access control of encrypted data. In Proceedings of the 13th ACM conference on Computer and communications security (pp. 89-98). Acm.

[3] Bethencourt, J., Sahai, A. and Waters, B., 2007, May. Ciphertext-policy attribute-based encryption. In Security and Privacy, 2007. SP'07. IEEE Symposium on (pp. 321-334). IEEE.

[4] Wang, G., Liu, Q., Wu, J. and Guo, M., 2011. Hierarchical attribute-based encryption and scalable user revocation for sharing data in cloud servers. computers & security, 30(5), pp.320-331.

[5] C. Gentry and A. Silverberg. Hierarchical ID-Based Cryptography. In Proceedings of ASIACRYPT 2002, pages 548-566.

[6] Ali, M., Mohajeri, J., Sadeghi, M.-R., 2018. Computer Science > Cryptography and Security On the security of the hierarchical attribute based encryption scheme proposed by Wang et al.. http://arxiv.org/abs/1810.05864.

[7] Liu, Q., Wang, G. and Wu, J., 2014. Time-based proxy re-encryption scheme for secure data sharing in a cloud environment. Information sciences, 258, pp.355-370.

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

[9] Huang, Q., Yang, Y. and Shen, M., 2017. Secure and efficient data collaboration with hierarchical attribute-based encryption in cloud computing. Future Generation Computer Systems, 72, pp.239-249

[10] Wan, Z., Liu, J.E. and Deng, R.H., 2012. HASBE: A hierarchical attribute-based solution for flexible and scalable access control in cloud computing. IEEE transactions on information forensics and security, 7(2), pp.743-754.

[11] Boneh, D. and Boyen, X., 2004, May. Efficient selective-ID secure identity-based encryption without random oracles. In International Conference on the Theory and Applications of Cryptographic Techniques (pp. 223-238). Springer, Berlin, Heidelberg.

[12] Boneh, D. and Boyen, X., 2004, August. Secure identity based encryption without random oracles. In Annual International Cryptology Conference (pp. 443-459). Springer, Berlin, Heidelberg.

[13] Boneh, D. and Franklin, M., 2001, August. Identity-based encryption from the Weil pairing. In Annual international cryptology conference (pp. 213-229). Springer, Berlin, Heidelberg.

[14] Lewko, A., Okamoto, T., Sahai, A., Takashima, K. and Waters, B., 2010, May. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In Annual International Conference on the Theory and Applications of Cryptographic Techniques (pp. 62-91). Springer, Berlin, Heidelberg.

[15] Bobba, R., Khurana, H. and Prabhakaran, M., 2009, September. Attribute-sets: A practically motivated enhancement to attribute-based encryption. In European Symposium on Research in Computer Security (pp. 587-604). Springer, Berlin, Heidelberg.

[16] Gentry, C. and Silverberg, A., 2002, December. Hierarchical ID-based cryptography. In International Conference on the Theory and Application of Cryptology and Information Security (pp. 548-566). Springer, Berlin, Heidelberg.

[17] Wang, S., Yu, J., Zhang, P. and Wang, P., 2014. A Novel File Hierarchy Access Control Scheme using Attribute-Based Encryption. Applied Mechanics & Materials.

[18] Wang, S., Zhou, J., Liu, J.K., Yu, J., Chen, J. and Xie, W., 2016. An efficient file hierarchy attribute-based encryption scheme in cloud computing. IEEE Transactions on Information Forensics and Security, 11(6), pp.1265-1277.

[19] Liu, X., Ma, J., Xiong, J. and Liu, G., 2014. Ciphertext-Policy Hierarchical Attribute-based Encryption for Fine-Grained Access Control of Encryption Data. IJ Network Security, 16(6), pp.437-443.

[20] Li, J., Wang, Q., Wang, C. and Ren, K., 2011. Enhancing attribute-based encryption with attribute hierarchy. Mobile networks and applications, 16(5), pp.553-561.

[21] Li M, Yu S, Zheng Y, Ren K, Lou W. Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption. IEEE Trans Parallel Distrib Syst 2013;24:13143.

[22] Yu S,Wang C, Ren K, Lou W. Attribute based data sharing with attribute revocation. In: Proceedings of the 5th ACM symposium on information, computer and communications security, ASIACCS 10. NewYork, NY, USA: ACM; 2010b. p. 261 70. doi:10.1145/1755688.1755720.

[23] Lindell, Y. and Katz, J., 2014. Introduction to modern cryptography. Chapman and Hall/CRC.

[24] Deng, H., Wu, Q., Qin, B., Domingo-Ferrer, J., Zhang, L., Liu, J. and Shi, W., 2014. Ciphertext-policy hierarchical attribute-based encryption with short ciphertexts. Information Sciences, 275, pp.370-384.

[25] Jung, T., Li, X.Y., Wan, Z. and Wan, M., 2013, April. Privacy preserving cloud data access with multi-authorities. In INFOCOM, 2013 Proceedings IEEE (pp. 2625-2633). IEEE.

[26] Fan, X., Gong, G. and Jao, D., 2008, August. Efficient pairing computation on genus 2 curves in projective coordinates. In International Workshop on Selected Areas in Cryptography (pp. 18-34). Springer, Berlin, Heidelberg.