Ciphertext-Policy Hierarchical Attribute-Based Encryption with Short Ciphertexts: Efficiently Sharing Data among Large Organizations

Hua Deng[a], Qianhong Wu*[b], Bo Qin[c], Josep Domingo-Ferrer[d], Lei Zhang[e], Jianwei Liu[b], Wenchang Shi[c]

[a]*School of Computer, Wuhan University, Wuhan, China*
[b]*School of Electronics and Information Engineering, Beihang University, Beijing, China*
[c]*School of Information, Remin University of China, Beijing, China*
[d]*Dept. of Computer Engineering and Mathematics, Universitat Rovira i Virgili, Catalonia*
[e]*Software Engineering Institute, East China Normal University, Shanghai, China*

Abstract

Attribute-based encryption (ABE) systems allow encrypting to uncertain receivers by means of an access policy specifying the attributes that the intended receivers should possess. ABE promises to deliver fine-grained access control of encrypted data. However, when data are encrypted using an ABE scheme, key management is difficult if there is a large number of users from various backgrounds. In this paper, we elaborate ABE and propose a new versatile cryptosystem referred to as ciphertext-policy hierarchical ABE (CP-HABE). In a CP-HABE scheme, the attributes are organized in a matrix and the users having higher-level attributes can delegate their access rights to the users at a lower level. These features enable a CP-HABE system to host a large number of users from different organizations by delegating keys, e.g., enabling efficient data sharing among hierarchically organized large groups. We construct a CP-HABE scheme with short ciphertexts. The scheme is proven secure in the standard model under non-interactive assumptions.

Keywords: Cloud storage, Access control, Attribute-based encryption, Ciphertext-policy attribute-based encryption

*Corresponding author. Address: School of Electronics and Information Engineering, Beihang University, Xueyuan Road 37, Haidian District, 100191, Beijing. Email: qhwu@xidian.edu.cn, Tel: +86 10 8233 9469.

1. Introduction

Attribute-based Encryption (ABE) [26] is a promising cryptographic primitive which provides fine-grained access control over the outsourced ciphertexts (quite relevant in cloud environments). It allows encrypting to uncertain decryptors by means of an access policy that specifies what attributes the intended decryptors should possess. ABE can be classified into two categories of key-policy ABE (KP-ABE) and ciphertext-policy ABE (CP-ABE). In KP-ABE, attributes are used to associate with ciphertexts and access policies over these attributes are ascribed to users' secret keys. The encryptor only needs to know the public attributes of the potential decryptors. CP-ABE is similar to KP-ABE, except that the access policy is associated with each ciphertext and a secret key is associated with a user's attributes. It allows the encryptor to associate the ciphertext with an access policy specifying the attributes that the authorized decryptors should have.

Although ABE systems have interesting features, they could be elaborated further to cater for more complicated applications. To see this point, let us consider the following scenario.

Motivating Scenario: Company A may allow its employees to outsource encrypted data to a server maintained by a third party, e.g., a cloud service provider. The outsourced data are only accessible to the employees of Company A and its peer companies. The employees of the companies may have attributes such as manager, department manager, project leader, and engineer, according to their hierarchical positions, together with other attributes with hierarchical features, e.g., working years. Company A signs contracts with its peer companies to share the encrypted data if the employees of the peer companies possess the required attributes.

To save communication, computation and management overhead, Company A should not be required to validate the attributes of the employees of its peer companies or generate secret keys for all matching employees. Moreover, the meaningful attributes of the employees of the peer companies should be kept private to prevent leaking sensitive information about the peer companies' internal organizations and most valuable employees. Indeed, what needed is a kind of delegation of access rights, which can enable the peer companies to generate secret keys for their own employees.

There already have some ABE schemes which can provide *limited* delegation of access rights. In [11, 18], the authors proposed two KP-ABE schemes which allow users to delegate decryption keys for more *restrictive* access structures. That is, the access structure of the delegated key must be more restrictive than that of the original key so as to accommodate new attributes. As for CP-ABE case, Bethencourt *et al.* [6], Goyal *et al.* [10] and Waters [29] presented CP-ABE schemes supporting delegation, while all of these schemes only allow delegating keys for attribute sets that are subsets of the original attribute sets. However, in the motivating scenario, an employee should be allowed to delegate to his/her subordinates without worrying whether the subordinates' attribute sets are subsets or not.

The difficulty of providing delegation for the motivating scenario relies on the deployment of secret sharing scheme. In most CP-ABE schemes [6, 10, 11, 15, 23, 29], a secret sharing scheme is employed to realize an access policy associated with a ciphertext. In these realizations, each attribute belonging to the access policy obtains a share of the secret. This requires that the same attribute included in the attribute set of the secret key possess a separate key component so that the share can contribute to reconstructing the secret key used in decryption. But a delegator's secret key is generated by a key generator; hence, without knowing the secret key of the key generator, the delegator can not generate a key component for a new attribute. This is the main problem that prevents the above mentioned CP-ABE schemes from being applicable to the motivating scenario.

1.1. Our contributions

In this paper we address the motivating application scenario with a new cryptographic primitive referred to as hierarchical attribute-based encryption (HABE). Observing that many attributes are hierarchical in the real world, in HABE the attribute universe is arranged in a matrix such that the attributes higher in the hierarchy (e.g. manager) sit in upper levels of the matrix while attributes lower in the hierarchy (e.g. engineer) sit in lower levels of the matrix. This arrangement naturally leads to the notion of *hierarchical attributes* or *attribute vectors*, which can be formed by sampling attributes from an upper level to a lower level.

By using the attribute vectors, we can achieve the required delegation mechanism that allows new attributes to be added into the original attribute set without requiring the delegator to know the secret key of the key generator. Our delegation mechanism is similar to that of HIBE but, unlike HIBE (which allows delegation to any identity), it ensures that only attributes valid according to the attribute matrix can be delegated.

We model ciphertext-policy HABE (CP-HABE), that can be applied for the motivating scenario. In CP-HABE, ciphertexts are generated with access policies specifying the attribute vectors that the potential decryptors should possess. Importantly, a user at a higher level can delegate secret key to a user at a lower level without the constraint that the latter's attribute set must be a subset of the former's. We then define the full security of CP-HABE. In full security, the attacker is allowed to obtain the public parameters, create attribute vectors, specify access policies, and query for the keys corresponding to the sets of attribute vectors. Then the attacker outputs two challenge messages and a challenge set of attribute vectors. One of the two messages is chosen to generate a ciphertext associated with the challenge policy. Even for such a polynomial-time attacker, it is not possible to distinguish which message was used to generate the ciphertext, provided that the attacker did not query for the key associated with the attribute vector set that satisfies the challenge access policy.

We construct a CP-HABE scheme by employing a linear secret sharing scheme (LSSS) to achieve suitable expressiveness of access structures. One might attempt a straightforward construction of CP-HABE from HIBE by treating each attribute vector as an identity vector in the underlying HIBE. Unfortunately, such a straightforward construction is vulnerable to collusion attacks in which a coalition of users manages to decrypt ciphertexts intended to none of them, provided that the union of the attribute vectors of the colluding users meets the access policy. This kind of attacks must be prevented in practice. We address this problem by randomizing the secret keys assigned to each user, and, by using the well-established dual encryption techniques, we prove that our CP-HABE scheme is fully secure in the standard model (i.e., without using random oracles) under several non-interactive assumptions. A trivial CP-HABE might be constructed from CP-ABE by associating each attribute vector with a unique public key of the underlying CP-ABE, just as a distinct public key is published for each attribute in CP-ABE schemes [11, 10, 15, 29]. Such a trivial construction incurs a public key size exponentially growing with the maximum hierarchy size. In our construction, the public key size is linear with the maximum hierarchy size and the ciphertext size is independent of the number of users or the user hierarchies.

1.2. Related work

To increase the convenience of public-key encryption, Shamir proposed the concept of Identity-based Encryption (IBE) [24]. In an IBE system, a user's public key is his/her identity, such as e-mail address or phone number. An encryptor can create a ciphertext under the receiver's identity without asking for the receiver's public key beforehand. The first fully functional IBE scheme was presented by Boneh and Franklin [4]. They constructed the IBE scheme by exploiting the Weil pairing and they proved its selective security in the random oracle model. In the selective security model, the adversary has to declare the identity to be challenged before seeing the system public parameters. To achieve stronger security, Waters [27] proposed a fully (adaptively) secure IBE scheme in the standard model.

Similarly to IBE, a number of identity-based cryptographic primitives have been proposed, such as identity-based proxy re-encryption [9, 30, 25], identity based signature [24, 33], identity-based key agreement [7, 34] and identity-based key updating [20].

Hierarchical Identity-based Encryption (HIBE), first proposed by [12], is another identity-based cryptographic primitive that extends IBE with key delegation to relieve the private key generator in IBE from heavy key management burden when there is a large number of users in the system. In a HIBE system, users are associated with identity vectors which denote their positions in the hierarchy. HIBE allows a user at a higher level to delegate keys to his subordinates. Boneh *et al.* proposed a HIBE scheme [3] which is selectively secure and has constant-size ciphertext. Waters [28] proposed a fully secure HIBE scheme under simple assumptions by adopting the dual system encryption proof technique. To obtain more general key delegation patterns in HIBE, Abdalla *et al.* [1] presented a "wicked" IBE scheme, in which a key is associated with a pattern vector where some entries are concrete identities and some entries can be left blank using wildcards.

ABE, introduced by Sahai and Waters [26], extends IBE by replacing the identity with a set of attributes. Goyal *et al.* [11] classified ABE into two categories, KP-ABE and CP-ABE. They presented a KP-ABE scheme supporting monotonic access trees which is proven to be selectively secure under the decisional bilinear Diffie-Hellman (BDH) assumption. To enable more flexible access policies, Ostrovsky *et al.* [21] developed a KP-ABE scheme to support non-monotone formulas in key policies. The first CP-ABE construction was proposed by Bethencourt *et al.* in [6]. To reduce the decryption time, Hohenberger and Waters [13] presented a KP-ABE with fast decryption. Chase proposed a multi-authority ABE scheme [8] which relieves of the heavy trust reliance on a single private key generation authority. Lin *et al.* improved Chase's ABE to obtain a threshold multi-authority ABE scheme [19] that needs no trusted central authority.

Some ABE schemes have achieved certain delegation functionalities. Goyal *et al.* [11] and Lewko *et al.* [18] proposed key delegation for KP-ABE schemes with the restriction that the tree structures of the new keys must be more restrictive than those of the original keys. The first CP-ABE scheme [6] supports delegation for sets of attributes with the security proved in the generic group model. Goyal *et al.* [10] promoted the security proof of the first CP-ABE and also provided delegation. Waters [29] proposed an efficient and expressive CP-ABE with delegation functionality. However, as mentioned before, these CP-ABE schemes only allow delegation for subsets of the original attribute sets, which means new attributes can not be added into the attribute sets of secret keys.

Okamoto and Takashima [22] proposed a hierarchical delegation mechanism for a predicate encryption (PE) scheme, i.e., a hierarchical inner-product PE (HIPE) scheme, but only selective security was proven. A fully secure HIPE scheme was proposed in [15]. Nevertheless, as remarked in [15], inner product predicates are less expressive than the LSSS access structures of ABE since, to use inner product predicates for ABE, formulas must be written in a CNF or DNF form, which can cause a superpolynomial blowup in size for arbitrary formulas. Also, using HIPE as a toolkit to provide delegation for CP-ABE encounters the same problems of huge-size public key and collusion attacks as when introducing HIBE into CP-ABE.

Several recent schemes use the term of HABE but in different senses. In the schemes [31, 32], the users are organized in a hierarchical tree but all attributes are at the same level. These schemes use CNF and DNF formulas to describe the access structure, which implies limited expressiveness. Also, they do not provide formal security or are only proven secure in the random oracle model. Indeed, their focus is to adapt ABE schemes for specific applications.

1.3. Paper organization

In Section 2, we give necessary background on LSSS, composite order bilinear groups and complexity assumptions. In Section 3, we formalize CP-HABE and define its security. We present a CP-HABE construction with formal security proof in Section 4. Finally, we conclude the paper in Section 5.

2. Preliminaries

2.1. Access Structures [2]

Definition 1. Let $\{P_1, P_2, \cdots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1, P_2, \cdots, P_n\}}$ is monotone if for $\forall B, C$, we have that $C \in \mathbb{A}$ holds if $B \in \mathbb{A}$ and $B \subseteq C$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $\{P_1, P_2, \cdots, P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1, P_2, \cdots, P_n\}} \backslash \{\emptyset\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets.

In traditional ABE (KP-ABE and CP-ABE), the role of the parties is played by the attributes. In our HABE, the role of the parties is taken by hierarchical attributes or attribute vectors. We only consider the monotone access structures in our HABE. Nevertheless, similarly to [11, 29, 15], it is possible to realize general access structures by having the negation of an attribute as a separate attribute in our scheme, at the cost of doubling the number of single attributes in the system.

2.2. Linear Secret Sharing Schemes [2]

Definition 2. A secret-sharing scheme Π over a set of parties $\mathcal{P}$ is called linear (over $\mathbb{Z}_p$) if

1. The shares for each party form a vector over $\mathbb{Z}_p$.
2. There exists a matrix $\mathbf{A}$ called the share-generating matrix for Π, where $\mathbf{A}$ has l rows and n columns. For all $i = 1, \cdots, l$, the i-th row of $\mathbf{A}$ is labeled by a party $\rho(i)$, where ρ is a function from $\{1, \cdots, l\}$ to $\mathcal{P}$. When we consider the column vector $\vec{s} = (s, s_2, \cdots, s_n)$, where $s \in \mathbb{Z}_p$ is the secret to be shared, and $s_2, \cdots, s_n \in \mathbb{Z}_p$ are randomly chosen, then $\mathbf{A}\vec{s}$ is the vector of l shares of the secret s according to Π. If A_i denotes the i-th row of $\mathbf{A}$, then $\lambda_i = A_i\vec{s}$ is the share belonging to party $\rho(i)$.

Linear Reconstruction. It has been shown in [2] that every LSSS Π enjoys the linear reconstruction property. Suppose Π is the LSSS for access structure $\mathbb{A}$ and S is an authorized set in $\mathbb{A}$, i.e., $\mathbb{A}$ contains S. There exist constants

$\{\omega_i \in \mathbb{Z}_N\}$ which can be found in time polynomial in the size of the share-generating matrix $\mathbf{A}$ such that if $\{\lambda_i\}$ are valid shares of s, then $\sum_{i\in I}\omega_i\lambda_i = s$, where $I = \{i : \rho(i) \in S\} \subseteq \{1, \cdots, l\}$.

2.3. Composite Order Bilinear Groups

Suppose that $\mathcal{G}$ is a group generator and ℓ is the security parameter. Composite order bilinear groups [5] can be defined as: $(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\ell)$, where p_1, p_2, p_3 are three distinct primes, $\mathbb{G}, \mathbb{G}_T$ are cyclic groups of order N and the group operation in $\mathbb{G}, \mathbb{G}_T$ is computable in polynomial time with respect to ℓ. A bilinear map $e : \mathbb{G}\times\mathbb{G} \to \mathbb{G}_T$ is an efficiently computable map with the following properties:

1. **Bilinearity**: for all $a, b \in \mathbb{Z}_N$ and $g, h \in G$, $e(g^a, h^b) = e(g, h)^{ab}$.
2. **Non-degeneracy**: $\exists g \in G$ such that $e(g, g)$ has order N in $\mathbb{G}_T$.

Let $\mathbb{G}_i$ denote the subgroup of order p_i and $\mathbb{G}_{(i,j)}$ denote the subgroup of order p_ip_j. The *orthogonality* property of $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3$ is defined as: for all $h_i \in \mathbb{G}_i, h_j \in \mathbb{G}_j$ it holds that $e(h_i, h_j) = 1$, where $i \neq j \in \{1, 2, 3\}$. Assume $h_1 \in \mathbb{G}_1, h_2 \in \mathbb{G}_2$ and g is an generator of $\mathbb{G}$. Then $g^{p_1p_2}$ generates $\mathbb{G}_3$, $g^{p_1p_3}$ generates $\mathbb{G}_2$, and $g^{p_2p_3}$ generates $\mathbb{G}_1$. Hence, for some a, b, we can rewrite h_1, h_2 as

$$h_1 = (g^{p_2p_3})^a, h_2 = (g^{p_1p_3})^b.$$

Then we have

$$e(h_1, h_2) = e(g^{p_2p_3a}, g^{p_1p_3b}) = e(g^a, g^{p_3b})^{p_1p_2p_3} = 1.$$

The orthogonality property is essential in our constructions and security proofs.

2.4. Computational Assumptions

The security of our schemes relies on the following non-interactive assumptions which were first introduced in [17] and subsequently used in [15, 18] to achieve fully secure ABE schemes.

Assumption 1. *Given a group generator $\mathcal{G}$ and $(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e) \overset{R}{\leftarrow} \mathcal{G}(1^\ell)$, define a distribution:*

$$g \overset{R}{\leftarrow} \mathbb{G}_1,\ g_3 \overset{R}{\leftarrow} \mathbb{G}_3,\ \mathbb{D} = (N, \mathbb{G}, \mathbb{G}_T, e, g, g_3),\ T_1 \overset{R}{\leftarrow} \mathbb{G}_{(1,2)},\ T_2 \overset{R}{\leftarrow} \mathbb{G}_1.$$

The advantage of an algorithm $\mathcal{A}$ in breaking Assumption 1 is defined as

$$Adv1_{\mathcal{A}}(\ell) = |\Pr[\mathcal{A}(\mathbb{D}, T_1) = 1] - \Pr[\mathcal{A}(\mathbb{D}, T_2) = 1]|.$$

We say that $\mathcal{G}$ satisfies Assumption 1 if for any polynomial-time algorithm $\mathcal{A}$, $Adv1_{\mathcal{A}}(\ell)$ is negligible in ℓ.

Assumption 2. *Given a group generator $\mathcal{G}$ and $(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e) \xleftarrow{R} \mathcal{G}(1^\ell)$, define a distribution:*

$$g, \theta_1 \xleftarrow{R} \mathbb{G}_1,\ \theta_2, \gamma_2 \xleftarrow{R} \mathbb{G}_2,\ g_3, \gamma_3 \xleftarrow{R} \mathbb{G}_3,$$

$$\mathbb{D} = (N, \mathbb{G}, \mathbb{G}_T, e, g, \theta_1\theta_2, g_3, \gamma_2\gamma_3),\ T_1 \xleftarrow{R} \mathbb{G},\ T_2 \xleftarrow{R} \mathbb{G}_{(1,3)}$$

The advantage of an algorithm $\mathcal{A}$ in breaking Assumption 2 is defined as:

$$Adv2_{\mathcal{A}}(\ell) = |\Pr[\mathcal{A}(\mathbb{D}, T_1) = 1] - \Pr[\mathcal{A}(\mathbb{D}, T_2) = 1]|.$$

We say that $\mathcal{G}$ satisfies Assumption 2 if for any polynomial-time algorithm $\mathcal{A}$, $Adv2_{\mathcal{A}}(\ell)$ is negligible in ℓ.

Assumption 3. *Given a group generator $\mathcal{G}$ and $(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e) \xleftarrow{R} \mathcal{G}(1^\ell)$, define a distribution:*

$$\alpha, s \xleftarrow{R} \mathbb{Z}_N,\ g \xleftarrow{R} \mathbb{G}_1,\ \theta_2, \gamma_2, \delta_2 \xleftarrow{R} \mathbb{G}_2,\ g_3 \xleftarrow{R} \mathbb{G}_3,$$

$$\mathbb{D} = (N,\ \mathbb{G},\ \mathbb{G}_T,\ e,\ g,\ g^\alpha\theta_2,\ g_3,\ g^s\gamma_2,\ \delta_2),\ T_1 = e(g,g)^{\alpha s},\ T_2 \xleftarrow{R} \mathbb{G}_T.$$

The advantage of an algorithm $\mathcal{A}$ in breaking Assumption 3 is defined as

$$Adv3_{\mathcal{A}}(\ell) = |\Pr[\mathcal{A}(\mathbb{D}, T_1) = 1] - \Pr[\mathcal{A}(\mathbb{D}, T_2) = 1]|.$$

We say that $\mathcal{G}$ satisfies Assumption 3 if for any polynomial-time algorithm $\mathcal{A}$, $Adv3_{\mathcal{A}}(\ell)$ is negligible in ℓ.

3. Ciphertext-Policy Hierarchical Attribute-based Encryption

3.1. Notations and basic ideas

In a CP-HABE system, the attribute universe is arranged in a matrix which is supposed to have L rows and D columns and is denoted by

$$\mathbf{U} = (U_1, \cdots, U_i, \cdots, U_L)^T,$$

where U_i is the i-th row of $\mathbf{U}$ and contains D attributes and $\mathbf{M}^T$ denotes the transposition of a matrix $\mathbf{M}$. We note that in each U_i there may be some *empty* attributes which can be represented by a special symbol "$\varnothing$".

We define the attribute vector of depth k (or hierarchical attribute at level k) as

$$\vec{u} = (u_1, u_2, \cdots, u_k),$$

where for each i from 1 to k, $u_i \in U_i$. This means that an attribute vector of depth k is composed of k attributes and the i-th attribute is selected from the i-th row of the attribute matrix.

We say that $\vec{u}'$ is a *prefix* of $\vec{u}$ if $\vec{u} = (\vec{u}', u_{k'+1}, u_{k'+2}, \cdots, u_k)$, where k denotes the depth of $\vec{u}$.

We let $S = \{\vec{u}\}$ denote a set of attribute vectors and its cardinality is denoted by $|S|$.

As stated in Definition 1, an access structure is a collection of non-empty subsets of a group of parties. Since the role of party is taken by attribute vectors, in a similar way we can define the access structure $\mathbb{A}$ regarding attribute vector of depth k such that $\mathbb{A}$ is a collection of non-empty subsets of a set of all the attribute vectors of depth k. If a user possesses a set S and $S \in \mathbb{A}$, then S is an authorized set in $\mathbb{A}$ and we say this user or the set satisfy $\mathbb{A}$.

Based on Definition 2, we can easily extend the LSSS to a multi-level access structure: for an access structure $\mathbb{A}$ regarding an attribute vector of depth k, form the share-generating matrix $\mathbf{A}$ such that the inner product of the i-th row of $\mathbf{A}$ and the vector taking the secret as the first coordinate is the share belonging to an attribute vector of depth k; define the *injection* function ρ which maps the i-th row of the matrix $\mathbf{A}$ to the attribute vector of depth k. The *injection* function means that a hierarchical attribute at level k is associated with at most one row of $\mathbf{A}$.

In a CP-HABE system, a ciphertext generated with an access structure $\mathbb{A}$ is decryptable by a key associated with set S under the condition that $S \in \mathbb{A}$. If a key for set S' is able to delegate a key for a set S, it is required that each attribute vector of S have a prefix in S'. This yields the concept of *Set Derivation* of two sets.

Definition 3. For a set S' of attribute vectors of depth k and a set S of attribute vectors of depth $k+1$, we say that S is derived from S', denoted by $S \Leftarrow S'$, if

$$\forall \vec{u} \in S, \exists \vec{u}' \in S', \text{ such that } \vec{u} = (\vec{u}', u_{k+1}),$$

where $u_{k+1} \in U_{k+1}$.

3.2. Modeling CP-HABE

We now give the definition of CP-HABE and its security model. A CP-HABE system for message space $\mathbb{M}$ and access structure space $\mathbb{AS}$ consists of the following five (probabilistic) algorithms.

> $(PK, MSK) \leftarrow \textbf{Setup}(1^\ell)$: The algorithm **Setup** takes no input other than the security parameter ℓ and outputs the public key PK and a master secret key MSK.
>
> $CT \leftarrow \textbf{Encrypt}(M, PK, \mathbb{A})$: The algorithm **Encrypt** takes as inputs a message M, the public key PK and an access structure $\mathbb{A}$. It outputs a ciphertext CT.
>
> $SK_S \leftarrow \textbf{KeyGen}(PK, MSK, S)$: The algorithm **KeyGen** takes as inputs a set S of attribute vectors, the master key MSK and public key PK. It outputs a secret key SK_S for S.
>
> $SK_S \leftarrow \textbf{Delegate}(PK, SK_{S'}, S)$: The algorithm **Delegate** takes as inputs the public key PK, a secret key $SK_{S'}$ for S' which is a set of attribute vectors of depth k and a set S of attribute vectors of depth $k+1$. It outputs the secret key SK_S for S if and only if $S \Leftarrow S'$.
>
> $M/\perp \leftarrow \textbf{Decrypt}(CT, SK_S, PK)$: The algorithm **Decrypt** takes as inputs a ciphertext CT associated with an access structure $\mathbb{A}$, a secret key SK_S for S, and the public key PK. If $S \in \mathbb{A}$, it outputs M; otherwise, outputs a false symbol $\perp$.

The correctness property requires that for all sufficiently large $\ell \in \mathbb{N}$, all universe descriptions $\mathbf{U}$, all $(PK, MSK) \leftarrow \textbf{Setup}(1^\ell)$, all $SK_S \leftarrow \textbf{KeyGen}(PK, MSK, S)$ or $SK_S \leftarrow \textbf{Delegate}(PK, SK_{S'}, S)$, all $M \in \mathbb{M}$, all access structures $\mathbb{A} \in \mathbb{AS}$, and all $CT \leftarrow \textbf{Encrypt}(M, PK, \mathbb{A})$, if S satisfies $\mathbb{A}$, then $\textbf{Decrypt}(CT, SK_S, PK)$ outputs M.

3.3. Security Definitions for CP-HABE

We next define the semantic security against chosen-access-structure and chosen-plaintext attacks in CP-HABE. To capture the realistic attacks, an adversary is allowed to access public keys and query for secret keys. The adversary is required to output two equal-length messages and a challenge access structure. The semantic security states that no probabilistic polynomial-time (PPT) adversary can distinguish the ciphertexts of the two messages using the challenge access structure, provided that the adversary does not have the secret keys that can be used to decrypt the challenge ciphertexts. Formally, the semantic security of a CP-HABE scheme is defined by the following game played between a challenger and an adversary $\mathcal{A}$.

Setup. The challenger runs the algorithm **Setup** to obtain public key PK and gives it to the adversary.

Phase 1. The adversary $\mathcal{A}$ adaptively issues key queries $Q_1, \cdots, Q_{q'}$ to the challenger, where Q_i for $1 \leq i \leq q'$ is one of the following three types:

> Create(S). $\mathcal{A}$ specifies a set S of attribute vectors. In response, the challenger creates a key for this set by calling the algorithm **KeyGen**(PK, MSK, S), and places this key in the set $\mathcal{SK}$, which is initialized to empty. It only gives $\mathcal{A}$ a reference to this key, not the key itself.
>
> Delegate(S, S'). $\mathcal{A}$ specifies a key $SK_{S'}$ for S' in the set $\mathcal{SK}$ and a set S. If $S \Leftarrow S'$, the challenger runs **Delegate**($PK, SK_{S'}, S$) to produce a key for S. It adds this new key to the set $\mathcal{SK}$ and again gives $\mathcal{A}$ only a reference to it, not the actual key.
>
> Reveal(S). $\mathcal{A}$ specifies a key in the set $\mathcal{SK}$. The challenger gives this key to $\mathcal{A}$ and removes it from the set $\mathcal{SK}$.

Challenge. The adversary $\mathcal{A}$ outputs two equal-length messages M_0 and M_1 and an access structure $\mathbb{A}^*$ with the constraint that for any revealed key SK_S for S, $S \notin \mathbb{A}^*$ and for any new key $SK_{S'}$ for S' that can be delegated from revealed one, $S' \notin \mathbb{A}^*$. The challenger then flips a random coin $\beta \in \{0, 1\}$, and encrypts M_β under $\mathbb{A}^*$, producing CT^*. It gives CT^* to the attacker.

Phase 2. The adversary $\mathcal{A}$ adaptively issues key queries $Q_{q'+1}, \cdots, Q_q$ to the challenger just as in Phase 1, with the added restriction that $\mathbb{A}^*$ must not be satisfied by the set of any revealed key or the set of any new key that can be delegated from a revealed one.

Guess. The adversary outputs a guess $\beta' \in \{0, 1\}$.

The advantage of $\mathcal{A}$ in this game is defined as

$$Adv_{\mathcal{A}}^{\text{CP-HABE}} = |\Pr[\beta = \beta'] - 1/2|.$$

Definition 4. We say a CP-HABE system is fully secure if for any PPT attacker $\mathcal{A}$ we have that $Adv_{\mathcal{A}}^{\text{CP-HABE}}$ is negligible in the above game.

4. The CP-HABE Scheme

In this section, we construct a CP-HABE scheme by using the linear secret sharing scheme and adopting the delegation mechanism of most HIBE schemes ([3, 17, 28]). The scheme has short ciphertexts and is proven to be fully secure without using random oracles.

4.1. Our techniques

Given the attribute matrix, we publish parameters for each row and each column. Since attributes always appear in the form of attribute vectors, we do not need to specially assign public parameters for every attribute. We then make the public parameters shared by all possible attribute vectors and thus achieve a relatively short system public key.

We employ the LSSS in our construction to realize the access structure. Thus, in the ciphertext associated with an access structure, each attribute vector belonging to this access structure will obtain a share of the secret. Correspondingly, the same attribute vector in the attribute set of the secret key is associated with a key component which is necessary to reconstruct the secret with the share.

We utilize the HIBE delegation mechanism to achieve the required delegation in our CP-HABE. For a secret key associated with a set of attribute vectors, since each attribute vector has a key component, we can delegate for separate key components of attribute vectors by running the HIBE delegation multiple times. However, if these key components are independent instances of the HIBE key, then the scheme is vulnerable to collusion attacks in the sense that several users who do not satisfy the access policy could collude to decrypt the ciphertext by using their independent key components included in their secret keys. To protect the scheme against this attack, we associate each key component with a common random value that uniquely corresponds to the user, such that without knowing this value, the user is unable to separately use a part of the key components to take part in a collusion.

4.2. The Construction

Given a group description $(N, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda)$, where N is a product of three distinct primes p_1, p_2, p_3, $\mathbb{G}$ and $\mathbb{G}_T$ are groups of order N and e is a map: $\mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Assume that attributes are elements of $\mathbb{Z}_N$ and the message space is $\mathbb{G}_T$. Our CP-HABE scheme is described as follows.

Setup(1^ℓ). Choose random generators $g \in \mathbb{G}_1$ and $g_3 \in \mathbb{G}_3$. Next, choose random elements $\alpha, a \in \mathbb{Z}_N$ and random group elements $h_1, h_2, \cdots, h_L, v_1, v_2, \cdots, v_D \in \mathbb{G}_1$. The public key PK and master secret key MSK are

$$PK = (\mathbf{U}, N, g, g_3, g^a, h_1, \cdots, h_L, v_1, \cdots, v_D, e(g,g)^\alpha), MSK = \alpha.$$

KeyGen(PK, MSK, S). To generate a secret key for the set S of attribute vectors of depth $k \leq L$, first choose random elements $w \in \mathbb{Z}_N$, $R_0, R_1 \in \mathbb{G}_3$ and compute

$$K_0 = g^\alpha g^{aw} R_0, \quad K_1 = g^w R_1.$$

Next, for each j from 1 to $|S|$, pick random elements

$$t_j \in \mathbb{Z}_N, R_{(j,0)}, R_{(j,1)}, R_{(j,k+1)}, R_{(j,k+2)}, \cdots, R_{(j,L)} \in \mathbb{G}_3.$$

For each attribute vector $\vec{u} = (u_1, u_2, \cdots, u_k)$ of S, choose v_x by finding u_1 as the x-th attribute of U_1 and compute its key component

$$K_{(j,0)} = v_x^w (h_1^{u_1} \cdots h_k^{u_k})^{t_j} R_{(j,0)}, \quad K_{(j,1)} = g^{t_j} R_{(j,1)}$$

$$K_{(j,k+1)} = h_{k+1}^{t_j} R_{(j,k+1)}, \quad \cdots, \quad K_{(j,L)} = h_L^{t_j} R_{(j,L)}.$$

Output $SK_S = \Big(K_0, K_1, \{K_{(j,0)}, K_{(j,1)}, K_{(j,k+1)}, \cdots, K_{(j,L)}\}_{j=1}^{|S|}\Big)$.

Delegate$(PK, SK_{S'}, S)$. To generate a secret key for S satisfying $S \Leftarrow S'$ by using

$$SK_{S'} = \Big(K'_0, K'_1, \{K'_{(j',0)}, K'_{(j',1)}, K'_{(j',k+1)}, \cdots, K'_{(j',L)}\}_{j'=1}^{|S'|}\Big),$$

first choose random elements $w \in \mathbb{Z}_N$, $R_0, R_1 \in \mathbb{G}_1$ and compute

$$K_0 = K'_0 g^{aw} R_1, \;\; K_1 = K'_1 g^w R_1.$$

Next, for each j from 1 to $|S|$, choose random elements

$$t_j \in \mathbb{Z}_N, R_{(j,0)}, R_{(j,1)}, R_{(j,k+2)}, R_{(j,k+3)}, \cdots, R_{(j,L)} \in \mathbb{G}_3.$$

For each attribute vector $\vec{u} \in S$, find its prefix $\vec{u}' \in S'$ such that $\vec{u} = (\vec{u}', u_{k+1})$ and use the key component $(K'_{(j',0)}, K'_{(j',1)}, K'_{(j',k+1)}, \cdots, K'_{(j',L)})$ of $\vec{u}'$ to compute

$$\begin{aligned} K_{(j,0)} &= K'_{(j',0)} v_x^w (K'_{(j',k+1)})^{u_{k+1}} (h_1^{u_1} \cdots h_{k+1}^{u_{k+1}})^{t_j} R_{(j,0)}, \\ K_{(j,1)} &= K'_{(j',1)} g^{t_j} R_{(j,1)}, \\ K_{(j,k+2)} &= K'_{(j',k+2)} h_{k+2}^{t_j} R_{(j,k+2)}, \\ &\vdots \\ K_{(j,L)} &= K'_{(j',L)} h_L^{t_j} R_{(j,L)}, \end{aligned}$$

Output

$$SK_S = \left(K_0, K_1, \{K_{(j,0)}, K_{(j,1)}, K_{(j,k+1)}, \cdots, K_{(j,L)}\}_{j=1}^{|S|} \right).$$

We note that, since $w + w'$ and $t_j + t'_{j'}$ (assume that $w', t'_{j'}$ are the random exponents used in generating $SK_{S'}$) are uniformly random, this key is identically distributed to one directly output by the key generation algorithm.

Encrypt$(M, PK, \mathbb{A})$. To encrypt a message $M \in \mathbb{G}_T$ under the access structure $\mathbb{A} = (\mathbf{A}, \rho)$ (where $\mathbf{A}$ is the share-generating matrix with l rows and n columns and ρ is the function mapping one row of $\mathbf{A}$ to an attribute vector of depth k), select random elements $s, s_2, s_3, \cdots, s_n \in \mathbb{Z}_N$ to form a vector

$$\vec{s} = (s, s_2, \cdots, s_n).$$

For the i-th row A_i of $\mathbf{A}$, which is mapped by ρ to an attribute vector $\vec{u} = (u_1, \cdots, u_k)$, compute its share as $\lambda_i = A_i \vec{s}$ and choose v_x by finding u_1 as the x-th attribute of U_1. Then, pick random element $r_i \in \mathbb{Z}_N$ and calculate

$$C_{(i,0)} = g^{a\lambda_i} v_x^{r_i},\ C_{(i,1)} = g^{r_i},\ C_{(i,2)} = (h_1^{a_1} \cdots h_k^{a_k})^{r_i}.$$

Also compute

$$C_0 = Me(g,g)^{\alpha s},\ C_1 = g^s.$$

Output

$$CT = \left(C_0, C_1, \{C_{(i,0)}, C_{(i,1)}, C_{(i,2)}\}_{i=1}^{l} \right).$$

Decrypt(CT, SK_S, PK). Given $CT = \big(C_0, C_1, \{C_{(i,0)}, C_{(i,1)}, C_{(i,2)}\}_{i=1}^{l}\big)$ for $\mathbb{A} = (\mathbf{A}, \rho)$ regarding attribute vectors of depth k and

$$SK_S = \Big(K_0, K_1, \{K_{(j,0)}, K_{(j,1)}, K_{(j,k+1)}, \cdots, K_{(j,L)}\}_{j=1}^{|S|}\Big)$$

for S of attribute vectors of depth k, if $S \in \mathbb{A}$, compute the constants $\{\omega_i \in \mathbb{Z}_N\}_{\rho(i)\in S}$ such that

$$\sum_{\rho_(i)\in S} \omega_i A_i = (1, 0, \cdots, 0).$$

Compute

$$B = \prod_{\rho(i)\in S} \left(\frac{e(C_{(i,0)}, K_1) \cdot e(C_{(i,2)}, K_{(j,1)})}{e(C_{(i,1)}, K_{(j,0)})} \right)^{\omega_i},$$

and

$$M' = e(C_1, K_0)/B.$$

Output $M = C_0/M'$.

Correctness. We observe that

$$\begin{aligned} B &= \prod_{\rho(i)\in S} \left(\frac{e\left(g^{a\lambda_i} v_x^{r_i}, g^w\right) \cdot e\left(g^{t_j}, (h_1^{u_1} \cdots h_k^{u_k})^{r_i}\right)}{e\left(v_x^w, g^{r_i}\right) \cdot e\left((h_1^{u_1} \cdots h_k^{u_k})^{t_j}, g^{r_i}\right)} \right)^{\omega_i} \\ &= e\left(g^a, g^w\right)^{\sum_{\rho(i)\in S} \omega_i \lambda_i} \\ &= e(g^a, g^w)^s \end{aligned}$$

and

$$M' = e\left(C_1, K_0\right)/B = \frac{e\left(g^s, g^\alpha g^{aw}\right)}{e\left(g^a, g^w\right)^s} = e(g^\alpha, g^s).$$

It follows that $M = C_0/M'$. The parts from group $\mathbb{G}_3$ cancel out because of the orthogonality property.

Performance. The maximum depth of the system is usually small in practice. Once it has been decided, the system public key is linear with the number of attributes in the row of the attribute matrix other than the number of total attributes in the system. The size of secret keys is linear with the product of the number of attribute vectors and their depth and will be only linear with the number of attribute vectors if in the deepest level. The encryption algorithm needs no pairing operation since the pair is pre-computed at the setup phase. For any receiver, the ciphertexts are independent of the receiver's hierarchy and only linear with the number of attribute vectors belonging to the access structure. The pairing operations in the decryption algorithm are linear with the number of attribute vectors of the set which satisfies the access structure. We note that the efficiency of decryption is similar as with other ABE schemes ([29, 15, 18]), which means the performance does not degrade when delegation is introduced.

Security. The security of the scheme is guaranteed by the following theorem.

Theorem 1. *Our CP-HABE system is fully secure in the standard model if Assumptions 1, 2 and 3 hold.*

Since our CP-HABE scheme is constructed on the CP-ABE scheme [15], our security proof is similar to the latter's. For a top-level view, we adopt the dual system encryption proof approach in [15] but refine their approach with less games. This approach has been shown a powerful tool in proving the full security of properly designed HIBE and ABE schemes (e.g., [17, 18, 15, 28]). We construct semi-functional CP-HABE keys and semi-functional CP-HABE ciphertexts. A semi-functional CP-HABE key (semi-functional key for short) can be used to decrypt normal ciphertexts and a semi-functional CP-HABE ciphertext (semi-functional ciphertext for short) can be decrypted by using normal keys. However, a semi-functional key cannot be used to decrypt a semi-functional ciphertext.

In the following proof, we define a sequence of games arguing that an attacker cannot distinguish one game from the next. The first game is $Game_{real}$, which is the real security game. The second game is $Game_0$, which is the same as $Game_{real}$ except that the challenge ciphertext is semi-functional. Let q denote the number of key queries made by the attacker. For all $\nu = 1, \cdots, q$, we define $Game_\nu$, in which the first ν keys are semi-functional and the remaining keys are normal, while the challenge ciphertext is semi-functional. Note that when $\nu = q$, in $Game_q$, all keys are semi-functional. The last game is defined as $Game_{final}$ where all keys are semi-functional and the ciphertext is a semi-functional encryption of a random message. In the following lemmas, we will prove that these games are indistinguishable under Assumptions 1, 2 and 3.

The semi-functional ciphertexts and keys are constructed as follows.

Semi-functional ciphertext. Let g_2 denote the generator of $\mathbb{G}_2$. We call the algorithm **Encrypt** to form normal ciphertext

$$(C'_0, C'_1, \{C'_{(i,0)}, C'_{(i,1)}, C'_{(i,2)}\}_{\forall i}).$$

For each row $A_i \in \mathbf{A}$, we choose a random exponent $\sigma_i \in \mathbb{Z}_N$. We choose a random integer $c \in \mathbb{Z}_N$ and a random vector $\vec{\chi} \in \mathbb{Z}_N^n$, as well as random elements $\zeta_1, \zeta_2, \cdots, \zeta_D, \eta_1, \eta_2, \cdots, \eta_L \in \mathbb{Z}_N$. We set the semi-functional ciphertext to be

$$C_0 = C'_0, \quad C_1 = C'_1 g_2^c,$$

$$\begin{aligned}
C_{(i,0)} &= C'_{(i,0)} g_2^{A_i\vec{\chi}+\sigma_i\zeta_x} = g^{a\lambda_i} v_x^{r_i} g_2^{A_i\vec{\chi}+\sigma_i\zeta_x}, \\
C_{(i,1)} &= C'_{(i,1)} g_2^{\sigma_i} = g^{r_i} g_2^{\sigma_i}, \\
C_{(i,2)} &= C'_{(i,2)} g_2^{\sigma_i \sum_{m=1}^{k} \eta_m u_m} = (h_1^{u_1} \cdots h_k^{u_k})^{r_i} g_2^{\sigma_i \sum_{m=1}^{k} \eta_m u_m}.
\end{aligned}$$

Semi-functional key. We first call the algorithm **KeyGen** to form normal key

$$(K'_0, K'_1, \{K'_{(j,0)}, K'_{(j,1)}, K'_{(j,k+1)}, \cdots, K'_{(j,L)}\}_{\forall j}).$$

Then we choose random elements $b, d \in \mathbb{Z}_N$ and random element $\varsigma_j \in \mathbb{Z}_N$ for each attribute vector. The semi-functional key is set as:

$$K_0 = K'_0 g_2^b, \; K_1 = K'_1 g_2^d,$$

$$K_{(j,0)} = K'_{(j,0)} g_2^{d\zeta_x + \varsigma_j \Sigma_{m=1}^{k} u_m \eta_m}, \; K_{(j,1)} = K'_{(j,1)} g_2^{\varsigma_j},$$

$$K_{(j,k+1)} = K'_{(j,k+1)} g_2^{\varsigma_j \eta_{k+1}}, \cdots, \; K_{(j,L)} = K'_{(j,L)} g_2^{\varsigma_j \eta_L}.$$

Remark 1. We note that if we use a semi-functional key to decrypt a semi-functional ciphertext, we will have

$$e(g_2, g_2)^{cb - d\chi_1},$$

where χ_1 is the first coordinate of vector $\vec{\chi}$. If $cb - d\chi_1 = 0 \mod p_2$, the decryption will still work. Then we call the semi-functional key satisfying this condition the *nominally* semi-functional key.

We need this nominally semi-functional key because there is a paradox when the simulator transforms the challenge key from being normal to being semi-functional. That is, the simulator (prepared to generate semi-functional challenge ciphertext and the challenge key) could have made a key and test whether it was normal or semi-functional by attempting to decrypt the challenge ciphertext using this key. To avoid this paradox, we make sure that the simulator transforms the challenge key from being normal to being nominally semi-functional, meaning that the challenge key can still decrypt the challenge ciphertext regardless of the nature of this key. We will prove that this kind of keys is properly distributed as the semi-functional keys.

Lemma 2. *If an attacker $\mathcal{A}$ can distinguish $Game_{real}$ from $Game_0$ with advantage ϵ, then there exists an algorithm $\mathcal{B}$ to break Assumption 1 with advantage ϵ.*

Proof. Given the input tuple (g, g_3, T) of Assumption 1, we construct an algorithm $\mathcal{B}$ to simulate $Game_{real}$ or $Game_0$ interacting with $\mathcal{A}$.

Setup: $\mathcal{B}$ chooses two random exponents $a, \alpha \in \mathbb{Z}_N$ and for all $i = 1, \cdots, D$ and all $j = 1, \cdots, L$, it chooses random exponents $\zeta'_i, \eta'_j \in \mathbb{Z}_N$. Then it computes

$$v_i = g^{\zeta'_i},\ h_j = g^{\eta'_j}.$$

It gives $\mathcal{A}$ the following public key

$$PK = (\mathcal{U}, N,\ g,\ g_3,\ g^a,\ v_1,\ \cdots,\ v_D,\ h_1,\ \cdots,\ h_L,\ e(g,g)^\alpha).$$

Key generation Phase 1, **Phase 2**: We note that knowledge of the master key $MSK = (\alpha, X_3)$ allows $\mathcal{B}$ to run the key generation algorithm to generate normal keys.

Challenge: The attacker $\mathcal{A}$ gives $\mathcal{B}$ two equal-length messages M_0 and M_1, and an access structure $\mathbb{A}^*$ over attribute vectors of depth k. $\mathcal{B}$ generates an LSSS $(\mathbf{A}^*, \rho^*)$ for $\mathbb{A}^*$. Then, for the i-th row of $\mathbf{A}^*$ $(1 \leq i \leq l)$, $\mathcal{B}$ chooses a random exponent $r'_i \in \mathbb{Z}_N$. It chooses random elements $s_2, \cdots, s_n \in \mathbb{Z}_N$ to form vector $\vec{s^*} = (1, s_2, \cdots, s_n)$. $\mathcal{B}$ flips a coin $\beta \in \{0, 1\}$ and sets:

$$C_0 = M_\beta e(g,T)^\alpha, C_1 = T,$$

$$C_{(i,0)} = T^{aA_i\vec{s^*}} T^{r'_i\zeta_x},\quad C_{(i,1)} = T^{r'_i},\quad C_{(i,2)} = T^{r'_i \sum_{m=1}^{k} \eta'_m u_m}.$$

If $T \in \mathbb{G}_{(1,2)}$, assuming $T = g^s g_2^c$, this implicitly sets

$$\vec{\chi} = ca\vec{s^*}, r_i = sr'_i, \sigma_i = cr'_i, \zeta_1 = \zeta'_i, \cdots, \zeta_D = \zeta'_D, \eta_1 = \eta'_1, \cdots, \eta_L = \eta'_L.$$

This means we reuse $a, c, r'_i, \zeta_1, \cdots, \zeta_D, \eta_1, \cdots, \eta_L$ to form the $\mathbb{G}_2$ part of this ciphertext. Note that these values modulo p_1 are not related to their values modulo p_2 by the Chinese Remainder Theorem. Thus this operation does not incur unwanted correlation. Then, this ciphertext is properly distributed as the semi-functional ciphertext. If $T \in \mathbb{G}_1$, this ciphertext is also identically distributed as a normal ciphertext by assuming $T = g^s$ for some unknown $s \in \mathbb{Z}_N$.

Guess: If $T \in \mathbb{G}_{(1,2)}$, we are in $Game_0$. If $T \in \mathbb{G}_1$, we are in $Game_{real}$. If $\mathcal{A}$ outputs β' such that $\beta' = \beta$, $\mathcal{B}$ outputs 0. Therefore, with the tuple (g, g_3, T), we have the advantage of $\mathcal{B}$ in breaking Assumption 1:

$$|\Pr[\mathcal{B}(g, g_3, T \in \mathbb{G}_{(1,2)}) = 0] - \Pr[\mathcal{B}(g, g_3, T \in \mathbb{G}_1) = 0]|$$

$$= |Game_0\text{Adv}_{\mathcal{A}} - Game_{real}\text{Adv}_{\mathcal{A}}| = \epsilon,$$

where $Game_0\text{Adv}_{\mathcal{A}}$ is the advantage of $\mathcal{A}$ in $Game_0$ and $Game_{real}\text{Adv}_{\mathcal{A}}$ is the advantage of $\mathcal{A}$ in $Game_{real}$.

□

Lemma 3. *If an attacker $\mathcal{A}$ can distinguish $Game_{\nu-1}$ from $Game_\nu$ with advantage ϵ, then there exists an algorithm $\mathcal{B}$ to break the Assumption 2 with advantage ϵ.*

Proof. Given the tuple $(g, \theta_1\theta_2, g_3, \gamma_2\gamma_3, T)$ of Assumption 2, algorithm $\mathcal{B}$ simulates $Game_{\nu-1}$ or $Game_\nu$ interacting with $\mathcal{A}$.

Setup: $\mathcal{B}$ generates PK as the same in Lemma 1 and gives PK to $\mathcal{A}$.

Key generation Phase 1, **Phase 2**: For the first $\nu - 1$ key queries, $\mathcal{B}$ simulates the semi-functional keys. It randomly chooses

$$w, b', d' \in \mathbb{Z}_N,\ R_0, R_1 \in \mathbb{G}_3, \text{ and } \zeta_i', \eta_j' \in \mathbb{Z}_N$$

for each i from 1 to D and each j from 1 to L. For each attribute vector of set S, $\mathcal{B}$ picks random elements

$$t_j, \varsigma_j' \in \mathbb{Z}_N, R_{(j,0)}, R_{(j,1)}, R_{(j,k+1)}, \cdots, R_{(j,L)} \in \mathbb{G}_3,$$

where $1 \leq j \leq |S|$. Then, $\mathcal{B}$ sets the secret key to be

$$\begin{aligned}
K_0 &= g^{\alpha} g^{aw} R_0 (\gamma_2\gamma_3)^{b'}, \quad K_1 = g^{w} (\gamma_2\gamma_3)^{d'} R_1, \\
K_{(j,0)} &= v_x^{w} (h_1^{u_1} \cdots h_k^{u_k})^{t_j} R_{(j,0)} (\gamma_2\gamma_3)^{d'(\zeta'_x + \varsigma'_j \sum_{m=1}^{k} \eta'_m u_m)}, \\
K_{(j,1)} &= g^{t_j} R_{(j,1)} (\gamma_2\gamma_3)^{d'\varsigma'_j}, \\
K_{(j,k+1)} &= h_{k+1}^{t_j} R_{(j,k+1)} (\gamma_2\gamma_3)^{d'(\varsigma'_j \eta'_{k+1})}, \\
&\vdots \\
K_{(j,L)} &= h_L^{t_j} R_{(j,L)} (\gamma_2\gamma_3)^{d'(\varsigma'_j \eta'_L)}.
\end{aligned}$$

If we assume $\gamma_2 = g_2^{c_2}$, this has implicitly set

$$b = b'c_2, d = d'c_2, \varsigma_j = \varsigma'_j d' c_2.$$

We note that this key is identically distributed as the semi-functional key.

For the rest of keys other than the ν-th one, $\mathcal{B}$ can generate the normal keys by simply running the key generation algorithm because it knows the master secret key.

To form the ν-th key for the set S, for each attribute vector $\vec{u}$ of S, $\mathcal{B}$ chooses random exponents

$$t'_j \in \mathbb{Z}_N, \ R_0, R_1, R_{(j,0)}, R_{(j,1)}, R_{(j,k+1)}, \cdots, R_{(j,L)} \in \mathbb{G}_3,$$

where $1 \leq j \leq |S|$, then computes

$$K_0 = g^{\alpha} T^{a} R_0, \quad K_1 = T R_1,$$

$$\begin{aligned}
K_{(j,0)} &= T^{\zeta_x} T^{t'_j \sum_{m=1}^{k} \eta_m u_m} R_{(j,0)}, \quad K_{(j,1)} = T^{t'_j} R_{(j,1)}, \\
K_{(j,k+1)} &= T^{t'_j \eta_{k+1}} R_{(j,k+1)}, \quad \cdots, \quad K_{(j,L)} = T^{t'_j \eta_L} R_{(j,L)}.
\end{aligned}$$

If $T \in \mathbb{G}_{(1,3)}$, this key is identically distributed as a normal key. If $T \in \mathbb{G}$, supposing $T = g^{w} g_2^{d} g_3^{c_3}$, this implicitly sets

$$b = da, t_j = wt'_j, \varsigma_j = dt'_j, \zeta_1 = \zeta'_1, \cdots, \zeta_D = \zeta'_D, \eta_1 = \eta'_1, \cdots, \eta_L = \eta'_L.$$

Because the values of $a, t'_j, \zeta_1, \cdots, \zeta_D, \eta_1, \cdots, \eta_L$ modulo p_1 are not related to their values modulo p_2 by the Chinese Remainder Theorem, there is no correlation between the $\mathbb{G}_1$ part and the $\mathbb{G}_2$ part of this key. Thus, this key is identically distributed as the semi-functional key.

Challenge: When $\mathcal{A}$ queries the challenge ciphertext by sending two equal-length messages M_0, M_1 and an access structure $\mathbb{A}^*$, $\mathcal{B}$ generates an LSSS $(\mathbf{A}^*, \rho^*)$ for $\mathbb{A}^*$ and chooses a random exponent $r'_i \in \mathbb{Z}_N$ for each row A_i of $\mathbf{A}^*$. It sets a vector

$$\vec{s^*} = (1, s_2, \cdots, s_n),$$

where $s_2, \cdots, s_n$ are random elements in $\mathbb{Z}_N$. Then, $\mathcal{B}$ flips a coin $\beta \in \{0, 1\}$ and sets the ciphertext to be:

$$C_0 = M_\beta e(g, \theta_1\theta_2)^\alpha,\ C_1 = \theta_1\theta_2,$$

$$C_{(i,0)} = (\theta_1\theta_2)^{aA_i\vec{s^*}} T^{r'_i\zeta_x},\ C_{(i,1)} = (\theta_1\theta_2)^{r'_i},\ C_{(i,2)} = (\theta_1\theta_2)^{r'_i \sum_{m=1}^{k} \eta_m u_m}.$$

Assume $\theta_1\theta_2 = g^s g_2^c$, which implicitly sets

$$\vec{\chi} = ca\vec{s^*}, r_i = sr'_i, \sigma_i = cr'_i, \zeta_1 = \zeta'_1, \cdots, \zeta_D = \zeta'_D, \eta_1 = \eta'_1, \cdots, \eta_L = \eta'_L.$$

But again there is no correlation between the $\mathbb{G}_1$ part and the $\mathbb{G}_2$ part by the Chinese Remainder Theorem. Thus, this ciphertext is identically distributed as the semi-functional ciphertext.

When the simulator $\mathcal{B}$ uses the ν-th key to decrypt the semi-functional ciphertext in order to test whether the key is normal or semi-functional, it will obtain

$$e(g_2, g_2)^{cb - d\chi_1} = 1.$$

This is because, in the simulation of the challenge ciphertext, we have $\vec{\chi} = ca\vec{s^*}$, then $\chi_1 = ca$; in the simulation of the ν-th key, we have $b = da$. Then it holds that $cb - d\chi_1 = d(ca - \chi_1) = 0$.

This implies that the decryption will still work and this key is either normal or nominally semi-functional. We have to argue that $\chi_1 = ca$ being shared in $\mathbb{G}_2$ is hidden to the attacker $\mathcal{A}$, who cannot request any keys that can be used to decrypt the challenge ciphertext.

Since the ν-th key cannot decrypt the challenge ciphertext, the vector $(1, 0, \cdots, 0)$ is not included in $\mathbf{R}_S$, which is a submatrix of $\mathbf{A}^*$ and formed by those rows whose mapped hierarchical attributes are in the key. From the basics of linear algebra and similarly to Proposition 11 in [16], we have the following proposition:

Proposition 1. A vector $\vec{\pi}$ is linearly independent of a set of vectors represented by a matrix $\mathbf{M}$ if and only if there exists a vector $\vec{v}$ such that $\mathbf{M}\vec{v} = 0$ while $\vec{\pi}\vec{v} = 1$.

Since $(1, 0, \cdots, 0)$ is linearly independent of $\mathbf{R}_S$, we can find a vector $\vec{v}$ such that for each row $A_i \in \mathbf{R}_S$, it holds that $A_i\vec{v} = 0$ and $\vec{v}(1, 0, \cdots, 0) = 1$. By choosing a random vector $\vec{\chi}''$, we can set

$$A_i\vec{\chi} = A_i(\xi\vec{v} + \vec{\chi}''),$$

where $\xi \in \mathbb{Z}_N$ needs to be hidden.

If $\rho^*(i) \in S$, then $A_i\vec{\chi} = A_i\vec{\chi}''$. Hence no information about ξ is revealed and $A_i\vec{\chi}$ is hidden.

If $\rho^*(i) \notin S$, then it holds that

$$A_i\vec{\chi} + \sigma_i\zeta_x = A_i(\xi\vec{v} + \vec{\chi}'') + \sigma_i\zeta_x,$$

where σ_i is a random value and appears only once because ρ^* is an injection function. As long as each σ_i is not congruent to 0 modulo p_2, the value of $A_i\xi\vec{v}$ is randomized by σ_i. Hence no information about ξ is revealed. The probability that all σ_i's are 0 modulo p_2 is negligible. Therefore the value being shared in $\mathbb{G}_2$ is information-theoretically hidden in $\mathcal{A}$'s view with probability very close to 1.

Guess: If $T \in \mathbb{G}_{(1,3)}$, we are in $Game_{\nu-1}$. Else if $T \in \mathbb{G}$, we are in $Game_\nu$. If $\mathcal{A}$ outputs $\beta' = \beta$, then $\mathcal{B}$ outputs 0. Therefore, with the input tuple $(g, \theta_1\theta_2, g_3, \gamma_2\gamma_3, T)$, the advantage of $\mathcal{B}$ in breaking Assumption 2 is:

$$|\Pr[\mathcal{B}(g, \theta_1\theta_2, g_3, \gamma_2\gamma_3, T \in \mathbb{G}_{(1,3)}) = 0] - \Pr[\mathcal{B}(g, \theta_1\theta_2, g_3, \gamma_2\gamma_3, T \in \mathbb{G}) = 0]|$$

$$= |Game_{\nu-1}\mathrm{Adv}_{\mathcal{A}} - Game_\nu\mathrm{Adv}_{\mathcal{A}}| = \epsilon,$$

where $Game_{\nu-1}\mathrm{Adv}_{\mathcal{A}}$ is the advantage of $\mathcal{A}$ in $Game_{\nu-1}$ and $Game_\nu\mathrm{Adv}_{\mathcal{A}}$ is the advantage of $\mathcal{A}$ in $Game_\nu$.

□

Lemma 4. *If an attacker $\mathcal{A}$ can distinguish $Game_q$ from $Game_{final}$ with advantage ϵ, then there exists an algorithm $\mathcal{B}$ to break Assumption 3 with advantage ϵ.*

Proof. Given the challenging tuple $(g, g^\alpha\theta_2, g_3, g^s\gamma_2, \delta_2, T)$ of Assumption 3, $\mathcal{B}$ will simulate $Game_q$ or $Game_{final}$ to interact with $\mathcal{A}$.

Setup: For all $i = 1, \cdots, D$ and all $j = 1, \cdots, L$, $\mathcal{B}$ chooses random exponents $\zeta_i', \eta_j' \in \mathbb{Z}_N$ and computes

$$v_i = g^{\zeta_i'}, h_j = g^{\eta_j'}.$$

Then $\mathcal{B}$ chooses a random $a \in \mathbb{Z}_N$, generates public key

$$PK = (\mathcal{U}, N, g, g^a, g_3, v_1, \cdots, v_D, h_1, \cdots, h_L, e(g, g^\alpha \theta_2))$$

and gives this PK to $\mathcal{A}$. We note that $\mathcal{B}$ does not know the secret α.

Key generation Phase 1, **Phase 2**: In order to create a semi-functional key for S, $\mathcal{B}$ chooses random exponents $w, d' \in \mathbb{Z}_N$ and for each attribute vector it selects random elements

$$t_j, \varsigma'_j \in \mathbb{Z}_N \text{ and } R_0, R_1, R_{(j,0)}, R_{(j,1)}, R_{(j,k+1)}, \cdots, R_{(j,l)} \in \mathbb{G}_3.$$

It then computes:

$$\begin{aligned}
K_0 &= g^\alpha \theta_2 g^{aw} \delta_2^{d'} R_0, \quad K_1 = g^w \delta_2^{d'} R_1, \\
K_{(j,0)} &= v_x^w (h_1^{u_1} \cdots h_k^{u_k})^{t_j} \delta_2^{d'(\zeta'_x + \varsigma'_j \sum_{m=1}^k u_m \eta'_m)} R_{(j,0)}, \\
K_{(j,1)} &= g^{t_j} \delta_2^{d'\varsigma'_j} R_{(j,1)}, \\
K_{(j,k+1)} &= h_{k+1}^{t_j} \delta_2^{d'\varsigma'_j \eta'_{k+1}} R_{(j,k+1)}, \\
&\vdots \\
K_{(j,L)} &= h_L^{t_j} \delta_2^{d'\varsigma'_j \eta'_L} R_{(j,L)}.
\end{aligned}$$

Assuming $\theta_2 = g_2^{c_2}, \delta_2 = g_2^{\tau_2}$, we implicitly set

$$b = c_2 + \tau_2 d', d = \tau_2 d', \varsigma_j = \tau_2 d' \varsigma'_j.$$

We note that this key is identically distributed as the semi-functional key in $\mathcal{A}$'s view.

Challenge: When $\mathcal{B}$ is given two equal-length messages M_0, M_1 and an access structure $\mathbb{A}^*$, it generates a LSSS $(\mathbf{A}^*, \rho^*)$ for that access structure. It chooses a vector $\vec{s}^* = (1, s_2, \cdots, s_n)$, where $s_2, \cdots, s_n$ are random elements in $\mathbb{Z}_N$. For each row A_i of $\mathbf{A}^*$, it chooses random exponent $r'_i \in \mathbb{Z}_N$. Then it flips a random coin $\beta \in \{0, 1\}$ and sets the ciphertext to be:

$$C_0 = M_\beta T, \ C_1 = g^s \gamma_2,$$

$$C_{(i,0)} = (g^s \gamma_2)^{aA_i \vec{s}^*} (g^s \gamma_2)^{r'_i \zeta'_x}, \ C_{(i,1)} = (g^s \gamma_2)^{r'_i}, \ C_{(i,2)} = (g^s \gamma_2)^{r'_i \sum_{m=1}^k u_m \eta'_m}.$$

If $T = e(g,g)^{\alpha s}$, assuming $\gamma_2 = g_2^c$, this implicitly sets

$$\vec{\chi} = ca\vec{s}^*, r_i = sr_i', \sigma_i = cr_i', \zeta_1 = \zeta_i', \cdots, \zeta_D = \zeta_D', \eta_1 = \eta_1', \cdots, \eta_L = \eta_L'.$$

But again there is no correlation between $a, r_i', \zeta_1, \cdots, \zeta_D, \eta_1, \cdots, \eta_L \mod p_1$ and their values modulo p_2 by the Chinese Remainder Theorem. Therefore, this ciphertext is the semi-functional ciphertext of message M_β. If T is a random element in $\mathbb{G}_T$, this ciphertext is a semi-functional ciphertext of a random message.

Guess: If $T = e(g,g)^\alpha$, we are in $Game_q$. If T is a random element in $\mathbb{G}_T$, we are in $Game_{final}$. $\mathcal{B}$ outputs 0 when $\mathcal{A}$ outputs $\beta' = \beta$. Then with the input tuple $(g, g^\alpha\theta_2, g_3, g^s\gamma_2, \delta_2, T)$, the advantage of $\mathcal{B}$ in breaking Assumption 3 is:

$$\Pr\left[\mathcal{B}\begin{pmatrix} g, g^\alpha\theta_2 \\ g_3, g^s\gamma_2, \delta_2 \\ T = e(g,g)^\alpha \end{pmatrix} = 0\right] - \Pr\left[\mathcal{B}\begin{pmatrix} g, g^\alpha\theta_2 \\ g_3, g^s\gamma_2, \delta_2 \\ T \leftarrow \mathbb{G}_T \end{pmatrix} = 0\right]$$
$$= |Game_q\text{Adv}_\mathcal{A} - Game_{final}\text{Adv}_\mathcal{A}| = \epsilon,$$

where $Game_q\text{Adv}_\mathcal{A}$ is the advantage of $\mathcal{A}$ in $Game_q$ and $Game_{final}\text{Adv}_\mathcal{A}$ is the advantage of $\mathcal{A}$ in $Game_{final}$.

□

In $Game_{final}$, the ciphertext completely hides the bit β, so the advantage of $\mathcal{A}$ in this game is negligibly close to 0. Through the previous lemmas, we have shown that the real security game $Game_{real}$ is indistinguishable from the $Game_{final}$. Therefore, the advantage of $\mathcal{A}$ in $Game_{real}$ is negligibly close to 0 too. Hence, there is no polynomial-time adversary with a non-negligible advantage in breaking our CP-HABE system. This completes the proof of Theorem 1.

5. Conclusion

In this paper we have extended ABE to CP-HABE to support hierarchical attributes and delegation of secret keys. The new primitive greatly relieves the key generator in conventional ABE systems from heavy key management burden and it facilitates sensitive data sharing between large organizations. CP-HABE is especially suited for applications where the key generator cannot directly generate keys for all the users due to various reasons. We realized an efficient CP-HABE scheme with short ciphertexts. The scheme is proven to be fully secure under three static assumptions in the standard model.

Acknowledgments and disclaimer

This work was partly funded by Nature Science Foundation of China through projects 61173154, 61003214, 61272501 and 61202465, by the National Key Basic Research Program (973 program) through project 2012CB315905, by the Beijing Natural Science Foundation through project 4132056, and by the Fundamental Research Funds for the Central Universities through project 2012211020212, by the European Commission under FP7 projects "DwB" and "Inter-Trust", and by the Spanish Government through projects IPT-2012-0603-430000, TIN2012-32757, and TIN2011-27076-C03-01. The fourth author is partly supported as an ICREA-Acadèmia researcher by the Government of Catalonia. He holds the UNESCO Chair in Data Privacy, but the views expressed in this paper are his own and do not commit UNESCO.

References

[1] M. Abdalla, E. Kiltz, G. Neven, Generalized key delegation for hierarchical identity-based encryption, in: ESORICS'07, Springer Berlin Heidelberg, 2007, pp. 139-154.

[2] A. Beimel, Secure Schemes for Secret Sharing and Key Distribution, PhD thesis, Israel Institute of Technology, Technion, Haifa, Israel (1996).

[3] D. Boneh, X. Boyen, E. Goh, Hierarchical identity based encryption with constant size ciphertext, in: EUROCRYPT'05, Springer Berlin Heidelberg, 2005, pp. 440-456.

[4] D. Boneh, M. Franklin, Identity-based encryption from the Weil pairing, in: CRYPTO'01, Springer Berlin Heidelberg, 2001, pp. 213-229.

[5] D. Boneh, E. Goh, K. Nissim, Evaluating 2-DNF formulas on ciphertexts, in: TCC'05, Springer Berlin Heidelberg, 2005, pp. 325-341.

[6] J. Bethencourt, A. Sahai, B. Waters, Ciphertext-policy attribute-based encryption, in: IEEE Symposium on Security and Privacy, IEEE, 2007, pp. 321-334.

[7] L. Chen, Z. Cheng, N. P. Smart, Identity-based key agreement protocols from pairings, International Journal of Information Security, 2007, 6(4): 213-241.

[8] M. Chase, Multi-authority attribute based encryption, in: TCC'07, Springer Berlin Heidelberg, 2007, pp. 515-534.

[9] M. Green, G. Ateniese, Identity-based proxy re-encryption, in: ACNS'07, Springer Berlin Heidelberg, 2007, pp. 288-306.

[10] V. Goyal, A. Jain, O. Pandey, A. Sahai, Bounded ciphertext policy attribute based encryption, in: ICALP'08, Springer Berlin Heidelberg, 2008, pp. 579-591.

[11] V. Goyal, O. Pandey, A. Sahai, B. Waters, Attribute-based encryption for fine-grained access control of encrypted data, in: ACM CCS'06, ACM Press, 2006, pp. 89-98.

[12] J. Horwitz, B. Lynn, Toward hierarchical identity-based encryption, in: EUROCRYPT'02, Springer Berlin Heidelberg, 2002, pp. 466-481.

[13] S. Hohenberger, B. Waters, Attribute-based encryption with fast decryption, in: PKC'13, Springer Berlin Heidelberg, 2013, pp. 162-179.

[14] J. Katz, A. Sahai, B. Waters, Predicate encryption supporting disjunctions, polynomial equations, and inner products, in: EUROCRYPT'08, Springer Berlin Heidelberg, 2008, pp. 146-162.

[15] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, B. Waters, Fully secure functional encryption: attribute-based encryption and (hierarchical) inner product encryption, in: EUROCRYPT'10, Springer Berlin Heidelberg, 2010, pp. 62-91.

[16] A. Lewko, A. Sahai, B. Waters, Revocation systems with very small private keys, in: IEEE Symposium on Security and Privacy, IEEE, 2010, pp. 273-285.

[17] A. Lewko, B. Waters, New techniques for dual system encryption and fully secure HIBE with short ciphertexts, in: TCC'10, Springer Berlin Heidelberg, 2010, pp. 455-479.

[18] A. Lewko, B. Waters, Unbounded HIBE and attribute-based encryption, in: EUROCRYPT'11, Springer Berlin Heidelberg, 2011, pp. 547-567.

[19] H. Lin, Z. Cao, X. Liang, J. Shao, Secure threshold multi authority attribute based encryption without a central authority, Information Sciences, 2010, 180(13): 2618-2632.

[20] S. Liu, Y. Long, K. Chen, Key updating technique in identity-based encryption, Information Sciences, 2011, 181(11): 2436-2440.

[21] R. Ostrovsky, A. Sahai, B. Waters, Attribute-based encryption with non-monotonic access structures, in: ACM CCS'07, ACM Press, 2007, pp. 195-203.

[22] T. Okamoto, K. Takashima, Hierarchical predicate encryption for inner-products, in: ASIACRYPT'09, Springer Berlin Heidelberg, 2009, pp. 214-231.

[23] T. Okamoto, K. Takashima, Fully secure functional encryption with general relations from the decisional linear assumption, in: CRYPTO'10, Springer Berlin Heidelberg, 2010, pp. 191-208.

[24] A. Shamir, Identity-based cryptosystems and signature schemes, in: CRYPTO'84, Springer Berlin Heidelberg, 1984, pp. 47-53.

[25] J. Shao, Z. Cao, Multi-use unidirectional identity-based proxy re-encryption from hierarchical identity-based encryption, Information Sciences, 2012, 206:83-95.

[26] A. Sahai, B. Waters, Fuzzy identity-based encryption, in: EUROCRYPT'05, Springer Berlin Heidelberg, 2005, pp. 457-473.

[27] B. Waters, Efficient identity-based encryption without random oracles, in: EUROCRYPT'05, Springer Berlin Heidelberg, 2005, pp. 114-127.

[28] B. Waters, Dual system encryption: realizing fully secure IBE and HIBE under simple assumptions, in: CRYPTO'09, Springer Berlin Heidelberg, 2009, pp. 619-636.

[29] B. Waters, Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization, in: PKC'11, Springer Berlin Heidelberg, 2011, pp. 53-70.

[30] H. Wang, Z. Cao, L. Wang, Multi-use and unidirectional identity-based proxy re-encryption schemes, Information Sciences, 2010, 180(20): 4042-4059.

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

[32] Z. Wan, J. Liu, R. H. Deng, HASBE: a hierarchical attribute-based solution for flexible and scalable access control in cloud computing, IEEE Transactions on Information Forensics and Security, 2012, 7(2): 743-754.

[33] J. Yu, R. Hao, F. Kong, X. Cheng, J. Fan, Y. Chen, Forward-secure identity-based signature: security notions and construction, Information Sciences, 2011, 181(3): 648-660.

[34] L. Zhang, Q. Wu, B. Qin, J. Domingo-Ferrer, Provably secure one-round identity-based authenticated asymmetric group key agreement protocol, Information Sciences, 2011, 181(19): 4318-4329.