# ID-HABE: Incorporating ID-based Revocation, Delegation, and Authority Hierarchy into Attribute-Based Encryption

Qiuxiang Dong, Dijiang Huang
ASU, Tempe, AZ, USA
Email: {Qiuxiang.Dong, dijiang}@asu.edu

Jim Luo, Myong Kang
Naval Research Lab, USA
{jim.luo, myong.kang}@nrl.navy.mil

***Abstract*—Ciphertext Policy Attribute-Based Encryption (CP-ABE) has been proposed to implement fine-grained access control. Data owners encrypt data with a certain access policy so that only data users whose attributes satisfy the access policy can decrypt the ciphertext. A user can be automatically assigned an access privilege based on whether his/her attributes satisfying a given access policy described by attributes and their logical relations. In order to provide more flexible policy-based access control, attribute-based revocation approaches had been proposed to provide the NOT logic on attributes to allow attribute-based revocation. However, previous solutions increase the attribute management overhead when considering each user's ID as an attribute for more precise revocations at the individual user-level. To address this issue, in this paper, an ID-ABE scheme is presented, where a user's ID is incorporated into the key generation procedure allowing user-ID-based revocation. In addition to ID-based revocation, ID-ABE also presents a hierarchical identity structure to build a delegation framework to enable group-based revocation. In the end, we also evaluate the performance of the proposed scheme in terms of computation, storage and communication overhead, which shows the practical value of the solution for secure data sharing applications.**

## I. Introduction

The literature proposes a diversity of access control systems supporting policies including basic access control lists [1], cryptographically-enforced capabilities [2], group-based [3], role-based [4] and attribute-based controls [5]. Most of these approaches rely on a fully-trusted access monitoring server to implement policy checking, which are not suitable for some practical applications, such as cloud computing where the cloud servers may not be fully trusted. Secure data sharing in these application scenarios pushes the development and usage of new cryptographic schemes in supporting access control models in a cloud-based data sharing service model. Among these cryptographic schemes, Ciphertext Policy Attribute-Based Encryption (CP-ABE) [6] is regarded as one of promising technologies and is a natural fit for building an attribute-based access control (ABAC) [7] architecture to support secure data sharing features such as policy-based data access control.

In CP-ABE scheme, each user is assigned a set of attributes based on his/her role and capabilities, which are used as public keys. The data owner can enforce data access control by encrypting the data with a data access policy expressed by a *policy tree* structure that is composed by a set of attributes and their logic relations (*e.g.*, AND, OR, $k$ out of $n$, *etc.*). The encrypted data can be placed on a public cloud storage server. A user can decrypt the ciphertext only if his assigned attributes can satisfy the logic enforced among attributes to reach the root of the policy tree. This approach is promising in that the access control policies are incorporated into the ciphertext naturally and there is no need for a trusted third-party to enforce the data access control during the run-time any more. Different from identity-based and role-based cryptographic schemes, the public key and ciphertext size of CP-ABE are not related with the number of data users and no interactions among data owners and data users are needed. Moreover, CP-ABE is resistant against collusion attacks from unauthorized users. All these nice properties make CP-ABE extremely suitable for implementing access control for secure data sharing in peer-to-peer environment.

Using attributes to present data access policies is a very flexible and scalable approach when data owners do not have a clear picture about who have been included in a desired policy confined group (called policy group, *i.e.*, described by a *policy tree*). However, one major drawback of using attributes is the introduced attributes management overhead when the data access requires the accuracy at the user-level, *e.g.*, revoking specific users from a policy group. To solve this problem, [8] proposed an indirect revocation approach in the cloud computing context. However, this approach relies on either key regeneration or complicated tree structure based on pre-defined relationships between users and attributes, thus bringing overwhelming overhead in dynamic system where users frequently join and leave. Another issue of this approach is that the trusted authority has to be online all the time to generate and distribute new private information to non-revoked users. Considering the aforementioned issues, this approach is not suitable for application in some application scenarios such as peer-to-peer networks. In P2P environment, no centralized trusted authority exists and all the points in the network only have constrained computing resources. Therefore, we need a revocation mechanism which is able to not only revoke users directly on the data owner end but also be efficient to be applied on mobile devices with limited computation power and battery life. In addition, the approach also needs to be

scalable for un-revoked users as well as when the number of revoked users is large.

To this end, Yamada *et. al* [9] proposed a scheme enabling negative logic (NOT), where a user can be revoked based on his assigned attributes or his ID that is regarded as a unique attribute. However, considering users' ID as an attribute suffers practical issues, *i.e.*, complicated attribute management due to significantly increased the number of attributes. In this paper, we present a new ID-based hierarchical ABE approach called ID-HABE to achieve precise user-level revocation. The salient feature of ID-HABE is incorporating a user's ID into his ABE private key. The encryption algorithm works by two integrated functions: (1) specify attribute literals in conjunctive/disjunctive normal forms as a policy tree structure to cover the recipients of the target policy group; (2) revoke unauthorized users by incorporating their identities into the ciphertext. In this way, only users whose attributes satisfy the policy tree structure and meanwhile are not revoked by the data owners can decrypt the ciphertext.

ID-HABE provides the following major security features compared to existing ABE solutions: (a) *Revocation:* it provides precise user-level control by including users' IDs into the *policy tree* for revocation. In this way, ID-HABE scheme can revoke a list of users regardless of their assigned attributes. This approach is desirable when a trusted revocation authority is not always online since the revocation list is always initiated by the data owner and the ciphertext is already incorporated the revocation information. Moreover, it overcomes the attributes exploration problem when considering a user's ID as an attribute. (b) *Delegation:* ID-HABE provides a hierarchical delegation framework established to manage users' IDs and assigned attributes. The delegation approach not only reduces the management overhead by distributing the key management tasks to multiple delegators, but also provides a scalable framework to revoke a large group of users by revoking their delegator's ID.

### A. Research Contributions

Our contributions are summarized as follows:

- ID-HABE provides an integrated approach to nicely incorporate users' ID into user's key generation procedure and eases the attribute management.
- ID-HABE not only supports revocation of individual users but also is able to effectively revoke all the users within the same delegation domain.
- ID-HABE supports key-generation delegation, where a domain authority is responsible for generating private keys for all the users within its management domain.
- We have proved that the presented ID-HABE construction is secure based on the proposed security definition.
- The performance evaluation demonstrates that the ID-HABE scheme is practical for cloud-based data sharing applications.

### B. Arrangement of the paper

The remainder of this paper is organized as follows. In section II, we introduce some preliminaries used in the following sections. In section III, we show the system model and the formal definition of ID-HABE together with its security definition. In section IV, we discuss a trivial but insecure construction as well as a basic construction of the simplified ID-HABE scheme. In section V, on the basis of the construction in section IV, we present constructions of the ID-HABE scheme. In section VI, delegable ID-HABE constructions are described. In section VII, we evaluate the efficiency of the proposed schemes. Section VIII discusses the related work. Finally, section IX concludes the paper.

## II. Preliminaries and Assumptions

In this section, we present the definition of *access structure*, *linear secret sharing schemes*, *bilinear map*, as well as the *M-q-parallel-BDHE* assumption used in the following sections.

### A. Access Structure

**Access Structure** [10]. Let $\{P_1, P_2, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$ then $C \in \mathbb{A}$. An access structure is a collection $\mathbb{A}$ of non-empty subsets of $\{P_1, P_2, \ldots, P_n\}$, *i.e.*, $\mathbb{A} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}} \backslash \{\emptyset\}$. The sets in $\mathbb{A}$ are defined as authorized sets, and sets that do not belong to $\mathbb{A}$ are defined as unauthorized sets.

### B. Linear Secret Sharing Schemes

**Linear Secret Sharing Schemes (LSSS)** [10]. A secret sharing scheme $\Pi$ over a set of parties is called linear over $\mathbb{Z}_p$ if the following two conditions are satisfied:

- the shares for each party form a vector over $\mathbb{Z}_p$;
- a share-generating matrix for $\Pi$ has $\ell$ rows and $n$ columns. For all $i = 1, \ldots, \ell$, the $i^{th}$ row of $M$, we define $\rho(i)$ as the party labeling row $i$. For the column vector $v = (s, r_2, r_3, \ldots, r_n)$, where $s \in \mathbb{Z}_p$ is the shared secret and $r_2, r_3, \ldots, r_n \in \mathbb{Z}$ are randomly chosen numbers, then $Mv$ is the vector of $\ell$ shares of the secret $s$ according to $\Pi$, where the share $(Mv)_i$ belongs to party $\rho(i)$.

As shown in [10], every linear secret sharing-scheme according to the above definition also enjoys the following linear reconstruction property.

**Definition 1** (Linear reconstruction)**.** *Assume that $\Pi$ is an LSSS for the access structure $\mathbb{A}$. Define $\mathbf{S} \in \mathbb{A}$ as an authorized set and $\mathbf{I} \subset [1, l]$ as $\mathbf{I} = \{i : \rho(i) \in \mathbf{S}\}$. Then, constants $\{w_i \in \mathbb{Z}_p\}_{i \in \mathbf{I}}$ can be derived in polynomial time such that such that for valid shares $\{\lambda_i\}$ of any secret $s$ we have $\sum_{i \in \mathbf{I}} w_i \lambda_i = s$.* □

### C. Bilinear Map

**Definition 2** (Bilinear Map)**.** *Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be multiplicative cyclic groups of prime order $p$. Let $g_1$ and $g_2$ be generator of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively. A bilinear map is a map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ with the following properties:*

- *Computable: there exists an efficiently computable algorithm for computing $e$;*
- *Bilinear: for all $u \in \mathbb{G}_1$, $v \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_p$, $e(u^a, v^b) = e(u, v)^{ab}$; For any $u \in \mathbb{G}_1, v_1, v_2 \in \mathbb{G}_2, e(u, v_1 v_2) = e(u, v_1) \cdot e(u, v_2)$;*
- *Non-degenerate: $e(g_1, g_2) \neq 1$;*

*The bilinear map is called symmetric, if $\mathbb{G}_1$ and $\mathbb{G}_2$ are a same group denoted by $\mathbb{G}$. Let $g$ denote the generator of $\mathbb{G}$.*□

### D. Security Assumption

Modified (decisional) $q$ parallel Bilinear Diffie-Hellman Exponent problem is similar to the Decisional Parallel Bilinear Diffie-Hellman Exponent ($q$-parallel BDHE ) problem [6]. The definition is as follows.

**Definition 3** (M-$q$-parallel-BDHE)**.** *Choose a group $\mathbb{G}$ of prime order $p$ according to the security parameter and a random generator $g$ of $\mathbb{G}$. Choose $a, s, b_1, b_2, \cdots, b_q \in \mathbb{Z}_p$ at random. Given*

$$\begin{aligned}\mathbf{y} =&\{g, g^s, g^a, \cdots, g^{(a^q)}, , g^{(a^{q+2})}, \cdots, g^{(a^{2q})},\\ &\forall_{1\leq j\leq q}\ g^{a/b_j}, \cdots, g^{a^q/b_j}, , g^{a^{q+2}/b_j}, ..., g^{a^{2q}/b_j},\\ &\forall_{1\leq j\leq q}\ g^{a\cdot s/b_j}, \cdots, g^{(a^q\cdot s/b_j)}\},\end{aligned}$$

*it is hard for a Probabilistic Polynomial Time (PPT) adversary to distinguish $e(g,g)^{a^{q+1}s} \in \mathbb{G}_T$ from a random element $R$ chosen from $\mathbb{G}_T$. An algorithm $\mathcal{B}$ that outputs $z \in \{0, 1\}$ has advantage $\epsilon$ in solving the M-$q$-parallel-BDHE problem defined as above if the follwing equation holds*

$$|Pr[\mathcal{B}(\mathbf{y}, T = e(g,g)^{a^{q+1}s}) = 0] - Pr[\mathcal{B}(\mathbf{y}, T = R) = 0]| \geq \epsilon.$$

□

The **M-$q$-parallel-BDHE assumption** holds if the advantage $\epsilon$ of any PPT adversary $\mathcal{B}$ to solve the **M-$q$-parallel-BDHE** problem is a negligible function of the security parameter.

**Theorem 1.** *The Modified (decisional) $q$ parallel Bilinear Diffie-Hellman Exponent assumption generically holds.*

## III. System Model and Definitions

In this section, we first show the system model and then present the definition of the proposed ID-HABE scheme as well as its security model.

### A. System Model

As shown in **Fig. 1**, the system under consideration includes four types of parties: the trusted authority, several domain authorities, data owners and data users. The data owners encrypt their data with a certain access policy. Data users whose attributes satisfy the access policy could decrypt the ciphertext. Each data owner or data user is managed by a certain domain authority. Each domain authority is managed by its parent domain authority or the trusted authority. The trusted authority is the root authority and is responsible for managing top-level domain authorities.

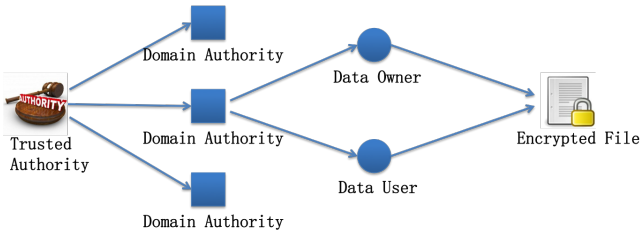


**Fig. 1** System Model.

### B. Algorithm Definition

The ID-HABE scheme consists of the following four algorithms:

- $\mathbf{Setup}(\lambda, \mathbf{U})$. The setup algorithm takes security parameter $\lambda$ and attribute universe $\mathbf{U}$ as inputs. It outputs the public parameters *PK* and a master secret key *MSK*.
- $\mathbf{KeyGen}(MSK, \mathbf{S}, ID)$. The key generation algorithm takes master secret key *MSK*, a set of attributes $\mathbf{S}$ that describe the private key, and an identity *ID* as inputs. It outputs the private key *SK*.
- $\mathbf{Encrypt}(PK, (M, \rho), \mathcal{M}, \mathbf{ID})$. The encryption algorithm takes the public parameters *PK*, the LSSS matrix *M* and its corresponding mapping $\rho$ to each attribute, the message $\mathcal{M}$ and the revoked identity set $\mathbf{ID}$. It outputs the ciphertext *CT*.
- $\mathbf{Decrypt}(CT, SK)$. The decryption algorithm takes the ciphertext *CT* and the private key *SK* as inputs and outputs the message $\mathcal{M}$ if the attributes of the secret key holder satisfy the access policy on the ciphertext *CT*.

If we enable private key delegation, there would be two types of key generation algorithms. The first one generates private key for the domain authority and the second generates private key for the users.

### C. Security Model

In our system, the root trusted authority could be fully trusted by all the users and domain authorities. The users might collude together in order to obtain access privilege which they do not have separately. In addition, we need to consider stronger adversaries whose attributes satisfy the attribute access policy of the challenge ciphertext but whose identity is in the revoked identity set. The ID-HABE security model is formalized by the game between a challenger and an adversary $\mathcal{A}$ below.

- **Init**: The adversary $\mathcal{A}$ commits to the challenge access structure $\mathbb{A}^*$ and the revoked identity set $\boldsymbol{ID}^*$ and send this to the challenger.
- **Setup**: The challenger runs the setup algorithm. The master secret key *MSK* is kept secret and the public parameters *PK* are given to the adversary $\mathcal{A}$.
- **Phase1**: The adversary $\mathcal{A}$ makes repeated private key queries $(\mathbf{S_i}, ID_i)_{i\in[1,q_1]}$ where if $\mathbf{S}_i$ satisfies $\mathbb{A}^*$ then the identity $ID_i = ID^*$.
- **Challenge**: The adversary submits two equal length messages $\mathcal{M}_0$ and $\mathcal{M}_1$. In addition, the adversary gives a

challenge LSSS access structure $\mathbb{A}^* = (M^*, \rho^*)$ and a set $\boldsymbol{ID}^*$ of revoked identities such that $\boldsymbol{ID}^*$ must include all identities that were queried. The challenger picks up a random coin $b$, and encrypts $\mathcal{M}_b$ under the access structure $\mathbb{A}^*$ and the revoked identity set $\boldsymbol{ID}^*$. Then the challenge ciphertext $\boldsymbol{CT}^*$ is sent to $\mathcal{A}$.

- **Phase2**: Repeat **Phase1** with the restriction that the queried sets of $(\mathbf{S_i}, ID_i)_{i\in[q_1+1,q]}$ where if $\mathbf{S}_i$ satisfies $\mathbb{A}^*$ then the identity $ID_i = ID^*$.
- **Guess**: The adversary outputs a guess bit $b'$ of $b$. Define $\mathrm{Adv}_{\mathcal{A}} = |\Pr[b' = b] - \frac{1}{2}|$ as the advantage of the adversary $\mathcal{A}$ winning the game.

**Definition 4** (ID-HABE Security)**.** *A ID-HABE scheme is secure if the advantage of any probabilistic polynomial time adversary $\mathcal{A}$ winning the above game is at most a negligible function of the security parameter.*

If the private keys of all the users are generated by the root trusted authority, then the security model above is complete. Whereas, if private key generation capability is delegated by the root trusted authority to some domain authorities, we need to consider what if the secret information used to generate the private keys are leaked. In particular, we need to ensure that if a domain authority is attacked successfully, all the private keys generated by this domain authority cannot be used to decrypt the ciphertext generated after this time point any more. We will discuss this in section VI.

## IV. Basic Constructions

The simplest case of ID-HABE system model is that there is only one authority, *i.e.*, the trusted authority. The private keys of all the users are generated by the trusted authority directly. There is no any domain authorities. Thus, ID-HABE only needs to support revocation of particular users rather than revocation of multiple users in a batch. We start from this simplest case at first, then step further to construct ID-HABE schemes with multiple domain authorities, and finally construct ID-HABE supporting private key generation delegation. The notations used in our constructions are presented below.

**TABLE I** Notations.

| | |
|---|---|
| $\mathbf{U}$ | the attribute set defined in the system, $\lvert\mathbf{U}\rvert = m$ |
| $p$ | the prime order of the multiplicative cyclic group $\mathbb{G}$ |
| $m$ | the number of attributes defined in the system |
| $\mathbb{Z}_p$ | $\mathbb{Z}_p = \{0, 1, \cdots, p-1\}$ |
| $[1, n]$ | $[1, n]$ denotes a set of integers *i.e.*, $\{1, 2, \cdots, n\}$ |
| $M_i$ | the $x^{th}$ row of matrix $M$ |
| $l$ | row number in matrix $M$ of an LSSS access structure $(M, \rho)$ |
| $H$ | the number of layers in the identity structure tree |
| $r_g$ | the number of revoked domain authorities |
| $r_u$ | the number of revoked users |
| $\mathbf{S}$ | the set of attributes created for a specific user |
| $ANC_i$ | the set $i's$ ancestor nodes on the path from root to $i$ |
| $\mathcal{I}_a$ | the set of all the domain authority identities |
| $\mathcal{I}_{nr}$ | the set of domain authority identities |

### A. Trivial Construction

Since CP-ABE schemes and identity-based revocation schemes have been proposed, a straight-forward two-step approach to constructing an identity revocable CP-ABE scheme can be described as follows

- **Step 1**: Enforcing CP-ABE-based access control by applying the CP-ABE scheme [6], [11].
- **Step 2**: Enforcing the identity-based revocation scheme [12] over the CP-ABE ciphertexts generated in **Step 1**.

The inner layer of CP-ABE insures the access policy enforcement while the outlier layer of identity-based revocation provides the functionality of identity-based user revocation. Therefore, this construction is a qualified one from the perspective of functionality. However, this constructions suffers from collusion attacks. For example, a data owner $C$ encrypts a file under access structure $\mathbb{A}$ with revoked identity set including user $B$ whose attributes satisfy $\mathbb{A}$. There is another user $A$ whose identity is not included in the revoked identity set but whose attributes do not satisfy the access structure. $A$ and $B$ could collude to decrypt the ciphertext even though they do not have the access privilege separately. The process is as follows. First, since $A$ is not revoked, so he/she could obtain the CP-ABE ciphertext. Second, $B$ has the attributes satisfying the access structure, so the plaintext can be obtained. Therefore, the above trivial construction is not secure.

### B. One-ID-One-Authority ID-HABE

The reason that trivial construction above does not work is that the identity and attributes of a user are separated, thus making it possible to combine identities and attributes of different users together. To this end, a feasible approach is to embed the identity and attributes together into each user's private key. Based on this idea, in this section, we show a scheme OO-ID-HABE for the basic case: one authority, *i.e.*, the trusted authority and one identity revocation. We prove that this construction is secure in terms of **Definition 1**. The four algorithms are presented as follows.

$\mathbf{Setup}(\lambda, \mathbf{U})$: The *Setup* algorithm chooses a group $\mathbb{G}$ of prime order $p$ (decided by the security parameter $\lambda$), a generator $g$ , and $m$ random group elements $h_1,\ h_2, \cdots, h_m$ that are associated with the $m$ attributes in the system. In addition, it chooses random exponents $\alpha, b \in \mathbb{Z}_p$.

The public key is published as

$$PK = \big(g, g^b, g^{b^2}, e(g,g)^\alpha, \{h_x^b\}_{x\in\mathbf{U}}\big).$$

$MSK = \{\alpha, b\}$ is the master secret key.

$\mathbf{KeyGen}(MSK, \mathbf{S}, ID)$: The *KeyGen* algorithm chooses a random $t \in \mathbb{Z}_p$ and generates the private key *SK* for user *ID*

$$SK = (K = g^\alpha g^{b^2 t}, \{K_x = (g^{b\cdot ID} h_x)^t\}_{\forall x\in\mathbf{S}}, L = g^{-t}).$$

$\mathbf{Encrypt}(PK, (M, \rho), \mathcal{M}, \mathbf{ID})$: $M$ is an $\ell \times n$ matrix and $\mathbf{ID} = \{ID'\}$ where $ID'$ is the revoked identity. The *Encrypt* algorithm chooses a random vector $v = (s, y_2, \cdots, y_n) \in \mathbb{Z}_p^n$. These values will be used to share an encryption exponent $s$. For $k \in [1, \ell]$, it calculates $\lambda_k = v \cdot M_k$. The ciphertext of the message $\mathcal{M}$ is

$$CT = (C, C_0, \hat{C}, (M, \rho), \mathbf{ID}), \text{ where}$$

$$C = \mathcal{M}e(g,g)^{\alpha s},$$

$$C_0 = g^s,$$

$$\hat{C} = \{C_k^* = g^{b\cdot\lambda_k}, C_k' = (g^{b^2\cdot ID'} h_{\rho(k)}^b)^{\lambda_k}\}_{k\in[1,\ell]}.$$

$\mathbf{Decrypt}(CT, SK)$: *CT* is the input ciphertext with access structure $(M,\rho)$ and *SK* is a private key for an attribute set $\mathbf{S}$. Suppose that $\mathbf{S}$ satisfies the access structure and define $\mathbf{I} \subset [1,\ell]$ as $\mathbf{I} = \{i : \rho(i) \in \mathbf{S}\}$. Let $\{\omega_i \in \mathbb{Z}_p\}_{i\in\mathbf{I}}$ be a set of constants such that if $\{\lambda_i\}$ are valid shares of any secret $s$ according to $M$, then $\Sigma_{i\in\mathbf{I}}\omega_i\lambda_i = s$. If the identity *ID* of the private key holder is not equal to *ID'*, we can get the value $A$. The decryption algorithm then divides out this value from $C$ and obtains the message $\mathcal{M}$.

$$\begin{aligned}
A &= \frac{e(C_0,K)}{(\prod_{i\in\mathbf{I}}[e(K_{\rho(i)},C_i^*)\cdot e(L,C_i')]^{\omega_i})^{1/(ID-ID')}} \\
&= e(g^s, g^\alpha g^{b^2 t})/(\prod_{i\in\mathbf{I}}[e((g^{b\cdot ID}h_{\rho(i)})^t, g^{b\cdot\lambda_i}) \\
&\quad \cdot e(g^{-t}, (g^{b^2\cdot ID'} h_{\rho(i)}^b)^{\lambda_i})]^{\omega_i})^{1/(ID-ID')} \\
&= e(g^s, g^\alpha)\cdot e(g^s, g^{b^2 t})/(\prod_{i\in\mathbf{I}}[e(g^{b\cdot ID\cdot t}, g^{b\cdot\lambda_i}) \\
&\quad \cdot e(h_{\rho(i)}^t, g^{b\cdot\lambda_i})\cdot e(g^{-t}, g^{b^2\cdot ID'\cdot\lambda_i}) \\
&\quad \cdot e(g^{-t}, h_{\rho(i)}^{b\cdot\lambda_i})]^{\omega_i})^{1/(ID-ID')} \\
&= e(g,g)^{\alpha s}\cdot e(g,g)^{b^2 st} \\
&\quad \cdot 1/(\prod_{i\in\mathbf{I}}[e(g,g)^{b^2 t\lambda_i(ID-ID')}]^{\omega_i})^{1/(ID-ID')} \\
&= e(g,g)^{\alpha s}\cdot e(g,g)^{b^2 st}/(\prod_{i\in\mathbf{I}} e(g,g)^{b^2 t\lambda_i\omega_i}) \\
&= e(g,g)^{\alpha s}\cdot e(g,g)^{b^2 st}/e(g,g)^{b^2 t\sum_{i\in\mathbf{I}}\lambda_i\omega_i} \\
&= e(g,g)^{\alpha s}.
\end{aligned}$$

**Theorem 2.** *Suppose the M-q-parallel-BDHE assumption holds. Then no PPT adversary can selectively break the OO-ID-HABE scheme with a challenge access structure* $(\mathbf{M}^*, \rho^*)$*, where the size of* $M^*$ *is* $\ell^* \times n^*$ *and* $\ell^*, n^* \le q$*.*

## V. ID-HABE Constructions

In this section, we will extend the basic OO-ID-HABE scheme to construct schemes with more complicated functionalities. In particular, in the first subsection we construct a scheme supporting both one particular user revocation and one domain authority revocation, so that all the users managed by the revoked domain authority can be revoked all at once. Then we extend this scheme to support revocation of multiple users and multiple domain authorities in the next subsection.

### A. One-ID Revocation in ID-HABE

Let us assume that the height of the identity structure tree is known in advance, which is set to be H. The trusted authority is on the $0^{th}$ layer.

$\mathbf{Setup}(\lambda, \mathbf{U})$: The *Setup* algorithm chooses a group $\mathbb{G}$ of prime order $p$, a generator $g$, and random group elements $\{h_x^b\}_{x\in\mathbf{U}}$ that are associated with the $m$ attributes. It also chooses random exponents $\alpha, b \in \mathbb{Z}_p$.

The public key is published as

$$PK = \big(g, g^b, g^{b^2}, e(g,g)^\alpha, \{h_x^b\}_{x\in\mathbf{U}}\big).$$

$MSK = \{\alpha, b\}$ is the master secret key.

$\mathbf{KeyGen}(MSK, \mathbf{S}, ID)$: The *KeyGen* algorithm chooses a random $t \in \mathbb{Z}_p$. The generated private key is as follows

$$SK = (K = g^\alpha g^{b^2 t}, K_{xa}, K_{xu}, L = g^{-t}), \text{ where}$$

$$K_{xa} = \{K_{xia} = (g^{b\cdot ID_i} h_x^i)^t\}_{\forall x\in\mathbf{S}, i\in[1,H]},$$

$$K_{xu} = \{(g^{b\cdot ID_u} h_x^{H+1})^t\}_{\forall x\in\mathbf{S}}.$$

$\mathbf{Encrypt1}(PK, (M,\rho), \mathcal{M}, ID_u')$: This is the algorithm for revoking a particular user. $M$ is an $l \times n$ matrix. The *Encrypt* algorithm first chooses a random vector $v = (s, y_2, \cdots, y_n) \in \mathbb{Z}_p^n$. These values will be used to share an encryption exponent $s$. Then extract $ID_a' = ID_H$ from $ID_u'$. For $k \in [1,l]$, it calculates $\lambda_k = v \cdot M_k$. Then, for message $\mathcal{M}$, the ciphertext is

$$CT = (C, C_0, \hat{C}_a, \hat{C}_u, (M,\rho), ID_u), \text{ where}$$

$$C = \mathcal{M}e(g,g)^{\alpha s},$$

$$C_0 = g^s,$$

$$\hat{C}_a = \{C_{ka}^* = g^{b\cdot\lambda_k}, C_{ka}' = (g^{b^2\cdot ID_a'} h_{\rho(k)}^{(H)\cdot b})^{\lambda_k}\}_{k\in[1,l]},$$

$$\hat{C}_u = \{C_{ku}^* = g^{b\cdot\lambda_k}, C_{ku}' = (g^{b^2\cdot ID_u'} h_{\rho(k)}^{(H+1)\cdot b})^{\lambda_k}\}_{k\in[1,l]},$$

$\mathbf{Decrypt1}(CT, SK)$: *CT* is the input ciphertext with access structure $(M,\rho)$ and *SK* is a private key for a set $\mathbf{S}$. Suppose that $\mathbf{S}$ satisfies the access structure and let $\mathbf{I} \subset [1,l]$ be defined as $\mathbf{I} = \{i : \rho(i) \in \mathbf{S}\}$. Let $\{\omega_i \in \mathbb{Z}_p\}_{i\in\mathbf{I}}$ be a set of constants such that if $\{\lambda_i\}$ are valid shares of any secret $s$ according to $M$, then $\Sigma_{i\in\mathbf{I}}\omega_i\lambda_i = s$. If the condition $ID_H \neq ID_a' \vee ID_u \neq ID_u'$ holds, we calculate $A_a$ or $A_u$

$$\begin{aligned}
A_a &= \prod_{i\in\mathbf{I}}[e(K_{\rho(i)Ha}, C_{ia}^*)\cdot e(L, C_{ia}')]^{\frac{\omega_i}{ID_H - ID_a'}} \\
&= e(g,g)^{b^2 ts},
\end{aligned}$$

$$\begin{aligned}
A_u &= \prod_{i\in\mathbf{I}}[e(K_{\rho(i)u}, C_{iu}^*)\cdot e(L, C_{iu}')]^{\frac{\omega_i}{ID_u - ID_u'}} \\
&= e(g,g)^{b^2 ts}.
\end{aligned}$$

We can get the value $e(g,g)^{\alpha s}$ by evaluating $\frac{e(C_0,K)}{A_a}$ or $\frac{e(C_0,K)}{A_u}$. The decryption algorithm then divides out this value from $C$ and obtains the message $\mathcal{M}$.

$\mathbf{Encrypt2}(PK, (M,\rho), \mathcal{M}, ID_a')$: This is the algorithm for revoking a particular authority. $M$ is an $l \times n$ matrix. The *Encrypt* algorithm first chooses a random vector $v = (s, y_2, \cdots, y_n) \in \mathbb{Z}_p^n$. These values will be used to share an encryption exponent $s$. For $k \in [1,l]$, it calculates $\lambda_k = v\cdot M_k$. Then, for message $\mathcal{M}$, the ciphertext is

$$CT = (C, C_0, \hat{C}_a, (M,\rho), ID_a'),$$

where

$$C = \mathcal{M}e(g,g)^{\alpha s},$$

$$C_0 = g^s,$$

$$\hat{C}_a = \{C_{ku}^* = g^{b\cdot\lambda_k}, C_{ku}' = (g^{b^2\cdot ID_a'} h_{\rho(k)}^{i\cdot b})^{\lambda_k}\}_{k\in[1,l]}.$$

$\mathbf{Decrypt2}(CT, SK)$: *CT* is the input ciphertext with access structure $(M,\rho)$ and *SK* is a private key for a set $\mathbf{S}$. Extract

the identity $ID_a$ which is on the same layer as $ID'_a$. Suppose that $\mathbf{S}$ satisfies the access structure and let $\mathbf{I} \subset [1, l]$ be defined as $\mathbf{I} = \{i : \rho(i) \in \mathbf{S}\}$. Let $\{\omega_i \in \mathbb{Z}_p\}_{i\in\mathbf{I}}$ be a set of constants such that if $\{\lambda_i\}$ are valid shares of any secret $s$ according to $M$, then $\Sigma_{i\in\mathbf{I}}\omega_i\lambda_i = s$. If the condition $ID_a \neq ID'_a$ holds, we calculate $A_a$ as in *Decrypt1*.

**Theorem 3.** *Suppose the M-q-parallel-BDHE assumption holds. Then no PPT adversary can selectively break the OM-ID-HABE scheme with a challenge access structure* $(\mathbf{M}^*, \rho^*)$, *where the size of* $M^*$ *is* $\ell^* \times n^*$ *and* $\ell^*, n^* \leq q$.

### *B. Multiple-ID Revocation in ID-HABE*

In this section, we show how to revoke multiple users and multiple domain authorities. For description simplicity, here we set $H = 1$. It is easy to extend this construction to $H > 1$.

$\mathbf{Setup}(\lambda, \mathbf{U})$: The *Setup* algorithm chooses a group $\mathbb{G}$ of prime order $p$, a generator $p$ and $m$ random group elements $\{h^b_x\}_{x\in\mathbf{U}}, \{h^b_{xi}\}_{x\in\mathbf{U},i\in[1,H]}$ that are associated with the $m$ attributes in the system. It also chooses random exponents $\alpha, b \in \mathbb{Z}_p$.

The public parameters are:

$$PK = \big(g, g^b, g^{b^2}, e(g,g)^\alpha, \{h^b_x\}_{x\in\mathbf{U}}, \{h^b_{xi}\}_{x\in\mathbf{U},i\in[1,H]}\big).$$

The master secret key is $MSK = \{\alpha, b\}$.

$\mathbf{KeyGen}(MSK, \mathbf{S}, ID)$: The generated private key is as follows

$$SK = (K = g^\alpha g^{b^2 t}, K_{xa}, K_{xu}, L = g^{-t}), \text{ where}$$
$$K_{xa} = \{K_{xia} = (g^{b\cdot ID_i} h_{xi})^t\}_{\forall x\in\mathbf{S},i\in[1,H]},$$
$$K_{xu} = \{(g^{b\cdot ID_u} h_x)^t\}_{\forall x\in\mathbf{S}}.$$

$\mathbf{Encrypt}(PK, (M, \rho), \mathcal{M}, \mathbf{ID})$: Denote the set of revoked domain authorities by $\mathbf{ID}_a = \{(ID'_{h_j})\}_{j\in[1,r_a]}$. The set of revoked user identities is denoted by $\mathbf{ID}_u = \{(ID'_{u,j})\}_{j\in[1,r_u]}$. We could extract the direct domain authority$ID'_{a,j}$ of $ID'_{u,j}$. $\mathbf{ID} = \mathbf{ID}_a \cup \mathbf{ID}_u$ and $|\mathbf{ID}_a| + |\mathbf{ID}_u| = r_a + r_u = r$. Let $M$ be an $l\times n$ matrix. The algorithm first chooses a random vector $v = (s, y_2, \cdots, y_n) \in \mathbb{Z}^n_p$. These values will be used to share the encryption exponent $s$. For $x \in [1, l]$, it calculates $\lambda_x = v \cdot M_x$. The algorithm chooses random $\mu_1, ..., \mu_{r_a}, \mu'_1, ..., \mu'_{r_u} \in \mathbb{Z}_p$ such that $\mu_a = \mu_1 + ... + \mu_{r_a}$, $\mu_u = \mu'_1 + ... + \mu'_{r_u}$. It generates the ciphertext

$$CT = (C, C_{a0}, C_{u0}, \hat{C}_a, \hat{C}_{au}, \hat{C}_u),$$

where

$$C = \mathcal{M}e(g,g)^{\alpha s\mu}, \text{ where } \mu = \mu_a + \mu_u$$
$$C_{a0} = \{C_a = g^{s\mu_a}, C_{a,j} = g^{s\mu_j}\}_{j\in[1,r_a]}$$
$$C_{u0} = \{C_{u,j} = g^{s\mu'_j}\}_{j\in[1,r_u]}$$
$$\hat{C}_a = \{C^*_{ka,j} = g^{b\cdot\lambda_k\mu_j}, C'_{ka,j} = (g^{b^2\cdot ID'_{h_j}} h_{\rho(k)h_j})^{\lambda_k\mu_j}\}^{j\in[1,r_a]}_{k\in[1,l]},$$
$$\hat{C}_{au} = \{C^*_{kau,j} = g^{b\cdot\lambda_k\mu'_j}, C'_{kau,j} = (g^{b^2\cdot ID'_{a,j}} h_{\rho(k)H})^{\lambda_k\mu'_j}\}^{j\in[1,r_u]}_{k\in[1,l]},$$
$$\hat{C}_u = \{C^*_{ku,j} = g^{b\cdot\lambda_k\mu'_j}, C'_{ku,j} = (g^{b^2\cdot ID'_{u,j}} h_{\rho(k)})^{\lambda_k\mu'_j}\}^{j\in[1,r_u]}_{k\in[1,l]}.$$

$\mathbf{Decrypt}(CT, SK)$: If neither the domain authority of *SK* holder or the *SK* holder him/herself is revoked, then we first calculate $A$ as follows

$$\begin{aligned} A &= \prod_{i\in\mathbf{I}}\prod_{j=1}^{r_a}[e(K_{\rho(i)h_j a}, C^*_{ia,j}) \cdot e(L, C'_{ia,j})]^{\frac{\omega_i}{ID_{h_j} - ID'_{h_j}}} \\ &= e(g,g)^{b^2 st\mu_a}. \end{aligned}$$

$A_a = \frac{e(C_a, K)}{A} = e(g,g)^{\alpha s\mu_a}$. For particular user revocation part. $\mathbf{R}_{u1} \subset \mathbf{R}_u$, where $\mathbf{R}_{u1} = \{(ID'_{a,j}, ID'_{u,j}))|ID'_{a,j} \neq ID_H\}$. $\mathbf{R}_{u2} \subset \mathbf{R}_u$ where $\mathbf{R}_{u2} = \{(ID'_{a,j}, ID'_{u,j}))|ID'_{a,j} = ID_H \wedge ID'_{u,j} \neq ID_u\}$. We calculate $A_{u1}$ and $A_{u2}$ as follows

$$\begin{aligned} A_{u1} &= \prod_{i\in\mathbf{I}}\prod_{j\in\mathbf{R}_{u1}}[e(K_{\rho(i)Ha}, C^*_{iau,j})e(L, C'_{iau,j})]^{\frac{\omega_i}{ID_H - ID'_{a,j}}} \\ &= e(g,g)^{b^2 st\sum_{j\in\mathbf{R}_{u1}}\mu'_j}, \end{aligned}$$

$$\begin{aligned} A_{u2} &= \prod_{i\in\mathbf{I}}\prod_{j\in\mathbf{R}_{u2}}[e(K_{\rho(i)Ha}, C^*_{iu,j})e(L, C'_{iu,j})]^{\frac{\omega_i}{ID_H - ID'_{a,j}}} \\ &= e(g,g)^{b^2 st\sum_{j\in\mathbf{R}_{u2}}\mu'_j}. \end{aligned}$$

Then we have:

$$\begin{aligned} A_u &= \frac{\prod_{j\in\mathbf{R}_{u1}} e(C_{u,j}, K)}{A_{u1}} \cdot \frac{\prod_{j\in\mathbf{R}_{u2}} e(C_{u,j}, K)}{A_{u2}} \\ &= e(g,g)^{\alpha s\mu_u}. \end{aligned}$$

We could obtain the message by evaluating $\frac{C}{e(g,g)^{\alpha s\mu}}$, where $e(g,g,)^{\alpha s\mu} = A_g \cdot A_u$.

**Theorem 4.** *Suppose the M-q-parallel-BDHE assumption holds. Then no PPT adversary can selectively break the MM-ID-HABE scheme with a challenge access structure* $(\mathbf{M}^*, \rho^*)$, *where the size of* $M^*$ *is* $\ell^* \times n^*$ *and* $\ell^*, n^* \leq q$.

## VI. Delegable ID-HABE Constructions

The constructions above can support revocation for each individual user as well as domain authorities (i.e., delegators); however, the trusted authority has to generate the private key for each user, which incurs the following drawbacks. First, the trusted authority has to generate the private key for each user. Second, the trusted authority must verify proofs of each user's attributes and also must establish secure channels to transmit the private keys. Therefore, the trusted authority becomes a bottleneck in the system. To this end, in this section, we propose delegable ID-HABE schemes,which allows the root trusted authority to delegate private key generation and attributes checking to some domain authorities. Similar to section V, we first present a construction supporting one user and one domain authority revocation and then extend this construction to support multiple users and multiple authorities.

### *A. One-ID-Multi-Authority Delegable ID-HABE*

In this section, we present the construction of one identity revocable delegable ID-HABE scheme, OMD-ID-HABE. For clarity, we divide the encryption and decryption algorithm into two parts. The first one corresponds to one user revocation, while the second one corresponds to one domain authority revocation.

$\mathbf{Setup}(\lambda, \mathbf{U})$: The *Setup* algorithm chooses a group $\mathbb{G}$ of prime order $p$, a generator $g$, and $m$ random group elements $h_1, \cdots, h_m$ that are associated with the $m$ attributes in the system. It also chooses random exponents $\alpha, b$ and $s_0 \in \mathbb{Z}_p$.

The published parameters are in the form:

$$PK = \big(g, g^b, g^{b^2}, e(g,g)^\alpha, \{h_x^b, h_x^{b^2}\}_{x\in\mathbf{U}}, \{g^{bs_i^{-1}}, g^{s_i^{-1}}\}_{i\in\mathcal{I}}\big),$$

where $s_i$ is evaluated based on the identity structure tree. $s_{\text{child}} = ID_{\text{child}}{}^{s_{\text{parent}}}$ and $s_{\text{root}} = s_0$.

The master secret key is

$$MSK = \big(\alpha, b, s_0\big).$$

$\mathbf{KeyGenforDA}(SK_{ia}, \mathbf{S}_{(i+1)a}, ID_{(i+1)a})$: This is an algorithm generating secret key for a domain authority $ID_{(i+1)a}$, which is at the $(i+1)^{th}$ layer. This algorithm is run by an $i^{th}$ level domain authority with identity $ID_{ia}$ with secret delegation key $SK_{ia}$ as follows:

$$SK_{ia} = \big(g^\alpha g^{b^2 t_{ia}}, g^{s_{jd}^{-1} t_{ia}}, g^{-t_{ia}}, (g^{bs_{ja}} h_x^b)^{t_{ia}},$$
$$g^{bs_{ja}}, h_x, g^{bt_{ia}}, h_x^{t_{ia}}, h_x^{bt_{ia}}, s_{ia}\big)_{j\in ANC_i\cup i}^{x\in\mathbf{S}_{ia}}.$$

Based on the attribute set $\mathbf{S}_{(i+1)a}$ (*s.t.*, $\mathbf{S}_{(i+1)a} \subset \mathbf{S}_{ia}$), $ID_{ia}$ sends the key $SK_{ia\to(i+1)a}$ to $ID_{(i+1)a}$.

$$SK_{ia\to(i+1)a} = \big(g^\alpha g^{b^2 t_{ia}}, g^{s_{ja}^{-1} t_{ia}}, g^{-t_{ia}}, (g^{bs_{ja}} h_x^b)^{t_{ia}},$$
$$g^{bs_{ja}}, h_x, g^{bt_{ia}}, h_x^{t_{ia}}, h_x^{bt_{ia}}, s_{(i+1)a}\big)_{j\in ANC_{(i+1)a}}^{x\in\mathbf{S}_{(i+1)a}}.$$

$ID_{(i+1)a}$ randomly selects $t' \in \mathbb{Z}_p$ and computes its own secret key as follows, where $t_{(i+1)a} = t_{ia} + t'$:

$$SK_{(i+1)a} = \big(g^\alpha g^{b^2 t_{(i+1)a}}, g^{s_{ja}^{-1} t_{(i+1)a}}, g^{-t_{(i+)a}}, (g^{bs_{ja}} h_x^b)^{t_{(i+)a}},$$
$$g^{bs_{ja}}, h_x, g^{bt_{(i+1)a}}, h_x^{t_{(i+1)a}}, h_x^{bt_{(i+1)a}}, s_{(i+1)a}\big)_{j\in ANC_{(i+1)a}\cup(i+1)}^{x\in\mathbf{S}_{(i+1)a}},$$

where some updated items are calculated as follows:

$$g^\alpha g^{b^2 t_{(i+1)a}} = g^\alpha g^{b^2 t_{ia}} \cdot (g^{b^2})^{t'},$$
$$g^{s_{ja}^{-1} t_{(i+1)a}} = g^{s_{ja}^{-1} t_{ia}} \cdot (g^{s_{ja}^{-1}})^{t'},$$
$$g^{-t_{(i+1)a}} = g^{-t_{ia}} \cdot g^{-t'},$$
$$(g^{bs_{ja}} h_x^b)^{t_{(i+1)a}} = (g^{bs_{ja}} h_x^b)^{t_{ia}} \cdot (g^{bs_{ja}} \cdot h_x^b)^{t'},$$
$$(g^{bs_{(i+1)a}} h_x^b)^{t_{(i+1)a}} = (g^{bt_{ia}} \cdot g^{bt'})^{s_{(i+1)a}} \cdot h_x^{bt_{ia}} \cdot h_x^{bt'},$$
$$g^{bs_{(i+1)a}} = (g^b)^{s_{(i+1)a}},$$
$$g^{t_{(i+1)a}} = g^{t_{ia}} \cdot g^{t'},$$
$$h_x^{t_{(i+1)a}} = h_x^{t_{ia}} \cdot h_x^{t'},$$
$$h_x^{bt_{(i+1)a}} = h_x^{bt_{ia}} \cdot (h_x^b)^{t'}.$$

$\mathbf{KeyGenforUser}(SK_a, \mathbf{S}, ID)$: $SK_a$ is the domain authority $ID_a$ that generates the secret key $SK$ for a user $ID$. The secret delegation key $SK_a$ of $ID_a$ is as follows.

$$SK_a = \big(g^\alpha g^{b^2 t_a}, g^{s_{ja}^{-1} t_a}, g^{-t_a}, (g^{bs_{ja}} h_x^b)^{t_a},$$
$$g^{bs_{ja}}, h_x, g^{bt_a}, h_x^{t_a}, h_x^{bt_a}, s_a\big)_{j\in ANC_a\cup d}^{x\in\mathbf{S}_a}.$$

The domain authority $ID_a$ chooses a random $t' \in \mathbb{Z}_p$, and distributes the following secret key $SK$ to the user $ID$:

$$SK = \big(K = g^\alpha g^{b^2 t_u}, \{L_a = g^{s_{ja}^{-1} t_u}\}_{j\in ANC_u}, L_u = g^{-t_u},$$
$$\{K'_{xa} = (g^{b\cdot s_{ja}} h_x^b)^{t_u}\}_{j\in ANC_u}^{\forall x\in\mathbf{S}}, \{K_{xu}{}' = (g^{b\cdot ID} h_x)^{t_u}\}_{\forall x\in\mathbf{S}}\big),$$

where $t_u = t_a + t'$.

$\mathbf{Encrypt1}(PK, (M,\rho), \mathcal{M}, \mathbf{ID})$: The is the encryption algorithm for revoking just one user. $\mathbf{ID} = \{(ID'_a, ID'_u)\}$. The *Encrypt* algorithm takes inputs as an LSSS access infrastructure $(M,\rho)$ and the function $\rho$ associates each row of $M$ to corresponding attributes. Let $M$ be an $l \times n$ matrix. The *Encrypt* algorithm first chooses a random vector $v = (s, y_2, \cdots, y_n) \in \mathbb{Z}_p^n$. These values will be used to share an encryption exponent $s$. For $x \in [1,l]$, it calculates $\lambda_x = v \cdot M_x$. Then, for message $\mathcal{M}$, the ciphertext is presented as follows

$$CT = (C, C_0, \hat{C}_a, \hat{C}_u, \mathbf{ID}), \text{ where}$$
$$C = \mathcal{M} e(g,g)^{\alpha s},$$
$$C_0 = g^s,$$
$$\hat{C}_a = \big(C^*_{ka} = g^{bs_i^{-1}\lambda_k}, C'_{ka} = (h^{b^2}_{\rho(k)})^{\lambda_k}\big)_{k\in[1,l]}^{i\in\mathcal{I}_{nr}},$$
$$\hat{C}_u = \big(C^*_{ku} = g^{b\cdot\lambda_k}, C'_{ku} = (g^{b^2\cdot ID'_u} h^b_{\rho(k)})^{\lambda_k}\big)_{k\in[1,l]}.$$

$\mathbf{Decrypt1}(CT, SK)$: $CT$ is the input ciphertext with access structure $(M,\rho)$ and $SK$ is a private key for a set $\mathbf{S}$. Suppose that $\mathbf{S}$ satisfies the access structure and let $\mathbf{I} \subset [1,l]$ be defined as $\mathbf{I} = \{i : \rho(i) \in \mathbf{S}\}$. Let $\{\omega_i \in \mathbb{Z}_p\}_{i\in\mathbf{I}}$ be a set of constants such that if $\{\lambda_i\}$ are valid shares of any secret $s$ according to $M$, then $\Sigma_{i\in\mathbf{I}}\omega_i\lambda_i = s$. Denote the identity of the direct domain authority administrating $ID_u$ by $ID_a$. The decryption algorithm runs as follows:

$$\begin{cases} \dfrac{e(C_0, K)}{\prod_{i\in\mathbf{I}}[e(K'_{\rho(i)a}, C^*_{ia}) \cdot e(L_a, C'_{ia})]^{\omega_i}}, & ID_a \neq ID'_a; \\ \dfrac{e(C_0, K_u)}{(\prod_{i\in\mathbf{I}}[e(K'_{\rho(i)u}, C^*_{ui}) \cdot e(L_u, C'_{ui})]^{\omega_i})^{\frac{1}{(ID_u - ID'_u)}}}, & ID_u \neq ID'_u; \\ \text{abort}, & \text{otherwise.} \end{cases} \quad (2)$$

If $SK$'s holder is not the revoked user, then by equation (2), we can get $e(g,g)^{\alpha s}$ and finally get the message $\mathcal{M}$ by evaluating $\frac{C}{e(g,g)^{\alpha s}}$.

$\mathbf{Encrypt2}(PK, (M,\rho), \mathcal{M}, \mathbf{ID})$: This is an encryption algorithm for revoking just one domain authority. The *Encrypt* algorithm takes inputs as an LSSS access infrastructure $(M,\rho)$ and the function $\rho$ associates each row of $M$ to corresponding attributes, where $M$ is an $l \times n$ matrix. $\mathbf{ID} = \{ID'_a\}$. The *Encrypt* algorithm first chooses a random vector $v = (s, y_2, \cdots, y_n) \in \mathbb{Z}_p^n$. These values will be used to share an encryption exponent $s$. For $x \in [1,l]$, it calculates $\lambda_x = v \cdot M_x$. Then, for message $\mathcal{M}$, the ciphertext is presented as follows

$$CT = (C, C_0, \hat{C}_a, \mathbf{ID}), \text{ where}$$
$$C = \mathcal{M} e(g,g)^{\alpha s},$$

$$C_0 = g^s,$$

$$\hat{C}_a = \big(C_k^* = g^{bs_i^{-1}\lambda_k}, C_k' = (h_{\rho(k)}^{b^2})^{\lambda_k}\big)_{k\in[1,l]}^{i\in\mathcal{I}_H}.$$

$\mathbf{Decrypt2}(CT, SK)$: *CT* is the ciphertext with access structure $(M,\rho)$ and *SK* is a private key for a set $\mathbf{S}$. Suppose that $\mathbf{S}$ satisfies the access structure and let $\mathbf{I}\subset[1,l]$ be defined as $\mathbf{I}=\{i:\rho(i)\in\mathbf{S}\}$. Let $\{\omega_i\in\mathbb{Z}_p\}_{i\in\mathbf{I}}$ be a set of constants such that if $\{\lambda_i\}$ are valid shares of any secret $s$ according to $M$, then $\Sigma_{i\in\mathbf{I}}\omega_i\lambda_i = s$. Denote the identity of the non-revoked domain authority administrating $ID_u$ by $ID_a$. The decryption algorithm runs as follows

$$\begin{cases} \dfrac{e(C_0,K)}{(\prod_{i\in\mathbf{I}}[e(K'_{\rho(i)a},C_i^*)\cdot e(L_a,C_i')]^{\omega_i}}, & \text{if } ID_a \neq ID'_a; \\ \text{abort}, & \text{otherwise.} \end{cases} \tag{3}$$

If *SK*'s holder is not in the revoked group, then by equation (3), we can get $e(g,g)^{\alpha s}$ and finally get the message $\mathcal{M}$ by evaluating $\frac{C}{e(g,g)^{\alpha s}}$.

**Theorem 5.** *Suppose the M-q-parallel-BDHE assumption holds. Then no PPT adversary can selectively break the OM-DID-HABE scheme with a challenge access structure* $\big(\mathrm{M}^*,\rho^*\big)$*, where the size of* $M^*$ *is* $\ell^*\times n^*$ *and* $\ell^*, n^*\le q$.

### *B. Multi-ID-Multi-Authority DID-HABE*

In this section, we present the construction of multiple identities revocable delegable ID-HABE scheme, MMD-ID-HABE. For clarity, we divide the encryption and decryption algorithm into three parts. The first one corresponds to revocation of multiple users, the second one corresponds to revocation of multiple domain authorities and the third one corresponds to revocation of multiple users together with multiple domain authorities.

$\mathbf{Setup}(\lambda,\mathbf{U})$: The *Setup* algorithm chooses a group $\mathbb{G}$ of prime order $p$, a generator $g$, and $m$ random group elements $h_1,\cdots,h_m$ that are associated with the $m$ attributes in the system. It also chooses random exponents $\alpha, b$ and $s_0\in\mathbb{Z}_p$.

The public parameters are as follows:

$$PK = \big(g, g^b, g^{b^2}, e(g,g)^\alpha, \{h_x^b, h_x^{b^2}\}_{x\in\mathbf{U}}, \{g^{bs_i^{-1}}, g^{s_i^{-1}}\}_{i\in\mathcal{I}}\big),$$

where $s_i$ is evaluated based on the structure tree of the domain authoritys. $s_{\text{child}} = ID_{\text{child}}{}^{s_{\text{parent}}}$, $s_{\text{root}} = s_0$.

The master secret key is in the form:

$$MSK = \big(\alpha, b, s_0\big).$$

$\mathbf{KeyGenforDA}(SK_{ia}, \mathbf{S}_{(i+1)a}, ID_{(i+1)a})$: This is an algorithm generating secret key for a domain authority $ID_{(i+1)a}$, which is at the $(i+1)^{th}$ layer. This algorithm is run by an $i^{th}$ level domain authority with identity $ID_{ia}$ with secret delegation key $SK_{ia}$:

$$SK_{ia} = \big(g^\alpha g^{b^2 t_{ia}}, g^{s_{jd}^{-1}t_{ia}}, g^{-t_{ia}}, (g^{bs_{ja}}h_x^b)^{t_{ia}},$$

$$g^{bs_{ja}}, h_x, g^{bt_{ia}}, h_x^{t_{ia}}, h_x^{bt_{ia}}, s_{ia}\big)_{j\in ANC_i\cup i}^{x\in\mathbf{S}_{ia}}.$$

Based on the attribute set $\mathbf{S}_{(i+1)a}$ (*s.t.*, $\mathbf{S}_{(i+1)a}\subset\mathbf{S}_{ia}$), $ID_{ia}$ sends the key $SK_{ia\to(i+1)a}$ to $ID_{(i+1)a}$:

$$SK_{ia\to(i+1)a} = \big(g^\alpha g^{b^2 t_{ia}}, g^{s_{ja}^{-1}t_{ia}}, g^{-t_{ia}}, (g^{bs_{ja}}h_x^b)^{t_{ia}},$$

$$g^{bs_{ja}}, h_x, g^{bt_{ia}}, h_x^{t_{ia}}, h_x^{bt_{ia}}, s_{(i+1)a}\big)_{j\in ANC_{(i+1)a}}^{x\in\mathbf{S}_{(i+1)a}}.$$

$ID_{(i+1)a}$ randomly selects $t'\in\mathbb{Z}_p$ and computes its own secret key as follows, where $t_{(i+1)a} = t_{ia} + t'$.

$$SK_{(i+1)a} = \big(g^\alpha g^{b^2 t_{(i+1)a}}, g^{s_{ja}^{-1}t_{(i+1)a}}, g^{-t_{(i+1)a}}, (g^{bs_{ja}}h_x^b)^{t_{(i+1)a}},$$

$$g^{bs_{ja}}, h_x, g^{bt_{(i+1)a}}, h_x^{t_{(i+1)a}}, h_x^{bt_{(i+1)a}}, s_{(i+1)a}\big)_{j\in ANC_{(i+1)a}\cup(i+1)}^{x\in\mathbf{S}_{(i+1)a}}.$$

$\mathbf{KeyGenforUser}(SK_a, \mathbf{S}, ID)$: $SK_a$ is the domain authority $ID_a$ that generates the secret key *SK* for a user *ID*. The secret delegation key $SK_a$ of $ID_a$ is as follows:

$$SK_a = \big(g^\alpha g^{b^2 t_a}, g^{s_{ja}^{-1}t_a}, g^{-t_a}, (g^{bs_{ja}}h_x^b)^{t_a},$$

$$g^{bs_{ja}}, h_x, g^{bt_a}, h_x^{t_a}, h_x^{bt_a}, s_a\big)_{j\in ANC_a\cup d}^{x\in\mathbf{S}_a}.$$

$ID_a$ chooses a random $t'\in\mathbb{Z}_p$, and distributes the secret key *SK* to user *ID*:

$$SK_u = \big(K = g^\alpha g^{b^2 t_u}, \{L_a = g^{s_{ja}^{-1}t_u}\}_{j\in ANC_u}, L_u = g^{-t_u},$$

$$\{K'_{xa} = (g^{b\cdot s_{ja}}h_x^b)^{t_u}\}_{j\in ANC_u}^{\forall x\in\mathbf{S}}, \{K_{xu}{}' = (g^{b\cdot ID}h_x)^{t_u}\}_{\forall x\in\mathbf{S}}\big),$$

where $t_u = t_d + t'$.

$\mathbf{Encrypt1}(PK, (M,\rho), \mathcal{M}, \mathbf{ID})$: This algorithm is for revoking multiple users. The *Encrypt1* algorithm takes inputs as an LSSS access infrastructure $(M,\rho)$ where $M$ is an $l\times n$ matrix and the function $\rho$ associates each row of $M$ to corresponding attributes. $\mathbf{ID} = \{\big(ID'_{a,j}, ID'_{u,j}, h_j\big)\}_{j\in[1,r_u]}$. The *Encrypt* algorithm first chooses a random vector $v = (s, y_2, \cdots, y_n)\in\mathbb{Z}_p^n$. These values will be used to share an encryption exponent $s$. For $x\in[1,l]$, it calculates $\lambda_x = v\cdot M_x$. The *Encrypt* algorithm chooses random $\mu_1, ..., \mu_{r_u}\in\mathbb{Z}_p$, $\mu = \mu_1 + ... + \mu_{r_u}$. Then, the ciphertext of message $\mathcal{M}$ is

$$CT = (C, C_0, \hat{C}_a, \hat{C}_u, \mathbf{ID}), \text{ where}$$

$$C = \mathcal{M}e(g,g)^{\alpha s\mu},$$

$$C_0 = g^{s\mu},$$

$$\hat{C}_a = \{C_{akj}^* = g^{bs_i^{-1}\lambda_k\mu_j}, C'_{ak} = (h_{\rho(k)}^b)^{\lambda_k\mu_j}\}_{k\in[1,l],j\in[1,r_u]}^{i\in\mathcal{I}_{nr}},$$

$$\hat{C}_u = \Big(\{C_{ukj}^* = g^{b\lambda_k\mu_j}\}_{k\in[1,l],j\in[1,r_u]},$$

$$\{C'_{ukj} = (g^{b^2\cdot ID'_{u,j}}h_{\rho(k)}^b)^{\lambda_k\mu_j}\}_{k\in[1,l],j\in[1,r_u]}\Big).$$

$\mathbf{Decrypt1}(CT, SK)$: *CT* is the input ciphertext with access structure $(M,\rho)$ and *SK* is a private key for a set $\mathbf{S}$. Suppose that $\mathbf{S}$ satisfies the access structure and let $\mathbf{I}\subset[1,l]$ be defined as $\mathbf{I}=\{i:\rho(i)\in\mathbf{S}\}$. Let $\{\omega_i\in\mathbb{Z}_p\}_{i\in\mathbf{I}}$ be a set of constants such that if $\{\lambda_i\}$ are valid shares of any secret $s$ according to $M$, then $\Sigma_{i\in\mathbf{I}}\omega_i\lambda_i = s$. For the $j^{th}$ revoked identity, denote the identity of the non-revoked domain authority managing *ID*

by $ID_{a,j}$. The decryption algorithm calculates $e(g,g)^{b^2 t s u_j}$ as follows:

$$\begin{cases} \prod_{i\in\mathbf{I}}[e(K'_{\rho(i)aj}, C^*_{aij})\cdot e(L_{aj}, C'_{aij})]^{\omega_i}, ID_{a,j} \neq ID'_{a,j} \\ (\prod_{i\in\mathbf{I}}[e(K_{\rho(i)u}, C^*_{uij})\cdot e(L_u, C'_{uij})]^{\omega_i})^{\frac{1}{(ID_u - ID'_{u,j})}}, ID_u \neq ID'_{u,j}. \end{cases} \quad (4)$$

If *SK*'s holder is not revoked, then we can get $e(g,g)^{\alpha s\mu}$ by $\frac{e(C_0,K)}{\prod_{j\in[1,r_u]} e(g,g)^{b^2 t \lambda_k \mu_j}}$. Finally get the message $\mathcal{M}$ by evaluating $\frac{C}{e(g,g)^{\alpha s\mu}}$.

$\mathbf{Encrypt2}(PK, (M,\rho), \mathcal{M}, \mathbf{ID})$: This algorithm is used to revoke multiple domain authorities. It takes as inputs an LSSS access infrastructure $(M,\rho)$, where $M$ is an $l\times n$ matrix and the function $\rho$ associates each row of $M$ to corresponding attributes. $\mathbf{ID} = \{ID_{a,j}\}_{j\in[1,r_g]}$. The *Encrypt2* algorithm first chooses a random vector $v = (s, y_2, \cdots, y_n) \in \mathbb{Z}_p^n$. These values will be used to share an encryption exponent $s$. For $x \in [1,l]$, it calculates $\lambda_x = v\cdot M_x$ and chooses random $s \in \mathbb{Z}_p$. Then, the ciphertext of the message $\mathcal{M}$ is as follows

$$CT = (C, C_0, \hat{C}_a, \mathbf{ID}), \text{ where}$$
$$C = \mathcal{M} e(g,g)^{\alpha s},$$
$$C_0 = g^s,$$
$$\hat{C}_a = \left(C^*_{ak} = g^{b s_i^{-1} \lambda_k}, C'_{ak} = (h^b_{\rho(k)})^{\lambda_k}\right)^{i\in\mathcal{I}_{nr}}_{k\in[1,l]}.$$

$\mathbf{Decrypt2}(CT, SK)$: *CT* is the ciphertext with access structure $(M,\rho)$ and *SK* is a private key for a set $\mathbf{S}$. Suppose that $\mathbf{S}$ satisfies the access structure and let $\mathbf{I}\subset[1,l]$ be defined as $\mathbf{I} = \{i : \rho(i)\in\mathbf{S}\}$. Let $\{\omega_i \in \mathbb{Z}_p\}_{i\in\mathbf{I}}$ be a set of constants such that if $\{\lambda_i\}$ are valid shares of any secret $s$ according to $M$, then $\Sigma_{i\in\mathbf{I}}\omega_i\lambda_i = s$. Denote the identity of a non-revoked domain authority managing $ID_u$ by $ID_a$. The decryption algorithm evaluates $A_a$ as follows:

$$\begin{aligned} A_a &= \frac{e(C_0,K)}{(\prod_{i\in\mathbf{I}}[e(K'_{d\rho(i)}, C^*_{di})\cdot e(L_d, C'_{di})]^{\omega_i}} \\ &= e(g,g)^{\alpha s}. \end{aligned}$$

If *SK*'s holder is not administered by revoked domain authorities, then we can get $e(g,g)^{\alpha s}$. Finally the decryption algorithm divides out this value from $C$ and obtains the message $\mathcal{M}$.

$\mathbf{Encrypt3}(PK, (M,\rho), \mathcal{M}, \mathbf{ID})$: This is an algorithm revoking both multiple users and multiple domain authorities. The *Encrypt3* algorithm takes as inputs an LSSS access infrastructure $(M,\rho)$, where $M$ is an $l\times n$ matrix and the function $\rho$ associates each row of $M$ to corresponding attributes. $\mathbf{ID} = \mathbf{ID}_a \cup \mathbf{ID}_u$ and $|\mathbf{ID}_a| + |\mathbf{ID}_u| = r_a + r_u = r$. Denote the set of revoked domain authority identities as $\mathbf{ID}_a = \{(ID'_{a,j}, h_{a,j})\}_{j\in[1,r_a]}$. The set of revoked user identities is denoted by $\mathbf{ID}_u = \{(ID'_{a,j}, ID'_{u,j}, h_{u,j})\}_{j\in[1,r_u]}$, where $ID'_{u,j}$ is managed by domain authority $ID'_{a,j}$. The *Encrypt* algorithm first chooses a random vector $v = (s, y_2, \cdots, y_n) \in \mathbb{Z}_p^n$. These values will be used to share an encryption exponent $s$. For $x\in[1,l]$, it calculates $\lambda_x = v\cdot M_x$. The *Encrypt3* algorithm chooses random $s\in\mathbb{Z}_p$. The algorithm chooses random $\mu_1, \mu_2$ such that $\mu = \mu_a + \mu_u$, and $\mu_1, \cdots, \mu_{r_a}, \mu'_1, \cdots, \mu'_{r_u} \in \mathbb{Z}_p$ such that $\mu_a = \mu_1 + \cdots + \mu_{r_a}$ and $\mu_u = \mu'_1 + \cdots + \mu'_{r_u}$. Then, for message $\mathcal{M}$, the ciphertext is presented as follows:

$$CT = (C, C_0, \hat{C}_a, \hat{C}_u, \hat{C_a}', \mathbf{ID}), \text{ where}$$
$$C = \mathcal{M} e(g,g)^{\alpha s\mu},$$
$$C_0 = g^{s\mu},$$
$$\hat{C}_a = \left(C^*_{akj} = g^{b s_i^{-1}\lambda_k u'_j}, C'_{ak} = (h^b_{\rho(k)})^{\lambda_k u'_j}\right)^{i\in\mathcal{I}_{nr}}_{k\in[1,l], j\in[1,r_a]}.$$
$$\hat{C}_u = \Big(\{C^*_{ukj} = g^{b\lambda_k u'_j}\}_{k\in[1,l], j\in[1,r_u]},$$
$$\{C'_{ukj} = (g^{b^2\cdot ID_{u,j}} h^b_{\rho(k)})^{\lambda_k u'_j}\}_{k\in[1,l], j\in[1,r_u]}\Big),$$
$$\hat{C_a}' = \left(C^*_{dk} = g^{b s_i^{-1}\lambda_k \mu_j}, C'_{dk} = (h^b_{\rho(k)})^{\lambda_k\mu_j}\right)^{i\in\mathcal{I}_{nr}}_{k\in[1,l], j\in[1,r_a]}.$$

$\mathbf{Decrypt3}(CT, SK)$: *CT* is the ciphertext with access structure $(M,\rho)$ and *SK* is a private key for a set $\mathbf{S}$. Suppose that $\mathbf{S}$ satisfies the access structure and let $\mathbf{I}\subset[1,l]$ be defined as $\mathbf{I} = \{i : \rho(i)\in\mathbf{S}\}$. Let $\{\omega_i\in\mathbb{Z}_p\}_{i\in\mathbf{I}}$ be a set of constants such that if $\{\lambda_i\}$ are valid shares of any secret $s$ according to $M$, then $\Sigma_{i\in\mathbf{I}}\omega_i\lambda_i = s$. For the $j^{th}$ revoked user identity, denote the identity of the non-revoked domain authority administrating $ID_u$ by $ID_{a,j}$. The decryption algorithm calculates $e(g,g)^{b^2 t s \mu'_j}$ as follows:

$$\begin{cases} (\prod_{i\in\mathbf{I}}[e(K_{\rho(i)u}, C^*_{uij})\cdot e(L_u, C'_{uij})]^{\omega_i})^{\frac{1}{(ID_u - ID_j)}}, ID_{u,j} \neq ID'_{u,j}, \\ \prod_{i\in\mathbf{I}}[e(K'_{\rho(i)aj}, C^*_{aij})\cdot e(L_{aj}, C'_{aij})]^{\omega_i}, ID_{a,j} \neq ID'_{a,j}. \end{cases} \quad (5)$$

Then we could get $e(g,g)^{b^2 t s \mu_u}$ in the following way:

$$e(g,g)^{b^2 t s \mu_u} = \prod_{j\in[1,r_u]} e(g,g)^{b^2 t s \mu'_j}.$$

For the $j^{th}$ revoked domain authority, denote the identity of the domain authority on the $h_j{}^{th}$ layer managing $ID_u$ by $ID_{a,j}$. The decryption algorithm evaluates $e(g,g)^{b^2 t s u_j}$ as follows:

$$e(g,g)^{b^2 t s u_j} = \prod_{i\in\mathbf{I}}[e(K'_{a\rho(i)}, C^*_{ai})\cdot e(L_a, C'_{ai})]^{\omega_i}.$$

Then we could get $e(g,g)^{b^2 t s \mu_a}$ in the following way:

$$e(g,g)^{b^2 t s \mu_a} = \prod_{j\in[1,r_a]} e(g,g)^{b^2 t s \mu_j}.$$

If *SK*'s holder is not managed by any revoked domain authority and is not among the revoked users, then we can get $e(g,g)^{\alpha s\mu} = \frac{C_0, K}{e(g,g)^{b^2 t s\mu_u}\cdot e(g,g)^{b^2 t s \mu_a}}$. Finally get the message $\mathcal{M}$ by evaluating $\frac{C}{e(g,g)^{\alpha s\mu}}$.

**Theorem 6.** *Suppose the M-q-parallel-BDHE assumption holds. Then no PPT adversary can selectively break the MM-DID-HABE scheme with a challenge access structure* $\left(\mathrm{M}^*, \rho^*\right)$*, where the size of* $M^*$ *is* $\ell^*\times n^*$ *and* $\ell^*, n^* \leq q$.

## VII. Performance Evaluation

In this section, the four schemes proposed in this paper are evaluated in terms of their computation, storage, and communication performance. The evaluation is performed in two parts: First, we analyze the performance complexity of the presented schemes by comparing with the original CP-ABE scheme. Second, we implement these schemes with the PBC library [13]. We conduct a computation performance evaluation and compare these scheme with the CP-ABE scheme.

### A. Complexity Analysis

Following the notations provided in **TABLE I**, a comparative analysis is carried out among the OM-ID-HABE scheme, MM-ID-HABE scheme, OM-DID-HABE scheme, MM-DID-HABE scheme, as well as the original CP-ABE scheme. There are four types of time-consuming operations: pairing, exponentiation, multiplication and inversion, included in the five schemes. According to [14], the pairing and exponentiation operations are the dominant costs. Therefore, we utilize the number of pairing and exponentiation operations as metrics for computation complexity of each scheme. **TABLE II** and **TABLE III** describes the asymptotic complexities of the setup, key generation, encryption and decryption algorithm respectively. In this comparison, we assume that the height of the identity structure tree is 2, *i.e.*, $H = 1$. In addition, for the delegable ID-HABE, the key generation for domain authority is system overall computation overhead, therefore is not included here.

*1) Computation Complexity Analysis:* In the setup algorithm of all the five schemes, only one pairing operation, which is incurred by the evaluation of the value of $e(g, g)^{\alpha}$.

In CP-ABE, the number of exponentiations in the setup algorithm is 3. In both OM-ID-HABE scheme and MM-ID-HABE scheme, $m + 3$ exponentiation operations are needed. While in the two delegable schemes, more exponentiation operations are needed because of the private key generation delegation functionality and resistance against domain authority impersonation as discussed in the security proof part of delegable schemes.

In the key generation algorithm of all the five schemes, no pairing operation is performed. In CP-ABE, the number of exponentiations needed is $|S| + 3$. In both OM-ID-HABE and MM-ID-HABE, this number is increased to $2|S| + 4$. This increase comes from the fact that both user identity and domain authority identity is embedded in the key component for each attribute.

For the encryption algorithm, the computation cost in terms of pairing is the same for CP-ABE, OM-ID-HABE, MM-ID-HABE, OM-DID-HABE and MM-DID-HABE. The same as the key generation algorithm, exponentiation operations dominate the cost. In the encryption algorithm of both CP-ABE and OM-ID-HABE scheme, the number of exponentiation operations is $2l+3$ when revoking a domain authority and $3l+4$ when revoking a particular user. In MM-ID-HABE, the number of exponentiation operations is $(2r_g+3r_u)l+r_g+r_u+1$, where $r_g$ denotes the number of revoked domain authorities and $r_u$ denotes the number of revoked particular users. In the OM-DID-HABE scheme, the number of exponentiation operations is $(|\mathcal{I}_{nr}| + 3)l + 3$ if only one particular user is revoked, and is $(|\mathcal{I}_{nr}| + 1)l + 2$ if only one domain authority is revoked. While for the MM-DID-HABE scheme, if only multiple users are revoked then $x = 1, y = 0$; if only multiple domain authorities are revoked then $x = 0, y = 1$; if there are both multiple users and multiple domain authorities revoked then $x = 1, y = 1$.

In CP-ABE, the number of pairing needed for decryption is $2|\mathbf{I}| + 1$, where $\mathbf{I}$ is the set of users' attributes used in the process of decryption. The decryption algorithm of the OM-ID-HABE and OM-DID-HABE scheme requires the same number of pairing operations as the CP-ABE scheme. Whereas the number increases to be $2|\mathbf{I}|(r_g + r_u) + r_u + 1$ and $2|\mathbf{I}|(r_u + 1)$ respectively since there are multiple users or domain authorities revoked. The numbers of exponentiations in CP-ABE, OM-ID-HABE, MM-ID-HABE, OM-DID-HABE and MM-DID-HABE are $|\mathbf{I}|$, $|\mathbf{I}|$, $|\mathbf{I}|(r_g + r_u)$, $|\mathbf{I}|$and $x(|\mathbf{I}|r_u) + y|\mathbf{I}|$ respectively, where the meaning of $x, y$ is the same as in **TABLE II**.

*2) Storage and Communication Overhead Analysis:* We evaluate the storage and communication overhead separately. The main storage overhead comes from the setup algorithm and key generation algorithm. The communication overheads come from the ciphertext generated by the encryption algorithm. **TABLE IV** and **TABLE V** summarize the storage and communication overhead of the five schemes.

The storage overhead in the setup algorithm of the CP-ABE scheme is $m + 4$. In OM-ID-HABE and MM-ID-HABE, it is $2m + 6$ because of the public parameters generated for the domain authorities. Whereas, in the OM-DID-HABE and MM-DID-HABE schemes, to resist against impersonation problem of the domain authorities, the storage overhead is $2m+2|\mathcal{I}_a|+7$.

In CP-ABE, the overhead of storing the private key is $|\mathbf{S}|+2$. In OM-ID-HABE and MM-ID-HABE, it is $2|\mathbf{S}| + 2$. In OM-DID-HABE and MM-DID-HABE, the private key storage is $2|\mathbf{S}| + 3$.

The size of ciphertext of the CP-ABE scheme is $2l+2$. The ciphertext size of OM-ID-HABE is $2l + 2$ when revoking a domain authority and $4l + 2$ when revoking a particular user. The ciphertext size of MM-ID-HABE, OM-DID-HABE and MM-DID-HABE is $2(r_g + r_u)l + r_g + r_u + 1$, $x(2l + 2r_u) + y(|I_{nr}|l + r_g + 2)$ and $x(2r_u + lr_u|\mathcal{I}_{nr}| + 2lr_u) + y(r_g + l|\mathcal{I}_{nr}|) + 2$ respectively, where the meaning of $r_g$, $r_u$, $x$ and $y$ are the same as the above.

Based on the analysis above, we can see that our proposed ID-HABE schemes incurs more computation overhead compared to the original CP-ABE scheme. The costs for the one identity revocable scheme OM-ID-HABE are almost the same as the CP-ABE scheme. The costs for the one identity revocable scheme OM-DID-HABE, some additional costs are brought because of impersonation resistance and the functionality of delegable private key generation. When

**TABLE II** Computation Complexity Comparison in terms of the Number of Pairing Operations.

| Schemes | CP-ABE | OM-ID-HABE | MM-ID-HABE | OM-DID-HABE | MM-DID-HABE |
|---|---|---|---|---|---|
| Setup | 1 | 1 | 1 | 1 | 1 |
| KeyGen | 0 | 0 | 0 | 0 | 0 |
| Encrypt | 0 | 0 | 0 | 0 | 0 |
| Decryption | $2\|\mathbf{I}\|+1$ | $2\|\mathbf{I}\|+1$ | $2\|\mathbf{I}\|(r_g+r_u)+r_u+1$ | $2\|\mathbf{I}\|+1$ | $2\|\mathbf{I}\|(r_u+1)$ |

**TABLE III** Computation Complexity Comparison in terms of the Number of Exponentiation Operations.

| Schemes | CP-ABE | OM-ID-HABE | MM-ID-HABE | OM-DID-HABE | MM-DID-HABE |
|---|---|---|---|---|---|
| Setup | 3 | $2m+3$ | $2m+3$ | $2m+\|\mathcal{I}_a\|$ | $2m+\|\mathcal{I}_a\|$ |
| KeyGen | $\|\mathbf{S}\|+3$ | $2\|\mathbf{S}\|+4$ | $2\|\mathbf{S}\|+4$ | $2\|\mathbf{S}\|+3$ | $2\|\mathbf{S}\|+3$ |
| Encrypt | $3l+2$ | $2l+3$ or $3l+4$ | $(2r_g+3r_u)l$ + $r_g+r_u+1$ | $(\|\mathcal{I}_{nr}\|+3)l+3$ or $(\|\mathcal{I}_{nr}\|+1)l+2$ | $x((\|\mathcal{I}_{nr}\|+2)r_ul+r_u+2)$ + $y((\|\mathcal{I}_{nr}\|+1)l+2)$ |
| Decrypt | $\|\mathbf{I}\|$ | $\|\mathbf{I}\|$ | $\|\mathbf{I}\|(r_g+r_u)$ | $\|\mathbf{I}\|$ | $x(\|\mathbf{I}\|r_u)+y\|\mathbf{I}\|$ |

**TABLE IV** Storage Overhead Comparison.

| Schemes | CP-ABE | OM-ID-HABE | MM-ID-HABE | OM-DID-HABE | MM-DID-HABE |
|---|---|---|---|---|---|
| Setup | $m+4$ | $2m+6$ | $2m+6$ | $2m+2\|\mathcal{I}_a\|+7$ | $2m+2\|\mathcal{I}_a\|+7$ |
| KeyGen | $\|\mathbf{S}\|+2$ | $2\|\mathbf{S}\|+2$ | $2\|\mathbf{S}\|+2$ | $2\|\mathbf{S}\|+3$ | $2\|\mathbf{S}\|+3$ |

**TABLE V** Communication Overhead Comparison.

| Schemes | CP-ABE | OM-ID-HABE | MM-ID-HABE | OM-DID-HABE | MM-DID-HABE |
|---|---|---|---|---|---|
| Encrypt | $2l+2$ | $2l+2$ or $4l+2$ | $2(r_g+2r_u)l$ + $r_g+r_u+1$ | $x(2l+2r_u)$ + $y(\|\mathcal{I}_{nr}\|l+r_g+2)$ | $x(2r_u+lr_u\|\mathcal{I}_{nr}\|+2lr_u)$ + $y(r_g+l\|\mathcal{I}_{nr}\|)+2$ |

there are multiple identities included in the revocation list, the computation overhead is proportional to the number of revoked identities. Although some times the computation overhead is high, the new functionalities and properties brought by our scheme are useful in cloud-based secure data sharing applications.

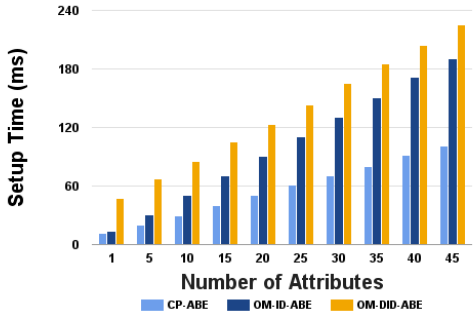


**Fig. 2** Relations between the amount of attributes and time consumption for setup.

## VIII. Related Work

Traditionally, access control is based on the identity of a user, either directly or through predefined attributes types, *e.g.*, roles or groups assigned to that user. However, practitioners have noted that this access control approach usually needs cumbersome management and identity, groups and roles are

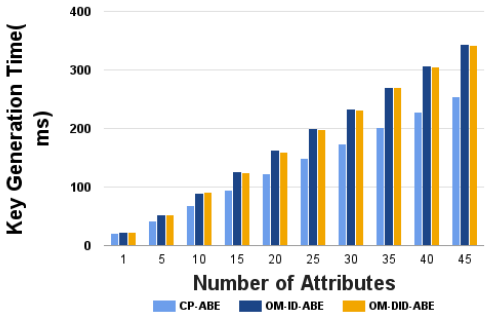


**Fig. 3** Relations between the amount of attributes and time consumption for key generation.

not sufficient in expressing the access control policies in the real world. Therefore, a new approach which is referred to as attribute-based access control (ABAC) is proposed [15]. With ABAC, whether a user's request is granted or not is decided by the attributes of the user, selected attributes of the object and environment conditions that can be globally recognized. Compared with role-based access control, ABAC provides the following nice properties. First, ABAC is more expressive; Second, ABAC enables access control policy enforcement without prior knowledge of the specific subjects. Because of its flexibility, ABAC is nowadays the fastest-growing access control model [7], [16], [17].

There are several approaches to implementing ABAC,

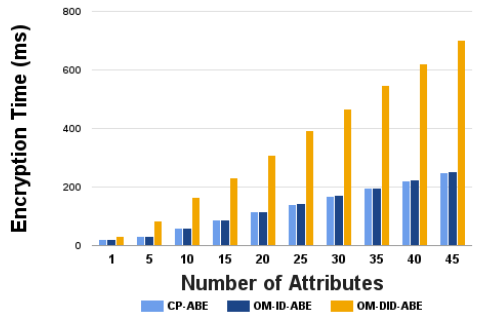


**Fig. 4** Relations between the amount of attributes and time consumption for encryption.

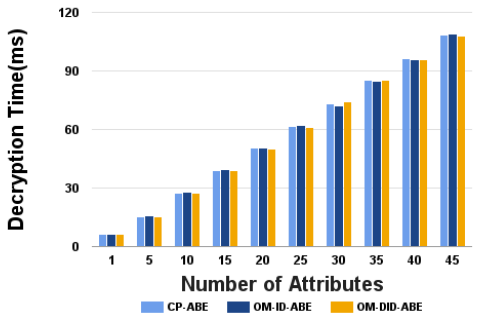


**Fig. 5** Relations between the amount of attributes and time consumption for decryption.

among which attribute-based encryption (ABE) is regarded as the most suitable one for data access control in applications scenarios where server-based access control cannot be implemented, *e.g.*, cloud computing and MANET. There exist two complementary forms of ABE, *i.e.*, Key-Policy ABE (KP-ABE) [18] where the decryption key is associated to the access control policy and Ciphertext-Policty ABE (CP-ABE) [11], [19]–[22] where the ciphertext is associated to the access control policy. CP-ABE is more suitable for enforcing data access control over data stored on the cloud servers. CP-ABE allows data owners to define an access structure on attributes and upload the data encrypted under this access structure to the cloud servers. Therefore, CP-ABE enables users to define the attributes a data user needs to possess in order to access the data. As promising as it is, CP-ABE suffers from user revocation problem. This issue is first addressed in [23] as a rough idea. There are also several following researches [24]–[27], which as we discussed in the introduction are not suitable for user revocation.

User revocation is always an important problem in the cryptographic research area. Boldyreva *et al*. [12] proposed an identity-based scheme with efficient user revocation capability. It applies key updates with significantly reduced computational cost based on a binary tree data structure, which is also applicable to KP-ABE and fuzzy IBE user revocation. However, its applicability to CP-ABE is not clear. Libert *et al*. [28] proposed an identity-based encryption scheme with stronger adaptive-ID sense to address the selective security issue of [12]. Lewko *et al*. [29] two novel broadcast encryption schemes with effective user revocation capability. EASiER [30] architecture is described to support fine-grained access control policies and dynamic group membership based on attribute-based encryption. It relies on a proxy to participate in the decryption and enforce revocation, such that the user can be revoked without re-encrypting ciphertexts or issuing new keys to other users. Chen *et al*. [31] presented an identity-based encryption scheme based on lattices to realize efficient key revocation. Li *et al*. [32] first introduced outsourcing computation in identity-based encryption and presented a revocable in the server-aided settings. As a result, it achieves constant computation cost at public key generator and private key size at user, and the user does not have to contact public key generator for key update.

## IX. Conclusions and Future Work

In this paper, we investigate the problem of how to revoke a user when applying the CP-ABE scheme for secure data sharing. Compared to previous solutions on attribute-based revocation, our approach focuses on identity-based revocation in CP-ABE, which solves the scalability issues when using attribute-based revocation to revoke users. We propose the primitive of ID-ABE, and give its security definition. We present four constructions and validate efficiency of these constructions through both complexity analysis and real implementation based tests.

There are several research issues need to be further investigated. First, the revoked users' identities must be included in the ciphertext, which might lead to private information leakage. Second, in this work we only focus on identity-based revocation, while previous researches focus on attribute-based revocation, and it will be interesting to investigate into how to combine these two revocation properties together to achieve more flexible revocation methods. Third, currently, all the private components of a user's private key are obtained from the trusted authority or the same domain authority. We plan to investigate how to revoke users whose private key incorporating private components from several different domain authorities.

## References

[1] J. H. Saltzer and M. D. Schroeder, "The protection of information in computer systems," *Proceedings of the IEEE*, vol. 63, no. 9, pp. 1278–1308, 1975.

[2] J. G. Steiner, B. C. Neuman, and J. I. Schiller, "Kerberos: An authentication service for open network systems." in *USENIX Winter*, 1988, pp. 191–202.

[3] R. Krishnan, J. Niu, R. Sandhu, and W. H. Winsborough, "Group-centric secure information-sharing models for isolated groups," *ACM Transactions on Information and System Security (TISSEC)*, vol. 14, no. 3, p. 23, 2011.

[4] R. Sandhu, "Rationale for the rbac96 family of access control models," in *Proceedings of the first ACM Workshop on Role-based access control*. ACM, 1996, p. 9.

[5] X. Jin, R. Krishnan, and R. Sandhu, "A unified attribute-based access control model covering dac, mac and rbac," in *IFIP Annual Conference on Data and Applications Security and Privacy*. Springer, 2012, pp. 41–55.

[6] B. Waters, "Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization," in *Public Key Cryptography - PKC 2011*. Springer-Verlag, 2011, pp. 53–70.

[7] V. C. Hu, D. R. Kuhn, and D. F. Ferraiolo, "Attribute-based access control." *IEEE Computer*, vol. 48, no. 2, pp. 85–88, 2015.

[8] A. Sahai, H. Seyalioglu, and B. Waters, "Dynamic credentials and ciphertext delegation for attribute-based encryption," in *Advances in Cryptology–CRYPTO 2012*. Springer, 2012, pp. 199–217.

[9] S. Yamada, N. Attrapadung, G. Hanaoka, and N. Kunihiro, "A framework and compact constructions for non-monotonic attribute-based encryption," in *International Workshop on Public Key Cryptography*. Springer, 2014, pp. 275–292.

[10] A. Beimel, *Secure schemes for secret sharing and key distribution*. Technion-Israel Institute of technology, Faculty of computer science, 1996.

[11] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *IEEE SP'07*, Oakland, CA, May 2007, pp. 321–334.

[12] A. Boldyreva, V. Goyal, and V. Kumar, "Identity-based encryption with efficient revocation," in *Proceedings of the 15th ACM conference on Computer and communications security*. ACM, 2008, pp. 417–426.

[13] B. Lynn, "The pairing-based cryptography library," 2006. [Online]. Available: https://crypto.stanford.edu/pbc/

[14] B. Li, A. P. Verleker, D. Huang, Z. Wang, and Y. Zhu, "Attribute-based access control for icn naming scheme," in *Communications and Network Security (CNS), 2014 IEEE Conference on*, 2014, pp. 391–399.

[15] V. C. Hu, D. Ferraiolo, and D. R. Kuhn, *Assessment of access control systems*. US Department of Commerce, National Institute of Standards and Technology, 2006.

[16] V. Hu, D. F. Ferraiolo, D. R. Kuhn, R. N. Kacker, and Y. Lei, "Implementing and managing policy rules in attribute based access control," in *Information Reuse and Integration (IRI), 2015 IEEE International Conference on*. IEEE, 2015, pp. 518–525.

[17] V. C. Hu, D. Ferraiolo, R. Kuhn, A. R. Friedman, A. J. Lang, M. M. Cogdell, A. Schnitzer, K. Sandlin, R. Miller, K. Scarfone *et al.*, "Guide to attribute based access control (abac) definition and considerations (draft)," *NIST Special Publication*, vol. 800, no. 162, 2013.

[18] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," *Proceedings of the 13th ACM conference on Computer and communications security*, pp. 89–98, 2006.

[19] L. Cheung and C. Newport, "Provably secure ciphertext policy abe," in *ACM conference on Computer and Communications Security*, Alexandria, Virginia, USA, 2007, pp. 456–465.

[20] J. Katz, A. Sahai, and B. Waters, "Predicate encryption supporting disjunctions, polynomial equations, and inner products," in *Proc. of EUROCRYPT 2008*. Springer-Verlag, 2008, pp. 146–162.

[21] R. Ostrovsky and B. Waters, "Attribute-based encryption with non-monotonic access structures," in *Proceedings of the 14th ACM conference on Computer and communications security*. ACM New York, NY, USA, 2007, pp. 195–203.

[22] V. Goyal, A. Jain, O. Pandey, and A. Sahai, "Bounded ciphertext policy attribute based encryption," in *Automata, languages and programming*. Springer, 2008, pp. 579–591.

[23] S. Yu, K. Ren, and W. Lou, "Attribute-based content distribution with hidden policy," in *Secure Network Protocols, 2008. NPSec 2008. 4th Workshop on*. IEEE, 2008, pp. 39–44.

[24] ——, "Attribute-based on-demand multicast group setup with membership anonymity," *Computer Networks*, vol. 54, no. 3, pp. 377–386, 2010.

[25] B. Li, Z. Wang, and D. Huang, "An efficient and anonymous attribute-based group setup scheme," in *2013 IEEE Global Communications Conference (GLOBECOM)*, 2013, pp. 861–866.

[26] J. Hur and D. K. Noh, "Attribute-based access control with efficient revocation in data outsourcing systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 7, pp. 1214–1221, 2011.

[27] K. Yang, X. Jia, and K. Ren, "Attribute-based fine-grained access control with efficient revocation in cloud storage systems," in *Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security*. ACM, 2013, pp. 523–528.

[28] B. Libert and D. Vergnaud, "Adaptive-id secure revocable identity-based encryption," in *Topics in Cryptology–CT-RSA 2009*. Springer, 2009, pp. 1–15.

[29] A. Lewko, A. Sahai, and B. Waters, "Revocation systems with very small private keys," in *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 2010, pp. 273–285.

[30] S. Jahid, P. Mittal, and N. Borisov, "Easier: Encryption-based access control in social networks with efficient revocation," in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*. ACM, 2011, pp. 411–415.

[31] J. Chen, H. W. Lim, S. Ling, H. Wang, and K. Nguyen, "Revocable identity-based encryption from lattices," in *Information Security and Privacy*. Springer, 2012, pp. 390–403.

[32] J. Li, J. Li, X. Chen, C. Jia, and W. Lou, "Identity-based encryption with outsourced revocation in cloud computing," *Computers, IEEE Transactions on*, vol. 64, no. 2, pp. 425–437, 2015.

[33] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization."

[34] D. Boneh, X. Boyen, and E.-J. Goh, "Hierarchical identity based encryption with constant size ciphertext," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2005, pp. 440–456.

## APPENDIX A
## SECURITY PROOF OF THEOREM 1

In this section, we briefly show that the $M$-$q$-parallel-BDHE assumption is generically secure. The generic proof template of BBG [33] and [34] is used. Using the terminology from BBG we need to show that $f = a^{q+1}s$ is independent of the polynomials $P$ and $Q$. We set $Q = \{1\}$ since all given terms are in the bilinear group. $P$ is set to be

$$P = \{1, s, \forall_{i\in[1,2q],j\in[1,q],i\neq q+1} a^i, a^i/b_j, a^i \cdot s/b_j\}.$$

We could choose a generator $u$ and set $g = u^{\prod_{j\in[1,q]} b_j}$. All the above terms are substituted by a set of polynomials with the maximum degree $3q + 1$.

Now, we check whether $f$ is symbolically independent of any two polynomials in $P$ and $Q$. To realize $f$ from $P$ and $Q$, a term of the form $a^{m+1}s$ is needed. It can be seen that no such terms can be realized from the product of any two polynomials $p, p' \in P$. To form such a term, a polynomial with a single factor of $s$ is needed. If $s$ is used as $p$ then $p'$ has to be $a^{q+1}$ which doesn't exist in $P$. If we set $p = a^i \cdot s/b_j$, there always exists $b_j$, which cannot be canceled. Based on the BBG framework, we can conclude that the $M$-$q$-parallel-BDHE assumption is generically secure.

## APPENDIX B
## SECURITY PROOF OF THEOREM 2

The basic idea of our proof is if there exists a PPT adversary $\mathcal{A}$ who wins the security game defined in our security model section, then we could use the attacking capability of this adversary to solve the $M$-$q$-parallel-BDHE assumption. Since this assumption is proven to be generically secure, we get a contradiction. Thus we could conclude $\mathcal{A}$ doesn't exist and the OO-ID-HABE scheme is secure as defined by in **Definition 1**. In particular, We show that $\mathcal{B}$ could use the items obtained in the $M$-$q$-parallel-BDHE game to simulate as the challenger in the query phases of OO-ID-HABE security game successfully. In addition, through embedding the $M$-$q$-parallel-BDHE challenge in the challenge ciphertext sent to $\mathcal{A}$, $\mathcal{B}$ could take advantage of $\mathcal{A}$'s attacking capability.

*Proof.* **Init** $\mathcal{B}$ takes in a modified decisional $q$-parallel $BDHE$ challenge $\{\mathbf{y}, T\}$. Then the adversary $\mathcal{A}$ declares the revoked user $\mathbf{ID}^*$ and gives the simulator the challenge access structure $(M^*, \rho^*)$, where the size of $M^*$ is $\ell^* \times n^*$ and $\ell^*, n^* \leq q$. Define the challenge matrix $M^*$ as $M^* = (M_1^*, M_2^*, \cdots, M_{l^*}^*)^T$.
**Setup** $\mathcal{B}$ a group $\mathbb{G}$ of prime order $p$, a generator $g$, and a random value $\alpha'$ and sets $e(g,g)^{\alpha} = e(g,g)^{\alpha'} e(g^a, g^{a^q})$, which implicitly sets $\alpha = \alpha' + a^{q+1}$. Additionally, it implicitly sets $b = a$ by setting the public parameters as

$$g, g^b = g^a, g^{b^2} = g^{a^2}$$

To embed the revoked identity $\mathbf{ID}^* = \{ID'\}$ and the challenge access structure in the public parameters $\{h_x^b\}_{x \in \mathbf{U}}$, we regard the challenge matrix $M^*$ as a row vector set and divide it into three subsets $M^{*\prime}$, $M^{*\prime\prime}$ and $M^{*\prime\prime\prime}$ such that $M^{*\prime} \cup M^{*\prime\prime} \cup M^{*\prime\prime\prime} = M^*$ and $M^{*\prime} \cap M^{*\prime\prime} \cap M^{*\prime\prime\prime} = \emptyset$. Specifically, $M^{*\prime}$, $M^{*\prime\prime}$ and $M^{*\prime\prime\prime}$ are initially set to be null. Define the $n^*$-dimension vector $\mathbf{e} = (1, 0, ..., 0)$ and vector $\mathbf{u} = (a^2, a^3, ..., a^{n+1})$. For $i \in [1, l]$, if $M_i^*$ is linearly independent on $M^{*\prime}$ and $\mathbf{e}$ cannot be linearly expressed by $M^{*\prime} \cup \{M_i^*\}$, then we merge $M_i^*$ into $M^{*\prime}$; if $M_i^*$ is linearly independent on $M^{*\prime}$ and $\mathbf{e}$ can be linearly expressed by $M^{*\prime} \cup \{M_i^*\}$, then we merge $M_i^*$ into $M^{*\prime\prime\prime}$; if $M_i^*$ is linearly dependent on $M^{*\prime}$, then we merge $M_i^*$ into $M^{*\prime\prime}$. As a result, $M^{*\prime}$ is a linear independent vector group while each vector in $M^{*\prime\prime}$ can be linearly expressed by $M^{*\prime}$. Although $\mathbf{e}$ cannot be spanned by $M^{*\prime}$, it can be linearly expressed by $M^{*\prime}$ merged with each vector in $M^{*\prime\prime\prime}$. Therefore, each vector in $M$ can be linearly expressed by $M^{*\prime} \cup \{\mathbf{e}\}$.

Let $X$ denote the set of index $i$, such that $\rho(i) = x$. Assume that there are $m$ vectors in $M^{*\prime}$ and let $M^{*\prime} = (M_1^{*\prime}, M_2^{*\prime}, ..., M_m^{*\prime})^T$. For each $i \in X$, its corresponding row vector $M_i$ can be written as $\varepsilon_{i0}\mathbf{e} + \varepsilon_{i1} M_1^{*\prime} + \varepsilon_{i2} M_2^{*\prime} + ... + \varepsilon_{im} M_m^{*\prime}$, where $(\varepsilon_{i0}, \varepsilon_{i1}, ..., \varepsilon_{im}) \in \mathbb{Z}_p^{m+1}$. By choosing a random value $z_x$, $\mathcal{B}$ programs $h_x^b$ as:

$$\begin{aligned} h_x =& g^{z_x} \Big( \prod_{i \in X} g^{(\varepsilon_{i0}\mathbf{e} + \varepsilon_{i1} M_1^{*\prime} + \cdots + \varepsilon_{im} M_m^{*\prime}) \cdot \mathbf{u}/b_i} \Big)^{-ID'} \\ =& g^{z_x} \Big( \prod_{i \in X} \prod_{j \in [1, n^*]} g^{(\varepsilon_{i0}\mathbf{e}_j + \varepsilon_{i1} M_{1j}^{*\prime} + \cdots + \varepsilon_{im} M_{mj}^{*\prime}) a^{j+1}/b_i} \Big)^{-ID'}. \end{aligned}$$

If $X$ is an empty set, we set $h_x^b = g^{z_x}$. Then $\mathcal{B}$ sends the above parameters $\left(g, g^b, g^{b^2}, e(g,g)^{\alpha}, \{h_x^b\}_{x \in \mathbf{U}}\right)$ to $\mathcal{A}$.

Note that the distribution of the generated public parameters is the same as that in the *Setup* of the OO-ID-HABE scheme. In addition, the revoked identity *ID* and the challenge access structure are embedded in the public parameters as well.
**Phase 1** For a query $(\mathbf{S}, ID)$, $\mathcal{B}$ constructs the private key as follows. Since $M^{*\prime}$ is a set of linearly independent vectors and the vector $\mathbf{e}$ is not in the span of $M^{*\prime}$, we can find a vector $\mathbf{w}$ with $\mathbf{w}_1 = -1$ and $\mathbf{w} \cdot M_i^{*\prime} = 0$, where $1 \leq i \leq m$.

Therefore, $\mathcal{B}$ selects a random element $r \in \mathbb{Z}_p$ and sets the private key $L$ to be

$$L = g^{r + \mathbf{w} \cdot \mathbf{v}} = g^r \prod_{i = 1, \cdots, n^*} (g^{a^{q-i}})^{\mathbf{w}_i},$$

which implicitly sets the random element $t$ as

$$t = r + \mathbf{w} \cdot \mathbf{v} = r + \mathbf{w}_1 a^{q-1} + \mathbf{w}_2 a^{q-2} + \cdots + \mathbf{w}_n a^{q-n^*},$$

where $\mathbf{v} = (a^{q-1}, a^{q-2}, \cdots, a^{q-n^*+2})$. Since $g^{a^2 t}$ contains a term of $g^{-a^{q+1}}$ we can cancel out with the unknown term in $g^{\alpha}$ when creating the $K$ component in the private key. $\mathcal{B}$ set $K$ as follows

$$K = g^{\alpha'} g^{a^2 r} \prod_{i \in [1, n^*]} (g^{a^{q+2-i}})^{\omega_i}.$$

For $\forall x \in \mathbf{S}$, if there is no $i$ such that $\rho^*(i) = x$, $\mathcal{B}$ simply sets $K_x = L^{z_x}$. For those used in the challenge access structure, we must make sure that there are no terms of the form $g^{a^{q+1}/b_i}$ that $\mathcal{B}$ can't simulate. Since $\mathbf{w} \cdot M_i^{*\prime} = 0$, all of these terms cancel. Define $X$ as the set of all $i$ such that $\rho^*(i) = x$, $\mathcal{B}$ creates $K_x$ as follows

$$\begin{aligned} K_x =& \Big( g^{aID} g^{z_x} \big( \prod_{i \in X} g^{(\varepsilon_{i0}\mathbf{e} + \varepsilon_{i1} M_1^{*\prime} + , \cdots, + \varepsilon_{im} M_m^{*\prime}) \cdot \frac{\mathbf{u}}{b_i}} \big) - ID' \Big)^{(r + \mathbf{w} \cdot \mathbf{v})} \\ =& L^{a \cdot ID + z_x} \Big( \big( \prod_{i \in X} g^{(\varepsilon_{i0}\mathbf{e} + \varepsilon_{i1} M_1^{*\prime} + , \cdots, + \varepsilon_{im} M_m^{*\prime}) \cdot \frac{\mathbf{u}}{b_i}} \big) - ID' \Big)^{(r + \mathbf{w} \cdot \mathbf{v})} \\ =& L^{a \cdot ID + z_x} \prod_{i \in X} \prod_{j \in [1, n^*]} \Big( g^{(a^j/b_i) r} \prod_{k \neq j}^{k \in [1, n^*]} (g^{\frac{a^{q+j-k}}{b_i}})^{\mathbf{w}_k} \Big)^{exp_{i,j}} \end{aligned}$$

where $exp_{i,j} = -ID' \cdot (\varepsilon_{i0}\mathbf{e}_j + \varepsilon_{i1} M_{1j}^{*\prime} + \cdots + \varepsilon_{im} M_{mj}^{*\prime})$.
**Challenge** $\mathcal{A}$ provides two equal length messages $\mathcal{M}_0$ and $\mathcal{M}_1$ to $\mathcal{B}$ as the challenge messages.

First, The simulator flips a coin $b$ and creates the ciphertext component $C = \mathcal{M}_b T \cdot e(g^s, g^{\alpha'})$. Then the simulator chooses random value $y_2', y_3', \cdots, y_{n^*}'$ and share the secret $s$ using the vector

$$\mathbf{v} = (s, y_2', y_3', \cdots, y_{n^*}').$$

Next, it calculates

$$\lambda_k = \mathbf{v} \cdot (\varepsilon_{k0}\mathbf{e} + \varepsilon_{k1} M_1^{*\prime} + \varepsilon_{k2} M_2^{*\prime} + \cdots + \varepsilon_{km} M_m^{*\prime})$$

and generates the ciphertext component $C_k^*$ as follows

$$C_k^* = g^{a \cdot \sum_{i=1}^m (\varepsilon_{k0}\mathbf{e} + \varepsilon_{ki} M_{1i}')s} \cdot \prod_{j=2}^{n^*} g^{a \cdot \sum_{i=1}^m \varepsilon_{kj} M_{ij}' y_j'}.$$

For $k \in [1, l^*]$, we define $X_k$ as the set of the index $i$ in such that $\rho^*(i) = \rho^*(k)$. Finally, the simulator builds the ciphertext component $C_k'$ as:

$$\begin{aligned} C_k' =& g^{\sum_{i=1}^m (\varepsilon_{k0}\mathbf{e} + \varepsilon_{ki} M_{i1}') z_{\rho^*(k)} s} \cdot g^{\sum_{j=2}^{n^*} \sum_{i=1}^m \varepsilon_{ki} M_{ij}' y_j' z_{\rho^*(k)}} \\ & \cdot \prod_{i \in X_k} \prod_{\xi=1}^{n^*} g^{-(\sum_{j=1}^m \varepsilon_{ij} M_{j\xi}')(1 + \sum_{j=1}^m \varepsilon_{kj} M_{j1}') ID' a^{\xi+1} s/b_i} \\ & \cdot \prod_{i \in X_k} \prod_{\xi=1}^{n^*} g^{-(\sum_{j=1}^m \varepsilon_{ij} M_{j\xi}')(\sum_{l=2}^{n^*} \sum_{j=1}^m y_l' \varepsilon_{kj} M_{jl}') ID' a^{\xi+1}/b_i}. \end{aligned}$$

**Phase II** Same as phase I.

**Guess** $\mathcal{A}$ outputs a bit $b'$. If $b = b'$ then $\mathcal{B}$ guesses $T = e(g,g)^{sa^{q+1}}$ else guesses $T$ to be a random group element in $\mathbb{G}_T$. When $T = e(g,g)^{sa^{q+1}}$, $\mathcal{B}$ could simulate perfectly, therefore we have

$$\Pr[\mathcal{B}(\mathbf{y}, T = e(g,g)^{sa^{q+1}}) = 0] = \frac{1}{2} + \mathrm{Adv}_{\mathcal{A}}.$$

When $T$ is a random group element, the message $M_\beta$ is completely hidden from $\mathcal{A}$. Thus we have $\Pr[\mathcal{B}(\mathbf{y}, T = R) = 0] = \frac{1}{2}$. If the adversary $\mathcal{A}$ could attack the OO-ID-HABE scheme with non-negligible advantage, then $\mathcal{B}$'s advantage in the $M$-$q$-parallel BDHE problem is non-negligible as well. Thus, we could conclude that our OO-ID-HABE scheme is secure.

□

## APPENDIX C
## SECURITY PROOF OF THEOREM 3

The basic idea of our proof is if there exists a PPT adversary $\mathcal{A}$ who wins the security game defined in our security model section, then we could construct an efficient adversary $\mathcal{B}$ to solve the $M$-$q$-parallel-BDHE assumption.

*Proof.* **Init** $\mathcal{B}$ takes in a modified decisional $q$-parallel $BDHE$ challenge $\{\mathbf{y}, T\}$. Then the adversary $\mathcal{A}$ declares the revoked user $\mathbf{ID}^*$ and gives the simulator the challenge access structure $(M^*, \rho^*)$, where the size of $M^*$ is $\ell^* \times n^*$ and $\ell^*, n^* \leq q$. Define the challenge matrix $M^*$ as $M^* = (M_1^*, M_2^*, \cdots, M_{l^*}^*)^T$.
**Setup** $\mathcal{B}$ a group $\mathbb{G}$ of prime order $p$, a generator $g$, and a random value $\alpha'$ and sets $e(g,g)^{\alpha} = e(g,g)^{\alpha'} e(g^a, g^{a^q})$, which implicitly sets $\alpha = \alpha' + a^{q+1}$, which implicitly sets $\alpha = \alpha_1' + a^{q+1}$.

Additionally, it implicitly sets $b = a$ by setting the public parameters as

$$g, g^b = g^a, g^{b^2} = g^{a^2}$$

To embed the revoked identity $\mathbf{ID}^* = \{ID'_{Ha}, ID'\}$ and the challenge access structure in the public parameters $\{h_x^b\}_{x \in \mathbf{U}}$, we regard the challenge matrix $M^*$ as a row vector set and divide it into three subsets $M^{*\prime}$, $M^{*\prime\prime}$ and $M^{*\prime\prime\prime}$ such that $M^{*\prime} \cup M^{*\prime\prime} \cup M^{*\prime\prime\prime} = M^*$ and $M^{*\prime} \cap M^{*\prime\prime} \cap M^{*\prime\prime\prime} = \emptyset$. Specifically, $M^{*\prime}$, $M^{*\prime\prime}$ and $M^{*\prime\prime\prime}$ are initially set to be null. Define the $n^*$-dimension vector $\mathbf{e} = (1, 0, ..., 0)$ and vector $\mathbf{u} = (a^2, a^3, ..., a^{n+1})$. For $i \in [1, l]$, if $M_i^*$ is linearly independent on $M^{*\prime}$ and $\mathbf{e}$ cannot be linearly expressed by $M^{*\prime} \cup \{M_i^*\}$, then we merge $M_i^*$ into $M^{*\prime}$; if $M_i^*$ is linearly independent on $M^{*\prime}$ and $\mathbf{e}$ can be linearly expressed by $M^{*\prime} \cup \{M_i^*\}$, then we merge $M_i^*$ into $M^{*\prime\prime\prime}$; if $M_i^*$ is linearly dependent on $M^{*\prime}$, then we merge $M_i^*$ into $M^{*\prime\prime}$. As a result, $M^{*\prime}$ is a linear independent vector group while each vector in $M^{*\prime\prime}$ can be linearly expressed by $M^{*\prime}$. Although $\mathbf{e}$ cannot be spanned by $M^{*\prime}$, it can be linearly expressed by $M^{*\prime}$ merged with each vector in $M^{*\prime\prime\prime}$. Therefore, each vector in $M$ can be linearly expressed by $M^{*\prime} \cup \{\mathbf{e}\}$.

Let $X$ denote the set of index $i$, such that $\rho(i) = x$. Assume that there are $m$ vectors in $M^{*\prime}$ and let $M^{*\prime} = (M_1^{*\prime}, M_2^{*\prime}, ..., M_m^{*\prime})^T$. For each $i \in X$, its corresponding row vector $M_i$ can be written as $\varepsilon_{i0}\mathbf{e} + \varepsilon_{i1} M_1^{*\prime} + \varepsilon_{i2} M_2^{*\prime} + ... + \varepsilon_{im} M_m^{*\prime}$, where $(\varepsilon_{i0}, \varepsilon_{i1}, ..., \varepsilon_{im}) \in \mathbb{Z}_p^{m+1}$. By choosing a random value $z_x$, $\mathcal{B}$ programs $h_x$ and $h_{xh}$ $(h \in [1, H])$ as

$$\begin{aligned} h_x =& g^{z_x} \Big( \prod_{i \in X} g^{(\varepsilon_{i0}\mathbf{e} + \varepsilon_{i1} M_1^{*\prime} + \cdots + \varepsilon_{im} M_m^{*\prime}) \cdot \mathbf{u} / b_i} \Big)^{-ID'} \\ =& g^{z_x} \Big( \prod_{i \in X} \prod_{j \in [1, n^*]} g^{(\varepsilon_{i0}\mathbf{e}_j + \varepsilon_{i1} M_{1j}^{*\prime} + \cdots + \varepsilon_{im} M_{mj}^{*\prime}) a^{j+1} / b_i} \Big)^{-ID'}. \end{aligned}$$

$$\begin{aligned} h_{xh} =& g^{z_x} \Big( \prod_{i \in X} g^{(\varepsilon_{i0}\mathbf{e} + \varepsilon_{i1} M_1^{*\prime} + \cdots + \varepsilon_{im} M_m^{*\prime}) \cdot \mathbf{u} / b_i} \Big)^{-ID'_{ah}} \\ =& g^{z_x} \Big( \prod_{i \in X} \prod_{j \in [1, n^*]} g^{(\varepsilon_{i0}\mathbf{e}_j + \varepsilon_{i1} M_{1j}^{*\prime} + \cdots + \varepsilon_{im} M_{mj}^{*\prime}) a^{j+1} / b_i} \Big)^{-ID'_{ah}}. \end{aligned}$$

If $X$ is an empty set, $h_x = h'_x = g^{z_x}$. Then $\mathcal{B}$ sends $\Big(g, g^b, g^{b^2}, e(g,g)^{\alpha}, \{h_x^b\}_{x \in \mathbf{U}}, \{{h_{xi}}^b\}_{x \in \mathbf{U}, i \in [1,H]}\Big)$ to $\mathcal{A}$.

Note that the distribution of the generated public parameters is the same as that in the *Setup* of the OM-ID-HABE scheme. In addition, the challenge access structure are embedded in the public parameters as well.

**Phase 1** For a query $(\mathbf{S}, ID)$, $\mathcal{B}$ constructs the private key as follows. Since $M^{*\prime}$ is a set of linearly independent vectors and the vector $\mathbf{e}$ is not in the span of $M^{*\prime}$, we can find a vector $\mathbf{w}$ with $\mathbf{w}_1 = -1$ and $\mathbf{w} \cdot M_i^{*\prime} = 0$, where $1 \leq i \leq m$.

Therefore, $\mathcal{B}$ selects a random element $r \in \mathbb{Z}_p$ and sets the private key $L$ to be

$$L = g^{r + \mathbf{w} \cdot \mathbf{v}} = g^r \prod_{i = 1, \cdots, n^*} (g^{a^{q-i}})^{\mathbf{w}_i},$$

which implicitly sets the random element $t$ as

$$t = r + \mathbf{w} \cdot \mathbf{v} = r + \mathbf{w}_1 a^{q-1} + \mathbf{w}_2 a^{q-2} + \cdots + \mathbf{w}_n a^{q-n^*},$$

where $\mathbf{v} = (a^{q-1}, a^{q-2}, \cdots, a^{q-n^*+2})$. Since $g^{a^2 t}$ contains a term of $g^{-a^{q+1}}$ we can cancel out with the unknown term in $g^{\alpha}$ when creating the $K$ component in the private key. $\mathcal{B}$ set $K$ as follows

$$K = g^{\alpha'} g^{a^2 r} \prod_{i \in [1, n^*]} (g^{a^{q+2-i}})^{\omega_i}.$$

For $\forall x \in \mathbf{S}$, if there is no $i$ such that $\rho^*(i) = x$, $\mathcal{B}$ simply sets $K_{xa} = K_{xu} = L^{z_x}$. For those used in the challenge access structure, we must make sure that there are no terms of the form $g^{a^{q+1}/b_i}$ that $\mathcal{B}$ can't simulate. Since $\mathbf{w} \cdot M_i^{*\prime} = 0$, all of these terms cancel. Define $X$ as the set of all $i$ such that $\rho^*(i) = x$, $\mathcal{B}$ creates $K_x$ as follows

$$\begin{aligned} K_{xia} =& \Big( g^{aID_{ai}} g^{z_x} \big( \prod_{i \in X} g^{(\varepsilon_{i0}\mathbf{e} + \varepsilon_{i1} M_1^{*\prime} + \cdots + \varepsilon_{im} M_m^{*\prime}) \cdot \frac{\mathbf{u}}{b_i}} \big)^{-ID'_{ai}} \Big)^{(r + \mathbf{w} \cdot \mathbf{v})} \\ =& L^{a \cdot ID_{ai} + z_x} \Big( \big( \prod_{i \in X} g^{(\varepsilon_{i0}\mathbf{e} + \varepsilon_{i1} M_1^{*\prime} + \cdots + \varepsilon_{im} M_m^{*\prime}) \cdot \frac{\mathbf{u}}{b_i}} \big)^{-ID'_{ai}} \Big)^{(r + \mathbf{w} \cdot \mathbf{v})} \\ =& L^{a \cdot ID_{ai} + z_x} \prod_{i \in X} \prod_{j \in [1, n^*]} \Big( g^{\frac{aj}{b_i} \cdot r} \prod_{k \in [1, n^*]}^{k \neq j} (g^{\frac{a^{q+j-k}}{b_i}})^{\mathbf{w}_k} \Big)^{exp_{i,j}} \end{aligned}$$

where $exp_{i,j} = -ID'_{ai} \cdot (\varepsilon_{i0}\mathbf{e}_j + \varepsilon_{i1}M^{*\prime}_{1j} + \cdots + \varepsilon_{im}M^{*\prime}_{mj})$.

$$K_x = \Big(g^{aID} g^{z_x} (\prod_{i\in X} g^{(\varepsilon_{i0}\mathbf{e}+\varepsilon_{i1}M_1^{*\prime}+,\cdots,+\varepsilon_{im}M_m^{*\prime})\cdot\frac{\mathbf{u}}{b_i}})^{-ID'}\Big)^{(r+\mathbf{w}\cdot\mathbf{v})}$$
$$= L^{a\cdot ID+z_x}\Big((\prod_{i\in X} g^{(\varepsilon_{i0}\mathbf{e}+\varepsilon_{i1}M_1^{*\prime}+,\cdots,+\varepsilon_{im}M_m^{*\prime})\cdot\frac{\mathbf{u}}{b_i}})^{-ID'}\Big)^{(r+\mathbf{w}\cdot\mathbf{v})}$$
$$= L^{a\cdot ID+z_x}\prod_{i\in X}\prod_{j\in[1,n^*]}\Big(g^{\frac{a^j}{b_i}\cdot r}\prod_{k\in[1,n^*]}^{k\neq j}(g^{\frac{a^{q+j-k}}{b_i}})^{\mathbf{w}_k}\Big)^{exp_{i,j}}$$

where $exp_{i,j} = -ID' \cdot (\varepsilon_{i0}\mathbf{e}_j + \varepsilon_{i1}M^{*\prime}_{1j} + \cdots + \varepsilon_{im}M^{*\prime}_{mj})$.

**Challenge** $\mathcal{A}$ provides two equal length messages $\mathcal{M}_0$ and $\mathcal{M}_1$ to $\mathcal{B}$ as the challenge messages.

First, The simulator flips a coin $b$ and creates the ciphertext component $C = \mathcal{M}_b T \cdot e(g^s, g^{\alpha'})$. Then the simulator chooses random value $y'_2, y'_3, \cdots, y'_{n^*}$ and share the secret $s$ using the vector

$$\mathbf{v} = (s, y'_2, y'_3, \cdots, y'_{n^*}).$$

Next, it calculates

$$\lambda_k = \mathbf{v} \cdot (\varepsilon_{k0}\mathbf{e} + \varepsilon_{k1}M_1^{*\prime} + \varepsilon_{k2}M_2^{*\prime} + \cdots + \varepsilon_{km}M_m^{*\prime})$$

and generates the ciphertext component $C^*_k$ as follows

$$\hat{C}_a = \{C^*_{ka} = g^{b\cdot\lambda_k}, C'_{ka} = (g^{b^2\cdot ID'_a} h^b_{\rho(k)H})^{\lambda_k}\}_{k\in[1,l]},$$
$$\hat{C}_u = \{C^*_{ku} = g^{b\cdot\lambda_k}, C'_{ku} = (g^{b^2\cdot ID'_u} h^b_{\rho(k)})^{\lambda_k}\}_{k\in[1,l]},$$

$$C^*_{ka} = g^{a\cdot\sum_{i=1}^m(\varepsilon_{k0}\mathbf{e}+\varepsilon_{ki}M'_{1i})s} \cdot \prod_{j=2}^{n^*} g^{a\cdot\sum_{i=1}^m \varepsilon_{kj}M'_{ij}y'_j}.$$

$$C^*_{ku} = g^{a\cdot\sum_{i=1}^m(\varepsilon_{k0}\mathbf{e}+\varepsilon_{ki}M'_{1i})s} \cdot \prod_{j=2}^{n^*} g^{a\cdot\sum_{i=1}^m \varepsilon_{kj}M'_{ij}y'_j}.$$

For $k \in [1, l^*]$, we define $X_k$ as the set of the index $i$ in such that $\rho^*(i) = \rho^*(k)$. Finally, the simulator builds the ciphertext component $C'_{ka}$ and $C'_{ku}$ as follows

$$C'_{ka} = g^{\sum_{i=1}^m(\varepsilon_{k0}\mathbf{e}+\varepsilon_{ki}M'_{i1})z_{\rho^*(k)}s} \cdot g^{\sum_{j=2}^{n^*}\sum_{i=1}^m \varepsilon_{ki}M'_{ij}y'_j z_{\rho^*(k)}}$$
$$\cdot \prod_{i\in X_k}\prod_{\xi=1}^{n^*} g^{-(\sum_{j=1}^m \varepsilon_{ij}M'_{j\xi})(1+\sum_{j=1}^m \varepsilon_{kj}M'_{j1})ID'_a a^{\xi+1}s/b_i}$$
$$\cdot \prod_{i\in X_k}\prod_{\xi=1}^{n^*} g^{-(\sum_{j=1}^m \varepsilon_{ij}M'_{j\xi})(\sum_{l=2}^{n^*}\sum_{j=1}^m y'_l\varepsilon_{kj}M'_{jl})ID'_a a^{\xi+1}/b_i}.$$

$$C'_{ku} = g^{\sum_{i=1}^m(\varepsilon_{k0}\mathbf{e}+\varepsilon_{ki}M'_{i1})z_{\rho^*(k)}s} \cdot g^{\sum_{j=2}^{n^*}\sum_{i=1}^m \varepsilon_{ki}M'_{ij}y'_j z_{\rho^*(k)}}$$
$$\cdot \prod_{i\in X_k}\prod_{\xi=1}^{n^*} g^{-(\sum_{j=1}^m \varepsilon_{ij}M'_{j\xi})(1+\sum_{j=1}^m \varepsilon_{kj}M'_{j1})ID'_u a^{\xi+1}s/b_i}$$
$$\cdot \prod_{i\in X_k}\prod_{\xi=1}^{n^*} g^{-(\sum_{j=1}^m \varepsilon_{ij}M'_{j\xi})(\sum_{l=2}^{n^*}\sum_{j=1}^m y'_l\varepsilon_{kj}M'_{jl})ID'_u a^{\xi+1}/b_i}.$$

**Phase II** Same as phase I.

**Guess** $\mathcal{A}$ outputs a bit $b$'. If $b = b'$ then $\mathcal{B}$ guesses $T = e(g,g)^{sa^{q+1}}$ else guesses $T$ to be a random group element in $\mathbb{G}_T$. When $T = e(g,g)^{sa^{q+1}}$, $\mathcal{B}$ could simulate perfectly, therefore we have

$$Pr[\mathcal{B}(\mathbf{y}, T = e(g,g)^{sa^{q+1}}) = 0] = \frac{1}{2} + Adv_{\mathcal{A}}.$$

When $T$ is a random group element, the message $M_\beta$ is completely hidden from $\mathcal{A}$. Thus we have $Pr[\mathcal{B}(\mathbf{y}, T = R) = 0] = \frac{1}{2}$. If the adversary $\mathcal{A}$ could attack the OM-ID-HABE scheme with advantage which is a non-negligible faction of the security parameter, then $\mathcal{B}$'s advantage in the $M$-$q$-parallel BDHE problem is non-negligible as well, which contradicts **Theorem 1**. Thus, we could conclude that our OM-ID-HABE scheme is secure.

□