PAPER

# Attribute-Based Identification: Definitions and Efficient Constructions[*,**]

Hiroaki ANADA[†,††a)], Seiko ARITA[†b)], *Members*, Sari HANDA[†c)], *and* Yosuke IWABUCHI[†d)], *Nonmembers*

**SUMMARY** We propose a notion of attribute-based identification (ABID) in two flavors: prover-policy ABID (PP-ABID) and verifier-policy ABID (VP-ABID). In a PP-ABID scheme, a prover has an authorized access policy written as a boolean formula over attributes, while each verifier maintains a set of attributes. The prover is accepted when his access policy fits the verifier's set of attributes. In a VP-ABID scheme, a verifier maintains an access policy written as a boolean formula over attributes, while each prover has a set of authorized attributes. The prover is accepted when his set of attributes satisfies the verifier's access policy. Our design principle is first to construct key-policy and ciphertext-policy attribute-based key encapsulation mechanisms (KP-ABKEM and CP-ABKEM). Second, we convert KP-ABKEM and CP-ABKEM into challenge-and-response PP-ABID and VP-ABID, respectively, by encapsulation-and-decapsulation. There, we show that KP-ABKEM and CP-ABKEM only have to be secure against chosen-ciphertext attacks on one-wayness (OW-CCA secure) for the obtained PP-ABID and VP-ABID to be secure against concurrent man-in-the-middle attacks (cMiM secure). According to the design principle, we construct concrete KP-ABKEM and CP-ABKEM with the OW-CCA security by enhancing the KP-ABKEM of Ostrovsky, Sahai and Waters and CP-ABKEM of Waters, respectively. Finally, we obtain concrete PP-ABID and VP-ABID schemes that are proved to be selectively secure in the standard model against cMiM attacks.

***key words:*** *access policy, attribute, identification, KEM*

## 1. Introduction

An identification (ID) scheme allows a prover to convince a verifier that the prover certainly knows a secret key that corresponds to the matching public key. For example, the Schnorr protocol [11], [22] is widely recognized. In ID schemes, the public key limits the corresponding secret key, and also, the corresponding prover, uniquely.

In this paper, we will describe an *attribute-based identification (ABID)*. In an ABID scheme, each entity has credentials called *attributes*. On the other hand, an *access policy* is written as a boolean formula over those attributes. Then, a verifier can identify that a prover has a certain access policy that fits the verifier's set of attributes, or, in the dual flavor, a verifier can identify that a prover possesses a certain set of attributes that satisfies the verifier's access policy. Hence, ABID schemes can be considered as an expansion of the usual ID schemes.

However, ABID schemes are not a mere expansion, but have useful applications beyond those of the usual ID schemes. For example, the following scenarios of smart card systems motivate us to apply ABID.

**Functional Tickets.** Suppose that we are going to stay at a resort complex, a ski resort, for instance. We search Web sites or brochures for information about services: available dates, accommodation, ski lifts, restaurants in ski areas and hot springs around the areas. For each service, we usually buy a ticket, paying with money or using a credit card. However, acquiring many tickets and carrying a wallet is inconvenient, and therefore, it would be more convenient if we could gain access to these services by using only one smart card. In the smart card, a service authority writes an access policy in terms of the service names that we choose, for instance, [January 1 to 4, 2014] AND [[Hotel A] OR [Ski Lift AND [Day OR Night]] OR [Lunch OR Beer] OR [Hot Spring X]]. A *functional ticket* is a ticket embedded in a smart card, which functions as an access policy over services, as in this scenario. That is, the access policy is chosen according to our requirements. Then the access policy is written in the smart card by the service authority in a randomized way. On the other hand, a set of attributes is written in each ticket-checker in a straightforward way.

**Functional Gates.** Suppose that we have to design a security gate system for an office building in which different kinds of people work: employees of several companies holding many different positions, security guards, food service staffs, cleaning staffs and janitors. There are also many types of security gates to be designed: building entrances, intelligent elevators to limit available floors, company gates, common refreshment areas and room doors for the above staffs. In this case, one solution is to use smart cards and gates with sensors. That is, an authority issues each person a smart card in which a set of attribute data is written. Each gate decides whether to "pass" each person carrying a smart card according to the gate's access policy, for instance, [Year 2014] AND [[[Company A] AND [Manager]] OR [Security Guard]]. A *functional gate* is a gate that functions as an access policy over attributes of people, as in this scenario.

Manuscript received September 17, 2013.
Manuscript revised January 7, 2014.

[†] The authors are with Institute of Information Security, Yokohama-shi, 221-0835 Japan.

[††] The author is with Institute of Systems, Information Technologies and Nanotechnologies (ISIT), Fukuoka-shi, 814-0001 Japan.

[*] A preliminary version of this paper appeared in Proceedings of the 18th Australasian Conference on Information Security and Privacy - ACISP 2013, Lect. Notes Comput. Sci., vol.7959, pp.168–186, Springer-Verlag [2], under the same title.

[**] Supported by MEXT-Supported Program for the Strategic Research Foundation at Private Universities, 2011–2013.

a) E-mail: anada@isit.or.jp
b) E-mail: arita@iisec.ac.jp
c) E-mail: mgs125502@iisec.ac.jp
d) E-mail: mgs123101@iisec.ac.jp

DOI: 10.1587/transfun.E97.A.1086

That is, the access policy is chosen according to kinds of people which the gate should allow. Then the access policy is implemented in the gate in a straightforward way. On the other hand, a set of attributes is written in each smart card by the service authority in a randomized way.

### 1.1 Our Contributions

We propose a notion of attribute-based identification (ABID) that has two flavors corresponding to the scenarios: *prover-policy* ABID and *verifier-policy* ABID.

**Prover-Policy ABID.** In a prover-policy ABID scheme (PP-ABID, for short), a prover has his own authorized access policy, while each verifier maintains its set of attributes. Here, the access policy is described over attributes. Sending his access policy, each prover queries an authority for his secret key. Then, using this secret key, each prover can convince the verifier that his access policy fits the verifier's set of attributes. Our PP-ABID defined in this way realizes a functional ticket system.

**Verifier-Policy ABID.** In a verifier-policy ABID scheme (VP-ABID, for short), a verifier maintains its access policy, while each prover has his own authorized set of attributes. Here, the access policy is described over attributes. Sending his set of attributes, each prover queries an authority for his secret key. Then, using this secret key, each prover convinces the verifier that his set of attributes satisfy the verifier's access policy. Our VP-ABID defined in this way realizes a functional gate system.

**Attack and Security Analysis.** An adversary's objective is *impersonation*: giving a *target* set of attributes (or, a *target* access policy) to a verifier, the adversary tries to make the verifier accept him.

First, to reflect a *collusion attack* (that is, an attack launched by collecting secret keys that satisfy a condition), we consider an attack model in which an adversary issues key-extraction queries, as is the case for attribute-based encryptions [21], [23]. The condition is that the adversary cannot collect any secret key whose intrinsic access policy fits the target set of attributes (or, whose intrinsic set of attributes satisfies the target access policy).

Our main objective is to define a model of *concurrent man-in-the-middle attack (cMiM attack)* in the setting of ABID. "Concurrent" means that an adversary can invoke provers that have *different* secret keys corresponding to different access policies (or, different sets of attributes). The adversary interacts with these provers in an arbitrarily interleaved order of messages. Then, interacting with a verifier with the target set of attributes (or, the target access policy, respectively) the adversary tries to impersonate a prover. The concurrent attack modeled in this way is a real threat, especially to smart card systems. On the other hand, "man-in-the-middle (MiM)" means that an adversary stands between a prover and a verifier simultaneously. Typically, the adversary first receives a message from the verifier, and then, the adversary begins to interact with the prover *adaptively* to the verifier's message. The MiM attack and the cMiM attack modeled in this way are real threats, especially to network applications.

As is the case for usual ID schemes, *reset attacks* should be considered. In a reset attack, an adversary aborts an interaction at any point, and then rewinds the interaction back to any other point to start the interaction again. At that re-starting point, the adversary is allowed to change messages as long as the interaction remains valid (as captured by the word "reset"). Such a reset attack is a strong threat, not only to smart card systems [10] (including the functional tickets and functional gates described above) but also to virtual machine services in cloud computing [25]. As our contribution, an ABID constructed using our generic conversion becomes secure against the reset attacks in both senses of prover-resettable and verifier-resettable [10].

It is desirable that a verifier learns nothing about a prover more than that he belongs to the set of entities that have access policies that fit the verifier's set of attributes (or, belongs to the set of entities that possess sets of attributes that satisfy the verifier's access policy). In fact, by this property (*anonymity*), the prover's privacy is protected when using a functional ticket, as opposed to using a credit card the track of which is recorded. As our contribution, our concrete ABID in Sect. 5 possesses this anonymity.

**Design Principle.** First, we construct key-policy and ciphertext-policy attribute-based key encapsulation mechanisms (KP-ABKEM and CP-ABKEM [21], [23]). Second, we convert the KP-ABKEM and CP-ABKEM into challenge-and-response PP-ABID and VP-ABID, respectively, by encapsulation-and-decapsulation. There, we show that KP-ABKEM and CP-ABKEM only have to be secure against chosen-ciphertext attacks on one-wayness (OW-CCA secure) for the obtained PP-ABID and VP-ABID to be secure against cMiM attacks (cMiM secure). We stress that the security of indistinguishability against chosen-ciphertext attacks (IND-CCA security) is excessive, and OW-CCA security is enough for constructing a cMiM secure ABID.

**Concrete Constructions.** We construct KP-ABKEM and CP-ABKEM with the OW-CCA security from the KP-ABKEM of Ostrovsky, Sahai and Waters [21] (OSW, for short) and CP-ABKEM of Waters [23]. Their KEMs are secure in the indistinguishability game of chosen-plaintext attack (IND-CPA secure). Our strategy is to apply the algebraic trick of Boneh and Boyen [5] and Kiltz [17] to attain CCA security. The application is not a black-box because, in security proofs, we have to adapt the trick to both simulations of decapsulation oracle and key-generation oracle. Finally, our generic conversion turns obtained OW-CCA secure KP-ABKEM and CP-ABKEM into cMiM secure PP-ABID and VP-ABID, respectively.

**New Number Theoretic Assumptions.** For our efficient constructions, We introduce the Computational Bilinear Diffie-Hellman Assumption with Gap on Target Group and the Computational $q$-Parallel Bilinear Diffie-Hellman Exponent Assumption with Gap on Target Group. The validity of these assumptions is explained by the generic bilinear group model [6] in Appendix E.

### 1.2 Related Works

**Anonymous Credential System.** Our ABID is comparable with widely studied *anonymous credential system* (AC for short). AC originates from a blind signature scheme by Chaum [13], and now it has been accomplished as Microsoft's U-Prove [4]. AC was further advanced by Camenisch et al. with group signature scheme [3], has been accomplished as IBM's Idemix [12]. Those AC schemes are characterized by two functionalities of *untraceability* from shown credentials to provers and *unlinkability* between credential showings. In contrast, our ABID does not have the untraceability because, in a VP-ABID scheme, a secret-key issuing authority knows what credentials it authorizes and gives them as a secret key to a prover. Nevertheless, our concrete ABID schemes have a kind of unlinkability because it never leaks information to a verifier except a fact that the prover's set of attributes satisfies the verifier's access policy. Furthermore, our ABID can treat any access policy written as a boolean formula, which may includes any combination of AND gates and OR gates. This is not the case for AC in at least credential-showing process. And finally, a PP-ABID scheme, in which a prover is given a secret key reflecting not a set of attributes but an access policy, is a new notion that is not achieved by AC.

**Anonymous Deniable Predicate Authentication.** We should refer to the work of Yamada et al. [24] of verifiable predicate encryption. As an application, they provided an anonymous deniable message authentication scheme. It is possible to see their message authentication scheme as an ABID scheme of challenge-and-response type like our generic construction. However, it differs in objectives. We simply try to attain a 2-round, fast ABID, while they proposed a 6-round protocol for deniability. In addition, we provide more efficient concrete ABID schemes by applying the algebraic trick ([5], [17]); a kind of (non-black box) *direct chosen-ciphertext security* technique [7]. In contrast, they used their versatile generic transformation; as a result, it causes a longer secret key, a longer ciphertext and more computational costs for encryption and decryption than our strategy (the difference being about $O(\lambda^2)$ in the security parameter $\lambda$). This is because the generic transformation involves a verification key of a one-time signature.

**Attribute-Based Encryption.** After the pioneering works on attribute-based encryption [16] (ABE, for short), efficient KP-ABE and CP-ABE schemes are proposed [21], [23] that are IND-CPA secure in the standard model. These ABE schemes can naturally be considered as ABKEMs. Our concrete constructions of ABKEM in Sect. 5 are enhanced versions of these ABKEMs to attain the OW-CCA security. We choose OSW KP-ABKEM [21] and Waters CP-ABKEM [23], which can be seen as basic schemes in the growth of attribute-based encryption, as representative examples. Concerning a constraint of adversary's behavior, those KP-ABKEM and CP-ABKEM are secure in the game of selectively declared target of attribute set and access structure, respectively. Here "selective" means that an adversary declares the target before getting a public key. There are KP-ABKEM and CP-ABKEM which are secure in the game of adaptively declared target [18] and we discuss the adaptive case in Sect. 5.2.

**Identification Scheme from KEM.** Anada and Arita [1] proposed a design principle to obtain a cMiM secure ID scheme by constructing KEM. Their concrete ID scheme is more efficient than known Σ-protocol-based cMiM secure ID schemes, such as [15]. Our scheme can be seen as an attribute-based version of theirs.

**Attribute-Based Signature.** Maji et al. [19] introduced a notion of attribute-based signature (ABS). As they noted, it can be used as a verifier-policy ABID (sending a random message as a challenge and getting an attribute-based signature as a response). But the VP-ABID scheme from their Instantiation 1 needs $O(\lambda^2)$ bits for a pair of challenge and response messages, while the VP-ABID scheme from their Instantiation 2 needs $O(\lambda^2)$ bits for a public key. In contrast, our concrete VP-ABID scheme needs $O(\lambda)$ bits for both a pair of challenge and response messages and a public key.

### 1.3 Organization of the Paper

In Sect. 2, we survey the required terms. In Sect. 3, we define the notions of PP-ABID and VP-ABID, cMiM attacks and security against it. In Sect. 4, we provide generic conversions from KP-ABKEM to PP-ABID and from CP-ABKEM to VP-ABID. In Sect. 5, we construct concrete KP-ABKEM and CP-ABKEM. Finally, we obtain concrete PP-ABID and VP-ABID. In Sect. 6, we present the conclusions of our study. Because of space limitation, the case of PP-ABID is described in the main text and the case of VP-ABID is only shortly described in the Appendix.

## 2. Preliminaries

The security parameter is denoted by $\lambda$. A prime of bit length $\lambda$ is denoted by $p$. A multiplicative cyclic group of order $p$ is denoted by $\mathbb{G}$. The ring of the exponent domain of $\mathbb{G}$, which consists of integers from 0 to $p-1$ with modulo $p$ operation, is denoted by $\mathbb{Z}_p$. When an algorithm $A$ with input $a$ outputs $z$, we denote it as $z \leftarrow A(a)$. When $A$ with input $a$ and $B$ with input $b$ interact with each other and $B$ outputs $z$, we denote it as $z \leftarrow \langle A(a), B(b) \rangle$. When $A$ has oracle-access to $O$, we denote it as $A^{O}$. When $A$ has concurrent oracle-access to $n$ oracles $O_1, \ldots, O_n$, we denote it as $A^{O_i|_{i=1}^{n}}$. Here "concurrent" means that $A$ accesses to oracles in arbitrarily interleaved order of messages. A probability of an event E is denoted by Pr[E]. A probability of an event E on condition that events $\mathrm{E}_1, \ldots, \mathrm{E}_m$ occur in this order is denoted as $\Pr[\mathrm{E}_1; \cdots; \mathrm{E}_m : \mathrm{E}]$.

### 2.1 Access Structure

Let $\mathcal{U} = \{\chi_1, \ldots, \chi_u\}$ be an attribute universe, or simply set $\mathcal{U} = \{1, \ldots, u\}$. We must distinguish two cases: the case that $\mathcal{U}$ is small (i.e. $|\mathcal{U}| = u$ is bounded by some polynomial in $\lambda$) and the case that $\mathcal{U}$ is large (i.e. $u$ is not necessarily bounded). We assume the *small case* unless we state the large case explicitly. An *access structure*, which reflects a given access policy, is defined as a collection $\mathbb{A}$ of non-empty subsets of $\mathcal{U}$. That is, $\mathbb{A} \subset 2^{\mathcal{U}} \backslash \{\phi\}$. An access structure $\mathbb{A}$ is called *monotone* if for any $B \in \mathbb{A}$ and $B \subset C$, $C \in \mathbb{A}$ holds. We will consider in this paper only monotone access structures.

## 2.2 Linear Secret-Sharing Scheme

A secret-sharing scheme $\Pi$ over a set of parties $\mathcal{P}$ is called a linear secret-sharing scheme (LSSS) over $\mathbb{Z}_p$ ([9]), if $\Pi$ satisfies the following conditions.
1. The shares for each party form a vector over $\mathbb{Z}_p$.
2. There exist a matrix $M$ called the *share-generating matrix* for $\Pi$, of size $l \times n$, and a function $\rho$ which maps each row index $i$ of $M$ to a party in $\mathcal{P}$, $\rho : \{1, ..., l\} \to \mathcal{P}$.
To make shares for a secret $s \in \mathbb{Z}_p$, we first choose $n-1$ random values $v_2, \ldots, v_n \in \mathbb{Z}_p$ and form a vector $\vec{v} = (s, v_2, \ldots, v_n)$. For $i = 1$ to $l$, we calculate each share $\lambda_i = \vec{v} \cdot M_i$, where $M_i$ denotes the $i$-th row vector of $M$ and $\cdot$ denotes the formal inner product. The share $\lambda_i$ belongs to the party $\rho(i)$.

Looking at $\mathcal{P}$ as an attribute universe $\mathcal{U}$, $\Pi$ determines an access structure $\mathbb{A}$ as $(M, \rho)$ ([21], [23]). Suppose that an attribute set $S \subset \mathcal{U}$ satisfies $\mathbb{A}$ ($S \in \mathbb{A}$). Then, there exists a set of constants $\{\omega_i \in \mathbb{Z}_p; i \in \rho^{-1}(S)\}$ called *linear reconstruction constants* ([9]) that satisfies $\sum_{i \in \rho^{-1}(S)} \omega_i \lambda_i = s$. These constants $\{\omega_i\}_{i \in \rho^{-1}(S)}$ can be computed in time polynomial in the size of $M$. We denote the algorithm by $\mathrm{Recon}(\rho^{-1}(S), M)$. If $S$ does not satisfy $\mathbb{A}$ ($S \notin \mathbb{A}$), then no such constants $\{\omega_i\}_{i \in \rho^{-1}(S)}$ exist, but instead, there is a vector $\vec{w} = (w_1, \ldots, w_n) \in \mathbb{Z}_p^n$ such that $w_1 = 1$ and $\vec{w} \cdot M_i = 0$ for all $i \in \rho^{-1}(S)$. $\vec{w}$ also can be computed in time polynomial in the size of $M$ ([23]).

## 2.3 Key-Policy Attribute-Based KEM

**Scheme.** A key-policy ABKEM, `KP-ABKEM`, consists of four probabilistic polynomial time algorithms (PPTAs, for short): (Setup, KeyGen, Encap, Decap).
**Setup**$(\lambda, \mathcal{U}) \to$ **(PK, MSK).** Setup takes as input the security parameter $\lambda$ and the attribute universe $\mathcal{U}$. It returns a public key PK and a master secret key MSK.
**KeyGen(PK, MSK,** $\mathbb{A}$**)** $\to$ **SK**$_{\mathbb{A}}$**.** A key generation algorithm KeyGen takes as input the public key PK, the master secret key MSK and an access structure $\mathbb{A}$. It returns a secret key $\mathrm{SK}_{\mathbb{A}}$ that corresponds to $\mathbb{A}$.
**Encap(PK,** $S$**)** $\to (\kappa, \psi)$**.** Encap takes as input the public key PK and an attribute set $S$. It returns a random KEM key $\kappa$ and its encapsulation $\psi$ (we also call it a ciphertext). We denote the set of all possible output $(\kappa, \psi)$ of Encap(PK, $S$) by [Encap(PK, $S$)]. If $(\tilde{\kappa}, \tilde{\psi}) \in$ [Encap(PK, $S$)], then $(\tilde{\kappa}, \tilde{\psi})$ is called *consistent* and otherwise, *inconsistent*.
**Decap(PK, SK**$_{\mathbb{A}}$**,** $\psi$**)** $\to$ $\hat{\kappa}$**.** Decap takes as input the public key PK, an encapsulation $\psi$ and a secret key $\mathrm{SK}_{\mathbb{A}}$. It returns a decapsulation result $\hat{\kappa}$ of $\psi$ under $\mathrm{SK}_{\mathbb{A}}$. We demand correctness of `KP-ABKEM` that for any $\lambda$ and $\mathcal{U}$, and if $S \in \mathbb{A}$, then $\Pr[(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathrm{Setup}(\lambda, \mathcal{U}); \mathrm{SK}_{\mathbb{A}} \leftarrow \mathrm{KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A}); (\kappa, \psi) \leftarrow \mathrm{Encap}(\mathrm{PK}, S); \hat{\kappa} \leftarrow \mathrm{Decap}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}}, \psi) : \kappa = \hat{\kappa}] = 1$.
**Chosen-Ciphertext Attack on One-Wayness of KP-ABKEM and Security.** The following experiment $\mathbf{Exprmt}^{\text{ow-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda, \mathcal{U})$ of an adversary $\mathcal{A}$ defines the game of chosen-ciphertext attack on one-wayness of `KP-ABKEM` (the OW-CCA game).

```
$\mathbf{Exprmt}^{\text{ow-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda, \mathcal{U}) : //\mathrm{Adaptive}S^*$
  $(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{Setup}(\lambda, \mathcal{U})$
  $S^* \leftarrow \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot),\mathcal{DEC}(\mathrm{PK},\mathrm{SK}_{\cdot},\cdot)}(\mathrm{PK}, \mathcal{U})$
  $(\kappa^*, \psi^*) \leftarrow \mathbf{Encap}(\mathrm{PK}, S^*)$
  $\hat{\kappa^*} \leftarrow \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot),\mathcal{DEC}(\mathrm{PK},\mathrm{SK}_{\cdot},\cdot)}(\psi^*)$
  If$\hat{\kappa^*} = \kappa^*$ then Return WIN else Return LOSE
```

In the experiment, $\mathcal{A}$ issues two types of queries. One is key-extraction queries to the key-generation oracle $\mathcal{KG}$. Giving an access structure $\mathbb{A}_i$, $\mathcal{A}$ queries $\mathcal{KG}(\mathrm{PK}, \mathrm{MSK}, \cdot)$ for the secret key $\mathrm{SK}_{\mathbb{A}_i}$. Another is decapsulation queries to the decapsulation oracle $\mathcal{DEC}$. Giving a pair $(\mathbb{A}_j, \psi_j)$ of an access structure and an encapsulation, $\mathcal{A}$ queries $\mathcal{DEC}(\mathrm{PK}, \mathrm{SK}_{\cdot}, \cdot)$ for the decapsulation result $\hat{\kappa}_j$. Here an attribute set $S_j$, which is used to generate a ciphertext, is included in $\psi_j$. When $S_j \notin \mathbb{A}_j$, $\hat{\kappa}_j = \perp$ is replied to $\mathcal{A}$.

The attribute set $S^*$ declared by $\mathcal{A}$ is called a *target attribute set*. The encapsulation $\psi^*$ is called a *challenge ciphertext*. Two restrictions are imposed on $\mathcal{A}$ concerning $S^*$ and $\psi^*$. In key-extraction queries, each access structure $\mathbb{A}_i$ must satisfy $S^* \notin \mathbb{A}_i$. In decapsulation queries, each pair $(\mathbb{A}_j, \psi_j)$ must satisfy $S^* \notin \mathbb{A}_j \vee \psi_j \neq \psi^*$. Both types of queries are at most $q_{\mathrm{k}}$ and $q_{\mathrm{d}}$ times in total, respectively, which are bounded by a polynomial in $\lambda$.

The *advantage* of $\mathcal{A}$ over `KP-ABKEM` in the OW-CCA game is defined as [†]

$$\mathbf{Adv}^{\text{ow-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda) \stackrel{\text{def}}{=} \Pr[\mathbf{Exprmt}^{\text{ow-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda, \mathcal{U}) \text{ returns } \textsc{Win}].$$

`KP-ABKEM` is called *secure against chosen-ciphertext attacks on one-wayness* if, for any PPT $\mathcal{A}$ and for any $\mathcal{U}$, $\mathbf{Adv}^{\text{ow-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda)$ is negligible in $\lambda$.
**Selective Security.** In the *selective game on a target attribute set* (OW-sel-CCA game), $\mathcal{A}$ declares $S^*$ *before* $\mathcal{A}$ receives PK. The following experiment $\mathbf{Exprmt}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda, \mathcal{U})$ defines the selective game.

```
$\mathbf{Exprmt}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda, \mathcal{U}) : //\mathrm{Selective}S^*$
  $(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{Setup}(\lambda, \mathcal{U})$
```

[†] More rigorously, we have to treat the right-hand side as $\sup_{|\mathcal{U}|:\mathrm{poly}(\lambda)}\{\Pr[\mathbf{Exprmt}^{(\text{game})}_{(\text{PPTA}),(\text{scheme})}(\lambda, \mathcal{U}) \text{ returns } \textsc{Win}]\}$.

```
$S^* \leftarrow \mathcal{A}(\lambda, \mathcal{U})$
$(\kappa^*, \psi^*) \leftarrow \mathbf{Encap}(\mathrm{PK}, S^*)$
$\hat{\kappa^*} \leftarrow \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot),\mathcal{DEC}(\mathrm{PK},\mathrm{SK}.,\cdot)}(\mathrm{PK}, \psi^*)$
If $\hat{\kappa^*} = \kappa^*$ then Return Win else Return Lose
```

The *advantage* in the OW-sel-CCA game is defined as

$$\mathbf{Adv}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda) \overset{\text{def}}{=} \Pr[\mathbf{Exprmt}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda, \mathcal{U}) \text{ returns } \textsc{Win}].$$

KP-ABKEM is called *selectively secure against chosen-ciphertext attacks on one-wayness* if, for any PPT $\mathcal{A}$ and for any $\mathcal{U}$, $\mathbf{Adv}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda)$ is negligible in $\lambda$.

### 2.4 Bilinear Map

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order $p$. We call $\mathbb{G}$ a source group and $\mathbb{G}_T$ a target group. Let $g$ be a generator of $\mathbb{G}$ and $e$ be a bilinear map, $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. The map $e$ satisfies the following conditions:
1. Bilinearity: $\forall u, v \in \mathbb{G}, \forall a, b \in \mathbb{Z}_p, e(u^a, v^b) = e(u, v)^{ab}$,
2. Non-degeneracy: $e(g, g) \neq$ (the identity element of $\mathbb{G}_T$).
Groups and a bilinear map are generated by a PPT algorithm **Grp** on input $\lambda$: $(p, \mathbb{G}, \mathbb{G}_T, g, e) \leftarrow \mathbf{Grp}(\lambda)$. We assume that the group operation in $\mathbb{G}$ and $\mathbb{G}_T$ and the bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ are computable in time polynomial in $\lambda$.

### 2.5 Computational Bilinear Diffie-Hellman Assumption with Gap on Target Group.

We introduce in this paper a new number theoretic assumption, which we call the *Computational Bilinear Diffie-Hellman Assumption with Gap on Target Group*. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map. Let $a, b, c \in \mathbb{Z}_p, a, b, c \neq 0$, be chosen at random. Put $A := g^a, B := g^b, C := g^c$. We denote $e(g, g)$ as $g_T$. Then our new assumption says it is at most with a negligible probability in $\lambda$ that, for any PPT algorithm $\mathcal{B}$ given input $(g, A, B, C)$, to output $Z = g_T^{abc} \in \mathbb{G}_T$, even with the aid of the decisional Diffie-Hellman oracle for $\mathbb{G}_T$: $\mathcal{DDH}_{\mathbb{G}_T}(g_T, \cdot, \cdot, \cdot)$. Here a tuple $(g_T, g_T^{z_1}, g_T^{z_2}, g_T^{z_3}) \in \mathbb{G}_T^4$ is called a Diffie-Hellman tuple (in $\mathbb{G}_T$) if $z_1 z_2 = z_3$. The oracle $\mathcal{DDH}_{\mathbb{G}_T}$ returns True or False according to whether an input tuple is a Diffie-Hellman tuple or not, respectively. The probability for $\mathcal{B}$ to output $g_T^{abc}$ is denoted as $\mathbf{Adv}^{\text{c-bdh-gap}}_{\mathcal{B},(e,\mathbb{G},\mathbb{G}_T)}(\lambda)$ (the advantage of $\mathcal{B}$ in the computational BDH game with gap on $\mathbb{G}_T$). Note that the above assumption is in general stronger than the *Gap Bilinear Diffie-Hellman Assumption* [8]. The validity of the assumption is explained by the generic bilinear group model [6] in Appendix E.

### 2.6 Target Collision Resistant Hash Functions

Target collision resistant (TCR) hash functions [20] are treated as a family $Hfam_\lambda = \{H_\mu\}_{\mu \in HKey_\lambda}$. Here $HKey_\lambda$ is a hash key space and $H_\mu$ is a function from $\{0, 1\}^*$ to $\{0, 1\}^\lambda$. We may assume that $H_\mu$ is from $\{0, 1\}^*$ to $\mathbb{Z}_p$. Given a PPT algorithm $\mathcal{CF}$, a collision finder, we consider the following experiment (the game of target collision resistance).

```
$\mathbf{Exprmt}^{\text{tcr}}_{\mathcal{CF},Hfam_\lambda}(\lambda)$
  $m^* \leftarrow \mathcal{CF}(\lambda), \mu \leftarrow HKey_\lambda, m \leftarrow \mathcal{CF}(\mu)$
  If $m^* \neq m \wedge H_\mu(m^*) = H_\mu(m)$ then Return Win
  else Return Lose.
```

Then we define *$\mathcal{CF}$'s advantage over Hfam$_\lambda$ in the game of target collision resistance* as

$$\mathbf{Adv}^{\text{tcr}}_{\mathcal{CF},Hfam_\lambda}(\lambda) \overset{\text{def}}{=} \Pr[\mathbf{Exprmt}^{\text{tcr}}_{\mathcal{CF},Hfam_\lambda}(\lambda) \text{ returns } \textsc{Win}].$$

We say that $Hfam_\lambda$ is *a TCR function family* if, for any PPT algorithm $\mathcal{CF}$, $\mathbf{Adv}^{\text{tcr}}_{\mathcal{CF},Hfam_\lambda}(\lambda)$ is negligible in $\lambda$. TCR hash function families can be constructed from a one-way function [20].

## 3. Attribute-Based Identification

In this section, we define a notion of *prover-policy* attribute-based identification (PP-ABID), a concurrent man-in-the-middle attack on PP-ABID and security against it. The case of *verifier-policy* ABID goes in a dual manner to PP-ABID on an access structure $\mathbb{A}$ and an attribute set $S$ and is described in Appendix A.

### 3.1 Prover-Policy ABID

**Scheme.** PP-ABID consists of four PPT algorithms: (Setup, KeyGen, P, V).
**Setup$(\lambda, \mathcal{U}) \to$ (PK, MSK).** Setup takes as input the security parameter $\lambda$ and the attribute universe $\mathcal{U}$. It outputs a public key PK and a master secret key MSK.
**KeyGen(PK, MSK, $\mathbb{A}$) $\to$ SK$_\mathbb{A}$.** A key-generation algorithm KeyGen takes as input the public key PK, the master secret key MSK and an access structure $\mathbb{A}$. It outputs a secret key SK$_\mathbb{A}$ corresponding to $\mathbb{A}$.
**P(PK, SK$_\mathbb{A}$) and V(PK, $S$).** P and V are interactive algorithms called a *prover* and a *verifier*, respectively. P takes as input the public key PK and the secret key SK$_\mathbb{A}$. Here the secret key SK$_\mathbb{A}$ is given to P by an authority that runs KeyGen(PK,MSK,$\mathbb{A}$). V takes as input the public key PK and an attribute set $S$. P is provided V's attribute set $S$ by the first round. P and V interact with each other for some, at most constant rounds. Then, V finally returns its decision bit $b$. $b = 1$ means that V *accepts* P in the sense P has a secret key SK$_\mathbb{A}$ such that $S$ satisfies $\mathbb{A}$. $b = 0$ means that V *rejects* P. We demand correctness of PP-ABID that for any $\lambda$ and $\mathcal{U}$, and if $S \in \mathbb{A}$, then $\Pr[(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathrm{Setup}(\lambda, \mathcal{U}); \mathrm{SK}_\mathbb{A} \leftarrow \mathrm{KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A}); b \leftarrow \langle \mathrm{P}(\mathrm{PK}, \mathrm{SK}_\mathbb{A}), \mathrm{V}(\mathrm{PK}, S)\rangle : b = 1] = 1$.
**Concurrent Man-in-the-Middle Attack on PP-ABID and Security.** An adversary $\mathcal{A}$'s objective is impersonation. $\mathcal{A}$ tries to make a verifier V accept with an attribute set $S^*$. The following experiment $\mathbf{Exprmt}^{\text{cmim}}_{\mathcal{A},\texttt{PP-ABID}}(\lambda, \mathcal{U})$ of an adversary $\mathcal{A}$ defines the game of concurrent man-in-the-middle

attack (cMiM attack, for short) on PP-ABID.

```
$\mathbf{Exprmt}^{\mathrm{cmim}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda, \mathcal{U})$ : //Adaptive$S^*$
  $(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{Setup}(\lambda, \mathcal{U})$
  $S^* \leftarrow \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot),\mathbf{P}_j(\mathrm{PK},\mathrm{SK}_\cdot)|_{j=1}^{q'_\mathrm{p}}}(\mathrm{PK}, \mathcal{U})$
  $b \leftarrow \langle \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot),\mathbf{P}_j(\mathrm{PK},\mathrm{SK}_\cdot)|_{j=q'_\mathrm{p}}^{q_\mathrm{p}}}, \mathbf{V}(\mathrm{PK}, S^*)\rangle$
  If $b = 1$ then Return Win else Return Lose
```

In the experiment, $\mathcal{A}$ issues key-extraction queries to the key-generation oracle $\mathcal{KG}$. Giving an access structure $\mathbb{A}_i$, $\mathcal{A}$ queries $\mathcal{KG}(\mathrm{PK}, \mathrm{MSK}, \cdot)$ for the secret key $\mathrm{SK}_{\mathbb{A}_i}$. We do not require any two input, $\mathbb{A}_{i_1}$ and $\mathbb{A}_{i_2}$, to be distinct. In addition, the adversary $\mathcal{A}$ invokes provers $\mathrm{P}_j(\mathrm{PK}, \mathrm{SK}_\cdot)$, $j = 1, \ldots, q'_\mathrm{p}, \ldots, q_\mathrm{p}$, by giving an access structure $\mathbb{A}_j$ of $\mathcal{A}$'s choice. Acting as a verifier with an attribute set $S_j$, $\mathcal{A}$ interacts with each $\mathrm{P}_j$.

The attribute set $S^*$ declared by $\mathcal{A}$ is called a *target attribute set*. Two restrictions are imposed on $\mathcal{A}$ concerning $S^*$. In key-extraction queries, each access structure $\mathbb{A}_i$ must satisfy $S^* \notin \mathbb{A}_i$. In interactions with each prover, $S^* \notin \mathbb{A}_j$, or, every transcript of messages of a whole interaction with a prover $\mathrm{P}_j(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}_j})$ must not be equal to a transcript of messages of a whole interaction with a verifier $\mathrm{V}(\mathrm{PK}, S^*)$ (that is, a mere *relay of messages* is prohibited in the game of man-in-the-middle attack). The number of key-extraction queries and the number of invoked provers are at most $q_\mathrm{k}$ and $q_\mathrm{p}$ in total, respectively, which are bounded by a polynomial in $\lambda$.

The *advantage* of $\mathcal{A}$ over PP-ABID in the game of cMiM attack is defined as

$$\mathbf{Adv}^{\mathrm{cmim}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda) \overset{\mathrm{def}}{=} \Pr[\mathbf{Exprmt}^{\mathrm{cmim}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda, \mathcal{U}) \text{ returns Win}].$$

PP-ABID is called *secure against cMiM attacks* if, for any PPT $\mathcal{A}$ and for any attribute universe $\mathcal{U}$, $\mathbf{Adv}^{\mathrm{cmim}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda)$ is negligible in $\lambda$.

**Selective Security.** In the *selective game on a target attribute set* (the game of sel-cMiM attack), $\mathcal{A}$ declares $S^*$ *before* $\mathcal{A}$ receives PK. The following experiment $\mathbf{Exprmt}^{\mathrm{sel\text{-}cmim}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda, \mathcal{U})$ defines the selective game.

```
$\mathbf{Exprmt}^{\mathrm{sel\text{-}cmim}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda, \mathcal{U})$ : //Seletive$S^*$
  $(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{Setup}(\lambda, \mathcal{U})$
  $S^* \leftarrow \mathcal{A}(\lambda, \mathcal{U})$
  $b \leftarrow \langle \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot),\mathbf{P}_j(\mathrm{PK},\mathrm{SK}_\cdot)|_{j=1}^{q_\mathrm{p}}}(\mathrm{PK}), \mathbf{V}(\mathrm{PK}, S^*)\rangle$
  If $b = 1$ then Return Win else Return Lose
```

The *advantage* in the game of sel-cMiM attack is defined as

$$\mathbf{Adv}^{\mathrm{sel\text{-}cmim}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda) \overset{\mathrm{def}}{=} \Pr[\mathbf{Exprmt}^{\mathrm{sel\text{-}cmim}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda, \mathcal{U}) \text{ returns Win}].$$

PP-ABID is called *selectively secure against cMiM attacks* if, for any PPT $\mathcal{A}$ and for any $\mathcal{U}$, $\mathbf{Adv}^{\mathrm{sel\text{-}cmim}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda)$ is negligible in $\lambda$.

**Anonymity** Anonymity that is discussed briefly in Introduction is formalized as follows. Consider the following experiment $\mathbf{Exprmt}^{\mathrm{anonym}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda, \mathcal{U})$. (In the experiment, an adversary $\mathcal{A}$ interacts with $\mathbf{P}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}_b})$ as a verifier with $S^*$.)

```
$\mathbf{Exprmt}^{\mathrm{anonym}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda, \mathcal{U})$ :
  $(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{Setup}(\lambda, \mathcal{U}), (\mathbb{A}_0, \mathbb{A}_1, S^*) \leftarrow \mathcal{A}(\mathrm{PK})$
        s.t. $(S^* \in \mathbb{A}_0 \wedge S^* \in \mathbb{A}_1) \vee (S^* \notin \mathbb{A}_0 \wedge S^* \notin \mathbb{A}_1)$
  $\mathrm{SK}_{\mathbb{A}_0} \leftarrow \mathbf{KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A}_0)$
  $\mathrm{SK}_{\mathbb{A}_1} \leftarrow \mathbf{KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A}_1)$
  $b \leftarrow \{0, 1\}, \hat{b} \leftarrow \mathcal{A}^{\mathbf{P}(\mathrm{PK},\mathrm{SK}_{\mathbb{A}_b})}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}_0}, \mathrm{SK}_{\mathbb{A}_1})$
  If $b = \hat{b}$ Return Win else Return Lose
```

We say that PP-ABID have *anonymity* if, for any PPT $\mathcal{A}$ and for any $\mathcal{U}$, the following advantage of $\mathcal{A}$ is negligible in $\lambda$.

$$\mathbf{Adv}^{\mathrm{anonym}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda) \overset{\mathrm{def}}{=}$$
$$|\Pr[\mathbf{Exprmt}^{\mathrm{anonym}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda, \mathcal{U}) \text{ returns Win}] - 1/2|.$$

## 4. Generic Conversions from ABKEM to ABID

In this section, we provide a generic conversion from a key-policy ABKEM to a prover-policy ABID. The conversion yields a challenge-and-response protocol of encapsulation-and-decapsulation. We show that KP-ABKEM only has to be OW-CCA secure for the obtained PP-ABID to be cMiM secure. A generic conversion from a ciphertext-policy ABKEM to a verifier-policy ABID is described in Appendix C.

### 4.1 Generic Conversion from KP-ABKEM to PP-ABID

Let KP-ABKEM= (KEM.Setup, KEM.KeyGen, KEM.Encap, KEM.Decap) be a KP-ABKEM. Then PP-ABID= (Setup, KeyGen, Encap, Decap) is obtained as a challenge-and-response protocol of encapsulation-and-decapsulation. Figure 1 shows this conversion. Setup of PP-ABID uses KEM.Setup. KeyGen of PP-ABID uses KEM.KeyGen. The verifier V, given a public key PK and an attribute set $S$ as input, invokes the encapsulation algorithm KEM.Encap on $(\mathrm{PK}, S)$. V gets a return $(\kappa, \psi)$. V sends the encapsulation $\psi$ to the prover $P$ as a challenge message. $P$, given a public key PK and the secret key $\mathrm{SK}_\mathbb{A}$ as input, and receiving $\psi$ as a message, invokes the decapsulation algorithm KEM.Decap on $(\mathrm{PK}, \mathrm{SK}_\mathbb{A}, \psi)$. P gets a return $\hat{\kappa}$. P sends the decapsulation $\hat{\kappa}$ to V as a response message. Finally, V, receiving $\hat{\kappa}$ as a message, verifies whether $\hat{\kappa}$ is equal to $\kappa$. If so, then V returns 1 and otherwise, 0.

**Theorem 1:** If KP-ABKEM is OW-CCA secure, then the derived PP-ABID is cMiM secure. More precisely, for any given PPT adversary $\mathcal{A}$ on PP-ABID in the game of cMiM attack, and for any given attribute universe $\mathcal{U}$, there exists a PPT adversary $\mathcal{B}$ on KP-ABKEM in the OW-CCA game that satisfies the following tight reduction.

```
Setup(λ, U):                                 | P(PK, SK_A):                               V(PK, S):
  (PK, MSK) ← KEM.Setup(λ, U)                 |                                              (κ, ψ) ← KEM.Encap(PK, S)
  Return (PK, MSK)                            |   Receiving ψ as input:           ←ψ——      Send ψ to P
KeyGen(PK, MSK, A):                           |   κ̂ ← KEM.Decap(PK, SK_A, ψ)
  SK_A ← KEM.KeyGen(PK, MSK, A)               |   Send κ̂ to V                     ——κ̂→      Receiving κ̂ as input:
  Return (SK_A)                               |                                              If κ̂ = κ then b := 1 else b := 0, Return b
```

**Fig. 1** A generic conversion from KP-ABKEM to PP-ABID.

$$\mathbf{Adv}^{\text{cmim}}_{\mathcal{A},\texttt{PP-ABID}}(\lambda) \leqslant \mathbf{Adv}^{\text{ow-cca}}_{\mathcal{B},\texttt{KP-ABKEM}}(\lambda).$$

*Proof.* Employing any given PPT cMiM adversary $\mathcal{A}$ on PP-ABID in Theorem 1, we construct a PPT OW-CCA adversary $\mathcal{B}$ on KP-ABKEMas follows.

```
B(PK, U) : //Adaptive S*
//Setup
  Initialize inner state, Invoke A on (PK, U)
//Answering A's Queries
  When A issues a key-ext. query for 𝔸
    SK_𝔸 ← KG(PK, MSK, 𝔸),  Reply SK_𝔸 to A
  When A sends a chal. msg. (𝔸, ψ) to P
    κ̂ ← DEC(PK, SK_𝔸, ψ),  Send κ̂ to A as the res. msg.
  When A outputs a target attribute set S*
    Output S* as its target attribute set
    Receive ψ* as a chal. ciphertext
  When A queries V for a chal. msg.
    Send ψ* to A as a chal. msg.
  When A sends the res. msg. κ̂* to V
    Return κ̂*
```

On input (PK, $\mathcal{U}$), $\mathcal{B}$ initializes its inner state and invokes $\mathcal{A}$ on (PK, $\mathcal{U}$). When $\mathcal{A}$ issues a key-extraction query for $\mathbb{A}$, $\mathcal{B}$ queries its key-generation oracle $\mathcal{KG}$(PK, MSK, ·) for the answer for $\mathbb{A}$ and gets a reply $\text{SK}_{\mathbb{A}}$. $\mathcal{B}$ reply $\text{SK}_{\mathbb{A}}$ to $\mathcal{A}$. When $\mathcal{A}$ sends a challenge message $(\mathbb{A}, \psi)$ to a prover **P**, $\mathcal{B}$ queries its decapsulation oracle $\mathcal{DEC}$(PK, SK., ·) for the answer for $(\mathbb{A}, \psi)$ and gets a reply $\hat{\kappa}$. $\mathcal{B}$ reply $\hat{\kappa}$ to $\mathcal{A}$. When $\mathcal{A}$ outputs a target attribute set $S^*$, $\mathcal{B}$ output $S^*$ as its target attribute set. Then $\mathcal{B}$ receives a challenge ciphertext $\psi^*$ from its challenger. When $\mathcal{A}$ queries **V** for a challenge message, $\mathcal{B}$ sends $\psi^*$ to $\mathcal{A}$ as a challenge message. When $\mathcal{A}$ sends the response message $\hat{\kappa}^*$ to **V**, $\mathcal{B}$ returns $\hat{\kappa}^*$ as its guess.

The view of $\mathcal{A}$ in $\mathcal{B}$ is the same as the real view of $\mathcal{A}$. If $\mathcal{A}$ wins, then $\mathcal{B}$ wins. Hence the inequality in Theorem 1 holds.

□

### 4.2 Discussion

**Selective Security.** In the game of selective $S^*$, $\mathcal{B}$ is constructed as follows.

```
B(PK, U) : //Selective S*
//Setup
  Initialize inner state, Invoke A on (λ, U)
//Answering A's Queries
  S* ← A(λ, U)
  Output S* as its target attribute set
  Receive ψ* as a chal. ciphertext,  Give PK to A
  When A issues a key-ext query for 𝔸
    SK_𝔸 ← KG(PK, MSK, 𝔸)
  When A sends a chal. msg. (𝔸, ψ) to P
    κ̂ ← DEC(PK, SK_𝔸, ψ),  Send κ̂ to A as the res. msg.
  When A queries V for a chal. msg.
    Send ψ* to A as a chal. msg.
  When A sends the res. msg. κ̂* to V
    Return κ̂*
```

Then the inequality of advantages becomes

$$\mathbf{Adv}^{\text{sel-cmim}}_{\mathcal{A},\texttt{PP-ABID}}(\lambda) \leqslant \mathbf{Adv}^{\text{ow-sel-cca}}_{\mathcal{B},\texttt{KP-ABKEM}}(\lambda).$$

**Resettable Security.** We note that the derived PP-ABID is prover-resettable in the sense in [10] because underlying KP-ABKEM has the OW-CCA security. PP-ABID is also verifier-resettable because PP-ABID consists of two rounds interaction.

## 5. Concrete Constructions of ABKEM

In this section, we construct a concrete KP-ABKEM that is OW-sel-CCA secure. Using the algebraic trick of Boneh and Boyen [5] and Kiltz [17], we build an enhanced version, KP-ABKEM, of the KP-ABKEM of Ostrovsky, Sahai and Waters [21] (OSW, for short). Then we obtain our concrete PP-ABID by applying the generic conversion. (Our concrete CP-ABKEM and VP-ABID is described in Appendix F).

### 5.1 Our Enhanced OSW KP-ABKEM and PP-ABID

The construction of our concrete KP-ABKEM is described in Fig. 2. We only explain the enhanced part from the original [21]. We indicate the part of the original scheme by the index: $_{\text{cpa}}$. In Setup, a second component $\alpha_2 \in \mathbb{Z}_p$ is added to the master secret key $\text{MSK}_{\text{cpa}}$. Also, the corresponding $Y_2 := e(g, g)^{\alpha_2 b}$ and a hash key $\eta$ is added to the public key $\text{PK}_{\text{cpa}}$. In KeyGen, components in $\text{SK}_{\text{cpa},\mathbb{A}}$ are doubled reflecting the index 2 (but randomness is chosen independently of index 1). So computational cost for

```
Setup($\lambda, \mathcal{U}$):
  $(p, \mathbb{G}, \mathbb{G}_{\mathrm{T}}, g, e) \leftarrow \mathbf{Grp}(\lambda)$
  For $x = 1$ to $u$: $T_x \leftarrow \mathbb{G}$
  $b \leftarrow \mathbb{Z}_p, B := g^b, \alpha_1, \alpha_2 \leftarrow \mathbb{Z}_p$
  $Y_1 := e(g,g)^{\alpha_1 b}, Y_2 := e(g,g)^{\alpha_2 b}$
  $\eta \leftarrow HKey_\lambda$
  $\mathrm{PK} := (g, T_1, \ldots, T_u, B, Y_1, Y_2, \eta)$
  $\mathrm{MSK} := (\alpha_1, \alpha_2)$
  Return $(\mathrm{PK}, \mathrm{MSK})$
```

```
KeyGen($\mathrm{PK}, \mathrm{MSK}, \mathbb{A} = (M, \rho)$):
  For $k = 1, 2$:
      For $j = 2$ to $n$: $v_{k,j} \leftarrow \mathbb{Z}_p$
  For $k = 1, 2$:
      $\vec{v}_k := (\alpha_k, v_{k,2}, \ldots, v_{k,n})$
  For $i = 1$ to $l$: $\lambda_{k,i} := \vec{v}_k \cdot M_i$
  For $k = 1, 2$: For $i = 1$ to $l$:
      $r_{k,i} \leftarrow \mathbb{Z}_p,\ K_{k,i} := B^{\lambda_{k,i}} T_{\rho(i)}^{r_{k,i}}$,
                  $L_{k,i} := g^{r_{k,i}}$
  $\mathrm{SK}_{\mathbb{A}} := (((K_{k,i}, L_{k,i});$
              $i = 1, \ldots, l); k = 1, 2)$
  Return $\mathrm{SK}_{\mathbb{A}}$
```

```
Encap($\mathrm{PK}, S$):
  $s \leftarrow \mathbb{Z}_p, C' = g^s$
For $x \in S : C_x = T_x^s$
  $\psi_{\mathrm{cpa}} := (S, C', (C_x; x \in S))$
  $\tau \leftarrow H_\eta(\psi_{\mathrm{cpa}})$
   For $k = 1, 2$: $\kappa_k := Y_k^s;\ d := \kappa_1^\tau \kappa_2$
  $(\kappa, \psi) := (\kappa_1, (\psi_{\mathrm{cpa}}, d))$
  Return $(\kappa, \psi)$
```

```
Decap($\mathrm{PK}, \mathrm{SK}_{\mathbb{A}}, \psi$):
  If $S \notin \mathbb{A}$ Return $\hat{\kappa} :=\perp$
  else
   $\{\omega_i; i \in \rho^{-1}(S)\}$
        $\leftarrow \mathbf{Reconst}(\rho^{-1}(S), M)$
  For $k = 1, 2$:
   $\hat{\kappa_k} := \prod_{i \in \rho^{-1}(S)}$
  $(e(K_{k,i}, C')/e(L_{k,i}, C_{\rho(i)}))^{\omega_i}$
   $\tau \leftarrow H_\eta(\psi_{\mathrm{cpa}})$
   If $\hat{\kappa_1}^\tau \hat{\kappa_2} \neq d$ then $\hat{\kappa} :=\perp$ else $\hat{\kappa} := \hat{\kappa_1}$
   Return $\hat{\kappa}$
```

**Fig. 2** Our concrete KP-ABKEM (an enhanced OSW KP-ABKEM).

KeyGen is doubled. In Encap, a temporal KEM key $\kappa_2$ is generated in the same way as $\kappa_1$. Next, a hash value $\tau \leftarrow H_\eta(\psi_{\mathrm{cpa}})$ and a *check sum* $d := \kappa_1^\tau \kappa_2$ are computed. Then $(\kappa, \psi) := (\kappa_1, (\psi_{\mathrm{cpa}}, d))$ is a new KEM key and encapsulation. In Decap, first, $\mathrm{Decap}_{\mathrm{cpa}}$ is executed twice for index 1 and 2 to yield $\hat{\kappa_1}$ and $\hat{\kappa_2}$. Then, whether $\psi_{\mathrm{cpa}}$ is a consistent ciphertext and $(e(g,g), Y_1^\tau Y_2, e(C', g), d)$ is a Diffie-Hellman tuple are verified. These two conditions are verified by one equation $\hat{\kappa_1}^\tau \hat{\kappa_2} = d$, though the verification equation overlooks inconsistent $\psi_{\mathrm{cpa}}$ only with a negligible probability. Finally, $\hat{\kappa} := \hat{\kappa_1}$ is returned only when the verification equation holds.

**Theorem 2:** If the computational bilinear Diffie-Hellman assumption with gap on target group holds, and an employed hash function family has target collision resistance, then our KP-ABKEM is OW-sel-CCA secure. More precisely, for any given PPT adversary $\mathcal{A}$ on KP-ABKEM in the OW-sel-CCA game and for any given attribute universe $\mathcal{U}$, there exist a PPT adversary $\mathcal{B}$ on $(e, \mathbb{G}, \mathbb{G}_{\mathrm{T}})$ in the computational BDH game with gap on $\mathbb{G}_{\mathrm{T}}$ and a PPT target collision finder $\mathcal{CF}$ on $Hfam_\lambda$ that satisfy the following tight reduction.

$$\mathbf{Adv}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda) \leqslant \mathbf{Adv}^{\text{c-bdh-gap}}_{\mathcal{B},(e,\mathbb{G},\mathbb{G}_{\mathrm{T}})}(\lambda) + \mathbf{Adv}^{\text{tcr}}_{\mathcal{CF},Hfam_\lambda}(\lambda).$$

*Proof.* Using any given PPT OW-sel-CCA adversary $\mathcal{A}$ as a subroutine, we construct a PPT solver $\mathcal{B}$ of the problem of the computational bilinear Diffie-Hellman assumption with gap on target group, as follows.

**Set up.** $\mathcal{B}$ is given a random instance of the problem, $g, A = g^a, B = g^b, C = g^c$, as input. $\mathcal{B}$ initializes its inner state. $\mathcal{B}$ chooses an attribute universe $\mathcal{U} = \{1, \ldots, u\}$ at random. $\mathcal{B}$ invokes $\mathcal{A}$ on input $(\lambda, \mathcal{U})$.

In return, $\mathcal{B}$ receives a target attribute set $S^* \subset \mathcal{U}$ from $\mathcal{A}$, For each $x = 1, \ldots, u$, $\mathcal{B}$ puts each component $T_x$ of PK as

If $x \in S^*$ then $t_x \leftarrow \mathbb{Z}_p, T_x := g^{t_x}$

else $\theta_x, \eta_x \leftarrow \mathbb{Z}_p$ s.t. $(\theta_x \neq 0), T_x := B^{\theta_x} g^{\eta_x}$.

Here, in else case, we have implicitly set $t_x := b\theta_x + \eta_x$. $\mathcal{B}$ sets $Y_1 := e(A, B) = e(g,g)^{ab}$ and $\mathrm{PK}_{\mathrm{cpa}} := (g, T_1, \ldots, T_u, B, Y_1)$. Here we have implicitly set $\alpha_1 := a$.

A challenge ciphertext is computed as follows (we implicitly set $s^* = c$):

$$\psi^*_{\mathrm{cpa}} := (S^*, C'^* = g^{s^*} := C, (C^*_x := C^{t_x}, x \in S^*)).$$

Then a public key PK and a whole challenge ciphertext $\psi^*$ is computed as

$$\eta \leftarrow HKey_\lambda,\ \tau^* \leftarrow H_\eta(\psi^*_{\mathrm{cpa}}),\ \mu \leftarrow \mathbb{Z}_p,\ Y_2 := e(B, g)^\mu / Y_1^{\tau^*},$$
$$\mathrm{PK} := (\mathrm{PK}_{\mathrm{cpa}}, Y_2, \eta),\ d^* := e(B, C'^*)^\mu,\ \psi^* := (\psi^*_{\mathrm{cpa}}, d^*).$$

Here we have an implicit relation $\alpha_2 b = b\mu - \alpha_1 b\tau^*, b \neq 0$. That is,

$$\alpha_2 = \mu - \alpha_1 \tau^*. \tag{1}$$

$\mathcal{B}$ inputs $(\mathrm{PK}, \psi^*)$ to $\mathcal{A}$.

**Answering $\mathcal{A}$'s Queries. (1) Key-Extraction Queries.** When $\mathcal{A}$ issues a key-extraction query for an attribute set $\mathbb{A} = (M, \rho)$, where $M$ is of size $l \times n$, $\mathcal{B}$ has to reply a corresponding secret key $\mathrm{SK}_{\mathbb{A}}$.

$\mathcal{B}$ computes a vector $\vec{w} = (w_1, \ldots, w_n) \in \mathbb{Z}_p^n$ such that $w_1 = 1$ and for all $i \in \rho^{-1}(S^*), \vec{w} \cdot M_i = 0$. Note here that $S^* \notin \mathbb{A}$, so such $\vec{w}$ surely exists. $\mathcal{B}$ chooses random values $u_{1,1}, \ldots, u_{1,n} \in \mathbb{Z}_p$ and put $\vec{u}_1 := (u_{1,1}, \ldots, u_{1,n})$. Then we implicitly set $\vec{v}_1 := \vec{u}_1 + (a - u_{1,1})\vec{w}$.

Here for each $i = 1, \ldots, l$, $\mathcal{B}$ can compute $g^{\lambda_{1,i}}$ as $g^{M_i \cdot \vec{v}_1} = g^{M_i \cdot (\vec{u}_1 - u_{1,1}\vec{w})} A^{M_i \cdot \vec{w}}$. Then $\mathcal{B}$ computes the index 1 components of $\mathrm{SK}_S$ as

```
For $i = 1$ to $l$ :
  If $i \in \rho^{-1}(S^*)$ then $r_{1,i} \leftarrow \mathbb{Z}_p$
    $K_{1,i} := B^{M_i \cdot \vec{u}_1} T_{\rho(i)}^{r_{1,i}}, L_{1,i} := g^{r_{1,i}}$
  else $r'_{1,i} \leftarrow \mathbb{Z}_p$
    $K_{1,i} := (g^{\lambda_{1,i}})^{-\eta_{\rho(i)}/\theta_{\rho(i)}} (B^{\theta_{\rho(i)}} g^{\eta_{\rho(i)}})^{r'_{1,i}}, L_{1,i} := (g^{\lambda_{1,i}})^{-1/\theta_{\rho(i)}} g^{r'_{1,i}}$.
```

Here, in else case, we implicitly set $r_{1,i} := r'_{1,i} - \lambda_{1,i}/\theta_{\rho(i)}$.

Now $\mathcal{B}$ has to compute the index 2 components $K_{2,i}, L_{2,i}$ for $i = 1, \ldots, l$. To do so, $\mathcal{B}$ chooses random values $u_{2',1}, \ldots, u_{2',n}, r_{2',i}$(or $r'_{2',i}$) $\in \mathbb{Z}_p$ and computes $K_{2',i}, L_{2',i}, i = 1, \ldots, l$ *just in the same way* as to the index 1. Then $\mathcal{B}$ converts them as follows:

$$K_{2,i} := B^{M_{i,1}\mu} (K_{2',i})^{-\tau^*}, L_{2,i} := (L_{2',i})^{-\tau^*}, i = 1, \ldots, l.$$

Then $\mathcal{B}$ replies $\mathrm{SK}_{\mathbb{A}} = (((K_{k,i}, L_{k,i}); i = 1, \ldots, l); k = 1, 2)$ to $\mathcal{A}$.

**(2) Decapsulation Queries.** When $\mathcal{A}$ issues a decapsulation query for $(\mathbb{A}, \psi = (\psi_{\mathrm{cpa}}, d))$ (where $\psi_{\mathrm{cpa}}$ is about $S$), $\mathcal{B}$ has to reply the decapsulation $\hat{\kappa}$ to $\mathcal{A}$. To do so, $\mathcal{B}$ computes as follows. (Note that the oracle $\mathcal{DDH}_{\mathbb{G}_{\mathrm{T}}}$ is accessed.)

```
If $S \notin \mathbb{A}$ then $\hat{\kappa} := \perp$
else If $\mathbf{Verify}(\mathrm{PK}_{\mathrm{cpa}}, \psi_{\mathrm{cpa}}) = \textsc{False}$ then $\hat{\kappa} := \perp$
    else $\tau \leftarrow H_\eta(\psi_{\mathrm{cpa}})$
        If $\mathcal{DDH}_{\mathbb{G}_{\mathrm{T}}}(e(g,g), Y_1^\tau Y_2, e(C',g), d) = \textsc{False}$ then $\hat{\kappa} := \perp$
        else If $\tau = \tau^*$ then Abort //Call this case $\textsc{Abort}$
            else $\hat{\kappa} := (d/e(B,C')^\mu)^{1/(\tau-\tau^*)}$
```

where **Verify** is the following PPT algorithm to check consistency of $\psi_{\mathrm{cpa}}$:

```
$\mathbf{Verify}(\mathrm{PK}_{\mathrm{cpa}}, \psi_{\mathrm{cpa}})$ :
  For $x \in S$ : If $e(T_x, C') \neq e(C_x, g)$ then Return $\textsc{False}$
  Return $\textsc{True}$.
```

**Guess.** When $\mathcal{A}$ returns $\mathcal{A}$'s guess $\hat{\kappa^*}$, $\mathcal{B}$ returns $Z := \hat{\kappa^*}$ as $\mathcal{B}$'s guess.

$\mathcal{B}$ can perfectly simulate the real view of $\mathcal{A}$ until the case Abort happens. To see why, we prove the following claims.

**Claim 1:** The reply $\mathrm{SK}_{\mathbb{A}}$ to a key-extraction query is a perfect simulation.

*Proof.* First, the index 1 components $K_{1,i}, L_{1,i}, i = 1, \ldots, l$ are correctly distributed, as is proved in the original work of Ostrovsky, Sahai and Waters [21]. By the construction, the index 2' components $K_{2',i}, L_{2',i}, i = 1, \ldots, l$ are distributed in the same way as the index 1 (but with independent randomness).

For the index 2 components $K_{2,i}, L_{2,i}, \quad i = 1, \ldots, l$, note that we have implicitly set $v_{2,j} := v_{2',j}(-\tau^*), j = 2, \ldots, n, r_{2,i} = r_{2',i}(-\tau^*), i = 1, \ldots, l$. Using another implicit relation (1) together, we get

$$\begin{aligned}
K_{2,i} &= B^{M_{i,1}\mu}(K_{2',i})^{-\tau^*} \\
&= B^{M_{i,1}\mu}(B^{\lambda_{2',i}} T_{\rho(i)}^{r_{2',i}})^{-\tau^*} \\
&= g^{bM_{i,1}\mu}(g^{bM_i \cdot (\alpha_1, v_{2',2}, \ldots, v_{2',n})} T_{\rho(i)}^{r_{2',i}})^{-\tau^*} \\
&= g^{b(M_{i,1}(\mu - \alpha_1\tau^*) + M_{i,2}v_{2',2}(-\tau^*) + \cdots + M_{i,n}v_{2',n}(-\tau^*))} T_{\rho(i)}^{r_{2',i}(-\tau^*)} \\
&= g^{b(M_{i,1}\alpha_2 + M_{i,2}v_{2,2} + \cdots + M_{i,n}v_{2,n})} T_{\rho(i)}^{r_{2,i}} \\
&= B^{M_i \cdot \vec{v}_2} T_{\rho(i)}^{r_{2,i}}, \\
L_{2,i} &= (L_{2',i})^{-\tau^*} = (g^{r_{2',i}})^{-\tau^*} = g^{r_{2,i}}, i = 1, \ldots, l.
\end{aligned}$$

□

**Claim 2:** The reply $\hat{\kappa}$ to a decapsulation query is a simulation that is computationally indistinguishable from a real, until the case Abort happens.

*Proof.* First note that the honest decapsulation algorithm **Decap** overlooks an inconsistent ciphertext $\psi$ only with negligible probability $1/p$ in its verification process by the verification equation ($\hat{\kappa_1}^\tau \hat{\kappa_2} = d$) (the reason is the same as in Sect. 5.2). On the other hand, the solver $\mathcal{B}$ never overlook inconsistency of $\psi$ because it uses subroutine **Verify** which checks consistency of $\psi_{\mathrm{cpa}}$ perfectly and hence, in combination with the verification equation, consistency of $\psi$ perfectly. Then the computationally bounded adversary $\mathcal{A}$ cannot distinguish the difference between the response of $\mathcal{DEC}$ and $\mathcal{B}$.

Hence it is enough to prove that if $\mathcal{DDH}_{\mathbb{G}_{\mathrm{T}}}(e(g,g), Y_1^\tau Y_2, e(C',g), d) = \textsc{True}$, then $\hat{\kappa} = Y_1^s$. This is deduced by using the implicit relations (1), as follows.

$$\begin{aligned}
\hat{\kappa} &= ((Y_1^\tau Y_2)^s / e(B,C')^\mu)^{1/(\tau-\tau^*)} = (e(g,g)^{(\alpha_1\tau + \alpha_2 - \mu)bs})^{1/(\tau-\tau^*)} \\
&= (e(g,g)^{\alpha_1(\tau-\tau^*)bs})^{1/(\tau-\tau^*)} = Y_1^s.
\end{aligned}$$

□

**Claim 3:** The challenge ciphertext $\psi^* = (\psi^*_{\mathrm{cpa}}, d^*)$ is correctly distributed.

*Proof.* Using the implicit relations (1), a direct calculation shows;

$$d^* = e(B, C'^*)^\mu = e(g,g)^{bs^*(\alpha_1\tau^* + \alpha_2)} = (Y_1^{\tau^*} Y_2)^{s^*}.$$

Hence $\psi^* = (\psi^*_{\mathrm{cpa}}, d^*)$ is legitimate and correctly distributed. □

Now we are ready to evaluate the advantage of $\mathcal{B}$ in the OW-sel-CCA game. First, the following claim holds.

**Claim 4:** The probability that Abort occurs is negligible in $\lambda$. More precisely, the following equality holds: $\Pr[\textsc{Abort}] = \mathbf{Adv}^{\mathrm{tcr}}_{\mathcal{CF}, Hfam_\lambda}(\lambda)$.

*Proof.* We construct a PPT target collision finder $\mathcal{CF}$ by using $\mathcal{A}$ as subroutine as follows. (Remark that the case Collision is defined. $\mathbf{Encap}_{\mathrm{cpa}}(\mathrm{PK}_{\mathrm{cpa}}, S)$ is a subalgorithm of $\mathbf{Encap}(\mathrm{PK}, S)$ that outputs $(\kappa, \psi_{\mathrm{cpa}})$.)

```
$\mathcal{CF}(\lambda)$ :
//Setup
  Initialize its inner state
  Choose an attribute universe $\mathcal{U} = \{1, \ldots, u\}$ at random,
      where $u$ is bounded by a polynomial in $\lambda$
  $S^* \leftarrow \mathcal{A}(\lambda, \mathcal{U}), (p, \mathbb{G}, \mathbb{G}_{\mathrm{T}}, g, e) \leftarrow \mathbf{Grp}(\lambda)$,
  $T_1, \ldots, T_u \leftarrow \mathbb{G}, b, \alpha_1 \leftarrow \mathbb{Z}_p, B := g^b, Y_1 := e(g,g)^{\alpha_1 b}$
  $\mathrm{PK}_{\mathrm{cpa}} := (g, T_1, \ldots, T_u, B, Y_1)$
  $(\kappa^*, \psi^*_{\mathrm{cpa}}) \leftarrow \mathbf{Encap}_{\mathrm{cpa}}(\mathrm{PK}_{\mathrm{cpa}}, S^*)$
  Output $\psi^*_{\mathrm{cpa}}$//a Target
  Receive $\eta \in HKey_\lambda, \tau^* \leftarrow H_\eta(\psi^*_{\mathrm{cpa}})$
  $\alpha_2 \leftarrow \mathbb{Z}_p, \mathrm{PK} := (\mathrm{PK}_{\mathrm{cpa}}, Y_2 := e(g,g)^{\alpha_2 b}, \eta), \mathrm{MSK} := (\alpha_1, \alpha_2)$
  $d^* := e(B, C'^*)^{\alpha_1\tau^*} e(B, C'^*)^{\alpha_2}, \psi^* := (\psi^*_{\mathrm{cpa}}, d^*)$
  Give $(\mathrm{PK}, \psi^*)$ to $\mathcal{A}$
//Answering $\mathcal{A}$'s Queries
  When $\mathcal{A}$ issues a key-extraction query by $\mathbb{A}$
    Reply $\mathrm{SK}_{\mathbb{A}} \leftarrow \mathbf{KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A})$ to $\mathcal{A}$
  When $\mathcal{A}$ issues a decapsulation query by $(\mathbb{A}, \psi = (\psi_{\mathrm{cpa}}, d))$
```

```
$\mathbf{P}(\mathrm{PK} = (g, T_1, \ldots, T_u, B, Y_1, Y_2, \eta),$
  $\mathrm{SK}_{\mathbb{A}} = (((K_{k,i} = B^{\lambda_{k,i}} T_{\rho(i)}^{r_{k,i}}, L_{k,i} = g^{r_{k,i}});$
                  $i = 1, \ldots, l); k = 1, 2))$:

$\mathbf{V}(\mathrm{PK}, S)$:
  $s \leftarrow \mathbb{Z}_p, C' := g^s$, For $x \in S : C_x := T_x^s$
  $\psi_{\mathrm{cpa}} := (S, C', (C_x; x \in S)), \tau \leftarrow H_\eta(\psi_{\mathrm{cpa}})$
  For $k = 1, 2$: $\kappa_k := Y_k^s$; $d := \kappa_1^\tau \kappa_2, (\kappa, \psi) := (\kappa_1, (\psi_{\mathrm{cpa}}, d))$
  Send $\psi$ to $\mathbf{P}$                                   $\xleftarrow{\psi}$

  Receiving $\psi$ as input;
  If $S \notin \mathbb{A}$ then $\hat{\kappa} := \perp$
  else $\tau \leftarrow H_\eta(\psi_{\mathrm{cpa}})$
   $\{\omega_i; i \in \rho^{-1}(S)\} \leftarrow \mathbf{Reconst}(\rho^{-1}(S), M)$
   For $k = 1, 2$ :
    $\hat{\kappa_k} := \prod_{i \in \rho^{-1}(S)} (e(K_{K,i}, C') / e(L_{K,i}, C_{\rho(i)}))^{\omega_i}$
   If $\hat{\kappa_1}^\tau \hat{\kappa_2} \neq d$ then $\hat{\kappa} := \perp$ else $\hat{\kappa} := \hat{\kappa_1}$
  Send $\hat{\kappa}$ to $\mathbf{V}$                            $\xrightarrow{\hat{\kappa}}$

  Receiving $\hat{\kappa}$ as input;
  If $\hat{\kappa} = \kappa$ then $b := 1$ else $b := 0$, Return $b$
```

**Fig. 3** An interaction of our concrete PP-ABID.

```
  $\tau \leftarrow H_\eta(\psi_{\mathrm{cpa}})$
  If $e(B, C')^{\alpha_1 \tau + \alpha_2} \neq d$ then $\hat{\kappa} := \perp$
  else
   If $\psi_{\mathrm{cpa}} \neq \psi^*_{\mathrm{cpa}} \wedge \tau = \tau^*$
    then Return $\psi_{\mathrm{cpa}}$//the CaseCollision
   else
    $\mathrm{SK}_{\mathbb{A}} \leftarrow \mathbf{KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A})$
    Reply $\hat{\kappa} \leftarrow \mathbf{Decap}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}}, \psi)$ to $\mathcal{A}$
//Stop
 When $\mathcal{A}$ returns its guess $\hat{\kappa^*}$, Stop.
```

By the construction, the view of $\mathcal{A}$ in $\mathcal{CF}$ is the same as the view of $\mathcal{A}$ in $\mathcal{B}$. (Remark that if $e(B, C')^{\alpha_1 \tau + \alpha_2} = d$ holds, then $\psi_{\mathrm{cpa}} \neq \psi^*_{\mathrm{cpa}}$ holds automatically, as is explained below.)

Let us evaluate the probability in Claim 4. If, in addition to $\tau = \tau^*$, it occurred that $\psi_{\mathrm{cpa}} = \psi^*_{\mathrm{cpa}}$ (and hence $S = S^*, C' = C'^*$), then it would occur that $d = d^*$. So $\psi = \psi^*$ holds. This is because the following two tuples are equal DH tuples: $(e(g, g), Y_1^\tau Y_2, e(C', g), d)$ and $(e(g, g), Y_1^{\tau^*} Y_2, e(C'^*, g), d^*)$. So both $S^* \in \mathbb{A}$ and $\psi = (\psi_{\mathrm{cpa}}, d) = (\psi^*_{\mathrm{cpa}}, d^*) = \psi^*$ would occur. This is ruled out by definition in decapsulation query.

Therefore, we have $\psi_{\mathrm{cpa}} \neq \psi^*_{\mathrm{cpa}}$. That is, $\mathcal{CF}$ gets a collision:

$$\psi_{\mathrm{cpa}} \neq \psi^*_{\mathrm{cpa}} \wedge H_\eta(\psi_{\mathrm{cpa}}) = \tau = \tau^* = H_\eta(\psi^*_{\mathrm{cpa}}).$$

Therefore, we get $\Pr[\textsc{Abort}] = \Pr[\textsc{Collision}]$. Substituting the advantage, we obtain

$$\Pr[\textsc{Abort}] = \mathbf{Adv}^{\mathrm{tcr}}_{\mathcal{CF}, Hfam_\lambda}(\lambda).$$

□

By definition, $\mathcal{A}$ wins in the OW-sel-CCA game if and only if $\hat{\kappa}^*$ is correctly guessed. That is, $\hat{\kappa^*} = Y_1^{s^*} = e(g, g)^{abs^*} = e(g, g)^{abc}$. This is the definition that $\mathcal{B}$ succeeds in computing the answer for the given instance $(g, A, B, C)$.

Therefore, the probability that $\mathcal{B}$ wins is equal to the probability that $\mathcal{A}$ wins and Abort never occurs. So we have:

$$\begin{aligned}\Pr[\mathcal{B} \text{ wins}] &= \Pr[(\mathcal{A} \text{ wins}) \wedge (\neg \textsc{Abort})] \\ &\geqslant \Pr[\mathcal{A} \text{ wins}] - \Pr[\textsc{Abort}].\end{aligned}$$

Substituting advantages and using the equality in Claim 4, we have:

$$\mathbf{Adv}^{\text{c-bdh-gap}}_{\mathcal{B},(e,\mathbb{G},\mathbb{G}_{\mathrm{T}})}(\lambda) \geqslant \mathbf{Adv}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda) - \mathbf{Adv}^{\mathrm{tcr}}_{\mathcal{CF}, Hfam_\lambda}(\lambda).$$

This is what we should prove in Theorem 5

□

Applying the generic conversion in 4.1 (from KP-ABKEM to PP-ABID) to our concrete KP-ABKEM above, we obtain a concrete PP-ABID. Figure 3 shows the interaction of the obtained PP-ABID.

**Theorem 3** (Corollary to Theorem 1 and 2): Our PP-ABID is selectively secure against cMiM attacks under the same assumptions. More precisely,

$$\mathbf{Adv}^{\text{sel-cmim}}_{\mathcal{A},\texttt{PP-ABID}}(\lambda) \leqslant \mathbf{Adv}^{\text{c-bdh-gap}}_{\mathcal{B},(e,\mathbb{G},\mathbb{G}_{\mathrm{T}})}(\lambda) + \mathbf{Adv}^{\mathrm{tcr}}_{\mathcal{CF}, Hfam_\lambda}(\lambda).$$

Figure 3 shows an interaction of our PP-ABID.

### 5.2 Discussion

**Anonymity.** Our concrete PP-ABID possesses the anonymity in the sense in Sect. 3.1 because the response message in the protocol of PP-ABID is a result of decapsulation; the result does not depend on prover's access structure $\mathbb{A}$ itself, but depends on whether $\mathbb{A}$ fits verifier's attribute set $S$ or not ($S \in \mathbb{A}$ or not). That is, when a verifier or a cheating adversary receives a result of decapsulation, the information of prover's access structure $\mathbb{A}$ never leaks. (Our concrete VP-ABID in Appendix F also has anonymity.)

**Large Universe Case.** If the attribute universe $\mathcal{U}$ is large, we have to modify our concrete schemes to make security reductions in time polynomial in $\lambda$. As is proposed by Waters [23], we use for $x \in \mathcal{U}$ a hashed value $H(x)$ instead of $T_x$ (and hence $T_x$ is removed from PK). Although the resulting schemes are proved to be secure only in the *random oracle model*, we do not have to rewrite the public key PK each time when a new attribute $x$ is added.

**Exiting the Gap Assumption.** Instead of using the aid of oracle $\mathcal{DDH}_{\mathbb{G}_{\mathrm{T}}}$, we can use the twin Diffie-Hellman trapdoor test of Cash, Kiltz and Shoup [14] in the security proofs. At the price of that, the resulting schemes have a secret key of double size and decapsulation costs twice as much.

**Security against Adaptive Target.** Our concrete PP-ABID (and our concrete VP-ABID in Appendix F) is secure in the game of selectively declared target. When we consider the security game in the *random oracle model*, we can apply our CCA secure enhancing technique to the dual system encryptions of Lewko, Okamoto, Sahai, Takashima and Waters [18], which are CPA secure against adaptive target. We remark that our CCA secure enhancing technique is a kind of (non-black box) *direct chosen-ciphertext security* technique [7].

## 6. Conclusions

We introduced the notion of attribute-based identification (ABID) and defined prover-policy ABID scheme and verifier-policy ABID scheme. We provided a design principle: construct a one-way CCA secure ABKEM and use it as challenge-and-response ABID. The obtained ABID is secure against concurrent man-in-the-middle attacks. We actually constructed concrete KP-ABKEM and CP-ABKEM with OW-CCA security and obtained concrete PP-ABID and VP-ABID schemes. We proposed that functional tickets and functional gates are useful applications of PP-ABID and VP-ABID, respectively.

## Acknowledgements

We sincerely appreciate helpful comments about anonymous credential systems from Prof. Mitchel of University of London at ACISP 2013, Brisbane. We also appreciate anonymous reviewers for comments about our new number theoretic assumptions and generic bilinear group model.

### References

[1] H. Anada and S. Arita, "Identification schemes from key encapsulation mechanisms," Proc. AFRICACRYPT 2011, Dakar, Senegal, July 2011, Lect. Notes Comput. Sci., vol.6737, pp.59–76, Springer-Verlag, Heidelberg.

[2] H. Anada, S. Arita, S. Handa, and Y. Iwabuchi, "Attribute-based identification: Definitions and efficient constructions," Proc. ACISP 2013, Brisbane, Australia, July 2013, Lect. Notes Comput. Sci., vol.7959, pp.168–186, Springer-Verlag, Heidelberg.

[3] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik, "A practical and provably secure coalition-resistant group signature scheme," Proc. CRYPTO 2000, Santa Barbara, California, USA, Aug. 2000, Lect. Notes Comput. Sci., vol.1880, pp.255–270, Springer-Verlag, Heidelberg.

[4] S. Brands, Rethinking public key infrastructures and digital certificates, MIT Press, 2000.

[5] D. Boneh and X. Boyen, "Efficient selective-ID secure identity-based encryption without random oracles," Proc. EUROCRYPT 2004, Interlaken, Switzerland, May 2004, Lect. Notes Comput. Sci., vol.3027, pp.223–238, Springer-Verlag, Heidelberg.

[6] D. Boneh, X. Boyen, and E.J. Goh, "Hierarchical identity based encryption with constant size ciphertext," Proc. EUROCRYPT 2005, Aarhus, Denmark, May 2005, Lect. Notes Comput. Sci., vol.3494, pp.440–456, Springer-Verlag, Heidelberg. Full version available at IACR Cryptology ePrint Archive, 2005/015, http://eprint.iacr.org/

[7] X. Boyen, Q. Mei, and B. Waters, "Direct chosen ciphertext security from identity-based techniques," Proc. ACM Conference on Computer and Communications Security, pp.320–329, 2005. Full version available at IACR Cryptology ePrint Archive, 2005/288, http://eprint.iacr.org/

[8] J. Baek, R. Safavi-Naini, and W. Susilo, "Efficient multi-receiver identity-based encryption and its application to broadcast encryption," Proc. PKC 2005, Les Diablerets, Switzerland, Jan. 2005, Lect. Notes Comput. Sci., vol.3386, pp.380–397, Springer-Verlag, Heidelberg.

[9] A. Beimel, "Secure schemes for secret sharing and key distribution," Ph.D. thesis, Israel Institute of Technology, Technion, Haifa, Israel, 1996.

[10] M. Bellare, M. Fischlin, S. Goldwasser, and S. Micali, "Identification protocols secure against reset attacks," Proc. EUROCRYPT 2001, Innsbruck, Austria, May 2001, Lect. Notes Comput. Sci., vol.2045, pp.495–511, Springer-Verlag, Heidelberg.

[11] M. Bellare and A. Palacio, "GQ and schnorr identification schemes: Proofs of security against impersonation under active and concurrent attacks," Proc. CRYPTO 2002, Santa Barbara, California, USA, Aug. 2002, Lect. Notes Comput. Sci., vol.2442, pp.162–177, Springer-Verlag, Heidelberg.

[12] J. Camenisch and A. Lysyanskaya, "An efficient system for non-transferable anonymous credentials with optional anonymity revocation," Proc. EUROCRYPT 2001, Innsbruck, Austria, May 2001, Lect. Notes Comput. Sci., vol.2045, pp.93–118, Springer-Verlag, Heidelberg.

[13] D. Chaum, "Blind signatures for untraceable payments," Proc. CRYPTO'82, Santa Barbara, California, USA, Aug. 1982, pp.199–203, Plenum Press, New York.

[14] D. Cash, E. Kiltz, and V. Shoup, "The twin Diffie-Hellman problem and applications," Proc. EUROCRYPT 2008, Istanbul, Turkey, April 2008, Lect. Notes Comput. Sci., vol.4965, pp.127–145, Springer-Verlag, Heidelberg. Full version available at Cryptology ePrint Archive, 2008/067, http://eprint.iacr.org/

[15] R. Gennaro, "Multi-trapdoor commitments and their applications to non-malleable protocols," Proc. CRYPTO 2004, Santa Barbara, California, USA, Aug. 2004, Lect. Notes Comput. Sci., vol.3152, pp.220–236, Springer-Verlag, Heidelberg.

[16] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," Proc. ACM Conference on Computer and Communications Security, pp.89–98, 2006.

[17] E. Kiltz, "Chosen-ciphertext security from tag-based encryption," Proc. TCC 2006, Tokyo, Japan, March 2013, Lect. Notes Comput. Sci., vol.3876, pp.581–600, Springer-Verlag, Heidelberg.

[18] A.B. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, "Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption," Proc. EUROCRYPT 2010, French Riviera, May-June 2010, Lect. Notes Comput. Sci., vol.6110, pp.62–91, Springer-Verlag, Heidelberg. Full version available at IACR Cryptology ePrint Archive, 2010/110, http://eprint.iacr.org/

[19] H.K. Maji, M. Prabhakaran, and M. Rosulek, "Attribute-based signatures," Proc. CT-RSA 2011, San Francisco, CA, USA, Feb. 2011, Lect. Notes Comput. Sci., vol.6558, pp.376–392, Springer-Verlag, Heidelberg. Full version available at Cryptology ePrint Archive, 2010/595, http://eprint.iacr.org/

[20] M. Naor and M. Yung, "Universal one-way Hash functions and their cryptographic applications," Proc. 21st Symposium on Theory of Computing 1989, pp.33–43, Association for Computing Machinery, New York.

[21] R. Ostrovsky, A. Sahai, and B. Waters, "Attribute-based encryption with non-monotonic access structures," Proc. ACM Conference on Computer and Communications Security, pp.195–203, 2007.

[22] C.P. Schnorr, "Efficient identification and signatures for smart

cards," Proc. CRYPTO'89, Santa Barbara, California, USA, Aug. 1989, Lect. Notes Comput. Sci., vol.435, pp.239–252, Springer-Verlag, Heidelberg.

[23] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient and provably secure realization," Proc. PKC 2011, Taormina, Italy, March 2011, Lect. Notes Comput. Sci., vol.6571, pp.53–70, Springer-Verlag, Heidelberg. Full version available at IACR Cryptology ePrint Archive, 2008/290, http://eprint.iacr.org/

[24] S. Yamada, N. Attrapadung, B. Santoso, J.C.N. Schuldt, G. Hanaoka, and N. Kunihiro, "Verifiable predicate encryption and applications to CCA security and anonymous predicate authentication," Proc. PKC 2012, Darmstadt, Germany, May 2012, Lect. Notes Comput. Sci., vol.7293, pp.243–261, Springer-Verlag, Heidelberg.

[25] S. Yilek, "Resettable public-key encryption: How to encrypt on a virtual machine," Proc. CT-RSA 2010, San Francisco, CA, USA, March 2010, Lect. Notes Comput. Sci., vol.5985, pp.41–56, Springer-Verlag, Heidelberg.

## Appendix A: Verifier-Policy ABID

### A.1 Scheme

`VP-ABID` consists of four PPT algorithms: (Setup, KeyGen, P, V).

**Setup**$(\lambda, \mathcal{U}) \rightarrow$ **(PK, MSK).** Setup takes as input the security parameter $\lambda$ and the attribute universe $\mathcal{U}$. It outputs a public key PK and a master secret key MSK.

**KeyGen(PK, MSK,** $S$**)** $\rightarrow$ $\mathbf{SK}_S$**.** A key-generation algorithm KeyGen takes as input the public key PK, the master secret key MSK and an attribute set $S$. It outputs a secret key $\mathrm{SK}_S$ corresponding to $S$.

**P(PK,** $\mathbf{SK}_S$**) and V(PK,** $\mathbb{A}$**).** P and V are interactive algorithms called a *prover* and a *verifier*, respectively. P takes as input the public key PK and the secret key $\mathrm{SK}_S$. Here the secret key $\mathrm{SK}_S$ is given to P by an authority that runs KeyGen(PK,MSK,$S$). V takes as input the public key PK and an attribute set $S$. P is provided V's access structure $\mathbb{A}$ by the first round. P and V interact with each other for some, at most constant rounds. Then, V finally returns its decision bit $b$. $b = 1$ means that V *accepts* P in the sense P has a secret key $\mathrm{SK}_S$ such that $S$ satisfies $\mathbb{A}$. $b = 0$ means that V *rejects* P. We demand correctness of `VP-ABID` that for any $\lambda$ and $\mathcal{U}$, and if $S \in \mathbb{A}$, then $\Pr[(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathrm{Setup}(\lambda, \mathcal{U}); \mathrm{SK}_S \leftarrow \mathrm{KeyGen}(\mathrm{PK}, \mathrm{MSK}, S); b \leftarrow \langle \mathrm{P}(\mathrm{PK}, \mathrm{SK}_S), \mathrm{V}(\mathrm{PK}, \mathbb{A})\rangle : b = 1] = 1$.

### A.2 Concurrent Man-in-the-Middle Attack on VP-ABID and Security

An adversary $\mathcal{A}$'s objective is impersonation. $\mathcal{A}$ tries to make a verifier V accept with an access structure $\mathbb{A}^*$. The following experiment $\mathbf{Exprmt}^{\mathrm{cmim}}_{\mathcal{A},\mathtt{VP\text{-}ABID}}(\lambda, \mathcal{U})$ of an adversary $\mathcal{A}$ defines the game of concurrent man-in-the-middle attack (cMiM attack, for short) on VP-ABID in the dual way to PP-ABID.

```
$\mathbf{Exprmt}^{\mathrm{cmim}}_{\mathcal{A},\mathtt{VP\text{-}ABID}}(\lambda, \mathcal{U}) : //\mathrm{Adaptive}\mathbb{A}^*$
  $(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{Setup}(\lambda, \mathcal{U})$
  $\mathbb{A}^* \leftarrow \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot),\mathbf{P}_j(\mathrm{PK},\mathrm{SK}_\cdot)|_{j=1}^{q'_\mathrm{p}}}(\mathrm{PK}, \mathcal{U})$
  $b \leftarrow \langle \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot),\mathbf{P}_j(\mathrm{PK},\mathrm{SK}_\cdot)|_{j=q'_\mathrm{p}}^{q_\mathrm{p}}}, \mathbf{V}(\mathrm{PK}, \mathbb{A}^*)\rangle$
  If $b = 1$ then Return WIN else Return LOSE
```

The *advantage* of $\mathcal{A}$ over `VP-ABID` in the game of cMiM attack is defined as

$$\mathbf{Adv}^{\mathrm{cmim}}_{\mathcal{A},\mathtt{VP\text{-}ABID}}(\lambda) \overset{\mathrm{def}}{=} \Pr[\mathbf{Exprmt}^{\mathrm{cmim}}_{\mathcal{A},\mathtt{VP\text{-}ABID}}(\lambda, \mathcal{U}) \text{ returns } \textsc{Win}].$$

`VP-ABID` is called *secure against cMiM attacks* if, for any PPT $\mathcal{A}$ and for any attribute universe $\mathcal{U}$, $\mathbf{Adv}^{\mathrm{cmim}}_{\mathcal{A},\mathtt{VP\text{-}ABID}}(\lambda)$ is negligible in $\lambda$.

### A.3 Selective Security

In the *selective game on a target access structure* (the game of sel-cMiM attack), $\mathcal{A}$ declares $\mathbb{A}^*$ *before* $\mathcal{A}$ receives PK. The following experiment $\mathbf{Exprmt}^{\mathrm{sel\text{-}cmim}}_{\mathcal{A},\mathtt{VP\text{-}ABID}}(\lambda, \mathcal{U})$ defines the selective game.

```
$\mathbf{Exprmt}^{\mathrm{sel\text{-}cmim}}_{\mathcal{A},\mathtt{VP\text{-}ABID}}(\lambda, \mathcal{U}) : //\mathrm{Seletive}\mathbb{A}^*$
  $(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{Setup}(\lambda, \mathcal{U})$
  $\mathbb{A}^* \leftarrow \mathcal{A}(\lambda, \mathcal{U})$
  $b \leftarrow \langle \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot),\mathbf{P}_j(\mathrm{PK},\mathrm{SK}_\cdot)|_{j=1}^{q_\mathrm{p}}}(\mathrm{PK}), \mathbf{V}(\mathrm{PK}, \mathbb{A}^*)\rangle$
  If $b = 1$ then Return WIN else Return LOSE
```

The *advantage* in the game of sel-cMiM attack is defined as

$$\mathbf{Adv}^{\mathrm{sel\text{-}cmim}}_{\mathcal{A},\mathtt{VP\text{-}ABID}}(\lambda) \overset{\mathrm{def}}{=} \Pr[\mathbf{Exprmt}^{\mathrm{sel\text{-}cmim}}_{\mathcal{A},\mathtt{VP\text{-}ABID}}(\lambda, \mathcal{U}) \text{ returns } \textsc{Win}].$$

`VP-ABID` is called *selectively secure against cMiM attacks* if, for any PPT $\mathcal{A}$ and for any $\mathcal{U}$, $\mathbf{Adv}^{\mathrm{sel\text{-}cmim}}_{\mathcal{A},\mathtt{VP\text{-}ABID}}(\lambda)$ is negligible in $\lambda$.

### A.4 Anonymity

Anonymity that is discussed briefly in Introduction is formalized as follows. Consider the following experiment $\mathbf{Exprmt}^{\mathrm{anonym}}_{\mathcal{A},\mathtt{VP\text{-}ABID}}(\lambda, \mathcal{U})$. (In the experiment, an adversary $\mathcal{A}$ interacts with $\mathbf{P}(\mathrm{PK}, \mathrm{SK}_{S_b})$ as a verifier with $\mathbb{A}^*$.)

```
$\mathbf{Exprmt}^{\mathrm{anonym}}_{\mathcal{A},\mathtt{VP\text{-}ABID}}(\lambda, \mathcal{U}) :$
  $(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{Setup}(\lambda, \mathcal{U}), (S_0, S_1, \mathbb{A}^*) \leftarrow \mathcal{A}(\mathrm{PK})$
        s.t. $(S_0 \in \mathbb{A}^* \wedge S_1 \in \mathbb{A}^*) \vee (S_0 \notin \mathbb{A}^* \wedge S_1 \notin \mathbb{A}^*)$
  $\mathrm{SK}_{S_0} \leftarrow \mathbf{KeyGen}(\mathrm{PK}, \mathrm{MSK}, S_0)$
  $\mathrm{SK}_{S_1} \leftarrow \mathbf{KeyGen}(\mathrm{PK}, \mathrm{MSK}, S_1)$
  $b \leftarrow \{0, 1\}, \hat{b} \leftarrow \mathcal{A}^{\mathbf{P}(\mathrm{PK},\mathrm{SK}_{S_b})}(\mathrm{PK}, \mathrm{SK}_{S_0}, \mathrm{SK}_{S_1})$
  If $b = \hat{b}$ Return WIN else Return LOSE
```

We say that `VP-ABID` have *anonymity* if, for any PPT $\mathcal{A}$ and for any $\mathcal{U}$, the following advantage of $\mathcal{A}$ is negligible in $\lambda$.

$$\mathbf{Adv}^{\mathrm{anonym}}_{\mathcal{A},\mathtt{VP\text{-}ABID}}(\lambda) \overset{\mathrm{def}}{=}$$

$$|\Pr[\mathbf{Exprmt}^{\text{anonym}}_{\mathcal{A},\texttt{VP-ABID}}(\lambda, \mathcal{U}) \text{ returns } \textsc{Win}] - 1/2|.$$

## Appendix B: Ciphertext-Policy Attribute-Based KEM

### B.1 Scheme

A ciphertext-policy ABKEM, `CP-ABKEM`, consists of four probabilistic polynomial time algorithms: (Setup, KeyGen, Encap, Decap). The definition goes in a dual manner to key-policy ABKEM on an access structure $\mathbb{A}$ and an attribute set $S$. $\mathbf{Setup}(\lambda, \mathcal{U}) \rightarrow (\mathbf{PK}, \mathbf{MSK})$. $\mathbf{KeyGen}(\mathbf{PK}, \mathbf{MSK}, S) \rightarrow \mathbf{SK}_S$. $\mathbf{Encap}(\mathbf{PK}, \mathbb{A}) \rightarrow (\kappa, \psi)$. $\mathbf{Decap}(\mathbf{PK}, \mathbf{SK}_S, \psi) \rightarrow \hat{\kappa}$. We demand correctness of `CP-ABKEM` that for any $\lambda$ and $\mathcal{U}$, and if $S \in \mathbb{A}$, then $\Pr[(\text{PK}, \text{MSK}) \leftarrow \text{Setup}(\lambda, \mathcal{U}); \text{SK}_S \leftarrow \text{KeyGen}(\text{PK}, \text{MSK}, S); (\kappa, \psi) \leftarrow \text{Encap}(\text{PK}, \mathbb{A}); \hat{\kappa} \leftarrow \text{Decap}(\text{PK}, \text{SK}_S, \psi) : \kappa = \hat{\kappa}] = 1$.

### B.2 Chosen-Ciphertext Attack on One-Wayness of CP-ABKEM and Security

The following experiment $\mathbf{Exprmt}^{\text{ow-cca}}_{\mathcal{A},\texttt{CP-ABKEM}}(\lambda, \mathcal{U})$ of an adversary $\mathcal{A}$ defines the game of chosen-ciphertext attack on one-wayness of `CP-ABKEM` (the OW-CCA game).

```
$\mathbf{Exprmt}^{\text{ow-cca}}_{\mathcal{A},\texttt{CP-ABKEM}}(\lambda, \mathcal{U})$ : //Adaptive$\mathbb{A}^*$
  $(\text{PK}, \text{MSK}) \leftarrow \mathbf{Setup}(\lambda, \mathcal{U})$
  $\mathbb{A}^* \leftarrow \mathcal{A}^{\mathcal{KG}(\text{PK},\text{MSK},\cdot),\mathcal{DEC}(\text{PK},\text{SK}_\cdot,\cdot)}(\text{PK}, \mathcal{U})$
  $(\kappa^*, \psi^*) \leftarrow \mathbf{Encap}(\text{PK}, \mathbb{A}^*)$
  $\hat{\kappa^*} \leftarrow \mathcal{A}^{\mathcal{KG}(\text{PK},\text{MSK},\cdot),\mathcal{DEC}(\text{PK},\text{SK}_\cdot,\cdot)}(\psi^*)$
  If$\hat{\kappa^*} = \kappa^*$ then Return Win else Return Lose
```

The *advantage* of $\mathcal{A}$ over `KP-ABKEM` in the OW-CCA game is defined as

$$\mathbf{Adv}^{\text{ow-cca}}_{\mathcal{A},\texttt{CP-ABKEM}}(\lambda) \overset{\text{def}}{=} \Pr[\mathbf{Exprmt}^{\text{ow-cca}}_{\mathcal{A},\texttt{CP-ABKEM}}(\lambda, \mathcal{U}) \text{ returns } \textsc{Win}].$$

`CP-ABKEM` is called *secure against chosen-ciphertext attacks on one-wayness* if, for any PPT $\mathcal{A}$ and for any $\mathcal{U}$, $\mathbf{Adv}^{\text{ow-cca}}_{\mathcal{A},\texttt{CP-ABKEM}}(\lambda)$ is negligible in $\lambda$.

### B.3 Selective Security

In the *selective game on a target access structure* (OW-sel-CCA game), $\mathcal{A}$ declares $\mathbb{A}^*$ *before* $\mathcal{A}$ receives PK. The following experiment $\mathbf{Exprmt}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{CP-ABKEM}}(\lambda, \mathcal{U})$ defines the selective game.

```
$\mathbf{Exprmt}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{CP-ABKEM}}(\lambda, \mathcal{U})$ : //Selective$\mathbb{A}^*$
  $(\text{PK}, \text{MSK}) \leftarrow \mathbf{Setup}(\lambda, \mathcal{U})$
  $\mathbb{A}^* \leftarrow \mathcal{A}(\lambda, \mathcal{U})$
  $(\kappa^*, \psi^*) \leftarrow \mathbf{Encap}(\text{PK}, \mathbb{A}^*)$
  $\hat{\kappa^*} \leftarrow \mathcal{A}^{\mathcal{KG}(\text{PK},\text{MSK},\cdot),\mathcal{DEC}(\text{PK},\text{SK}_\cdot,\cdot)}(\text{PK}, \psi^*)$
  If$\hat{\kappa^*} = \kappa^*$ then Return Win else Return Lose
```

The *advantage* in the OW-sel-CCA game is defined as

$$\mathbf{Adv}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{CP-ABKEM}}(\lambda) \overset{\text{def}}{=} \Pr[\mathbf{Exprmt}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{CP-ABKEM}}(\lambda, \mathcal{U}) \text{ returns } \textsc{Win}].$$

`CP-ABKEM` is called *selectively secure against chosen-ciphertext attacks on one-wayness* if, for any PPT $\mathcal{A}$ and for any $\mathcal{U}$, $\mathbf{Adv}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{CP-ABKEM}}(\lambda)$ is negligible in $\lambda$.

## Appendix C: Generic Conversion from CP-ABKEM to VP-ABID

### C.1 The Conversion

Let `CP-ABKEM`= (KEM.Setup, KEM.KeyGen, KEM.Encap, KEM.Decap) be a CP-ABKEM. Then `VP-ABID`= (Setup, KeyGen, Encap, Decap) is obtained as a challenge-and-response protocol of encapsulation-and-decapsulation. Figure A· 1 shows this conversion.

**Theorem 4:** If `CP-ABKEM` is OW-CCA secure, then the derived `VP-ABID` is cMiM secure. More precisely, for any given PPT adversary $\mathcal{A}$ on `VP-ABID` in the game of cMiM attack, and for any given attribute universe $\mathcal{U}$, there exists a PPT adversary $\mathcal{B}$ on `CP-ABKEM` in the OW-CCA game that satisfies the following tight reduction.

$$\mathbf{Adv}^{\text{cmim}}_{\mathcal{A},\texttt{VP-ABID}}(\lambda) \leqslant \mathbf{Adv}^{\text{ow-cca}}_{\mathcal{B},\texttt{CP-ABKEM}}(\lambda).$$

The proof of Theorem 4 goes in a dual manner to the proof of Theorem 1 on an access structure $\mathbb{A}$ and an attribute set $S$ and is omitted.

## Appendix D: Computational $q$-Parallel Bilinear Diffie-Hellman Exponent Assumption with Gap on Target Group

Let $a, s, b_1, \ldots, b_q \in \mathbb{Z}_p$, all of which is not zero, be chosen at random. Denote $e(g, g)$ as $g_\text{T}$. Let

$$\begin{aligned}\vec{y} := (&g, g^s, g^a, \ldots, g^{(a^q)}, g^{(a^{q+2})}, \ldots, g^{(a^{2q})},\\ &\forall_{1 \leqslant j \leqslant q} \; g^{sb_j}, g^{a/b_j}, \ldots, g^{(a^q/b_j)}, g^{(a^{q+2}/b_j)}, \ldots, g^{(a^{2q}/b_j)},\\ &\forall_{1 \leqslant j,\, k \leqslant q,\, k \neq j} \; g^{asb_k/b_j}, \ldots, g^{a^q sb_k/b_j}).\end{aligned}$$

Then our new assumption says it is at most with a negligible probability in $\lambda$ that, for any PPT algorithm $\mathcal{B}$ given input $\vec{y}$ (parametrized by $q$), to output $Z = g_\text{T}^{a^{q+1}s} \in \mathbb{G}_\text{T}$, even with the aid of the decisional DDH oracle $\mathcal{DDH}_{\mathbb{G}_\text{T}}(g_\text{T}, \cdot, \cdot, \cdot)$. The validity of the assumption is explained by the generic bilinear group model [6] in Appendix E.

## Appendix E: Validity of Our Assumptions in the Generic Bilinear Group Model

We explain validity of our two assumptions in Sect. 2.5 and Appendix D in the generic bilinear group model, especially

| | | | |
|---|---|---|---|
| $\mathbf{Setup}(\lambda, \mathcal{U})$: | $\mathbf{P}(\mathrm{PK}, \mathrm{SK}_S)$: | | $\mathbf{V}(\mathrm{PK}, \mathbb{A})$: |
| $(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{KEM.Setup}(\lambda, \mathcal{U})$ | | | $(\kappa, \psi) \leftarrow \mathbf{KEM.Encap}(\mathrm{PK}, \mathbb{A})$ |
| Return $(\mathrm{PK}, \mathrm{MSK})$ | Receiving $\psi$ as input: | $\xleftarrow{\psi}$ | Send $\psi$ to $\mathbf{P}$ |
| $\mathbf{KeyGen}(\mathrm{PK}, \mathrm{MSK}, S)$: | $\hat{\kappa} \leftarrow \mathbf{KEM.Decap}(\mathrm{PK}, \mathrm{SK}_S, \psi)$ | | |
| $\mathrm{SK}_S \leftarrow \mathbf{KEM.KeyGen}(\mathrm{PK}, \mathrm{MSK}, S)$ | Send $\hat{\kappa}$ to $\mathbf{V}$ | $\xrightarrow{\hat{\kappa}}$ | Receiving $\hat{\kappa}$ as input: |
| Return $(\mathrm{SK}_S)$ | | | If $\hat{\kappa} = \kappa$ then $b := 1$ else $b := 0$, Return $b$ |

**Fig. A· 1** A generic conversion from CP-ABKEM to VP-ABID.

in the light of Boneh, Boyen and Goh [6]. In Appendix A of [6], a template is given for the *decisional* BDH assumption and its variants, so we adapt the template for our *computational* BDH assumption and PBDHE assumption *with gap on target group*. According to the template, our two assumptions can be treated in a unified manner.

Let $\xi_0, \xi_1$ be two random encodings of the additive group $\mathbb{Z}_p^+$: i.e. injective maps $\xi_0, \xi_1 : \mathbb{Z}_p^+ \to \{0,1\}^m$. We write $\mathbb{G} = \{\xi_0(x); x \in \mathbb{Z}_p^+\}$ and $\mathbb{G}_\mathrm{T} = \{\xi_1(x); x \in \mathbb{Z}_p^+\}$.

**Lemma 1** (Comp.$(P, Q, f)$-DH Problem with Gap on $\mathbb{G}_\mathrm{T}$): (Notations are the same as in [6].) Let $P$, $Q \in \mathbb{Z}_p[X_1, \ldots, X_n]^s$ be two $s$-tuples of $n$-variate polynomials over $\mathbb{Z}_p$ and let $f \in \mathbb{Z}_p[X_1, \ldots, X_n]$. Let $d = \max(2d_P, d_Q, d_f)$. If $f$ is independent of $P, Q$ in the sense in [6], then for any algorithm $\mathcal{A}$ that makes a total of $q$ queries to the oracles of group operation in $\mathbb{G}$, $\mathbb{G}_\mathrm{T}$, the oracle of bilinear pairing $e$: $\mathbb{G} \times \mathbb{G} \to \mathbb{G}_\mathrm{T}$ and the oracle $\mathcal{DDH}_{\mathbb{G}_\mathrm{T}}(g_\mathrm{T}, \cdot, \cdot, \cdot)$, we have:

$$
\begin{aligned}
&|\Pr[x_1, \ldots, x_n \leftarrow \mathbb{Z}_p : \\
&\quad \mathcal{A}(p, \xi_0(P(x_1, \ldots, x_n)), \xi_1(Q(x_1, \ldots, x_n))) \\
&\qquad = \xi_1(f(x_1, \ldots, x_n))]| \leqslant (q + 2s + 1)^3 \ (d/p).
\end{aligned}
$$

Especially, if $q, s, d$ are polynomial in $\lambda$, then the success probability for $\mathcal{A}$ to solve $(P, Q, f)$-Diffie-Hellman problem is negligible in $\lambda$.

*Proof* (sketch). According to the template in [6], we construct an algorithm $\mathcal{B}$ that simulates replying to $\mathcal{A}$'s queries. For the simulation, $\mathcal{B}$ maintains two lists as in [6]:

$$L_0 = \{(p_i, \xi_{0,i}); i = 1, \ldots, \tau_0\}, L_1 = \{(q_i, \xi_{1,i}); i = 1, \ldots, \tau_1\}.$$

Here $p_i \in \mathbb{Z}_p[X_1, \ldots, X_n]$, $q_i \in \mathbb{Z}_p[X_1, \ldots, X_n, Y]$, and $\xi_{*,*}$ are strings in $\{0,1\}^m$. At step $\tau = 0$, $L_0, L_1$ are initialized as $\tau_0 = s, \tau_1 = s + 1$, $L_0 := \{(p_i, \xi_{0,i}); p_i \in P \text{ for } i = 1, \ldots, s\}$, $L_1 := \{(q_i, \xi_{1,i}); q_i \in Q \text{ for } i = 1, \ldots, s, q_{s+1} = Y\}$, where $\xi_{*,*}$ strings are chosen at random from $\{0,1\}^m$.

The first and the second types of queries can be simulated in the same way as in [6]. The third type of queries to $\mathcal{DDH}_{\mathbb{G}_\mathrm{T}}(g_\mathrm{T}, \cdot, \cdot, \cdot)$ is simulated as follows. When queried about $(\xi_{1,i}, \xi_{1,j}, \xi_{1,k})$ by $\mathcal{A}$, $\mathcal{B}$ sets $\tau_2 \leftarrow \tau_2 + 1$ and replies True if $q_i q_j = q_k$ as polynomials, and False otherwise. $\tau_0 + \tau_1 + \tau_2 = \tau + 2s + 1$ holds at each step $\tau$.

The simulation of $\mathcal{B}$ is perfect except three error-cases. The first and the second error-cases are the same as in [6]. The third error-case is the case that the following holds: putting $y = f(x_1, \ldots, x_n) \in \mathbb{Z}_p$;

$$
\begin{aligned}
&1 \leqslant {}^\exists i,\ {}^\exists j,\ {}^\exists k \leqslant \tau_1 \text{ such that} \\
&\quad i, j, k : \text{ pairwise distinct}, \\
&\wedge\ q_i(x_1, \ldots, x_n, y)\, q_j(x_1, \ldots, x_n, y) = q_k(x_1, \ldots, x_n, y), \\
&\wedge\ q_i(X_1, \ldots, X_n, Y)\, q_j(X_1, \ldots, X_n, Y) \neq q_k(X_1, \ldots, X_n, Y).
\end{aligned}
$$

Here the upper equality is an equality as elements in $\mathbb{Z}_p$, and the lower equality is an equality as polynomials in $\mathbb{Z}_p[X_1, \ldots, X_n, Y]$.

If none of the three error-cases occurs, then $\mathcal{A}$ never return the correct answer $\xi_1(f(x_1, \ldots, x_n))$ because $f(X_1, \ldots, X_n)$ never appear in the list $L_1$. This is because $f$ is independent of $(P, Q)$ as polynomials, in the sense defined in [6] (see [6] for the precise argument).

Therefore, when $\mathcal{A}$ returns the correct answer $\xi_1(f(x_1, \ldots, x_n))$, it must be in one of the three error-cases. The probability that the three error-cases occur is evaluated in a similar way as in [6], but in our case the above third error-case is dominant and we have the upper bound $(q + 2s + 1)^3(d/p)$, which is negligible in $\lambda$. □

**Corollary 1:** In the generic bilinear group model, our assumptions in Sects. 2.5 and Appendix D are valid.

*Proof.* Just put $P = (1, X_1, X_2, X_3)$, $Q = (1, 1, 1, 1)$ and $f = X_1 X_2 X_3$ for the former. By degree argument, we can see that $f$ is independent of $(P, Q)$ in the sense in [6], hence the assumption in Sect. 2.5 holds. For the latter, put $P$ as in [23] (the exponents of components of $\vec{y}$), $Q = (1, \ldots, 1)$ and $f = X_1 X_2 X_3$. By the argument in [23], $f$ is independent of $(P, Q)$ in the sense in [6], hence the assumption in Appendix D holds. □

## Appendix F: Concrete Construction: Our Enhanced Waters CP-ABKEM and VP-ABID

The construction of our concrete CP-ABKEM is described in Fig. A· 2.

**Theorem 5:** If the computational $q$-parallel bilinear Diffie-Hellman exponent assumption with gap on target group holds, and an employed hash function family has target collision resistance, then our CP-ABKEM is OW-sel-CCA secure with a challenge matrix of size $l^* \times n^*$, $l^*, n^* \leqslant q$. More precisely, for any given PPT adversary $\mathcal{A}$ on CP-ABKEM in the OW-sel-CCA game and for any given attribute universe $\mathcal{U}$, there exist a PPT adversary $\mathcal{B}$ on $(e, \mathbb{G}, \mathbb{G}_\mathrm{T})$ in the computational $q$-PBDHE with gap-for-$\mathbb{G}_\mathrm{T}$ game and a PPT target collision finder $\mathcal{CF}$ on $Hfam_\lambda$ that satisfy the following tight reduction.

$$\mathbf{Adv}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{CP-ABKEM}}(\lambda) \leqslant \mathbf{Adv}^{\text{c-pbdhe-gap}}_{\mathcal{B},(e,\mathbb{G},\mathbb{G}_\mathrm{T})}(\lambda) + \mathbf{Adv}^{\text{tcr}}_{\mathcal{CF},Hfam_\lambda}(\lambda).$$

```
**Setup**$(\lambda, \mathcal{U})$:
  $(p, \mathbb{G}, \mathbb{G}_{\mathrm{T}}, g, e) \leftarrow \mathbf{Grp}(\lambda)$
  For $x = 1$ to $u$: $T_x \leftarrow \mathbb{G}$
  $a \leftarrow \mathbb{Z}_p, A := g^a$
  $\alpha_1, \alpha_2 \leftarrow \mathbb{Z}_p$
  $Y_1 := e(g,g)^{\alpha_1}, Y_2 := e(g,g)^{\alpha_2}$
  $\eta \leftarrow HKey_\lambda$
  $\mathrm{PK} := (g, T_1, \ldots, T_u, A, Y_1, Y_2, \eta)$
  $\mathrm{MSK} := (g^{\alpha_1}, g^{\alpha_2})$
  Return $(\mathrm{PK}, \mathrm{MSK})$
```

```
**KeyGen**$(\mathrm{PK}, \mathrm{MSK}, S)$:
  For $k = 1, 2$: $l_k \leftarrow \mathbb{Z}_p$
  For $k = 1, 2$:
      $K_k := g^{\alpha_k} A^{l_k}, L_k := g^{l_k}$
      For $x \in S$: $K_{k,x} := T_x^{l_k}$
  $\mathrm{SK}_S := ((K_k, L_k,$
      $(K_{k,x};\ x \in S));\ k = 1, 2)$
  Return $\mathrm{SK}_S$
```

```
**Encap**$(\mathrm{PK}, \mathbb{A} = (M, \rho))$:
  $s \leftarrow \mathbb{Z}_p$, For $j = 2$ to $n$: $v_j \leftarrow \mathbb{Z}_p$
  $\vec{v} = (s, v_2, \ldots, v_n)$
  For $i = 1$ to $l$: $\lambda_i = \vec{v} \cdot M_i,\ r_i \leftarrow \mathbb{Z}_p$
  $C' := g^s$
  For $i = 1$ to $l$:
      $C_i := A^{\lambda_i} T_{\rho(i)}^{-r_i}, D_i := g^{r_i}$
  $\psi_{\mathrm{cpa}} := (\mathbb{A}, C', ((C_i, D_i); i = 1, \ldots, l))$
  $\tau \leftarrow H_\eta(\psi_{\mathrm{cpa}})$
  For $k = 1, 2$: $\kappa_k := Y_k^s; d := \kappa_1^\tau \kappa_2$
  $(\kappa, \psi) := (\kappa_1, (\psi_{\mathrm{cpa}}, d))$
  Return $(\kappa, \psi)$
```

```
**Decap**$(\mathrm{PK}, \mathrm{SK}_S, \psi)$:
  If $S \notin \mathbb{A}$ then Return $\hat{\kappa} := \perp$
  else $\tau \leftarrow H_\eta(\psi_{\mathrm{cpa}})$
   $\{\omega_i \in \mathbb{Z}_p; i \in \rho^{-1}(S)\}$
       $\leftarrow \mathbf{Reconst}(\rho^{-1}(S), M)$
   For $k = 1, 2$:
    $\hat{\kappa_k} := e(K_k, C')/$
    $\prod_{i \in \rho^{-1}(S)} (e(L_k, C_i) e(K_{k,\rho(i)}, D_i))^{\omega_i}$
   If $\hat{\kappa_1}^\tau \hat{\kappa_2} \neq d$ then $\hat{\kappa} := \perp$ else $\hat{\kappa} := \hat{\kappa_1}$
   Return $\hat{\kappa}$
```

**Fig. A·2** Our concrete CP-ABKEM (an enhanced Waters CP-ABKEM). ("←" from a set ($\mathbb{Z}_p$ or $HKey_\lambda$) means uniform random sampling.)

*Proof.* Using any given OW-sel-CCA adversary $\mathcal{A}$ as subroutine, we construct a solver $\mathcal{B}$ that solves an instance of the problem of computational $q$-parallel bilinear Diffie-Hellman exponent assumption with gap on target group, as follows.

**Set up.** $\mathcal{B}$ is given a random instance of the problem, $\vec{y}$, as input (see Appendix D). $\mathcal{B}$ initializes its inner state. $\mathcal{B}$ chooses an attribute universe $\mathcal{U} = \{1, \ldots, u\}$ at random, where the size $u$ is bounded by a polynomial in $\lambda$. $\mathcal{B}$ invokes $\mathcal{A}$ on input $(\lambda, \mathcal{U})$. In return, $\mathcal{B}$ receives a target access structure $\mathbb{A}^* = (M^*, \rho^*)$ from $\mathcal{A}$, where $M^*$ is of size $l^* \times n^*$. By the assumption in Theorem 5, $l^*, n^* \leqslant q$.

For each $x = 1, \ldots, u$, let $X_x := (\rho^*)^{-1}(x)$. Then $\mathcal{B}$ puts each component $T_x$ of PK as

$$z_x \leftarrow \mathbb{Z}_p,\ T_x := g^{z_x} \prod_{i \in X_x} (g^{a/b_i})^{M^*_{i,1}} (g^{a^2/b_i})^{M^*_{i,2}} \cdots (g^{a^{n^*}/b_i})^{M^*_{i,n^*}}.$$

Note that if $X_x$ is empty, we have $T_x := g^{z_x}$.

$\mathcal{B}$ chooses $\alpha' \in \mathbb{Z}_p$ at random and sets $Y_1 := e(g^a, g^{a^q}) e(g,g)^{\alpha'}$. Here we implicitly set

$$\alpha_1 := a^{q+1} + \alpha'. \tag{A·1}$$

$\mathcal{B}$ puts $\mathrm{PK}_{\mathrm{cpa}} := (g, T_1, \ldots, T_u, A := g^a, Y_1)$.

$\mathcal{B}$ chooses random values $y'_2, \ldots, y'_{n^*} \in \mathbb{Z}_p$ and $r'_1, \ldots, r'_{l^*} \in \mathbb{Z}_p$. For each $i = 1, \ldots, l^*$, let $R_i := (\rho^*)^{-1}(\rho^*(i)) \backslash \{i\}$. Then challenge ciphertext components are computed as

$$\begin{aligned}
\psi^*_{\mathrm{cpa}} := (\mathbb{A}^*,\ & C'^* := g^{s^*} = g^s, \\
& (C^*_i := T^{r'_i}_{\rho^*(i)} \Big( \prod_{j=2,\ldots,n^*} (g^a)^{M^*_{i,j} y'_j} \Big) (g^{b_i s})^{-z_{\rho^*(i)}} \\
& \qquad \times \Big( \prod_{k \in R_i} \prod_{j=1,\ldots,n^*} (g^{a^j s b_i / b_k})^{M^*_{k,j}} \Big), \\
& D^*_i := g^{-r'_i} g^{s b_i}), i = 1, \ldots, l^*).
\end{aligned}$$

Here we have set an implicit relation $s^* = s$.

Then a public key PK and a whole challenge ciphertext $\psi^*$ is computed as

$$\eta \leftarrow HKey_\lambda,\ \tau^* \leftarrow H_\eta(\psi^*_{\mathrm{cpa}}),\ \mu \leftarrow \mathbb{Z}_p,\ Y_2 := e(g,g)^\mu / Y_1^{\tau^*},$$
$$\mathrm{PK} := (\mathrm{PK}_{\mathrm{cpa}}, Y_2, \eta),\ d^* := e(C'^*, g)^\mu,\ \psi^* := (\psi^*_{\mathrm{cpa}}, d^*).$$

Here we have set an implicit relation

$$\alpha_2 = \mu - \alpha_1 \tau^*. \tag{A·2}$$

$\mathcal{B}$ inputs $(\mathrm{PK}, \psi^*)$ to $\mathcal{A}$. Note that PK determines the corresponding MSK uniquely.

**Answering $\mathcal{A}$'s Queries. (1) Key-Extraction Queries.** When $\mathcal{A}$ issues a key-extraction query for an attribute set $S$, $\mathcal{B}$ has to reply a corresponding secret key $\mathrm{SK}_S$.

First, $\mathcal{B}$ chooses a random value $r_1 \in \mathbb{Z}_p$. $\mathcal{B}$ computes a vector $\vec{w} = (w_1, \ldots, w_{n^*}) \in \mathbb{Z}_p^{n^*}$ such that $w_1 = -1$ and for all $i \in (\rho^*)^{-1}(S)$, $\vec{w} \cdot M^*_i = 0$. Note here that $S \notin \mathbb{A}^*$, so such $\vec{w}$ surely exists. Then we implicitly set $l_1$ as

$$l_1 := r_1 + w_1 a^q + w_2 a^{q-1} + \cdots + w_{n^*} a^{q-n^*+1}.$$

Then $\mathcal{B}$ computes the component $L_1$ and $K_1$ of $\mathrm{SK}_S$ as

$$L_1 := g^{l_1} = g^{r_1} \prod_{i=1,\ldots,n^*} (g^{a^{q+1-i}})^{w_i},$$
$$K_1 := g^{\alpha'} (g^a)^{r_1} \prod_{i=2,\ldots,n^*} (g^{a^{q+2-i}})^{w_i}.$$

Finally, for each $x \in S$, $\mathcal{B}$ computes $K_{1,x}$ as

$$K_{1,x} := L^{z_x} \prod_{i \in X_x} \prod_{j=1,\ldots,n^*} \Big( (g^{a^j/b_i})^{r_1} \prod_{k=1,\ldots,n^*, k \neq j} (g^{a^{q+1+j-k}/b_i})^{w_k} \Big)^{M^*_{i,j}}.$$

Note that if $X_x$ is empty, we have $K_{1,x} := L^{z_x}$.

Now $\mathcal{B}$ has to compute the index 2 components $L_2, K_2, K_{2,x}$ for all $x \in S$. To do so, $\mathcal{B}$ chooses a random value $r_{2'} \in \mathbb{Z}_p$ and computes $L_{2'}, K_{2'}, K_{2',x}, x \in S$ *just in the same way* as to the index 1. Then $\mathcal{B}$ converts them as follows.

$$K_2 := g^\mu (K_{2'})^{-\tau^*}, L_2 := (L_{2'})^{-\tau^*},\ K_{2,x} := (K_{2',x})^{-\tau^*}, x \in S.$$

Then $\mathcal{B}$ replies $\mathrm{SK}_S = ((K_k, L_k, (K_{k,x}; x \in S)); k = 1, 2)$ to $\mathcal{A}$.

**(2) Decapsulation Queries.** When $\mathcal{A}$ issues a decapsulation query for $(S, \psi = (\psi_{\mathrm{cpa}}, d))$ (where $\psi_{\mathrm{cpa}}$ is about $\mathbb{A}$), $\mathcal{B}$ has to reply the decapsulation $\hat{\kappa}$ to $\mathcal{A}$. To do so, $\mathcal{B}$ computes as follows. (Note that the oracle $\mathcal{DDH}_{\mathbb{G}_{\mathrm{T}}}$ is accessed.)

If $S \notin \mathbb{A}$ then $\hat{\kappa} := \perp$

```
$\mathbf{P}(\mathrm{PK} = (g, T_1, \dots, T_u, A, Y_1, Y_2, \eta),$
  $\mathrm{SK}_S = ((K_k = g^{\alpha_k} A^{l_k}, L_k = g^{l_k},$
  $(K_{k,x} = T_x^{l_k}; x \in \mathcal{S}); k = 1, 2))):$
                                                    $\mathbf{V}(\mathrm{PK}, \mathbb{A})$ :
                                                      $s \leftarrow \mathbb{Z}_p$, For $j = 2$ to $n$: $v_j \leftarrow \mathbb{Z}_p, \vec{v} := (s, v_2, \dots, v_n)$
                                                      For $i = 1$ to $l$: $\lambda_i := \vec{v} \cdot M_i,\ r_i \leftarrow \mathbb{Z}_p$
                                                      $C' := g^s$
                                                      For $i = 1$ to $l$: $C_i := A^{\lambda_i} T_{\rho(i)}^{-r_i}, D_i := g^{r_i}$
                                                      $\psi_{\mathrm{cpa}} := (\mathbb{A}, C', ((C_i, D_i); i = 1, \dots, l)), \tau \leftarrow H_\eta(\psi_{\mathrm{cpa}})$
                                                      For $k = 1, 2$: $\kappa_k := Y_k^s; d := \kappa_1^\tau \kappa_2, (\kappa, \psi) := (\kappa_1, (\psi_{\mathrm{cpa}}, d))$
  Receiving $\psi$ as input:          $\xleftarrow{\psi}$      Send $\psi$ to $\mathbf{P}$
  If $S \notin \mathbb{A}$ then $\hat{\kappa} := \perp$
  else $\tau \leftarrow H_\eta(\psi_{\mathrm{cpa}})$
  $\{\omega_i; i \in \rho^{-1}(S)\} \leftarrow \mathbf{Reconst}(\rho^{-1}(S), M)$
  For $k = 1, 2$ :
    $\hat{\kappa_k} := e(K_k, C') / \prod_{i \in \rho^{-1}(S)} (e(L_k, C_i) e(K_{k,\rho(i)}, D_i))^{\omega_i}$
  If $\hat{\kappa_1}^\tau \hat{\kappa_2} \neq d$ then $\hat{\kappa} := \perp$ else $\hat{\kappa} := \hat{\kappa_1}$
  Send $\hat{\kappa}$ to $\mathbf{V}$              $\xrightarrow{\hat{\kappa}}$      Receiving $\hat{\kappa}$ as input:
                                                      If $\hat{\kappa} = \kappa$ then $b := 1$ else $b := 0$, Return $b$
```

**Fig. A· 3** An interaction of our concrete VP-ABID.

```
else If $\mathbf{Verify}(\mathrm{PK}_{\mathrm{cpa}}, \psi_{\mathrm{cpa}}, S) = \textsc{False}$ then $\hat{\kappa} := \perp$
  else $\tau \leftarrow H_\eta(\psi_{\mathrm{cpa}})$
    If $\mathcal{DDH}_{\mathbb{G}_{\mathrm{T}}}(e(g,g), Y_1^\tau Y_2, e(C', g), d) = \textsc{False}$ then $\hat{\kappa} := \perp$
    else If $\tau = \tau^*$ then Abort //Call this case Abort
      else $\hat{\kappa} := (d / e(C', g)^\mu)^{1/(\tau - \tau^*)}$
```

where **Verify** is the following PPT algorithm to check consistency of $\psi_{\mathrm{cpa}}$:

```
$\mathbf{Verify}(\mathrm{PK}_{\mathrm{cpa}}, \psi_{\mathrm{cpa}}, S)$ :
  $\{\omega_i \in \mathbb{Z}_p; i \in \rho^{-1}(S)\} \leftarrow \mathbf{Reconst}(\rho^{-1}(S), M)$
  If $e(A, C') = \prod_{i \in \rho^{-1}(S)} \{e(C_i, g) e(T_{\rho(i)}, D_i)\}^{\omega_i}$
    then Return True else Return False.
```

**Guess.** When $\mathcal{A}$ returns $\mathcal{A}$'s guess $\hat{\kappa^*}$, $\mathcal{B}$ returns $Z := \hat{\kappa^*} / e(C'^*, g)^{\alpha'}$ as $\mathcal{B}$'s guess.

By the above construction, $\mathcal{B}$ can perfectly simulate the real view of $\mathcal{A}$ until the case Abort happens. To see why, we prove the following claims.

**Claim 5:** The reply $\mathrm{SK}_S$ to a key-extraction query of $\mathcal{A}$ is a perfect simulation.

*Proof.* First, the index 1 components $(K_1, L_1, (K_{1,x}; x \in S))$ are correctly distributed, as is proved in the original work of Waters [23]. By the construction, the index 2' components $(K_{2'}, L_{2'}, (K_{2',x}; x \in S))$ are distributed in the same way as the index 1 (but with independent randomness).

For the index 2 components $(K_2, L_2, (K_{2,x}; x \in S))$, note that we have implicitly set $l_2 = l_{2'}(-\tau^*)$. Using another implicit relation (A· 2) together, we get

$$K_2 = g^\mu (K_{2'})^{-\tau^*} = g^\mu (g^{\alpha_1} A^{l_{2'}})^{-\tau^*} = g^{\mu - \alpha_1 \tau^*} A^{l_2} = g^{\alpha_2} A^{l_2},$$

$$L_2 = (L_{2'})^{-\tau^*} = (g^{l_{2'}})^{-\tau^*} = g^{l_2},$$

$$K_{2,x} = (K_{2',x})^{-\tau^*} = (T_x^{l_{2'}})^{-\tau^*} = T_x^{l_2}, x \in S.$$

□

**Claim 6:** The reply $\hat{\kappa} = (d / e(C', g)^\mu)^{1/\tau - \tau^*}$ to a decapsulation query of $\mathcal{A}$ is correct.

*Proof.* It is enough to prove that if $\mathcal{DDH}_{\mathbb{G}_{\mathrm{T}}}(e(g,g), Y_1^\tau Y_2, e(C', g), d) = \textsc{True}$, then $\hat{\kappa} = Y_1^s$. This is deduced by using the implicit relations (A· 2) as follows.

$$\hat{\kappa} = ((Y_1^\tau Y_2)^s / e(C', g)^\mu)^{1/(\tau - \tau^*)} = (e(g,g)^{(\alpha_1 \tau + \alpha_2 - \mu) s})^{1/(\tau - \tau^*)}$$
$$= (e(g,g)^{\alpha_1 (\tau - \tau^*) s})^{1/(\tau - \tau^*)} = Y_1^s.$$

□

**Claim 7:** The challenge ciphertext $\psi^* = (\psi^*_{\mathrm{cpa}}, d^*)$ is correctly distributed.

*Proof.* Using the implicit relations (A· 2), a direct calculation shows

$$d^* = e(C'^*, g)^\mu = e(g,g)^{s^* (\alpha_1 \tau^* + \alpha_2)} = (Y_1^{\tau^*} Y_2)^{s^*}.$$

Hence $\psi^* = (\psi^*_{\mathrm{cpa}}, d^*)$ is legitimate and correctly distributed. □

Now we are ready to evaluate the advantage of $\mathcal{B}$ in the OW-sel-CPA game. First, we evaluate the probability that the case Abort occurs.

**Claim 8:** The probability that Abort occurs is negligible in $\lambda$. More precisely, the following equality holds.

$$\Pr[\textsc{Abort}] = \mathbf{Adv}^{\mathrm{tcr}}_{\mathcal{CF}, Hfam_\lambda}(\lambda).$$

*Proof.* To reduce to the target collision resistance of an employed hash function family $Hfam_\lambda$, we construct a PPT target collision finder $\mathcal{CF}$ by using $\mathcal{A}$ as a subroutine. The construction is done in a similar way to the construction for Claim 4.

□

By definition, $\mathcal{A}$ wins in the OW-sel-CCA game if and only if $\hat{\kappa^*}$ is correctly guessed. That is, $\hat{\kappa^*} = Y_1^s = e(g,g)^{\alpha_1 s}$. Using (A· 1) and $C'^* = g^s$,

$$Z := \hat{\kappa^*} / e(C'^*, g)^{\alpha'} = e(g,g)^{(a^{q+1} + \alpha') s} / e(C'^*, g)^{\alpha'} = e(g,g)^{a^{q+1} s}.$$

This is the definition that $\mathcal{B}$ succeeds in computing the answer for the given instance $\vec{y}$.

Therefore, the probability that $\mathcal{B}$ wins is equal to the probability that $\mathcal{A}$ wins and Abort never occurs. So we have:

$$\begin{aligned}\Pr[\mathcal{B}\text{ wins}] &= \Pr[(\mathcal{A}\text{ wins}) \wedge (\neg\textsc{Abort})]\\ &= \Pr[\mathcal{A}\text{ wins}] - \Pr[(\mathcal{A}\text{ wins}) \wedge \neg(\neg\textsc{Abort})]\\ &\geqslant \Pr[\mathcal{A}\text{ wins}] - \Pr[\textsc{Abort}].\end{aligned}$$

Substituting advantages and using the equality in Claim 8, we have:

$$\begin{aligned}\mathbf{Adv}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{CP-ABKEM}}(\lambda) &= \Pr[\mathcal{A}\text{ wins}] \leqslant \Pr[\mathcal{B}\text{ wins}] + \Pr[\textsc{Abort}]\\ &= \mathbf{Adv}^{\text{c-pbdhe-gap}}_{\mathcal{B},(e,\mathbb{G},\mathbb{G}_{\mathrm{T}})}(\lambda) + \mathbf{Adv}^{\text{tcr}}_{\mathcal{CF},\mathit{Hfam}_{\lambda}}(\lambda).\end{aligned}$$

This is what we should prove in Theorem 5.

□

Figure A·3 shows the interaction of the obtained `VP-ABID`.

**Theorem 6** (Corollary to Theorem 4 and 5): Our `PP-ABID` is selectively secure against cMiM attacks under the same assumptions. More precisely,

$$\mathbf{Adv}^{\text{sel-cmim}}_{\mathcal{A},\texttt{VP-ABID}}(\lambda) \leqslant \mathbf{Adv}^{\text{c-pbdhe-gap}}_{\mathcal{B},(e,\mathbb{G},\mathbb{G}_{\mathrm{T}})}(\lambda) + \mathbf{Adv}^{\text{tcr}}_{\mathcal{CF},\mathit{Hfam}_{\lambda}}(\lambda).$$

**Hiroaki Anada** received his B.E. and M.E. from Waseda University, and Ph.D. from Institute of Information Security (IISEC), Japan. He is with Institute of Systems, Information Technologies and Nanotechnologies (ISIT), Japan and a visitor researcher at IISEC. He is interested in interactive proofs.

**Seiko Arita** received his B.E. and M.E. from Kyoto University, and Ph.D. from Chuo University. He has been interested in prime numbers, algebraic curves and cryptographic protocols. He is with Institute of Information Security (IISEC), Japan.

**Sari Handa** received her B.E. from Meiji University, and she is with Institute of Information Security (IISEC), Japan. She is interested in lattice and multi-linear map.

**Yosuke Iwabuchi** received his B.E. from Tokyo University of Science in 2012, and he is with Institute of Information Security (IISEC), Japan.